D6.2

USEMP privacy setting framework – v1

v1.3 / 2015-03-10

Georgios Petkos (CERTH), Symeon Papadopoulos (CERTH), Juxhin Bakali (CERTH), Theodoros Michalareas (VELTI), Timotheos Kastrinogiannis (VELTI), Yiannis Kompatsiaris (CERTH)

This is a report accompanying the first version of the USEMP privacy setting framework that aims at assisting the USEMP users to better perceive and control the exposure of their data. Importantly, it includes tools both for facilitating privacy control within the context of an OSN, and for supporting privacy control with regard to their browsing behaviour, as well as tools that help users understand the value of their personal data. With respect to OSN privacy settings, we present the USEMP privacy settings framework, a user-defined policy that effectively extends the privacy setting facilities provided by the OSN. Moreover, we present a number of automatic and non-automatic methods that can assist the users to appropriately tune the USEMP privacy settings model in a manner that best meets their privacy needs. As part of the non-automatic methods, we present a set of audience monitoring tools that help USEMP users better perceive their audience and the value of their personal data. Regarding automatic methods, we present and evaluate a number of approaches that tackle the problem of social circle detection, i.e. the automatic organization of a user's friends into meaningful groups, and we also present an approach for the automatic characterization of a user's images as private or public. Finally, with respect to browsing behaviour privacy, we expose the operation of web trackers and present the USEMP web tracking assistance tool.

Project acronym	USEMP
Full title	User Empowerment for Enhanced Online Presence Management
Grant agreement number	611596
Funding scheme	Specific Targeted Research Project (STREP)
Work program topic	Objective ICT-2013.1.7 Future Internet Research Experimentation
Project start date	2013-10-01
Project Duration	36 months

Workpackage 6	User Assistance for Shared Personal Data Management
Deliverable lead org.	CERTH
Deliverable type	Prototype
Authors	Georgios Petkos (CERTH) Symeon Papadopoulos (CERTH) Juxhin Bakali (CERTH) Theodoros Michalareas (VELTI) Timotheos Kastrinogiannis (VELTI) Yiannis Kompatsiaris (CERTH)
Reviewers	Noel Catterall (HWC) Katja de Vries (ICIS) Tom Seymoens (IMINDS) Ali Mohammad Padyab (LTU)
Version	1.3
Status	Final
Dissemination level	RE: Restricted Group
Due date	2015-01-31
Delivery date	2015-03-10

Version	Changes
0.1	First outline ToC by CERTH.
0.2	Revised outline ToC and first content posted by CERTH.
0.3	Chapters 2 and 3 filled by CERTH.
0.4	Chapter 5 and Appendix I filled by CERTH.

0.5	Various revisions and corrections by CERTH.
0.6	Chapters 4 and 7 and part of Chapter 3 filled by VELTI, Chapter 6 by CERTH.
0.7	Chapter 3 updated and Annex 2 added.
0.8	Refinements on presentation. Version submitted for internal review.
0.9	Various fixes and improvements based on the reviewers' comments.
1.0	Pre-final version after second internal reviewing.
1.1	Updates and additions to Chapter 4 by VELTI.
1.2	Final proof-reading and style fixing by CERTH.
1.3	Addition of documentation for collaborative filtering prototype by VELTI.

Table of Contents

1. Introduction

The purpose of this deliverable is to document the first version of the USEMP privacy setting framework prototype implementation, a set of tools for assisting the user to perceive and control various aspects of their online presence, both within an OSN and in the Web in general. As described in D4.1, there are two types of privacy issues that may arise in this context: social privacy issues as well as institutional privacy issues. The first kind may happen when a user uploads content on an OSN and it becomes visible beyond the intended audience, while the second type may occur when that content is used by the OSN site itself (Taddicken, 2014). Clearly, users are typically more concerned about social privacy and indeed the primary aim of the privacy assistance framework is to help users better manage their social privacy. Nevertheless, the framework also offers considerable support for the management of institutional privacy. Largely, the work that is presented in this deliverable spans two directions.

The first direction involves the development of a privacy setting framework for controlling a user's presence in an OSN. This privacy setting framework is essentially a privacy policy that will be implemented on top of the privacy setting facilities that are provided by the OSN. It allows for fine-grained control over who has access to the data of a user in the OSN. As will be described in a subsequent chapter, the privacy policy has the form of a set of triplets that link together: (a) a piece of content posted by (or associated with) a USEMP user, (b) the set of users which have access to this content, and (c) the set of access permissions on this piece of content. We define these three elements in a particularly flexible manner so that the policy can be represented in a compact and expressive form. Importantly, the model leverages the expressive power of the privacy scoring framework that was presented in D6.1 – i.e. a hierarchical and multi-dimensional scoring framework that represents different privacy related aspects of the presence of an OSN user. The use of the privacy scoring framework in the privacy setting policy has many advantages, the most important of which is that it allows the user to control access to their content based on the type of personal information that could potentially be revealed. Moreover, we examine various ways by which users can be supported in efficiently defining their privacy policies in a manner that best meets their needs. These include both a set of non-automatic methods, methods that require interaction with the user, as well as a set of automatic methods. Automatic methods include the use of community detection algorithms for automatic friend grouping (including a novel approach that is based on a learned similarity metric), as well as a method for classification of images as private or non-private.

The second direction of work looks into privacy issues not within an OSN environment, but instead with respect to a user's web browsing behavior. Clearly, browsing behavior can reveal a lot of information about an Internet user. To this end, we examine the operation of web trackers, i.e. third-party software entities that can monitor the browsing behavior of users. Moreover, we develop the USEMP web tracking permission framework, which will allow the development of applications that support the user to better understand what part of their browsing behavior is tracked, by whom and how the user controls which information is accessed by trackers.

The rest of the document is structured as follows. Chapter 2 presents an overview of recent research on the topics of privacy policies and privacy setting assistance. Subsequently,

Chapter 3 presents the overall USEMP privacy setting model, with an emphasis on the representation of USEMP privacy policies. Then, Chapter 4 presents a set of non-automatic mechanisms for assisting users to define their own privacy policies. Chapter 5 on the other hand presents a set of automatic methods for assisting users with the effortless definition of their privacy policies. Subsequently, Chapter 6 turns the focus on privacy with respect to web browsing behavior, examining the use of web trackers, and presenting the USEMP web trackers permission framework. Finally, Chapter 7 concludes the deliverable and discusses future work. Three Annexes are included at the end of this document; Annex 1 presents a JSON serialization of the USEMP privacy setting model, Annex 2 presents details on the Facebook privacy settings model, and Annex 3 provides details about the prototype implementations that accompany the report.

2. Related Work

In this chapter, we review previous work on the problem of assisting OSN users to protect their data. As mentioned in (Madejski et al., 2012) and in (Strater & Lipford, 2008), there are two primary ways by which OSN users can protect their data. The first is to not post the data at all. The second is to define privacy controls in order to manage who can see what.

Although the first option may seem somewhat paradoxical, there has indeed been some work that focused on assisting users to decide as to whether a particular piece of content should be posted or not. A relevant example is the work by (Kawase et al., 2013). In their study, Twitter users posting tweets that express a negative feeling against their job are warned about the entailed risks and are advised to either delete or modify them. In fact, as shown in (Sleeper et al., 2013), situations like the above are quite common and a system that would warn users to not post content that they would later regret could be of great value. Another piece of work that attempts to prevent users from posting content that may be sensitive is presented in (Wang et al., 2013). In their study, three different “nudges” are introduced. These are small hints provided by the user interface that allow the user to perceive the impact of the content that they are about to share and reconsider whether they will eventually share the content or not. The first suggested nudge displays a sample of the people that will have access to the content; the second nudge introduces a time delay between the moment that the user posts the content and the moment when it is actually posted to the OSN, allowing the user to reconsider it; the third nudge detects posts with negative sentiment and warns the user. Importantly, nudges are one of the few tools that can be used for protecting not only social privacy, but also institutional privacy. The OSN site always has access to all content posted by the user, regardless of the privacy settings that have been defined; thus, the only way to protect institutional privacy is to not post the content at the first place. To this end, the second and the third nudges could prove to be useful for protecting institutional privacy, whereas a nudge that would highlight any type of personal or sensitive information would be even more useful for managing social privacy.

Compared to the approach of nudging to prevent users from posting certain pieces of data, there has been more research on approaches that support users with the definition of appropriate privacy controls. In the following sections, we review several such works.

2.1. Challenges of defining privacy control settings

Before presenting some approaches that aim at supporting users in defining their privacy control settings, it is useful to review works discussing the challenges of the task. As shown by such studies, the definition of appropriate privacy settings in OSNs is a notoriously difficult task. Indicative is the study of (Madejski et al., 2012) that presents a review of privacy setting errors in Facebook: the privacy settings of 65 participants were examined and it was found that all of them had at least one sharing violation, i.e. they were all sharing content with people that they really would not like to. They also present percentages of confirmed violations per information category; e.g. information related to family status, political views and alcohol consumption were those that were most often inadvertently disclosed. They conclude that the main culprit for privacy flaws in Facebook is the reliance of policies on data types (e.g. photos, events, status updates) and the neglection of the actual information that is

expressed in the content. For instance, an image that shows the user smoking may be perceived as more sensitive than an image that shows the user reading a book. They suggest that a key improvement would be to automatically categorize content according to the information it contains and make privacy settings reflect the user's preferences with respect to disclosure of different types of information.

A pertinent study that investigates users' concerns with respect to privacy settings and the related challenges is presented in (Johnson et al., 2012). This concludes that users are typically concerned about particular friends having access to content revealing specific types of information. This in fact points to the need for privacy settings that take into account both particular people in the network of a user as well as the type of information reflected in the content that the user posts. Their study also argues that for a typical user, the division of their friends into two groups, strangers - people that the user has not really met in person - and real-life friends makes sense for assisting the user in defining their high-level privacy settings. As expected, users are more concerned with sharing their content with strangers, rather than with real-life friends; but still, a large number of users were also concerned with sharing some of their content with at least one real-life friend.

An important complication with respect to privacy in social media is related to incidental data. Incidental data is posted by some user but refers or is related to another (see the taxonomy presented in Section 2.1 of D6.1). For example, a user checks in at a location but also mentions the friends that are with him/her, or posts an image and tags some of their friends that appear in the picture. Clearly, in such scenarios the privacy settings of the person that posted the content and the privacy settings of the other people that are related to the content may be in conflict. This issue is recognized in (Squicciarini et al., 2009) and (Hu et al., 2011), where different approaches are introduced in order to resolve such conflicts.

To conclude this section, it needs to be stressed that both the study of (Madejski et al., 2012) and the study of (Johnson et al., 2012) indicate that tools helping users with their privacy settings should provide assistance with respect to a) the type of information that is expressed in the content that they post and b) the more fine-grained and accurate definition of the audience to which their content is visible. In the following two sections, we will review some existing approaches that target each of these goals. In fact, the development of our own privacy setting assistance framework that is presented in the next chapter will also focus on these two goals.

2.2. Content control assistance

A recent piece of work that aims at assisting users on defining their privacy settings based on the type of information expressed in the content that they post can be found in (Klemperer et al., 2012). In their study, it is shown that the tags of users' posted pictures can be used to automatically build accurate access control rules, without user effort. In fact, tags can be considered as a proxy to the actual content of the images and therefore this approach does not require any intricate image analysis process in order to infer the information that is contained in the images. The usefulness of tags and other user-provided elements that express the semantics of content in order to build access control policies has also been demonstrated in other studies such as in (Vyas et al., 2009) and (Hart et al., 2009).

Of particular interest in this class of methods is the method presented in (Squicciarini et al., 2011), where the authors use the tags and metadata, as well as the actual content of images, in order to assign images to categories. For a new image, they extract these categories and use them in order to retrieve the access control rules for other similar images, as provided by the user. Finally, the retrieved rules are combined to obtain a single access control rule for the new image. Thus, this approach both extracts the significant information from the content and metadata and learns to predict access control rules based on rules that have been provided by the user for other images.

All in all, the work on privacy assistance methods that take into account the content is rather limited and it appears that there is indeed need for further research.

2.3. Audience control assistance

Contrary to methods that focus on content, methods that focus on audience have gained more attention. At a high level, these methods belong to two categories. The first involves automatically identifying groups of users with common characteristics, which may be used to allow or disallow access to specific pieces of content. Such groups of users are often called social circles. The second involves examining individual friends of a user and either characterizing the strength of their bond or automatically predicting privacy settings that apply to them.

The first class of methods involves a number of approaches that have applied a variety of community detection algorithms (Fortunato, 2010; Papadopoulos et al., 2012) to graphs representing the friendship networks of OSN users. For instance, (Fogues et al., 2013) compares a number of community detection algorithms and finds that Infomap (Rosvall & Bergstrom, 2008) performed best: the produced communities better matched the ground-truth communities provided by the users. Another similar piece of work can be found in (Adu-Oppong et al., 2008), where the method of (Mishra et al., 2007) is used. More advanced approaches such as the CESNA method (Yang et al., 2013) consider not only the structure of the friend network but also the characteristics of the friends, e.g. their location, the school they attended, etc. A similar approach is presented in (McAuley & Leskovec, 2012); both approaches work by building a generative probabilistic model. Other advanced approaches also take into account the strength of interactions (Dev et al., 2014).

Importantly, there are a number of studies that examine the criteria that users actually consider when putting their friends to groups; such criteria would also be useful for automatic methods. One such study (Jones & O'Neill, 2010) identifies six relevant criteria:

1. Existence of cliques, involving groups of friends being highly connected to each other.
2. Tie strength, which in fact may have various aspects: closeness, emotional intensity of relationship, level of trust and frequency of communication.
3. Temporal episodes. Some groups tend to emerge from people that are all present in a significant event.
4. Geographical locations and proximity.
5. Functional roles. Some links in a social network tend to form because they have some particular use or provide some specific service to the user. For instance, one participant of the study reported that she had added as a contact a person that she had encountered through a classified advertisement, so that the link could be used as a bookmark for communication regarding the transaction.
6. Organizational boundaries. For instance, people that work in the same company.

Interestingly, (Jones & O'Neill, 2010) also argue that the SCAN clustering algorithm (Xu et al., 2007) is very suitable for capturing most of these factors and they perform some relevant experimental analysis supporting this claim. Moreover, they identified that users have difficulty grouping particular friends and they found that this was because these friends either had a weak association with any of the groups or they had strong associations with multiple groups. Since SCAN could successfully identify such people as outliers or hubs, it could be useful not only for grouping of friends but also for identification of friends to which users should pay more attention when building their privacy policy.

A further study that examines the mechanisms by which users group their OSN friends is presented in (Kelley et al., 2011). They identify specific types of groups of OSN friends:

- General friends, with some specific sub-categories: location-based, generic friends, close friends and friends of friends.
- College friends, either general college friends or college club/group friends.
- Friends from other education: high school friends and grade school friends.
- Other categories such as family, work, church and "don't know" friends, people that the user hardly knows or has never met in person.

Nevertheless, (Jones & O'Neill, 2010) conclude that there are important pitfalls when attempting to build a completely automated method for grouping friends. The reason is that they found that different people considered different grouping criteria to varying degrees. Thus, it is necessary to consider the different prioritization of the users and this is hard to achieve in a completely automated manner.

In the second class of methods, those that examine individual friends of the user, a number of methods focus on characterizing the tie strength between a pair of friends. We have already seen that tie strength is an important factor that users often take into account when grouping their friends, however, tie strength can also be used to derive per-friend access rules. (Gilbert & Karahalios, 2010) present an approach in which tie strength between a pair of users is predicted using a simple linear regression model. Their model takes into account features such as the number of inbox messages exchanged, the number of "social wall" messages exchanged, the number of photos in which both users are present, the number of days since last communication, various network measures, etc., for a total of 74 features. (Fogues et al., 2013) presents a similar approach but with the number of predictors reduced to 14, leading to a faster system, as the OSN API needs to be queried far fewer times. A further approach along the same lines was proposed by (Spiliotopoulos et al., 2014).

A very interesting study is presented in (Fang & LeFevre, 2010). The authors propose a scheme in which the privacy settings for a specific friend of an OSN user are predicted using a decision tree classifier. Two types of features are used as input to the classifier: profile information, such as gender, age, education, etc., as well as community structure features. The latter are extracted by performing multi-granular community detection on the ego-network of the user, i.e. the network consisting of the user, his/her friends and their pairwise connections. Moreover, the authors propose an active learning scheme, in which the user is asked to explicitly provide the correct output of the classifier, particularly for those friends for which the prediction of the classifier is the most uncertain. It should also be noted that the use of a decision tree classifier allows for a quite simple and intuitive visualization of the privacy policy, a feature that proves to be quite useful for the end user.

2.4. Other methods for privacy control assistance

In the previous sections, we reviewed a number of approaches for assisting the definition of privacy settings by focusing either on specific pieces of content or parts of the friends' network of a user. Nevertheless, there are approaches for privacy setting assistance that do not fall in any of these categories. One of them can be found in (Bonneau et al., 2009), which proposes the idea of users adopting the privacy settings of friends or trusted experts.

Another interesting approach can be found in (Minkus & Memon, 2014) who recognize that different users have different perceptions of privacy and therefore it would make sense to apply recommendation techniques to suggest appropriate privacy settings to them. The proposed approach utilizes personality traits, demographics as well as the self-reported level of privacy concern in order to perform this recommendation task.

Some further approaches focus on visualization or advanced interface designs to assist users in perceiving and defining their privacy settings. For instance, (Egelman et al., 2011) proposes the use of a Venn diagram-like visualization that allows the better perception of the applied privacy settings. (Watson et al., 2009) propose the use of AudienceView, an interface that allows users to better perceive their audience in the OSN.

3. USEMP Privacy Setting Model

In this chapter we introduce the USEMP privacy setting model. At the heart of the privacy setting model, there is a set of user-defined privacy policies. These specify a compact, highly expressive and easy to understand set of privacy settings. The model implementation is carried out by a) interaction with the privacy settings facilities that are provided by the OSN, and b) suggesting to the user specific privacy settings that they will then have to define directly through the OSN. Ideally, all privacy settings would be set directly through the OSN (e.g. Facebook) API; however, there are often strong limitations imposed by the respective APIs. For instance, as described in the next section, it is possible to delete a post using the Facebook API, only if it has been published through the same application; moreover, it is not possible to update an existing post. Thus, some of the suggested privacy settings will have to be defined directly by the user. All in all, the privacy setting model is positioned one level above the privacy setting facilities of the OSN and uses them, together with the set of privacy settings that are suggested to the user, to implement the higher level policy that it represents. This is schematically depicted in Figure 1. Note that although we use Facebook as a working example, the presented USEMP framework is applicable to additional OSNs (e.g. Twitter) taking into account the specific capabilities and limitations of the respective APIs.

