Undoing of Privacy Policies on Facebook

Vishwas T. Patil(✉) and R.K. Shyamasundar

Department of Computer Science and Engineering,
Information Security R&D Center, Indian Institute of Technology Bombay,
Mumbai 400076, India
ivishwas@gmail.com, shyamasundar@gmail.com

Abstract. Facebook has a very flexible privacy and security policy specification that is based on intensional and extensional categories of user relationships. The former is fixed by Facebook but controlled by users whereas the latter is facilitated by Facebook with limited control to users. Relations and flows among categories is through a well-defined set of protocols and is subjected to the topology of underlying social graph that continuously evolves by consuming user interactions. In this paper, we analyze how far the specified privacy policies of the users in Facebook preserve the standard interpretation of the policies. That is, we investigate whether Facebook users really preserve their privacy *as they understand it* or certain of their innocuous actions leak information contrary to their privacy settings. We demonstrate the kind of possible breaches and discuss how plausibly they could be set right without compromising performance. The breaches are validated through experiments on the Facebook.

1 Introduction

Social networks help individuals, groups of individuals, organizations, and governments to establish their digital identity online and allow other digital identities to interact with it. Establishment of connections and subsequent interactions allow the digital identities to engage with their audience. The platform also facilitates search for new audience. It keeps track of interactions among identities and categorizes them according to their individual likes and dislikes, which helps the platform in presenting relevant content and advertisement to its audience.

Protection of one's digital identity and associated information is desired and expected. Social network platforms deploy access control systems to ensure that security and privacy of their users is maintained. However, by definition, social networks are formed by dynamic connections among stakeholders that independently control their privacy specifications. It is challenging to ensure the privacy of a conservative individual who is connected with a liberal individual. In this paper, we shall study a set of such scenarios and analyze the impact of actions of independent stakeholders' on privacy.

The navigability over the Facebook has been succinctly captured by Boyd and Ellison [3] who characterize social network systems like Facebook by three

G. Livraga and S. Zhu (Eds.): DBSec 2017, LNCS 10359, pp. 239–255, 2017.
DOI: 10.1007/978-3-319-61176-1_13

functions: *(i) identity representation:* allow users to create a profile by populating a pre-defined template of fields with personal information, *(ii) distributed relationship articulation:* facilitate relationship with other identities and organize the relationships into categories like Friend, Family, etc., *(iii) traversal-driven access:* allow users to traverse the social space and grant access based on the access policy specified on the reachable node.

Abstraction of relationships into categories like Family, Friends, Friends of Friends, etc., help individuals to specify access control in a natural and understandable way. On the other hand, it makes access control enforcement very challenging when there are billions of nodes in the social graph and edges, which are added/deleted spontaneously as users interact. It is indeed an achievement by Facebook that they achieve the underlying access control in quite a performance-centric manner [4,7]. Thus, it is interesting and important to understand the access control mechanism of such a dynamic system so that one can attempt to analyze or reason about its properties and compliances with settings permissible in a user's profile.

The paper is organized as follows: Sect. 2 highlights user representation in Facebook and overall global model to enable analysis of access control and policy preservation mechanisms. In Sect. 3, we discuss how the categories, as defined and facilitated by Facebook, can be used to provide/restrict access to users. Section 4 illustrates the gaps between user interpretation of privacy-preservation and user actions that undo privacy. In Sect. 5, we discuss possible approaches to mitigate various privacy anomalies discovered. Section 6 presents current related work and other works on Facebook privacy evaluations that have become irrelevant as over the years Facebook has updated its architecture, features and mechanisms for access control.

2 Access Control in Facebook: User Representation, Social Graph

Facebook organizes its content retrieval and distribution around 3 pillars: *(i) Newsfeed:* responsible to present content/updates to users about their friends, *(ii) Timeline:* where users curate their own content, and *(iii) Graph search:* also known as *social graph* that consumes all of the actions of Facebook users and simultaneously allows them to query the graph. Queries to the graph are resolved in context of the requester and the content being requested. Access to content is governed by an access policy specified by its owner. To study access control in Facebook we briefly explain its social graph and some of the relevant query functions.

