Primates: A Privacy Management System for Social Networks

Imen Ben Dhia Talel Abdessalem Mauro Sozio
Institut Mines-Télécom
Télécom ParisTech; CNRS LTCI
Paris, France
firstname.lastname@telecom-paristech.fr

ABSTRACT

While online social networks (OSN) present unprecedented opportunities for sharing information and multimedia content among users, they raise major privacy issues as users could often access personal or confidential data of other users. Most social networks provide some basic access control policies, which however seem to be very limited given the diversity of user relationships in the current social networks (e.g. friend, acquaintance, son) as well as the needs of social network users who might want to express sophisticated access control policies (e.g. "*invite all children of my colleagues to my child's birthday party*").

In this demonstration proposal, we present `Primates` a privacy management system for social networks. `Primates` allows users to specify access control rules for their resources and enforces access control over all shared resources. The set of users who are allowed to access a given resource is defined by a set of constraints on the paths connecting the owner of a resource to its requester in the social graph. We demonstrate the accuracy of our access control model and the scalability of our system.

1. INTRODUCTION

In the last few years, we have been witnessing an explosion of online social networks (OSNs) such as Facebook, LinkedIn, and Twitter which nowadays account hundred million users across the globe using them on a daily basis. At the time this demonstration proposal is written, Facebook reports over 901 million active users [2], while Twitter has 140 million active users [1].

In an OSN, each user can easily share information and multimedia content (e.g., personal data, photos, videos, contacts, etc.) with other users in the network, as well as organize different kind of events (e.g., business, entertainment, religion, dating etc.). While this presents unprecedented opportunities, it also gives raise to major privacy issues, as users could often access personal or confidential data of other users. Most social networks provide some basic access control policies, e.g., a user can specify whether a piece of information shall be publicly available, private (no one can see it) or accessible to friends only. However, the set of relationships represented in an OSN nowadays is quite rich and diverse, with relative relationships as well as the possibility of distinguishing between acquaintances and close friends becoming increasingly common; as a result there is an increasing need to providing more sophisticated access control policies. For instance, one would like to say "*invite all children of my colleagues to our child's birthday party*" or "*show this picture of myself wearing a funny costume to my friends and the friends of my friends while not to my colleagues*".

In this demonstration proposal, we present `Primates` a privacy management system for social networks implementing the access control model we proposed in [3]. Our model can be described as follows. We represent a social network as a labeled directed graph where nodes represent users and edges represent social relationships between them. Labels on the edges specify the type of relationships between users, such as 'friends', 'acquaintances', 'son', etc. Each edge can also be associated with a weight measuring the degree of trust between two given users.

Privacy preferences are expressed in our system by a set of access rules, each one being associated with a given resource (i.e, a shared resource) and specifies, through a reachability constraint, the set of users who can access such a resource. Each reachability constraint is represented by a path expression over the social graph, that combines constraints on labels, edge directions, label order, and/or distance between nodes. For instance, one would like to express the constraint that *Alice* can access the content published by *Bill* only if *Bill* is a friend of a friend of *Alice*, while he is not a colleague of *Alice*. In the social network graph, this corresponds to verify whether there is a path of length at most two between *Alice* and *Bill* whose edges are labeled 'friend' and there is no edge labeled 'colleagues' between the two users. We also allow resource owners to grant access to other users who are trusted "enough". Trust between two users is measured by the weight associated to the edge connecting them, if any, or it can be computed using trust propagation models [8, 7] if the there is no such an edge. We do not discuss trust propagation models any further, as this is out of the scope of this demonstration proposal.

Determining whether a requester should be granted access is done at query time, that is when a requester tries to access a shared resource. Our objective in this demonstration proposal, is to demonstrate the accuracy of `Primates` as well as its scalability.

The remainder of the demonstration proposal is organized as follows. In the rest of this section, we discuss some related work. In Section 2, we present the access control model used in `Primates`. In Section 3, we describe the way we enforced this access control model. We, then, describe the high-level architecture of our system in Section 4, and, detail our demonstration scenario.

