UPPLA - Users Powered Privacy with Label-based Access Control for Online Social Networks

Leila Bahri, Barbara Carminati, Elena Ferrari, and William Lucia

STRICT Social Lab, Insubria University, Italy

ABSTRACT

Online Social Networks (OSNs) allow and encourage users to share their information, mostly personal, with their network contacts, many of whom are often strangers. Generally, access control to information in OSNs is enforced using a relationship-based model. This limits the flexibility offered to users in expressing their privacy preferences as they can do so only w.r.t. the types of relationships they establish with their OSN contacts. To answer this, in this paper, we propose UPPLA, a model that, by leveraging on LBAC (Label Based Access Control), operates with a labeling strategy that governs access to data items based on their classification versus that of the requesting entity. In UPPLA, users are empowered to assign customized labels to their friends and to their information; whereas access requests and privileges are evaluated by security properties carefully thought of and designed to establish orders between requestor's and information's labels. We have developed a prototype implementation of UPPLA and ran a usability study that revealed high scores for UPPLA's usability on five usability criteria.

1. INTRODUCTION

Online Social Networks (OSNs) are web-services that enable people to create connections with each other and to share content using these established links. OSNs have become very popular and invasive in a relatively short period of time. The number of their users, as well as the amount, type, and variety of data shared over them, has been knowing exponential growth. For instance, Facebook has more than 1,1 Billion users, and Google+, surpassing Twitter, has 500 Million users [25]. This explosion in OSNs adoption has enabled users more freedom and proximity in keeping in touch with their friends and in expanding their social networks. However, this has also created serious privacy breaches and concerns given the personal nature of information users share via OSNs on almost a daily basis [13].

In fact, users publish their personal stories and updates without being fully aware, in most cases, of the size of the audience that gets access to this information [7]. This might be a consequence of the large number of connections we maintain and we have to manage on our OSN accounts, of the unavailability of privacy settings that allow expressing our needs, or of the hard-to-manage and hard-to-understand privacy and access control settings, or of all.

For instance, the Dunbar number theory states that humans might not be in the cognitive ability of managing and maintaining more than 150 connections or friendship links [10]. However, statistics on OSNs show an average of 350 friends per user; a number that gets up to 650 friends per user for people between 18 and 24 years old [24]. Many people tend to agree with Dunbar's theory,[1] making it easily understood how hard it should be for an OSN user to be in fathomable control of the audiences of their published information. Consequently, some conservative individuals have withdrawn from OSNs as a definite solution to protecting their online social privacy.[2]

There is no disagreement that information, when accessed or used by inappropriate individuals, can be embarrassing, or damaging to persons, careers, governments, or countries [21]. Current OSN providers seem to understand the importance of empowering their users with tools to manage their privacy. For example, Facebook has been continuously updating its privacy settings interface providing more options to the users with, every time, finer granularity of access control on their data. Whereas Google+ has introduced the notion of circles that allows users to group their contacts and content in closed circles ensuring that only members of a circle can access the data disseminated through it. However, the privacy settings currently available in OSNs remain both complicated to use, and not flexible enough to model all the privacy preferences of OSN users [19]. This limitation seems to come, fundamentally, from the underlying access control model adopted and implemented by nowadays OSNs. In fact, current OSNs base their solutions on an access control model known as Relationship-Based Access Control (ReBAC) [12]. This model is characterized by the explicit tracking of interpersonal relationships among users, and the expression of access control policies in terms of these relationships. For example, users can organize their friends into lists based on the types of relationship existing

[1] http://www.forbes.com/sites/tjmccue/2013/01/15/social-media-maximum-150-friends

[2] https://nakedsecurity.sophos.com/2013/09/18/half-of-facebook-quitters-leave-over-privacy-concerns/

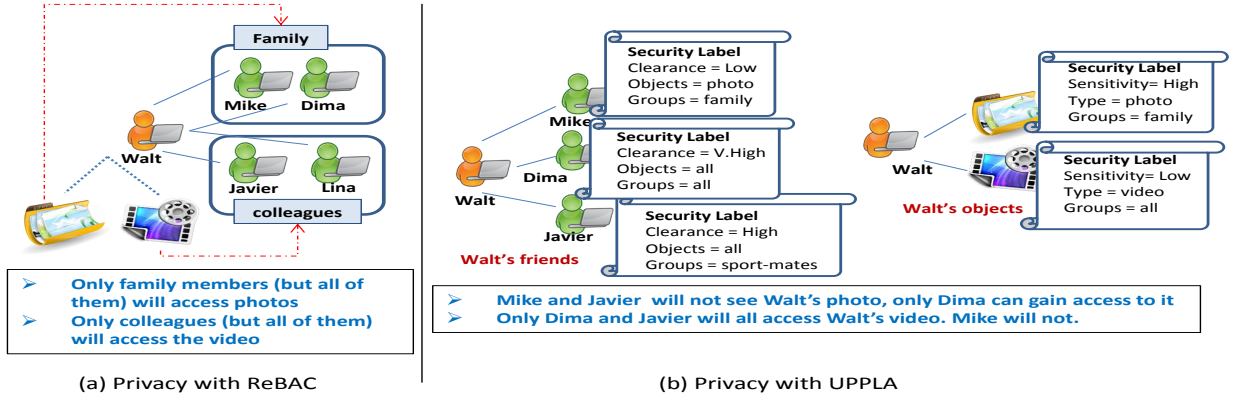

Figure 1: Privacy management with ReBAC vs. UPPLA

between them. As an example, a user can create a *family* list and a *colleagues* list. Information can then be set to be shared only with one or more lists.

However, this type of access control limits, by design, the options that could be available for privacy settings. For instance, defining privacy settings based only on the social relationships implicitly enforces that all the friends of a user who belong to the same relationship type are equal and, hence, will enjoy the same access and interaction rights. For example, if Walt puts his friends Mike and Dima in a *family* list and Jane and Lina in a *colleagues* list, then all the information he shares with the *family* list will be accessible by both Mike and Dima and the same goes for information shared with the *colleagues* list (see Figure 1(a)). However, Walt might not always want to share his information based on this categorization. For example, if he wants to share an item only with Mike and Jane then the only available way is to create a new list that contains these two friends. In addition to this clear inflexibility in expressing access preferences, both Mike and Dima, for instance, will enjoy the same interaction rights on Walt's objects specified for the *family* list. That is, both of them can equally comment on, like, or share a photo that Walt made available for the *family* list. However, it might be that Walt does not want one of them to be able to comment on the photo, for example. Moreover, the information that users create in OSNs is intermingled resulting in more complex access scenarios. For example, if Dima is allowed to comment on Walt's photo, she might also want to limit the audience of her comment and not to have it subject to the one set by Walt for the photo on which she commented

To overcome these limitations, in this paper, we propose a new model, UPPLA (Users Powered Privacy with Label-based Access-control), that introduces a new and more flexible way for users to express finer levels of granularity for their information privacy. UPPLA achieves this by basing on adapting Label-based Access Control (LBAC) to OSNs. More precisely, UPPLA exploits the security design of Mandatory Access Control (MAC), that sets levels of order between data requestors (i.e., subjects) and requested data (i.e., objects), within an underlying framework that establishes relationships between subjects and between them and their owned objects. That is, the OSN relationships between users and between data elements make the underlying framework on top of which MAC is utilized.

