

Automating privacy friendly information disclosure

*Steven Gevers
Bart De Decker*

Report CW 441, April 2006



Katholieke Universiteit Leuven
Department of Computer Science

Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

Automating privacy friendly information disclosure

*Steven Gevers
Bart De Decker*

Report CW441, April 2006

Department of Computer Science, K.U.Leuven

Abstract

When using electronic services, people are often asked to provide personal information. Nowadays, this information is typically provided by typing it in using a keyboard. This is, however, an unreliable and insecure way for both user and service provider. The service provider has no guarantee about the correctness of the data. Also, the user is not protected against fraud. This paper introduces a privacy agent. The privacy agent is able to automatically disclose data, certified data and properties of credentials to service providers. The privacy agent makes personal information disclosure much more secure. By using credentials, service providers are assured of the correctness of the personal information. Users are protected against theft and fraud. Privacy is a important concern in this paper. Users keep a set of privacy preferences which the privacy agent will try to match with the privacy policy of the service provider before disclosing the personal information.

Keywords : privacy, credential, policies.

CR Subject Classification : K6.5, H4.3, K4.1

Contents

1	Introduction	2
1.1	Information structures	3
1.1.1	Uncertified data	3
1.1.2	Certified data	3
1.1.3	Credentials	4
1.1.4	Comparison	4
1.2	Scenario	4
2	Basic functionality	6
2.1	Information Structure Description Language	7
2.2	Privacy policies	7
2.2.1	The basics of P3P	7
2.2.2	Extending P3P	11
2.3	Privacy preferences	18
2.3.1	The basics of APPEL	18
2.3.2	Extending APPEL	20
3	Automating information structure exchange	26
3.1	Automatic retrieval of privacy preferences	26
3.1.1	Obtaining privacy preferences from a trusted database	27
3.1.2	Retrieval of information structure descriptions	28
3.2	Privacy agent	28
3.2.1	Components	29
3.2.2	Interactions	29
4	Conclusion	32
4.1	Discussion and related work	33
4.2	Future work	34
A	Information structure descriptions	35
A.1	X.509 certificate	35
A.2	Certified data	35
A.3	Uncertified data	35

Chapter 1

Introduction

When using electronic services, people are often asked to provide personal information. This can be, for example, a password to log into a site or a user's contact data for sending advertisements. Nowadays, this information is typically provided by typing it in using a keyboard. However, this is an unreliable and very insecure way for both user and service provider. The service provider has no guarantee that the data is correct. Also, the user is not protected against fraud.

For example, if a person wants to pay an organisation using paypal [9], he has to authenticate himself to the paypal-site using his e-mail address and a password. This is of course not a secure way to handle financial information. Trojans can easily steal information that is typed in using a keyboard. Also, more and more people are a victim of phishing attacks.

Another example is identity theft. A rogue obtains data or documents belonging to a person and then passes himself off as the victim. In 2002 identity theft had become the top consumer fraud problem[28]. Billions of dollars are lost every year because of this problem.

Technologies to counter this problem already exist. Credentials such as X.509 certificates[22] and anonymous credentials[16, 18, 17, 20] can be used to show personal information in a secure way. By using credentials, users can be sure that no one else can use their personal information. Also, service providers can have trust in the correctness of the information.

Credentials start to become more common. For example in Belgium, the government issues electronic identity cards to every citizen. These identity cards contain X.509 certificates which can be used to authenticate or to sign documents in a secure way. Technologies such as anonymous credentials are also getting mature. It is to be expected that in some years users will get a variety of credentials, ranging from membership cards, payment credentials for renting dvd's, ...

A problem with credentials is their complexity. Users do not want to put time in understanding them. Furthermore, when users will obtain more and

more credentials, it will be difficult to have an overview of the credentials that can be used for a certain action.

This report provides a means to automate the disclosure of personal information in a privacy and user friendly way. When a service provider asks personal information from a user, a *privacy agent* checks a vault with personal information of the user to see whether it is possible to provide it. An important concern in this report is privacy. The privacy agent has to check the privacy preferences of the user to see if the user does indeed want to give away the information to that particular service provider. For example, a user may not want the privacy agent to automatically disclose health information to an insurance company. The privacy agent automates every aspect of information disclosure. User intervention should only be needed when absolutely necessary.

This system is very useful for service providers as well as users. By specifying a certain level of assurance, the service provider can have more trust in the correctness of the data he receives. The user is better protected against fraud. Furthermore, he does not have to manually provide personal information anymore while his privacy is protected. The system makes electronic services more privacy and user friendly while making them much more secure.

The rest of this report is structured as follows. Next section introduces the different *information structures* the privacy agent is able to handle. The following section describes a scenario which will be used throughout the rest of the text. Chapter 2 describes the basic functionality of the system. A representation language for information structures is proposed. Also two privacy languages (P3P and APPEL) are extended to incorporate the necessary features to be able to include credentials. Chapter 3 shows how information can be automatically sent to the service provider. Most aspects focus on the client side. First, a way to gather new personal information is described. Afterwards a protocol for showing this information is explained. The last chapter gives some conclusions and evaluates our approach.

1.1 Information structures

There are three ways in which a user can prove personal information to a service provider. He can provide this information as uncertified data, certified data or embedded in a credential. In this report, the term *information structure* denotes all three. Next paragraphs describe them in more detail.

1.1.1 Uncertified data

Uncertified data is data that is not certified by a trusted party. Nowadays, this is the most used type on the internet. When signing in to a service a user has to type in some information about himself. There is no guarantee

at all that this information is correct. It can also easily have been stolen, forged or made up.

1.1.2 Certified data

Certified data consists of some data together with a signature on this information from a certification authority. An example of certified data can be found in the Belgian electronic ID card (eID)[5]. The Belgian eID contains an identity file containing personal information about the owner of the card. Furthermore, the government has included a signature on this data. When a service provider receives such information, he can be sure that there is a person with those attributes. This type of information, however, can easily be abused. When a person first receives another person's identity file and the corresponding signature, he can easily pretend to be the legitimate owner.

1.1.3 Credentials

A credential is *a piece of information attesting to the integrity of certain stated facts* [4]. Examples of credentials are X.509 certificates[22], anonymous credentials [16, 18, 17, 20] and the liability-aware certificates introduced in [21].

Credentials contain certified data. The major difference with certified data is that credentials offer a means to ensure the service provider that the person sending the data is indeed the one to whom the data was issued. This is achieved by a secret that only the owner of the credential knows and with which he can prove to be the owner. In order to obtain secure services, this type of information structure should be used.

1.1.4 Comparison

When designing a service it is important to select the kinds of information structures that are going to be accepted carefully. Table 1.1 shows some important properties of information structures. Credentials are much safer than uncertified data. However, the more complex the information structure is, the more computing power is needed. Therefore, depending on the security needs of a service, the appropriate kind of accepted information structure(s) can vary.

1.2 Scenario

This section describes a scenario that will be used throughout the rest of this report. The Belgian eID contains two X.509 certificates. One of them

	Made up	Forged	Stolen
Uncertified data	√	√	√
Certified data	X	X	√
Credentials	X	X	X

Table 1.1: Comparison between information structures

can be used for authentication purposes. The purpose of the other one is to generate legally valid digital signatures.

In this scenario we suppose that persons also have a driver license credential. This driver license is an anonymous credential containing a birth date, a name and the type of vehicle the person is allowed to drive. The birth date is included using different attributes for year, month and day.

A restaurant celebrates its anniversary. An evening is organized where people can come to eat for free. Only customers older than thirty are allowed to come. Every customer has to subscribe online. When a customer subscribes, he has to prove being over thirty. This can be done by proving that the year of birth on his driver's license is below a certain threshold. An other option is to sign a nonce with his authentication certificate. The owner of the restaurant is able to deduce the age of the person from one of the attributes of the certificate.

When a customer has proved his age he receives an anonymous credential which he must show when he arrives at the restaurant.

Chapter 2

Basic functionality

The report is going to automate personal information exchange. From now on, the owner of the restaurant is referred to as service provider. The customer is the client. In order to let the customer know what personal information he has to provide, the service provider defines privacy policies. Privacy policies include also information about what is going to happen with the data, what the customer should do if there are complaints, ... By doing this the user can trust the service provider more easily. In this report P3P (The Platform for Privacy Preferences[6]) is used to define privacy policies.

Users are not willing to give every type of information to every organisation. Therefore, the client has to define privacy preferences. Users can define several restrictions in privacy preferences. For example, it is possible to define that a telephone number is not allowed to be used for marketing purposes. This report uses APPEL (A P3P Preference Exchange Language [7]) to define privacy preferences. APPEL is a privacy preferences language that can easily be matched with P3P.

