Guarding a Walled Garden — Semantic Privacy Preferences for the Social Web

Philipp Kärger and Wolf Siberski

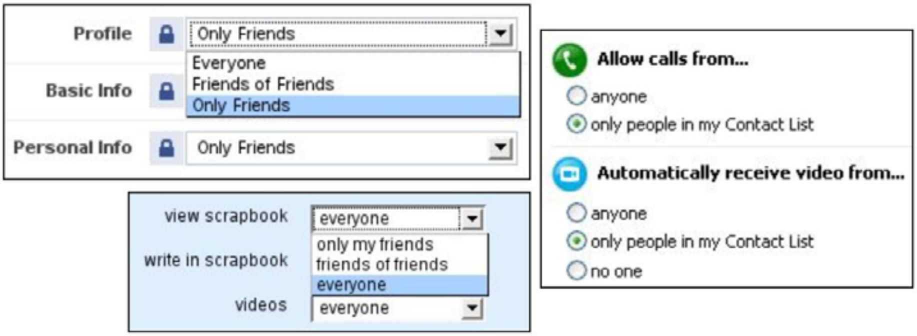
L3S Research Center & University of Hannover, Germany

Abstract. With increasing usage of Social Networks, giving users the possibility to establish access restrictions on their data and resources becomes more and more important. However, privacy preferences in nowaday's Social Network applications are rather limited and do not allow to define policies with fine-grained concept definitions. Moreover, due to the walled garden structure of the Social Web, current privacy settings for one platform cannot refer to information about people on other platforms. In addition, although most of the Social Network's privacy settings share the same nature, users are forced to define and maintain their privacy settings separately for each platform. In this paper, we present a semantic model for privacy preferences on Social Web applications that overcomes those problems. Our model extends the current privacy model for Social Platforms by semantic concept definitions. By means of these concepts, users are enabled to exactly define what portion of their profile or which resources they want to protect and which user category is allowed to see those parts. Such category definitions are not limited to one single platform but can refer to information from other platforms as well. We show how this model can be implemented as extension of the OpenSocial standard, to enable advanced privacy settings which can be exchanged among OpenSocial platforms.

1 Introduction

The Social Web gained momentum in the last years. This is shown not only by the high number of users currently registered and communicating on Social Network applications but also by the growing number of different platforms and applications available. Since more and more data is shared and more and more personal information is exposed on the Social Web, the need for advanced and fine-grained privacy preferences emerges [1]. But when looking at the privacy features of current Social Web applications, one is presented with a restricted and simple mapping between predefined categories of things to protect and categories of people who are allowed to access those information (see Figure 1 for examples). A standard policy, for example, is that specific data or requests, e.g., a user's profile information, is only allowed for people the user has a contact or friendship relation with. But as soon as more complex restrictions are needed to be put on the requester, a mapping between simple categories does not suffice. Examples are "being member of a group", "sharing the same interest", "working for the same company",

L. Aroyo et al. (Eds.): ESWC 2010, Part II, LNCS 6089, pp. 151–165, 2010.



Profile 🔒 Only Friends
Everyone
Friends of Friends
Only Friends
Basic Info 🔒
Personal Info 🔒 Only Friends

view scrapbook everyone
only my friends
friends of friends
everyone
write in scrapbook
videos everyone

Allow calls from...
○ anyone
● only people in my Contact List

Automatically receive video from...
○ anyone
● only people in my Contact List
○ no one

Fig. 1. Example mappings from object categories to subject categories in the privacy settings of Facebook (upper left), Orkut, and Skype (right)

or “being over 18”. On social platforms, just like in real life, people base privacy decisions on social information, such as “is the requester my friend”, “did I ever talk to her”, “is she working in the same project”, etc. The problem is not only that this social information is not available for privacy preferences, unfortunately, the existing Social Web platforms hide all this information behind fences: the Social Web is partitioned into various platforms and thus social data is not linked but encapsulated in proprietary data silos. This issue is often referred to as the “Walled Garden” of the Social Web [2]. The main problem arising from this separation is that information about people and their relationships is trapped inside the platforms and not available outside the platform it was stated in. Thus, social data and contexts are not available for privacy preferences [3].

But not only the social data is isolated, the privacy preferences themselves are trapped as well. Assume a user who may have stated on one platform that only friends of friends can see her profile and only friends can send her messages. This person cannot reuse her privacy policy on another platform, although the second platform may as well have friendship relations, a profile and the possibility to send messages. Consequently, the first thing a user has to do when creating a new profile on a new platform is to recreate her privacy settings since there is no way to exchange those settings and apply one platform’s privacy settings on another social application.