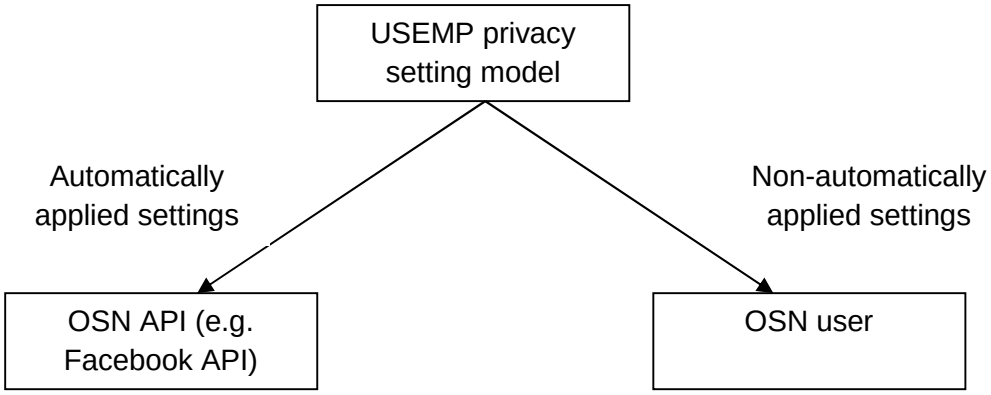

Figure 1. High-level implementation of the USEMP privacy setting model

In subsequent chapters, we will also examine mechanisms, some of them similar to those described in the previous chapter, by which the user can be further supported in defining their privacy setting model in a manner that best suits their needs.

3.1. OSN privacy settings in Facebook

Before introducing the formal description of the USEMP privacy setting model, it is important to review the privacy settings facilities offered by Facebook (at the time of writing this report – January 2015). This will serve two purposes. On one hand, it will provide the foundations on which our own privacy settings model will be based (in the sense that the realization of our policy is carried out on top of the services provided by the Facebook mechanisms). On the other hand, it will help us identify some drawbacks of the privacy settings facilities provided by Facebook, so that our model focuses on alleviating them. In the following, we provide only

a general description and point to Annex 2 for a more detailed presentation. The descriptions we provide are based on version 2.2 of the Facebook Graph API.

At a very high level, Facebook provides two ways by which privacy settings can be managed. The first is through the web UI. From the UI, the user has two main options for privacy control over the content that they post. The first is to determine who will have access to each piece of posted content. The second is to determine who will have access to each type of data (e.g. posts, images, videos, etc.) that the user posts. This is in fact a first drawback of the privacy setting mechanism offered by Facebook: privacy settings are managed on a per item basis or with very coarse rules. So, the user typically defines the access level for each of their posts, messages, images, profile attributes, etc., one at a time. For a user with a strong presence in the OSN, managing their privacy settings in that way will be a laborious task. Ideally, the user should be able to determine both a per-item policy, as well as more general policies that apply not only to single items but to groups of items that are defined according to the type of information contained in them. A second shortcoming of the privacy setting mechanism offered by Facebook (and any other OSN in fact) is that it ignores inferred information. That is, since the policies that can be defined in Facebook typically consider only the type of data and not the information contained in them, they also do not take into account inferences that are possible about the user and the type of personal information that these inferences may reveal. Nevertheless, inferred information about a user may in many cases be critical to protect and therefore a privacy policy should also consider such information.

The second way by which privacy settings can be managed is through the Graph API. The Graph API allows for programmatically managing Facebook content. Importantly, the Graph API will be used by our system in order to partly implement the USEMP privacy settings model. Unfortunately, as described in more detail in Annex 2, there are strong limitations about what actions can be performed using the Graph API. For instance, it is in general not possible to obtain access to all content that the user has posted. Moreover, it is not possible, using the Graph API, to edit or delete posts that have not been posted through the same application. That is, the application that will realize the privacy settings assistant will not be able to change the privacy settings of posts that the user has posted directly to Facebook, without using the application. Thus, for some cases, the user will have to directly, manually take action in order to implement part of the policy. Annex 2 provides more details about the current capabilities of the Facebook privacy settings model.

3.2. Overview of USEMP privacy settings model

Having reviewed the privacy settings facilities offered by Facebook, we now proceed to describe in more detail the USEMP privacy settings model. As already discussed earlier in this chapter, the USEMP privacy settings model sits on top of the privacy settings facilities offered by Facebook (or any other OSN) and utilizes its services.

In order to design the USEMP privacy setting model, we will draw from classic access control models. A typical access control model consists of a set of triplets, each of which contains a subject, an object and a type of access permission. For instance, in a typical access control model implemented by an operating system to control access to its resources, such a triplet may contain a user (or group of users), a file (or any other resource managed by the operating system, such as a printer) and a number of permissions that are applicable to the type of the particular object. Similarly, our privacy setting model will also contain triplets of

the same form; however, it will be adapted so that it is suited to the interactions that may take place in an OSN. For instance, in order to better match with the OSN setting, we will use the term “audience” rather than the term “subject”, and the term “content” rather than the term “object”. This correspondence is shown in Figure 2. Most importantly, the presented USEMP privacy assistance framework incorporates flexible and efficient ways by which these parts of the triplets can be defined.

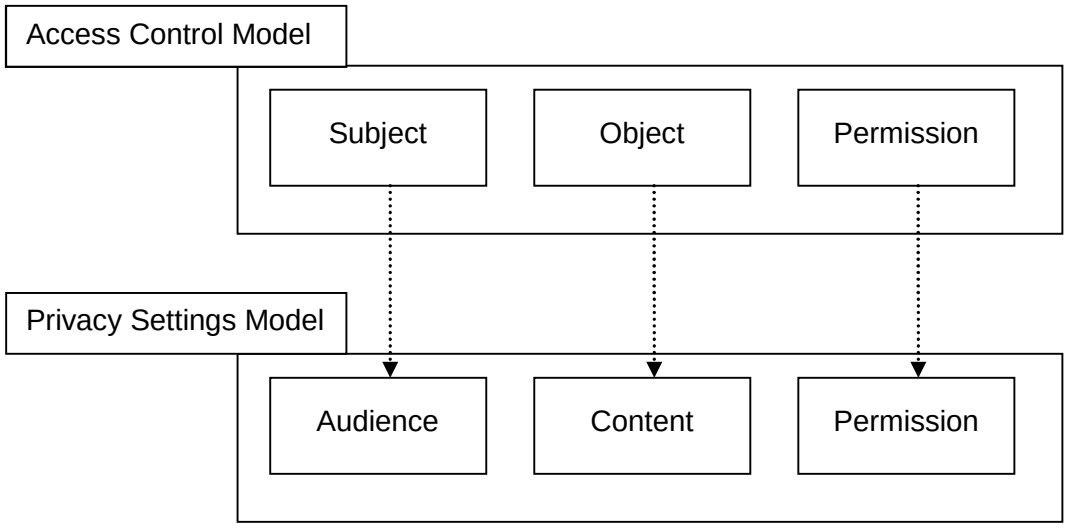

Figure 2. Relationship of the USEMP privacy setting model to a typical access control model.

In the following, we discuss the details of the “audience”, the “content” and the “permission” parts of our model. Finally, we discuss implementation details of the model; in particular, we examine details regarding the computation of the permissions for some particular piece of content when multiple rules are involved.

3.3. Audience model

In a typical access control system, the subject is typically defined using either a unique user identifier or a group identifier and groups are created manually. In our approach, we will consider alternative ways to define the audience in a privacy setting triplet. This will allow the user to select the mechanism that better suits their needs given the scenario at hand.

Clearly, the first option is to allow the user to select a single user. This will allow the finest possible definition of audience. Moreover, we will allow the user to manually define a group of users. As shown from the actual experience, this is not very handy and not preferred by most users but it may be preferred by some (Facebook has stated that only ~5% of users have created at least one list of friends[1]). Nevertheless, we will also allow the user to create sets of users using two kinds of alternative mechanisms.

The first will require some interaction with the user and will involve the filtering of the friends of the user according to user-specified criteria. The proposed USEMP assistance framework allows users to define a set of keywords and retrieve users that are relevant to that set of keywords. Another option is to define a set of demographics-based criteria, information that

[1] http://www.fastcompany.com/1693443/facebooks-new-groups-dashboards-and-downloads-explained-video

is explicitly provided by the users, for instance, a group of users may be defined by selecting all users that work in some company, or all users that come from a specific city. More details on the possibilities of manually defining one's target audience will be provided in section 4.2.

The second option for audience definition will be automatic and will involve the application of community detection algorithms in the user's ego-network. Community detection algorithms analyze the structure of such networks in order to identify groups of users that are more densely connected to each other than to the rest of the network, and hence typically correspond to meaningful groupings, e.g. friends from work, family, friends from school, etc. More details about these will be provided in Chapter 5.

In short, the audience model of the USEMP privacy setting model allows the user to define the audience in a privacy triplet by any of the following ways:

- Selecting a single user.
- Manually creating a set of users.
- Defining a set of explicit criteria based on which users are filtered (based on demographics, e.g. attended school, or more loosely defined criteria).
- Automatically grouping friends according to the structure of the ego-network of the user, by community detection algorithms, i.e. finding groups of friends that are more densely connected to each other compared to the rest of friends.

All in all, these mechanisms allow for a much more flexible way to identify audiences than the existing mechanisms offered by Facebook (and other OSNs) and are a first step towards dealing with the problems that were outlined before.

3.4. Content model

The audience model of our privacy setting model that was presented in the previous section, achieves greater flexibility and efficiency than the standard privacy setting mechanisms of Facebook because a) it provides multiple ways to define audiences and b) it allows for the easy definition of sets of users. In order to achieve similar flexibility in our content model as well, we will provide a variety of alternative content selection mechanisms.

With reference to our previous discussion about the privacy mechanisms in Facebook, let us repeat that in Facebook a user may define the privacy settings for individual items (e.g. a single post, status update, profile information, etc.) and for types of content (e.g. posts, images, etc.). What is really missing is that the selection of content should also be possible based on information regarding the type of personal data at hand and in addition, it should consider inferred information. This is also a key conclusion from previous analyses, as described in the previous chapter.

To achieve this, we leverage the privacy scoring framework that was presented in D6.1. As a reminder, the USEMP privacy scoring framework comprises a hierarchy of privacy dimensions, attributes and values. The attributes are the characteristics of a user, each of which can take a number of values, and dimensions group together similar attributes. For instance the privacy dimension "demographics" includes attributes such as the age, the sex, the family status, etc. of a person. Additionally, for each element of the hierarchy, a number of scores are defined, expressing various privacy-related aspects of the OSN presence of the user with respect to that particular dimension, attribute or value. A schematic of the scoring framework is presented in Figure 3. For more details on the framework please see D6.1.

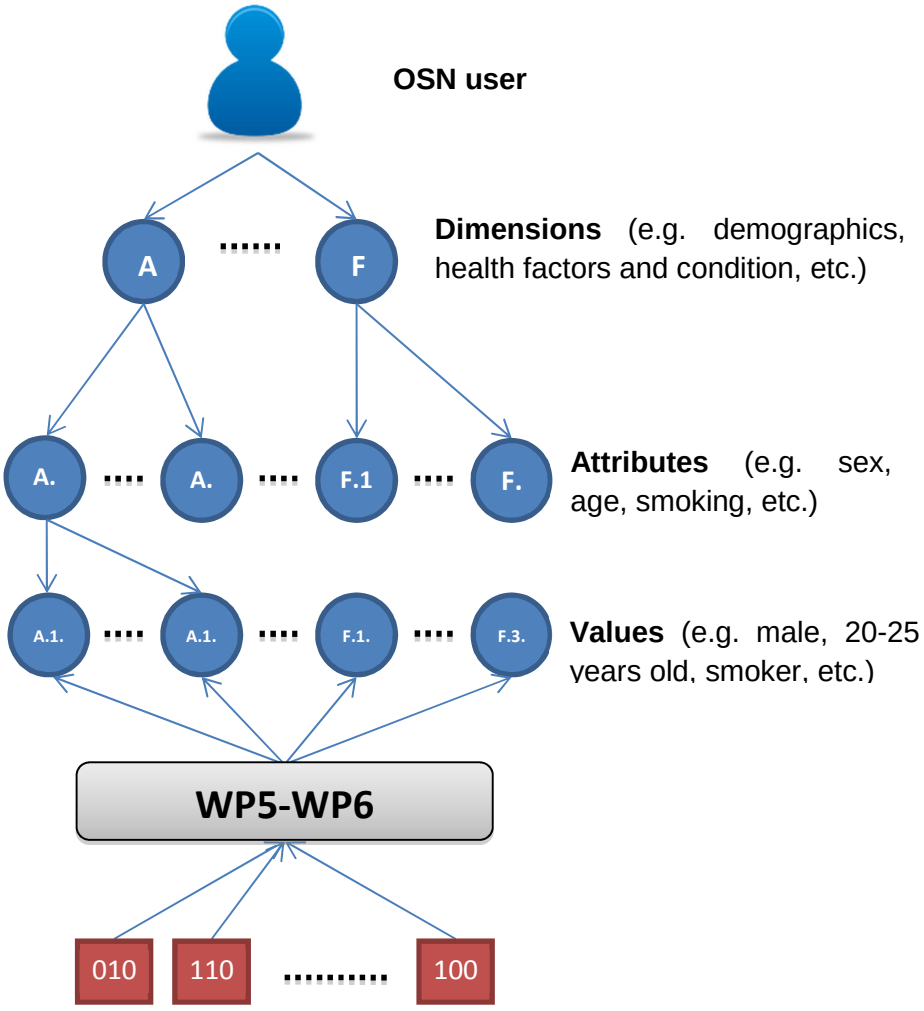

Figure 3. Overview of privacy scoring mechanism

The privacy scoring framework is useful for our goal because it provides a simple and concise way to group content based on the type of shared personal information. Moreover, the privacy scores in combination with appropriate visualizations support the user in accurately perceiving and reasoning about privacy risks in order to consciously form an appropriate privacy policy. Thus, we will allow the user to select some dimension, attribute or value as the content identifier to which some triplet applies. For instance, we allow the user to select all content that is related to their demographic information, their smoking habits or to their age. The link between criteria related to specific dimensions, attributes of values and specific OSN presence data is performed by utilizing the "support" field of each value. As described in D6.1, the "support" field points to specific OSN presence data that either directly or indirectly indicate that some particular value is true for some user. If the criterion is related to some attribute or dimension, rather than to some value, then resolution of the criterion to relevant content can be performed in a top-down manner.

Moreover, the privacy scoring framework can be used to select content in additional ways. In particular, content may be selected based on privacy scores, rather than on specific dimensions, attributes or values. For instance, a user may opt for selecting all content that is related to some value with privacy score above some threshold. It is also reminded that the

overall privacy score for some value is a function of partial scores that express the confidence about that value, the sensitivity of that value and the visibility of the value. A user could also specify criteria based on either the overall privacy score or any of the partial scores.

The use of the privacy scoring framework allows for a flexible mechanism that deals with the drawbacks that were identified for the Facebook privacy setting mechanisms. However, we also foresee additional mechanisms for content selection. At a basic level, we will allow the manual selection of single items. In addition, in line with audience selection, we will allow content selection based on user-defined criteria, in particular keywords or temporal criteria.

To summarize, the USEMP privacy settings model provides the following mechanisms for defining sets of content items on which the privacy triplets apply:

- The user can select a single content item.
- The user can define a specific privacy dimension, attribute or value.
- The user can define criteria based on privacy scores.
- The user can define a combination of a privacy dimension, attribute or value and a privacy score criterion.
- The user can define some keywords or some temporal criteria.

3.5. Permissions model

The third component of the privacy triplets involves the set of possible permission values. As mentioned, in a general access control model, this depends on the nature of the considered object. In an access control system that manages a file system for example, possible permissions include “read”, “write” and “execute”. In an OSN setting, the set of possible permissions would include “permit access” and “deny access”.

Moreover, we enrich the set of possible values for the third part of the triplet, also with values that express actions and not only access permissions. In particular, we have actions for “untag”, and “delete”. These actions would be applied on the content defined in the triplet. It should be stressed that actions will not be automatically performed; rather, actions will be performed after being explicitly approved by the user. It can also be noted that, in some sense, this third part of the triplet can be considered not as a permissions part but as a permissions/actions part. It is clear that since actions pertain to content and not to audiences, then the first part of the triplet for such rules is redundant and can thus be left blank.

A further possible permission could include “permit access but not reproduce”, which would mean that the subject can view the object but cannot repost it or share it. It is not feasible using the services currently provided by OSNs to implement such permissions even though they would offer a very powerful information control feature.

To sum up, we provide a formal, set-based, definition of the overall privacy setting model. In short, the proposed formulation specifies that a policy is a set of triplets, each of which consists of an audience, a content and a permission definition. The audience and content definitions correspond to sets and they can be constructed in a variety of manners. The permissions part is either permission (allow or deny) or an action (delete or untag).

Policy	= {Triplet}
Triplet	= {Audience, Content, Permission}
Audience	= {User | User has been manually picked}
	= {User | User meets filtering or demographic criteria}
	= {User | User set automatically produced using community detection}
Content	={Content | Content has been manually picked}
	={Content | Content under specific privacy dimension, attribute or value}
	={Content | Content meets criteria based on privacy score}
	={Content | Content meets both privacy dimension / attribute / value and scoring criteria}
	={Content | Content meets keyword-based or temporal filtering criteria}
Permission	$\in$ {Permission| Permission $\in$ Permissions or Permission $\in$ Actions }
Permissions	$\in$ {Allow, Deny}
Actions	$\in$ {Delete, Untag}

3.6. Implementation of privacy settings framework

The last step for completing the privacy setting model is to specify the computational process by which it is implemented. As mentioned early in this chapter, part of the policy will be implemented by utilizing appropriate OSN (Facebook) API and part will be implemented by asking the user to manually define some of the settings that the framework will suggest to them. This is due to the fact that not all settings can be applied in an automatic manner. Different limitations in using the Graph API have been mentioned. For instance, it is possible to add or remove friends from a friends list using the API; however, as already mentioned, it is not possible to use the API to update a post that has already been posted. Limitations like these are quite unfortunate, but, other than suggesting to the user to manually define the recommended settings, there is nothing that can be done about it (at least at the moment). In the following, we will examine some additional implementation details, other than this issue.

One first detail is related to the granularity of the implemented policy. Facebook allows only per item or per data type control. Since we want to account for the type of information that can be inferred from each piece of content, we will have to translate the USEMP policies to per item Facebook access rights in order to implement our policies. Thus, behind the scenes, our mechanism will work on a per item basis, even though the set of privacy rules will also contain definitions that apply to groups of items. For each item in these groups, our system will either directly set the corresponding privacy settings (e.g. for a new post that has been posted through the application, it will be possible to change the privacy setting) or it will

suggest to the user to change the Facebook privacy settings for it (e.g. for an older post that has not been posted through the application).

Another issue has to do with the fact that we need a default policy for pieces of content and friends that are not captured by any of the rules. There are two general options, either to default allow or to disallow access. Although it is reasonable to say that the default should be to allow access, as most users would think that their policy would be used to restrict access to their content and thus anything that is not captured by their policy is allowed, a number of users may instead perceive the situation the other way around. That is, they may disallow access to all content to everyone and just create policy rules as to who should have access to what, rather than not. Thus, the default will be to allow access but users will have the option to change this.