There is a lot of criticism on P3P and APPEL[25, 14, 11]. We choose to extend P3P for creating privacy policies since it is the standard for privacy policies towards users. APPEL is chosen because of its resemblance to P3P and because it is relatively easy to understand.

The two languages are chosen to give a picture of how the system works. However, it should be easy to adapt other privacy languages to contain the concepts introduced in this report.

Next section introduces a description language for information structures. With the help of these descriptions a service provider can state which credentials or data he is willing to accept from a user. Next it is shown how P3P can be extended to define several ways of showing information. The same is done for privacy preferences.

By including information structures in the privacy languages, the service provider can have more trust in the correctness of personal information. Also, the privacy of the user is protected as much as possible.

2.1 Information Structure Description Language

The following sections show a description language that is able to define the different information structures in XML. The descriptions are used in privacy policies and privacy preferences to state which information structures are acceptable.

Uncertified data keeps information about the owner of the data. Certified data and credentials also include information about the certifying entity. Furthermore, they may have several properties. An information structure description can thus be divided in one part about the owner, another part about the certifier and yet another part about properties.

Figure 2.1 shows how a driver license can be described. The *name*-attribute of the <INFORMATIONSTRUCTURE>-tag is used to make references to it. *Kind* defines the kind of information structure. Currently, *anonymouscredential*, *X.509Certificate*, *uncertifieddata* and *certifieddata* are defined. A generic value *credential* can be used to denote both kinds of credentials.

The <OWNER>-part in figure 2.1 defines properties of the owner. In this case, the credential includes information about the person's age and name.

Credentials are verified through the use of other credentials. This leads to a chain of credentials that ends with a root credential (e.g. X.509 certificate chains). The first credential of a chain denotes the certifier of the credential. The <CERTIFIERCREDENTIAL>-tag points to an information structure description of this first credential. The <ROOTCREDENTIAL>-tag defines the root credential of the chain.

The last part defines the properties of the information structure. Certified data typically has a signature algorithm as property. Common properties of credentials are, for example, the validity period and revocation information.

This example shows the most important parts of the information structure description language. Appendix A contains an elaborate example of how X.509 certificates can be converted to this notation. Furthermore, it gives an example of descriptions of certified and uncertified data.

It is important that every information structure is mapped on the tags in a consistent way. That way it is possible to use the different kinds of information structures interchangeably.

```
<INFORMATIONSTRUCTURE name="driverLicense"
    kind="anonymouscredential" file="license.cred">
  <OWNER>
    <PROPERTY name="name">
      Bill Clinton
    </PROPERTY>
    <PROPERTY name="birthday">
      18
    </PROPERTY>
    <PROPERTY name="birthmonth">
      10
    </PROPERTY>
    <PROPERTY name="birthyear">
      1950
    </PROPERTY>
    <PROPERTY name="type">
      B
    </PROPERTY>
  </OWNER>

  <CERTIFIERCREDENTIAL name="municipality"/>

  <PROPERTIES>
    <PROPERTY name="notValidBefore">
      03-06-00 20:00:00
    </PROPERTY>
    <PROPERTY name="notValidAfter">
      03-06-10 20:00:00
    </PROPERTY>
  </PROPERTIES>
</INFORMATIONSTRUCTURE>
```

Figure 2.1: Description of a credential

2.2 Privacy policies

For every type of request where a service provider wants to get information from the user he will have to create a privacy policy¹. These policies make references to information structure descriptions like the ones described previously. First, an introduction to P3P is given. Next, standard P3P is extended to incorporate uncertified data, certified data and credentials.

2.2.1 The basics of P3P

P3P is able to define the information a service provider wants to collect about a user in XML. The most important aspects of P3P are described in this section. For more information about P3P we refer to [6].

If the restaurant has a web site where the user has to provide his year of birth using a form, the P3P policy in figure 2.2 could be associated with it. Next paragraphs explain the different groups of the P3P policy.

The ENTITY-group This part of the policy gives more information about the service provider. This is the place where a service provider can put his name, contact information, ...

The DISPUTES-group When a service provider does not handle information as specified in the policy, a user must be able to file a complaint. This group defines where this can be done. The group also includes which actions can be taken against the service provider. In this case a user is only able to correct the data. It can also be specified that legal action is possible or that damage will be compensated. By including trusted dispute handlers, users may be more willing to show private information.

The STATEMENT-group This is the most important group. It defines what information is collected and what is going to happen with it. It consists of the following subgroups:

- *PURPOSE* describes why the information is requested. The example uses the information only for completion and support of the activity for which data was provided. Another type of purpose which is also included in P3P is *contact* which means that the information is used for contacting the user.
- *CONSEQUENCE* contains a string telling what service a user will get. That way a user can assess the consequences if he does not provide the requested information.

¹Nowadays one privacy policy applies to a whole site. In this approach a privacy policy is associated with every request.

```

<POLICY>
<ENTITY>
  <DATA-GROUP>
    <DATA ref="#business.name">Restaurant</DATA>
    <DATA ref="#business.contact-info.postal.city">Leuven</DATA>
    <DATA ref="#business.contact-info.postal.postalcode">3000</DATA>
  </DATA-GROUP>
</ENTITY>
<DISPUTES-GROUP>
  <DISPUTES resolution-type="independent" service="www.truste.org">
    <REMEDIES>
      <correct/>
    </REMEDIES>
  </DISPUTES>
</DISPUTES-GROUP>
<STATEMENT>
  <PURPOSE>
    <current/>
  </PURPOSE>
  <CONSEQUENCE>
    Being invited to the dinner
  </CONSEQUENCE>
  <RECIPIENT>
    <ours/>
  </RECIPIENT>
  <RETENTION>
    <indefinitely/>
  </RETENTION>
  <DATA-GROUP>
    <DATA ref="#user.bdate.ymd.year"/>
  </DATA-GROUP>
</STATEMENT>
</POLICY>

```

Figure 2.2: P3P privacy policy

```
<DATA-GROUP>
  <DATA ref="#dynamic.miscdata">
    <CATEGORIES>
      <demographic/>
    </CATEGORIES>
  </DATA>
</DATA-GROUP>
```

Figure 2.3: Using P3P categories

- *RETENTION* defines how long the information is kept. The example states that the information will be kept forever. Another example of retention is *stated-purpose*. In this case the information is deleted as soon as possible.
- *RECIPIENT* describes to which parties the information is released. In the example, *ours* means that the service provider will not give the information to anyone. Other P3P-examples are *ourselves* to say that information is given to other entities with the same practices and *unknown* to say that information is given to entities whose practices are unknown to the service provider.
- *DATA-GROUP* provides a list of the data items that are collected. P3P provides some predefined data items such as year of birth in the example. It is also possible to define custom data items.
Categories can be used to make privacy policies more general. P3P defines several categories such as *financial*, *uniqueId* and *demographic*. The category *demographic* includes data about an individual's characteristics (gender, age, income, ...). Therefore, a webpage that requests the year of birth of a user could also use the data-group defined in figure 2.3.

It is possible to include different statements in one privacy policy. Different statements can have different values for every subgroup (purpose, consequence, ...) Using more than one statement in one policy can be useful, for example, if the service provider wants to give some information to third parties while keeping other data confidential.

An important aspect of P3P privacy policies is that they are *promises*. Even if a service provider promises to give no information to third parties, a user cannot have a proof of this. Therefore, it is important for service providers to achieve a certain level of trust. For example, he can get an accreditation by a trusted privacy organisation such as TrustE[10]. It is also possible to sign privacy policies to make them (more) binding.

2.2.2 Extending P3P

To include information structures in P3P the service provider first has to define which information structures he is willing to accept from users. He can do this by using the information structure description language. Then, the service provider has to make references to these descriptions in the P3P policy. He also has to define how the information has to be shown.

Defining acceptable information structures

The service provider uses the description language to create *information structure descriptions*. This is done by defining distinguishing tags for a certain information structure. An information structure matches a description if it contains at least every tag in the description. The service provider gives every information structure description a name to which he can refer in his privacy policies.

Figure 2.4 shows information structure descriptions of a driver license, the certificates on the Belgian eID, a credential of a driver license CA and a X.509 certificate of a citizen CA.

The driver license is a anonymous credential containing the attributes *name*, *birthday*, *birthmonth*, *birthyear* and *type*.

