Privacy through Web-Traveler Policies in Social Network Environments

Smitha Sundareswaran and Anna C Squicciarini
Information Sciences & Technology
Pennsylvania State University
University Park, PA
Email: {sus263,asquicciarini}@ist.psu.edu

***Abstract*— Social networking sites are ingrained in the fabric of our day-to-day lives, with these sites being widely used to exchange personal information and content. Some forms of access control are provided by the sites themselves to protect the user-uploaded content. However, these types of control are limited in that they require the user's input for the effective protection, otherwise a default policy which provides minimal protection is often applied. Towards providing extended content protection, in this paper we propose an approach for automated user-uploaded content control. Automatic enforcement allows us to extend the protection of content for unprotected files, preventing underage viewers from accessing adult content, tracking stolen or misused copyright-free images. This work builds upon the notion of ' Web-Traveler policies', previously introduced as a new class of content control policies. In the paper, we also provide a proof of concept implementation of our algorithms for automatic propagation for images in order to prove the applicability and strength of our approach.**

***Index Terms*— Privacy, collaboration, Web 2.0, folksonomies**

I. Introduction

Social Networks (SNs) are used as a daily form of communication among users, to exchange information and share various types of content, such as pictures and videos. These shared data items are a gold mine of detailed personal information about the users' social habits, traits and behavior [15]. Although SN and photo sharing websites provide mechanisms and default configurations for data sharing control, they offer very limited control of the content to the user and are often ineffective. The current controls apply to the user's content only as long as it is within his profile. Once this content is disclosed to other users, they can upload it on their profile and then the original user has no control over it. For example, another user who can view some exposed content can download it and then upload a copy of the content on his profile with more public settings and the original user has no control over who can access this even though the content belongs to him. This problem is compounded by the fact that many users often do not set the policies for all of their content as they find the process cumbersome[3] and time consuming, as shown by several studies[1]. As a result, user-provided content can been easily stolen, sold, and used for blackmailing, leading to serious cyber crimes, such as identity theft and financial losses[5], [23].

In this paper we explore how to achieve strong content protection and how to manage traits automatically to sensitive user-uploaded content. By strong content protection, we mean being able to control not only who access the content but also what operations the content recipient is allowed to perform on it.

Our approach toward extended content protection builds on the notion of *Web-Traveler* policies [21] previously introduced by us. Web-traveler policies can seamlessly travel with the content, as they were attached to it. Policy propagation means applying these policies to unprotected content. Propagation is achieved by analyzing personal annotations, i.e., tags, and by considering content similarity. By deploying such extended content protection, we show how our approach can help prevent underage viewers from accessing adult content, and tracking misused copyright-free data.

We study the application of Web-Traveler to images, and focus on critical issues such as management of special content, and how to deal with the case of users who do not explicitly attach a policy to their content upon upload. In the paper, we also show the feasibility of this approach by implementing extending our initial prototype on a real open source SN platform. We conduct performance and scalability tests of our architecture.

The remaining of the paper is organized as follows. In Section 2, we provide an overview of the infrastructure. In Section 3, we describe the automated propagation of policies to protect unprotected content and special content such as adult content and copyright-free content. In Section 4, we provide an initial evaluation of our prototype followed by related work in Section 5 and concluding remarks in Section 6.

II. Overview of Web-Traveler Infrastructure

The Web-Traveler policies specify access control requirements for a given image and are enforced within the boundary of closed domain, such as a SN site. They can be extended for other file types. Each policy defines the access conditions of the object being protected and the users to whom they belong. They are encoded using XML as shown in Appendix A. The conditions in the policies can be expressed based on either the relationship between users, such as friends or colleagues, or based on users' attributes, like group membership or city of residence.