Related Work. Previous work on access control in social networks can be classified into two main categories: (i) *machine learning-based* approaches as in [6, 9], which try to automatically configure user privacy settings, based on available explicit access authoriza-

tions and the underlying graph structure (i.e., communities), and, (ii) *rule-based* approaches as in [5], which introduced trust and distance in the social graph as the key criteria for access rules. Our work can be classified as a rule-based approach. It generalizes access constraints by taking into account the properties of the users, the paths connecting them, and allows expressing complex relationships (i.e, sequence of direct relationships of different types).

2. ACCESS CONTROL MODEL

In our model, an OSN is represented as a labeled directed graph, where nodes represent users and edges denote social relationships between users. User properties (age, gender, etc.) are expressed as attributes of the graph nodes.

Privacy preferences are expressed by a set of access rules, each one being associated with a given resource (i.e, a shared resource) and specifies, through a reachability constraint, the set of users who can access such a resource. Each reachability constraint is represented as a path expression over the OSN graph.

Online Social Network (OSN). We formally represent an OSN as a directed graph $G = (V, E)$, where V is a set of nodes denoting social network users and E is a set of directed edges representing social relationships between them. A labeling function $l : E \rightarrow 2^{\Sigma}$ specifies the set of labels (where Σ is a set of labels such as friend, colleague, etc.) associated to each edge, while a function $t : E \rightarrow [0, 1]$ measures trust between users. In case a trust value is not specified, a default value (e.g, 0.5) can be used. Each node is associated with a set of pairs (attr,value) specifying a set of attributes and their values for the corresponding user.

Access Rules. An access rule expresses a set of constraints that should be met in order to access a given resource. Formally, an access rule is defined as a tuple (u, r, P, C) where u denotes the owner of a resource r, C is a set of constraints on the attributes of the requester (such as location='Paris' or trust ≥ 0.8) and $P = p_1, \ldots, p_k$ expresses a set of constraints on the paths connecting the requester to the owner of the resource; each p_j is defined as a triple $p_j = (l, dir, I)$ where l is a label in Σ, $I = (min, max)$ is a pair of integers specifying the minimum and maximum length of p_j and $dir \in \{\leftarrow, \rightarrow, \Leftrightarrow\}$ indicates the direction of p_j. Given a requester v for resource r, an access rule (u, r, P, C) is satisfied if all constraints C are satisfied and there is a path between u and v satisfying P; v is granted access to r if there is an access rule at least that is satisfied.

For instance $P = (\prime friend\prime, \rightarrow, (1, 2)), (\prime colleague\prime, \rightarrow, (1, 1))$ expresses the constraint that there must be a path between the owner and a requester that first traverses a path of length in $(1, 2)$ whose edges are labeled 'friend' and then an edge labeled 'colleague'.

Suppose that *Elena* wants to make her baby-sitting advertisement (identified as *ad*) accessible to her direct friends. She can specify an access rule as follows:

$$AR_1 = (Elena, ad, (\prime friend\prime, \rightarrow, (1, 1)), -)$$

If *Elena* wants to extend access to her indirect friends (i.e, the friends of her friends) the path that should be specified would be $(\prime friend\prime, \rightarrow, (1, 2))$. Again, if she wants to extend it to the users that consider her as a friend, she should specify an other access rule which is the following:

$$AR_2 = (Elena, ad, (\prime friend\prime, \leftarrow, (1, 2)), -)$$

If *Elena* wants to change the access rules associated to her baby-sitting advertisement and make it accessible to the trustworthy (trust threshold = 0.8) baby-sitters of her friends within 2 hops then she should specify the following access rule:

$$AR_3 = (Elena, ad, \{(\prime friend\prime, \rightarrow, (1, 2)), (\prime babysitter\prime, \rightarrow, (1, 1))\}, [trust = 0.8])$$

Finally, the authorization can be limited to the baby-sitters living in Paris by adding an additional condition to the access rule as follows:

$$AR_4 = (Elena, ad, \{(\prime friend\prime, \rightarrow, (1, 2)), (\prime babysitter\prime, \rightarrow, (1, 1))\}, [location = Paris])$$

3. ACCESS CONTROL ENFORCEMENT

The access control enforcement mechanism is performed by the *reference monitor*, which is a trusted software module that intercepts each access request submitted by a requester to access a resource, and, based on the specified access policy, determines whether access should be granted or denied to the requester. Suppose that a user *v* is requesting for a resource *r*. When *v* submits his/her access request to access the resource *r*, the system retrieves the set of access rules associated to that resource. If there are no access rules related to the requested resource, then, the system will apply the *default access rule* which is defined by the user and applied whenever there are no access rules associated to the requested resource. This prevents the access control strategy from being too loose (by setting resources having no associated rules to public) or too restrictive (by setting resources having no associated rules to private). Then, the system evaluates the set of retrieved access rules and stops, either when the requester satisfies one of these rules, or when all the rules were evaluated and the requester satisfies no rule. In the former case, the requester is authorized to access the resource that he asked for. In the latter case, the requester is denied access because his profile is not consistent with the target audience.