In this paper, we analyse the format and features of privacy settings in current Social Network applications. Based on this analysis, we provide a formal model for privacy preferences on the Social Web that bases on rules and that straightforwardly extends the pairwise mapping in current privacy preferences. Our approach is solving the three aforementioned shortcomings of Social Web privacy settings: (1) It lets users freely define complex categories of persons like “people who are either friends or colleagues”. It also allows for complex categories of actions or objects that are to be protected such as “sending me group invitations” or “seeing my pictures that are tagged with `eswc` and taken in June 2010”. (2) It allows users to define categories of people by referring to arbitrary social data either stored on other platforms or available on the Semantic Web. (3) Privacy preferences defined in this model can be exchanged among platforms since they refer to well-defined semantic categories gathered from all over the Social Web. We show how privacy preferences are enforced based on those category definitions and describe an extension of the OpenSocial standard that implements this model and allows any platform that supports OpenSocial to make use of our model. This implementation also features an RDF serialization of privacy preferences and allows them to be exchanged between platforms.

The remainder of this paper is structured as follows. In the next section we motivate our approach with a scenario and extracted requirements. In Section 3 we review today's privacy preferences on the Social Web. Based on these observations we describe our model in Section 4 and its implementation based on OpenSocial in Section 5. Related work is described in Section 6 and Section 7 concludes the paper.

2 Motivation and Requirements

To illustrate the goal of our approach we start with a motivating scenario. It serves to extract the requirements and it will be used throughout the paper to explain our approach.

Alice is a member of several Social Network applications and platforms, she has an account on Facebook as well as on Orkut. To keep in contact with her friends, she is using Skype for chat and IP telephony. On LinkedIn she is managing her business contacts. As a privacy setting, Alice wants only her friends to access her profile which contains all personal information such as name, age, organizations, address, interest, etc.. With her age, she is even more strict: only her family members are allowed to see it.

Beyond personal data, Alice generally wants to share any uploaded picture only with her friends. As "friend" she considers her contacts in Skype and her friends in Orkut and Facebook. Her contacts on LinkedIn are not included since they are rather business contacts. On Facebook, Alice recently uploaded some pictures she took at ESWC and tagged them with `eswc`. With these pictures she is not as restrictive: she wants to share them with anybody she calls a Semantic Web fellow, that is, anybody who is in the group *ESWC*, who has stated as interest *Semantic Web*, or who is listed as Friend in her FOAF profile.

On all the four Social Networking applications, one can send messages. Alice is quite restrictive here because she is facing a message overload since her network grew. That is why she allows only her Skype contacts to send messages to her on any of the platforms. An exception is messages on LinkedIn, since Alice plans to change her job, she wants all LinkedIn contacts (i.e., business contacts) to be allowed to send messages.

Extracted Policies. To sum up the described scenario we can extract the following policies that make up Alice's privacy preferences:

P1 Disclose my profile information only to my contacts in Facebook, Skype, Orkut and LinkedIn.
P2 Disclose my age on any platform only to my family.

P3 Disclose pictures only to friends.
P4 Disclose ESWC-pictures to Semantic Web fellows only.
P5 Accept messages sent from Skype contacts only.
P6 Accept LinkedIn-messages sent from business contacts.

On top of that, Alice used a personal vocabulary to define her privacy preferences: what she considers an ESWC picture or a business contact may differ from other users. Later in the paper we will refer to those concepts as category definitions. So we can extract the following definitions of categories in her privacy preferences:

D1 *Profile information* is everything that is name, organizations, address, interest, or age
D2 *Family* is everybody who is in my *family*-group on Facebook.
D3 An *ESWC picture* is everything that is both, a picture and tagged with `eswc`.
D4 A *Semantic Web fellow* is everybody who is in the Facebook group *ESWC*, who has stated as interest *Semantic Web*, or who is listed in my FOAF profile.
D5 A *business contact* is everybody who is a contact on LinkedIn.

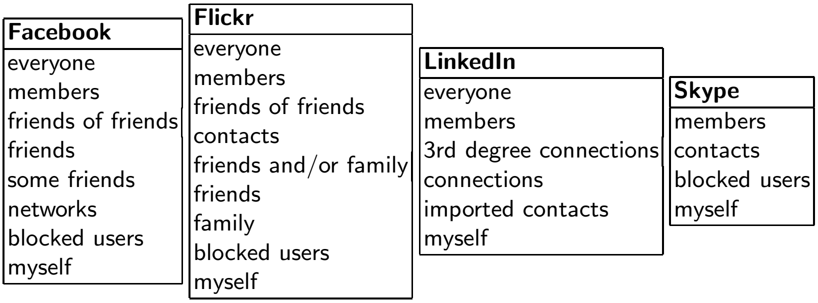
In Figure 3, later in the paper, we provide a graphical representation of Alice's preferences and category definitions.

Requirements. The given scenario requires several extensions to current approaches to privacy on the Social Web. (1) Users may be allowed to freely define new categories of people (like "family members") or of objects (like "ESWC pictures"); thus reflecting their particular social environment. (2) Privacy preferences are expressed crossing the borders of social platforms. Properties of a requester may be gathered from different platforms (like "contacts on LinkedIn") or data sources (like "friends in my FOAF profile") in order to allow a certain action. (3) Policies referring to generic concepts should hold on all platforms, regardless on which platform they were defined. For example, the rule that only contacts are allowed to see Alice's profile information should apply on all platforms where Alice is participant.