Web-Traveler policies apply to the five basic operations possible for a given object posted on a SN site: view, upload,

COLLABORATECOM 2011, October 15-18, Orlando, United States

DOI 10.4108/icst.collaboratecom.2011.247169

download , tag and comment. Access rights for these operations are enforced separately, whenever possible. A policy can for example state that Alice's picture $Pic1$ can be viewed and downloaded by Alice's friends, but that they cannot upload it in their profile. Policies are enforced under the assumption that the user who originally uploaded the content within the SN is in charge of specifying a protection policy for it[1]. Clearly, the privileges for some of the operations are dependent on the others. For example, Alice to download an image Pic, she must be able to view it. However, the reverse is not necessarily true.

We extend the Web-Traveler policies in this paper by automatically protecting two categories of user-uploaded content: 1) content for which there are no policies defined and 2) special content, i.e. either adult content or copyright free content.

Automatic propagation of policies is built in with the upload protection of the initial Web-Traveler infrastructure. After ensuring that the content being uploaded is not a copy of some other user's content, the extended Web-Traveler infrastructure verifies whether the content being uploaded can be classified as either adult content or copyright free content, using a copy-detection mechanism similar to the one used in our initial prototype. If none of the cases of above apply, i.e., the object the user is trying to upload is not already under others' protection -and it does not fall into a special class of objects- the user can directly enter the policy using a simple web interface. If the user fails to enter the policy for the object, then the Web-Traveler privacy preferences of the user are inferred and the closest applicable policy is enforced to the newly loaded object on behalf of the user.

III. Extended privacy protection through policy propagation

In what follows we present the solutions we have devised to propagate user-entered Web-Traveler policies on behalf of end users. We examine in depth, the notion of similarity between files and discuss how to enforce policies in special cases, where a policy is propagated based on the file's content, rather than set by the user.

A. Automated Propagation of Web-Traveler policies

In order to guarantee appropriate privacy protection of files even in case users do not indicate Web-Traveler policies, we propose a propagation algorithm to apply customized protection to user-uploaded files. Given an uploaded file t of user u, if the user does not specify his own policy for t, then the Web-Traveler system selects the most similar user-owned file previously uploaded by u, say t', so as to let t''s policy travel with t.

Identifying the most similar file, the policy of which, can be applied to the new content presents a few challenges. In order to well represent users' intentions, objective criteria may not be sufficient. From an end user standpoint, two files may be "similar", not only based on actual content, but also based on other, subjective dimensions. For example, images taken during a trip may be different with one another, but semantically correlated for the user, and hence require a similar treatment, in terms of privacy.

In order to ensure that the policies propagated are in accordance with the users' preferences while still preserving objective similarity criteria, we use two main notions of similarity: i) the *semantic* similarity of data, as indicated by users through semantic tags; ii) the *actual* content similarity.

a) Semantic-based similarity: Semantics-based similarity is defined based on user-created folksonomies.

Folksonomies leverage the words [16], referred to as tags, that users often associate to their content so as to give a context or a topic to it. For simplicity we focus on the case where users add up to k tags per each object. As such, for a given object, we associate at most k tags, $\{t_1, \ldots, t_k\}$. This meta-data is used to conduct similarity analysis with objects of the same kind previously posted by the same user. We represent each object as a vector of (non-empty) tags. That is, let T=$\{\overrightarrow{t_1}, \overrightarrow{t_2}, ..., \overrightarrow{t_n}\}$ be a set of objects of type O controlled by u. Let $\overrightarrow{t}$ be the object whose policy is to be defined. In order to identify the best policy to associate with $\overrightarrow{t}$, we conduct similarity analysis among the objects in T and $\overrightarrow{t}$. To this extent, we rely on the informal classification system resulting from the practice of collaborative tagging. This user-generated classification system, is referred to as *folksonomy* [16], and is generally defined in terms of a collection of posts, each associated with one or more tags.

