

Personal Information Management and Distribution

Kathy Bohrer, Xuan Liu, Dogan Kesdogan, Edith Schonberg, Moninder Singh, Susan L. Spraragen
IBM Thomas J. Watson Research Center, P.O. Box 704, Yorktown Heights, NY 10598
E-Mail: {bohrer, xuan, kesdogan, ediths, moninder, sprara}@us.ibm.com

Abstract

With the increasing importance of the Internet as a channel for conducting business, protecting consumer privacy has become a critical challenge. Privacy protection deals with methods giving the data owner control of his data so that he can decide when and to what extent he wants to disclose his information, and then enforcing the user's privacy preferences, either through technology, business practice or laws.

This paper presents a software component architecture that enables individuals and companies to manage, use, and distribute personal data according to privacy preferences of individuals. We call these components myPrivacy. In an enterprise privacy architecture solution, the myPrivacy components can be used by enterprises to set, enforce, and monitor access to customer information according to privacy policies agreed to by the customer. The components can also support contacting the customer for consent, and allowing the customer access to their data and policies.

1. Introduction

Most privacy-focused research has as its goal releasing no information about the person. The main reason for this is the fact that if you give away your personal data in the digital world then this information is lost. Even though this was also true in the "good old" days, where the information was printed on paper, in the digital world copying and distributing digital data is effortless. With inexpensive storage media, huge amount of data can be collected and automatically analyzed in seconds. The more information you give the more privacy is lost. This scenario may explain the restrictive privacy policy (i.e. to give no information) of some privacy professionals, when they design their systems.

In our modern world, there is no chance to live outside the digital world, that is, to have no personal digital data stored somewhere at some company or institution. Therefore, the above starting point for considering privacy is somehow artificial. From this point, the question here is not "to give or not to give" personal data. The question is how to deal with the given data and how to control the personal data (new and old) in the future. Furthermore, this is no longer just the problem of individuals but also the problem of business and society.

We do not want to either analyze deeply the overall privacy problem of our society in this paper nor do we have the unique solution for it. But, some developments in the privacy area and the industry encourage us to suggest the software myPrivacy. Before presenting myPrivacy, we want to sketch these developments:

1. Not long ago privacy was considered an annoying job that handicapped businesses. Times are changing. Now, no company can afford to have bad press about bad privacy practices. Quite the contrary is true. Some companies have even discovered that good privacy practices are good for sales promotions (e.g. Earthlink).
2. The privacy policies of companies are no longer just written in small print somewhere, invisible to the user. A continual effort is taking place to express the privacy policies of companies in a formal way so that the customer can check it automatically.
3. People enjoy and want to benefit from the Internet, e.g. the ability to shop or just search for the best offer without traveling. Flexibility in shopping is contrary to the total and old restrictive privacy policy demands. To enable business, appropriate information has to be transferred and the terms have to be negotiable.
4. Privacy matters to everyone. It is no longer only the "business" of some privacy specialist. The non-specialist audience has to balance between their privacy needs and the trust they want to invest.

To address these evolving requirements and to make it possible for parties to adequately express, control, and negotiate their privacy demands, software techniques are needed. The myPrivacy software component architecture described in this paper has this goal. It provides a flexible structure, so that it can support a strict privacy policy if people want to manage everything themselves but also give the user the flexibility to choose to

trust third parties and enterprises, and to transfer this privacy task to them. Between these two constellations, there are of course many intermediate constellations. We do not know yet which of these will be the favorite. But, for the purpose of democracy we provide an infrastructure and give people the room to search for it.

2. MyPrivacy Concepts

In a business application, there are always several parties involved. These parties are connected by a network interacting online or offline with each other. Each of the parties may wish to act in a specific role (e.g. customer, provider), where each role is associated with a specific profile. Users also may play different roles at different times. To facilitate the exchange of personal information while enforcing individual privacy policies, myPrivacy provides the following capabilities:

1. **Profile management** for different users
2. **Abstraction and negotiation** between the systems of users to overcome the problems of differing configurations
3. **Authorization**

2.1 Profile Management

The myPrivacy profile data model, which is derived from CPEXchange [CPEX], includes the concept of a role. Accordingly, the base class for modelling profile data is a *RolePlayer*, which is either a Person, an Organization, or a *PartyRole*. A *PartyRole* is always associated with a real Party (Person or Organization). Thus, a myPrivacy user can define a variety of roles for him/herself, with different profile information associated with each role. A *PartyRole* profile shares properties of its associated Person or Organization, and it can redefine properties, mask properties, and define new properties.

For a typical user, say Joe, examples of possible roles include: Joe at home, Joe at work, and Joe disguised by an alias. These roles may share the name, address, contact points, etc, which are part of Joe's basic Person profile. The Joe-at-work profile may include different contact points, such as his e-mail address and phone number at work. Joe's alias may include false data that Joe wants to give to annoying marketers.

A *PartyRole* provides a natural unit for granting access. For example, Joe may grant access to his at-home profile to his doctors and relatives, his at-work profile to his business associates, and his alias to retailers.