3 Today's Social Web Privacy Preferences

Most of the Social Network applications[1] share similar concepts: there is always a profile containing name, an image, contact information, etc. In most of the cases, there is a way to communicate (text messages, wall posts, etc.) and there are connections among people and new connections can be set up. Consequently, the privacy preferences also share the same nature among platforms: they are typically a set of mappings between objects or actions other users can access – called *object categories* in the following – and groups of people that are allowed

[1] With this term we refer to any kind of application that is based on a Social Network, ranging from Web platforms like Facebook to social communication tools like Skype.



Facebook
everyone
members
friends of friends
friends
some friends
networks
blocked users
myself

Flickr
everyone
members
friends of friends
contacts
friends and/or family
friends
family
blocked users
myself

LinkedIn
everyone
members
3rd degree connections
connections
imported contacts
myself

Skype
members
contacts
blocked users
myself

Fig. 2. Subject categories available in the privacy settings of current Social Network applications. They only roughly capture real-life relationships among people.

to access these objects – called *subject categories* (see Figure 1 for example mappings). Examples for *object* categories are "send a chat message", "view a picture", "see address" etc. On the other hand, *subject* categories include "contacts", "friends of friends", etc. Figure 3 lists the subject categories that are available for privacy settings in Facebook, Skype, LinkedIn and Flickr.

Limited options of categories. Objects and actions whose access can be restricted in today's privacy preferences fall into five high level categories:

1. accessing certain parts of a user's profile such as contact data or date of birth
2. accessing specific content such as pictures or videos uploaded to the platform or the user's wall or message board
3. communication actions such as sending chat messages, "poking", inviting to games or sending gifts
4. actions related to connections among persons, e.g., creating a friendship link
5. other actions such as tagging people in pictures

All these categories of objects can be restricted to be allowed only by a specific group of subjects. However, combinations of object categories as they are used in the scenario's definitions D1 and D3, are not possible. Subject categories are even more restricted: looking at Figure 3, it is easy to see that subject categories as they are currently offered in privacy preferences do not reflect the complex considerations one undertakes while making privacy decisions in real life.

No cross-platform definition of categories possible. Privacy preferences can only be defined based on information that is available on the very same platform [4]. Although other platforms may share the same concept of friendship, one cannot define and use categories like "people that are my friends on any platform". Even publicly available social information (that may be available on the Semantic Web) cannot be considered.

No portability among platforms. The object categories and the subject categories are similar in most of the platforms (see Figure 3). Still, there is no method to apply privacy setting defined on one platform to another one. For each platform, an identical privacy setting has to be defined manually.

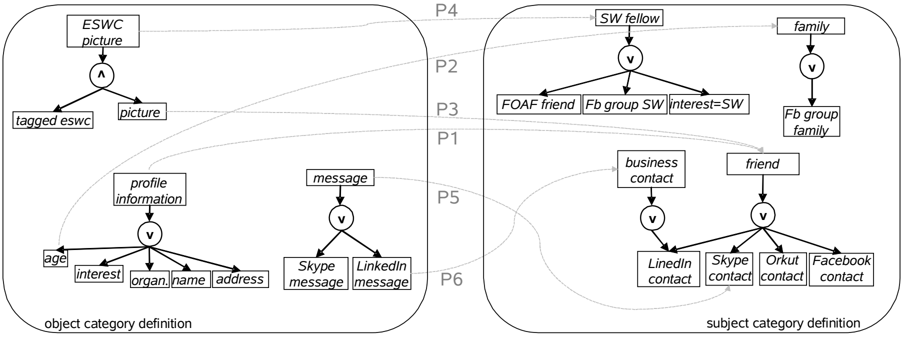


Fig. 3. A graphical representation of the example privacy preferences from Section 2. Category definitions are represented as AND/OR graphs. The policies (P1-P6) mapping object categories (left) to subject categories (right) are represented as dotted lines.

Authorization pairs. Interestingly, the classical authorization or access control triple is not used in Social Web applications. This classical scheme is known from access control in databases or file systems and requires a privacy statement to consist of three parts: an object that is going to be protected, an action whose performance on the object is to be restricted (for example, *access*, *write* and *execute*) and a subject (typically a role) that is allowed to perform the action on the object (cf. Clark-Wilson model in [5, Chapter 2.8]). On Social Web platforms, however, the concept of object and action is merged, resulting in a subject-object pair. An example is given in Figure 1 where writing and viewing a scrapbook in Orkut's settings form two separate object categories actually composed of an action and the actual object. The reason for refraining from the classical triple scheme in the context of Social Web platforms is obvious: first, it is too complicated to be maintained by the average user since the complexity is increased by one dimension. Second, either the variety of different actions that can be performed on the same object is low (profile data can only be viewed, messages can only be sent, etc.) or an action and an object can be merged where necessary (e.g., the scrapbook example in Fig. 1). These two arguments let us keep the authorization pairs pattern for our approach as well.