2.1 Social Graph of Facebook

Social graph in Facebook is a representation of user information on Facebook. Each and every action or event created by Facebook's users is consumed by the social graph. The graph evolves reflecting user actions. Facebook's social graph is composed of [8]:

- ***nodes*** - basically "things" such as a *user*, a *photo*, a *page*, a *comment*
- ***edges*** - the connections between those "things", such as relationships (*friends*)
- ***fields*** - info about those "things", such as a *user*'s *birthday*, or the *name* of a *page*

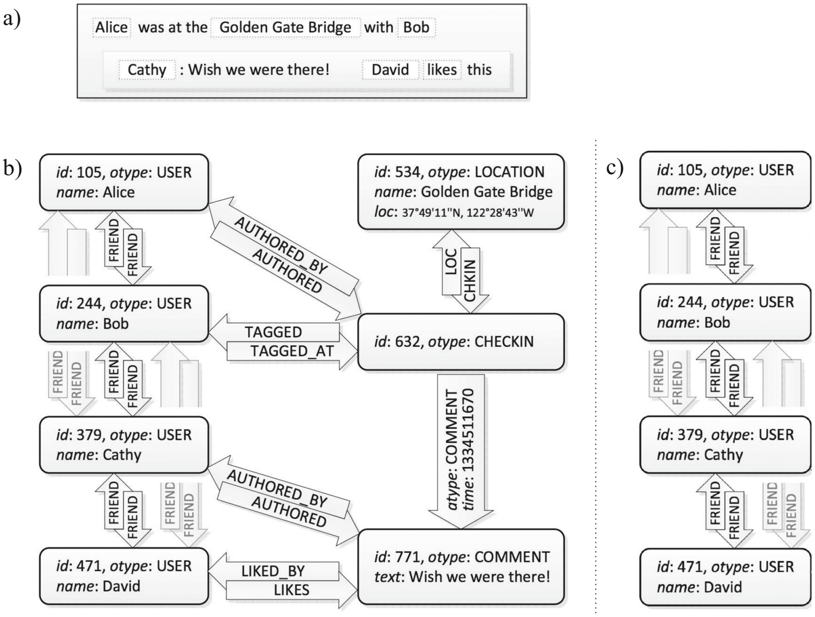
Each user of Facebook is represented by a node (identified uniquely by a 64-bit number) on Facebook's social graph and a user's relationship with other nodes is captured through labelled edges. For example, $\textcircled{A} \xleftrightarrow{\text{friends}} \textcircled{B}$ is a representation of *friendship* relationship between user A and B. When user Z *follows* user B on Facebook, that relationship is absorbed by the social graph and the graph evolves to: $\textcircled{A} \xleftrightarrow{\text{friends}} \textcircled{B} \xleftarrow{\text{follower}} \textcircled{Z}$.

Updates to social graph happen by adding/deleting nodes (or updating fields of nodes), and adding/deleting/updating labelled edges. Social graph allows its nodes to be queried. A user is allowed to compose a query by specifying a particular node (of type *root* [8]) about which the requester needs information. It is very likely that different sets of information about a node are presented based on who the requester is. For example, in the graph shown above, assume user B posts a photo with access set to his "Friends"; the access control mechanism of Facebook will allow user A to reach the post, whereas user Z will be denied access. If user B changes the access setting to "Public"; both user A and user Z will be allowed to access the post. In the following subsections we shall briefly introduce the reader to Facebook's social graph and its access control mechanism based on lists (information classification labels/categories).

2.2 Representation of User Events and Interpretation of Privacy Policies

Consider a typical user event on Facebook as depicted in Fig. 1(a). The event is created by Alice and its interaction with Bob, Cathy, and David is depicted in Fig. 1(b) through the social graph. Upon careful observation on nodes in Fig. 1(b), one can notice the node *id*s are in chronological order, signifying sequence of events and user actions. As per Facebook's current (05/2017) working, we can reason the following about Fig. 1(b):

- Alice has "checked-in" to a "location" and has *tagged* Bob in that event, therefore Bob has visibility (access) to the event. This is because Facebook's default policy allows a tagged user to access the event in which the user is tagged.
- Cathy could comment on Alice's event because Cathy has visibility of the event. This could happen in two circumstances: either Alice has set her access policy on this event to "Friends" or "Public".
- David could like the comment made by Cathy because David had visibility to the event. This implies Alice's access policy for this event must be "Public" since David and Alice are not friends.