2.1.1 Users

The users of myPrivacy are either profile owners or requesters. The profile owners use the service primarily to manage and control their profile information. Requesters call myPrivacy components to obtain profile data. Requesters may be either registered or anonymous. A registered requester will also have a profile stored by the myPrivacy components. Since registration enables requester authentication, a registered requester may be granted authorization to data that is not available to unregistered requesters.

2.1.2 Data Subjects

Each object in the profile repository belongs to a particular *DataSubject*, where a *DataSubject* is simply a *PartyRole*. A *DataSubject* may own objects for one or more *RolePlayers*, providing a very flexible framework. Thus, it is possible for a profile, represented by a *DataSubject*, to consist of information for a single person, a person with multiple roles, an organization, or even multiple people.

2.1.3 Userids and Groupids

A *userid* is the identifier of an authenticated user. Each authenticated user is associated with a *DataSubject*. A username and password will be associated with a *userid* when a user registers. A *userid* will be associated with a user session after a user logs in with a valid username and password.

A user group is a collection of registered users. Each user group has a *groupid*. There is an association between a *groupid* and the set of *userids* in the group.

An *authorizedPartyId* is either a *userid* or a *groupid*. It is used for specifying authorization of a user or group to profile data.

2.2 Abstraction and Negotiation

In the myPrivacy data model, a *class* defines an object type (such as telephone number); a *property* defines a simple data type (such as area code); and an instance refers to a specific object (such as Joe's telephone number). MyPrivacy provides the capability to group classes, properties, and objects into abstract sets, called *views*. The purpose of a view is to enable an individual or business to define appropriate data abstractions which have meaning according to a specific privacy regulation, policy, or preference. For example, all personal financial data or medical data may be grouped into a financial view and a medical view, which are to be revealed to different requesters, based on different policies.

Since it is possible for views to consist of only classes and properties, a view may be generally defined independent of specific profile objects. However, a view may also contain individual objects, and be quite specific.

2.2.1 Privacy View Levels

Privacy is fundamentally about protecting identity. Even with privacy controls, once a piece of information is released to another party, there can be no guarantee that the information is protected from being used or accessed in unauthorized ways. Therefore, a goal of a privacy service is to release as little information as possible, and to provide extra protection for information which identifies an individual. In the context of privacy, Murphy's Law translates into "Trust Noone."

Using abstraction, we can classify profile data into different privacy view levels:

- **View Level 1:** Information that can be used to precisely identify the individual. This information includes name, address, telephone number, e-mail address, social security number, etc.
- **View Level 2:** Information that does not identify the individual, but is still quite personal. The release of this information alone does not violate privacy, provided View Level 1 information is not also released with it. For example, age, salary, and marital status are personal data. However, if these items are never associated with an individual, then releasing this information does not in itself violate privacy. However, once this information is released, there is no 100% guarantee that it will never be coupled with View Level 1 information.
- **View Level 3:** Information derived from identity data, but it is fuzzier and less informative. For example, someone may be more willing to release a salary or age range than an exact age or salary. Derived data need not be actually stored. However, we can still provide the capability of defining authorization policies for classes of derivation rules.
- **View Level 4:** Don't care information. For example, preferences, may have very little personal/privacy value to the owner.

2.2.2 Negotiation

View levels allow users to assign different privacy values to different groups of profile data, so that a privacy agent can better protect the identity of the owner. For example, a myPrivacy agent could keep track of which data has been released to different requesters, and enforce a rule that does not allow View Level 1 and View Level 2 data to be released to certain requesters. Levels are also potentially useful for negotiation. An example of a negotiation is:

1. A business requests View Level 1 data (name, address, etc) and View Level 2 data (salary, assets). The business sends a privacy policy indicating that the data will be used to approve credit.
2. MyPrivacy denies this request, but offers to send the business View Level 3 data (an alias profile, plus a salary and asset range).
3. The business will accept this View Level 3 data, but changes its privacy policy indicating that the data will be used to analyze whether credit is possible, but not to approve the credit, as originally requested.
4. MyPrivacy sends the business the View Level 3 data for analysis purposes only.

2.2.3 Aggregation

P3P [17] allows a requester to specify that data will never be used to track an individual, but used only for analysis. Often, market researchers will perform analysis by aggregating individual pieces of information. However, in the spirit of “Trust No one”, one of the services that a myPrivacy may provide is to perform analysis itself, and return aggregated results to a requester. Thus, a requester may make a request for aggregated information, protecting the individual privacy of profile owners.

2.3 Authorizations

Authorization is the process of granting profile data access to a data requester. The myPrivacy components maintain authorization rules associated with profile data, which are specified by either data subjects or businesses.

An *authorization rule* consists of five subelements: a data subject list, an authorization data view, a privacy preference rule, an access list, and an authorization action. By expressing an authorization rule, a data subject defines a mapping from the first four subelements to a result action specified by the authorization action. In other words, an authorization rule declares that for a specified authorization data view, the specified privacy preference rule is applied for the specified access list to determine an authorization action. We described each subelement in more detail below.

The **data subject list** of a rule identifies which profile(s) in the profile and policy databases the rule applies. A data subject list consists of one or more data subjects.

The **authorization data view** in a rule contains the data items that can be released according to the rule. Each data view can be one or more classes, properties, or instances.