Reachability paths evaluation. The enforcement of an access rule consists in evaluating the path P that is associated to it. The problem of evaluating these paths boils into a reachability problem in graph databases, which is well-known in the database community. Evaluating a reachability query, in our case, consists in determining whether two nodes u and v (for instance, the owner and the requester) in the graph are connected through a path with constraints on labels and distance. We devised an algorithm which is an adapted version of the breadth-first-search (BFS) algorithm applied to the graph together with the specified constraints to reduce the search space. The trust computation process between u and v is done at the same time, when the graph is explored. However, since the focus in `Primates` is on reachability, we implemented a simple trust propagation function and consider transitivity as the only way of propagation. The inferred trust value between two nodes u and v, connected through a path p, is computed by multiplying the explicit trust values associated with each edge in p.

4. SYSTEM DESCRIPTION

4.1 Global architecture

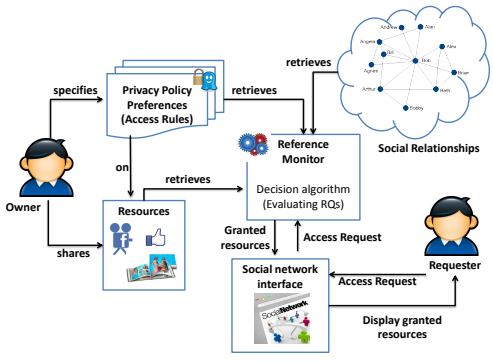

Figure 1: System Architecture

The global architecture of our system is depicted in Figure 1. Each user can share multiple resources. Resources are information (photos, videos, comments, etc.) to which access may need to be