By relying on a folksonomy, we can compare any two files and assign them a similarity score, based on the tags associated with each of them. Tags' relatedness can be constructed according to several metrics [18], [11]. In our case, we employ the following modified notion of co-occurrence of tags. We consider the tags associated to a given object $\overrightarrow{t}$ by users in U. Given the set of objects $\mathcal{T}$ posted by U, the weight of a tag pair, say $t_1,\ t_2$ is given by the number of times the pair is tagged to describe the same object.

$$h(t_1; t_2) := card\{T | t_1; t_2 \in T_{ur}, T \in \mathcal{T}\} \tag{1}$$

For a given tag $t \in T$, the tags that are most related to it are thus all the tags $t' \in T$ with $t' \neq t$ such that $h(t, t')$ is maximal. Notice that in equation 2, we do not restrict to any specific subset of the users in the SN, although it is straightforward to concentrate on the similarities of tags as perceived by a smaller set of users, e.g. user u friends, users in the same network etc.

Based on these notions, we define *tag-based similarity* of files as the overall relatedness among the tags associated with the objects. Given two objects $\overrightarrow{t}$, $\overrightarrow{t'}$ their tag-based similarity (τ_sim) is determined as follows.

$$\tau\_sim(\overrightarrow{t}, \overrightarrow{t'}) = \sum_{i=1}^{k} \sum_{j=1}^{n} h(t_i, t'_j) \tag{2}$$

The equation 2 returns a similarity value expressed as non-negative number. Based on this value, we can extract the short list of similar items to be considered: $L = \{\overrightarrow{t}_1, \ldots, \overrightarrow{t}_n\}$.

[1]Issues related to co-ownership and legal rights to handle private images are beyond the goal of this work.

b) Content-based similarity: As tags are completely under users' control, they can capture the user's perceived similarity between objects, leaving room for errors and possible ambiguity. The tag-based analysis may result in multiple potential candidates for policy propagation, or on the other hand, it may not be significant, if tags are seldom used by the object owner. To overcome these limitations, we take an additional step to refine the results obtained by doing tag-based similarity, and leverage copy detection techniques (already employed by the Web-Traveler infrastructure for controlling upload of existing images). That is, we associate to the items in the list L an additional similarity score, based on the object's features, so as to identify the best object to be used for propagation. While the specific copy detection algorithm varies according to the specific object type ([10] is suitable for images), the same approach can be used across all content types. First, the copy detection algorithm returns a normalized similarity score, $\delta^t_{t'}$, to each object $t' \in L$ with respect to t. Second, the $ProP$ algorithm (see Algorithm 1) is executed, so as to identify and select the policy of the most similar image among L, based on the actual similarity values returned running the copy algorithm. In a snapshot, the $ProP$ algorithm works as follows. Consider the case of a short list of objects similar to t, returning t' and t'', respectively[2]. The $ProP$ algorithm proceeds as follows:

- If there is a significant difference in the percentage of similarity to the newly uploaded object between the most similar object, t' and the second most similar one, t'' (i.e., $\delta^t_{t'} >> \delta^t_{t''}$), then t's policy is applied to the newly uploaded object (i.e., t) irrespective of whether t' itself has a policy set by the user or whether t' has a propagated policy applied to it.
- If the difference in similarity of the two objects t' and t'' is marginal (i.e., $\delta^t_{t'} - \delta^t_{t''} < 0.005$), we check if t'''s policy is the result of a previous propagation process or if it was a user-specified policy. If t'''s policy was user-specified, it is propagated even though t' is more similar to t. Otherwise t'''s policy is propagated. This holds without loss of generalization when a lot of objects have marginal difference in the similarity score; if more than one object with a marginal difference in the similarity score is returned, then the one with the highest score having a user-specified policy is chosen.

The propagated policies help users' policy authoring tasks for the Web-Traveler policies, but are not meant to replace his decisions; and can be overridden at any time if the user specifies a policy for the object. In case the user has no previous objects of that kind on the system, he/she is asked to explicitly enter one. If he opts not to do so, a default policy is applied.