As described in section 2.1, a data subject can categorize his/her personal data into multiple view levels (layers) so that the data in each view level have the same privacy preference, access and authorization constraints, whereas data in different view levels have different constraints. Views can be specified in a hierarchy, so that rules specified for a given view level are inherited by view levels lower in the hierarchy. By allowing the data subject to arrange all data into layers and associate privacy/access rules with each layer, we enable automated data exchange with considerable privacy protection for the data subject. All critical, highly personal data may be placed in one layer with very stringent privacy and access controls, while least sensitive data may be placed in a layer with the least amount of controls, with other data occupying intermediate layers. This, combined with the ability to associate rules with specific classes, instances, and properties of data gives the data subject a wide degree of flexibility in specifying different privacy and access controls for different categories of data, different types of requests, different classes of requesters, etc.

The **access list** in a rule declares who can access the specified data view, provided the privacy preference also matches. Each access list contains one or more authorized parties, which can be a user, a group, a token, or a special value “all”. A user refers to a registered data requester, who has a userid. A group contains multiple users, where one user may belong to multiple groups. A data subject can define one authorization rule for one or more groups of users.

A *token* provides a way to allow controlled access to the data by limiting, for example, the number of times the data may be accessed. Any data requester in possession of the token is allowed access to the data as long as the token is valid. Moreover, a token may also be issued to a data requester to enable access to data that is held by a third party. The token is then presented by the data requester to the third-party which uses it to identify the requester as well as the data to which the requester is authorized. The token mechanism can be used to protect the identity of the data subject.

If the authorized party has the special value “all”, then access is granted to all requesters of the data for whom the privacy preference rules match.

The **authorization action** declares which one of the actions should be taken if this rule is matched. The action “grant” means that the request should be granted if this rule is matched, the action “notify” specifies that data subject should be notified when the request is granted, and the action “get consent” specifies that special consent should be obtained before the request can be granted.

The **privacy preference rule** in an authorization rule specifies the privacy preference and action control that the data users in the access list have to satisfy in order to access the data. The privacy preference specifies why and how the data can be used by the requester in terms of the P3P standards [P3P]. Its purpose is to specify how the data may be used by a requester once it has been released. A privacy preference includes: Purpose, Retention, and Access.

- Purpose specifies the purposes for which the data can be acquired and processed, for example, current, administrative, pseudo-analysis, historical, telemarketing, etc.
- Retention refers to the kind of retention policy applied to the data. According to P3P, there are 5 possible sub-elements for retention: no-retention, stated-purpose, legal-requirement, business-practices, and indefinitely.
- Recipient specifies the legal entity, or domain, beyond the requester where the data may be distributed. There are 6 possible sub-elements: ours, same (delivery services), other recipient (legal services following our practice), legal services following different practices, unrelated, and public.
- Access in the privacy preference describes whether the data in the corresponding authorization data set should be accessible by the data subject after the data has been released, that is, access declares the access capability of the data subject to his/her own data.

The *action control* in the privacy preference rule specifies whether an access action is allowed. It contains several action permissions: create, update, delete, and read. In addition, since profile data may be stored by third parties and not locally by myProfile, we provide a “3rd-party read” permission, which authorizes the release of 3rd party profile data to the requester.

3. Technical Approach

The components of myPrivacy and their relationships are shown in Figure 1. These components are described below.

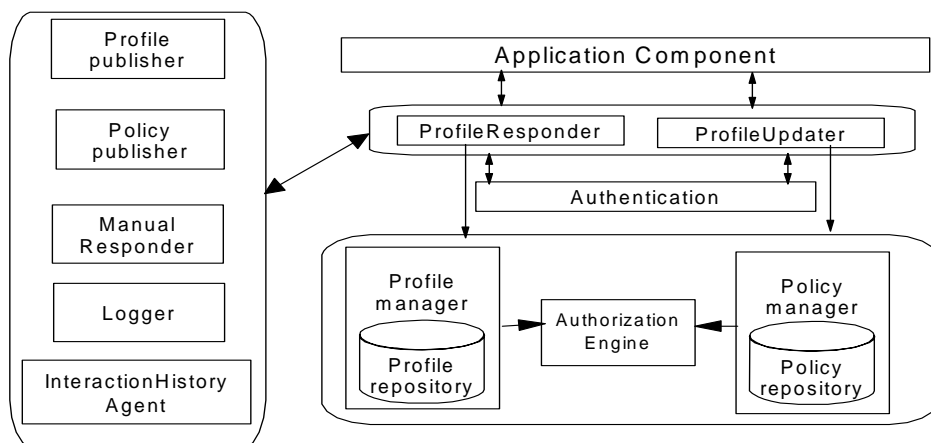


Figure 1: MyPrivacy Component Architecture

Profile Manager: maintains basic profile information and provides programming interfaces for creating, querying, modifying and deleting profile information

Policy Manager: maintains authorization rules and other policy information and provides programming interfaces for creating, querying, modifying and deleting authorization rules.

