Inferring Unknown Privacy Control Policies in a Social Networking System

Amirreza Masoumzadeh
Department of Informatics
University at Albany–SUNY
Albany, NY, USA
amasoumzadeh@albany.edu

ABSTRACT

Social networking systems (SNSs) such as Facebook allow users to control accesses to certain information belonging to them via a set of privacy settings. However, due to various potential system design considerations and usability restrictions such settings are never complete, i.e., not all the applicable policies to information related to a user are configurable. In fact, access to user information is governed by the collection of the privacy settings and a set of fixed policies specified by the SNS. We observe that an SNS such as Facebook is less than transparent about such fixed policies; although some might be communicated to users via help pages and nudges (e.g., profile picture is public on Facebook), they tend to be incomplete and inaccurate.

In this paper, we propose an approach to infer the enforced privacy control policy by an SNS and consequently the unknown policies to the user given the explicit privacy settings and other policies communicated to the users by the SNS. Such an approach helps end users understand better the implicit policies imposed by the system and can be leveraged by an SNS operator to improve the transparency of their system.

Categories and Subject Descriptors

K.4.1 [**COMPUTERS AND SOCIETY**]: Public Policy Issues—*Privacy*

Keywords

Social Networking Systems, Privacy Control Policies, Unknown Policies, Policy Inference

1. INTRODUCTION

Privacy control policies that govern user-to-user interactions in an SNS can be composed of many different policies and therefore be very complex to understand. This complexity affects both SNS operators who need to ensure about the consistency and integrity of the enforced policies, and end users who deserve to know about their privacy status in the system. We argue that it is essential to give a clear picture of enforced policies to both groups. However, our observations about transparency of privacy control policies in major SNSs such as Facebook is not promising.

We categorize privacy control policies in an SNS into those that are *configurable* by users (e.g., through privacy setting controls) and those that are *fixed* and specified by the SNS. For example, on Facebook, a user can configure who can see a post of her (configurable policy) while her profile picture is publicly accessible (fixed policy). Orthogonally, we can categorize policies into *known* and *unknown* policies. A policy is considered known when it is stated clearly to the user. Examples include the policy that the user specifies by indicating the audience for her post, or publicly accessibility of profile pictures which is explained in Facebook's help pages. In contrast, an unknown policy is not clearly communicated to the user, although it might be implied.

We observe that in an SNS such as Facebook, that has a complex policy model, there are many instances of unknown privacy control policies as well as incorrectly-expressed, known policies. This can severely affect users' ability to understand and control their privacy in a system that privacy can be a critical factor given the amount of personal material shared. In our previous work [7], we proposed an ontology-based approach to reason about the existence of user resources that are governed by non-configurable/unknown policies in an SNS. In this work, we aim not only to detect unknown policies but also to infer them. Furthermore, the goal is to detect such unknown policies at the more fine-granular level of policies compared to course-granular level of resources in our previous work.

We approach this problem by observing runtime behavior of the system, testing it using a comprehensive set of scenarios and inferring the enforced policy of the system using classification techniques. In order to achieve this, we propose to develop an automated tool that collects access logs by simulating a set of users on Facebook that are involved in a number of photo sharing scenarios. By iterating through and configuring all different policy combinations for users and observing access decisions, our tool collects a comprehensive access log. Once converted to policy rules such access policy is representative of the enforced privacy control policy. By constructing a representative policy model of the known (configurable and fixed) policies, we analyze the differences between the enforced and known policies to infer about the unknown policies enforced by the system.

WPES'15, October 12, 2015, Denver, Colorado, USA.
© 2015 ACM. ISBN 978-1-4503-3820-2/15/10 ...$15.00.
DOI: http://dx.doi.org/10.1145/2808138.2808151.

The rest of the paper is organized as follows. In Section 2, we explain our approach to infer enforced policies by an SNS, including the policy model, access scenarios, and data collection. We then discuss the results of employing our approach in inferring about applicable policies to photo sharing scenarios on Facebook in Section 3. We briefly review the related work in Section 4, followed by conclusions in Section 5.