B. Propagation in case of special content

Our Web-Traveler infrastructure handles automatically cases where a user essentially attempts some illegal or illegitimate uploads, thus claiming control over objects which should be treated according to some predefined privacy protection rules. Under this category are copyright-free files and adult content. A copyright-free file is one which can be used by anyone in anyway for any purpose[4].While copyright enforcement is a recurring issue for objects protected by conventional access control policies, the problem is compounded when objects are strongly coupled with the policies (such as with Web-Traveler policies), since these policies essentially prohibit the copies of the file by being uploaded by other users in the SN. For example, uploading copyright-free file (say an image of the Mona Lisa painting), and assigning a policy disabling everyone else from uploading or using such a file should not be possible. Similar considerations apply for adult content, such as a pornographic images, which should be accessible only to users over 18 and under certain restrictions.

We prevent such privacy violations by applying default policies policies for the considered content type. On upload, before allowing the user to set a policy for the object, the copy detection algorithm specific for the data type is executed to

Algorithm 1 ProP: Algorithm for propagation of Web-Traveler policies to unprotected images

Input: List of scores of similar files $\delta^x_1, \ldots, \delta^x_n$ of the user who has uploaded the file x, where n is the number of files returned by the copy detection algorithm
$User_i$ indicating whether the policy is set by the user for the file i; $User_i = 1$ if the policy is set by the user and $User_i = 0$ if the policy is an automated policy
Output: file y, i.e. the file whose policy is propagated to the file x
$GREATEST = \delta^x_1$
for $i = 1$ to n **do**
  if $\delta^x_i > GREATEST$ **then**
    $GREATEST = \delta^x_i$
  end if
end for
for $i = 1$ to n **do**
  if $User_i = 1$ **then**
    $a(i) = \delta^x_i$
  end if
end for
$GREATEST2 = a(1)$
for $i = 1$ to n **do**
  if $a(i) > GREATEST2$ **then**
    $GREATEST2 = a(i)$
  end if
end for
if $GREATEST = GREATEST2$ **then**
  $y = GREATEST$
else
  if $(GREATEST - GREATEST2) <= 0.005$ **then**
    $y = GREATEST2$
  else
    $y = GREATEST$
  end if
end if

[2]This algorithm holds true when a larger list of objects are returned since each image in the list has a different similarity score.If there are two identical objects found then all the objects have the same policy, viz. that of the original object.

check if the content matches any special object. If any file with a significant similarity score is found in the special content data set,the default policy is applied depending on the specific file's classification (i.e. copyright-free, adult content etc). For copyright-free files, we impose a lax policy, which leaves room for free download or upload, while for files that convey adult content, we rely on a more restrictive setting, enforcing a policy that prohibits access to underage[3]. It is straightforward to apply more conservative policies, such as simply blocking material that does not appear appropriate. The applied policies are irrevocable, so as to avoid possible attempts of overriding of these controls. It should be noted that this approach can easily be extended for any other type of files requiring special management. For example, if certain files coming from specific locations are to be made public, a specific policy can be automatically set, and location information retrieved based upon the IP address of the end user.

Clearly, the challenge of guaranteeing this form of protection lies in the ability of creating a comprehensive and accurate data set, collecting a large and frequently updated number of objects having special content. To achieve this, we use Yahoo Pipes [22] to search Google's database for special content. The content itself is identified using Google's "similar" content search option and searching for content similar to previously cached special content. We omit a detailed discussion for lack of space.

IV. Prototype Evaluation

We deployed the Web-Traveler propagation infrastructure as part of the Drupal system[20], an open source content management system used to build SNs and online communities. The infrastructure builds upon the preliminary version of the Web-Traveler infrastructure. In this section, we report a few interesting test results carried out using images.Our tests were conducted using a Dell Latitude D630 Laptop, with 2G Ram and a Intel(R) Core(TM)2 Duo CPU T7500 @ 2.20GHz processor. During the tests we measured only the server response time and do not factor in possible network delays.