4 A Unified Model for Privacy Preferences

The requirements and shortcomings described in the previous sections are used in the following to develop a unified and interoperable preference model that fits the needs for the Social Web while still ensuring privacy policies to be enforced.

4.1 Defining New Subject and Object Categories

As stated in the requirements, to express access restrictions on more complex categories of objects, users shall be allowed to define new categories based on the ones offered by a social application. New categories shall be defined as disjunctions of other categories in order to group together concepts that deserve similar privacy preferences. Conjunctions are required wherever existing categories are not fine-grained enough. Categories defined that way shall also be reused to define other personalized categories.

Recalling the example from the motivation (Definition D3), *eswc_picture* is the conjunction of everything that is a *picture* and that is *tagged_with_eswc*. To support both, disjunction and conjunction as well as the reuse of defined categories for more definitions, we represent definitions as Datalog rules as follows.

Definition 1 (Category, category definition, context). *An object category c_o rsp. subject category c_s is a unary predicate whose argument is an object rsp. a subject. An* object/subject category definition *P on a set of categories C is a set of rules of the form $H \leftarrow B_1, \ldots, B_n$. (conjunctive rule) or $H \leftarrow B_1; \ldots; B_n$. (disjunctive rule)*[2] *where $H, B_i \in C$ are object/subject categories and there is no pair of rules "$H_1(X) \leftarrow body_1$." and "$H_2(Y) \leftarrow body_2$." (with $body_i$ being either a conjunction or disjunction of predicates) with $H_1 = H_2$. Properties of objects/subjects are represented as sets of ground facts of object/subject categories stating the category an object/subject belongs to. We call this set of facts the* context *of an object/subject, denoted by Con.*

The two boxes in Figure 3 shows the graphical representations of definition rules in form of an AND/OR graph. An example rule defining an object category is

$$eswc\_picture(X) \leftarrow picture(X), tagged\_with\_eswc(X).$$

New categories of subjects can be defined in a similar way. For example, *Semantic Web fellow* from Definition D4:

$$sw\_fellow(X) \leftarrow inSWGroup(X); swAsInterest(X).$$

The context of a file f that is a picture (i.e., belongs to the category *picture*) and is tagged with `eswc` has the context $\{picture(f)., tagged\_with\_eswc(f).\}$. Consequently, given an object o with its context $Con(o)$ and a category definition P, o belongs to a category c if $P \cup Con(o) \models c(o)$, that is, $c(o)$ is in the semantic consequence of the logic program $P \cup Con(o)$[3]. Contexts of subjects are typically available as RDF, e.g., subject information in FOAF, or gathered from proprietary sources. The context of objects is determined on the platform where the request is happening.

The restriction that a category definition should not contain two rules with the same category in the head is justified by the fact that a category used in privacy preferences is only defined once, either as conjunction or disjunction of other categories. We chose this simplification here because Social Web users are not expected to understand a nesting of conjunction and disjunction. However, by introducing auxiliary predicates, nesting could be simulated easily.

[2] We use ; to denote disjunction in a rule's body. A rule $H \leftarrow B_1; \ldots; B_n$. is a shortcut for the list of n rules of the form $H \leftarrow B_i$. $(1 \leq i \leq n)$.

[3] In the remainder of the paper we may omit $Con(o)$ and use the shortcut $P \models c(o)$ where unambiguously applicable.

$$\boxed{\begin{array}{l} O := \{\mathit{everybody}(\_), \mathit{contact}(\_), \mathit{blocked\_user}(\_), \mathit{myself}(\_)\} \\ S := \{\mathit{seeContactNumber}(\_), \mathit{sendChat}(\_), \mathit{call}(\_), \mathit{sendVideo}(\_)\} \\ \text{Default mapping } \mathcal{M} \text{ when installing Skype:} \\ \qquad\qquad \begin{array}{rl} \mathcal{M}(\mathit{sendChat}(\_)) & := \mathit{everybody}(\_)), \\ \mathcal{M}(\mathit{call}(\_)) & := \mathit{everybody}(\_)), \\ \mathcal{M}(\mathit{sendVideo}(\_)) & := \mathit{contact}(\_), \\ \mathcal{M}(\mathit{seeNumerOfContacts}(\_)) & := \mathit{everybody}(\_) \end{array} \end{array}}$$

Fig. 4. The object and subject categories (O and S) available in Skype and Skype's default privacy preferences $\langle O, \emptyset, S, \emptyset, \mathcal{M}\rangle$ when being installed. Skype has empty category definitions as every current Social Network application.

Restricting categories to be unary predicates is a conceptual simplification. For example, $\mathit{tagged\_with\_eswc}(f)$ could well be understood as syntactic sugar for $\mathit{tagged\_with}(f, \text{"}\mathit{eswc}\text{"})$. Again, we assume users to use unary category definitions in their privacy preferences rather than categories with two or more variables. In fact, in our implementation (see Section 5), we realize a predicate determining tags of an object by SPARQL queries allowing for more than one variable where the tag explicitely stated in the category name is shifted as an argument of the predicate that accepts general tags.