Policy Authorization Engine: performs automatic authorization for release of requested data based on authorization rules including information about privacy policy information associated with data, templates and request types, as well as users authorized to access the data. Privacy policies can be different for different groups of requesters and may also be specified at a data class, data instance, template or request level. Definitions of

privacy policies can also include policies on profile data held by third-parties. As such, it also handles requests for authorization for release of personal data held by a third-party, just as if the data was held directly by the current system. The Policy authorization engine also supports the data subject by providing the data requester with an explicit one time authorization for a particular request type as an alternative to privacy policy matching against data and requester class/group.

The Profile Responder: receives requests for profile information along with privacy statements on how the data will be used. It uses the Authentication engine to authenticate the requester and uses the Policy authorization engine to check the authorization and privacy policies. It may need to check return from the Policy Authorization engine to see if external authorization is needed on any of the requested data, and handle requests to those external authorization sources. Similarly, it must recognize the need for manual authorization and invoke the Manual Profile Responder. It uses the Profile and Policy Managers to get only the subset of information that is allowed by the policy checks and uses the Logger component to record the request and the response. It also checks data returned by the query on the Policy Authorization engine to see if some data needs to be obtained from an external source, and handles requests to those external sources. It also verifies and filters the response to ensure that no unauthorized data is returned. This step may not be necessary if queries for the exact allowed and requested data are used and supported. However, it may not be possible to trust external sources to return only the requested data since the query might return a larger set of data than the policies allow or was requested. Finally, it also supports requests for only authorization of release of data, and uses the Policy Authorization engine to return information on the data that is authorized for release under the requested policy.

Manual Authorization Engine: provides the ability to perform manual authorization of a portion of the requested data. Invoked by the Profile Responder (or possibly the Policy Engine) to initiate a manual authorization, it sends an e-mail or otherwise notifies the data provider of the need for manual authorization. The data subject can use various computer programs, such as e-mail software, to respond to such a request.

Profile Updater: receives requests to create, delete, or modify profile information. Like the Profile Responder, this component must authenticate the requester, log the request and response, check for authorization to make the profile update, and update the profile information.

Interaction History Agent: provides support for user specification of what actions should be intercepted. This could be stored in the form of rules or explicit action types. An intermediary or proxy can then be used to intercept the desired online actions and record them as additional profile data.

Profile Publisher: and the **Privacy Policy Publisher** provide the user interfaces for specifying authorization information, privacy policies, profile data and templates. They can use the Profile and Policy Managers for reading and writing the information, or could go more indirectly through the Profile Updater and Profile Responder components.

The Manual Profile Responder: provides the user interfaces for gathering missing data in order to fulfill a profile data request. Invoked by the Profile Responder (or possibly the Profile Manager) to initiate entry of missing requested profile data. Sends an e-mail or otherwise notifies data provider for missing data entry. E-mail provides an easy way to launch served user interface to specify the needed data.

The Logger: supports logging of requests to and responses by the system, and uses the Profile Manager to make any updates to the profile.

The data subject sets up data profiles and privacy policies using the Profile Publisher and the Privacy Policy Publisher. These would be stored in the Profile and Policy database, respectively. A data requester sends a request for data describing the data desired along with privacy policies describing the intended usage. The request is handled by the Profile Responder which uses the Authentication engine to verify the identity of the requester and the Policy Authorization engine to check the authority of the requester to access the requested data by using authorization rules and privacy policies, possibly invoking the Manual Authorization engine (to get manual authorizations). It then gathers the authorized data (possibly invoking the Manual Profile Responder to get missing data from the data subject), logs the request and reply using the Logger, updates the profile with this information using the Interaction History Agent, and send a reply along with the data to the data requester.

4. Architecture: Object-Oriented Design of the Components

In this section, we describe in more detail the data model, XML message structure, and algorithms for the MyPrivacy architecture components. We focus on the core components of the architecture: the Profile Responder, the Policy Authorization Engine, the Profile Manager, and the Policy Manager. The interaction diagram of these components is shown in Figure 2.

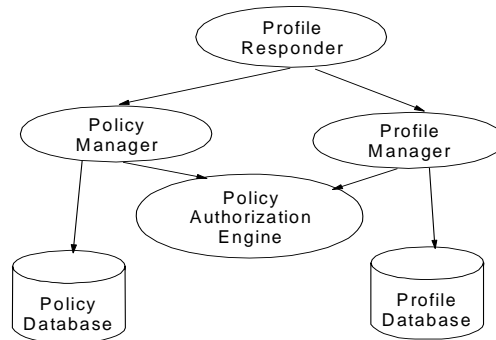


Figure 2: Component Interaction Diagram

All requests for profile-related data are processed by the Profile Responder. Both profile data and policies, which are associated with individual profiles, are protected by privacy rules and subject to authorization checking. Both the Policy Manager and Profile Manager in our architecture implement a Privacy-Enabled Resource Manager interface. The Privacy-Enabled Resource Manager (PERM) interface authorizes each data request by calling the Policy Authorization Engine. The Policy Authorization Engine uses the Authorization Rules to perform the authorization. Unauthorized requests are not fulfilled. The Data Responder examines each data request to determine whether the request is for Policy or Profile data, and calls the corresponding PERM (Policy Manager or Profile Manager). While Figure 2 shows a configuration with two Privacy-Enabled Managers, this architecture scales to support an arbitrary number of PERMS. The components may be collocated on the same machine, or distributed across multiple machines. In section 4.1, we describe the relationship between the data model and the message structure and section 4.2 outlines the algorithm for processing a request for profile data.