MAC consists at having a data owner or administrator assign ordered security levels (that is, labels) to subjects, and ordered sensitivity levels to objects [11]. These labels express the privacy and control requirements on the subjects and their interactions with the objects in the system. Access decisions are then made based on well defined axioms that ensure that subjects can only gain specific privileges on the objects based on the relationship existing among the object sensitivity levels than their own security levels. In UPPLA every user is considered the ultimate administrator of her/his data objects by assigning to them sensitivity labels, and to their friends security labels that express the access limitations she/he wants to put on them. Access decisions are then made based on properties that we carefully define based on the specificity and types of activities taking place in current OSNs. To account for the different aspects based on which users might choose their privacy settings w.r.t. their friends and to their data objects, we design the labels considering other relevant features, other than the security and sensitivity levels as it is in standard MAC. More precisely, labels in our model are expressed in terms of security or sensitivity levels, the types of relationships, and the types of objects.

As a basic illustration, Figure 1(b) depicts the security and sensitivity labels that Walt assigns to each of his friends and of his objects, respectively. As it can be seen from the Figure, access to an object is granted only to those friends having an equal or higher label. However, the richness of interaction types and access privileges that OSNs allow (e.g., tag, share, comment, etc) makes the picture more complex and requires careful design of different properties to account for all possible privilege request scenarios. This is what we dutifully explain, detail, and formalize in this work.

We have also implemented a prototype of our solution and ran a usability study. The results reveal that: 1) users suffer from limitations in expressing their privacy preferences on current OSNs, and 2) our method allows both easier and more flexible expression of users' privacy preferences with a satisfaction level of more than 80%.

The remainder of this paper is organized as follows. Section 2 introduces a background on the interactions allowed in OSNs nowadays. Section 3 describes the proposed model,

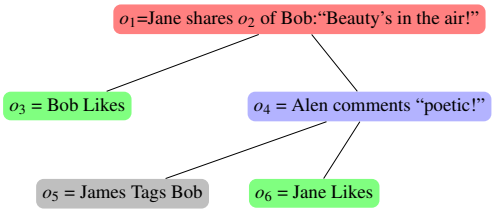

Figure 2: Typical objects' structure in OSNs.

whereas Section 4 provides its security and complexity properties. Section 5 describes the usability experiments, whereas Section 6 surveys related work. Finally, Section 7 concludes the paper and outlines future research directions.

2. BACKGROUND

In this paper, we consider the scenario of general-purpose OSNs that allow users to establish different types of relationships with their contacts and share different types of information. There are number of such OSNs in the online market nowadays, but it remains safe to say that the two biggest worldwide ones currently in the box are Facebook and Google+. Through each of them, users can establish connections with each other, and share information in multiple supported formats. Generally, users are associated with an information space typically made of two main blocks: a profile and a wall or timeline. The profile contains personal information, like full name, contact address, gender, age, education etc, whereas the wall organizes, in a graphically timeline manner, the data objects the user shared with her friends or those that her friends have shared with her.

Type	Dependency
Text (TX) [text posts & profile data]	Independent
Photo (P)	Independent
Video (V)	Independent
Like (L)	Dependent
Comment (C)	Dependent
Tag (TG)	Dependent
Geo-Location (GL)	Dependent
Friend-Post (FP)[3]	Independent

Table 1: Types of information shared in OSNs.

Users can upload content of different types to their information space, or interact on their friends' shared information by liking it, commenting on it, or tagging one or more other of their friends in it, etc. Such interactions result in the creation of other data elements that are related to each other, and that might also be owned by different users. Figure 2 illustrates an example of a published content, the data elements resulting from interactions on it, and the relationships between them.

Clearly, some types of information shared on OSNs can stand as *independent* by itself, while other types only make sense and are always *dependent* on another information. For example, a comment cannot exist unless attached to a previously published post. Table 1 lists the different types of information that users can share and manage in nowadays OSNs specifying whether they can be independent or not.

[3]This type refers to an object posted on a user's wall by one of her friends.

Privilege	Privacy settings
Read	Manageable for all types except C and L
Add-Comment	No available privacy settings
Add-Like	No available privacy settings
Add-Tag	Manageable both for who can add a tag and who can view added tags
Share	The audience specified by the original owner is automatically enforced on the shared copy
Write	Manageable at limiting who can post on one's wall and on who can see these posts but as a batch setting

Table 2: Users interactions in current OSNs and their associated possible privacy settings.

In addition to publishing content, OSN users can also specify their privacy preferences, from a privacy settings interface, regarding who can access what information. Generally, a user can group her friends into groups and put some interaction limitations on some groups (e.g., cannot share content with me, or cannot tag me in photos). Additionally, at the moment of publishing a new *independent* information, the user can choose the friends or the groups to whom this content is addressed.

Without loss of generality, Table 2 provides a summary of the actions users can perform on OSNs along with the highest and finest available privacy settings, as currently available either in Google+ or in Facebook. We note that the *write* privilege refers to a user posting on her friends' walls and not to writing on one's own wall. As we can read on Table 2, the *read* privilege can be customized only for independent types of information, such as text, photo, etc. Dependent information, however, with the exception of photo tags, are automatically available to whoever has a *read* privilege on the information they depend on. This might be a real limitation, as I might want to customize who can see my comments on others' posts. Moreover, users can only control the *read*, and *add-tag* privileges, whereas all the other privileges are available to whoever got a *read* access to an information. For example, if I allow a friend X to read a post Y, I cannot prevent X from commenting on Y. However, users might need not only to control the visibility of their information but also the interactions on it. In addition to this, users cannot control the *share* privilege. More interestingly, as on Facebook for instance, any friend who got a *read* privilege on a post can *share* it; however, this shared copy will only be available to the initial audience allowed by the original owner of the post. This might constitute a major limitation to the functionality of the *share* action in itself. For example, I might make a post available to only three of my friends, but I might also want them to share it with their trusted friends all while ensuring that non else of my friends will have access to it. Finally, users can control who can

create content in their information space, but only in a binary manner. That is, it is either that a friend is allowed to post on a user's wall or not. However, users might want to control the levels of visibility of this content published by others in their space, instead of such basic on/off settings.

Overall, and as mentioned earlier, the available privacy settings for users present limitations to the expression and enforcement of users refined privacy needs. Our suggested method addresses these limitations and offers a more flexible and more usable privacy setting functionality to OSN users. We define and and detail the underlying model in the following section.

3. THE UPPLA MODEL

We start by presenting basic definitions of used elements. Second, we define how privacy requirements of users can be expressed using labels. Finally, we introduce the mechanism governing access decisions, based on defined properties and axioms.

3.1 Basic Definitions

Given their networked nature, OSNs are generally abstracted as graphs where nodes represent users and edges model relationships among them. Overall, OSN graphs can be of two types, directed or undirected, based on the underlying logic of the modeled service. Without loss of generality, in this paper, we model an OSN as an undirected graph $FG = (U, FR)$, where U is the set of users, and FR is the set of edges. We say that two users $a, b \in U$ are friends in the OSN if and only if there exists one direct edge connecting them. That is, $\exists e_{a,b} \in FR$.