As a consequence of those simplifications plus the fact that no negation is included, the rules used in the category definitions are very simple and the evaluation of a goal (e.g., if an object or subject belongs to a specific category) can be evaluated with `PTIME` complexity [6]. Such simple programs can be represented as AND/OR graphs (see Figure 3) where the nodes that have no outgoing edges are the basic categories offered by the platform and are contained in the context of an object or can be retrieved for a subject (e.g., if someone is a friend in a FOAF profile). The restriction that no category is defined twice is reflected in the graph by the fact that no concept node has two outgoing edges and the outgoing edge always leads to either an AND or an OR node.

Further, it is worth noting that this category definition with rules coincides with the simple Description Logic featuring only conjunction and disjunction of concepts [7]. We stick to the rule representation here, since syntactic restrictions are expressible in a more straightforward way. Further, rules better reflect our implementation based on rule-based Semantic Web policies [8].

4.2 Defining Privacy Preference Mappings

Until now, users are enabled to define new categories for their privacy settings. Following the binary mapping scheme (see Figure 1) we now define how to use these categories to build up a policy. We define privacy preferences as a mapping between object and subject categories—with the difference that those categories are user-defined following the category specifications from Definition 1.

Definition 2 (Privacy preference). *A privacy preference* $\mathcal{P}$ *is a quintuple* $\langle O, P_O, S, P_S, \mathcal{M}\rangle$*, where* O *is a set of object categories,* P_O *is an object category* *definition on O, S is a set of subject categories, P_S is a subject category definition on S, and $\mathcal{M} : \mathcal{O} \mapsto \mathcal{S}$ is a mapping from object categories to subject categories.*

Example 1. Privacy preferences of current social platforms are always of the form $\langle O, \emptyset, S, \emptyset, \mathcal{M} \rangle$ because – as pointed out in Section 3 – category definitions are not allowed. As a more detailed example, Figure 4 shows the default privacy preferences implemented in Skype clients. Further, a graphical representation of the privacy preferences from the scenario in Section 2 is given in Figure 3.

Such a privacy preference is applied in the realm of a Social Web application where the sets of objects and subjects are defined as well as an ownership relation determining who is allowed to enforce policies on which object. Further, each application offers a set of subject as well as object categories which can be used to define personalized categories (see Def. 1).

Definition 3 (Social Web application). *A Social Web application is a hextuple $\langle Obj, Subj, O, S, Cat, Owns \rangle$ whereas Obj is the set of objects in the application, $Subj$ the set of subjects, O and S a set of object and subject categories. Cat is a function assigning a context to each subject and object. $Owns : Obj \mapsto Subj$ is a function defining the ownership of objects, i.e., the subject that is supposed to define privacy preferences for a given object.*

4.3 Enforcing Privacy Preferences

Based on our model for privacy settings, we now define what a request is and how to determine if a request meets a privacy preference or not. Generally, a request to access a specific object is allowed if the requester matches the privacy restriction attached to the category the object belongs to. Since objects may belong to several object categories it is important to determine the correct, most descriptive object categories for a given object. For example, a file f may belong to the category *picture*, to the category *tagged_with_eswc* and thus—according to Definition D3 from the scenario—as well to the category *ESWC_picture*. In this case, the category that describes best what f belongs to is *ESWC_picture*. We refer to those categories as *descriptive categories*. It is intuitive to apply the policy that is defined for *ESWC_picture* instead of the one defined for *picture*. Since Alice defined pictures being visible only to friends but ESWC pictures being visible to Semantic Web fellows (which is a far more general than friends), this intuition is actually what she intended: a requester accessing a picture that is tagged with `eswc` has to meet different conditions than a requester accessing a picture not having this tag. We formally define descriptive categories as follows:

Definition 4 (Descriptive category). *Let P be an Object Category Definition and o an object, then an object category c is a descriptive category of o if*

1. $P \models c(o)$ *and*
2. *there is no* $c' \neq c$ *with* $P \models c'(o)$ *and* $\forall X : P \models c(X) \rightarrow P \models c'(X)$

We define $Des(o)$ to be the set of all descriptive categories of an object o.

A request is a pair $\langle o, s_v \rangle$ where a subject s_v (the viewer or requester in this case) is requesting access to an object o.

Definition 5 (Granting access). *Given a Social Web application and the privacy preference* $\mathcal{P} = \langle O, P_O, S, P_S, \mathcal{M} \rangle$*,* s_v *is allowed to access* o *iff there is a subject category* c_s *and an object category* $c_o \in Des(o)$ *such that* $c_s = \mathcal{M}(c_o)$ *and* $P_S \models c_s(s)$*. That is, the subject must belong to at least one subject category that is mapped to one of the object's descriptive categories.*

So far, we have defined which object category is to be considered for deciding which subject category the requester has to belong to. But looking at the scenario in Section 2 and Figure 3, it may happen that some object categories do not have a subject category mapped. For example, the object $o =$"sending Alice a Skype message" will always lead to denial of access because $Des(o)$ has the only element $Skype\_message$ and $\mathcal{M}(Skype\_message)$ is empty. However, Figure 3 reveals the intuition that the subject category mapped to the super concept of "Skype message" shall be applied, in this case, it is the category that is mapped to "message". Thus, informally spoken, the super categories of a descriptive category shall apply, if for no descriptive category of an object a subject category is mapped. For this we need to define the set of super categories Sup according to disjunctive rules in the object category definition.

