Privacy-as-a-Service: Models, Algorithms, and Results on the Facebook Platform

E. Michael Maximilien*, Tyrone Grandison*, Tony Sun+, Dwayne Richardson+, Sherry Guo+, Kun Liu*

*IBM Almaden Research Center, 650 Harry Road, San Jose, CA 95120 USA

+IBM Silicon Valley Lab, 555 Bailey Road, San Jose, CA 95141 USA

{maxim, tyroneg, tonysun, drichardson, sxguo, kun}@us.ibm.com

Abstract

The current set of social networking platforms, e.g. Facebook and MySpace, has created a new class of Internet applications called social software. These systems focus on leveraging the real life relationships of people and augment them with the facilities and the richness of the Web. However, social platforms and software are not without drawbacks and significant concerns. One of the most important considerations is the need to allow strong security and privacy protections. In addition, these protections need to be easy to use and apply uniformly across platforms and applications. This paper provides a first step in resolving these issues.

1. Introduction

Social networks are the current craze. People, young and old, are conducting a large part of their lives online. While these social utilities are thriving and gaining significant traction --- e.g. Facebook's daily active user count in March of 2009 surpassed 175 million --- a clear issue that has yet to be satisfactorily resolved is how users of these social networks and social applications can easily, uniformly, and effectively control the privacy of the data that they are adding, contributing and sharing. For example, in the case of Facebook:

1) the privacy model and engine does not prevent social applications (coming from heterogeneous developers) from collecting additional data from users nor does it help these application developers to easily build privacy functionality into their applications.
2) the growing number of privacy settings multiplied by number of social applications, present a significant cognitive burden on end-users who typically accept the defaults and do not revisit their options until damage is done.
3) the mechanisms for privacy settings are primitive at best, are mostly manual, and do not take full advantage of the social and trust relationships that users build.

In this paper, we present initial work that begins to address these issues. We describe a framework, service, privacy model and algorithm for social platforms and applications that enables the concept of Privacy-as-a-Service (PaaS). We have implemented our PaaS framework initially on the Facebook platform and have deployed a live application[1] to showcase its features and to enable further refinement of the system.

2. Background, Survey, and Motivation

Facebook allows users to completely block another user from interacting with them. It also allows users to tweak the privacy of other functionalities in the system in the following categories: Profile, Search, News Feed and Wall, as well as Applications. For each category, there are sub-categories, which allow users to even further distill their privacy elections. For instance, for the Search settings, a user can select that their profile appear in search results of: *Everyone, My networks and friends of friends, My networks and friends, Friends of friends, Only friends, or a combination of the above.* In addition to the elections that users make, particular data and activities have limited access in Facebook.

To determine the sensitive aspects of a user's profile, we conducted a user survey. The explicit goal of the survey was to determine what information potential users of online social networking sites were willing to expose and to whom. The survey was done via the online tool Survey Monkey. We received 153 complete responses from 18 countries/political regions. Among the participants, 53.3% are male and 46.7% are female, 75.4% are in the age of 23 to 39, 91.6% hold a college degree or higher, and 76.0% spend 4 hours or more everyday surfing online.

Included in the survey were questions intended to ascertain individual user's privacy concerns surrounding information commonly listed in the profiles of online social networking sites. To provide users with a frame of reference, each was asked to consider their answers with respect to Facebook. Since Facebook strongly encourages real-world identification with the online persona, basic demographic information, such as name, is generally available. This observation was confirmed by our study, where nearly (60%) of respondents were comfortable providing visibility of their first name to everyone.