OSN users create and share different types of information (see Table 1) leveraging on the relationships they establish with others. We define the set of all information types in the OSN as $OTS = \{TX, P, V, L, C, TG, GL, FP\}$, and we refer to all types of information as an *object* that we formally define as follows:

DEFINITION 1 (OSN OBJECT). *An OSN object o is represented by a tuple* $(type, owner, parent, children, copyof)$, *where:*[4]

(i) $type \in OTS$ *denotes the type of the object.*

(ii) $owner \in U$ *denotes the owner of the object. If* $o.type \notin \{TG, FP\}$, *the owner is the user who generated the object (dependent or independent). Otherwise, it is the user who is tagged or the user on whose wall the friend post is made.*

(iii) parent: if o is a dependent object, this refers to the object on which it depends. It is set to null, otherwise.

(iv) children is the set of objects that depend on o, as generated by interactions on it, if any. It is set to null, otherwise.

(v) copyof: if $o.parent = null$ *and o is a shared copy of another object ob, this is equal to ob, otherwise it is set to null.*

In addition to the objects that users can create, we consider the existence of another special object type, that we refer to as the *root* object, and that models user walls. We consider that every OSN user owns exactly one *root* object that is created by default upon her subscription to the OSN. Root objects serve for controlling the function of *writing* on users' walls by their friends as we will detail later (see Algorithm 1). A *root* object associated with user *a* is modelled as $root_a$ = *(root, a, null, null, null)*.

Example 1. Consider the OSN objects on Figure 2. By Definition 1, object o_1 is modeled as *(TX,Jane, null,* $\{o_3, o_4\}$, o_2*)*, object o_4 as *(C, Alen,* o_1, $\{o_5, o_6\}$, *null)*, whereas the tag object o_5 is modeled as *(TG, Bob,* o_4, *null, null)*.

Besides creating and sharing objects, there are different types of interactions that users can perform on them (see Table 2). As performing an action on an object results in the creation of another one, object owners need to have control over the privileges they want to grant to their friends with this regard. Based on a thorough understanding of the different actions on objects possible to be performed in an OSN, as presented in Table 2, we accordingly define the set of controllable privileges in our model as PS = {*read, add-comment, add-like, add-tag, share, write*[5]}.

Users should be able to express access requirements on their objects w.r.t. which privileges from PS can performed on an object and by whom. As has been mentioned earlier, access requirements in our proposed model are expressed by means of labels and access decisions are evaluated based on defined properties. We start by defining the labels that users assign, to their friends and to objects, to express their privacy requirements, then we detail how access control is performed.

3.2 Privacy requirements formulation

To express privacy requirements, users assign security labels both to each of their owned objects and to each of their friends. By labeling friends, users express the limitations they want to put on them, whereas object labeling serves for expressing the sensitivity of the information and its accessibility criteria. Before formally defining these two types of labels, we first mention that we assume a set of possible groups that users can organize their friends into. These groups are created by users depending on their organizational preferences and can be viewed as similar to users' lists or users' circles as available on Facebook and Google+, respectively. A possible example of a set of user groups might be, {friends, colleagues, teammates, schoolmates, family, acquaintances}. This set is created at user's side and is only used for the purpose of creating their friends and objects se-

[4]We use the dot notation to refer to the parameters in an object tuple.

[5]In this paper, the *write* privilege refers to users posting on their friends' walls.

curity labels. In addition to this, we also assume in the system a totally ordered set of levels *LOS* = {*Unclassified (UC), Very Low (VL), Low (L), Medium (M), High (H), Very High (VH)*}, with UC < VL < L < M < H < VH.

We refer to a friend label as *Friend Clearance Label (FCL)* and we formally define it as follows:

DEFINITION 2 (FRIEND CLEARANCE LABEL - FCL). *Let $a,b \in U$ be two friends in the OSN (i.e., $\exists e_{a,b} \in FR$). The label $FCL_{a,b}$ assigned by user a to user b, is denoted by the tuple $(CL_{a,b}, TS_{a,b}, GS_{a,b})$, where:*

- $CL_{a,b} \in LOS$ *is the clearance level that a grants to b.*
- $TS_{a,b} \subseteq OTS$ *is the set of object types that a wants b to get access to.*
- $GS_{a,b}$ *is the set of groups, out of the groups set created by a, that a assigns to b.*

We refer to object labels as *Object Sensitivity Labels (OSL)*, and we formally define them as follows:

DEFINITION 3 (OBJECT SENSITIVITY LABEL - OSL). *Let $a \in U$ be a user in the OSN and let o be an OSN object such that $o.owner = a$. The label OSL_o, of object o, is denoted by the tuple (SL_o, TY_o, GS_o), where:*

- $SL_o \in LOS$ *is the sensitivity level of o.*
- $TY_o = o.type$ *is the type of o.*
- GS_o *is the set of groups, out of the groups set created by a, for which object o's availability should be considered.*

Example 2. Let Walt be an OSN user who uploads a new photo (*GP*) from his graduation ceremony assigning to it OSL_{GP} =(*L, P,* {*colleagues, family, university*}). This means that *GP* has a low sensitivity and it concerns Walt's colleagues, family, or university-mates. Suppose now that Walt has assigned $FCL_{w,j}$ =(*H,* {*P, T, V*}, {*colleagues, university*}) to his friend Jane. This means that Walt grants to Jane a high clearance level, allows her to see his photos, texts, and videos, and he considers her a member of the colleagues and university groups.

3.3 Access control decisions

Assuming that all users' friends and all their owned objects are labeled with FCLs and OSLs, respectively, we need to detail how they can be used in managing access control. In fact, access control management is typically triggered by an access request made by a subject for an object. As we have provided earlier, access to objects in an OSN can be related to one of the privileges enclosed in the privileges set (PS). To reflect this broader range of privileges, compared to traditional scenarios formalizing only read and write privileges, we refer to access requests as *interaction requests - (IR)* . An IR can be of two types: 1) performed on an object (e.g., read or comment on an object), or 2) made to create an object related to another user (i.e., tag a user or write on her wall). We define an *interaction request* as follows:

DEFINITION 4 (INTERACTION REQUEST - IR). *Let $a,b \in U$ be two friends in the OSN (i.e., $\exists e_{a,b} \in FR$). An IR by user b is denoted by the tuple $ir_{b,x} = (x,\ p,\ b)$, where $p \in PS$ is the requested privilege, b is the user requesting it, and:*

- *x = o, where o is an object s.t. $o.owner = a$ and $o.parent = null$; **if** $p \in$ {read, share}.*
- *x = a, where a is the target user; **if** $p = write$.*
- *x = (a,o), where a is the target user and o is an object; **if** p =add-tag.*
- *x = o, where o is an object s.t. $o.owner = a$; **if** $p \in$ {add-like, add-comment}.*

Like in standard MAC, the evaluation of IRs is performed based on defined properties that establish orders between subject labels and object labels and grant or deny access requests based on these orders. Generally, in standard MAC, there are two properties that govern the system. These are related to the *read* and to the *write* requests, respectively. However, our model requires not only the redefinition of these two properties to account for all the features in the FCLs and the OSLs, but it also requires the introduction of new axioms. Indeed, the interactions in an OSNs are not limited to simple *read* and *write* privileges. To account for the specificity of the OSN interactions scenario, the evaluation of IRs in our model is carried out based on three different axioms and three different relationship properties. We detail these in what follows.