Definition 6 (Disjunctive super category). *Given an Object Category Definition* P *and an object category* c_o*, the set of disjunctive super categories* Sup *of* c_o *is defined as* $Sup(c_o) := \{c'_o \mid \exists r \in P : r = c'_o \leftarrow B_1; \ldots; c_o; \ldots; B_n.\}$

Given this concept we can relax Definition 5 to accept incomplete mappings and inherit the subject categories mapped to super categories of an object's descriptive categories.

Definition 7 (Granting access (incomplete mappings)). *Given a Social Web application and the privacy preference* $\mathcal{P} = \langle O, P_O, S, P_S, \mathcal{M} \rangle$*,* s_v *is allowed to access* o *iff there is a subject category* c_s *and an object category* $c_o \in Des(o)$ *such that* $\exists x_1, \ldots, x_n : x_i \in Sup(x_{(i+1)}) \wedge x_n = c_o \wedge \forall x_i (2 \leq i \leq n) \mathcal{M}(x_i) \neq \emptyset \wedge (c_s = \mathcal{M}(x_n)) \wedge (P_S \models c_s(s))$*.*

The chain defined by the $x_1, \ldots, x_n$ is a sequence of object categories defined in P_O where the user defined the category x_i by a disjunction containing x_{i+1}. If for x_1 a subject category is defined and for the remaining x_2 to x_n no mapping has been provided, the mapping for x_1 will be applied. Note that for the case where $\mathcal{M}(c_o)$ is not empty this definition coincides with Definition 5 and $i = 1$.

5 Implementation

In order to validate the applicability of our privacy preference model, we first built a preference reasoner based on the policy engine Protune [9] that handles general category definitions and considers OpenSocial data as well as general Social Web data for the reasoning process. Second, we extended the OpenSocial-based Social Platform Shindig in a way, that privacy preferences defined

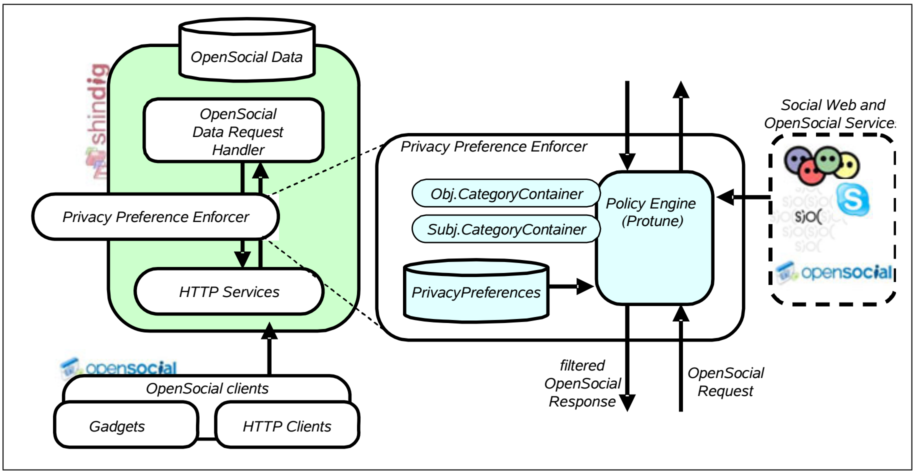


Fig. 5. The OpenSocial container Shindig (left) extended by a general privacy preference enforcement (center). The policy engine integrates external social and Semantic Web data into the reasoning process (right).

according to our model are enforced in any Social Web platform based on Shindig (see an architecture overview in Figure 5). We made our implementation available on line at `www.L3S.de/~kaerger/SocialWebPrivacy`. In the following, we shortly describe the components our implementation is based on and then detail the implementation itself.

Protune. Protune [9] is a policy framework featuring a logic programming-inspired policy language and a policy engine that supports credential handling, trust negotiation, and automated explanation generation of evaluation outcomes. The Protune engine is able to integrate external data sources into the reasoning process such that ground facts do not have to be present explicitly but can be retrieved on demand from external sources during the reasoning process. In the present work we use this feature to incorporate social data from the (Semantic) Web into the reasoning process [4].

OpenSocial. Facing the bulk of Social Web platforms that went on line in the last years each with proprietary technology, OpenSocial [10] is an interface definition describing functions that are common in most SocialWeb platforms. If a platform supports OpenSocial, gadgets or remote procedure calls which were initially implemented for a different OpenSocial-enabled platform can easily be imported. OpenSocial offers four types of requests: asking for information about people (i.e., profile information), about activity notifications (e.g., an image was uploaded, a group was joined), about application data (data that is stored for specific applications inside a social platform, such as applications for sending gifts), and about sending messages.