Suppose every driver license is issued by a municipality. Furthermore, the certificates of the municipality that are used to sign the driver licenses are in turn issued by a *driver license CA*. The driver license CA is certified by the Belgian government. The fact that a driver license is indirectly issued by the driver license CA is a distinguishing property of the credential. To define this indirect link, a *level*-attribute is introduced in the <CERTIFIERCREDENTIAL>-tag. The value "2" means that the credential is issued by CAs certified by the driver license CA. Applied to the information structure descriptions of the user, this means that if the <CERTIFIERCREDENTIAL>-tag of a driver license is followed two times, a credential that complies to the description of the 'DriverLicenseCA' in figure 2.4 is found.

The default value of the *level*-attribute is "1". In this case the credential should be directly issued by the denoted issuer. Figure 2.5 shows how different levels can be defined. A + or - can be added to the level attribute to cover multiple levels.

Specifying which attributes a credential must have and pointing to the issuer can be a sufficient description to cover every driver license without covering other credentials. If the CAs, certified by the Driver License CA, only issue driver licenses, it would not be necessary to include the attributes.

An important thing to note is that it is possible to point to different information structures with only one description. The description of the eIDcredential refers to both the authentication and the non-repudiation X.509

```

<INFORMATIONSTRUCTURE name="driverLicense"
                                kind="anonymouscredential">
  <OWNER>
    <PROPERTY name="name"/>
    <PROPERTY name="birthday"/>
    <PROPERTY name="birthmonth"/>
    <PROPERTY name="birthyear"/>
    <PROPERTY name="type"/>
  </OWNER>

  <CERTIFIERCREDENTIAL level="2" name="DriverLicenseCA"/>
</INFORMATIONSTRUCTURE>

<INFORMATIONSTRUCTURE name="DriverLicenseCA" kind="credential">
  <OWNER>
    <PROPERTY name="CN">Driver License CA</PROPERTY>
  </OWNER>
</INFORMATIONSTRUCTURE>

<INFORMATIONSTRUCTURE name="eIDCredential" kind="credential">
  <CERTIFIERCREDENTIAL name="CitizenCA"/>
</INFORMATIONSTRUCTURE>

<INFORMATIONSTRUCTURE name="CitizenCA" kind="X.509Certificate">
  <OWNER>
    <PROPERTY name="CN">
      Citizen CA
    </PROPERTY>
  </OWNER>
</INFORMATIONSTRUCTURE>

```

Figure 2.4: Information structure descriptions

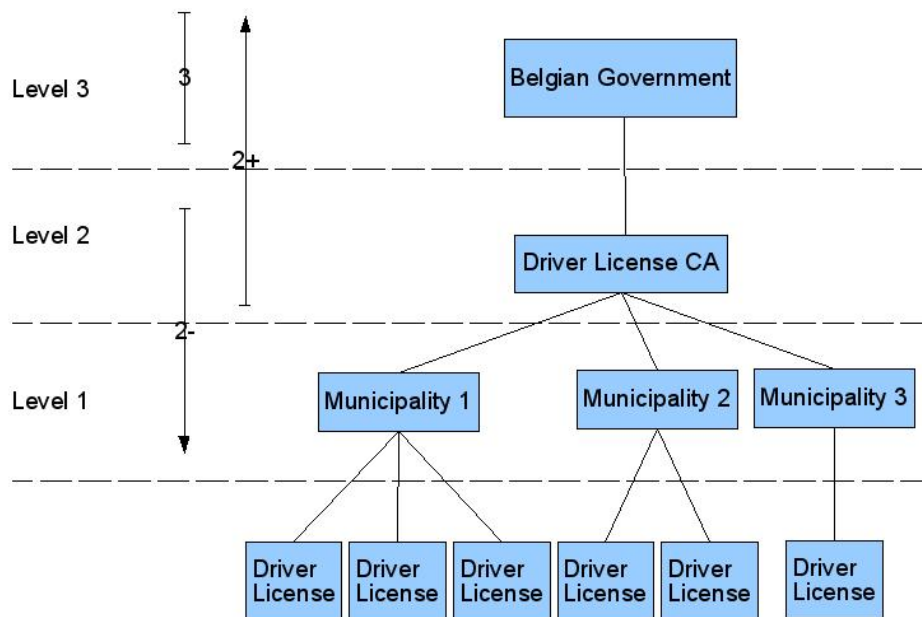


Figure 2.5: CA hierarchy of a driver license

certificates on the Belgian eID since both are issued by the Citizen CA. Furthermore, the *kind* of the information structure is 'credential', pointing at every different kind of credential. If in some years the citizen CA will also issue anonymous credentials, these credentials are already covered by this description.

Creating references to information structures

The service provider has to refer to information structure descriptions in his privacy policies. First, he must specify the location of the information structure descriptions. Next, he has to define which information structures correspond to names used in the policy. This is shown in figure 2.6.

The `<IS-DEFS>`-tag has a *url*-attribute which links to a file containing the different information structure descriptions. The descriptions are put in a separate file to allow different privacy policies to refer to the same descriptions.

The `<DATA-DEF>`-tag defines the information structures a user is able to use if the service provider refers to a certain data item. A `DATA-DEF` contains one or more references to credentials, data and certified data. By including more than one reference, the service provider gives the user more options to prove certain attributes.

In the `DATA-DEF` definitions the '@'-symbol indicates the parts of the

```
<DATA-SCHEMA>
  <IS-DEFS url="credentials.xml"/>
  <DATA-DEF name="birthyear"
    short-description="The birth year of the client">
    <INFORMATIONSTRUCTURE name="DriverLicense">
      <OWNER>
        <@PROPERTY name="birthyear"/>
      </OWNER>
    </INFORMATIONSTRUCTURE>
  </DATA-DEF>

  <DATA-DEF name="serialnumber">
    <INFORMATIONSTRUCTURE name="eIDCredential">
      <OWNER>
        <@PROPERTY name="serialnumber"/>
      </OWNER>
      <PROPERTIES>
        <PROPERTY name="usage">signing</PROPERTY>
      </PROPERTIES>
    </INFORMATIONSTRUCTURE>
  </DATA-DEF>
</DATA-SCHEMA>
```

Figure 2.6: References to information structures

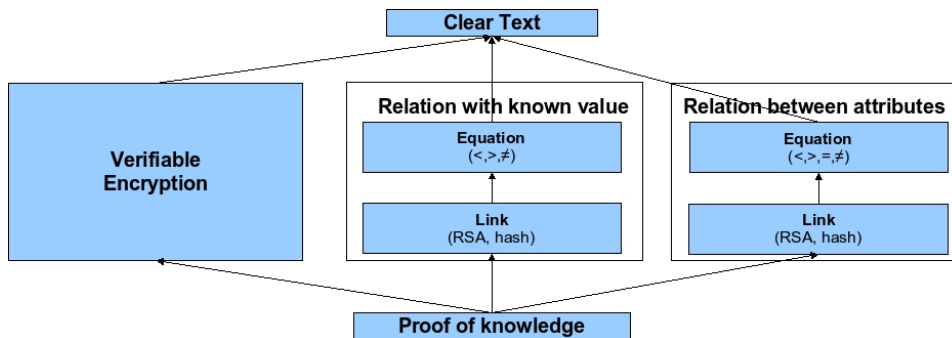


Figure 2.7: Different ways of disclosure

information structure a user has to disclose. Hence, when using anonymous credentials, the owner may hide the other attributes. The example shows that when the birthyear must be proved, only the *birthyear*-attribute of a driver license should be shown. To request the entire information structure the '@' is written in the <INFORMATIONSTRUCTURE>-tag. By including more than one information structure in a DATA-DEF, the service provider gives the user a choice to prove certain attributes.

Note that in case of X.509 certificates, a user will have to show the whole certificate even though only one attribute is asked. When the service provider asks the serialnumber of the authentication credential, the user can only show this by showing every aspect of the certificate.

It is possible to define new constraints on information structure descriptions. The example defines that an authentication credential is a credential that matches eIDcredential. Furthermore, the *usage* attribute must contain the value 'signing'².

Defining different ways of disclosure

In standard P3P a service provider can only state he needs certain information in clear text. With anonymous credentials, however, there are more possibilities. The different ways of disclosure used in this report are defined in the papers *A general certification framework with applications to privacy-enhancing certificate infrastructures* [19] and *Anonymous yet Accountable Access Control* [15]. Figure 2.7 gives an overview.

- it is possible to show/prove the value of credential attributes (i.e. *clear text*).

²On the Belgian eID card the authentication certificate contains 'signing' as usage. The certificate to make digital signatures has 'non-repudiation' as usage value.