---

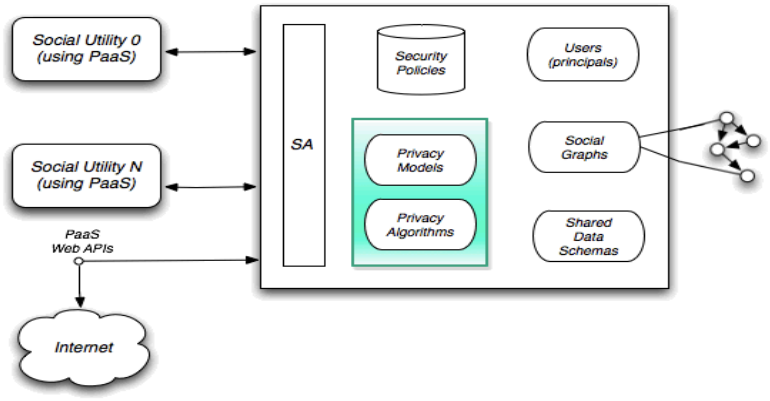
[1] http://apps.facebook.com/p_a_m_p

Other attributes that users felt they would make readily available to “Everyone” were last name (48%), gender (57.8%) and a photo (37%) of themselves. Individuals were willing to expose birthday (37.7%), birthday with year (29.2%) hometown (35.1%), relationships status (33.8%) and name of spouse or partner (31.8%) to “Friends” albeit at lower percentages than simply their first and last names. Information that most users felt was the most private and should be exposed to “No One” included mother’s maiden name (73%), gender interested in (34.4%), type of relationships sought (35%) and religious views (29.9%). The information derived from this survey was critical in the algorithm portion of our work, which will be presented in forthcoming sections.

3. PaaS: Architecture, Model and Algorithm

For our purposes, a social utility refers to a social network or a social network application. Figure 1 shows the architecture of a typical PaaS system.

Figure 1 The PaaS framework



The service consists of the following eight components:

1. a Security Assistant (SA) that ensures that access to the information in the PaaS server strictly follows the rules in the Security Policies repository;
2. a set of security rules that store the social utility's reference information, their associated credentials, a list of the information that the utility can retrieve;
3. a directory of privacy principals, e.g., users;
4. a graph of relationships between principals;
5. a collection of data schemas shared between principals, e.g., profile data;
6. a collection of privacy index algorithms that can return the privacy index of a user for any piece of data that the user is trying to view or expose. We will explain the concept of a privacy index in the next few subsections;
7. a collection of privacy models that contain the means for users to make elections between other users in their graphs (based on relationships, e.g., friend, friend of friends, networks, and so on) as well as a specific privacy algorithm to be used;
8. a collection of Web APIs exposing the main functions of the privacy system such that it can be remotely invoked and incorporated (in a secure manner) into existing systems that do not have privacy concerns realized or solved.

3.1 Model

First, we define a model for an arbitrary social network, which we assume is a set of interconnected entities and containers. Entities are the primary artifacts of a social network, i.e., users, and containers are special structures formed around these entities to foster a community, activity, or for greater purposes, e.g., a social network applications, groups, and networks. We assume that entities may opt to be members of containers and that entities interact with other entities and with containers.

Social Entity, Descriptor, and Container: In our context, a social entity will be referred to as *se*. The set of all entities for this particular social network, *E*, is $\{se_1,\ldots,se_x\}$, where *x* is the total number of entities in the network.

We assume that *d* is a descriptor that is used to describe the attribute utilized to create the profile for an entity. *d* is a tuple of the form {*d_name, d_type*}. The set *D* is the complete set of descriptors used to describe a particular entity and is equal to $\{d_1,\ldots,d_n\}$, where *n* is the total number of descriptors needed to describe this particular entity. We also assume that *D** is the universal set for *D*. Each entity can be described by a set of descriptors (i.e. attribute-value pairs), e.g. {(name, "Sam"), (birth_date, 09/09/1988)}. Formally, $\forall u \in E^{®}$ $((D_u \in D^*) \wedge (D_u$=state(u)) where D_u=state(u) means that D_u accurately describes the current state of *u*.

A container *c* is the set $\{ \{a_1,\ldots,a_m\}, \{u_1,\ldots,u_p\}, D_c, \{D_{u1},\ldots,D_{um}\} \}$, where $\{a_1,\ldots,a_m\}$ are administrators of the container, $\{u_1,\ldots,u_p\}$ are the users of the container, D_c is the set of descriptors for the container and $\{D_{u1},\ldots,D_{um}\}$ is the data on the users of the container. It should be noted that $\{a_1,\ldots,a_m\} \subseteq \{u_1,\ldots,u_p\}$ and $m \leq p$. We define *C* as the universal set of all containers in the network. We also define a set of applications ($A \subseteq C$), groups ($G \subseteq C$) and networks ($N \subseteq C$).