Alice was at the Golden Gate Bridge with Bob

Cathy : Wish we were there! David likes this

Fig. 1. (a) Creation of event, (b) its representation on social graph [4], (c) deletion of event

Similarly, reasoning about Fig. 1(c), we can conclude that the change in social graph is not because of Alice setting her event's access policy to "Only Me" but because of Alice deleting the event altogether. Owner of an event can delete the event at will and all dependent nodes of that event node also get deleted. If Alice sets the access policy of her event to "Only Me", then the event node, its dependent nodes, and their associations will be part of the social graph but visibility of the event gets restricted to Bob alone. To understand the scenario in which Alice changes the access policy of the event to "Only Me" we need to understand the access control model of Facebook. Social graph provides traversal paths from a requester to the node being requested. Existence of path between a requester and the node being requested is not a sufficient condition to access the node but the requester has to also satisfy the access policy specified on that node by node's owner. To analyze Facebook's access control model, let us first understand the types of nodes in its social graph, and set of queries that can be run on them, followed by an understanding of categories of access/privacy policies provided by Facebook.

Querying the Social Graph

1. Let $\mathbb{G}(\mathbb{V}, \mathbb{E})$ denote the social graph of Facebook. Let $\mathbb{V}$ denote the set of vertices or nodes and $\mathbb{E}$ as the set of edges, where $\mathbb{E} \subseteq \mathbb{V} \times \mathbb{V}$.

2. Let $T_V : \mathbb{V} \longrightarrow \mathbb{S}_V$ provide type of a node (e.g., *user, photo, event, ...*)
   The members of this set can further be divided into two sets: *subjects* and *objects*. Users are subjects and the content generated by the users is treated as objects. The fields populated by a user in her profile are also treated as objects.
3. Let $T_E : \mathbb{E} \longrightarrow \mathbb{S}_E$ provide type of an edge (e.g., *friend, like, comment, ...*)
   The members of this set are associations between subjects and objects.
   Let $\mathcal{E} : \mathbb{S}_E \longrightarrow 2^{\mathbb{E}}$ be a generic function that returns the set of all edges of same type. Let $\mathcal{E}_t \; \forall \; t \in \mathbb{S}_E$ be a function on edge of type t.

(E_1) $\mathcal{E}_t(x) := \{y \mid (x, y) \in \mathcal{E}_t\}$
(E_2) $\mathcal{E}_{friends}(user) = \{subject \mid (user, subject) \in \mathcal{E}_{friends}\}$
(E_3) $\mathcal{E}_{authored}(user) = \{object \mid (user, object) \in \mathcal{E}_{authored}\}$
(E_4) $\mathcal{E}_{authored\_by}(object) = \{user \mid (object, user) \in \mathcal{E}_{authored\_by}\}$
(E_5) $\mathcal{E}_{likes}(user) = \{object \mid (user, object) \in \mathcal{E}_{likes}\}$
(E_6) $\mathcal{E}_{liked\_by}(object) = \{user \mid (object, user) \in \mathcal{E}_{liked\_by}\}$
(E_7) $\mathcal{E}_{tagged\_at}(user) = \{object \mid (user, object) \in \mathcal{E}_{tagged\_at}\}$
(E_8) $\mathcal{E}_{tagged}(object) = \{user \mid (object, user) \in \mathcal{E}_{tagged}\}$

While the above set of query functions is useful for a requester to traverse the social graph, it also raises questions as how Facebook preserves the access of users/resources in the system. When a requester traverses towards a node of its interest on the social graph, underlying topology of the graph determines existence of a path from the requester to the node in question. However, existence of traversal path is not a sufficient condition for access. Each non-*root* type of node in the social graph is access controlled by a policy specified by its owner. In the following, we shall discuss the various types of policies Facebook provides its users.

3 Policy Specification for Access over Users in Facebook

In this section, we shall discuss usage of relationship categories (i.e., labels) as policy handles and its efficacy enforcing user stated privacy policies.