3.3.1 The fundamental security property

Like in standard MAC, order relations among labels must be defined to have the basis on which they can be compared. In our model, we refer to this relation as the *dominance relationship*, and we formally define it as follows:

DEFINITION 5 (THE DOMINANCE RELATIONSHIP). *Let $a,b \in U$ be two friends in the OSN (i.e., $\exists e_{a,b} \in FR$) and let $FCL_{a,b}$=($CL_{a,b}$, $TS_{a,b}$, $GS_{a,b}$) be the friend label a assigned to b. Let o be an object s.t. $o.owner = a$, and let $OSL_o = (SL_o, TY_o, GS_o)$ be its OSL. $FCL_{a,b}$ dominates OSL_o, and we write $FCL_{a,b} \gg OSL_o$, if an only if,*

$$CL_{a,b} > SL_o \wedge TY_o \in TS_{a,b} \wedge GS_o \cap GS_{a,b} \neq \emptyset$$

Example 3. Consider Example 2. The FCL that Walt assigned to Jane dominates the OSL that he assigned to his graduation photo because: 1) the clearance level assigned to Jane (High) is higher than the sensitivity level of the photo (Low); 2) object type photo is within the types set in Jane's label; and 3) the group sets in Jane's and in the photo's labels have groups in common (university, colleague).

Using the dominance relationship, we introduce the first property in our model, referred to as the *Fundamental Security Property (FSP)*, and defined as follows:

DEFINITION 6 (FUNDAMENTAL SECURITY PROPERTY). *Let $a,b \in U$ be two friends in the OSN (i.e., $\exists e_{a,b} \in FR$) and*

let $FCL_{a,b} = (CL_{a,b}, TS_{a,b}, GS_{a,b})$ be the friend label a assigned to b. Let o be an object s.t. $o.owner = a$, and let $OSL_o = (SL_o, TY_o, GS_o)$ be its label. Let $ir_{b,o} = (o, p, b)$ be an IR made by b on o s.t $p \in \{$read, share, add-like, add-comment$\}$. $ir_{b,o}$ satisfies FSP, if and only if:

$$FCL_{a,b} \gg OSL_o \ .$$

Informally, the FSP dictates that for a *read*, *share*, *add-like*, or *add-comment* IR on an object to be granted, the FCL of the requestor must dominate the OSL of the object it targets. It is to be noted that a primitive requirement of the FSP is that the requestor and the object owner are friends in the OSN as otherwise the friend label to be compared would not exist. As a consequence, in our model, only friends of a user can have access to and interact on her information space. This is not fully aligned with the classical approach of OSNs where there is the possibility of making information available to friends-of-friends or public to all the OSN users.

Regarding the case of public information, this is the simple equivalent of making the object unclassified (i.e., the object has a sensitivity level equal the value $UC \in LOS$ from the labels' levels ordered set). Similarly, the system, by default, assigns to all strangers to a user (her non-friends in the OSN) a default label $FCL_{default}$=(UC, GS, OTS), where GS refers to all the groups a user defined, and OTS is the set of all object types in the OSN. This ensures that unclassified objects are accessible to all strangers. As for the friend-of-friend scenario, the approach of our model is to allow users a means of enforcing their privacy preferences and offer them an environment that facilitates the understanding and control of their objects audiences. Indeed, this can be ensured by the suggested labels assignment, as users can be fully aware of (and track) who might gain access to what, contrary to the approach of allowing the unlimited and uncontrollable friend-of-friend access. That being said, our method can still be extended to cover this scenario by offering a mechanism for automatic FCLs assignment to friends of friends. A possible solution is to combine FCLs on a path from requestor to object owner. This is similar to the issue of calculating the trust between two nodes in a network given trust values of all the edges making the path or paths connecting them [18]. This issue is a research subject in itself. For this, we keep this scenario out of the scope of this paper and we plan to address it as future work.

Another important note regarding the FSP is that a *read* IR that satisfies it grants a *read* access to the targeted object, that, according to Definition 4, is an independent object. Normally, when a user requests access to an independent object, a photo for example, she also implicitly requests *read* access to all its dependent information. However, we recall that in our model access to dependent objects is also restricted by their respective OSLs as assigned to them by their owners and not by the owner of their parent object. For this, when a *read* IR on an object o is granted (it satisfies the FSP), the system ensures its propagation downward all the children of o (i.e., $o.children$). That is, the system issues from every granted *read* IR on an object, a similar IR on all its children to ensure their evaluation independently of their parent object and only based on their proper OSLs. If one of these system IRs is granted, the same propagation is performed on its second level children, and so on. It is to be noted that *share* IRs can also be issued on independent objects only (Definition 4), however this propagation does not apply to them as the share should be applied only to the target object and not to its children. For instance, a user sharing a photo does not share its related comments (see Section 3.3.4 for more details).

The FSP is sufficient for all the IRs it targets, except from the *share* one. This is what we address in the following subsection.

3.3.2 The share privilege and the share up property

A *share* IR is special in the sense that, if granted, it results in the creation of a copy of the original object. The problem here is on what should be the label of this shared copy. The simplest idea is to make the copy inherit the same label as the original object; however, this might limit the intended purpose of the share functionality. Indeed, when a user allows a friend to share an object, this means that the user wants the object to be available to a wider audience from the friend's side. If the user wants to limit the audience of an object strictly to her allowed friends, then she will not allow any share on the object. For this, it is more logical for the shared copy to have a different label than that of the original object it refers to. However, there should be some limitations on the copy's label in such a way that the copy might be available to a wider audience without neglecting the privacy preferences set on the original copy.

To address this, we introduce an additional axiom, called the *Share Higher Property*. Before formally defining it, we first introduce a new concept that governs the relationship between the label of an object and that of its copies made from a granted *share* IR. We refer to this as the *Not-declassify relationship* and we define it as follows:

DEFINITION 7 (NOT-DECLASSIFY RELATIONSHIP). *Let $OSL_o = (SL_o, TY_o, GS_o)$ be the label of an object o. Let $\bar{OSL} = (\bar{SL}, \bar{TY}, \bar{GS})$ be the label assigned to object $\bar{o}$ that is a copy of o (i.e., $\bar{o}.copyof = o$). Label $\bar{OSL}$ does not declassify label OSL_o, and we write $\bar{OSL} \not\leq OSL_o$, if and only if: $\bar{SL} >= SL_o$*

The Not-declassify relationship requires that the sensitivity level of an object's copy is equal or higher than that of the original object. This would ensure that a share action does not declassify the shared information. It is to be noted that no restriction is made on the groups set parameter of the label. This is because the organization of friends into groups is dependent to each user without necessarily mapping to the groupings adopted by their friends. For example, those who might be family to a user, might be colleagues to

another. However, it is worth mentioning that one purpose of users, via allowing the share action, is to disseminate their objects to wider and different audiences as long as this does not result in explicitly making the information available to a non-desired friend.