4.1 Data Model and Message Structure

The MyPrivacy data model and message structure are based on two XML standards: CPExchange [16] for describing personal profile data, and P3P [17], for describing privacy policies. UML was used to specify the data model, from which the XML message structure and database contents were derived. The advantages of using an XML standard for message exchange are:

- A common standard message format facilitates exchange within and across enterprises.
- A common data model can be mapped onto different, existing enterprise customer data models

Party Name Descriptive Information

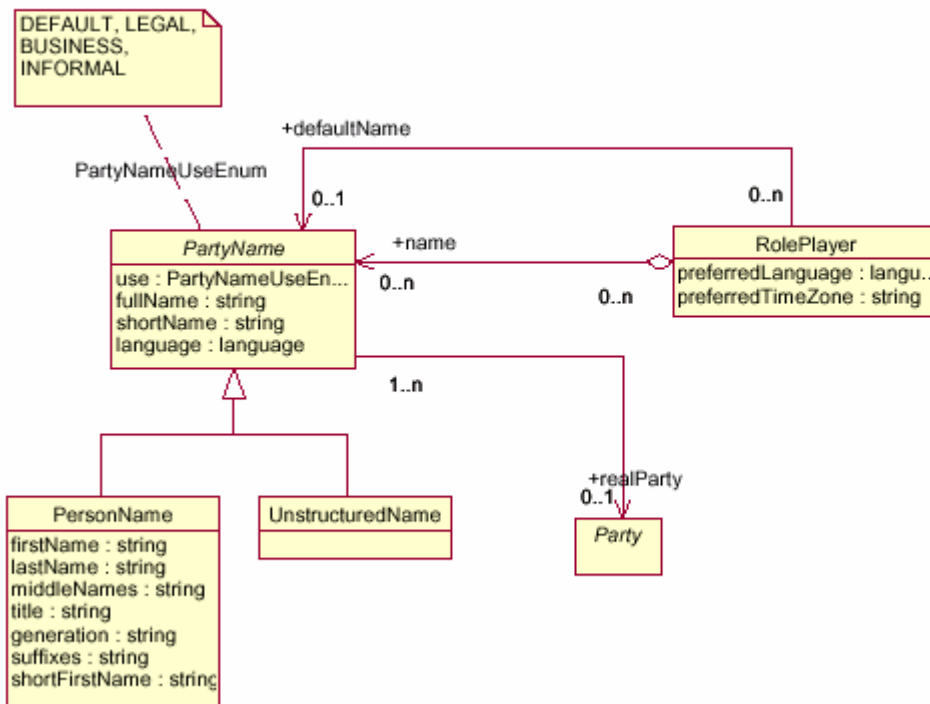


Figure 3: UML Diagram from Profile Data Model

To illustrate how the XML messages are defined from the data model, we show a sample UML definition from the profile in Figure 3. In Figure 3, a RolePlayer is a person, role, or an organization that has profile information associated with it. A PartyName has two subtypes: PersonName or UnstructuredName. There is a many-to-many relationship between RolePlayer and PartyName. The properties of each of these types are shown in their respective boxes.

To request data for a particular RolePlayer, we specify both a query-by-example form and a QueryFilter, using the XML which is derived from the UML data model. For example, if we wish to request the RolePlayer properties and PersonName properties for the person whose name is John Q. Public, then the request would look like:

Query Filter:

```

<AND>
  <PersonName>
    <firstName> John </firstName>
    <middleName> Q. </middleName>
    <lastName> Public </lastName>
  </PersonName>
</AND>

```

Query-by-example Form:

```

<instances>
  <PersonName/>
  <RolePlayer/>
</instances>

```

The profile manager translates such XML queries into an SQL query against the Profile Repository internal data model. The profile manager uses an internal form of the UML data model and a mapping table, which maps the UML data model onto an internal relational schema, to perform the translation.

A *request* for data consists of the following components:

- The privacy policy of the requester.
- Identification of the DataSubject
- A query-by-example “form”

The requester must be able to uniquely specify the DataSubject to the myPrivacy components. Thus, the identification must be some sort of public name, public key, or other encrypted token, which does not allow a requester to impersonate the owner. In the request shown above, the individual is identified by name in the QueryFilter. The QueryFilter may specify any data which uniquely identifies the individual, which the requester has access to. For example, the requester may reference the individual by address, social security number, or e-mail address. MyPrivacy will ensure that the requester is authorized to know this information before granting the request.

Alternatively, a profile owner may obtain a token from myPrivacy and pass this token to a requester. The token, which is a one-time authorization to obtain a subset of profile data, preserves the anonymity of the profile owner. Token authorization data is maintained by the Policy Component. Tokens must be encrypted, so that requesters cannot forge them.

Predefined authorization rules are convenient for granting long-term and repeated access to data. For example, Joe may wish to grant long-term authorization to his wife, his lawyer, and his accountant to some of his profile data. However, frequently profile owners will want to grant only one-time authorization to data. For example, for transactions, such as mortgage application, shopping, etc., long-term authorization to data is not desirable. Tokens serve this function.