To evaluate the scalability of our extended infrastructure, we verified the enforcement time for policies of different complexity ranging from 50 to 150 conditions based on groups, finding an enforcement time less than 0.003 seconds, as in our previous tests. The set of tests for analyzing the scalability of our system under heavy loads initially reported that the average waiting time for about 1,000,000 users (where roughly half of them perform an action for which they are granted access and the other half are denied access for their actions), never exceeded 0.06 seconds. To test the scalability of our infrastructure, we further verified the time taken to access an image on a remote Apache server hosting a Web-Traveler enabled SN. The number of users attempting to access the image were simulated using a multi-thread program, with one thread to simulate a single user's actions, and the number of users varied between 1 and 1000. We find that the time taken increases slowly for up to about 600 users starting at

[3]The age is derived by the profile's attributes.

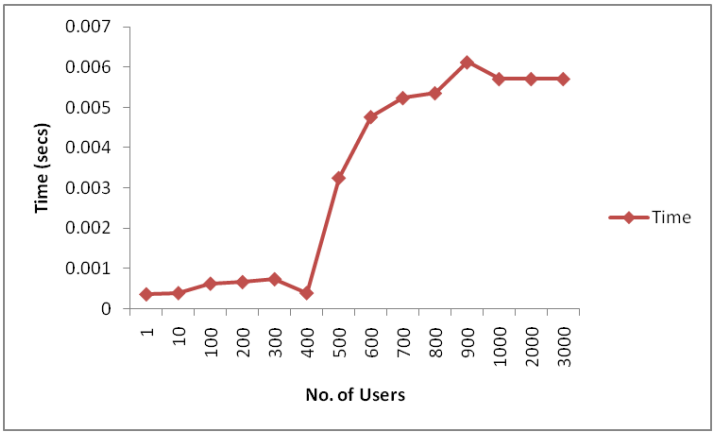

Fig. 2. Access time for apache server

0.0003572 seconds for a single user and the exponentially increases to about 0.005709 seconds after which it stabilizes as shown in Figure 2. Thus, we conclude that our architecture is scalable under a significant load even on a less responsive server like Apache.

To assess the overhead added by the propagation of policies, we compared the average time taken to set a policy both manually and automatically. Precisely, we measured the average time delay between the policies of various complexities being set by a user, and compared them with the time taken for identifying and propagating similar policies automatically. For user- entered policies, we focused on the time taken from the moment they are actually created and stored as XML files, excluding any delay due to the human input. For system's selected policies we include the time taken to create the shortlist of candidate images (we assume that each image has up to 5 tags) from a user profile having an average of 100 images, and to identify the exact image which policy is to be propagated. The time taken to identify the exact image whose policy is to propagated is about 0.06 seconds. This time is not affected by the number of images available at the user's profile, unless this number is higher of at least an order of magnitude. Even when the number of images available are higher by an order of magnitude, the times for running our extended image detection algorithm are very close to the original image detection algorithm from Jacob[10]: 0.1856 secs for about 1000 images and 0.4111 seconds for 10000 images The average time for policies set by end users is 0.01900694 secs for simple policies, which give access rights to all the users trying to access the image. The time taken for a simple relationship based policy at a very high level of granularity, created by an end user is about 0.000691 seconds as opposed to the time for propagating the policies which is 0.006194 secs. The time taken for propagating a very complex policy where each and every operation has a unique right set for it, is still the same. This is because to the Web-Traveler infrastructure, there is no difference between propagating different rights for each operation (i.e. view, download or upload) as compared to propagating the same rights for each operation. As demonstrated by these results, although the time for propagating the policies and