3.1 Lists as Policy: Extensional vs Intensional Information Classification

Each user on Facebook is provided with pre-defined relationship categories, called lists, along which users can organize their relationships with others. “Friends” is the basic relationship category to which every user-to-user relationship (friendship) is added. A user is allowed to organize friendship relations into other pre-defined categories like “Family”, “Close Friends”, “Acquaintances” so that a distinct affinity level could be imposed on relations. This is how people, in real-world, intend to organize their relationships. Aforementioned list of relationship categories is common across all user profiles. This notion of categorizing (or listing of) friends into affinity levels help users to specify who can have access to their information. Users are also provided with an option to create "Private Lists". When a user posts something, the post is labelled with one of these list names. Any requester who satisfies membership to the label assigned with the post can access the post. Since members of these aforementioned lists are finite and known to the list owner, this category of labels is called *extensional* labels. And the information published using this type of labels is classified as extensional information. In other words, extensional labels are labels in user's local namespace.

There is another set of labels for information classification that is *intensional.* The labels that fall under this category are: "Friends of friends", "Public", and the set of "Social Lists". Social Lists[1] are automatically added to user's account at the time of profile creation/update, when a user provides her school, university, affiliation information. A post labelled with intensional label is available to a requester who satisfies membership to it. Owners of such posts do not have *complete* control over who will be accessing the intensional information because membership to intensional labels is not directly under their control. In other words, intensional labels are labels in user's extended namespace.

Labels are used as access control policies over a user's information. A typical access policy consists of a label available to the user. Facebook also provides *Custom* policies that involves a combination of labels. Custom policy's interpretation involves *set algebraic union and intersection* operations over labels used to compose it. We shall study the exact evaluation sequence of labels, at the time of access, later. The whole gamut of information labelling in Facebook provides a very rich and flexible access (thus privacy) policy specification over a user's information. Users are allowed to change labels of their objects as per their discretion. However, this flexibility in policy specification is not well-understood by majority of the users and users end up in a state where their policy specification may look innocuous, whereas it may not. We shall study such scenarios in next section. Given below is a typical set of labels (information classification categories) provided to express access control policies:

- Only Me: is a label/list in which user herself is the only member
- Public: is a label, when used, the associated object is accessible publicly
- Friends: is the primary list under which all friendship relations are enlisted
- Restricted: is a list of friends to whom only Public labelled information is allowed
- Family: is a list of friends who are assigned as family members
- Close Friends: is a list of friends who are assigned as close friends
- Acquaintances: is a list of friends who are assigned as acquaintances
- Friends of friends: is an *intensional* list consisting of users who have friendship relation with some member in "Friends" list, therefore;

[1] Facebook categorizes this type of list as Smart List because they are created based on common affiliations across the Facebook users. Facebook treats Close Friends, Family also as Smart Lists because, based on interactions among users, smart membership suggestions to these categories are provided by Facebook. To disambiguate, we use the term Social List.

(E_9) $\mathcal{E}_{ffriends}(user) = \{subject \mid \exists\ y\ s.t.\ (user, y) \in \mathcal{E}_{friends}$ $and\ (y, subject) \in \mathcal{E}_{friends}\}$

- *University*: is a social list of friends who are also members of Smart List *University*
- *School*: is a social list of friends who are also members of Smart List *School*
- *Cycling*: is a Private List to which user has assigned a set of friends

Let us understand the usage of labels for access control with an example: assuming current state of social graph is represented by $\textcircled{A} \xleftrightarrow{\text{friends}} \textcircled{B} \xleftarrow{\text{follower}} \textcircled{Z}$ Let P_1 be a post by user B. Access to object P_1 is determined by the access policy associated with P_1. User B can change access policy of his objects it owns at will. Let us discuss access implications of different policies on P_1

1. B $\xrightarrow[\text{Only Me}]{\text{authored}}$ P_1 : since B himself is the only member of list named “Only Me”, no one else except B will have access to P_1
2. B $\xrightarrow[\text{Friends}]{\text{authored}}$ P_1 : since $\mathcal{E}_{friends}(B) = \{\text{A}\}$ [cf. (E_2)], user A can access P_1
3. B $\xrightarrow[\text{Public}]{\text{authored}}$ P_1 : since the policy is “allow all”, both A and Z can access object P_1
4. B $\xrightarrow[\text{Only Me}]{\text{authored}}$ $P_2 \xrightarrow{\text{tagged}}$ A : since $\mathcal{E}_{tagged}(P_2) = \{\text{A}\}$ [cf. (E_8)], even though the access policy on P_2 is “Only Me”, user A will be able to access object P_2. Tagging can be seen as adding an exception to the default access policy of an object.