Apache Shindig. Apache Shindig[4] is an OpenSocial container for hosting OpenSocial web applications. It is an open source implementation for OpenSocial clients (e.g., JavaScripts accessing OpenSocial services) and OpenSocial servers (social platforms offering OpenSocial services). In the present paper, we extended the server-side Java implementation of Shindig to support the filtering of requests from OpenSocial clients.

5.1 A Category-Based Policy Engine

Given a privacy preference as defined in Definition 2, the evaluation of a request is performed in two steps: first, the object categories for the object are determined and second, the subject categories, that are mapped to the object categories are checked for the requester:

1. For the given object and its context (the set of basic categories the object belongs to) the set of descriptive categories (see Definition 4) is derived from the object category definition in the privacy preference.
2. It is then checked if the requester belongs to the subject categories that are mapped to these descriptive categories. For each of those categories, a query to the policy engine is posed.

The available object categories and the specification how to find out if given an object's identifier the object belongs to the category, is defined in the *Object Category Container* (see Figure 5). This container can easily be extended if, for example, a certain environment requires specific object categories, e.g., images that are larger than 2 MB, etc. The available subject categories are stored in the *Subject Category Container* which can also be extended easily in case new subject categories are required. For example, in order to retrieve project memberships of people, a new platform may be integrated that stores projects and the people working in them. Currently, the following subject categories are supported: a person is listed in my FOAF profile, is following me on Twitter, is my friend on Flickr, is my friend on some OpenSocial platform, shows a specific value in the OpenSocial.Person.FIELD[5] on some OpenSocial platform, is my co-author on DBLP. For details about how to gather the social data for the reasoning process we refer the reader to our previous work in [4]. In the following section we describe how this privacy enforcement is integrated into Shindig.

5.2 An OpenSocial Container with General Privacy Preferences

We extended the OpenSocial container Apache Shindig to provide advanced privacy control over data that is exposed by Shindig's OpenSocial interface. If an HTTP request (either JSON RPC or REST, both OpenSocial implementations are available in Shindig) arrives, it is first checked which object is requested and

[4] See `http://incubator.apache.org/shindig/`

[5] See `code.google.com/apis/opensocial/docs/0.7/reference/opensocial.Person.Field.html`

the according object categories are collected. OpenSocial requests typically ask for several objects at once. For example, the request for a person's complete profile contains all the requests for viewing the person's name, hobbies, address, etc. Consequently, such an OpenSocial request is internally transformed into a set of requests for each single object and these requests are passed to the policy engine. As a result, if one of the requests fails, the OpenSocial request is not rejected as a whole but the objects that are not allowed to be accessed by the requester are removed from the response.

5.3 Results

In the following we summarize the features of our approach and explain how our implementation solves the requirements identified in Section 2.

Category definitions. New categories can be defined based on the basic categories that are implemented in the category containers. Object categories currently implemented refer to OpenSocial concepts only, thus each profile information field, sending messages, activity notifications, and application data can be protected. As described in Section 5.1, subject categories can be defined arbitrarily.

Crossing borders of social platforms. Subject categories can be defined based on social data inside Shindig, in other OpenSocial platforms, on other Social Web platforms and on Social Semantic Web data.

Platform independence. In its current implementation, any Shindig-based platform can use the presented privacy model, thus, privacy preferences can be exchanged at least among those platforms.[6] Apart from its actual implementation described in this section, our model is platform independent with the following conditions. The basic subject categories are platform independent and can be applied out-of-the-box. Basic object categories that are part of an object's context instead are platform dependent - however, it is a small effort to adopt the current implementation such that object category decisions can be made on another platform as well. Such decisions are, for example, if a request is for accessing a person's age, for sending a message, or for seeing something tagged with `eswc`. However, as soon as user-defined categories are concerned, be it subject or object categories, they are completely platform independent.

On top of that, our implementation features an RDF export of privacy preferences (basically a serialization of the format defined in Def. 2) that allows the exchange of privacy preferences between Shindig-based OpenSocial platforms. A possible scenario is that a user stores her preferences on a central personal location to be accessed by the platforms and applications she is working with.

6 Related Work

In this section we relate our approach to other work either using Semantic Web techniques for privacy control or extending Social Web standards towards privacy control crossing platform borders.

[6] More details are available at `www.L3S.de/~kaerger/SocialWebPrivacy`

Lockr [11] is an access control scheme for sharing content. It exploits a local address book storing social relationships to be applied for access control policies on several content sharing platforms on the Web. Lockr also motivates the use of one privacy setting across platforms but focuses more on the authentication of subjects (via so-called attestations) than on definition of complex privacy preferences. Our approach adds the idea of defining which categories of objects to share with whom by relying on arbitrary social data whereas Lockr requires users to manually maintain a social network on their local machine and to define a privacy setting for each resource separately.