- Users are able to *prove knowledge* of a certain attribute. This corresponds to proving ownership of a credential containing the attribute without revealing any attributes.
- A user can disclose information as a *verifiable encryption*. A verifiable encryption is associated with a condition and a third party. If the condition is fulfilled, the third party is allowed to decrypt the encrypted information. This can for example be useful to identify a person in case of abuse.
- A user can prove *relations*. It is possible to compare an attribute with a *known value* or an *attribute of another credential*. Two types of relations can be distinguished.
 - *Link*: a link between an attribute and another value defines that there is *some link*. A link typically defines one way functions such as RSA or hash functions. If f is a one way function and $f(x) = y$, the user can prove knowing x without revealing it. This type of relation leaks no information about the attribute.
 - *Equation*: less than (or equal), greater than (or equal), not equal and equal are equations. A service provider can use extra summations or products in equations. These equations can be relatively complex: e.g. $attr1 + 7 \leq attr2 \cdot (4 + attr3)$.
 Service providers frequently request the interval a certain attribute belongs to. An example can be a site which needs to know that the user's income or age is in a certain interval. These intervals are typically defined by a startpoint and a step. The user then has to provide a number k and prove $start + k.step \leq attribute \leq start + (k + 1).step$.
 In some cases it is possible to derive extra information out of a set of equations. For example, proving $age < 20$ and $age > 18$ reveals the actual age of a user. Intervals with 1 as step size also disclose the exact value of an attribute. Proving equality of an attribute to a known value is the same as showing the attribute in clear text. Therefore, when privacy policies are matched against privacy preferences, it must be made sure that service providers cannot derive more information. These cases should be treated the same as showing attributes in clear text.

The service provider must be able to define how information can be shown. Furthermore, it is useful to state that as a consequence of providing information, the user receives an information structure. To do this, the <CONSEQUENCE> and the <DATA-GROUP> part of the P3P policy are extended.

```

<CONSEQUENCE>
  <INFORMATIONSTRUCTURE name="invitation"/>
  <EXPLANATION>
    Being invited to the dinner
  </EXPLANATION>
</CONSEQUENCE>

```

Figure 2.8: Defining the consequence of information disclosure in P3P

<CONSEQUENCE> This part needs to define what information or credentials a user receives if the service succeeds. This is useful because a user may want to give certain information only if he is sure that afterwards he will get a membership credential of the organization. Figure 2.8 shows how it is possible to define that the user receives an invitation credential when he releases the requested data. This credential is not further defined in this text. As in standard P3P, it is possible to provide an explanation which can be shown to the user.

<DATA-GROUP> This is the most important part. In this part the service provider states which information he needs and in which way. In figure 2.9 the user can either prove being born before 1976 or show the serialnumber of the authentication credential.³ Since the first two numbers of the serialnumber represent the year of birth, the age can be checked with this number.

Additional tags are introduced to support the different ways of disclosure: **<VERIFIABLE_ENCRYPTION>**, **<LINK>** and **<PROOF_OF_KNOWLEDGE>**. **<GT>**, **<LT>**, **<GEQ>**, **<LEQ>**, **<EQ>**, **<NEQ>**, **<SUM>** and **<PRODUCT>** and **<INTERVAL>** can be used to define equations. Clear text disclosure of attributes can still be specified with the standard **<DATA ref="">** tag. The tags **<AND>** and **<OR>** can be used to define more complex policies such as the ones in [19] and [15]. Appendix B shows how the example used in [15] can be defined.

Figure 2.10 shows how an interval can be defined. The service provider asks whether the user is born between 1900 and 1910 years, between 1910 and 1920, ...

Our approach allows for a distinction between the personal information requested by the service provider and the technologies that are used to realise the disclosure. For instance, if a user owns a driver license, a zero knowledge proof can be used to prove his age. If he owns an X.509 certificate, he has to

³Note that in a **<DATA REF>**-tag a # is put in front of the name of the attribute. This follows the XML specification about creating anchors and how to make references to them. The # indicates that the anchor is included in the same document as the reference.

```
<DATA-GROUP>
  <OR>
    <LT>
      <DATA ref="#birthyear"/>
      <VALUE>1976</VALUE>
    </LT>
    <LT>
      <DATA ref="#serialnumber"/>
      <VALUE>76000000000</VALUE>
    </LT>
  </OR>
</DATA-GROUP>
```

Figure 2.9: Extending a data group in P3P

```
<DATA ref="#birthyear">
  <INTERVAL start="1900" step="10"/>
</DATA>
```

Figure 2.10: Defining intervals

show the certificate and prove being the legitimate owner. Note that when using a X.509 certificate, the user discloses more than only his age.

The *level of disclosure* is a partial relation between the different ways of disclosure. x has a higher level of disclosure than y if x reveals, in *every* case, more information than y . For example, showing an attribute in clear text reveals more information than proving an equation. An arrow from x to y in figure 2.7 denotes that x has a lower level of disclosure than y . The level in which the service provider wants to receive the attributes is the *minimum level of disclosure*. When a user has to disclose a property of an attribute, the service provider will of course accept it if the user gives it away in clear text. Moreover, he will also accept it if the user discloses more information than necessary (for example, by using X.509 certificates).

2.3 Privacy preferences

The service provider has specified which information he wants to receive and how the user should provide it. Now the user has to state his preferences about the ways he is willing to disclose information using privacy preferences.

If the P3P privacy policies comply with the privacy preferences the information disclosure can be performed automatically. Otherwise the transaction should be stopped. Next section gives an overview of APPEL. The

```

<appel:RULE behavior="block">
  <p3p:POLICY>
    <p3p:STATEMENT>

      <p3p:PURPOSE>
        <p3p:contact/>
      </p3p:PURPOSE>

      <p3p:DATA-GROUP appel:connective="or">
        <p3p:DATA ref="#dynamic.miscdata">
          <p3p:CATEGORIES>
            <p3p:online/>
          </p3p:CATEGORIES>
        </p3p:DATA>
        <p3p:DATA ref="#user.bdate.ymd.year"/>
        <p3p:DATA ref="#user.name"/>
      </p3p:DATA-GROUP>

    </p3p:STATEMENT>
  </p3p:POLICY>
</appel:RULE>

```

Figure 2.11: An APPEL privacy preferences rule

section thereafter describes the extensions to be made to APPEL in order to include the different information structures.

2.3.1 The basics of APPEL

APPEL privacy preferences consist of a collection of *rules*. A rule states a certain privacy preference of a user. Figure 2.11 defines a rule which states that a request will 'block' if data from the category *online* (e.g. online contact information), the year of birth or the name of the user is requested for contact purposes.

Next paragraphs describe the important aspects of APPEL.

Rule matching The privacy preferences of a user is a collection of rules defined as in the example. When privacy preferences are checked with a privacy policy, a user agent checks every rule to see if it matches the policy.

A rule matches a policy if every aspect of it is also included in the policy. For example, figure 2.11 considers two things. First, the rule states that the purpose of the policy must be *contact*. Second, it states something about the data-group. The data-group must contain information of the category

online, the year of birth *or* the name of the user. A policy that requests information about one of the items in the data-group and uses it for contact purposes matches the rule since it includes every aspect of the rule. Figure 2.2 requests the year of birth of a user. However, since it uses the information for the purpose *current* instead of *contact*, there is no match.

Behavior The *behavior* of the first rule that matches defines whether the information should be shown or not. Every rule is associated with such a behavior. APPEL defines three types of behaviors.

1. *Block*: The request is blocked. No more data is exchanged.
2. *Request*: Proceed with the request.
3. *Limited*: Proceed with the request, but only show information that is absolutely necessary.

APPEL connectives To make privacy preferences more flexible APPEL introduces some connectives to use in the rules. In figure 2.11, the *or*-connective defines that only one of the included elements should also be included in the P3P policy. The possible connectives are the following.

- *or* When some elements of the privacy preferences also appear in the privacy policy, the rule can still match.
- *and* To match with the policy, every element should be included in it.
- *non-or* For the rule to match the privacy policy, none of the elements may be included in it.
- *non-and* Not all of the elements may be found in the privacy policy.
- *or-exact* At least one element included in this connective has to be included in the privacy policy. If the policy contains elements not listed in the rule, matching fails.
- *and-exact* Matches if all contained expressions can be found in the privacy policy. The matching fails if the privacy policy contains elements which are not included in the rule. This is the default matching semantics.