2. INFERRING ENFORCED POLICIES

In this paper, we focus on Facebook as an SNS with fairly complex privacy control policies. Also, we consider a limited scope of policies as our first experiment with this methodology. We study the privacy control policies involved in access to photos since photo sharing is one of the main use cases of Facebook and involvement of owner and tagged users makes it a rather complex policy situation. Our goal is to infer unknown or incorrectly expressed, known policies by observing the enforced policy by Facebook.

In a photo access scenario, three users are involved: access subject S, photo owner O, and tagged user T. We aim to build all potential access scenarios based on the involved users and their applicable privacy control policies in order to observe and infer the implemented policy. We make a number of assumptions to make this task tractable which will be discussed as we describe the policies and scenarios.

2.1 Applicable Policies and Access Decision Information

There are two known, configurable policies involved in a photo access scenario in Facebook. The policy by the photo owner O can be one of the following options:

- Public
- FoF [+FoT] [-X]
- F [+FoT] [-X]
- I [+FoT] [-X]
- Myself

In the above, the arguments in brackets are optional. FoF indicates friends of friends, F indicates friends, I is a subset of friends to include, FoT indicates the friend of the tagged persons will be included, and X is a subset of friends to be excluded from accessing the resource. Following the same notations, the policy by a tagged user T can be specified using one of the options below:

- F [-X]
- I [-X]
- Myself

There are also two known, fixed policies involved. The audience selection screen that is shown to the owner indicates that tagged users will have access to the photo regardless of the settings. Also, Facebook states that anyone who can see a post can like or comment on it in its help pages. An example of unknown, fixed policies in this scenario is who can tag a photo. Although Facebook has a tag review mechanism in place against misuse of tag feature, there is no clear description about who can add tags to a photo when visiting it.

We make an assumption that access decision is made based on only the information captured in the above policies. Therefore access decision information includes the policy specified by owner and tagged user ("FoF", "F", etc.), the friendship distance of subject to owner and tagged user, subject being part of the inclusion/exclusion sets, etc. Note that policies might be unknown, but are assumed to be based on the same information. For example, a photo access will not be denied because of subject being friends with too many other users.

2.2 Access Scenarios

In order to infer the enforced policy by Facebook, we create a comprehensive set of access scenarios to observe and record how Facebook responds to various access requests. Such a comprehensive observation is only viable through an automated process. As we aim to build and run an automated tool to interact with and collect data from Facebook, we should minimize the size and dynamic changes made by the tool due to the inherent cost of running such an automated tool. Our solution is to create a fixed network of Facebook users that dynamically change their policies in order to create different access scenarios.

Our proposed network of users is depicted in Figure 1. There are total of 7 users in two separate connected components. In the first connected component, we have owner O, owner's friend OF, owner's friend of friend $OF2$, and owner's friend of friend of friend $OF3$. In the second connected component, we have user N, her friend NF, and her friend of friend $NF2$. O owns one untagged photo and five other photos in which O, OF, $OF2$, $OF3$, and N are individually tagged. Each user can play the role of access subject S who can access each of the photos, upon which the user tagged in the photo is considered as the tagged person (T) for that specific access scenario.

Based on the policies presented in Section 2.1, the major access decision information in a photo access scenario is the friendship distance of the access subject (S) to the owner (O) and to the tagged user (T). We define friendship distance between users A and B as their distance in the friendship network, denoted as $dist(A, B)$. $dist(A, B) = \infty$ indicates A and B are unconnected and $dist(A, B) = 0$ indicates A and B are the same user. Our layout can support access scenarios with $dist(S, O) \in \{0, 1, 2, 3, \infty\}$ and $dist(S, T) \in \{0, 1, 2, \infty\}$. We assume that with regards to the owner's policy a subject at distance $dist(S, O) = 3$ is representative of any subject at $dist(S, O) \geq 3$ in the same connected component. Also, a subject at $dist(S, O) = \infty$ is representative of any unconnected subject. Similarly, we assume that with regards to a tagged user's policy a subject at distance $dist(S, T) = 2$ is representative of any subject at $dist(S, T) \geq 2$ in the same connected component.