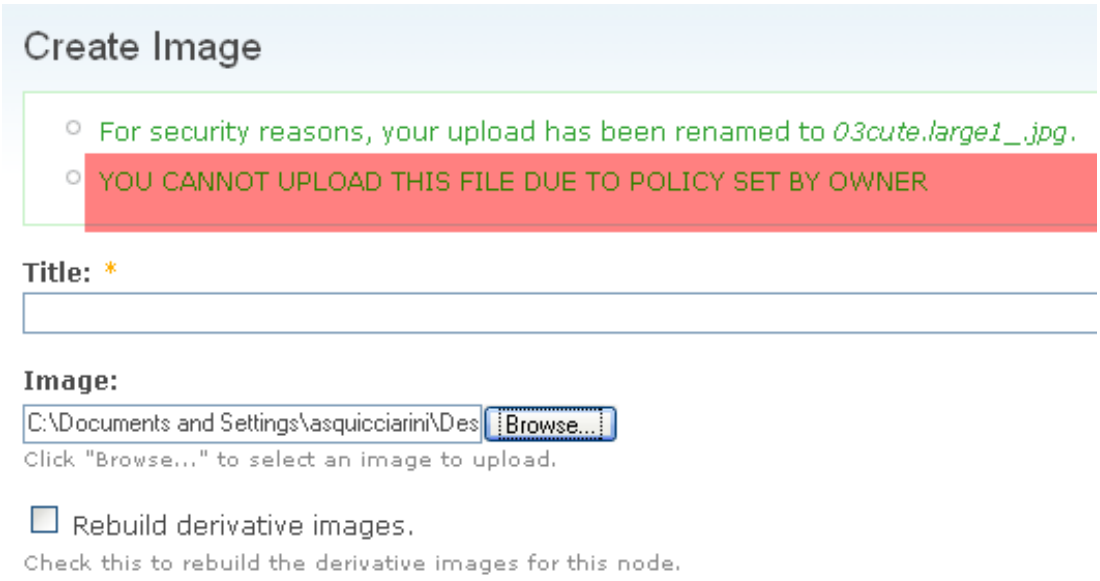

Fig. 1. Example of image upload being blocked due to the presence of a copy in the system

the time for setting automated policies for special images are higher due to the copy detection component, the overhead for end-users is quite small.

V. Related Work

Several studies have been conducted to investigate users' privacy attitudes, and possible risks which users face when poorly protecting their personal data in SNs. For example, Gross et al. [7] provide an interesting analysis insights of users' privacy attitudes across SNs. Following such considerations, we provide an approach that allows users to automatically ensure the privacy of users' uploaded data.
Web-Traveler policies were designed to address one of the access control requirements identified by Gates[6], referred to as "sticky policies", where the need of policies following the data for online content posted on SN was first indicated as a requirement for the Web 2.0. While some work has addressed the other requirements identified by Gates, such as relationship-based requirements and interoperability issues, to the best of our knowledge no other work has ever proposed a solution along this line. One of the most well-known works dealing with sticky-policy [17] is geared toward business domains, and it does not apply to online SN data. Hence, the solution from Casassa-Mont is very different from our work, in that it relies on strong requirements for the underlying software and hardware architecture. Additionally, it is only applicable within the business-domain and it applies to text and document files. We relax these strong requirements and integrate our solution in a system that provides policies to prevent inadvertent disclosure of personal information across users' profiles. Hong et al [8] examine back-end privacy based on the assumption that people will share data because they gain value from that sharing. Similar to us, they develop a traveling annotation mechanism that supports auditing the retention and use of data. While their mechanism develops a logical context data model and a physical data store, our mechanism deals with the concrete challenges of making such type of protection feasible, without relying on third parties' verification or auditing.
Concerning policy propagation based on folksonomies, our work builds on related approaches on customization and personalization of tag-based information retrieval [12], [19], [14]. Several techniques involved in exploring social annotations include association rule mining [14] and EM-based probabilistic learning approach [12], [19], [24].