2.3.2 Extending APPEL

The privacy preferences of a user must now be extended to be able to handle every type of information structure. Next sections explain how references to information structures can be made in APPEL. Then it is explained how to define whether or not information should be disclosed. Furthermore,

```

<appel:RULE behavior="block">
  <p3p:POLICY>
    <p3p:STATEMENT>
      <p3p:DATA-GROUP>
        <p3p:DATA ref="#birthyear" appel:connective="not-or">
          <p3p:LT>
            <p3p:VALUE>1980</p3p:VALUE>
          </p3p:LT>
          <p3p:VERIFIABLE_ENCRYPTION>
            <P3P:DECRYPTION-ORGANIZATION ref="www.truste.com"/>
            <P3P:DECRYPTION-CONDITION ref="abuse"/>
          </p3p:VERIFIABLE_ENCRYPTION>
        </p3p:DATA>
      </p3p:DATA-GROUP>
    </p3p:STATEMENT>
  </p3p:POLICY>
</appel:RULE>

```

Figure 2.12: Defining how an attribute can be disclosed

a way to link information structures to APPEL categories is shown. By doing this it is possible to generate privacy preferences more easily. The last section of this chapter concludes with a construct to select the appropriate combination of information structures when users have more than one option to show certain information.

Creating references to information structures

In the privacy preferences references to information structures have to be made. To do this, information structure descriptions of every information structure of the user have to be made. If an information structure has well defined semantics (e.g. X.509 certificates), it is possible to generate the tags of its description automatically. If these semantics are not specified, it is impossible to automate the generation of information structure descriptions. In this case, the entity that issued the information structure could provide the description. It is possible to refer to these descriptions the same way as in P3P.

Defining different ways of disclosure

Each APPEL-rule specifies at which level data may be released. In figure 2.12, the year of birth can be given away as a comparison to prove being born before 1980. The year of birth can also be disclosed as a verifiable

```
<p3p:DATA ref="#birthyear">
  <p3p:RELATION/>
</p3p:DATA>
```

Figure 2.13: Defining that the year of birth can be given away as a relation

```
<p3p:DATA ref="#birthyear">
  <p3p:INTERVAL step="10"/>
</p3p:DATA>
```

Figure 2.14: Defining an interval on the *birthyear*-attribute

encryption. In this case only TrustE should be able to decrypt it in case of abuse. The not-or connective defines that if the age is requested in another way, the rule will match and the request is blocked.

Figure 2.13 shows another example. Suppose this code is included in an ALLOW-rule. In this case the user is willing to show his year of birth only as a relation (or to perform a proof of knowledge). In the <RELATION>-tag, it is also possible to include a <VALUE> or a <DATA> tag to allow comparing to a known value or to another attribute.

Users are able to define an interval in which they are willing to disclose information. That way users can state that a service provider is allowed to know their year of birth within a range of 10 years. Figure 2.14 shows how this can be done. Note that a service provider can mimic intervals by using greater than and smaller than relations. In this case the intervals should of course also be respected. Also, the misuse of relations described previously must be prevented.

The level at which the user wants to show attributes is the *maximum level of disclosure*. When a user wants to give away something in clear text, he is also willing to give away a verifiable encryption of it.

Comparisons can also be made weaker. If a user is willing to prove being over eighteen, he will also be prepared to prove being older than fifteen. However, he will certainly never prove being over sixty or reveal his age in clear text.

Linking information structures with categories

Visualization It is possible to use categories in privacy preferences in order to reduce their complexity. This section explains how information structures can be linked to categories. Standard P3P already defines some categories. However, when lots of sensitive information is included in information structures, it is necessary to define more groups in order to define

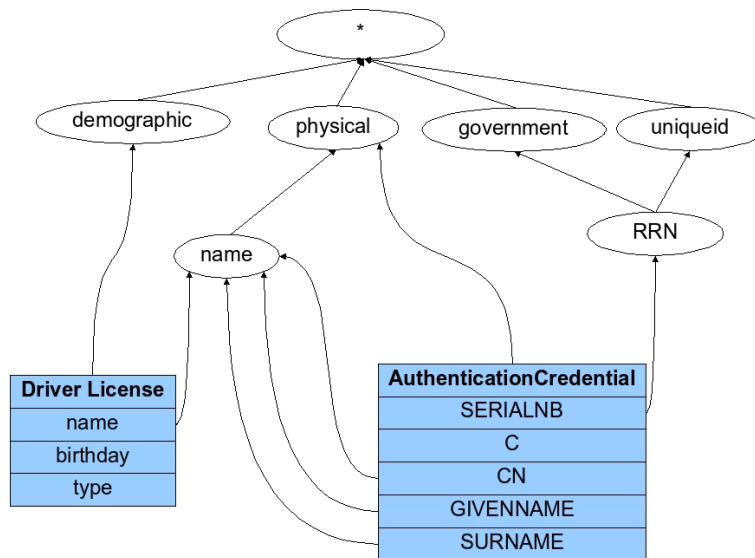


Figure 2.15: Linking information structures with categories

more fine grained privacy preferences.

Figure 2.15 shows an example visualization of a categorisation. The ellipses represent the different categories, the rectangles define information structures and the information included in them. Different aspects about the figure should be noted.

- Every information structure should point to at least one category.
- An attribute itself can link to categories. The attribute belongs to its own categories as well as those of its information structure. This is done because if a person shows an attribute of an information structure, he may also give away other information implicitly. When a user shows for example an attribute included in an information structure from a political organization, he also shows that he is part of that organization.
- The categories form a lattice. An attribute that belongs to a certain category is subject to the rules of that node as well as the rules of the ancestor(s) of that category.
- A category can have many parents. For example, in figure 2.15 the category RRN is the child of both uniqueid and government. Information in this category follows the rules of the categories *, government, uniqueid and RRN.

```
<CATEGORIZATION>
  <CHILDREN category="*">
    physical
    government
    uniqueid
    demographic
  </CHILDREN>
</CATEGORIZATION>
```

Figure 2.16: Defining the category hierarchy

```
<INFORMATIONSTRUCTURE name="DriverLicense">
  <CATEGORIES>
    <demographic/>
  </CATEGORIES>
  <OWNER>
    <PROPERTY name="name"/>
    <CATEGORIES>
      <name/>
    </CATEGORIES>
  </PROPERTY>
</OWNER>
</INFORMATIONSTRUCTURE>
```

Figure 2.17: Assigning categories to information structures

- A good approach is to use only categories when defining privacy preferences. Even when only one attribute needs special privacy rules it should receive a category of its own. An example in figure 2.15 is RRN. This approach helps to keep the creation of privacy preferences more consistent and simple. Furthermore, it is possible that different information structures share the same attribute. When new information structures are introduced these can be very easily integrated.

XML representation The visualization of the categorization can be converted to an XML representation to be easily exchanged and automatically handled. The categorization can be transformed as in figure 2.16. Every category has a children tag. These include all the direct children of the category. Figure 2.17 shows how the driver license should be changed in order to state that a driver license belongs to the category 'demographic'. The *name*-attribute also belongs to the category 'name'.

Information structure sensitivities

With the techniques described above it is possible to describe privacy preferences that apply to the different information structures. To be able to show information automatically, there is still an issue. Sometimes users can have a choice about which information structure to use. In that case, the most privacy friendly option should be selected. In the scenario a user can prove his age by showing his driver license or by showing his authentication certificate.

When showing an information structure, a user gives away information about himself. The user must try to minimize this as much as possible. The approach is based on following rules.

- Whenever a user gives away an attribute of an information structure, he loses privacy.
- The different types of information disclosure influence the amount of privacy the user loses.
- Whenever a users discloses an information structure, he also loses privacy. For example, when a user shows an attribute included in an information structure from a political organization, he also discloses being part of that organization.

Every information structure receives a sensitivity value between 0 and 100. Furthermore, the attributes of that information structure also receive such a value for every type of disclosure. When a user has a choice between more than one (combination of) credential(s) to show to a service provider, a *disclosure rate* is calculated as follows:

$$disclosure_rate = \sum_{i \in cred} sensitivity(i) + \sum_{i \in attr} sensitivity(i)$$

Cred is the set of disclosed credentials and *attr* is the set of disclosed attributes of these credentials. If a certain attribute is included in more than one credential, the value of the disclosure type that gives away most information is chosen. For example, if a user proves being over eighteen, but also gives away his birth date with another credential, only the sensitivity value of giving away a birth date in clear text is counted.

The credential combination with the lowest disclosure rate is the best combination to disclose.

A disadvantage of this simple approach is that it does not include dependencies between the different attributes. For example, a person with sensitivity values $sensitivity(name)=40$ and $sensitivity(income)=40$ may find a value 80 for disclosing them together too low.

Every combination of credentials, however, is first checked against the privacy preferences of the user. Therefore, every proposed combination is

allowed by the user. By using this heuristic the best credential combination will be chosen most of the times. There are few exceptions in which not the best, but still an acceptable alternative is selected. A method which takes into account aspects such as dependencies is unrealistic. Furthermore, it would be too time consuming to use.