Based on the Not-declassify relationship, the second axiom of our model, the *Share Higher Property*, is defined as follows:

DEFINITION 8 (SHARE HIGHER PROPERTY - SHP). *Let $a, b \in U$ be two friends in the OSN (i.e., $\exists e_{a,b} \in FR$), and let $FCL_{a,b} = (CL_{a,b}, TS_{a,b}, GS_{a,b})$ be the friend label a assigned to b. Let o be an object s.t. $o.owner = a$, and let $OSL_o = (SL_o, TY_o, GS_o)$ be the label a assigned to o. Let $ir_{b,o} = (o, share, b)$ be a share IR made by b on o. Let $\bar{OSL} = (\bar{SL}, \bar{TY}, \bar{GS})$ be the object label that b intends to assign to the copy of o. We say that $ir_{b,o}$ satisfies SHP, if and only if:*

$$\bar{OSL} \not\preceq OSL_o$$

Informally, the SHP enforces that users can share their friends' objects only if they assign to the shared copy an OSL that does not declassify the one of the original object. We recall that a *share* IR should first satisfy the FSP. However, closely analyzing the SHP, we can see that it still does not fully solve the disclosure risk when an object owner and the sharing friend have a friend in common who is not initially allowed to access the object by the owner. To make it clearer, consider the scenario in Example 4.

Example 4. Consider Example 2, and assume that Walt and Jane have Mina as a common friend in the OSN. Let Walt's and Jane's FCLs for Mina be $FCL_{w,m}$ =*(VL, {T}, {university})*, and $FCL_{j,m}$ =*(M, {P, T, V}, {university})*, respectively. We recall that the OSL of Walt's photo is, OSL_{GP} =*(L, P, {colleagues, family, university})*. Clearly, Mina does not have access to Walt's photo as $FCL_{w,m}$ does not dominate OSL_{GP}. Assume that Jane shares the photo and assigns to the shared copy OSL' =*(M, P, {colleagues, university})*. This satisfies the SHP and so the share will be granted. This will also make Mina able to see Walt's graduation photo from the copy that Jane shared. Based on Walt's preferences, Mina should not have been able to view the photo.

The disclosure represented in Example 4 might happen when the sharing friend and the owner of the original object have friends in common. Though the SHP ensures that the shared copy does not declassify the object, it cannot control the FCLs the sharing user is assigning to their friends. To prevent such *horizontal disclosures*, we enforce the preferences of objects' owners by considering their labels and not the copy's when the requestor, the sharer, and the owner are mutual friends. To explain it, assume $a, b, c \in U$ are mutual friends in the OSN (i.e., $\exists e_{a,b}, e_{a,c}, e_{b,c} \in FR$). Suppose b owns o_b that is a shared copy of o_a owned by a. Assume c makes a *read* IR, ir_{c,o_b}, on object o_b. Before evaluating ir_{c,o_b}, it is first checked if the target object is a copy of an original one. If it is, it is checked if the requestor c has a direct relationship with the copy's owner ($o_b.copyof.owner = a$). If such a relationship exists, the request is rewritten substituting the target object by the original one of which it is a copy. That is, ir_{c,o_b} becomes ir_{c,o_a} and hence the security labels assigned by a both to c and to o_a are the ones used to evaluate ir_{c,o_a}. This is enforced as per Algorithm 1 as will be presented later.

3.3.3 The add-tag & write privileges and the write higher property

Both the *add-tag* and the *write*[6] interactions, if granted, result in the creation of objects by a user who is not their owner (Definition 1). This creates the problem of what should be the labels to be assigned to these resulting objects. For instance, assume Bob has a very high level of trust in Alice and allows her to post on his wall. Alice might hold sensitive information about Bob and, as such, her posts on his wall might reflect some of it. Thus, Bob would want that Alice's posts on his wall be managed with a label that reflects their expected sensitivity; that is a label that is at least as high as the one he assigned to Alice. On the other hand, if Bob's trust in Alen is low, then he might need to impose more control on her posts on his wall as she might post something to embarrass him, for example. These same examples apply to the *add-tag* interaction as well.

To cope with this, we suggest enforcing two conditions on the labels that requestors of granted *write* or *add-tag* IRs can assign to the resulting objects. The first condition imposes a sensitivity level for the created object that is at least as high as the clearance level that the affected user assigned to the requestor, if this latter is higher than the medium level. However, if the clearance level assigned by the affected user to the requestor is lower than the medium level, its inverse is set as the least requirement for the sensitivity level of the resulting object. For example, if Bob assigns a high clearance level to Alice, Alice's posts on his wall will have at least a high sensitivity level. However, if Bob assigns to Alen a low clearance level, Alen's posts on his wall will have a sensitivity level at least equal to the inverse of Alen's clearance level (i.e., a high sensitivity level). As mentioned before, the rationale behind this is that if Bob highly trusts Alice, then most probably the content she will post about him would be sensitive and should be available only to other friends he trusts at least as much as he trusts Alice. However, if Bob's trust in Alen is very low, he might want to enforce more control over what she posts about him, and so the inverse of his trust in her is used as a least requirements for Alen's posts on Bob's wall.

The second condition imposes that the group set allowed by the resulting object's label is exactly the group set that the affected user specified in her label for the requestor. This is because, if Bob assigned the group "colleagues", for example, to Alice, then it is likely that Alice's posts about him will also concern the "colleagues" group.

To formalize these conditions, we introduce the third ax-

[6]We remind that a *write* refers to a user posting on a friend's wall.

iom of our model, referred to as the *Write Higher Property - (WHT)*, and defined as follows:

DEFINITION 9 (WRITE HIGHER PROPERTY - WHT). *Let $a,b \in U$ be two friends in the OSN (i.e., $\exists e_{a,b} \in FR$) and let $FCL_{a,b}$=($CL_{a,b}$, $TS_{a,b}$, $GS_{a,b}$) be the label a assigned to b. Let $ir_{b,x} = (x,p,b)$ be an IR such that $p \in \{write, add\text{-}tag\}$ and $x = a$, if $p = write$, $x = (a,ob)$, if $p = add\text{-}tag$, with ob the object to be tagged. Let o be the object resulting from granting $ir_{b,x}$ and OSL_o be its assigned label by b. $ir_{b,x}$ satisfies WHT if and only if: $o.owner = a \wedge OSL_o = (SL_o, TY_o, GS_o)$ s.t., $GS_o = GS_{a,b}$,*

$$TY_o = \begin{cases} TG & \text{if p=add-tag} \\ FP & \text{if p=write.} \end{cases},$$

and

$$SL_o \geq \begin{cases} CL_{a,b} & \text{if } CL_{a,b} \geq M \\ f^{-1}(CL_{a,b}) & \text{otherwise.} \end{cases}$$

With $f^{-1}(y), y \in LOS$ being the inverse function for security levels. For example $f^{-1}(VH) = VL$, $f^{-1}(L) = H$, etc

By Definition 9, in addition to enforcing the conditions discussed before on the labels of the resulting objects from granted *write* and *add-tag* IRs, the WHT property ensures that the owners of these objects are not the users making the request but the ones affected by it.

Example 5. Consider Example 2, and assume Jane has the right to post on Walt's wall and that she posts on it a video of Walt's that she has made during his graduation ceremony. We recall that Walt assigned to Jane $FCL_{w,j}$ =(*H*, {*P, T, V*}, {*colleagues, university*}). By the WHT property, the video's owner id Walt, and a possible granted label that Jane can assign to it is: $OSL(video)$ =(*H, FP,* {*colleagues, university*}). This means that the video will be available only to the friends of Walt to whom he assigned a high or a very high clearance level, who belong to the colleagues or to the university groups, and who are allowed to see Walt's friends' posts.

3.3.4 Access control enforcement

Putting it up all together, Algorithm 1 defines how access control is enforced in the system based on the privacy preferences of users as expressed in FCLs and OSLs and on the properties defined above.