4.2 Request Processing Algorithm

In this section, we describe the steps for processing a request for profile data, similar to the request in section 4.1.

1. The Profile Responder receives the data request.
2. The Authentication engine authenticates the requester. Authentication, which is the process of guaranteeing that the requester is who they say they are, can be carried out in several ways, including the use of a userid and password, or a cookie, etc. If the authentication is not successful, then the response is returned with an empty response item list.
3. The next step is to identify the data subject of the request. The data subject is identified from the QueryFilter parameter in the request. The Profile Responder calls a PERM both to retrieve the data subject and to check that the requester is authorized to retrieve the data subject using the QueryFilter. This step determines whose profile is being queried. Note that authorization of the QueryFilter itself is important. For example, a requester may ask for data for a person whose social security number is 331-39-5432. However, if the owner of this social security number has not authorized the requester to see his social security number, then the request must fail. This prevents an attacker who is flooding the system with random queries from accidentally obtaining unauthorized information.
4. The Profile Responder determines that the data requested belongs to the Profile Manager, and the request is then passed to the Profile Manager.
5. The profile manager calls the Policy Authorization Engine to authorize the request.
6. The Policy Authorization Engine retrieves all authorization rules of the data subject specified in the request from the policy database. If there are no authorization rules for the data subject, the Profile Responder returns a response with an empty response item list.
7. Otherwise, the next step is to examine the access list of each of the retrieved authorization rules and discard those rules which do not pertain to the requester. After this process, if there are no authorization rules left, then the Profile Responder returns an empty response. For each access list, if the requester is not found in the access list, then the authorization is discarded. The requester is in the access list if the requester is either a user in the list or a user in a group which is in the list, or if the access list explicitly authorizes “all” requesters.
8. For the remaining authorization rules, the policy authorization engine next compares the privacy declarations in the request with the privacy preference rules in the authorization rules for each profile data item in the request. (See the policy-preference matching process described in section 4.3.) For each application of the matching process for a profile data item, the result is deny or authorize. If the result

is deny, then the data item must not be included in the list of data items returned in the response . If the result is authorized, then the data item may be included in the response item list. Additionally, the authorization rule may require the data subject to be notified or the consent of the data subject to be obtained. Each data item in the request is annotated with the authorization result.

9. When the entire request list has been processed, the annotated request is returned to the profile manager, which retrieves the requested, authorized data. If all the data is available locally on the system, the profile manager collects it, and returns it to the profile responder. If, however, all data is not available locally, then it collects whatever data is locally available, and returns this data to the profile responder.
10. The profile responder tries to collect missing the data as follows:
 - The Profile Responder contacts the data subject to retrieves missing data items from him/her .
 - The Profile Responder determines if any of the missing data is available from third parties, and sends a request for the data to these third parties.
11. Finally, the Profile Responder filters the data again by matching the data returned by the third parties with the request. This ensures that only that data is returned that is both requested and allowed, that is, that the third parties are not providing any additional information.
12. The Profile Responder then sends the aggregation of all authorized, collected data to the requester.

With this gather and filter process, a data requester can obtain data held not only by the system locally but also held by third parties, either via collection by the system itself or directly from third parties after authorization by the system. Since the data subject provides authorization rules describing the privacy preferences and access permissions for all data that is held locally or by third parties, the process ensures that only that data is released, or authorized to be released, for which the data requester is authorized and for which the requester's privacy policies match that of the data subject.

If authorization is granted to part of a request, that is, to some but not all data items, then this algorithm returns partial information to the requester. Alternatively, if any of the data items are not authorized, an argument can be made to deny all information. Often, data is provided to a requester in order to complete a transaction, and partial information often causes the transaction to fail. In such cases, it is better not to reveal personal information unnecessarily. Thus, denying a request whenever any data item is denied enhances the privacy protection for the data subject, without compromising service.

4.3 Policy-Preference Matching Algorithm

This section describes the matching process between the privacy policy in a data request and a privacy preference associated with an authorization rule, defined by a DataSubject. Since both the privacy preference rule and the privacy policy are based on the P3P standard, the matching process works by matching each element of the policy, namely, the purpose, retention, recipient, and access. We extend the P3P elements by adding an action control element, which restricts the access method to the data.

PURPOSE: Both the privacy policy as well as the privacy preference rule purposes may contain one or more subelements. If the subelements declared in the privacy policy is a subset of the subelements specified in data subject's privacy preference rule, the purposes match.

RETENTION: The retention element can be one of the five subelements: **no-retention**, **stated-purpose**, **legal-requirement**, **business-practices** and **indefinitely**. These subelements express different privacy levels, some of them are more restricted than others. The subelement **no-retention** has the most restrictive privacy level, and the subelement **indefinitely** is the least restrictive. **Stated-purpose** is more restricted than **legal-requirement** and **business-practices**, both of which are more restrictive than **indefinitely**. There is no ordering between **legal-requirement** and **business-practices**. The retention matching algorithm respects this partial ordering. If the retention element in the privacy policy is as restrictive or more restrictive then the policy preference retention element, then the retentions match.