An important issue pertains to the need for a computational model to resolve the most appropriate privacy setting in cases of combinations of privacy rules. As explained, behind the scenes, decisions will be made on a per item basis. However, a piece of content may conceptually belong to different sets - e.g. it may be related to different privacy attributes - that may be part of different rules. Also, an OSN user may belong to different audiences. Therefore, a number of triplets may be applicable for the same user and piece of content and they need to be combined. For instance, consider the case of a piece of content for which the user's policy says that it should not be accessible to her family but should be accessible to her co-workers. Now, suppose that the user's brother works in the same company as the user; thus, he belongs both in the 'Family' circle as well as in the 'Co-workers' circle. Should the user's brother have access to the content? Another scenario could be the following. Suppose the user posts a picture of him smoking in his holidays. Then, his policy may dictate that content indicating that he is smoking should not be accessible by his sport club colleagues and that at the same time that content related to holidays should be accessible to them. Should this image be accessible to his sport club colleagues? All in all, issues like those are quite challenging; in fact, multiple memberships of friends in different groups has already been identified in Chapter 2 as a key privacy setting problem.

There has been some research on this problem; for instance, (Benferhat, 2003) attempts to automatically create a prioritization of access control rules so that this prioritization can be used for conflict resolution. In practice, we foresee the implementation of a simpler approach. In particular, we will allow the user to select one of the following four options. The first option is to permit access to the content if there is at least one rule that supports such access. The second is to deny access if there is at least one rule that forbids such access. The third option is to use a majority vote (in case an odd number of rules refer to the particular item). The fourth option is to explicitly ask the user when there are both positive and negative rules. The default will be the second option (i.e. avoid disclosure of a piece of information if there is even a single piece of support in favor of not disclosing it) but again, will it will be possible for users to change this.

A formal description of the USEMP privacy policies in the form of a JSON schema can be found in Appendix 1.

4. Non-automatic Privacy Settings Definition

In this chapter, we present different mechanisms by which a user may be supported by the system in their effort to appropriately define the different components of a privacy policy. In this chapter, we focus on methods that require interaction with the user and are not fully automatic. In the next chapter we present some completely automatic methods. We start with discussing methods for assisting the user to define the audience and then we examine methods for assisting with the definition of the content for some triplet. Moreover, we briefly examine the option of adopting privacy settings from OSN friends. Finally, we look at the problem of personal data value. As will be discussed, personal data value depends on a number of factors that are related to the audience of a user and these factors have a close relationship to the privacy settings of the user. To assist users in better perceiving their audience and therefore the value of their data, we conclude this chapter by presenting a number of audience monitoring tools.

4.1. Assistance with audience definition

In the following, we discuss various ways by which a user may be assisted in selecting the audience for some privacy rule or specific piece of content.

A first option is to develop appropriate user interface mechanisms. In particular, we intend to utilize the first nudge of (Wang et al., 2013) that was mentioned in the previous chapter. This nudge lists, next to each piece of content that was about to be posted, a number of indicative friends that would have access to it, providing a more specific impression about the user's audience (cf. Figure 4 for an example). Importantly, as mentioned, nudges are one of the few tools that can be used for dealing with institutional privacy risks. Therefore, our implementation will also include explicit warnings about such risks.

Figure 4. A nudge that describes who will have access to some content that is about to be posted. Taken from (Wang et al., 2013)

At a second stage, we intend to examine non interface-based types of mechanisms for audience selection assistance. In the following, we list some specific options. These will build on the various audience selection mechanisms described in Chapter 3. As a reminder, the following mechanisms were defined:

- Selecting a single user.
- Manually creating a set of users.

- Defining a set of criteria based on which users are filtered (based on demographics, e.g. attended school, or more loosely defined criteria).
- Automatically grouping friends according to the structure of the ego-network of the user using community detection.

The fourth involves a completely automated process and will be examined in more detail in the next chapter. For the first two, an option is to suggest to the user, after he/she has provided some privacy rule, other users that are also similar either a) to users that he/she has already defined in the rules or b) to users that are related to the content of that rule. This can be achieved using simple textual or visual similarity (cf. related modules in D5.1 and D5.2) to search in the set of the user's friends. It could also involve both the connectivity-friendship patterns between the users as well as the predicted tie strength between users.

Regarding the third mechanism, assistance can be provided by suggesting to the user appropriate keywords or demographic criteria. That is, the most prevalent demographic groups or distinctive keywords in the neighborhood of a user could be identified and be suggested to the user. To take this one step further, a possibility would also be to use the topical profile of friends (e.g. using the topic modelling techniques presented in D6.1) and use these to obtain representative keywords, based on which filtering can be carried out.

4.2. Assistance with content definition

In the following we discuss various ways by which a user may be assisted in selecting the content to which some privacy rule should apply. These are somewhat analogous to those that were proposed for audience definition. As a reminder, in Chapter 3 we identified the following content definition mechanisms:

1. The user can select a single content item.
2. The user can define a specific privacy dimension, attribute or value.
3. The user can define criteria based on privacy scores.
4. The user can define a combination of a privacy dimension, attribute or value and a privacy score criterion.
5. The user can define some keywords or some temporal criteria.

For each of them, there is space for providing the user with valuable assistance. In the first case, a possibility is again to support the user by recommending, after he/she has provided some privacy rule for some specific piece of content, other pieces of content that are similar to the one of the provided rule. This can be achieved in two ways:

- Through a retrieval process that is based on content similarity (textual or visual depending on the type of content), making use of multimedia processing techniques of D5.1 (e.g. Explicit Semantic Analysis) and D5.2 (e.g. copy detection) respectively.
- Through an interactive retrieval process that is based on the privacy scoring framework. That is, once a piece of content is related to some value in the scoring framework, we could suggest to the user to extend the rule to include all content items that are related to that value or to the relevant attribute.

For the second, third and fourth mechanisms the user can be assisted through the visualization of the privacy framework, which is ongoing within T6.3 (a first version of the visualization framework, focusing on the presentation and exploration-browsing of the privacy

dimensions, is provided in D6.3). In particular, values, attributes and dimensions that are either very sensitive or have high privacy scores will be highlighted by the system, so that the attention of the user is drawn to them. In fact, priority will be given to developing mechanisms falling into this category, and then the other options will be considered. The reason for this is that we want to focus on the actual semantics of the content and on inferred content.

For the fifth, just like in the case of audience definition, an option is to pre-extract important keywords or topics and to suggest them to the user. For instance, we could perform topic detection (using the technique presented in D6.1), determine the most salient topics and suggest the most important keywords for these topics. Another option is to determine the most important keywords not for a set of independent topics but for the relevant privacy dimensions, attributes or values.

4.3. Social propagation of privacy settings

As mentioned in Chapter 2, a possibility for assisting users to define their privacy settings is to allow them to import – and then adapt - privacy rules from their friends (Bonneau, 2009). This is a direction of ongoing research and several open issues have been identified:

- Privacy settings may also be a source of information about the user. Thus, users must explicitly share their privacy settings before others can use it. Another option is to collect the most popular rules and suggest them to the user without revealing who created them.
- A significant part of the friends in the social network of some user will not be present in the network of the user to which the rule is transferred. Thus, some rules are not applicable to the new user and should thus be filtered out.
- We need a good way to make the user understand the implications of adopting privacy rules created by some other user. For each of the adopted rules, we need to display examples of pieces of content and users to which those pieces of content will be accessible. Another option is to only show examples of changes when adopting a new privacy rule. That is, who can now see content that before could not and who cannot see it while before he/she could.

4.4. Audience monitoring

In addition to the privacy attributes for developing privacy related assistants, USEMP aims at developing a framework for providing indications of personal data value to end users. As discussed also in D6.1, the USEMP value indicators depend on the user online social behavior as this is captured from a) the OSN itself and b) user online web behavior. In the following, we develop the model for providing value indications to end-users based on their audience and the framework for OSN users to monitor their audience as this develops based on their behavior.

4.4.1. Value model and audience attributes

The approach of USEMP towards providing users with value indications is based on the following model:

- The value of each user in an OSN comes from his dual role as a content consumer and a content publisher.
- The value of each OSN user as a micro-publisher comes from the qualitative and quantitative characteristics of their audience (for example the larger the audience the more the value the user is generating) and how important are the items the user is publishing for that audience.
- As content consumers, OSN users are targeted with ads and marketing campaigns based on their propensity to purchase a certain advertised product. The preferences are either explicitly declared from users in the OSN (for example users declare in Facebook that they are interested in sports) or they can be inferred by four types of behavior tracked by the OSN: a) the behavior of the user in the OSN, b) the behavior of their friends in the OSN, c) the web browsing behavior in sites that use OSN web trackers, d) the behavior of the user in the OSN and the web when targeted with advertisements (which advertisement they like and those which they do not). This data is built into a model that estimates the propensity of a user to consume a certain item (content or advertisement). The higher the propensity of a user to buy some product, the higher the value of that user shall be. Note that methods for inferring consumer interests based on the first two types of behavior (user behavior in OSN and user's friends behaviour in the OSN) were discussed in D6.1.

In the following sections, we will present some tools, each of which aims at enhancing the awareness of the user with respect to the aforementioned factors.

4.4.2. Audience characteristics

In terms of audience characteristics that are shown to OSN users, we have selected to track and estimate two sets of attributes that are used from advertisers to evaluate the importance of a publisher audience: a) demographic data of the audience, b) social graph data related to the importance of user posts in their respective audience.

a. Demographic data

The following demographic data are used to identify the audience of OSN users (these can be extended by future work in WP6):

Location: The registered location of the audience of an OSN user (and their friends).

Age & Gender: The registered age and gender of an OSN user (and their friends).

Interests: The explicitly expressed interest of the audience of an OSN user (and their friends) in a set of predefined list of topics.

Figure 5 illustrates an example of a plot that utilizes the collected data and provides the OSN user with insights on the value of their audience. For example, in this example plot, the OSN user's audience is primarily male (75%) and mostly located around the MENA region (62%), mostly below 35 years of age and largely prefer Rock music (82%).

Figure 5. Example plot depicting demographics data in an audience report for USEMP DataBait users.

b. Social graph data

For estimating the influence of a user to their OSN audience, the influence metrics defined in D6.1 can be used (these can be extended in future work of WP6). These influence metrics are computed to identify which are the most important actions an OSN user has performed for each member of their audience. Using these metrics we can identify:

- The most influential actions the OSN user has performed.
- The most influenced users from the OSN user audience in different distances in the social graph (for example directly connected users (distance=1), or friends-of-friends (distance=2).

Figure 6 presents an example of a plot that utilizes the estimated data pertaining to influence and depicts the most influenced user and the most influential action for that user. In this depiction, the most influenced users for some particular network distance are plotted along with an indication about the most influential action.

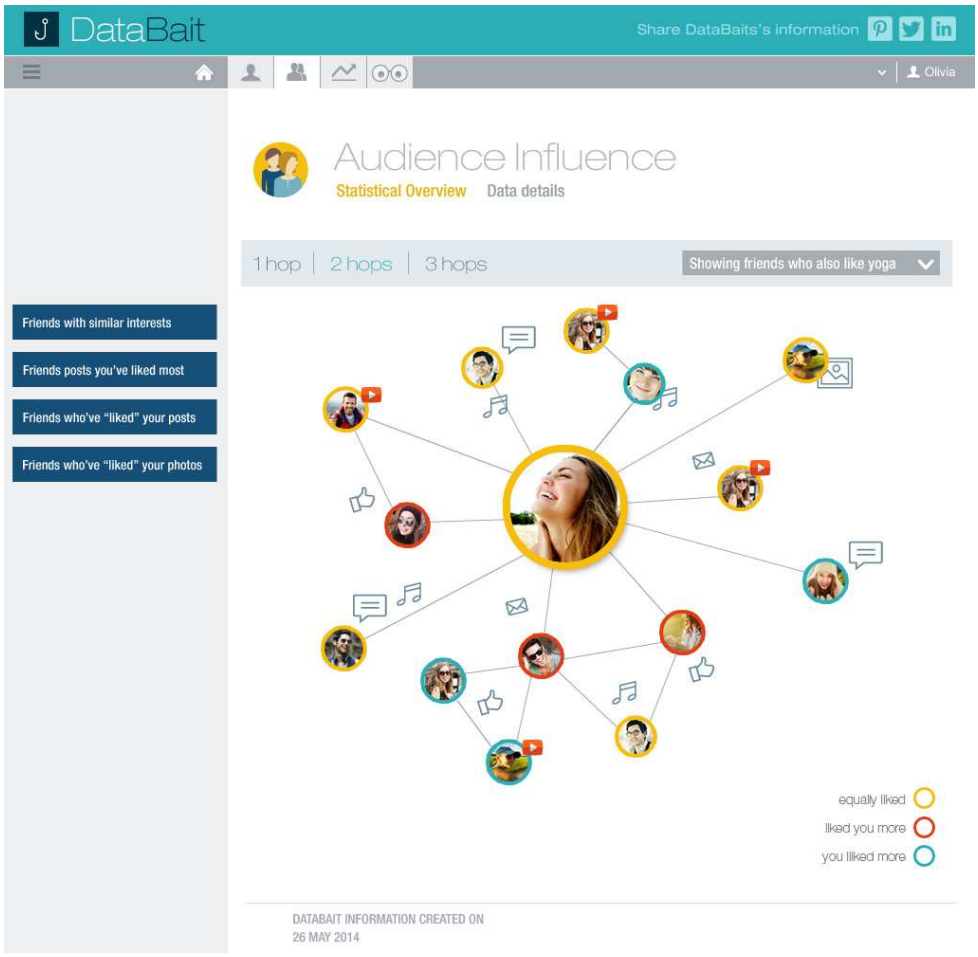

Figure 6. Example plot depicting the importance of OSN user actions to the most influenced members of the OSN user audience.

4.4.3. Propensity model based on content consumption

In addition to the insights provided to the OSN users for their activities as a micro-publisher, insights are also provided about their respective actions as content consumers. In this approach, a vector of user propensity is estimated based on their actions to the OSN and the web. The related data are collected from actions on the OSN, from their friends on the OSN, from web browser behavior, and from their friends' behaviour on the OSN web browser.

Such data are collected, stored and used to estimate the propensity for each user; that is given this amount of information about the user, what is the probability for them to visit a site or perform an action (for example like a page). Initially, for identifying the most probable items to be liked/actions performed by OSN users, an item-based collaborative filtering algorithm based on past content consumption behaviour of the user and their OSN friends will be utilized. Collaborative filtering systems have many forms, but the systems that will be used for USEMP will utilize two steps:

1. Look for users who share the same browsing/OSN actions patterns with the active user (the user whom the prediction is for).
2. Use the OSN actions/browsing behaviour from those like-minded users found in step 1 to calculate a prediction for the active user.

The item-based collaborative filtering computes propensities on an item-centric manner:

1. Determines relationships between pairs of items (OSN actions, browsed websites) by building an item-item matrix.
2. Infers the tastes of the current user by examining the matrix and matching the user's data profile.

This item-based approach is similar to a recommendation to the OSN users of the form: OSN users who browsed/liked *x* also browsed/liked *y*). In terms of visualization, this is still under investigation since the items for a propensity model that includes OSN actions and web browsing can be numerous (hundreds of items daily for frequent OSN and web browsing users), but an initial plot of the higher scoring propensity items along with that propensity will be plotted. Additional techniques for estimating the propensity of the OSN user will be considered in future work of WP6 (for example use of logistic regression and decision tree classifiers).

In the following paragraph we describe in detail a preliminary experiment we performed with synthetic data from OSN users based on the following scenario:

- each OSN user indicates which mobile application he/she has installed through OSN actions (“likes”);
- the item-based recommender computes a list of most probable applications that the OSN user is most probable to like based on the behavior of OSN users with a list of applications they liked similar to his/hers.

This example also demonstrates the relation of such recommendation to the value of shared data from each OSN user. A successful item-based recommender can improve the performance of an advertising campaign for mobile applications (or any other promoted item) by selecting to promote items that the end users are more eager to select. This improvement is based on the data shared from all OSN users, which are used by the recommender to generate the proposed list of applications for each user. In a simple example if there is an increase from 5% to 15% of the applications installed from end users due to the operations of the recommender this will triple the installations of the mobile application promoted for the same amount of money spent from the publisher of the application.

In our experiment, we generated 100 synthetic users with random selection of installed apps from a set of 15 mobile apps that has evolved for 100 epochs (each epoch representing a time of measurement). This synthetic dataset provides us with an indicative behavior of 100 OSN users that have shared their online behavior with respect to “liked” mobile apps for 100 observed moments in time. Our goal has been to experiment with the implementation of the item-based recommender and evaluate the performance of different distance metrics used to measure the distance between items (vectors of installed mobile apps in our case). We used mahout[2], a scalable machine learning library, for the implementation of the experiments).

We have used the following item-based similarity functions for our tests

- Euclidean Distance
- Weighted Euclidean Distance
- Tanimoto Coefficient Distance

[2] http://mahout.apache.org/

- LogLikelihood Distance

More information for the aforementioned distances are available at mahout documentation and in online articles[3]. These metrics define how different items (in this case mobile apps) are close to or far from each other.

For each different similarity function used for the item-based collaborative filtering algorithm we have evaluated the success of each metric based on IR evaluation tests that provide two measures of how well the algorithm performs with each used metric: recall and precision.

The evaluator works by removing the most popular apps from each user and checking if the recommender would include them in the recommendation list. The number of the removed apps from the generated list of installed apps took values between 1 and 10.

For each evaluation two metrics were recorded:

- *precision:* proportion of recommendations that are "good"; a precision score of 1.0 means that every item recommended in the list was valid
- *recall:* proportion of good recommendations that appear in top recommendations; perfect recall score of 1.0 means that all good recommended items were suggested in the list

After running the tests (see Figure 7 and Figure 8), the best performing metrics are the TanimotoCoefficient and the LogLikelihood ones. Overall, we have identified the Tanimoto Coefficient as the most promising metric to use with item-based recommenders that can be used over collected information of historical data from OSN users. Furthermore we have experimented with the mahout framework that will be used further in the project to compute correlations based on OSN user behaviors. The prototype implementation of the experiments along with the synthetic data is made available with this report (cf. Annex 3).

In the next phases of the project, we will experiment further with distributed operations on top of mahout that will allow a scalable operation for large amounts of data collected during the USEMP experiments. These tests will be repeated more extensively once historical data are collected from the pilots and will be extended to include other items that relate to value from OSN actions such as likes to web pages, friends' likes or similar.