We create access scenarios by considering each user as a subject, accessing every photo in all possible combinations of owner policies and tagged user policies. Note that variation of policies include excluding friends for "F" and "FoF" policies. In such cases, without loss of generality, we consider excluding one friend at a time. In each policy configuration, four access rights on a photo are tested: "read", "like", "comment", and "tag".

2.3 Implementation and Data Collection

We manually created the initial layout (Figure 1) including users, photos, and tags. We then developed an auto-

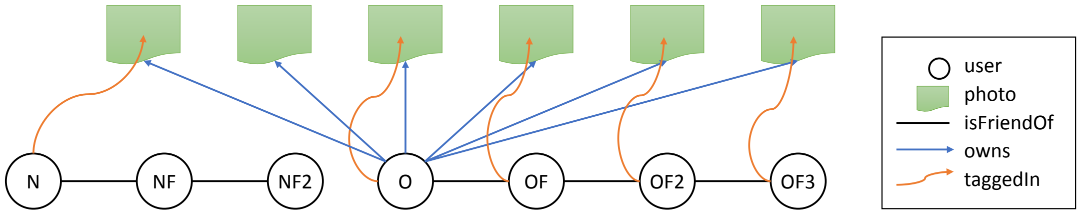

Figure 1: Layout of Facebook Entities for Access Scenarios

Attribute	Description
action	{"read", "like", "comment", and "tag"}
S_O_dist	$dist(S, O)$
T_exists	"1" if someone is tagged in the photo, else "0"
S_T_dist	$dist(S, T)$
owner_policy	{"me", "f", "fof", and "public"}
Opx_exists	"1" if O has excluded a friend in her policy, else "0"
S_is_Opx	"1" if S is the excluded friend in O's policy, else "0"
taggedu_policy	{"me", "f"}
TUpx_exists	"1" if T has excluded a friend in her policy, else "0"
S_is_TUpx	"1" if S is the excluded friend in T's policy, else "0"
decision	grant ("TRUE") or deny ("FALSE")

Table 1: Access Log Dataset Fields

mated tool using Selenium package for Python that simulates users' interactions with Facebook in order to create different configurations discussed in Section 1 and record the observed access decisions made by the system. The tool iteratively sets up each policy configuration in the network, performs all possible accesses, and logs the observed decisions. Developing such an automated tool that interacts with a live system proved to be a fairly complicated task. We collected 3024 access records in total. Each access record includes attributes such as subject, tagged user, owner's policy, tagged user's policy, and decision.

2.4 Data Analysis

We employed the popular data mining software Weka [4] in order to build classification models based on the observed access decisions. Several classifiers were trained on our dataset. In this paper, we report classification results based on BFTree classifier (Best-First decision tree) as it correctly classified all records and produced relatively small size trees. Investigating an optimal choice of classifier in terms of being easily interpretable by users will be of our future work.

In order to retain only useful information in our dataset and avoid confusion for our classification tasks, we use a converted version of our dataset: we convert identity attributes to distance measures or boolean values. For instance, instead of keeping the identities of subject S and target T in each record we include $dist(S, O)$ and $dist(S, T)$, respectively. Table 1 lists the attributes in our converted dataset. All but the last attribute listed are considered for building a classification model.

3. EXPERIMENT RESULTS

Our first classification task aims to infer the enforced policy by Facebook, using attribute decision in Table 1 as the class label. We report in Figure 2 the inferred decision tree classifier based on the observed access decisions. According to this notation, an access control rule can be constructed by the conjunction of the conditions leading to a leaf node. A leaf node itself is indicated by a class label (here, TRUE/FALSE) with the number of instances that are correctly/incorrectly classified mentioned in the brackets. For instance, according to the results in Figure 2, if $dist(S, O) \leq 2 \wedge dist(S, O) \neq 0 \wedge dist(S, T) = 0$ then access is granted. Note that minor aesthetic changes has been applied to the raw output produced by Weka to make it more easily readable. The result perfectly represents the enforced policy that Facebook implements for a photo without any classification error.