Privacy: For each profile item, users set a *privacy level* that determines their willingness to disclose information associated with this item. The privacy levels picked by all *N* users for the *n* profile items are stored in an $n \times N$ response matrix R. The rows of R correspond to profile items and the columns correspond to users. We use $R(i,j)$ to refer to the entry in the *i*-th row and *j*-th column of R, *i.e.,* $R(i,j)$ refers to the privacy setting of user *j* for item *i*. If the entries of R are restricted to take values in $\{0,1\}$, we say that R *is a dichotomous response matrix.* Otherwise, if $R(i,j)$ takes any non-negative integer values in $\{0,...,l\}$, we say that R *is a polytomous response matrix.*

In a dichotomous response matrix R, $R(i,j)=1$ means that user *j* has made the information about profile item *i* publicly available, whereas $R(i,j)=0$ means that user *j* has kept the item *i* private. The interpretation of values appearing in polytomous response matrix is similar: $R(i,j)=0$ means that user *j* does not share item *i* with any one while $R(i,j)=k$ with $k \in \{1,...,l\}$ means that *j* discloses item *i* to other users that are at most *k*-hops away in the social graph.

3.2 Privacy Algorithm

The *privacy index* (or the *privacy risk score*) of a user quantifies the user's privacy risk caused by his privacy settings. The basic premises of the definition of privacy risk are the following: 1) the more sensitive information a user reveals, the higher his privacy risk, and 2) the more people know some piece of information about a user, the higher his privacy risk. We define the privacy risk of user *j* to be a *monotonically increasing* function of two parameters: the *sensitivity* of the user's profile items, and the *visibility* these items get.

Sensitivity of a profile item: We use β_i to denote the sensitivity of *item* $i \in \{1,...,n\}$. We note that this sensitivity depends on the nature of the item itself. For example, one's *mother's maiden name* is usually considered more sensitive than his *work phone number*.

Visibility of a profile item: The visibility of a profile item *i* due to user *j* captures how widely known the value of *i* becomes in the social network; the more it spreads, the higher the item's visibility. Naturally, the visibility, denoted by $V(i,j)$, depends on the user's privacy level setting for item *i,* $R(i,j)$. The simplest possible definition of visibility is $V(i,j) = \mathrm{I}_{(R(i,j)=1)}$, where $\mathrm{I}_{\text{condition}}$ is an indicator variable that becomes 1 when "condition" is true. We call this the *observed visibility* for item *i* and user *j*. In statistics, one can assume that R is a sample from a probability distribution over all possible response matrices. Thus, we can compute the *true visibility* by using the formula $V(i,j) = P_{ij} \times 1 + (1 - P_{ij}) \times 0 = P_{ij}$, where $P_{ij} = \mathrm{Prob}\{R(i,j)=1\}$. It is obvious that probability P_{ij} depends both on the item *i* and the user *j*.

Privacy risk of a user: The privacy risk of individual *j* due to item *i*, denoted by $\mathrm{PR}(i,j)$, can be any combination of sensitivity and visibility. That is, $\mathrm{PR}(i,j)=\beta_i \otimes V(i,j).$ Operator $\otimes$ is used to represent any arbitrary combination function that respects the fact that $\mathrm{PR}(i,j)$ is monotonically increasing with both sensitivity and visibility, e.g., product. In order to evaluate the overall privacy risk of user *j*, denoted by $\mathrm{PR}(j)$, we can combine the privacy risk of *j* due to different items. Again, any combination function can be employed to aggregate the per-item privacy risks. For simplicity, we use summation operator here. That is, we compute the privacy risk of individual as follows:

$$\mathrm{PR}(i) = \sum_i \mathrm{PR}(i,j) = \sum_i \beta_i \times V(i,j) = \sum_i \beta_i \times P_{ij} \quad . \qquad (1)$$

3.2.1 Privacy risk computation

From Equation 1, we can see that in order to compute the privacy risk $\mathrm{PR}(j)$, we need to know the values of sensitivity $\beta_i$and visibility P_{ij}. In this section, we provide a simple way of doing this.