Sensitivities are easily included in information structure descriptions by including *sensitivity*-attributes to the tags. By choosing a default value (for example 50) this can be made more manageable.

Chapter 3

Automating information structure exchange

With the use of the privacy policies the service provider can define what information he needs. Users have privacy preferences that can be matched to these privacy policies.

One difficulty that should be addressed is how users can get privacy preferences that represent the privacy level they want. Creating consistent privacy preferences is a very complex process. Next section describes a solution that uses privacy preferences generated by a trusted third party. Section 3.2 introduces a *privacy agent* that combines the different aspects of the report in order to be able to automatically exchange information in a privacy friendly and secure way.

3.1 Automatic retrieval of privacy preferences

Users need to have privacy preferences according to the privacy level they want. To do this automatically, [30] is used as a starting point. A semi-automated approach is used for the derivation of personal privacy policies. Users get privacy rules according to a privacy level they choose.

The system makes use of a trusted third party. This would preferably be some sort of privacy organisation. This trusted third party provides three things.

- *A database with privacy preferences.* There is a database with privacy preferences that are created as a results of surveys and studies. A user can define a personalized level of privacy through a number of privacy indicators. Based on these indicators, privacy preferences are retrieved from the trusted third party. This procedure is described in [30].
- *A database with information structure descriptions.* Users must be able to link the hierarchy of categories with the privacy preferences.

The trusted third party provides information structure descriptions to do this. These descriptions define the categories and the sensitivities of the different information structures and their attributes. This way the user can get all the information he needs.

- A file with the hierarchy of the categories.

Next sections go deeper into how a user gets suitable privacy preferences from the database and how information structure descriptions are obtained and used.

3.1.1 Obtaining privacy preferences from a trusted database

This part is taken from [30]. A trusted third party has a collection of privacy preferences. Based on some user settings privacy preferences are automatically generated. That way a user can easily obtain privacy preferences that fit his needs. More detailed, the approach is explained as follows.

A trusted third party performs some accurate surveys on a regular basis. In these surveys users are questioned about how personal information can be requested or used by service providers. The users have to give each statement a *privacy level*. After analysing the surveys it is possible to give each rule a privacy level which represents most users.

For example: people are asked in which way they want to disclose their age. Out of an analysis of the survey four rules are extracted.

-
1. *Only allow proving being older than 18 → privacy level 7-10*
 2. *Don't give away my exact age → privacy level 4-6*
 3. *Age can be given away in clear text → privacy level 1-3*
 4. *Don't give away age to service providers without privacy seal → privacy level 3-10*

The database consists of a lot of rules of this kind stated in APPEL. The user will have to extract the appropriate rules. Therefore, he specifies his own privacy level. He reports his level to the trusted third party and receives the rules belonging to his level. If, for example, a user thinks privacy is important, he may assign a privacy level of 7. When he contacts the trusted third party, he will receive rule 1 and 4. A person with privacy level 2 will only receive rule 3.

[30] provides only one privacy level for a user. However, it would be better to define a privacy level for different categories. The rules of a category are chosen by the privacy level of that category. Whenever a rule contains more than one category, the maximum privacy level of both categories is chosen. By doing this, privacy preferences will better fit the needs of the user.

3.1.2 Retrieval of information structure descriptions

When a user receives a new information structure, he needs to know to which categories it (and its attributes) belongs to and its sensitivity values.

The user has a information structure description of his information structure. Either it was automatically generated or retrieved from entity that issued the information structure. The structure of the information structure is then sent to the trusted third party. The trusted third party responds with a information structure description containing the categories and sensitivities. The user can then merge his own information with the retrieved information. The result is an information structure description containing all the necessary data.

When new types of information structures are just being issued by an organization, the trusted third party will not have included this type in his database. Therefore it is possible for the service provider to send his own categorization and descriptions to the user. When the user receives such categorization, he first contacts one or more trusted third parties to check whether there is another categorization. Of course the user should also be able to assign sensitivities and categorizations manually.

3.2 Privacy agent

This report explained how service providers can specify the information they require and how users can get privacy preferences automatically. This section describes a *privacy agent* which combines both to obtain automatic personal information disclosure in a privacy friendly way.

The purpose of the privacy agent is to keep personal information disclosure hidden from the user as much as possible. The privacy agent will store every information structure in a secure way. When the user receives an information structure, the privacy agent takes care of keeping the user's privacy preferences up to date. Furthermore, when the user wants to access a service, the privacy agent will interact with the service in order to exchange the information without any human interaction.

First the different components of the privacy agent are shown. Then the interactions between the service provider and the privacy agent are described.

3.2.1 Components

The privacy agent consist of three things.

- *Vault* The privacy agent makes use of a *vault*. The purpose of this vault is to store every information structure of a user in a secure way. The vault provides several security measures to ensure that only the owner of the credential can use it. It is comparable to, for example, a Java

keystore. For security reasons however, it could also be implemented in a smart card or an online repository. For a secure implementation of a vault we refer to future work.

- *Privacy preferences* When a user accesses a service, the privacy agent will use the privacy preferences of the user to check whether or not the service may be accessed. The privacy agent will contact a trusted third party to see if the privacy preferences are still up to date. This is done whenever a user receives a new credential or after a certain period of time.
- *Information structure descriptions* When a user receives credentials, data or certified data, their descriptions are added to a description file. The privacy agent also contacts the trusted third party to receive categories and sensitivities.

3.2.2 Interactions

This section describes the actions the privacy agent takes when he receives a new information structure or when a service is accessed.

Receiving an information structure

The initial vault is completely empty. Whenever the privacy agent receives an information structure, the following things will happen:

1. The information structure is stored securely in the vault.
2. A description of the received information is added to the information structure descriptions.
3. The privacy agent contacts the trusted third party to receive the category and sensitivity information.
4. This information is merged with the information structure descriptions.
5. The privacy agent contacts the trusted third party to receive an update of the privacy preferences of the user.

Accessing a service

Figure 3.1 shows how the provicy agent interacts with a service provider. The figure is explained as follows:

1. The user application requests the service.

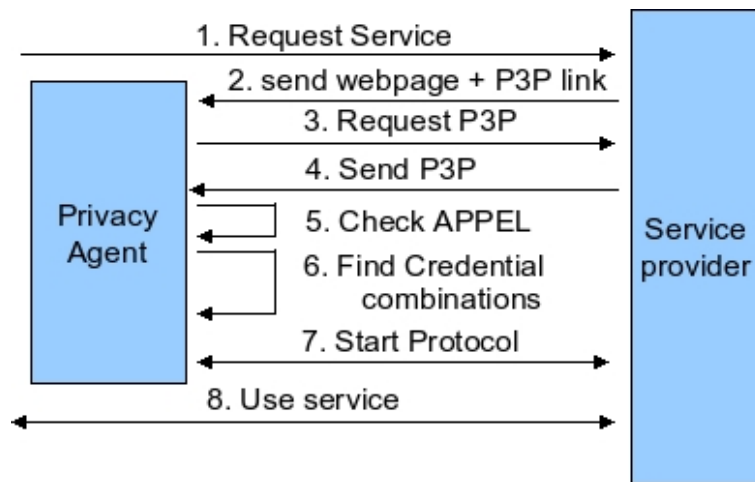


Figure 3.1: Description of the system

2. The service provider sends an webpage containing a (hidden) link to a privacy policy. This page (e.g. a login page) can be used by users without a privacy agent. Therefore, it is not necessary that every user has installed the privacy agent.
3. The privacy agent requests the privacy policy.
4. The service provider sends it to him.
5. The privacy agent checks whether the policy matches the privacy preferences of the user.
6. The privacy agent selects the necessary credentials that comply with the policy. This action comprises more than just selecting information structures that contain the requested attributes. For example, when using X.509 certificates, all attributes are revealed which may be more than necessary. The privacy preferences must be checked again to see whether it is allowed to show every attribute included. If there are different possibilities to show the information, the combination with the lowest disclosure rate will be used.
7. A protocol is started to disclose the information. First, the privacy agent informs the service how the information will be shown. When this information is exchanged, the correct protocol can be started. The system makes it necessary for the service provider to support the appropriate protocols for the different information structures. To deal with this, a protocol compiler as mentioned in [19] can be used.
8. The user application can make use of the service.

The interaction shows only the necessary steps. It can for example be extended with policy negotiations[29]. This, however, is beyond the scope of this report.