An approach similar to Lockr is presented in [12]. There, content is shared by emailing secret links referring to the content to a set of subjects. Again, the focus lies more on the authentication of users. Our approach could apply the presented techniques for credential exchange and attribute assertion to establish a subject's context.

The work presented in [13] describes a formal model for privacy preservation on Social Network systems. The focus of this work is to model access control on Facebook-style platforms including the step-wise establishment of friendship relations, etc. Our work builds on top of that since we consider an extension of privacy preferences and their evaluation.

Privacy and OpenSocial is subject of the research presented in [14]. The proposed solution is meant to help users in justifying their privacy settings with the help of a privacy score: the higher the score, the better, the more secure, the more restrictive the privacy settings. An extension to the OpenSocial interface is suggested that is able to deal with privacy scores. This work shares our approach's motivation and complements it, since its goal is to evaluate privacy settings instead of providing better means to express them.

In [15], a Description Logic based access control model for Web 2.0 is described where access control policies are defined as triples of subjects, objects, and permissions. This approach focuses on the use of lightweight ontologies to structure subjects, objects, and permissions. In contrast to our approach, object and subject categories are not defined by rules, but organized in a tree-like hierarchy (in contrast to AND/OR graphs) thus featuring only disjunction of concepts. Further, [15] does not provide a formal definition for the evaluation of access requests.

7 Conclusions

In this paper, we introduce a privacy preference scheme for Social Network applications that is flexible enough to express user-defined categories of objects and subjects. This enables users to reflect their personal social environment in the privacy preferences. Since these categories may leave the realm of one single social platform, privacy preference can be expressed based on arbitrary social data. Furthermore, since our scheme is based on generic categories that are common to most Social Network applications, it can be ported from one platform to another thus avoiding redundant definitions of privacy preferences.

Our implementation shows that this scheme, realized as extension to a standard OpenSocial platform, can be used to provide a simple privacy setting format that works for any OpenSocial compliant platform. Further, since it is based on standards like RDF and OpenSocial, privacy preferences can be exchanged easily among Social Network applications.

References

1. Rosenblum, D.: What anyone can know: The privacy risks of social networking sites. IEEE Security & Privacy 5(3) (May-June 2007)
2. Breslin, J., Decker, S.: The future of social networks on the internet: The need for semantics. IEEE Internet Computing 11(6), 86–90 (2007)
3. Grandison, T., Maximilien, E.M.: Towards privacy propagation in the social web. In: Workshop on Web 2.0 Security and Privacy at the 2008 IEEE Symposium on Security and Privacy, Oakland, California, USA, May 18-21 (2008)
4. Kärger, P., Kigel, E., Olmedilla, D.: Reactivity and social data: Keys to drive decisions in social network applications. In: Second ISWC Workshop on Social Data on the Web, SDoW 2009 (2009)
5. Ferraiolo, D.F., Kuhn, R., Chandramouli, R.: Role-Based Access Control. Artech House (2003), ISBN: 1580533701
6. Baral, C.: Knowledge representation, reasoning and declarative problem solving. Cambridge University Press, Cambridge (2003)
7. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F.: Description Logic Handbook. Cambridge University Press, Cambridge (2003)
8. Bonatti, P.A., Duma, C., Fuchs, N., Nejdl, W., Olmedilla, D., Peer, J., Shahmehri, N.: Semantic web policies - a discussion of requirements and research issues. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, pp. 712–724. Springer, Heidelberg (2006)
9. Bonatti, P.A., Olmedilla, D.: Driving and monitoring provisional trust negotiation with metapolicies. In: 6th IEEE Policies for Distributed Systems and Networks (POLICY 2005), Stockholm, Sweden, June 2005, pp. 14–23. IEEE Computer Society, Los Alamitos (2005)
10. OpenSocial Foundation: OpenSocial API v0.9 (August 2009), `http://code.google.com/apis/opensocial/`
11. Tootoonchian, A., Saroiu, S., Ganjali, Y., Wolman, A.: Lockr: better privacy for social networks. In: CoNEXT 2009: Proceedings of the 5th international conference on Emerging networking experiments and technologies, December 2009, pp. 169–180. ACM, New York (2009)
12. Sun, S.T., Hawkey, K., Beznosov, K.: Secure web 2.0 content sharing beyond walled gardens. In: Proceedings of the 25th Annual Computer Security Applications Conference (ACSAC) (December 2009)
13. Fong, P., Anwar, M., Zhao, Z.: A privacy preservation model for facebook-style social network systems, pp. 303–320 (2009)
14. Liu, K., et al.: Towards privacy-aware opensocial applications. Google Talk (May 2009), `http://www.almaden.ibm.com/cs/projects/iis/hdb/Publications/slides/pr_google_v5_3.pdf`
15. Fausto Giunchiglia, R.Z., Crispo, B.: Ontology Driven Community Access Control. In: Proceedings of the First International Workshop on Trust and Privacy on the Social and Semantic Web (SPOT 2009), Heraklion, Greece (2009)