Algorithm 1 takes as input an IR to be evaluated and produces as output a *granted* or *denied* message. The algorithm switches the privilege requested by the IR and ensures the enforcement of the corresponding properties. More precisely, for a *read* IR, the algorithm starts by checking if the target object is original or a copy. If it is a copy, it makes a call to the *NoHorDisclosure* function that ensures the rewriting of the IR, as long as the involved parties are mutual friends (the *IsFriend* call in line 36), to enforce the labels of the copied object and of its owner and not of the copy's (lines 3,4). After making the IR immune to horizontal disclosures, a *granted* or *denied* message is returned, depending on whether or not the IR satisfies the FSP (lines 5-8).

Algorithm 1 Access control enforcement

Input: An IR, $ir = (x,p,b)$
Output: *granted* or *denied*

```
 1: Switch p do
 2:     Case read //x is an object
 3:         if x.copyof ≠ null then
 4:             ir ← NoHorDisclosure(ir)
 5:         if SatisfyFSP(ir) then
 6:             send granted
 7:         else
 8:             send denied
 9:     Case write //x is a target user
10:         root_x ← GetRootOf(x)
11:         ir_allowed ← (root_x, read, b)
12:         if SatisfyFSP(ir_allowed) ∧ SatisfyWHP(ir) then
13:             send granted
14:         else
15:             send denied
16:     Case share //x is an object
17:         if SatisfyFSP(ir) ∧ SatisfySHP(ir) then
18:             send granted
19:         else
20:             send denied
21:     Case add-tag //x is a pair (usr, obj)
22:         ir_allowed ← (obj, read, b)
23:         ir ← (usr, p, b)
24:         if SatisfyFSP(ir_allowed) ∧ SatisfyWHP(ir) then
25:             send granted
26:         else
27:             send denied
28:     Case add-like ∨ add-comment //x is an object
29:         ir_read ← (x, read, b)
30:         if SatisfyFSP(ir_read) ∧ SatisfyFSP(ir) then
31:             send granted
32:         else
33:             send denied

34: function NoHorDisclosure(ir_a,o){
35:     while o.copyof ≠ null do
36:         if isFriend(a, o.copyof.owner) then
37:             NoHorDisclosure(ir_a,o.copyof)
38:         else
39:             ir_a,o ← ir_a,o.copyof
          return ir_a,o}
```

For *write* IRs, the algorithm first checks if the requesting user has a *write* privilege on the targeted user's wall. This is achieved by making a *read* IR on the *root* object of the targeted user (lines 10,11). If this system generated *read* IR satisfies the FSP and the input *write* IR satisfies the WHP, this latter is *granted*, otherwise a *denied* message is returned (lines 12-15). In case the input IR is a *share* one (line 16), it is checked if it satisfies both the FSP and the SHP sending a *granted* message if it does, or a *denied* one in the failing case. As for *add-tag* IRs, a *read* IR on the object to be tagged is formulated by the system (line 22). Then it is checked if this system issued IR satisfies the FSP and if the input *add-tag* IR satisfies the WHP. If the two conditions are satisfied, the input IR is *granted*, otherwise it is *denied*. Finally, for *add-like* and *add-comment* IRs, they are granted only if they

satisfy the FSP and if a system issued *read* IR on the object they target (line 29) is granted (i.e., it satisfies the FSP too) (lines 30-33).

4. SECURITY AND COMPLEXITY PROPERTIES

In this section, we present the security properties of our model and we discuss the complexity of Algorithm 1.[7]

4.1 Security properties

Access control in UPPLA is enforced by Algorithm 1 that ensures that all IRs are granted only if they satisfy the model's properties applying to them. To formalize the security property of the system, we define the concept of its *state* as the set, $SIR = \{ir_1, ir_2, ..ir_n\}$, of interaction requests currently granted in the system. We say that the state of the system is secure, and we refer to it as the *secure state* if and only if $\forall ir_i \in SIR, ir_i$ satisfies all the model's properties. The system changes its state only when a new IR is received and its processing by Algorithm 1 returns a granted message. Given a *secure state*, the following security property holds:

THEOREM 4.1 (SECURE SYSTEM). *Let SIR be the current state of the system. Let ir_{new} be a new interaction request issued to the system. Algorithm 1 issues a granted message if and only if $SIR_{new} = SIR \cup \{ir_{new}\}$ is a secure state.*

4.2 Complexity analysis

The complexity of Algorithm 1 is $\mathcal{O}(m)$, where m is the maximum number of chained shares an OSN object is subject to. m is expected to be small as the highest shares that an object can be subject to is to broadcast to all the OSN. Research works show that the principle of six-degrees of separation exists in today's OSNs with even a shorter scale (i.e., four-degrees) [9]. Therefore, it can be estimated that at most $m = 6$. Besides Algorithm 1, UPPLA's performance also depends on the propagation of granted *read* IRs on an independent object along the tree of its dependent children. The complexity of this propagation is $\mathcal{O}(d)$, where d is the longest object dependency in the system (i.e., the maximum depth of the tree of dependent children on an independent object). d is also expected to be very small as, though the number of interactions on an object can be very big (e.g., a picture of a celebrity might receive thousands of likes/comments), the number of chained interactions is usually limited. That is, users, normally, do not comment on the comment of a comment, etc. Therefore, UPPLA is expected to have usable performance levels that we also plan to demonstrate by experimental results in future works.

5. USABILITY EXPERIMENTS

We have implemented a prototype of our method as a web application (hereafter referred to as *UPPLA app*) that we interfaced with Facebook graph API. The choice for Facebook was because of the flexibility its graph API offers and also because of the number of users it has, dominating all the other available OSNs. Participants were asked to login to UPPLA app using their Facebook credentials and to accept to grant to it access to their wall both for *read* and *write* privileges.[8] After login, the participants were first asked to take a pre-experiment survey that asks about their understanding and appreciation of the current privacy settings as available in Facebook. Afterwards, they were asked to follow a quick guide to use the app. More precisely, users could view, from UPPLA app, a sub-list of their friends retrieved from their Facebook friends list[9] and use the app's interface to assign FCLs to them. After labeling some of their friends from the presented ones, participants were guided to publish a text post from the app's interface assigning to it an OSL. Upon publishing of the labeled text post, users could view the list of friends, selected by UPPLA app from the ones they have labeled already, who should make the audience of the post. After each performed step, participants were asked to answer survey questions[10] designed to get their feedback on their perceived usability of the features they have just used on the app. Users could exit the app at any time to revoke all access rights they granted to it on their Facebook account. Upon doing so, they were asked to answer a post-experiment survey that contained all the questions that were asked while using the app as well as other questions on their general satisfaction with the new method and how they perceive it compared to the privacy settings available on Facebook.

Recruitment was mainly done through social media dissemination. 26 people (16% Italian students, 10% non-students from France, US, Spain and Greece, and 74% non-students from Morocco) consented to the terms of and used UPPLA app. All the survey questions were multiple choice with answers ordered on a scale of five ordered items (i.e., Strong No, No, Moderate, Yes, Strong Yes). When processing the survey results, these items were mapped to numerical values in the range [1,5] for the computation of averages.