[3] http://shanon-shanghai.blogcn.com/articles/mahout-based-recommender-system-introduction.html

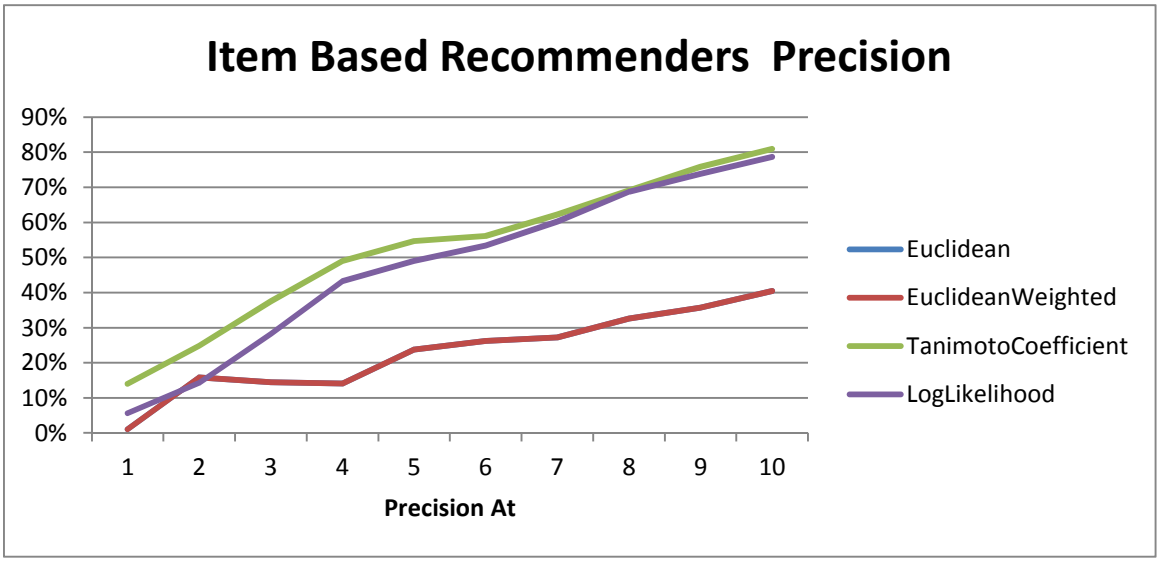

Figure 7. Item-based recommender comparison of different metrics based on precision metric.

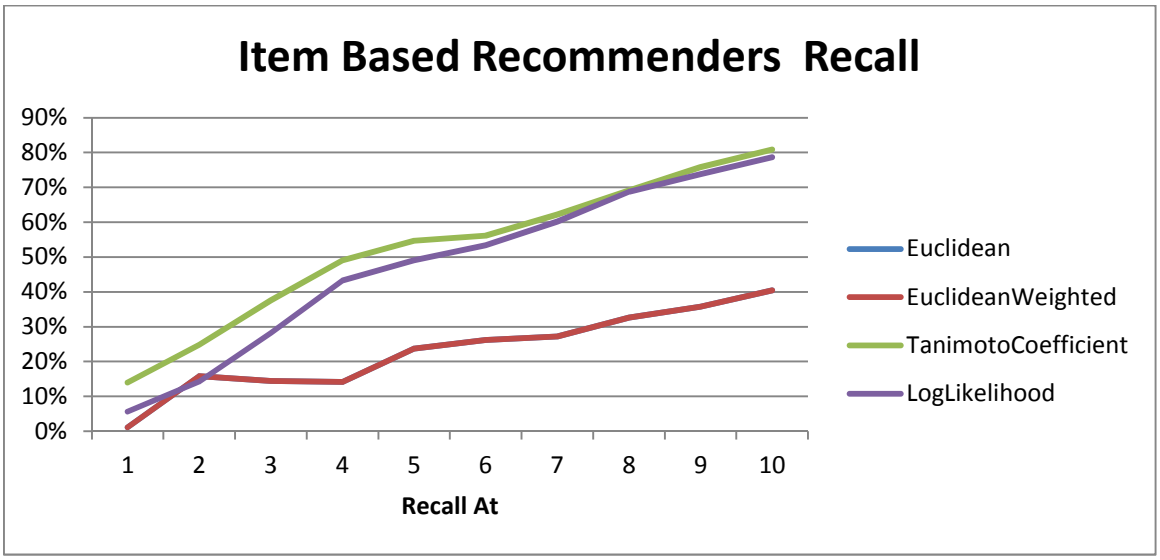

Figure 8. Item-based recommender comparison of different metrics based on recall metric.

5. Automatic Privacy Settings Definition

In this chapter, we present methods for automatically assisting the definition of privacy policies. At a very high level, there are two general types of methods for automatically assisting the definition of privacy policies. The first involves the automatic grouping of the friends of a user and of the content posted. The second involves the automatic prediction of the appropriate access permissions for some piece of content or friend of the user. In the following two sections we will examine a number of methods that fall in these two categories. We will first examine the problem of automatically grouping the friends of a user in appropriate social circles. Please note that we will not further delve into the problem of grouping together content, as it can be considered that this is in fact achieved by the privacy dimensions framework and our inference algorithms (developed within WP5 and T6.1); that is, the privacy dimensions, attributes and values effectively provide a high level grouping layer over a user's content. Subsequently, in the second part of the chapter, we examine the problem of predicting appropriate access permissions for some piece of content.

5.1. Automatic definition of social circles

As discussed in Chapter 2, a number of graph clustering methods have been used in order to group the friends of OSN users. Given the wide range of methods that are available, in the following we proceed to evaluate a number of them on the task of inferring social circles, which directly addresses the requirement for automatic organization of a user's OSN friends in meaningful groups that can be used for improved audience management and definition. Importantly, we also introduce a pre-processing step, which, as will be shown, can improve the quality of the results produced by many of the graph clustering methods.

5.1.1. Overview of graph clustering methods

Let us first review the community detection algorithms that were evaluated.

- OSLOM (Order Statistics Local Optimization Method) (Lancichinetti et al., 2011). This is a popular method that can detect overlapping communities as well as hierarchies of communities. Moreover, OSLOM is able to take into account edge directions and weights. It is based on examining the "statistical significance" of clusters, which is a quantity that reflects how likely it is for a cluster that appears in the graph to appear in a random graph with node degrees distribution similar to that of the examined graph. OSLOM works by applying local search in order to discover structures with high "statistical significance". The source code for OSLOM has been made available by its creators[4] and has been used in our experiments.
- COPRA (Community Overlap PRopagation Algorithm) (Gregory, 2010). COPRA works by performing label propagation, i.e. by assigning some label to each node in the graph and iteratively recomputing it based on the labels of the neighbors of each node. Eventually, after convergence, the labels express the membership of nodes to

[4] http://www.oslom.org/

communities. In fact, COPRA is based on an earlier community detection algorithm (Raghavan et al., 2007), which also utilizes label propagation, but COPRA, extends it so that it is also able to detect overlapping communities. COPRA can be fairly fast on very large networks. An implementation of the algorithm is publicly available[5].

- Louvain (Blondel et al., 2008). This is a method based on the well-known "modularity" score. Modularity is a quantity that compares the density of edges inside communities to the expected density of edges in a random graph with the same number of edges. Louvain attempts to greedily optimize modularity using a two-step process. In the first, modularity is optimized in a local manner, i.e. small communities are built by optimizing modularity around each node. In a second step, these communities are treated as single nodes and are aggregated in larger communities. This two-step process is iteratively executed so that a maximum modularity score is obtained. Moreover, the Louvain method is also able to detect hierarchical community structure and additionally, it has good computational behaviour (it runs in time $O(n \log n)$).
- Girvan-Newman (Newman & Girvan, 2004). This method works by iteratively removing edges from the network. Eventually, the graph becomes disconnected and the connected components of the graph are the communities produced by the method. The edges to be removed are selected based on "edge betweeness", which is a quantity that expresses the number of shortest paths between any pair of nodes in the graph that pass through some edge. Edges with highest betweeness are removed first, as they are more likely to connect different communities. In our experiments, a publicly availably implementation[6] has been used.
- Laplacian Eigenmaps (Belkin & Niyogi, 2003). This method works by performing spectral analysis on the graph for non-linear dimensionality reduction. This provides a representation of the nodes of the graph that may then be clustered using any data clustering algorithm. In short, eigen-decomposition is performed on the Laplacian of the graph and vectors composed of the top eigenvectors are used to produce a new representation of the nodes. Eventually, these new representations are clustered using the k-means clustering algorithm. An existing implementation[7] was used in order to perform the experiments.
- Demon (Coscia et al., 2012). This is another approach that is based on label propagation. Nevertheless, label propagation in Demon is carried out locally, that is, the ego-network of each node is extracted and label propagation is carried out only on it. Subsequently, the communities extracted from the local process are combined to produce the overall clustering result. The implementation provided by the creator of the method[8] has been used to produce our results.
- Clique Percolation Method (Palla et al., 2005). This works by first finding k-cliques, i.e. fully connected sets of at least k nodes. It then joins together adjacent k-cliques (k-cliques are considered adjacent if they share at least k-1 nodes) in order to produce communities.

[5] https://www.cs.bris.ac.uk/~steve/networks/software/copra.html
[6] http://www-rohan.sdsu.edu/~gawron/python_for_ss/course_core/book_draft/_downloads/girvan_newman1.py
[7] https://github.com/MKLab-ITI/reveal-user-classification/blob/master/reveal_user_classification/embedding/competing_methods.py
[8] http://www.michelecoscia.com/?page_id=42

- Structural Clustering Algorithm for Networks (Xu et al., 2007). SCAN is a graph clustering algorithm that was also used in (Jones & O'Neill, 2010) where it was found to be suitable for OSN friend grouping. As discussed in Chapter 2, its main advantage is that it does not only detect communities, but it can also detect hubs (vertices that bridge communities) and outliers (vertices that are only marginally connected to clusters). SCAN works by clustering vertices based on a structural similarity measure that quantifies the similarity of the neighborhoods of two nodes.

5.1.2. Same Social Circle Model (SSCM)

In previous studies that examined the use of community detection algorithms for social circle mining, the input, that is the network of friends around a user, has been used without any preprocessing. Typically, this network of friends represents explicit OSN links between pairs of users, e.g. Facebook friendships. Nevertheless, it is clear that implicit connections are also possible, i.e., a pair of users may belong to the same circle but for a variety of reasons are not connected in the OSN. For example, two users may be members of the same class at school, but they may not be directly connected in the OSN (because they are not aware of each other's presence in the OSN). In some cases, the community detection algorithm may take advantage of common connections to other users and successfully put such pairs of users in the same social circle.

Here, we propose to take advantage of other user features (attributes), if available, and attempt to make such implicit connections explicit before performing community detection. The method that we propose is utilizing what we call the "Same Social Circle Model" (SSCM). The SSCM is a model that predicts whether two users belong to the same social circle. It does this by taking as input the set of per feature similarities between the pair of users. Essentially, it examines how similar two users are with respect to a number of criteria and produces a score that says how likely it is that these two users belong to the same social circle. In some sense, the SSCM is a learned similarity metric between a pair of users. Once learned, the SSCM can be used in order to add "implicit" edges to the network of friends. The SSCM approach may be considered as an instance of the link prediction problem (Liben-Nowell & Kleinberg, 2007).

To make this more explicit, let us assume that a user is represented with the following four features: age, city, university attended, and list of hobbies. Now, the SSCM would take as input four values: the first would indicate the age difference between the two users, the second would indicate whether the two users live in the same city, the third whether they attended the same university and the fourth the number of common hobbies.

In order to train the SSCM model, some training data are required. More particularly, we need both user features, such as those just mentioned, in order to be able to compute the input of the SSCM, but at the same time we need to know some cases of users that do and that do not belong to the same social circle. In the following, we will use part of the ground truth social circles for training the SSCM. It should be noted that of course, we do not use the same ground truth social circles both for training the SSCM and for testing at the same time.

5.1.3. Dataset and experimental setup

Dataset: The evaluation dataset was originally used for the Kaggle challenge "Learning Social Circles in Networks"[9], a competition whose goal was to split the friends of a number of (anonymized) OSN users into appropriate social circles. For each of those users, the dataset provides a list of their friends, anonymized Facebook profiles of each of those friends, and a network of connections between them (i.e. their "ego network"). In total there are 110 ego networks and for 60 of those we have the hand-labelled communities provided by each user. For our clustering methods, we used the latter group of ego networks in order to be able to compare our results with the user hand-labelled data. Moreover, we utilized the features provided for the anonymized Facebook profiles of users. Examples of features provided include the following: birthday, classes attended, degree attended, school attended, year school was completed, family name, gender, location, political views, religious views, work position, employer, etc.

Evaluation measures: Two evaluation measures were used. The first one is the "edit distance". This is the performance measure used by the competition organizers. It represents the minimum number of single edits that are required to change the produced clustering to the ground truth clustering. Clearly, produced clusterings should minimize the edit distance. Each of the following operations cost one edit:

- Add a user to an existing circle
- Create a circle with one user
- Remove a user from a circle
- Delete a circle with one user

For example, suppose our solution contains the following circles for a given user. circle1: {3 1 2}; circle2: {2 3}; circle3: {5 4 6}. Also, suppose that the correct solution is circleA: {4 5}; circleB: {1 3 2 4}. Then the minimum number of required edits would be:

1. Add user 4 to circle1
2. Delete user 2 from circle2
3. Delete user 3 from circle2
4. Delete user 6 from circle3

Thus the proposed solution has an error of 4 edits.

It should be noted that because results will be produced for all 60 networks for which the ground truth is available and because the edit distance can vary significantly between them, we will report the sum of edit distances for all 60 networks. This was also done in the Kaggle competition. The second performance measure that we will use is Normalized Mutual Information (NMI), a typical measure of clustering performance, which quantifies the extent to which two sets of clusters (the reference one and the automatically produced) match with each other (Vinh et al., 2010).

Algorithm settings: Let us now examine some details about the tuning of some of the algorithms that were tested. These settings apply to both the standard case and the case that SSCM is used.

[9] http://www.kaggle.com/c/learning-social-circles

- OSLOM. Our best results were produced by discarding communities with a count smaller than the threshold of 5 nodes.
- COPRA. For each ego network the algorithm is executed 10 times and it computes the overall modularity of each solution in order to decide on the best one to keep. We also tested different values for the maximum number of communities per node, starting from 1 up to 7. Better results were achieved using the value of 2. Again, using a minimum community size of 5 improved the results.
- Louvain. The best results were achieved with a threshold of 3 nodes per community.
- Laplacian Eigenmaps. For this method, there are two main settings: the number of top eigenvectors to extract and the number of clusters for the k-means algorithm. After a thorough test procedure, the best values for our dataset were found to be 19 both for the number of eigenvectors as well as for the number of clusters.
- Demon. We experimented with different thresholds for the maximum number of nodes per community and we observed significant improvement when that value was set to 80 nodes.
- CPM communities. For this algorithm, we found that a clique size of 5 had good enough results. Nevertheless, it should be noted that this algorithm is very demanding in terms of memory consumption and there were issues in its execution.
- SCAN has two main parameters: mu and epsilon. The first denotes the minimum size of nodes in a community and was set to 3. Epsilon is a value from 0 to 1, with a value of 1 denoting that detected communities should be fully connected sub-graphs. In our experiments, it was set to 0.7.
- SSCM. All available features were utilized to learn the SSCM. For each of the networks, we learned the SSCM from the ground truth of other networks and used it to generate new links on it. In particular, we evaluated the SSCM for each pair of nodes in the graph that are not linked in the original network. A Support Vector Machine (SVM) (Cristianini, 2000) was used to learn the SSCM. Importantly, in early experiments it was found that we could achieve an accuracy of 71%. This is clearly not sufficiently accurate, and would thus result in a large number of incorrectly added links. Instead, we use a significantly higher threshold in the SSCM. In early tests it was found that when setting the positive threshold to 0.95 the true negatives rate was 0.99 whereas the true positives rate dropped to 0.13. This means that hardly any edges that should not be added will actually be added, but only 13% of the additional edges that are not present in the network will actually be added. This may not seem very promising, but as the results indicate, results in measurable improvement.

Finally, for those algorithms that can also take into account the weights on edges, we have tested an alternative scheme, in which the weight between a pair of nodes represents number of identical features that the two corresponding users have.

5.1.4. Results

The results of our experiments are presented in Table 1. According to it, the best performing method in terms of both edit distance and NMI is COPRA. We also see a small improvement in edit distance for the weighted version of COPRA over the non-weighted. Second best appears to be SCAN and then Louvain. The other algorithms have significantly lower performance. Another point to note is that the use of the SSCM appears in most cases, but not always, to improve performance.

	Plain networks		Enriched with SSCM	
	Edit distance	**NMI** (mean/std)	**Edit distance**	**NMI** (mean/std)
COPRA	13240	**0.633 / 0.123**	13355	0.607 / 0.161
COPRA with weights	**13215**	0.617 / 0.142	**13222**	0.598 / 0.158
Louvain	14229	0.618 / 0.140	13645	**0.614** / 0.140
SCAN	14212	0.618 / 0.129	13489	0.613 / 0.166
Girvan-Newman	15684	0.477 / 0.209	16136	0.433 / 0.225
Laplacian Eigenmaps	15450	0.572 / 0.119	14444	0.583 / **0.121**
OSLOM	15666	0.582 / 0.148	14134	0.583 / 0.153
OSLOM with weights	15095	0.583 / 0.145	13708	0.583 / 0.163
Demon	16369	0.520 / 0.139	17159	0.493 / 0.174
Demon with weights	16399	0.523 / 0.141	17036	0.493 / 0.174
CPM	18831	0.354 / 0.300	18899	0.343 / 0.298
All friends in one circle	20799	0.070 / 0.057	20799	N/A
Each friend in a circle	29931	0.485 / 0.099	29931	N/A

Table 1. Performance of community detection algorithms on the problem of social circle detection.

Let us also examine some specific results that were produced. We will look at one specific network for a given user. In Figure 9 we can see the ground assignment of friends to social circles for that particular user. In this and the following figures, each user is denoted by a node of the displayed graph and different colors indicate different social circles. It is clear that a friend can belong to multiple social circles. Small grey circles denote users that are not assigned to any social circle. Additionally, grey edges denote explicit OSN connections and come from the original ego-network of the user, whereas red edges denote implicit connections that were added by the SSCM.