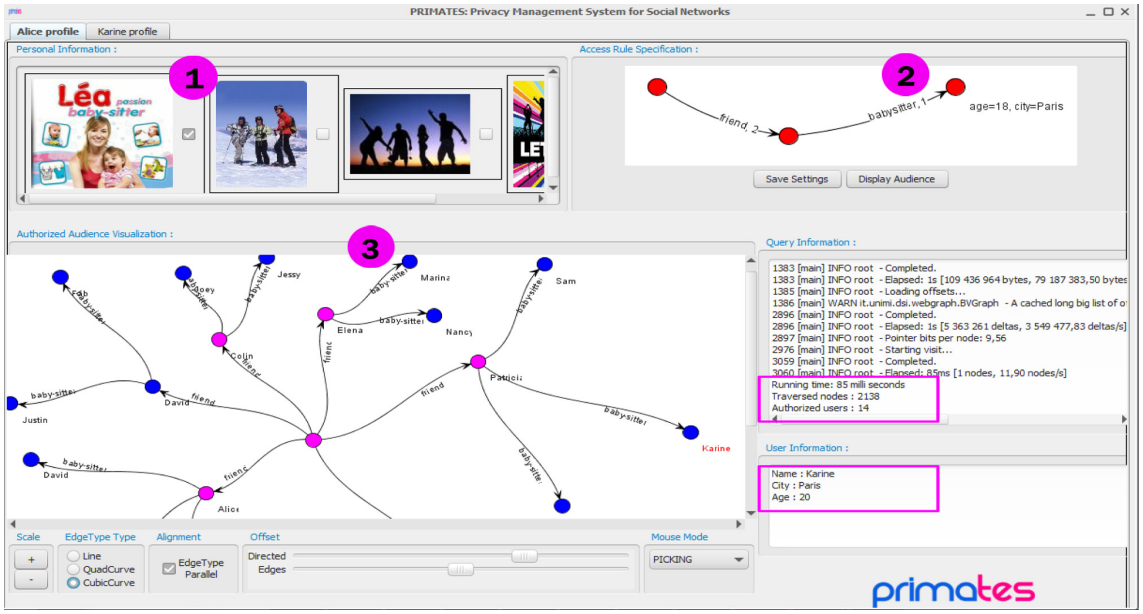

Figure 2: User interface for specifying access rules

controlled. A user (the owner) can express his privacy policy preferences for his resources by specifying one or several access rules to them. The requester, also called subject, is a user who is trying to access some resources of another user (the owner). The subject can send a request to access resources via the social network interface. This request is sent to the reference monitor, which is the component that implements user privacy preferences. It takes as input the access request and based on the access rules that are associated to the requested resource and social connections in the social netwok graph, it will authorize rendering only the resources for which the requester is part of its authorized profiles. We run our system on a compressed version of a snapshot of the *liveJournal* graph consisting of 5M nodes and 80M edges. The graph was compressed using the *WebGraph* framework [4] which provides algorithms for accessing compressed graphs without any decompressing. The compressed version of the *liveJournal* graph fits into main memory and can be accessed efficiently. User relationships have only a single type, which is *Friend*. For this reason, we artificially introduced other relationship types and associated them to edges on the *liveJournal* graph. The introduction of these types was done with respect to natural characteristics of human relations (e.g., a person can have on average 3 children and at most 2 close friends, etc.). We associated access rules to randomly selected user information. The response time is less than 1 sec (on average).

4.2 Demonstration Scenario

The demonstration shows two contributions of our work. The first one is the design of an access control model that allows users to specify more sophisticated privacy policies which fit their privacy needs. The second one is the enforcement of this model in such a way that allows users to intuitively specify their privacy settings and efficiently visualize the set of users that are allowed to get access to their information.

As shown in Figure 2, using `Primates`, visitors can select some information and express the desired privacy settings for it either from a set of access rules predefined by the system or by expressing new access rules in a user-friendly graphical way. They can express constraints on edge label and distance, and, node properties. Provinding users with a graphical interface to specify access rules allows them to express their privacy settings in a simple and intuitive way rather than expressing it in terms of paths as they are defined in the model. A parser converts then user actions into path patterns that would be evaluated as reachability queries.

Once access rules are specified, visitors will have the possibility to browse and visualize the graph of authorized users according to the specified privacy preferences. They can then navigate this graph and see explicitly who are the authorized users and can eventually refine access rules based on that. Displayed users on the authorized audience graph can be clicked on to to display their profile information such as the name, age, city, etc.

Visitors can also see how each user in the social network sees only information to which he is authorized to get access. They can see the profile (all shared personal information) of a user as an information owner on one side and try to get access to that profile from the perspective of another user on the other side. They will clearly see that not all the information in the owner's profile is displayed, but only a subset of that information to which he/she is allowed to access. An example scenario run on `Primates` can be found in the following url:
`http://perso.telecom-paristech.fr/~ibendhia/demo.html`.

5. REFERENCES

[1] https://business.twitter.com/basics/what-is-twitter/.

[2] http://www.facebook.com/press/info.php?statistics.

[3] T. Abdessalem and I. B. Dhia. A reachability-based access control model for online social networks. In *Proc. of the 1st ACM SIGMOD Workshop on Databases and Social Networks (DBSocial)*, pages 31–36, 2011.

[4] P. Boldi, M. Rosa, M. Santini, and S. Vigna. Layered label propagation: A multiresolution coordinate-free ordering for compressing social networks. In *Proceedings of the 20th international conference onWorld Wide Web*, 2011.

[5] B. Carminati, E. Ferrari, and A. Perego. Enforcing access control in web-based social networks. *ACM Trans. Inf. Syst. Secur.*, pages 6:1–6:38, 2009.

[6] L. Fang and K. LeFevre. Privacy wizards for social networking sites. WWW, pages 351–360, 2010.

[7] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. WWW, pages 403–412, 2004.

[8] H. Liu, E.-P. Lim, H. W. Lauw, M.-T. Le, A. Sun, J. Srivastava, and Y. A. Kim. Predicting trusts among users of online communities: an epinions case study. In *EC*, pages 310–319, 2008.

[9] M. Shehab, G. Cheek, H. Touati, A. C. Squicciarini, and P.-C. Cheng. Learning based access control in online social networks. WWW, pages 1179–1180, 2010.