Chapter 4

Conclusion

The report proposed a privacy agent that is able to disclose information in a privacy friendly way. The service provider uses privacy policies to specify the information he needs to know. Users have privacy preferences that define what information they are willing to disclose. Based on this information the disclosure of information is automated. The use of the privacy agent makes exchanging information easier than the way it happens now. Furthermore, the privacy of the user is protected. By being able to use technologies such as credentials, a more secure information disclosure is guaranteed. Users can be sure that no one else can use their information. Also, service providers can have trust in the correctness of data.

This report defines a way to use three types of *information structures*: *uncertified data*, *certified data* and *credentials*. P3P and APPEL are extended with *information structure descriptions* to include them. Furthermore, different *types of disclosure* of credentials (verifiable encryption, ...) can be defined. This gives both languages the ability to handle more complex situations.

When complex technologies will be used to prove (certified) information, users need technologies to help them. The *privacy agent* is able to check whether he is allowed to show the information asked for. With the extension of P3P and APPEL, the privacy agent is able to handle the three types of information structures and the different types of disclosure. Services can be made more accessible by allowing users to show their information in different ways. Furthermore, when the different complex properties of anonymous credentials are not a requirement for a certain application, it may be better to use other types because these may require less computational effort.

To make the system more workable a way is proposed to *extend the categorisation* of P3P. Sometimes a user can prove information in more than one way. By making use of *sensitivities* a *disclosure rate* is calculated to find the most privacy friendly way. For users to automatically receive privacy preferences the report makes use of [30].

The *privacy agent* is able to *automate the information disclosure* of users without violating their privacy. *Interactions* are defined which automate both the *retrieval* and the *showing* of information.

4.1 Discussion and related work

This section discusses the *flexibility*, *extensibility* and the *transparency* of the privacy agent. Furthermore, the privacy agent is compared to some existing work.

The system is *flexible* in several ways. If users want a more sophisticated way for computing the disclosure rate of a set of credentials, other formulas as the one described in this paper can easily be introduced. Also, it is possible for the privacy agent to support other privacy languages such as, for example, Rei[24] and XPref[14]. To our knowledge, however, no privacy policy language is able to include the different kinds of information structures and to use the different ways of disclosure.

It is also possible to *extend* the system to make it more useable. New kinds of credentials can be easily included in the system by mapping them on the information structure description language. Also, the system can be extended to support trust negotiation. Instead of requesting the privacy policy of a service provider, the trust-target graph procedure discussed in [27] can be used. The extensions of the privacy languages proposed in this paper can be used to support the communication between the different parties.

The system is supposed to be as *transparent* as possible towards the user. Almost every aspect of personal information disclosure is automated. User intervention is only required when absolutely necessary. This way, complex technologies (e.g. anonymous credentials) can be used without even understanding their basics.

Our privacy agent has several advantages over existing user agents such as AT&T's privacy bird [1], JRC P3P Proxy [13], Netscape Browser [2] and Microsoft Internet Explorer [3]. These use P3P (some also use APPEL) to warn users when a site does not respect their preferences. Our privacy agent is able to use the different information structures and to define different ways of disclosure. Hence, service providers can allow users to disclose personal information in several ways. This makes services *more accessible*. Also, services are more *privacy friendly* since the user can choose the information structures that disclose the least personal information. By making use of credentials, services can be made *more secure*.

An advantage of our approach is that it can *easily* be *adopted* by existing systems. They can keep their old way of personal information disclosure (the webpage) while introducing more secure means. An example could be PayPal[9]. Today, users have to log in by providing their email address and a password. PayPal can generate automatic login systems that asks

the privacy agent to provide either this information (to achieve weak authentication) or to show a credential (to achieve strong authentication). By changing their service this way, new users can choose to use a much more secure way of authentication while existing users are still able to use the system. If a user wants to access a service, the system responds with a webpage containing a link to the privacy policy. Therefore, even users without a privacy agent can still use the service.

[19] and [15] propose a specification language that allows one to specify what data to release and how to release it. Their specifications can be converted to the XML notation proposed in this paper. Our work extends the functionality by allowing the user to use different kinds of information structures. Our work also uses privacy preferences to check whether the personal information disclosure is allowed.

4.2 Future work

Different aspects should be improved in order to be able to work in a secure and user friendly way with the different information structures. An important aspect is a *credential vault* that can be used to securely store credentials. Different aspects have to be taken into account, such as, for example, back up possibilities.

Also, the system can be made more flexible. By including *policy negotiation*, policies can be made that are close to the user's needs. *Dispute handling* could help users to have more trust in applications, which in turn can result in a higher use of a certain service.

Appendix A

Information structure descriptions

A.1 X.509 certificate

This section explains how X.509 certificates can be converted to the information structure descriptions proposed in the report. X.509 certificates version 3 are defined in RFC 2459[22]. This RFC uses ASN.1 to define the content of a X.509 certificate. Figure A.1 shows the most important part.

Using this ASN.1 notation, it is possible to create a structured overview of X.509 certificates. Figure A.2 shows the content of the authentication credential of a Belgian eID.

It is now possible to convert figure A.2 to an information structure description (figure A.3). The different values of the certificates should always be mapped on the same tags. The different properties of the citizen CA can be found in the information structure description of its X.509 certificate.

A.2 Certified data

Figure A.4 shows the information structure description of the identity file on the Belgian eID. The properties of the owner are included in the file. Note that it is not possible to generate this description automatically for every different kind of file. Service providers should provide the description in order to be able to use them.

A.3 Uncertified data

Figure 2.3.2 shows an information structure description of a username password combination.

```

Certificate ::= SEQUENCE {
    tbsCertificate      TBSCertificate,
    signatureAlgorithm  AlgorithmIdentifier,
    signatureValue      BIT STRING }

TBSCertificate ::= SEQUENCE {
    version             [0] EXPLICIT Version DEFAULT v1,
    serialNumber        CertificateSerialNumber,
    signature            AlgorithmIdentifier,
    issuer              Name,
    validity            Validity,
    subject             Name,
    subjectPublicKeyInfo SubjectPublicKeyInfo,
    issuerUniqueID     [1] IMPLICIT UniqueIdentifier OPTIONAL,
                      -- If present, version shall be v2 or v3
    subjectUniqueID    [2] IMPLICIT UniqueIdentifier OPTIONAL,
                      -- If present, version shall be v2 or v3
    extensions         [3] EXPLICIT Extensions OPTIONAL
                      -- If present, version shall be v3 }

```

Figure A.1: ASN.1 syntax of a X.509 v3 certificate

-
- Certificate
 - version: Version 3
 - serialNumber: 10:00:00:00:00 ...
 - signature: PKCS 1 SHA-1 With RSA Encryption
 - issuer
 - * type: SERIALNUMBER, value: 200501
 - * type: CN, value: Citizen CA
 - * type: C, value: BE
 - validity
 - * notBefore: 03-06-05 10:21:43
 - * notAfter: 03-09-10 10:21:43
 - subject
 - * type: SERIALNUMBER, value: 82101840232
 - * type: CN, value: Steven Gevers (Authentication)
 - * type: C, value: BE
 - * type: GIVENNAME, value: Steven
 - * type: SURNAME, value: Gevers
 - subjectPublicKeyInfo
 - * algorithm: PKCS #1 RSA Encryption
 - * subjectPublicKey: 30 81 89 02 ...
 - extensions
 - * keyUsage: Critical Signing
 - * CRLDistributionPoints: URL=http://crl.eid.belgium.be/eidc200501.crl
 - signatureAlgorithm: PKCS #1 SHA-1 With RSA Encryption
 - signatureValue: 31 94 15 74 62 d4 ...
-

Figure A.2: Structured overview of the content of an authentication certificate

```

<INFORMATIONSTRUCTURE name="authenticationcredential" kind="X.509certificate">
  <OWNER>
    <PROPERTY name="SERIALNUMBER">82101840232</PROPERTY>
    <PROPERTY name="CN">
      Steven Gevers (Authentication)
    </PROPERTY>
    <PROPERTY name="C">BE</PROPERTY>
    <PROPERTY name="GIVENNAME">Steven</PROPERTY>
    <PROPERTY name="SURNAME">Gevers</PROPERTY>
  </OWNER>

  <CERTIFIERCREDENTIAL name="CitizenCA"/>

  <PROPERTIES>
    <PROPERTY name="version">Version 3</PROPERTY>
    <PROPERTY name="serialNumber">10:00:00:00:00 ...</PROPERTY>
    <PROPERTY name="notValidBefore">03-06-05 10:21:43</PROPERTY>
    <PROPERTY name="notValidAfter">03-09-10 10:21:43</PROPERTY>
    <PROPERTY name="public-key-algorithm">
      PKCS #1 RSA Encryption
    </PROPERTY>
    <PROPERTY name="public-key">30 81 89 02 ...</PROPERTY>
    <PROPERTY name="usage">
      Critical
      Signing
    </PROPERTY>
    <PROPERTY name="CRLDistributionPoints">
      URL=http://crl.eid.belgium.be/eidc200501.crl
    </PROPERTY>
    <PROPERTY name="signatureAlgorithm">
      PKCS #1 SHA-1 With RSA Encryption
    </PROPERTY>
    <PROPERTY name="signatureValue">31 94 15 74 62 d4 ...</PROPERTY>
  </PROPERTIES>
</INFORMATIONSTRUCTURE>