Our second and main classification task aims to infer unknown policies and incorrectly expressed, known policies in a photo access scenario. For this purpose, we develop an access control decision function that simulates Facebook access decision making according to the known policies to us. As we discussed in Section 2.1, those include configurable policies for "read" action by owner O and tagged user T through privacy settings, fixed policies for "like"/"comment" actions, and fixed policy for accesses made by T. We also need to have a conflict resolution strategy for potential conflicts between O and T policies (i.e., O includes someone in the audience while T excludes the same person, or vice versa). But such a conflict resolution policy has not been clearly stated by Facebook. In order to avoid further complexity in our tool, we select a conflict resolution strategy (treat this policy as known) by testing different choices and comparing against the enforced policy. Moreover, for the "tag" action, we assume the same policy as for "like"/"comment" actions since it has not been explicitly expressed by Facebook.

We run our simulated access decision function according to our access log dataset attributes and compare the result with the actual decision by Facebook. As we expected, the enforced policy is not completely consistent with our simulated policy. In order to contrast the differences, we create a new class label in our dataset (Table 1) that indicates agreement ("same") or disagreement ("different") of known policy access decision with the enforced policy. Then, we run a classifier based on the newly-created attribute in order to identify the characteristics of accesses that are different between our known/assumed policy model and the enforced policy model. Figure 3 reports the resulting classification tree. For brevity, we have summarized the tree to include only the leaves that indicate disagreement. According to the results, for example, if the owner policy is "public", S is O's friend of friend, the distance between S and T is more than

```
S_O_dist <= 2
|  S_O_dist = 0: TRUE(432.0/0.0)
|  S_O_dist != 0
|  |  S_TU_dist = 0: TRUE(192.0/0.0)
|  |  S_TU_dist != 0
|  |  |  owner_policy=(me)|(f)
|  |  |  |  owner_policy=(me): FALSE(112.0/0.0)
|  |  |  |  owner_policy!=(me)
|  |  |  |  |  Opx_exists = 0
|  |  |  |  |  |  S_O_dist <= 1: TRUE(56.0/0.0)
|  |  |  |  |  |  S_O_dist >= 2
|  |  |  |  |  |  |  S_TU_dist <= 1
|  |  |  |  |  |  |  |  taggedu_policy=(me): FALSE(8.0/0.0)
|  |  |  |  |  |  |  |  taggedu_policy!=(me)
|  |  |  |  |  |  |  |  |  S_is_TUpx = 0: TRUE(12.0/0.0)
|  |  |  |  |  |  |  |  |  S_is_TUpx = 1: FALSE(8.0/0.0)
|  |  |  |  |  |  |  S_TU_dist >= 2: FALSE(28.0/0.0)
|  |  |  |  |  Opx_exists = 1
|  |  |  |  |  |  S_O_dist <= 1: FALSE(56.0/0.0)
|  |  |  |  |  |  S_O_dist >= 2
|  |  |  |  |  |  |  S_TU_dist <= 1
|  |  |  |  |  |  |  |  taggedu_policy=(me): FALSE(8.0/0.0)
|  |  |  |  |  |  |  |  taggedu_policy!=(me)
|  |  |  |  |  |  |  |  |  S_is_TUpx = 0: TRUE(12.0/0.0)
|  |  |  |  |  |  |  |  |  S_is_TUpx = 1: FALSE(8.0/0.0)
|  |  |  |  |  |  |  S_TU_dist >= 2: FALSE(28.0/0.0)
|  |  |  owner_policy!=(me)|(f)
|  |  |  |  S_is_Opx = 0
|  |  |  |  |  action=(like)|(comment)
|  |  |  |  |  |  S_TU_dist <= 1: TRUE(70.0/0.0)
|  |  |  |  |  |  S_TU_dist >= 2
|  |  |  |  |  |  |  S_O_dist <= 1: TRUE(28.0/0.0)
|  |  |  |  |  |  |  S_O_dist >= 2: FALSE(42.0/0.0)
|  |  |  |  |  action!=(like)|(comment): TRUE(140.0/0.0)
|  |  |  |  S_is_Opx = 1: FALSE(56.0/0.0)
S_O_dist > 2
|  S_TU_dist < 2
|  |  action=(tag): FALSE(78.0/0.0)
|  |  action!=(tag)
|  |  |  S_TU_dist = 0: TRUE(108.0/0.0)
|  |  |  S_TU_dist >= 1
|  |  |  |  owner_policy=(me)|(fof)|(f)
|  |  |  |  |  taggedu_policy=(me): FALSE(30.0/0.0)
|  |  |  |  |  taggedu_policy!=(me)
|  |  |  |  |  |  S_is_TUpx = 0
|  |  |  |  |  |  |  owner_policy=(me): FALSE(9.0/0.0)
|  |  |  |  |  |  |  owner_policy!=(me): TRUE(36.0/0.0)
|  |  |  |  |  |  S_is_TUpx = 1: FALSE(30.0/0.0)
|  |  |  |  owner_policy=(public): TRUE(21.0/0.0)
|  S_TU_dist >= 2
|  |  owner_policy=(fof)|(f)|(me): FALSE(1180.0/0.0)
|  |  owner_policy=(public)
|  |  |  action=(like)|(comment)|(tag): FALSE(177.0/0.0)
|  |  |  action=(read): TRUE(59.0/0.0)
```