RECIPIENT: The recipient element can be one of six subelements: **ours**, **delivery**, **same**, **other-recipient**, **unrelated**, and **public**. These subelements express different privacy levels with **ours** being the most restrictive, followed by **same** and **delivery**, then **other-recipient** and **unrelated**, with **public** being the least restrictive. The matching process is successful if the recipient declared in the privacy policy is not less restricted than the recipient specified in the privacy preference rule.

ACCESS: The access element specifies whether the data subject can have access to his data after the data requester acquires the data. The access checking is successful if the access requirement of the data subject

matches exactly the access statement declared in the policy. Finally, action matching checks if the actions requested by the data requester is allowed according to the action control specified in data subject's privacy preference rule. This is done by simple checking the corresponding Boolean tag of a specific action in the preference. If all four element matches are successful, and the action matching, then the two policies match, otherwise any failure in the matching process results in authorization denial.

5. Prototype Description

IBM T.J. Watson Research Center is building a prototype of the "myPrivacy" components described in this paper to store an individual's privacy policies. In general, these policies can be centralized or distributed. Similarly, the personal data controlled by the policies can be centralized or distributed. For this particular prototype, individuals put their policies and a small amount of data in a trusted third party that acts on behalf of the individual (called the Trusted Privacy Agent, or TPA). The bulk of the individual's personal data continues to reside in various enterprises. However, the individual also puts the policies for the use of this distributed data in the TPA. For example, a TPA could have all an individual's policies for all their personal data, including data held by organizations with whom they have a relationship like their employer, doctor, and bank. During any online transaction, when the individual is asked to provide personal data the individual instead provides a reference to a service url of their TPA with some identifying information that is passed to the TPA to identify the individual. The online transaction then makes a request to the TPA with the provided information and specifying what data is being requested and the privacy policies that will be applied to that data. The TPA authorizes the release of the data. If the TPA holds the authorized data locally it is returned. If the data is remote to the TPA, the TPA can either fetch that data from the external sources or just return the information needed for the requester to fetch the data from the external sources. This might require the TPA to sign the request that will go to the external sources, so that the external data providers know it is permissible to release the requested data.

The prototype currently under development in IBM Research is based on a scenario in which a user, Joe, is applying for a mortgage, and the mortgage company obtains the personal data it needs from Joe's TPA. Joe's TPA holds Joe's privacy policies and some of Joe's data. The TPA aggregates the rest of Joe's data from external sources, in particular, Joe's Bank and Joe's Employer. Figure 4 shows the flow of data between the entities involved.

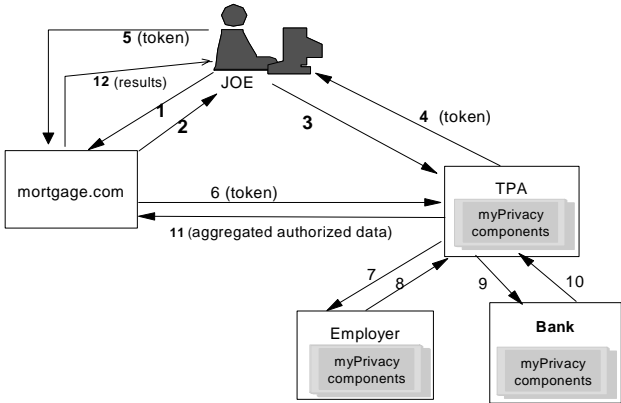


Figure 4:Flow of data between components, user, and enterprise

6. Security and Privacy Evaluation

In order to evaluate the security and privacy of myPrivacy, first we have to define security and privacy goals. Since myPrivacy is a whole system, all security and privacy requirements have to be demanded from the system. The threats that we have to consider are

- unauthorized access to information,
- modification of information and

- impairment of functionality.

Under such threats, the confidentiality, integrity and availability of the system have to be ensured.

The second step is to choose an attacker model, against which we want to provide security and privacy. Now, this is the difficult part, because in a classical approach (taken from military) the model is clear: two honest people want to exchange information over an insecure line and the adversary should not access, modify the information or disturb the functionality. We may use this “old” model for the transmission path of the parties, but our problem in this area is that we do not know which of the involved end-to-end parties are honest.

It is hard to provide privacy and consider all other parties as dishonest. There have to be some parties to be trusted outside the data subject’s own machine. The academic work follows attempts to reduce the “must trust” property of the system as much as possible. Now, with this work, we want to give the users the possibility to choose their attacker models themselves, that is, decide how trustworthy their peers and business partners are. But in real life, no one mathematically rates other persons. They just model them from experience: stranger, known, well-known etc. and along their trust background. The task of myPrivacy is to support technically the ability to express this trust relationship and equip it with security and privacy techniques.

6.1 Fixing a constellation

The first important question is whether the Trusted Privacy Agent (TPA) should be on the user’s computer or on the computer of a Trusted Third Party? As we know from the literature the TTP solution is comparable to the infomediary approach. In [15] following evaluation is given about Infomediaries:

However, from the privacy point of view, a great disadvantage of present-days infomediaries is the lack of user control: The personal data are stored on the server which might log and analyze each access to gain a detailed picture of the user’s behavior. Sometimes infomediaries even build psychograms with sensitive personal data about the user’s psyche. In many cases the providers will not tell the users about their profile because they regard the process of analyzing the user’s behavior as a trade secret.