We start by introducing the results of the pre-experiment survey with Table 3 presenting its five questions and their corresponding answers average across all the 26 participants. Based on the obtained results, we can understand that participants highly care about their privacy on Facebook (question number 5), the majority of them is aware of the custom privacy setting in Facebook (question number 1), but its reported usage is low (question number 2). Another inter-

[7]Proofs of the results in this section are reported in a TR at, http://strict.dista.uninsubria.it/?page_id=747

[8]With Facebook's new graph policy of 30-05-2015, the request of these privileges needed consent from Facebook. As such, the app was deployed in a testing environment and participants were explicitly declared as testers.

[9]Facebook graph API does not allow access to friends lists. UPPLA app fetched the friends sub-lists from recent posts on users' walls.

[10]The survey is available at: http://strict.dista.uninsubria.it/?page_id=747

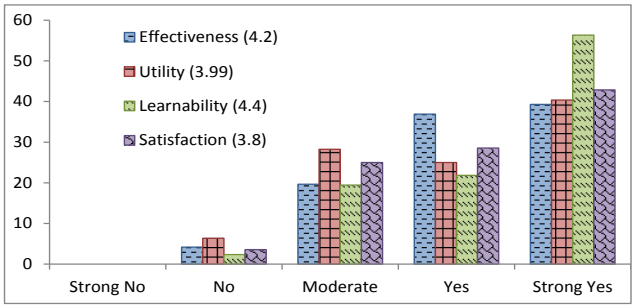

Figure 3: Participants feedback on UPPLA's usability

esting result from this part is related to the fact that participants admitted the existence of some Facebook friends that they could have forgotten about. However, when asked if their friends are organized into groups with defined privacy settings on them (question number 4), the reported answers were almost all a strong no. This contrast is important to consider as it shows a gap between what users think about their privacy needs and how they actually enforce them. This gap could be the result of complicated privacy settings, as it could be related to other factors, such as laziness or lack of time to spend on making these settings. For this, we believe that our method that enforces the expression of users privacy requirements by means of assigning labels would bridge over this gap. That is, as suggested in our method, users are required to assign FCLs to their friends, and this can be set to take place right upon the establishment of a new friendship link, resulting in the enforcement of intended privacy settings without the burden of organizing the friends into groups or customizing the settings afterwards.

Question	Answers Avg
1. I am aware of the CUSTOM privacy setting offered by Facebook to restrict the audience of my posts and I think it is convenient to use	**4.19**
2. I often use the CUSTOM privacy setting on my Facebook when I create new posts or upload new pictures	**3.92**
3. If I now review my Facebook friends list, I think I will find some people who I completely forgot I am friends with on Facebook	**3.38**
4. In my current Facebook account, I have almost all my friends organized into groups and I have set custom privacy settings on almost all of them	**1.76**
5. I care about my privacy on Facebook and I have concerns about how to better manage it	**4.11**

Table 3: Pre-experiment survey (answers range= [1-5])

As mentioned earlier, participants were asked to answer survey questions both while using the app and upon exiting it to get their feedback on the usability of our method. More specifically, three usability attributes, *effectiveness*, *utility*, and *learnability*, were targeted by 8 different questions. Each of these questions was asked while using the app and was also re-asked in the post-experiment survey presented upon exiting it. In addition, 3 more questions about users' *satisfaction* with UPPLA, in terms of whether they would like to see it deployed instead of the current settings, were added to the post-experiment survey. Figure 3 summarizes the obtained results by presenting the distribution, in percentage of participants on the y-axis, for each usability attribute over the five possible answers. Moreover, the numerical total average is presented for each of these attributes (the number between parentheses in front of the labels). As we can see on the figure, participants perceived UPPLA as being highly usable over different dimensions. First, for the *effectiveness* attribute, that concerns the strength of UPPLA in correctly modeling users' privacy requirements, about 77% of the participants answered with a yes or a strong yes, and more than 95% answered with at least a moderate level. Second, 93% of the participants agreed with at least a moderate level (64% with at least a high level - *yes*) on the utility of UPPLA that reflects its added value compared to the existing methods. Third, about 80% of the participants expressed at least a high level (*yes*) of learnability for UPPLA, reflecting by this its ease and efficiency of use. Finally, More than 55% of the participants were highly satisfied with UPPLA and more than 83% of them answered that they would like to see such a method deployed in their Facebook accounts. Additionally, over all the usability attributes, the highest scored percentage for the lowest feedback is given only with a low answer (*no*) and is of 6% only.

6. RELATED WORK

Access control (AC) is one of the core building blocks of any information system. AC can be defined as the mechanism that controls granting and/or denying *subjects* to access target *objects* [11]. Two of the main AC models are Discretionary AC (DAC) and Mandatory AC (MAC). DAC is based on subjects' identities and on a set of policies that specify the objects that a user is authorized to access in the system. DAC is known for its flexibility in the range of configuration requirements that it allows. As such, DAC makes the widely deployed AC model in current commercial and industrial information systems. However, DAC does not provide security guarantees on information flow as subjects can rewrite acessed objects so as to make them available to a wider audience. This is where MAC comes into play as it controls information flows by imposing access rules based on subjects and objects classification levels.[11]

MAC was originally designed by Bell and LaPadula (BLP) who suggested managing AC by assigning ordered sensitivity levels to objects (specifying their classification) and ordered clearance levels to subjects (expressing their access privilege) [3]. The confidentiality of information and the control of its flow are then protected by two principles: 1) no read-up and 2) no write-down. The first principle enforces that subjects can only read objects that are at most equal to their clearance levels; whereas the second states that subjects write information only to at least equal levels than their owns. This ensures that information can never flow against the levels set in the system, as subjects having ac-

cess to confidential information cannot, by mistake or maliciously, make it available to other subjects at lower levels. In the past, MAC has been mainly used in domains where data confidentiality is of high importance, such as the military, and is generally practical for systems that require securing information flows, and where both subjects and objects can be statically classified by an administrator with minimum or no change [11]. Today, and despite its age, BLP is still a cornerstone of modern computer security that is being utilized and built upon for the creation of AC policies in different systems and domains [23, 21].

Indeed, today's new information domains, where highly confidential data is often intermingled with much less sensitive data and where information might take uncontrollable flows, introduced new AC requirements of providing both strong security and flexibility. Accordingly, some suggestions were made to take advantage of the strengths of MAC all while introducing more flexibility to its settings [21, 20]. For instance, Oracle Inc. has introduced the Oracle Label Security (OLS) that allows managers of tables in a database to implement MAC under the allowed discretionary privileges that are in the hands of legitimate users, offering by this row-level AC granularity [21]. By OLS, in alignment with MAC, each row in a table can be associated with a security label that reflects its classification level. Similarly, users are associated with clearance labels and AC is enforced based on MAC defined properties. OLS, and other similar systems (such as [20, 17, 22]) exploited MAC to introduce flexible AC systems that, still provide a level of information flow security, and that offer to users a framework of defined security levels where they can safely operate. This results in combining the benefits of strong security offered by MAC with flexibility in operation.