Figure 2: Inferred Decision Tree for Enforced Policy

```
owner_policy=(public)
|  S_O_dist = 2
|  |  S_TU_dist >= 2
|  |  |  action=(like)|(comment): different(14.0/0.0)
|  S_O_dist >= 3
|  |  action=(tag)|(like)|(comment)
|  |  |  S_TU_dist < 2
|  |  |  |  action=(tag): different(13.0/0.0)
|  |  |  S_TU_dist >= 2: different(177.0/0.0)
owner_policy!=(public)
|  S_TU_dist = 0
|  |  action=(tag)
|  |  |  S_O_dist >= 3: different(30.0/0.0)
|  S_TU_dist >= 1
|  |  owner_policy=(fof)
|  |  |  S_O_dist = 2
|  |  |  |  action=(like)|(comment)
|  |  |  |  |  S_TU_dist >= 2: different(28.0/0.0)
|  |  |  S_O_dist >= 3
|  |  |  |  S_TU_dist < 2
|  |  |  |  |  action=(tag)
|  |  |  |  |  |  taggedu_policy=(f)
|  |  |  |  |  |  |  S_is_TUpx = 0: different(6.0/0.0)
|  |  owner_policy!=(fof)
|  |  |  S_TU_dist < 2
|  |  |  |  action=(tag)
|  |  |  |  |  S_O_dist >= 3
|  |  |  |  |  |  owner_policy=(f)|(public)|(fof)
|  |  |  |  |  |  |  taggedu_policy=(f)
|  |  |  |  |  |  |  |  S_is_TUpx = 0: different(6.0/0.0)
```

Figure 3: Inferred Differences between Known and Enforced Policies (Summarized)

one, and action is either "like" or "comment" the known and enforced policy decisions are different.

We make the following observations based on the results reported in Figure 3. First, our simulated model fully captures the enforced policy model in case of "read" accesses since such accesses are not represented in the reported section of the decision tree (disagreements). Second, "like" and "comment" do not follow the "read" access policy as expressed in Facebook help pages. This proves to be an instance of known-but-incorrect, fixed policy. Third, the same policy is applicable to both "like" and "comment" actions although not according to the expressed policy by Facebook. Fourth, the applicable policy to the "tag" action is not comparable to (always more restrictive/open than) the policy for "like"/"comment" actions. This calls for a deeper investigation to understand the underlying rationale for such unknown policies.