Web-Traveler policies share some similarities with work conducted on digital right management and intellectual property [2]. Digital Right Management refers to access control technologies used by hardware manufacturers, publishers and copyright holders to limit usage of digital media or devices. Copy protection, instead, only attempts to prohibit unauthorized copies of media or files, digital rights management allows the issuer of the media or file to control in detail what can and cannot be done with a single instance. Although we strive to achieve a similar goal, e.g., controlling the distribution of protected content, our goal is to provide a SN enabled with easy to deploy content protection techniques, based on a flexible privacy setting interface. The users, differently from DRM work, do not have to have access to any specific device or software, to enforce our policies.

With respect to download controls, an approach similar to ours is taken by the commercial software ImageSafe [9]. This software too attempts to prevent the download of images using Java Applets. However we differ from it in that we control the download of images via screen capture or using hotkeys. Further, we aim to extend the protection to videos and text files, as well as control other actions such as viewing and upload.

VI. Conclusion and Future Work

In this paper, we presented an approach to strongly protect user's content uploaded on SN sites. One of the main features of our proposed solution is the ability to protect users' content even in case the user does not explicitly suggest a policy, and in case the content is of special nature that requires special privacy protection. Our approach is the first of this kind, in that it aims at providing back-end data privacy protection that, to our knowledge, is not supported by any of the existing solutions. The architecture not only supports automatic setting of policies, but it also ensures that special content is appropriately protected; it protects adult content from underage users while protecting copyright free content from being claimed by any one user. While our solution tackles some non-trivial challenges, there are still a number of open issues to be addressed. First, the accuracy of our approach relies in large part on the accuracy of the copy detection algorithm employed. We will investigate content-based image retrieval techniques [13], to further improve the performance and the effectiveness of copy detection detection. Secondly with regards to special images, the definition of copyright-free images or videos can be quite ambiguos. We plan to investigate how to better handle images and that are of public domain, but technically not copyrighted. Finally, the deployment of the presented techniques has several issues, which we did not discuss for lack of space. In particular, how to control unwanted download operations is an important challenge.

References

[1] A. Acquisti and R. Gross. *Imagined Communities: Awareness, Information Sharing, and Privacy on the Facebook*, volume 4258 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2006.

[2] E. J. Cohen. Drm and privacy. In *Berkley Technolog Law Journal*, 2003.

[3] G. Danezis. Inferring privacy policies for social networking services. In *AISec09, November 9, 2009, Chicago, Illinois, USA.*, New York, NY, USA, 2009. ACM.

[4] S. Fishman. *Public Domain: how to find and user copyright-free writings, musics, art and more*. Nolo, 4th Edition, 2008.

[5] Four Teens Sued for Obscene Fake Facebook Profile. http://mashable.com/2009/09/25/fake-facebook-profile/.

[6] E. C. Gross. Access control requirements for web 2.0 security and privacy. In *Position paper accepted to the Workshop on Web 2.0 Security and Privacy (W2SP) 2007, Oakland, CA,*, New York, NY, USA, 2007. ACM.

[7] R. Gross, A. Acquisti, and H. J. Heinz, III. Information revelation and privacy in online social networks. In *WPES '05: Proceedings of the 2005 ACM workshop on Privacy in the misc society*, pages 71–80, New York, NY, USA, 2005. ACM.

[8] J. I. Hong. The context fabric: an infrastructure for context-aware computing. In *CHI '02: CHI '02 extended abstracts on Human factors in computing systems. Minneapolis, Minnesota, USA.*, pages 554–555, New York, NY, USA, 2002. ACM.

[9] ImageSafe http://www.cellspark.com/imagesafe.html

[10] C. E. Jacobs, A. Finkelstein, and D. Salesin. Fast multiresolution image querying. In *SIGGRAPH*, pages 277–286, 1995.

[11] J. Jiang and D. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proc. of the Int'l. Conf. on Research in Computational Linguistics*, pages 19–33, 1997.

[12] K. Lerman, A. Plangprasopchok, and C. Wong. Personalizing image search results on flickr. *CoRR*, abs/0704.1676, 2007.