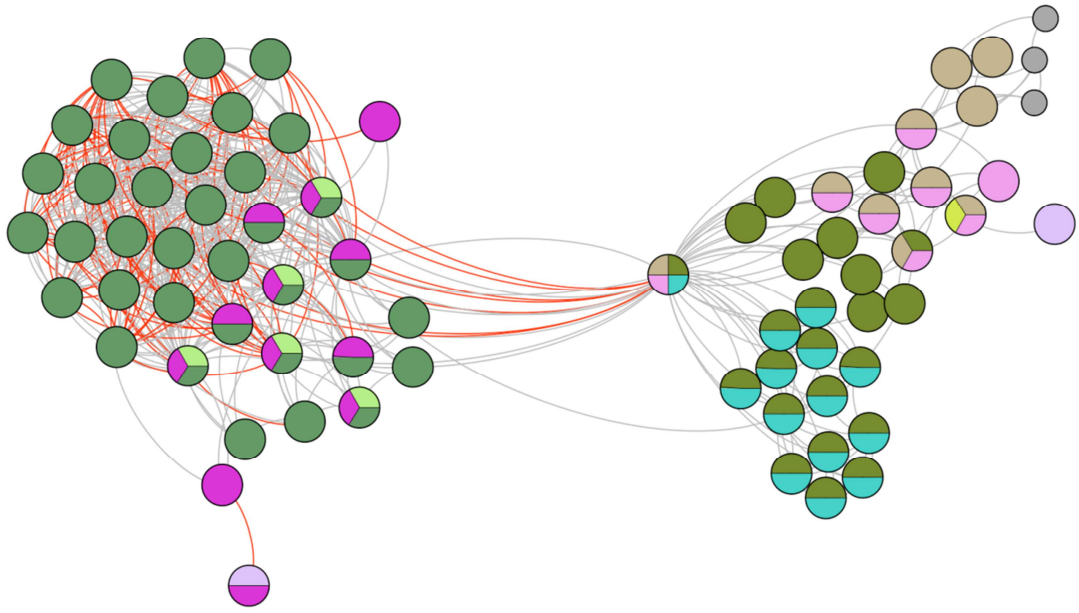

Figure 9. Ground truth assignment of friends to social circles.

Let us now examine the results produced by some of the algorithms. Figure 10 depicts the assignment of friends to social circles as produced by SCAN, without using the SSCM.

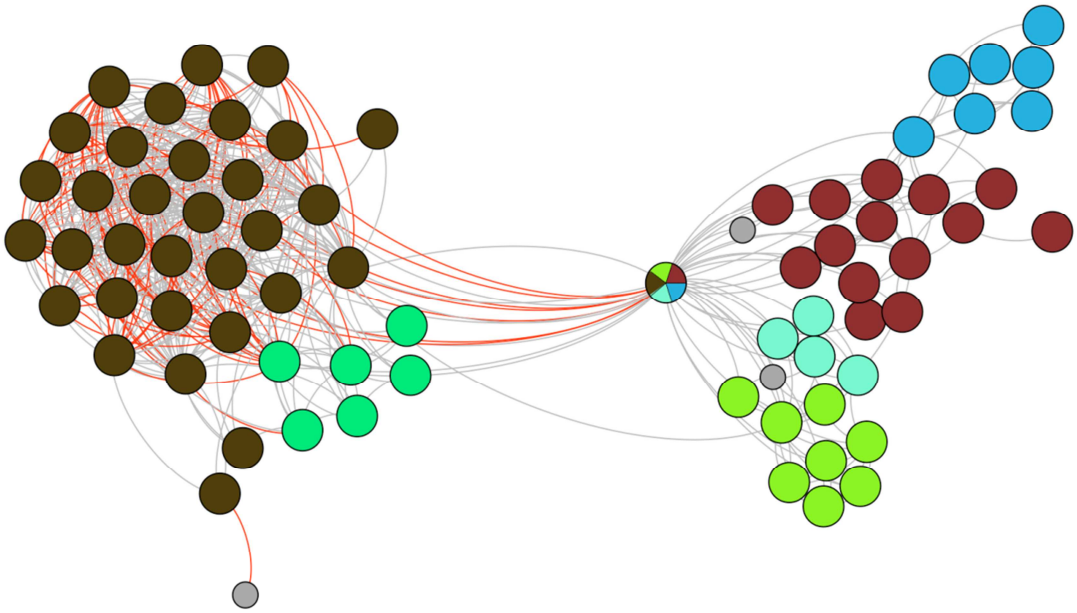

Figure 10. Grouping of friends into social circles as produced by SCAN (no SSCM)

There are two things to note. The first is that SCAN is able to successfully identify hubs. The second is that there are some isolated nodes. Especially, it is noteworthy that the algorithm left unassigned the node near the bottom left. This node is connected to another node through only a single red link. It is reminded that a red link is produced by the SSCM, so, the results in Figure 10 do not really use this link. Let us now see what happens when we feed the same network to SCAN but this time with the additional edges produced by the SSCM. The results can be seen in Figure 11.

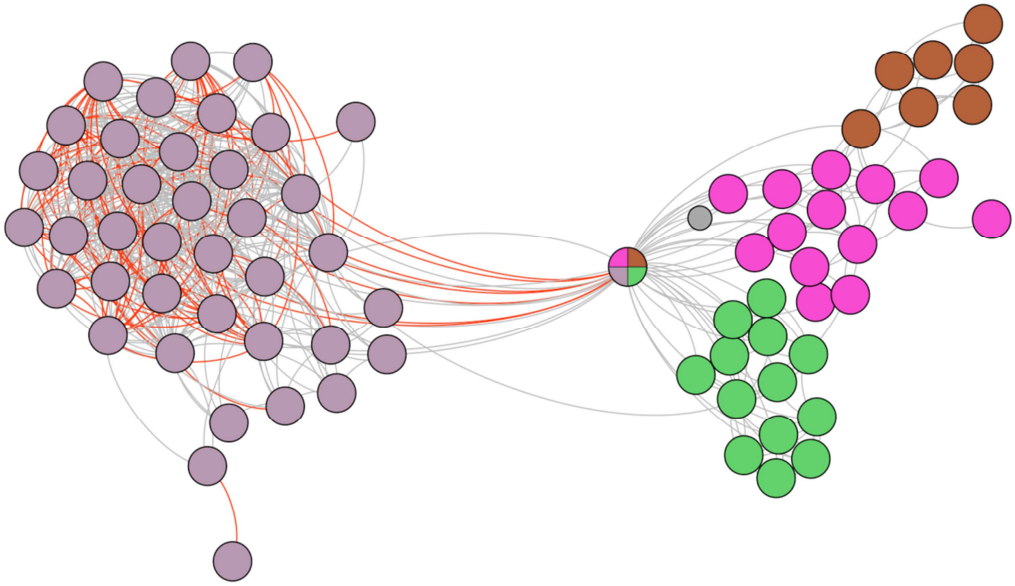

Figure 11. Grouping of friends into social circles as produced by SCAN (SSCM used)

This time, the previously isolated node has been correctly assigned to the big community on the left side of the figure. Thus, the SSCM had a successful effect with respect to that particular case. It can also be seen that the SSCM made the big community on the left much denser than before and now all these nodes are correctly identified as belonging to the same social circle. On the other hand, some information that could be used to infer hierarchical structure (which SCAN is not able to recognize anyway) was lost. Finally, the same results using OSLOM and SSCM can be seen in Figure 12.

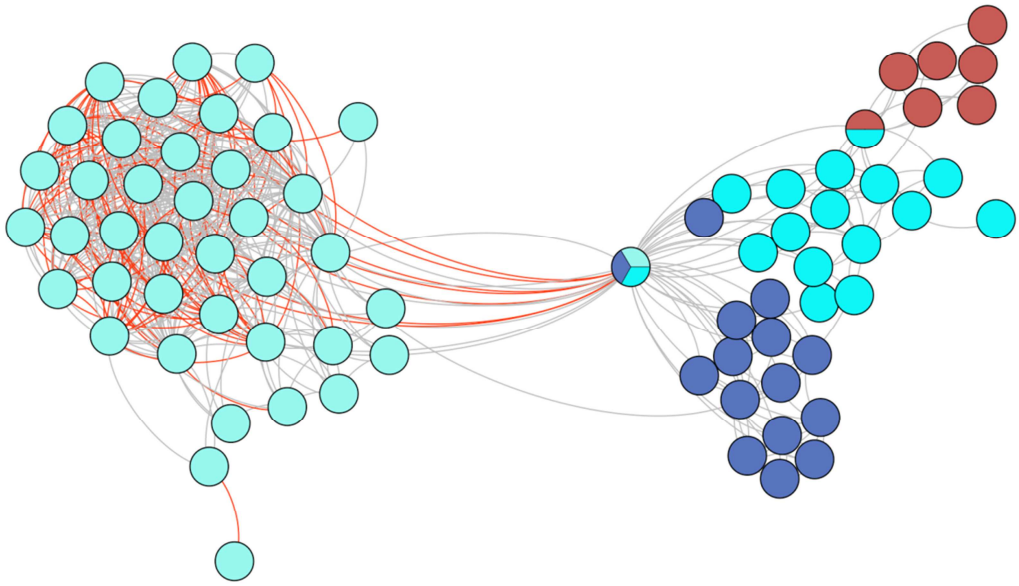

Figure 12. Grouping of friends into social circles as produced by OSLOM (SSCM used)

Again, the isolated node at the bottom left has been correctly grouped. OSLOM also seems to be able to identify some nodes that belong to the same social circle at the same time.

This report is accompanied by prototype implementations and data that can be used to reproduce these results.

5.2. Example-based privacy settings prediction

We now turn to the problem of automatically computing the appropriate access permissions for some particular piece of content. In fact, this problem can appear in three different forms. In the first, computation is carried out specifically for some piece of content. In the second, it is focused on some specific friend of the user. Finally, in the third, it is focused both on some specific piece of content and on some specific friend. Clearly, the third allows for control at the finest level. For instance, it could be used as follows. Suppose that a user posts a new piece of content. Then, a series of predictions could be computed, providing the access rights for each of the user's friends. Realizing such a scenario may be feasible, provided that an appropriate training set is available. Unfortunately, for the moment at least, we do not have such data. When the pilot studies progress, relevant data may become available and the possibility of realizing this scenario will be reconsidered. Instead, we focus here on the first option, i.e. to predict access permissions for specific pieces of content. The second option may be considered equivalent to "tie-strength" prediction, a problem that was briefly discussed in Chapter 2.

5.2.1. Dataset and experimental settings

The experiments presented here follow closely those of (Zerr et al., 2012). In fact, the dataset that we use is based on the one used in their study. In short, the authors present a series of experiments in which images uploaded to Flickr are classified as either “private” or “public”. The images have been annotated manually by users (not the original owners) and for each of them a number of textual features are available: title, description and tags. The authors also use a number of visual features extracted from the images: detected faces, color histograms, edge-direction coherence vectors, SIFT features, brightness and sharpness. They performed a number of classification tasks, each time using a different set of features. They provide precision-recall diagrams and their main performance metric is the Break-Even Point (BEP), the point where precision equals recall. Using all visual features at the same time they achieved a BEP of 0.74. Using only textual features they achieved a BEP of 0.78 and using both textual and visual features they achieved a BEP of 0.80.

In the following, we utilize part of their dataset in order to investigate the possibility of automatically predicting appropriate privacy settings for some specific content. The set of textual features was kindly provided to us in order to experiment. Unfortunately, only part of the images and the set of visual features were still available and it was therefore not possible to utilize the visual content. Out of the images that came with textual data, 4,665 of them were labeled as private and 26,375 were labeled as public. In the original experiments, a random subset of the public images was sampled, in order to balance the number of private and public images that were used for training and testing. Moreover, 60% of the images from each class were used for training and 40% for testing. We did the same and ended up with 2,799 private and 2,799 public images that were used for training, as well as with 1,866 private and 1,866 public images that were used for testing.

In order to get a feeling about the complexity of the task, let us examine the textual features for some randomly selected private and public images. A list appears in Table 2. Examining the list, one can see that for some of the pictures it may be possible to identify some words that could potentially indicate the presence of sensitive content. For instance, a couple of private images contain the word “kid”, but some contain no such features.

For learning the classifiers we used the Weka LibSVM[10] implementation of Support Vector Machines (Cristianini, 2000) and a standard Bag of Words approach (Manning, 2008). Importantly, we examined the use of two preprocessing techniques: stemming and lemmatization, while we did not consider in our vocabulary words that appeared less than five times in the corpus.

[10] http://weka.wikispaces.com/LibSVM

Example metadata from images of private nature
Description: - *Title: thinker* *Tags:* swimming pool canon G9 light boy shadow kid portrait underwater water bubbles'
Description: We're in the La Salle Room, drinking a toast to Seattle's first “working lady”, Nellie. *Title:* Charles Finkel of Pike Place Brewery *Tags:* lesacooks.com Brewery The Pink Hobart Pike place'
Description: - *Title:* - *Tags:* kids burlington cal btv vermont playing'
Description: - *Title:* Fox with style *Tags:* Fox Flying'
Description: - *Title:* arriving at the forts *Tags:* THCC sea kayak Tower hamlets canoe club shivering sands wind farm'
Example metadata from images of public nature
Description: - *Title:* no city limit *Tags:* depth of field horseshoe orlando scenery florida lakeland chelsea renay country nature'
Description: - *Title:* house (space bathroom) *Tags:* buildings interior'
Description: My brand new Kindle 2. a gift from my beloved wife. The world in 2010 is going to be exciting. A beginning of a new decade filled with hopes and promises. Unlimited space for creativity and reflection on human greed might add a touch of wisdom to our relentless forward-looking risk-taking attitude.I made this image yesterday to express the transition to the future - to mark the boundary between past and future. The magazine in the back is the Annual Issue of The Economist reflecting on how the current year will be. Its a juxtaposition of the now and what will be? *Title:* The World in 2010 *Tags:* eReader 2010 future the economist gadget save trees Amazon electronic Kindle 2 technology ebook reader the world in 2010 Kindle'
Description: This is the main entrance of Umm Haram's mosque in Larnaca. *Title*: The Passage *Tags*: larnaca multicultural pictures muslim islam minarets cyprus holy places mosque'
Description: *Title:* Favorite Things *Tags:* retro vintage aqua kitch'

Table 2. Textual features for some randomly collected private and public images.

5.2.2. Results

The achieved results are presented in Figure 13. As it can be seen, there are no big differences between the three curves. The red curve, which is the precision-recall curve when lemmatization is used, is only just a bit higher than the other two at some points. This is also indicated in BEP. The BEP in the case that no pre-processing is performed is 0.7518. When stemming is used it drops to 0.7449 and when we use lemmatization it increases to 0.7529. Moreover, it should be noted that in all cases the accuracy rate for public images was somewhat higher than for private images. The achieved performance is encouraging, since it indicates that the "correct" privacy setting can be predicted for the majority of images. For the cases, where the prediction fails to capture the correct result, the user would need to manually correct the respective setting. Given the current state of things, where a default privacy setting is used for content items, it is obvious that this solution already constitutes an important improvement. Moreover, we expect that incorporating additional features in the classifier would result in further performance gains for the approach and hence would render it even more suitable for use with the USEMP system.

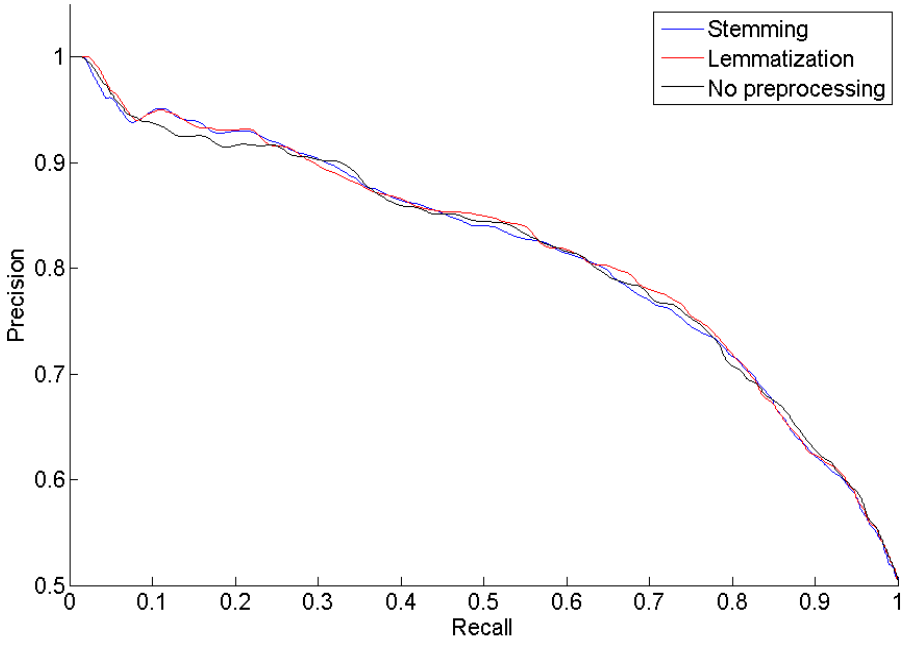

Figure 13. Precision-recall curve for the privacy prediction task on images

Code and data in order to reproduce the above experimental results are provided with this deliverable.

6. Web Trackers and Do-Not-Track Policies

In the previous three chapters we presented the USEMP privacy setting model, as well as a number of automatic and non-automatic methods that support USEMP users in defining their privacy policy. Nevertheless, the tools that we develop in USEMP consider not only the presence of users in OSNs, but also privacy issues related to their web browsing behavior. Browsing behavior, just like the information posted by a user to an OSN can reveal personal information. Furthermore these two types of information can be combined to provide additional insights to user online behavior. Therefore privacy policy tools that apply to the browsing behavior of the user, in addition to those related to its OSN presence are also required. Web trackers are entities that are able to monitor the browsing behavior of users. In this chapter we will discuss the background of web trackers, Do-Not-Track policies and a suggested permission model that will be implemented by the USEMP web tracking tools.

6.1. Web trackers and third-party cookies

A Web tracker is an entity that records user web behavior (pages that have been visited, links that have been followed, and actions that have been taken on websites). The enabling technology behind web tracking is the support from web browsers of a feature referred to as HTTP cookie.

HTTP cookie as a tracking technology

An HTTP cookie is a piece of data sent by a website to a web browser and stored there. The website receives back the value of that piece of data for each subsequent webpage request. This feature is used to implement record keeping of various entities related to an individual user's browser history.

Figure 14 exposes the basic mechanism of HTTP cookie tracking. Initially, the client-browser requests a page from a web server. The server returns the content of the web page and an HTTP cookie named session-id with value 12345. In each subsequent request for additional pages, the client includes the session-id cookie variable with its value assigned previously from the server. This feature allows the web server to identify that these requests are coming from a distinct client/web browser (the one with the value 12345 assigned initially).

Figure 14. HTTP cookie tracking mechanism between a client (web browser) and a server.

HTTP cookie security model: Each HTTP cookie is associated with the domain of the entity (web service) that sets the cookie initially and a path of that entity (for example root path is /). These two attributes define the scope of the cookie for the web browser: the web browser returns the cookie value only to requests from the same domain and path (there are additional attributes for a cookie like “Max-Age” and “Expires” that we do not examine here since these are not used by the USEMP tools). HTTP cookies are classified into two types depending on the domain of the website that sets the cookies initially:

- *first-party cookies:* in this case, the domain of the web service that sets the cookie and is eligible to read it is the same with that of the website (as in Figure 12).
- *third-party cookies:* in this case, the domain of the web service that sets the cookie and is eligible to read/receive it is different to that of the website.

First-party cookies are typically used to implement internal services required for the functionality of a website (for example the personalized pages for an e-commerce site or the internal analytical tools for evaluating the performance of a site).

Third-party cookies are mostly used from advertising and marketing networks to collect information for users’ browsing history for the purposes of improved targeting and optimization of their campaigns. We refer to such organizations as “web personal trackers”. Those are the primary object of permission actions for the proposed USEMP web trackers permission framework in the following sections.