4. RELATED WORK

Machine learning approaches have been previously employed to infer (properties about) access control policies. Martin and Xie [6] propose to find potential misconfiguration in XACML access control policies. Based on access decisions for a set of generated access requests, they infer policy properties (rules) using a rule classification algorithm. The rational is that any mismatch between access decision based on inferred rules and original policy could be a potential buggy instance of misconfiguration in the original policy. They present successful result of finding a previously-known bug in a very small policy scenario (3 subjects, 2 resources, and 2 actions). Bauer et al. [1] propose to detect misconfiguration in access control policies and suggest corrections based on previous access patterns. By mining associative rules on a dataset of previous access logs they detect potential misconfiguration instances for authorizations of individual users. For instance, if a pattern is detected that (usually but not always) people who have accessed resources X and Y (premises of the rule) have also accessed resource Z (conclusion of the rule) there might be a misconfiguration if the premises hold for a certain user but the conclusion does not. They also propose a feedback approach to avoid recursive false positives of such a detection approach. Similar to the work by Martin and Xie [6] we use rule classification to infer policies. However, our problem is different in nature as we employ it to detect unknown policies (and not to detect errors). Furthermore, the policy model of our application domain is far more complex than the simple policy they studied. Also, machine learning has been employed in the context of SNSs for intelligent generation of privacy control policies based on factors such as network structure, profile information, and user feedback [3, 2].

Our work is also related to conformance checking of access control policies [5, 8]. The goal of conformance checking is to test the correctness of enforcing a specific policy. Our inferred model of the enforced policy can be used for such testing purpose. However, we approach a different problem in this paper. We do not have access to a complete specification of the enforced policy by the system, and our goal is to detect missing or incorrectly captured pieces of the policy according to the known components of it.

5. CONCLUSIONS

We proposed an approach in this paper to infer enforced privacy control policies and consequently its unknown pieces in a complex policy system governed by SNSs such as Facebook. We have developed an automated tool that generates scenario at runtime and observes enforced policy by simulating a network of SNS users. We demonstrated how such an observed policy can be modeled and contrasted against known policies. For example, we show that the policy about like-ing and commenting on photos has been incorrectly expressed by Facebook. There are several instances where a user might not have "like" and "comment" accesses to a photo while she can access the photo itself, in contradiction to what has been suggested in Facebook help pages. We believe that both end users and developers of today's increasingly complex systems will benefit from such automated testing and inference mechanisms.

Although we were able to detect and characterize the unknown and incorrectly expressed policies in the studied scenarios, building a more easily human-interpretable presentation of such policies will be one of our future tasks. Our approach was also limited in handling unknown meta-policies such as conflict resolution between owner and tagged user decisions. We tackled this problem by manually assuming a policy and verifying it against the enforced policy. We plan to develop a more systematic approach to check for such unknown meta-policies as our future work.

6. REFERENCES

[1] L. Bauer, S. Garriss, and M. K. Reiter. Detecting and resolving policy misconfigurations in access-control systems. *ACM Trans. Inf. Syst. Secur.*, 14, 2011.

[2] G. Danezis. Inferring privacy policies for social networking services. In *Proceedings of the 2nd ACM workshop on Security and artificial intelligence - AISec '09*, page 5, New York, New York, USA, Nov. 2009. ACM Press.

[3] L. Fang and K. LeFevre. Privacy wizards for social networking sites. In *Proc. 19th Int'l Conference on World Wide Web*, WWW '10, pages 351–360, Raleigh, North Carolina, USA, 2010. ACM.

[4] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10, Nov. 2009.

[5] H. Hu and G. Ahn. Enabling verification and conformance testing for access control model. In *Proceedings of the 13th ACM symposium on Access control models and technologies - SACMAT '08*, page 195, New York, New York, USA, June 2008. ACM Press.

[6] E. Martin and T. Xie. Inferring Access-Control Policy Properties via Machine Learning. In *Seventh IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'06)*, pages 235–238. IEEE, 2006.

[7] A. Masoumzadeh and J. Joshi. Privacy Settings in Social Networking Systems: What You Cannot Control. In *Proc. 8th ACM Symposium on Information, Computer and Communications Security (ASIACCS 2013)*, pages 149–154. ACM Press, May 2013.

[8] D. Xu, L. Thomas, M. Kent, T. Mouelhi, and Y. Le Traon. A model-based approach to automated testing of access control policies. In *Proceedings of the 17th ACM symposium on Access Control Models and Technologies - SACMAT '12*, page 209, New York, New York, USA, June 2012. ACM Press.