```

Figure A.3: Information structure description of a X.509 certificate

```

<INFORMATIONSTRUCTURE name="eIDSignedData" kind="certifieddata"
                                file="ID#RN.bin">
  <OWNER>
    <PROPERTY name="Lastname">
      Gevers
    </PROPERTY>
    <PROPERTY name="FirstName">
      Steven
    </PROPERTY>
    <PROPERTY name="BirthDate">
      18 OKT 1982
    </PROPERTY>
    <PROPERTY name="Sex">
      M
    </PROPERTY>
  </OWNER>

  <CERTIFIERCREDENTIAL name="nationalRegister"/>

  <PROPERTIES>
    <PROPERTY name="signatureAlgorithm">
      SHA-1, RSA
    </PROPERTY>
    <PROPERTY name="signatureValue">
      00 15 1F ...
    </PROPERTY>
  </PROPERTIES>
</INFORMATIONSTRUCTURE>

```

Figure A.4: Information structure description of certified data

```

<INFORMATIONSTRUCTURE name="username_password">
  <OWNER>
    <PROPERTY name="username">
      user1
    </PROPERTY>
    <PROPERTY name="password">
      password1
    </PROPERTY>
  </OWNER>
</INFORMATIONSTRUCTURE>

```

Figure A.5: Information structure description of uncertified data

Appendix B

Advanced privacy policy

This appendix shows how the example in [15] can be defined in P3P. Since this example only states what information the service provider needs, only the <DATA-GROUP> part is worked out.

In the example, a user has to prove to a service provider that either his income is higher than 2000 or that the subsidies he receives together with his income is higher than 2000. [15] uses following formula:

$$\begin{aligned} & (Income.amount \geq 2000 \wedge e_{(1,1)} = Passport holder \wedge \\ & Income.amount = Passport holder) \\ & \vee (Income.amount + Subsidy.amount \geq 2000 \wedge \\ & Income holder = Subsidy holder \wedge \\ & e_{(1,1)} = Passport holder \wedge Income holder = Passport holder) \end{aligned}$$

This formula can be converted to the XML-code in figure B.

```

<DATA-GROUP>
  <OR>
    <AND>
      <GT>
        <DATA ref="Income.amount"/>
        <VALUE>2000</VALUE>
      </GT>
      <DATA ref="Passport.holder">
        <VERIFIABLE-ENCRYPTION>
          <ENCRYPTION-KEY ref="PublicKeyDeanonymizerAmerica"/>
          <DECRYPTION-ORGANIZATION ref="www.truste.com"/>
          <DECRYPTION-CONDITION ref="abuse"/>
        </VERIFIABLE-ENCRYPTION>
      </DATA>
      <EQUALS>
        <DATA ref="Income.holder"/>
        <DATA ref="Passport.holder"/>
      </EQUALS>
    </AND>

    <AND>
      <GT>
        <SUM>
          <DATA ref="Income.amount"/>
          <DATA ref="Subsidy.amount"/>
        </SUM>
        <VALUE>2000</VALUE>
      </GT>
      <EQ>
        <DATA ref="Income.holder"/>
        <DATA ref="Subsidy.holder"/>
      </EQ>
      <DATA ref="Passport.holder">
        <VERIFIABLE-ENCRYPTION>
          <ENCRYPTION-KEY ref="PublicKeyDeanonymizerAmerica"/>
          <DECRYPTION-ORGANIZATION ref="www.truste.com"/>
          <DECRYPTION-CONDITION ref="abuse"/>
        </VERIFIABLE-ENCRYPTION>
      </DATA>
      <EQ>
        <DATA ref="Income.holder"/>
        <DATA ref="Passport.holder"/>
      </EQ>
    </AND>
  </OR>
</DATA-GROUP>

```

Figure B.1: An advanced privacy policy

Bibliography

- [1] Privacy bird. <http://www.privacybird.com/>.
- [2] Netscape browser. <http://browser.netscape.com/>.
- [3] Microsoft internet explorer. <http://www.microsoft.com/windows/ie/preview/default.asp>.
- [4] Adapid glossary.
- [5] The belgian eid. <http://eid.belgium.be/>.
- [6] The platform for privacy preferences project (p3p). <http://www.w3.org/P3P/>.
- [7] A p3p preference exchange language 1.0 (appell.0). <http://www.w3.org/TR/P3P-preferences/>.
- [8] The enterprise privacy authorization language (epal 1.1). <http://www.zurich.ibm.com/security/enterprise-privacy/epal/>.
- [9] Paypall. <http://www.paypal.com/>.
- [10] Truste. <http://www.truste.org/>.
- [11] A critique of p3p : Privacy on the web. <http://dollar.ecom.cmu.edu/p3pcritique/>.
- [12] Oasis extensible access control markup language (xacml) tc. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml.
- [13] Jrc p3p resource centre. <http://p3p.jrc.it/>.
- [14] Rakesh Agrawal, Jerry Kiernan, Ramakrishnan Srikant, and Yirong Xu. An xpath-based preference language for p3p. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 629–639, New York, NY, USA, 2003. ACM Press.
- [15] Michael Backes, Jan Camenisch, and Dieter Sommer. Anonymous yet accountable access control. Technical report, IBM Research, Zurich Research Laboratory, 2006.

- [16] Stefan A. Brands. *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy*. MIT Press, Cambridge, MA, USA, 2000.
- [17] J. Camenisch and E. Van Herreweghen. Design and implementation of the idemix anonymous credential system, 2002.
- [18] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *EUROCRYPT '01: Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques*, pages 93–118, London, UK, 2001. Springer-Verlag.
- [19] Jan Camenisch, Dieter Sommer, and Roger Zimmermann. A general certification framework with applications to privacy-enhancing certificate infrastructures. Technical report, IBM Research, Zurich Research Laboratory, 2006.
- [20] David Chaum. Security without identification: transaction systems to make big brother obsolete. *Commun. ACM*, 28(10):1030–1044, 1985.
- [21] Elsie Van Herreweghen. *Unidentifiability and accountability in electronic transactions*. PhD thesis, K.U.Leuven, 2004.
- [22] R. Housley, W. Ford, W. Polk, and D. Solo. RFC 2459: Internet X.509 Public Key Infrastructure Certificate and CRL Profile. IETF RFC Publication, January 1999.
- [23] Keith Irwin and Ting Yu. Determining user privacy preferences by asking the right questions: an automated approach. In *WPES '05: Proceedings of the 2005 ACM workshop on Privacy in the electronic society*, pages 47–50, New York, NY, USA, 2005. ACM Press.
- [24] Lalana Kagal, Tim Finin, and Anupam Joshi. A Policy Based Approach to Security for the Semantic Web. In *2nd International Semantic Web Conference (ISWC2003)*, September 2003.
- [25] Pranam Kolari, Li Ding, Lalana Kagal, Shashidhara Ganjugunte, Anupam Joshi, and Tim Finin. Enhancing P3P Framework through Policies and Trust, September 2004.
- [26] Jiangtao Li and Ninghui Li. Oacerts: Oblivious attribute certificates. In *ACNS*, pages 301–317, 2005.
- [27] Jiangtao Li, Ninghui Li, and William H. Winsborough. Automated trust negotiation using cryptographic credentials. In *CCS '05: Proceedings of the 12th ACM conference on Computer and communications security*, pages 46–57, New York, NY, USA, 2005. ACM Press.

- [28] Neil Mitchison, Marc Wilikens, Robin Urry, Lothar Breitenbach, and Silvia Portesi. Identity theft - a discussion paper. *Technical EUR report, EUR 21098 EN*, 2004.
- [29] G. Yee and L. Korba. The negotiation of privacy policies in distance education. In *IRMA '03: Proceedings of the 4th International IRMA Conference*, 2003.
- [30] G. Yee and L. Korba. Semi-automated derivation of personal privacy policies. In *IRMA '04: Proceedings of the 2004 Information Resources Management Association International Conference*, 2004.