Third-party JavaScript libraries: In addition to the use of third-party HTTP cookie tracking, trackers also utilize JavaScript libraries. For example, in the case of advertising, such JavaScript libraries are provided by the advertising networks and are included in websites from their web developers to retrieve advertisements.

During the registration of the website with the advertiser, a unique identifier that is stored in the advertiser servers is provided to the web developers, so that it is used along with the third-party JavaScript library to retrieve advertisements for that website. This unique identifier used by the website for its request for advertisements allows the ad network to identify the site from which the request for an ad was sent. This unique site identifier is also used for counting the number of ad displays and other related business metrics in some website.

Third-party JavaScript libraries can also collect a variety of information from the web browser (e.g., page title, current time, or the characteristics of the browser). More details on how these capabilities offer an additional set of challenges will be discussed later in this chapter.

An example of how web personal trackers function is depicted in Figure 15, where the role of the web tracker is taken by an advertising network with domain “adnetwork.com” (could be similar to the Google AdWords online advertising network). In this example, we assume the owners of site-1 and site-2 have already registered with the adnetwork.com to receive advertisements and they have included the provided third-party JavaScript library using the provided unique identifiers from the advertising network.

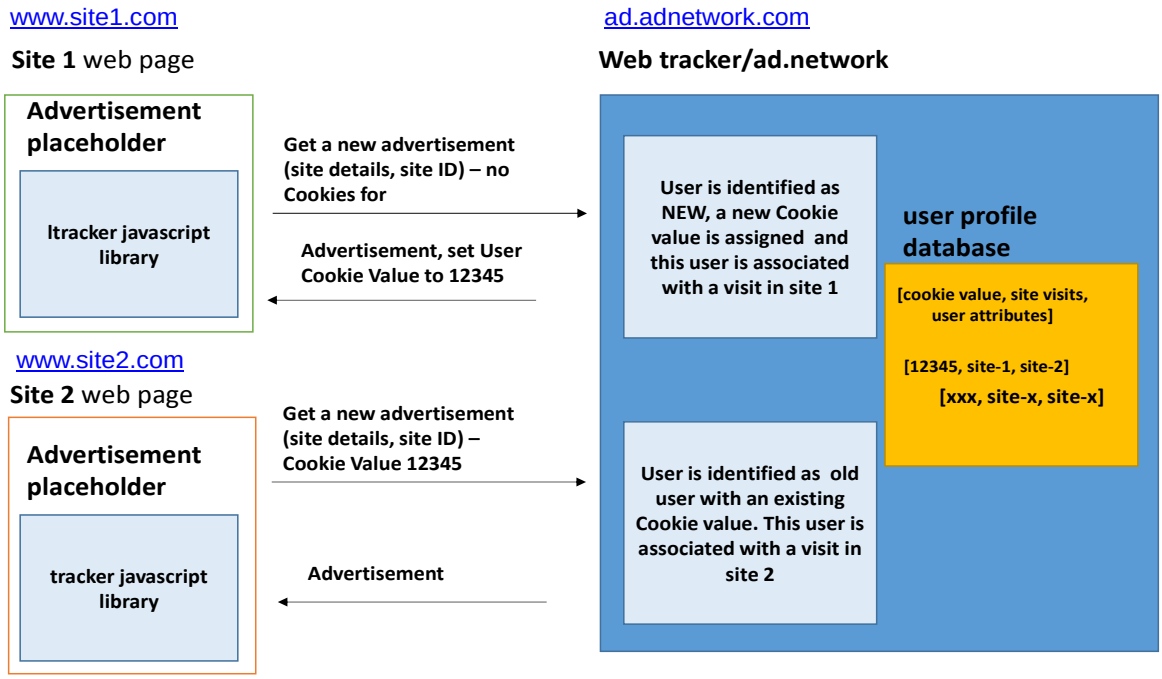

Figure 15. HTTP third-party cookie tracking example of a user visiting two sites and the related operations in web trackers backend.

In this case, let us assume a user visits site-1 at its URL www.site1.com that includes an online ad from ad.adnetwork.com for the first time. The downloaded ad sets a cookie for the advertiser domain (ad.adnetwork.com). The advertising network will register that a user with the specific cookie value has visited www.site1.com (the site is identified by the data passed from the JavaScript library calls to the advertising server for requesting the advertisement). After that visit, let us assume the user visits another website www.site2.com that includes an advertisement from the same advertising network ad.adnetwork.com. In this case the request for the advertisement will include the cookie with the same value set from the advertisement in site1. This way, the advertiser can detect that the same user has visited both sites.

Modern browsers have some coarse-grain features that allow users to control the behaviour of third-party cookies:

- Users can delete all the cookie values at any point in time.
- Users can block all the cookies at any point in time.
- Users can block all the third-party cookies at any point in time.
- Users can block the execution of JavaScript.

These permissions are controlled by the end user through the web browser settings for each individual web browser they use and for each terminal they use and, although useful, they apply universally, not allowing the end-user to differentiate the blocking behaviour for each potential web tracker. For example some web trackers may be offering an interesting service that some users may wish to have access to (such as a personalized recommendation service for items they may be interested in purchasing), while other web trackers simply collect data without offering users' transparency on their use.

The USEMP web trackers permissions model described later offers the ability to build more sophisticated digital assistants (driven by end-user behaviour or some data processing) that offer end-users finer-grain control and transparency about which third-party JavaScript libraries and related web trackers are loaded and active for each webpage that they visit.

6.2. Do-Not-Track Policies

An additional tool that is available on the web for developing digital assistant tools around web browsing privacy and transparency is that of the HTTP header field DNT. This is one of the proposed W3C specifications on how end-users and web browsers can indicate that they do not wish to be tracked.

This feature allows a client-browser to send along with each HTTP request sent to a web server or web tracker an HTTP header variable named DNT with one of the following values and semantics for the behavior of the web server receiving the request with respect to tracking:

- The DNT header is sent and has value 1 in case the user does not wish to be tracked (opt out) from the specific web tracker.
- The DNT header is sent and has value 0 in case the user wishes to be tracked (opt in) from the specific web tracker.
- The DNT header is not sent in case the user has not expressed a preference. The behavior of the web tracker then depends on its internal default logic

The feature is implemented in popular web browsers allowing end users to either turn on or off DNT by changing its related header value (the control option as implemented in Chrome is depicted in Figure 16).

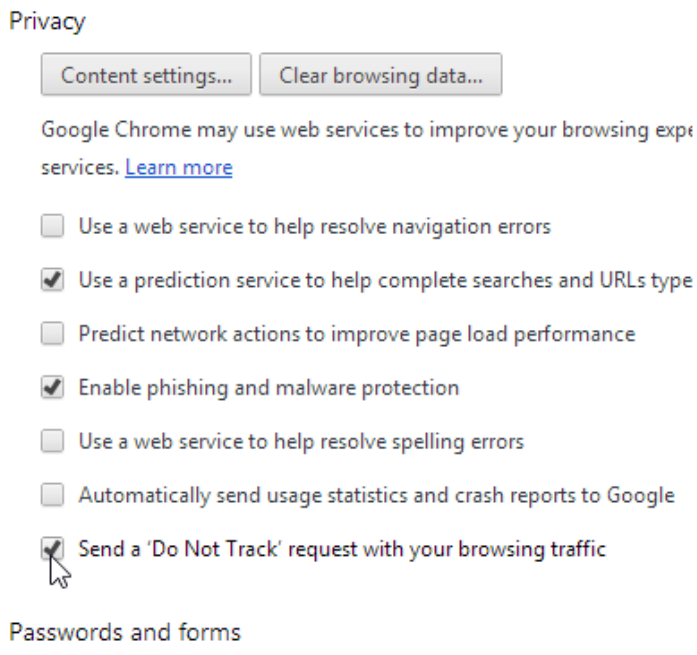

Figure 16. DNT option implemented in Chrome popular browser.

The drawback of the DNT feature is that it has not yet been widely implemented from web trackers and does not have the wider support from the online advertising industry (there has recently been a discussion[11] after the Mozilla foundation decided to implement the DNT feature with a default value of 1 for the popular Mozilla browser, a decision that has been reverted since).

[11] http://www.businessinsider.com/apple-mozilla-blast-advertisers-in-do-not-track-fight-2013-7

Related, and in addition to DNT, there is one additional specification from W3C, named Privacy Preferences Project (P3P), that has been defined as a protocol to enable websites to declare the intended use of information collected by them or by web trackers to web browsers and thus to the end-users. The semantics of the protocol are described in detail in the P3P1.1 specification[12].

Unfortunately due to the lack of incentives for its adoption by website publishers and web trackers, this tool has also not received wide implementation by web browser vendors (it is currently implemented only by Internet Explorer).

The proposed USEMP permissions model for web browsing behavior includes provisions for the support of DNT features, so that as these become more important among web trackers and web publishers, they can be supported from the USEMP tools.

6.3. USEMP web online trackers model

Based on the two sets of capabilities that modern web browsers offer, described in the previous paragraphs, the USEMP permission model for online web browser tracking behavior is defined with the following objectives:

- Allow the creation of digital assistants that help users increase their understanding on which entities collect their online behavioral data.
- Help users control, in an intuitive manner, which of these entities may receive online behavioral information.

It is expected that the proposed framework can be used to develop: a) user-driven digital assistants that allow the end users to fully control which web tracker entities can track them b) semi or fully automated digital assistants that can take into account the behavior of the end users with respect to privacy settings for web trackers and provide the necessary recommendations.

6.3.1. USEMP online tracking permissions framework

The following principles constitute the USEMP tracking permission framework:

1. *Web trackers identification model:* defines what a tracker is and how it can be identified.
2. *Web trackers data model:* defines the type of information that needs to be stored for each web tracker and how it can be represented at the collection level (across pages and sites).
3. *Web tracker action model:* defines the type of actions that can be applied to a tracker from end users.
4. *Web trackers user personal and social graph information:* defines the type of information on the user profile and social graph that is stored along with web trackers users' action information.
5. *Web trackers visualization:* defines a set of basic principles on the minimum type of visual representations of web trackers to the end user.

[12] http://www.w3.org/TR/P3P11/

USEMP web trackers identification model

In USEMP, we consider as a web tracker the URL that hosts each third-party library that is loaded in a web page. As third-party library URL, we define the URL of the web tracker server that includes that library in a web page, which is different to the domain of the website.

A USEMP web tracker can be identified by: a) listening for remote connections performed by the web browser to third-party domains (other than the current page) to load third-party JavaScript libraries, b) by examining the Document Object Model (DOM) of each web page loaded in the web browser and searching for URLs that load JavaScript libraries.

USEMP web trackers data model

For each identified USEMP web tracker, the following information can be stored and retrieved to facilitate its further processing from privacy digital assistants:

1. *Website information:* which site or web page the tracker has been identified in.
2. *Time information:* when the tracker has been identified.
3. *Context information:* web browser type, device information, client IP and location where the tracker has been identified.
4. *Web tracker information:* this is an additional set of data that can be retrieved from third-party sources and can provide additional context to help digital assistants. For example, this can include the domain and IP information of the web tracker, its geolocation according to their IP address, privacy related information (examples of such information can be: tracker relation to well known cases of privacy violations, a privacy-friendly index, their group membership or their affiliation; for example, such information are maintained by entities like http://disconnect.me/ - an open source repository with information about trackers)

USEMP web tracker action model

For each identified USEMP web tracker it should be possible to:

- Block any requests from the web tracker per individual page.
- Allow (unblock) any requests from the web tracker per individual page.

This action can be enforced by the following list of alternative means:

1. Cancelling/enabling the loading/operation of the web tracker JavaScript library on a web page.
2. Cancelling/enabling requests to the domain of the web tracker in the browser for a given web page.
3. Sending DNT header appropriate values (opt in/opt out) to the web tracker for request from a given web page.

It should be possible to define:

1. The default policy for all trackers, if the user takes no action (block/unblock), for each web page.
2. Lists of web pages with different predefined behavior (block/unblock) from the default.
3. Users should be able to override the dictated behavior for the default policy or the policy by defined lists.

USEMP web trackers user personal and social graph information

It should be possible also to maintain personal information (anonymized) and social graph information (from OSN) for each user so that digital assistants can develop automated tools that will propose/apply different personal policies.

USEMP web trackers visualization

In terms of visualizations we define two recommendations with respect to the two forms of visualization that should be supported at minimal by digital assistants adhering to the USEMP framework. These two types of visualization are suggested to ensure that end users, in addition to any other visual representation they may have access to, can always get access to one of the following two suggested basic views: a) list view that provides a list of trackers and their respective information, b) graph view that provides a graph representation of trackers related to the user (for example a graph where trackers are depicted as nodes to a graph of a page visited by the user) with access to all collected information.

6.3.2. USEMP web online trackers tools and permissions network

The planned USEMP experiments offer the opportunity to implement and test digital assistant tools for online privacy settings based on the suggested USEMP web online trackers permissions framework. In the first iteration of the experiments, a user-driven digital assistant tool will be implemented as part of the USEMP DataBait toolkit.

The form of the digital assistant that will be tested during USEMP pilots will include:

- a web browser extension (referred to as add-on) for a popular web browser.
- a web application that can be accessed by any browser.

DataBait online privacy settings digital assistant - web browser add-on

In recent years, web browsers have become more modular and programmable. In that respect, modern web browsers have allowed for the creation of micro-applications (referred to as extensions) that have access to a web page once it is loaded to the web browser and can perform a number of actions (defined as the extension context). More importantly, these extensions can be made available to the non-technical web user for use/installation from marketplaces (such as the Chrome Web Store or Firefox Add-on marketplaces). A typical browser extension architecture is depicted in Figure 17.

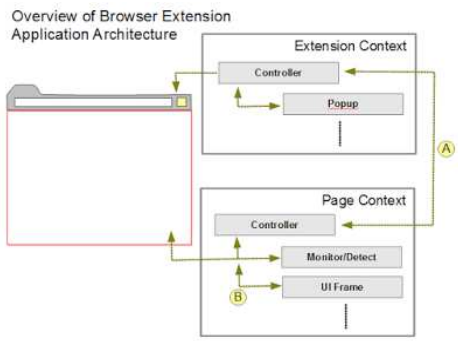

Figure 17. Browser extension architecture used in USEMP DataBait privacy settings digital assistant.

The web browser add-on that is developed as part of the DataBait digital assistant for privacy settings allows for:

1. Collecting web trackers information.
2. Enforcing actions selected by the end users with respect to web trackers – initially the prevention of third-party library loading will be used.
3. Experimenting with web browser add-ons UI/UX limitations.

As a platform of choice, the Chrome web browser has been selected; the reason being that it is one of the most popular browsers in Europe (40% of online users use Chrome in Europe according to Figure 18[13].

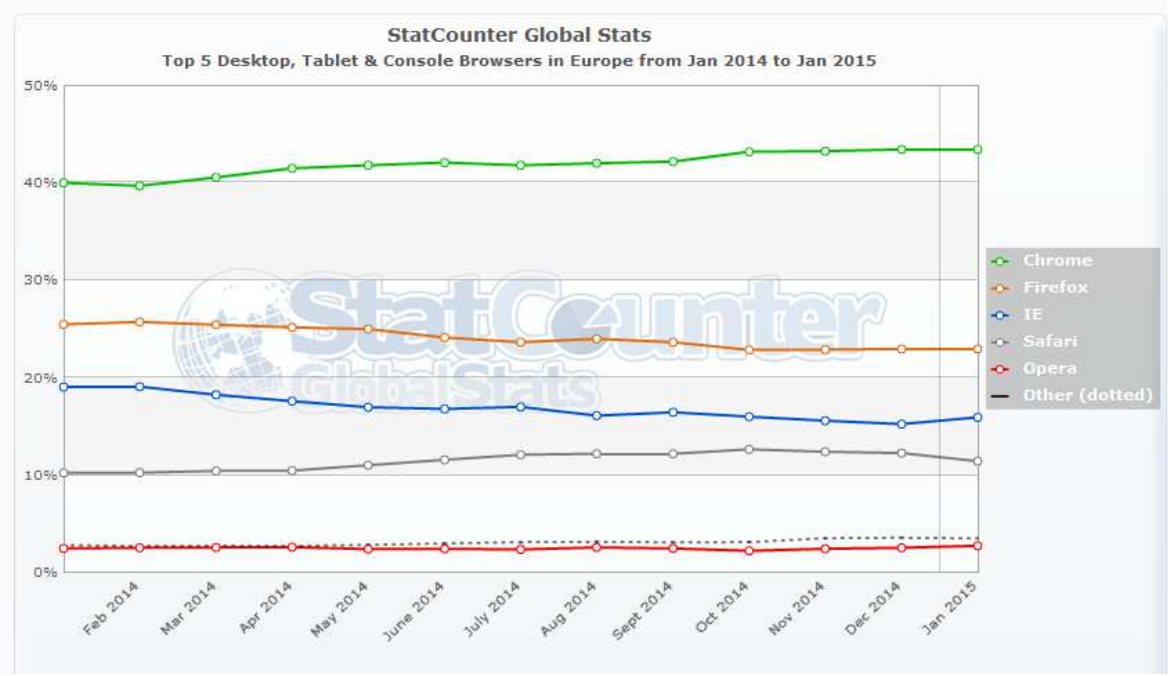

Figure 18. Most popular web browsers statistics for Europe for 2014-2015 from statcounter.com

DataBait online privacy settings digital assistant - web application

In addition to the web browser add-on, a web application will also be developed as part of DataBait, so that:

- Users can access/view the information collected from the web trackers.
- Users can select suggested actions they would like to apply to web trackers.

While the DataBait web application can not apply the suggested actions, these can be communicated to the developed web browser extensions that can act on them. Furthermore, in terms of UI/UX design, the DataBait web application does not comply to any limitations that apply to web browser extensions and thus can be used to experiment with a richer interface for the digital assistant visualizations.

[13] Taken from http://gs.statcounter.com/#browser-eu-monthly-201401-201501

6.3.3. Future challenges for web tracking policy tools

In the previous paragraphs, we analyzed the methods that are currently used to track online users (third-party HTTP cookie tracking and third-party libraries) and presented the USEMP web trackers permission framework that can be used to develop online behavior privacy settings digital assistants. Here, we will discuss a set of developments that are expected to change the online behavior of end-users in the forthcoming years. These developments are expected to present a number of challenges to the proposed methods and need to be considered for future work:

- *Use of mobile devices for online browsing:* Increasingly more users' online traffic is expected to originate from mobile devices; that is, smartphones or tablets. While mobile devices have capable mobile web browsers that allow their users to browse online, these are currently much more monolithic and much less programmable than their desktop-based counterparts are. Currently, there are no popular mobile web browsers that will allow the development of web browser extensions. Thus, a different enforcing tool has to be developed / used for mobile devices.
- *Use of mobile applications and fingerprinting:* More and more users are expected to use mobile applications instead of mobile browsers. While smartphone vendors have taken a number of measures to ensure that a user can control their identification for targeting purposes, sophisticated tools have been developed that collect different traits of the devices and perform probabilistic matching to identify users without the use of HTTP cookie mechanisms. This is a different set of trackers, focusing on mobile environments, and requires a drastically different permission control mechanism that does not resort to control of HTTP cookies or third-party JavaScript libraries loading.
- *Wearables & Internet of Things:* In the next few years wearable devices operating in the Internet of Things domain will provide additional sources of information that can be used for digital fingerprinting/identification of users. This is an additional challenge for online privacy permissions frameworks that are to be defined to be effective in such an emerging environment.