[13] A. P. Leung and P. Auer. An efficient search algorithm for content-based image retrieval with user feedback. In *ICDMW '08: Proceedings of the 2008 IEEE International Conference on Data Mining Workshops*, pages 884–890, Washington, DC, USA, 2008. IEEE Computer Society.

[14] X. Li, L. Guo, and Y. E. Zhao. Tag-based social interest discovery. In *WWW*, pages 675–684, 2008.

[15] H. Liu, P. Maes and G. Davenport. Abstract Unraveling the Taste Fabric of Social Networks, pp. 42-71, 2006.

[16] A. Mathes. Folksonomies: cooperative classification and communication through shared metadata, 2004.

[17] M. C. Mont, S. Pearson, and P. Bramhall. Towards accountable management of identity and privacy: Sticky policies and enforceable tracing services. In *DEXA Workshops*, pages 377–382, 2003.

[18] G. Pirro' and N. Seco. Design, implementation and evaluation of a new semantic similarity metric combining features and intrinsic information content. In *Proceedings of On the Move to Meaningful Internet Systems*, 2008.

[19] A. Plangprasopchok and K. Lerman. Exploiting social annotation for automatic resource discovery. *CoRR*, abs/0704.1675, 2007.

[20] The Drupal Platform. www.drupal.org. accessed february, 2010.

[21] A. C. Squicciarini and S. Sundareswaran. Web-traveler policies for images on social networks. *World Wide Web* Journal, 12(4):461–484, 2009.

[22] Yahoo Pipes. http://pipes.yahoo.com/pipes/

[23] Privacy still still a nagging concern on Facebook. http://www.boston.com/business/technology/articles/2010/02/04/privacy_still_a_nagging_concern_on_facebook/.

[24] X. Wu, L. Zhang, and Y. Yu. Exploring social annotations for the semantic web. In *WWW*, pages 417–426, 2006.

Appendix

An example of a policy is shown in Figure 3. The main components of the policy are the unique owner ID and a set of relationship-based constraints specified for each of the actions. A set of optional constraints with a value optionally specified for each are also included. The policy has two main levels; the first is the subject, which specifies the owner of the image and therefore of the policy; and the second is the target, comprised of the actions, the relationship-based constraints, the optional controls; the values for optional controls and the combining operator.

The policy in Figure 3 states that image is owned by user with User ID 12. It also states that the image can be uploaded, viewed and downloaded by All users who stay in Rome Italy and belong to the group Fashionistas. Note that in this policy the optional constraints based on group membership, city and country are combined with the relationship based constraints with an "AND" operator, meaning that only users who satisfy both the relationship based constraints and the optional constraints can access the image.

```
<?xml version="1.0"?>
<Policy>
<Subject>12</Subject>
<Target>
<Action>The image can be uploaded by</Action>
<Object>All</Object>
<Action>The image can be viewed by</Action>
<Object>All</Object>
<Action>The image can be downloaded
by</Action>
<Object>All</Object>
<Action>Combining Operator</Action>
<Object>AND</Object>
<OptionalControl>
The city for which access is to be allowed
</OptionalControl>
<OptionalControlValue>rome</OptionalControlValue>
<OptionalControl>
The country for which the access is to be
allowed
</OptionalControl>
<OptionalControlValue>Canada</OptionalControlValue>
<OptionalControl>
General group based setting</OptionalControl>
<OptionalControlValue>Fashionistas</OptionalControlValue>
<OptionalControl>Group Based Setting for
View</OptionalControl>
<OptionalControlValue>
Fashionistas</OptionalControlValue>
<OptionalControl>Group Based Setting for
Upload</OptionalControl>
<OptionalControlValue>
Fashionistas</OptionalControlValue>
<OptionalControl>
Group Based Setting for
Download</OptionalControl>
<OptionalControlValue>
Fashionistas</OptionalControlValue>
</Target>
</Policy>
```

Fig. 3. Example of XML Based Policy