The OSNs scenario provides a good example of information domains where information flow requires both flexibility and controllable strong security. Typically, AC in OSNs has been enforced using a relationship-based approach, a form of DAC, allowing users to define access policies to their objects based on the types of relationships they establish with their peers [11]. The research community in this domain provides number of proposals that can be generally grouped in two categories: 1) a cryptography based approach, and 2) a trust based approach. For the first category, proposals based on ensuring strong security guarantees by means of data encryption and focused on finding usable ways for the dissemination, management, and revocation of the related security keys [15, 2, 8, 4], as well as on deploying cryptography techniques that can offer some levels of granularity, such as the proposals in [2, 4, 16]. Given the amounts of data and the number of connections users create and maintain in OSNs, and with the required granularity and changing requirements for privacy settings, this crypto-based approach could not overcome its by-design inherent limitations especially in terms of efficiency [5]. The second approach builds its rationale on controlling information flow based on the trust users have in each other. For instance, authors of [6] suggest a rule based AC that models authorizations based on type, depth, and aggregate trust of the path between requestor and object owner. In [1], authors adopt a multi-level AC approach where the security levels assigned to both users and resources are specified on the basis of trust levels. Users are associated with security levels that are computed based the trust values assigned to her/him by other users in the system and objects inherit the security levels of their owners. AC is then enforced according to a challenge-response based protocol. The main problem with this proposal is the cost of computing and updating security levels given the dynamic nature of OSNs and the frequent changes that users trust levels might be subject to. Other works suggested the deployment of game-based techniques for AC management in OSNs, such as [14], but all these remain bound to a pure relationship-based approach that does not consider the relationships between OSN objects, differences between their types, and between the interactions allowed and the new settings they result in.

Differently from these approaches, but similar to adaptations of flexible MAC, our suggested method exploits a label-based strategy centered on users as individual data owners. Our labels are designed to answer the specificity of OSNs AC requirements and consider more relevant features, compared to standard sensitivity and clearance levels, that offer more flexibility in expressing privacy needs. Moreover, our method's axioms and properties are designed to cover the richer interactions set that OSNs allow in addition to the classical read and write privileges. Our model also accounts for those special interactions that result in the creation of objects by subjects that should not be their owners, a scenario that, to the best of our knowledge, has not been addressed yet in other AC systems.

7. CONCLUSION AND FUTURE WORK

In this paper, we presented an access control mechanism for Online Social Networks that adopts the core concepts of MAC and models them in a way that respects and answers the specific requirements of OSNs in terms of the types of objects, interactions, and information flows that take place within their realms. Our method is both efficient and usable, as obtained from results of preliminary usability experiments, and it ensures more flexibility to OSN users in expressing and enforcing their privacy preferences. Besides, our method is designed to fit within the different types of OSNs available nowadays.

As a natural continuation to the work, we plan to extend our usability experiments both in terms of the number of participants and the completeness of the actions and interactions available on the application. We also plan to extend the model to account for decentralized social networks, cutting by this the role of the central authority. Moreover, we plan to design accompanying mechanisms for the automatic assignment of friend and object labels based on general policies

that the users can set or on learning strategies. This would improve the method by making it friendlier and lighter to use for the OSN users.

8. REFERENCES

[1] B. Ali, W. Villegas, and M. Maheswaran. A trust based approach for protecting user data in social networks. In *Proc. of the conf. of the center for advanced studies on collaborative research*. IBM Corp., 2007.

[2] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin. Persona: an online social network with user-defined privacy. In *ACM SIGCOMM Computer Communication Review*. ACM, 2009.

[3] D. E. Bell and L. J. LaPadula. Computer security model: Unified exposition and multics interpretation. *MITRE Corp., Bedford, MA, Tech. Rep. ESD-TR-75-306, June*, 1975.

[4] O. Bodriagov, G. Kreitz, and S. Buchegger. Access control in decentralized online social networks: Applying a policy-hiding cryptographic scheme and evaluating its performance. In *Pervasive Computing and Communications Workshops (PERCOM Workshops)*. IEEE, 2014.

[5] S. Buchegger and A. Datta. A case for p2p infrastructure for social networks-opportunities & challenges. In *Wireless On-Demand Network Systems and Services, WONS. Sixth International Conference on*. IEEE, 2009.

[6] B. Carminati, E. Ferrari, and A. Perego. Enforcing access control in web-based social networks. *ACM Transactions on Information and System Security (TISSEC)*, 2009.

[7] T. F. Claypoole. Privacy and social media. `http://www.americanbar.org/publications/blt/2014/01/03a_claypoole.html`. Accessed: 2015-05-29.

[8] L. A. Cutillo, R. Molva, and T. Strufe. Safebook: A privacy-preserving online social network leveraging on real-life trust. *Communications Magazine, IEEE*, 2009.

[9] E. Y. Daraghmi and S.-M. Yuan. We are so close, less than 4 degrees separating you and me! *Computers in Human Behavior*, 2014.

[10] R. Dunbar. *How many friends does one person need?: Dunbar's number and other evolutionary quirks*. Faber & Faber, 2010.

[11] E. Ferrari. *Access Control in Data Management Systems*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2010.

[12] P. W. Fong and I. Siahaan. Relationship-based access control policies and their policy languages. In *Proceedings of the 16th ACM symposium on Access control models and technologies*, pages 51–60. ACM, 2011.

[13] H. Gao, J. Hu, T. Huang, J. Wang, and Y. Chen. Security issues in online social networks. *Internet Computing, IEEE*, 2011.

[14] C. Griffin and A. Squicciarini. Toward a game theoretic model of information release in social media with experimental results. In *Security and Privacy Workshops (SPW), 2012 IEEE Symposium on*. IEEE, 2012.

[15] S. Jahid, P. Mittal, and N. Borisov. Easier: Encryption-based access control in social networks with efficient revocation. In *Proc. of the 6th ACM Symposium on Information, Computer and Communications Security*. ACM, 2011.

[16] S. Jahid, S. Nilizadeh, P. Mittal, N. Borisov, and A. Kapadia. Decent: A decentralized architecture for enforcing privacy in online social networks. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on*. IEEE, 2012.

[17] S. Jajodia and B. Kogan. Integrating an object-oriented data model with multilevel security. In *Research in Security and Privacy, 1990. Proceedings., 1990 IEEE Computer Society Symposium on*. IEEE, 1990.

[18] G. Liu, Y. Wang, and M. A. Orgun. Trust transitivity in complex social networks. In *AAAI*, 2011.

[19] M. Madejski, M. L. Johnson, and S. M. Bellovin. The failure of online social network privacy settings. *Columbia University Academic Commons*, 2011.

[20] C. J. McCollum, J. R. Messing, and L. Notargiacomo. Beyond the pale of mac and dac-defining new forms of access control. In *Research in Security and Privacy, 1990. Proceedings., IEEE Computer Society Symposium on*. IEEE, 1990.

[21] ORACLE. Label security administrator's guide. `http://docs.oracle.com/cd/B19306_01/network.102/b14267/intro.htm`. Accessed: 2015-05-29.

[22] P. Samarati, E. Bertino, A. Ciampichetti, and S. Jajodia. Information flow control in object-oriented systems. *Knowledge and Data Engineering, IEEE Transactions on*, 1997.

[23] O. S. Saydjari. Multilevel security: reprise. *Security & Privacy, IEEE*, 2004.

[24] Statista. Average number of facebook friends of users in us as of feb. 2014, by age group. `http://statista.com/statistics/232499/americans-who\-use\-social\-networking\-sites-several-times-per-day`. Accessed: 2015-05-29.

[25] Statista. Leading social networks worldwide as of march 2015. `http://statista.com/statistics/272014/global\-social-networks\-ranked-by-number-of-users`. Accessed: 2015-05-29.