7. Conclusions and Future Work

This deliverable introduced the first version of the USEMP privacy setting framework and its prototype implementation, including a set of tools that assist the user to perceive and control various aspects of their online presence, both within an OSN and in the web in general.

The USEMP privacy settings framework was presented first. This entails a set of privacy policies defined by the user to control access to their OSN data. It uses the privacy facilities provided by the OSN and effectively implements a much more flexible and expressive privacy settings model. Moreover, we discussed a number of mechanisms, automatic and non-automatic, by which users can be assisted to tune the privacy settings framework.

An additional set of tools focused on personal data value. In particular, we examined tools that present a) a variety of audience characteristics, b) data related to the influence of content and audience members and c) content consumption, knowledge about which can improve the understanding of the user and the value of their personal data.

Finally, a set of tools concerns privacy not in the context of an OSN, but with respect to web browsing behavior. We have examined the issue of web trackers and presented the USEMP tracking permission framework.

Research and development in the upcoming period will be carried out at two levels. The first involves integration and development. As the pilots are approaching, we will complete the preparation of modules that are responsible for the realization of the privacy setting framework. Moreover, the presented formulation of the privacy setting framework is expected to be adapted according to feedback that we will receive from the pilots.

At a second level, work will focus on open research questions and alternative approaches:

- We will further explore the challenges arising from the problem of social propagation of online privacy settings (Section 4.3).
- We will work on enriching our work on collaborative filtering for assisting the user to perceive their personal data value (Section 4.4).
- We intend to further experiment with methods for social circle detection (Section 5.1). In particular, we have seen that using the same social circle model significantly improved the results; it would be interesting to examine how we could improve such a model in order to further improve the quality of the produced social circles. Moreover, the benefits of the use of the same social circle model indicate that taking into account the features of users and not only their OSN connectivity can prove quite useful; thus, we intend to explore alternative ways of taking user features into account. For instance, we could utilize user features in order to create a preferential attachment model that could then be used for enriching the existing graph.
- We plan to explore, in collaboration with WP5, new ways for classifying content as private or public (Section 5.2). This work could include the extraction of visual and latent features or experimentation with different classifiers and feature selection mechanisms.

Annex 1 – USEMP Policy Representation

In the following, we present a set of JSON schemas for representing the privacy policy of some user. The designed schema will allow the fast retrieval of relevant rules in order to decide on the access rule for some user. The first thing to note is that we would like to reuse audience and content definitions. That is, if for example an audience set has been identified by some rule, then this set may be reused in other rules as well and similarly for content definitions. Thus, we will define separate schemas for records that represent content and audience definitions for some user. The first one, representing the definition of an audience set is listed in Table 3.

```
{
  "type":"object",
  "$schema":"http://json-schema.org/draft-03/schema",
  "name":"audience_record",
  "description":"This is a record that represents a particular audience definition, a set of users defined by one of
                 the mechanisms listed in Chapter 3”.,
  "id":"http://jsonschema.net",
  "required":false,
  "properties":{
    "user_id":{
      "type":"number",
      "description":"This links this particular audience record to some user in the database",
      "id":"http://jsonschema.net/user_id",
      "required":true
    },
    "audience_record_id":{
      "type":"string",
      "description":"This field should be unique in the set of audience definition records for some user. It will be
                     used by the triplet record to refer to the particular audience set",
      "id":"http://jsonschema.net/audience_record_id",
      "required":true
    },
    "list_of_friends_ids":{
      "type":"array",
      "description":"This array contains pointers to the ids of the friends that belong in the particular audience
                     definition.",
      "id":"http://jsonschema.net/list_of_friends_ids",
      "required":true
    },
    "demographic_criteria":{
      "type":"string",
      "description":"This is a string representing the demographic criteria that was used to match the particular set
                     of users.",
      "id":"http://jsonschema.net/demographic_criteria",
      "required":false
    },
```

```
    "retrieval_criteria":{
      "type":"string",
      "description":"This is a string representing the textual retrieval criteria, keywords, that were used to match
                    the particular set of users.",
      "id":"http://jsonschema.net/retrieval_criteria",
      "required":false
    },
    "produced_automatically":{
      "type":"boolean",
      "description":"This field defines if this record represents an audience set that has been produced by an
                    automatic community detection algorithm. If yes, then the fields demographics_criteria and
                    retrieval_criteria should not be present. Also, please note that at most one of the fields
                    demographics_criteria and retrieval_criteria should be present in any case“,
      "id":"http://jsonschema.net/retrieval_criteria",
      "required":true
    }
  }
}
```

Table 3. JSON Schema for audience definition.

It should be noted that for fast retrieval, regardless of the mechanism by which the particular audience record has been created, the set of ids of friends is kept explicitly in the record. One could argue that when the set has been formed by some retrieval or demographic criteria, the set of friends' ids need not be kept, but only the criteria should be kept. Nevertheless, this may require repeated execution of the retrieval process and may be computationally costly.

A similar design principle is used for the schema that represents the content definition records, which is listed in Table 4.

```
{
  "type":"object",
  "$schema":"http://json-schema.org/draft-03/schema",
  "name":"content_record",
  "description":"This is a record that represents a particular content definition, a set of pieces of content defined
                by one of the mechanisms listed in Chapter 3.”.,
  "id":"http://jsonschema.net",
   "required":false,
  "properties":{
    "user_id":{
      "type":"number",
      "description":"This links this particular audience record to some user in the database",
      "id":"http://jsonschema.net/user_id",
      "required":true
    },
    "content_record_id":{
      "type":"string",
      "description":"This field should be unique in the set of content definition records for some user. It will be
```

```
                    used by the triplet record to refer to the particular content set",
      "id":"http://jsonschema.net/content_record_id",
      "required":true
    },
    "list_of_content_pointers”:{
      "type":"array",
      "description":"This array contains pointers to the content that belong in this particular content definition",
      "id":"http://jsonschema.net/list_of_content_pointers",
      "required":true
    },
    "retrieval_criteria":{
      "type":"string",
      "description":"This is a string representing the textual retrieval criteria, keywords, that were used to match
                    the particular set of content.",
      "id":"http://jsonschema.net/retrieval_criteria",
      "required":false
    },
    "privacy_framework_aspect":{
      "type":"string",
      "description":"This is the privacy dimension, attribute or value that is used as a criterion for this content set”
      "id":"http://jsonschema.net/privacy_framework_aspect",
      "required":false
    },
    "privacy_framework_criterion":{
      "type":"string",
      "description":"This is a string that represents the scoring criterion that was used to produce this content set.
                    For example, it could be something like ‘sensitivity>0.5‘ ”
      "id":"http://jsonschema.net/privacy_framework_criterion",
      "required":false
    }
  }
}
```

Table 4. JSON Schema for content definition.

It should be noted that none of the fields “retrieval_criteria”, “privacy_framework_aspect” and “privacy_framework_criterion” need to be present. If none of these are, then the content set will have been defined manually. Moreover, “privacy_framework_aspect” can be present without “privacy_framework” being present and the other way around accounting for the second and third content definition mechanisms.

We also provide a schema for the record type that represents a privacy triplet. This is further specified in Table 5.

```
{
  "type":"object",
  "$schema":"http://json-schema.org/draft-03/schema",
  "name":"triplet_record",
  "description":"This is a record that represents a single privacy rule, in the form of a triplet, as described in
                 Chapter 3.",
  "id":"http://jsonschema.net",
   "required":false,
  "properties":{
    "user_id":{
      "type":"number",
      "description":"This identifies the user to which this privacy rule applies ",
      "id":"http://jsonschema.net/user_id",
      "required":true
    },
    "audience_record_id":{
      "type":"string",
      "description":"This points to a specific audience definition record which mush already be present in the
                     database. ",
      "id":"http://jsonschema.net/audience_record_id",
      "required":true
    },
    "content_record_id":{
      "type":"string",
      "description":"This points to a specific content definition record which mush already be present in the
                     database. ",
      "id":"http://jsonschema.net/content_record_id",
      "required":true
    },
    "access_level":{
      "type":"string",
      "description":"This can be one of ‘allow’ and ‘deny’. ",
      "id":"http://jsonschema.net/access_level",
      "required":true
    },
    "action":{
      "type":"string",
      "description":"This can be one of ‘untag’ and ‘delete’. ",
      "id":"http://jsonschema.net/action",
      "required":true
    }
  }
}
```

Table 5. JSON Schema for the basic privacy triplet.

Finally, we have a single record for each user which is simply used to store the default policy. The schema for this is listed in Table 6.

```
{
  "type":"object",
  "$schema":"http://json-schema.org/draft-03/schema",
  "name":"user_record",
  "description":"This is a record that is used for storing the default access level for some user”.,
  "id":"http://jsonschema.net",
   "required":false,
  "properties":{
    "user_id":{
      "type":"number",
      "description":"This identifies the user to which this default access level applies ",
      "id":"http://jsonschema.net/user_id",
      "required":true
    },
    "default_access_level”:{
      "type":"string",
      "description":"This can be one of ‘allow’ and ‘deny’. ",
      "id":"http://jsonschema.net/default_access_level",
      "required":true
    }
  }
}
```

Table 6. JSON Schema for a single user's core privacy setting.

Annex 2 – Facebook Privacy Settings

In this Annex, we look in more detail at Facebook privacy settings. We first describe the overall privacy settings model of Facebook and then look at specific options that are available through the web UI and the Graph API. The discussion below is based on version 2.2 of the Graph API.

Facebook privacy settings model

Facebook has developed an interaction model that is based on the concept of a social graph (the model has been released from Facebook as the Open Graph Model, and is available at: http://ogp.me/ and free to use for development of other social networks or applications).

In this model, each user has a list of friends they are connected to, their own private profile with personal information, and actions that can be applied by Facebook users in: a) their own personal pages (referred to as the user's wall), b) the personal pages of their friends (friend's walls) that have provided access to them in some form. A user can also "be-friend" a page or a Facebook application. The input from all the latter is collected and shown to the end user in the form of their News Feed page.

In addition to individual users, Facebook supports the creation of pages and applications from third parties. For example: a) a large brand like Coca Cola will have its own Facebook page https://www.facebook.com/cocacolagr (which actually corresponds to a local branch of the company), b) a game such as Candy Crush will have its own page as well https://apps.facebook.com/candycrush. For third-party applications such as the previous one, Facebook provides a permission framework to allow users to control which data they share with them.

The Facebook privacy settings model is based on the premise that it can act as a safe repository for three types of its users' data:

- Demographic data (name, home country, telephone number, email, religion)
- Explicitly defined interests (for example foreign languages, hobbies)
- Social graph related data
    - List of connected friends
    - List of actions on the social graph:
        - Status updates, comments, photos, video on user's private wall/timeline
        - Posts, comments, likes in other Facebook friends' walls (through the news feed)

Facebook itself can have access to all user attributes and information for the purposes of its operation and the advertising/marketing tools that are built on top of its platform.

The available permission settings provide Facebook users means to control which of their personal data can be shared to and interacted with from third parties (public users, Facebook friends, third-party applications and Facebook pages). In all cases Facebook utilizes the following type of access to personal data:

- *Public:* data that can be shared and accessed by anyone.
- *Friends:* data that is shared and accessed by all friends of the user.

- *Friends-of-friends:* data that can be shared not only with friends but also with their friends.
- *Custom lists:* data that is shared only to specific private groups.
- *Explicit access requests:* data that is explicitly shared with Facebook application only if the user provides their consent.

In the following sections we describe how these permissions can be set through the Facebook API and through Facebook applications.

Facebook privacy settings for user profile data via Facebook UI

Privacy settings in Facebook can be defined either through the Facebook website or through native Facebook applications running in mobile phones and tablets. Here, we will examine the settings interface that is available through the website. Settings are accessible through the settings menu. There are two categories of settings related to privacy settings: a) Privacy, b) Timeline & Tagging.

In the “Privacy” section (see Figure 19) the user can define the following settings: 1) limit who can view their shared activity onwards (public, friends, only the user, custom lists of friends), 2) limit the audience of past posts (public, friends, friends-of-friends). It is also possible to determine a limitation about who can contact them and who can search for them through Facebook Search using their email or phone number (some of this information is required during Facebook registration).

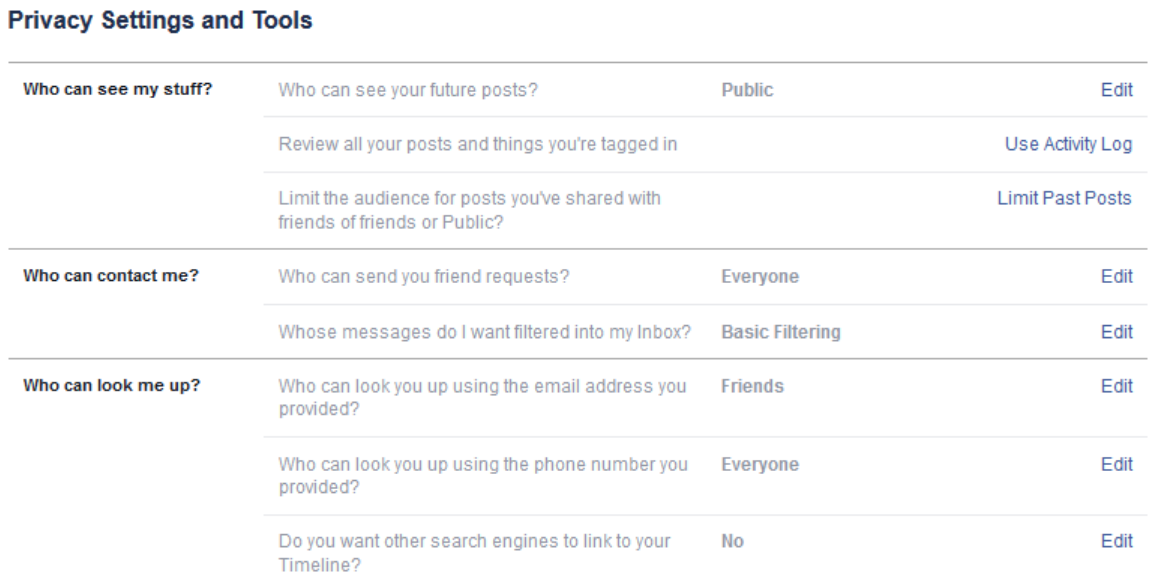

Figure 19. Facebook UI, Privacy settings that users can define in Facebook (desktop web).

In the “Timeline & Tagging” section (Figure 20), the user can control how other users can interact with the items posted in their Timeline page. A user can limit who can post on their Timeline page and can ask to review tags in which other people in Facebook mention them.

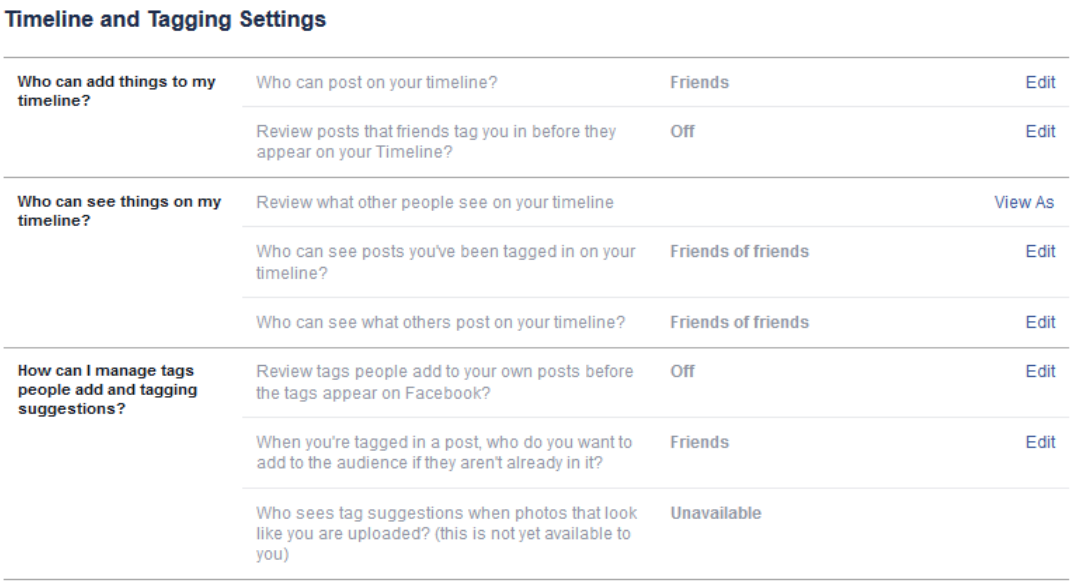

Figure 20. Timeline & Tagging Settings for Facebook application (desktop browser).

In addition to these two related sections in Facebook pages, the user is also offered a ”Privacy Shortcuts” option that provides the users with a set of links to the most important Privacy settings (as perceived by Facebook) and an option to run a Privacy Check-up. The “Privacy Check-up” (Figure 21) option provides the user with an overview of their privacy settings they have setup for the most important information they are sharing and the possibility to change them.