Computation of sensitivity: The sensitivity of item *i,* β_i, intuitively captures how difficult it is for users to make information related to item *I* publicly available. If $|R_i|$ denotes the number of users that set $R(i,j)=1$, then the

sensitivity of item *i* is computed as the proportion of users that are reluctant to disclose item *i*. That is, $\beta_i = \frac{N - |R_i|}{N}$ (2). The sensitivity as computed in Equation 2 takes values in [0,1]; the higher the value of β_i, the more sensitive item *i*.

Computation of visibility: The computation of visibility requires an estimate of the probability $P_{ij} = \text{Prob}\{R(i,j) = 1\}$. Assuming independence between items and users, we can computer P_{ij} to be the product of the probability of an 1 in the *i*-th row of R and the probability of an 1 in the *j*-th column of R. That is, if $|R^j|$ is the number of items for which *j* sets $R(i,j) = 1$, we have $P_{ij} = \frac{|R_i|}{N} \times \frac{|R^j|}{n} = (1 - \beta_i) \times \frac{|R^j|}{n}$ (3).

Probability P_{ij} is higher for less sensitive items and for users that have the tendency to disclose lots of their profile items. The overall privacy risk score (aka privacy index) is obtained by applying Equations 2 and 3 to Equation 1.

4. Privacy-aware Marketplace (PaMP)

PaMP allows one to create posts that are related to items for sale, housing, and jobs. Amongst other things, PaMP may be used to enable private postings (e.g., the resale of holiday gifts received from family and friends without embarrassing them) and targeted marketing (e.g., setting the visibility or target audience for one's ads). PaMP has two types of users: ordinary and administrators. Ordinary users can create and search postings and set privacy settings based on the privacy risk score (Figure 2). A user is able to view their settings and can see their current privacy index and a recommendation of a privacy index based on the other users in her network (Figure 2).

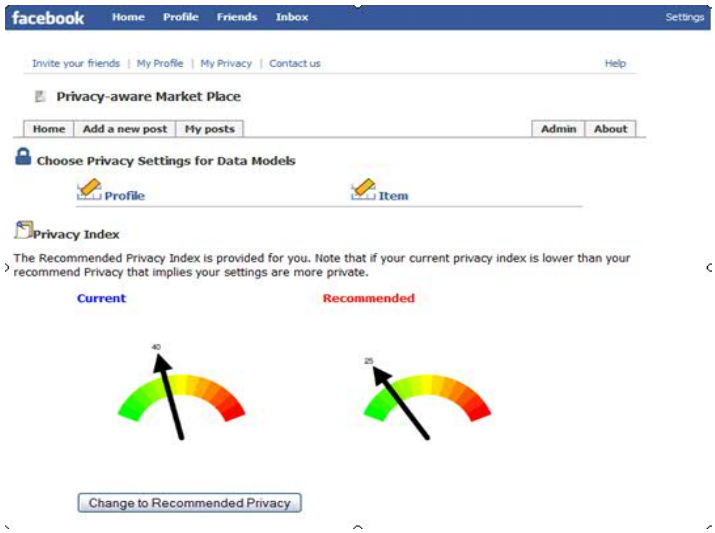


Figure 2 User Privacy Score Index and Recommendation

Administrators are required to perform initial setup and have the ability to add and or modify the underlying privacy algorithms associated with each model and which data attributes are taken into account and which are ignored.

5. Directions

We have presented a framework, service, model, and algorithm that is a start in addressing the current shortcomings with social platforms and applications. Through a non-trivial exemplar social software we demonstrated that our approach can be implemented for a large social platform. The generation of a privacy score built on the collective wisdom of users and the associated recommendation service help us address problems (2) and (3).

While our approach is not conceptually restricted to one particular social platform, the current implementation is specific to Facebook and one particular social application framework. We want to further prove the framework in two axes. First, we would like to demonstrate that the framework is agnostic to social application platforms, e.g., OpenSocial. Second, we also want to expose parts of our framework as a collection of REST APIs which would allow PaaS to truly be universally available on the Web.