On the other hand, managing everything with your own machine may have serious availability problems, i.e. the machine may sometimes be off-line or even down. Furthermore, a TPA provider may have better security experts to protect your data to outsiders than you.

However, we also consider in myPrivacy techniques to reduce the “must trust” property for the Trusted Third Party (TTP) solution. We will not discuss in particular these techniques here and evaluate them theoretically along a given attacker model, since it would extend beyond the scope of this paper. The two types of possible constellations: user oriented (TPA is on user’s machine) solution and central oriented (TPA is on a TTPs machine) solution without any additional privacy enforcement builds the security and privacy framework of this approach.

7. Related Work

This paper addresses problems of privacy in the context of personal data management. There are in general two opposite approaches in the area of personal information management and distribution: *information intermediaries* (infomediaries) and *personal information management and protection on user’s own computer*. The first approach is often called the “trust us” approach and the second can be characterized as “do it your self” approach.

7.1 Information Intermediaries (Infomediaries)

Infomediaries store the user data (called **P**ersonal **I**dentifiable **I**nformation or PII) on their own central databases, thus the user has to trust infomediaries [4, 5, 6, 7, 8, 9, 10, 11, 12, 13]. In order to use their service, all communication has to be routed through the Infomediaries. The emphasis of these systems is that convenience to the user. Infomediaries manage data, provide security, and require the user to enter personal data only once, freeing the user from having to fill forms repeatedly with personal information.

A number of infomediaries are known in the Internet. They follow more or less the same philosophy: collect user data and promise to protect them and maximize user’s benefit. They provide services like protect and control children’s online privacy (PrivaSeek), dating service (yenta.com) or electronic wallets.

7.2 Personal Information Management and Protection on User's own Computer

The main difference between this approach and the previous is that all personal information is stored on the user's own computer, thus the user has only to trust his computer and the software. As there is no trust outside his own domain, the goal of the tools is to protect the computer and support the user so that they give as little information as possible to the outside [1, 2, 3, 14].

In some business cases users have to disclose some information about themselves without disclosure of their identity. There might be also cases where disclosing some portion of identity information is required in order to recognize the same person, e.g. in a long lasting negotiation. There are some products of "privacy companies" like Zero Knowledge, Privada. Their products are very similar and consist of software running on the user's computer to protect the computer and support minimum disclosing of information (pseudonym system, cookie manager etc.) and software running on company behalf to hide the traffic of the users (see for example Zero knowledge's Freedom product)

Advanced supporting techniques (algorithms) with more security (cryptographic techniques) are known in this field that provide electronic coins, stamps, tickets (tokens) and credentials. They do not contain any personal information that can be correlated to the "real" identity of their holder.

8. Conclusion

The myPrivacy component approach described in this paper has many novel features, which provide flexible and powerful building blocks for enhancing the privacy of individuals using the Internet. These features are summarized below.

- An individual can express privacy preference policies for controlling access to personal data distributed across multiple enterprises and repositories.
- An individual has complete freedom to specify his own privacy preference policies for any data exchange request, rather than having to decide whether to accept or refuse the policies of the requester.
- An individual can specify complex privacy preferences that include who can access the data, specific purposes for which the data can be accessed, context of the request, how long the data can be retained, who can the data be shared with and for what purposes, and whether the data should be subsequently accessible to the data subject.
- Our components support non-interactive matching of privacy policies specified by an individual to that of the data requester for use automated exchange of data.
- An individual can express complex policies on a large set of personal data in a way that is applicable regardless of the specific representation and data model used by enterprises that store that data.
- Our components support the collection and release of data owned and held by third parties, as well as authorizing release of such data, based on the matching of the privacy policies of the data subject and the data requester.
- Our approach is highly flexible in that it allows for several different ways of conducting business.

References

- [1] S. Brand: *Rethinking Privacy*, Ponsen & Looijen BV, 1999, ISBN: 90-901-3059-4
- [2] J. Camenisch and A. Lysyanskaya. *An efficient system for non-transferable anonymous credentials with optional anonymity revocation*, EUROCRYPT '2001, LNCS 2045, Springer-Verlag, 2001.
- [3] D. Chaum: *Security without identification: Transaction systems to make big brother obsolete*, CACM, 28(10), 1985.
- [4] Microsoft Hailstorm see: <http://www.passport.com/>
- [5] www.PrivacyBank.com
- [6] <http://www.persona.com>

- [7] Jotter: www.jotter.com
- [8] Digitalname: www.digitalname.com
- [9] Lumeria: www.lumeria.com
- [10] Privaseek: www.privaseek.com
- [11] @yourcommand: www.yourcommand.com
- [12] InterOmni: www.interomni.com
- [13] Novell: www.digitalme.com
- [14] Zero-Knowledge-Systems, Inc., The Freedom Network Architecture, <http://www.freedom.net> (2001)
- [15] O. Berthold , M. Köhntopp: Identity Management Based on P3P, Volume 2009, Issue , pp 0141, LNCS
- [16] Customer Profile Exchange <http://www.cpexchange.org/standard>
- [17] Platform for Privacy Preferences <http://www.w3.org/TR/P3P>.