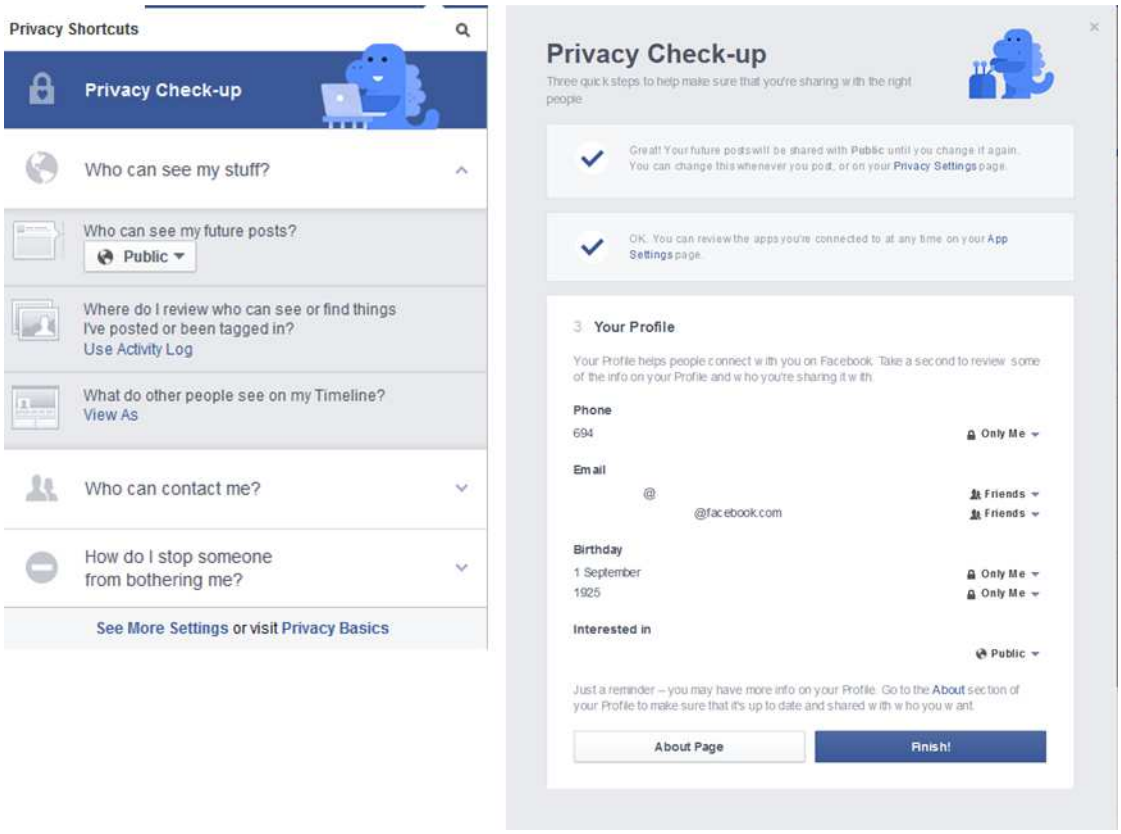

Figure 21. Facebook UI, Privacy shortcuts and Privacy Check-up Wizard with User Profile data showing user current settings.

Facebook Graph API and privacy settings for digital assistants

In addition to the tools offered through the web UI, Facebook also offers a permission framework and an API to programmatically control all social graph items and develop third-party applications. The Facebook data sharing permission framework allows the developer to develop third-party applications based on the data that users decide to share with them. This development framework is referred to as the Facebook Graph API and we will be presenting its features related to privacy digital assistants and its limitations in the following.

Facebook Graph API Permissions

For a third-party application to access users' data, it is necessary for the application to request the user to login to the application using their Facebook credentials. The first time the user logs in they will be presented with a dialogue controlled by Facebook, requesting from the user to confirm that they allow the third-party application to access their personal data. Third-party applications can request different types of permissions. At a very high level there are two classes of permissions that the application can request: permissions that do and permissions that do not require review from Facebook. In the following, we will examine the permissions under each of these classes.

Facebook Permissions That Do Not Require Review

The following are data access permissions that do not require a review from the Facebook team before they are requested from the applications developers.

Public profile (default) permissions: this set of permissions includes reading of some basic attributes about the person, which are part of a person's public profile on Facebook.

Application friends: this optional permission grants your app the ability to read a list of friends of the user who also use the application.

Email permissions: this gives the application access to the person's primary email address.

Facebook Permissions That Require Review

The following are data access permissions that require review from Facebook, before these are requested from the end users. These are generally reviewed within three business days and there are some guidelines from Facebook on what is permissible (see also below paragraphs on limitations of use of the Facebook Graph API). Some permissions may take up to seven days to review, and are marked as such in the reference.

Extended profile properties: these permissions are all sensitive properties that may or may not be part of a person's public profile.

Extended permissions: these include the most sensitive pieces of profile information. One of these permissions is publishing stories to a person's face profile. All extended permissions appear on a separate screen during the login flow so a person can decide if they want to grant them.

Open Graph permissions: these permissions are for gaining access to any Open Graph data stored in someone's profile.

Page permission: this allows users to administer any Facebook Pages that they manage.

Figure 22. Facebook permissions framework UI for requesting access to use data from third-party applications (desktop web)

Once appropriate permissions have been established, the following actions can be applied to related items through the Facebook Graph API (latest v2.2 is assumed):

Posts: the following are the feasible/non-feasible actions for Facebook posts[14]:

- **read** the post if the post is public or if the **read_stream** permission has been granted;
- **publish** a new post controlling its privacy setting (the post can be available to everyone, all friends, friends-of-friends, private, or to a custom list of friends) and the tags it includes (it requires **publish_actions** permission if a custom interface is used in the app);
- **delete** a post, if it has been created by the same Facebook application (this requires **publish_actions** permission);
- it is not possible to **update** a post through the Graph API.

Status: the following are the feasible/non-feasible actions for Facebook status updates[15]:

- **read** the status if it is public or if the application has the **read_stream** permission;
- **publish** a new status update (it requires **publish_actions** permission if a custom interface is used in the app);
- **delete** a status update if it has been created by the same application (it requires **publish_actions** permission);
- it is not possible to **update** a status through the Graph API.

Comments: the following are the feasible/non-feasible actions for comments[16]:

- **read** the comment provided the application has access to the object the comment is attached to;
- **publish** a new comment (it requires publish_actions permission if a custom interface is used in the app to the object the comment is attached to);
- **delete** a comment the user has created (it requires publish_actions permission);
- **edit/update** a comment through Graph API (it requires publish_actions permission).

[14] https://developers.facebook.com/docs/graph-api/reference/v2.2/post
[15] https://developers.facebook.com/docs/graph-api/reference/v2.2/status
[16] https://developers.facebook.com/docs/graph-api/reference/v2.2/comment

Photos: the following are the feasible/non-feasible actions with respect to photos[17]:

- read some photo if it is public or if the user_photos permission has been granted, read a photo a user has been tagged to;
- publish a new photo (it requires publish_actions permission if a custom interface is used in the app) with its privacy related attributes;
- delete a photo the application has created (it requires publish_actions permission);
- it is not possible to update a photo through the Graph API.

Photo Tags: the following are the feasible/non-feasible actions with respect to photo tags[18]:

- **read** the tag requires **user_photos** permissions for the photos in which a user has been tagged;
- **publish** a new tag (it requires **publish_actions** permission if a custom interface is used in the app) with its privacy related attributes (Tag Review which if enabled will require a tag review from the end user);
- it is not possible to **delete** a photo tag through the Graph API;
- **update** a photo tag (it requires **publish_actions** permission).

Videos: the following are the feasible/non-feasible features with respect to videos[19]:

- **read** the video if it is public or if the application has been granted the **user_videos** permission, read a video a user has been tagged to;
- **publish** a new video (it requires **publish_actions** permission if a custom interface is used in the app) with its privacy related attributes (location);
- it is not possible to **delete** a video through the Graph API;
- it is not possible to **update** a video through the Graph API.

Limitations of Facebook development framework for privacy digital assistants

There are some limitations that apply to the access of personal data even for permissions that require a review process by Facebook. The following are the most important ones for the purposes of USEMP.

read_stream restricted access: this permission provides access to read all the posts in a person's News Feed, or the posts on their Profile. This type of access is currently restricted to applications building a Facebook-branded client on platforms where Facebook is not already available. For example, Android and iOS apps will not be approved for this permission. In addition, Web, Desktop, in-car and TV apps will not be granted this permission.

Clearly though, access to all posts of a user would be required by a privacy settings assistance tool. It is possible to get access to a person's New Feed and post if the user accepts to become a test user for your application, which will allow the development of the DataBait related tools and experimentation during the pilots.

[17] https://developers.facebook.com/docs/graph-api/reference/v2.2/photo
[18] https://developers.facebook.com/docs/graph-api/reference/v2.2/photo/tags
[19] https://developers.facebook.com/docs/graph-api/reference/v2.2/video

publish_actions review access: this type of permission allows an application to explicitly publish content to Facebook from within an app with utilizes a customised UI (for example inform user of potential privacy leaks before they publish). This permission is required for our digital personal assistant so that the user is able to post through a custom interface that informs them of potential privacy leaks of their content. It is also required to be used if the user is expected to use the digital assistants to perform "corrective" actions to their shared data (for example delete a post). This is because third-party Facebook applications such as digital personal assistants can have access only to posts that have been posted through the application. While this permission is not constrained to a certain type of actions Facebook does retain the right to review the benefit such custom dialogues bring to its end users.

In terms of customized dialogues, DataBait tools can circumvent any informational constraints by developing part of the personal digital assistants for privacy as a web browser add-on rather than a Facebook application. This option will be explored further in USEMP experimentations to understand better what limitations it may have in terms of UI/UX for the personal digital assistants.

Annex 3 – Prototype Implementations

This document is accompanied by a number of prototype implementations. More details about each of them are provided in the following.

Community detection algorithms for Social Circles detection

The first prototype implementation puts together the community detection algorithms that were tested in Section 5.1. In particular, we provide implementations of the following algorithms:

- COPRA
- Louvain
- SCAN
- Girvan-Newman
- Laplacian Eigenmaps
- Demon
- CPM

Moreover, we provide scripts for batch testing of the algorithms on the data that were used to perform our tests. Some of the implementations were developed by us, whereas some were obtained from existing packages. Please see the ReadMe file in the package for details about how to run the different algorithms.

Classification of images as private or public

We provide code and data for training and testing the classifier that was used to identify private or public images that were presented in Section 5.2. Please see the ReadMe file in the provided package for details about how to run it.

Collaborative filtering (item-based) tests

The first prototype implementation for estimating how personal information can be used to infer preferences useful for recommendations, as discussed in Section 4.4.3, was developed using mahout (and deployed on an HDFS cluster). The prototype implementation in the form of the required Java functions is provided in a zip archive along with the synthetic data used and the necessary instructions to run these on a Hadoop/HDFS cluster.

Representation of the privacy settings model

Finally, we provide in separate JSON files the schemas for the privacy settings model that were presented in Annex 1. The set of schemas can be found in the files:

- audience.json
- content.json
- privacy_triplet.json
- user.json

References

Adu-Oppong, F., Gardiner, C.and Tsang, P. Social Circles: Tackling Privacy in Social Networks. Symposium on Usable Privacy and Security (SOUPS). 2008

Belkin, M., and Niyogi, P. Laplacian eigenmaps for dimensionality reduction and data representation. Neural computation 15, no. 6 2003.

Benferhat, S., El Baida, R. and Cuppens, F. A Stratification-Based Approach for Handling Conflicts in Access Control. SACMAT. 2003.

Blondel, V., Guillaume, J., Lambiotte, R. and Lefebvre, E. Fast unfolding of communities in large networks. J. Stat. Mech. 2008.

Bonneau, J., Anderson, J. and Church, L. Privacy suites: shared privacy for social networks. Proceedings of the 5th Symposium on Usable Privacy and Security (SOUPS). 2009

Carminati, B., Ferrari, E. and Perego, A. Rule-Based access control for social networks. Proceedings of the 2006 international conference on On the Move to Meaningful Internet Systems. 2006Danezis, G. Inferring Privacy Policies for Social Networking Services. Proceedings of the 2nd ACM workshop on Security and artificial intelligence. ACM, 2009

Cristianini, N. and Shawe-Taylor, J. An Introduction to Support Vector Machines and other kernel-based learning methods, Cambridge University Press, 2000.

Coscia, M., Rossetti, G.. Giannotti, F. and Pedreschi, D.. Demon: a local-first discovery method for overlapping communities. In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 615-623. ACM, 2012.

Egelman, S., Oates, A. and Krishnamurthi, S. Oops, I Did it Again: Mitigating Repeated Access Control Errors on Facebook. SIGCHI Conference on Human Factors in Computing Systems, 2011

Dev, H., Eunus Ali, M. and Hashem, T. User Interaction Based Community Detection in Online Social Networks. Database Systems for Advanced Applications. Lecture Notes in Computer Science Volume 8422, pp 296-310. 2014.

Fang, L. and LeFevre, K. Privacy wizards for social networking sites. In Proceedings of the 19th international conference on World wide web (WWW '10). ACM, New York, NY, USA, 351-360, 2010.

Fogues, R., Such, J. and Espinosa, A. BFF: A tool for eliciting tie strength and user communities in social networking services. Information Systems Frontiers 16.2: 225-237. 2014.

Fortunato, S. Community Detection in Graphs. Physics Reports, 486 (3-5). 75-174. 2010

Gilbert, E. and Karahalios, K. Predicting Tie Strength with Social Media. In Proceedings of the 27th International Conference on Human Factors in Computing Systems. 2009.

Gregory, S. Finding overlapping communities in networks by label propagation. New Journal of Physics 12, no. 10 2010

Hart, M., Castille, C., Johnson, R. and Stent, A. Usable Privacy Control for Blogs. International Conference on Computational Science and Engineering, 2009. CSE '09.

Hu, H., Ahn, G. and Jordensen, J. Detecting and resolving privacy conflicts for collaborative data sharing in online social networks. Proceedings of the 27th Annual Computer Security Applications Conference. ACM, 2011.

Johnson, M., Egelman, S. and Bellovin, S. Facebook and privacy: it's complicated. Proceedings of the 8th Symposium on Usable Privacy and Security. 2012.

Jones, S. and O'Neill, E. Feasibility of structural network clustering for group-based privacy control in social networks. Proceedings of the 6th Symposium on Usable Privacy and Security. 2010.

Kawase, R. Nunes, B. , Herder, E., Nejdl, W. and Casanova, M. Who wants to get fired? Proceedings of the 5th Annual ACM Web Science Conference. 2013.

Kelley, P., Brewer, R., Mayer, Y., Cranor, L. and Sadeh, N. An Investigation into Facebook Friend Grouping. Human-Computer Interaction – INTERACT 2011. Lecture Notes in Computer Science Volume 6948, pp 216-233. 2011

Klemperer, P., Liang, Y., Mazurek, M., Sleeper, M., Ur, B., Bauer, L., Cranor, L., Gupta, N. and Reiter, M. Tag, you can see it!: using tags for access control in photo sharing. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, 2012.

Lancichinetti, A,, Radicchi, F., Ramasco, J. and Fortunato, S.. Finding statistically significant communities in networks. PloS one 6, no. 4 (2011): e18961.

Lujun, F. and LeFevre, K. Privacy wizards for social networking sites. Proceedings of the 19th International Conference on World Wide Web. ACM, 2010

Liben-Nowell, D., and Kleinberg, J. The link-prediction problem for social networks. Journal of the American society for information science and tech., 58(7), 1019-1031, 2007

Madejski, M., Johnson, M. and Bellovin, S. A study of privacy settings errors in an online social network. Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on. IEEE, 2012.

Manning, C., Raghavan, P. and Schutze, H. Introduction to Information Retrieval, Cambridge University Press. 2008.

McAuley, J. and Leskovec, J.. Learning to discover social circles in ego networks. Proceedings of NIPS 2012.

De Meo, P., Ferrara, E., Fiumara, G. and Provetti, A. Generalized louvain method for community detection in large networks. In Intelligent Systems Design and Applications (ISDA), 2011 11th International Conference on, pp. 88-93. IEEE, 2011

Minkus, T. and Memon, N. Leveraging Personalization to Facilitate Privacy. arXiv e-Prints. 2014.

Mishra, N., Schreiber, R., Stanton, I. and Tarjan, R. Clustering Social Networks. In Algorithms and Models for the Web-Graph, 5th International Workshop, volume 4863 of LNCS, pages 56–67. Springer, 2007

Newman, M. and Girvan, M. Finding and evaluating community structure in networks. Physical review E 69, no. 2 2004

Palla, G., Derenyi, I., Farkas, I. and Vicsek, T. Uncovering the overlapping community structure of complex networks in nature and society. Nature 435, 814. 2005.

Papadopoulos, S., Kompatsiaris, Y., Vakali, A., Spyridonos, P. Community Detection in Social Media. Data Mining and Knowledge Discovery 24(3), 515-554, Springer. 2012

Raghavan, U., Albert, R. and Kumara, S. Near linear time algorithm to detect community structures in large-scale networks. Physical Review E 76, 036106. 2007.

Rosvall, M. and Bergstrom, C. Maps of Random Walks on Complex Networks Reveal Community Structure. Proceedings of the National Academy of Sciences, 105 (4), 1118-1123. 2008

Squicciarini, A., Sundareswaran, S, Lin, D. and Wede, J. A3P: adaptive policy prediction for shared images over popular content sharing sites. Proceedings of the 22nd ACM conference on Hypertext and hypermedia. ACM, 2011.

Squicciarini, A., Shehab, M. and Paci, F. Collective privacy management in social networks. Proceedings of the 18th international conference on World wide web. ACM, 2009.

Spielman, D. and Teng, S.-H. (2008). A local clustering algorithm for massive graphs and its application to nearly-linear time graph partitioning. arXiv preprint:0809.3232, 2008

Spiliotopoulos, T., Pereira, D. and Oakley, I. Predicting Tie Strength with the Facebook API. Proceedings of the 18th Panhellenic Conference on Informatics. Pages 1-5. 2014

Sleeper, M., Cranshaw, J., Kelley, P., Ur, B., Acquisti, A., Cranor, L. and Sadeh, N. I read my Twitter the next morning and was astonished. A Conversational Perspective on Twitter Regrets. SIGCHI Conference on Human Factors in Computing Systems. 2013.

Strater, K. and Lipford, H. Strategies and struggles with privacy in an online social networking community. Proceedings of the 22nd British HCI Group Annual Conference on People and Computers: Culture, Creativity, Interaction-Volume 1. British Computer Society, 2008.

Taddicken, M. (2014). The “Privacy Paradox” in the Social Web: The Impact of Privacy Concerns, Individual Characteristics, and the Perceived Social Relevance on Different Forms of Self-Disclosure. Journal of Computer-Mediated Communication, 19(2), 248–273. doi:10.1111/jcc4.12052

Tehila, M. and Memon, N. Leveraging Personalization To Facilitate Privacy." Available at SSRN 2448026. 2014.

Vinh, N. X., Epps, J., and Bailey, J. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. The Journal of Machine Learning Research, 11, 2837-2854, 2010.

Vyas, N., Squicciarini, A., Chang, C. and Yao, D. Towards Automatic Privacy Management in Web 2.0 with Semantic Analysis on Annotations. CollaborateCom Conference, IEEE. November 2009.

Wang, Y., Leon, P., Scott, K., Chen, X., Acquisti, A. and Cranor, L. Privacy nudges for social media: an exploratory Facebook study. Proceedings of the 22nd International Conference on World Wide Web compansion. 2013.

Watson, J., Whitney, M. and Lipford, H. Configuring audience-oriented privacy policies. Proceedings of the 2nd ACM workshop on Assurable and usable security configuration. Pages 71-78

Xu, X., Yuruk, N., Feng, Z. and Schweiger, T. SCAN: a structural clustering algorithm for networks. In Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2007.

Yang, J., McAuley, J. and Leskovec, J. Community Detection in Networks with Node Attributes, ICDM '13.

Yousra, J. and Shehab, M. Access Control Policy Misconfiguration Detection in Online Social Networks. Social Computing (SocialCom), 2013 International Conference on. IEEE. 2013

Zerr, S., Siersdorfer, S., Hare, J. And Demidova, E. I Know What You Did Last Summer!: Privacy-Aware Image Classification and Search. SIGIR 2012.