Access Control of objects in Facebook is a simple check on associated list's membership. If a requester of an object is a member of the list with which the object is protected, the requester gets access. Tagging is a positive exception to the membership check. There are two negative exceptions to the membership check: “Restricted” list and “Blocked” list. If a requester of an object is member of one of these lists, access is denied even when the requester is member of the list with which the object is protected.

When a *Custom* policy is used for access control, sequence of evaluation of lists used to build *Custom* policy becomes important. In the following we explain how policies are evaluated and enforced at the time of access.

3.2 Policy Evaluation and End-to-End Enforcement

Each object in Facebook is labelled with an access policy, which is a combination of labels with positive/negative exceptions. At the time of access, policies are evaluated in context of object's owner. In other words, evaluation of labels (i.e., Lists) is done in the namespace of object's owner. Every object has an access policy. Users are allowed to change access policies of own objects. Any combination of labels, from a user's namespace, is permitted to express access policies.

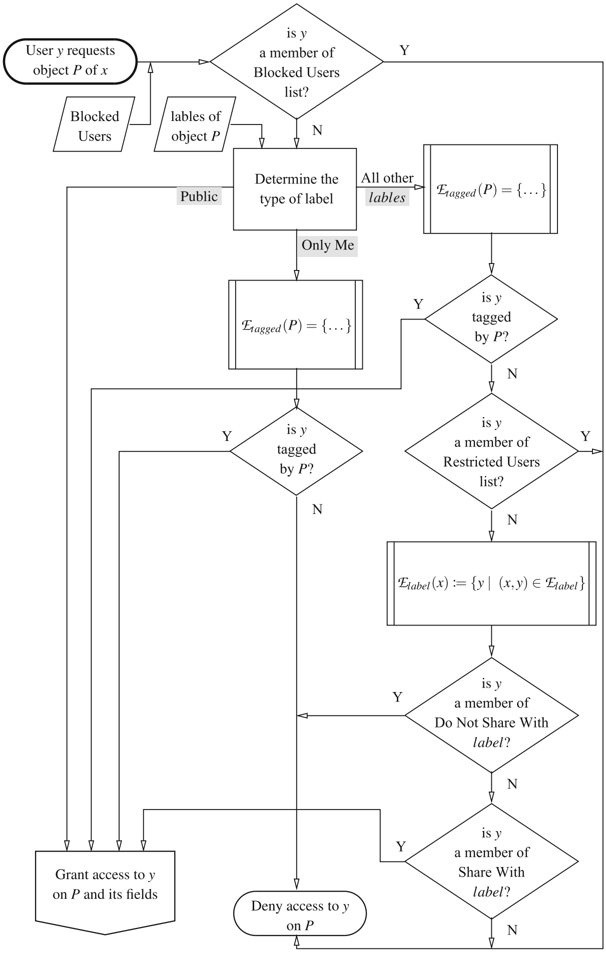

User y requests object P of x
is y a member of Blocked Users list?
Y
N
Blocked Users
lables of object P
Determine the type of label
Public
All other *lables*
Only Me
$\mathcal{E}_{tagged}(P) = \{\dots\}$
$\mathcal{E}_{tagged}(P) = \{\dots\}$
is y tagged by P?
Y
N
is y tagged by P?
Y
N
is y a member of Restricted Users list?
Y
N
$\mathcal{E}_{label}(x) := \{y \mid (x,y) \in \mathcal{E}_{label}\}$
is y a member of Do Not Share With *label*?
Y
N
is y a member of Share With *label*?
Y
N
Grant access to y on P and its fields
Deny access to y on P

Fig. 2. Sequence of policy evaluation [cf. (E_1), (E_8)]

Figure 2 outlines the sequence of policy evaluation and enforcement when user y requests an access to object P. All labels, except the labels “Public” and “Only Me”, are treated as Custom policy internally. Since a Custom policy is nothing but a policy composed of a user’s extensional and intensional labels and exceptions.

In other words, a Custom policy is a combination of two fields: “Share with” and “Do not share with”. The “Share with” field can accept any user label, except “Public”. The “Do not share with” field accepts any user label, except “Public”, “Friends”, and “Friends of friends”. Both the fields accept user’s individual friends. Every user profile is provisioned with two lists: Blocked Users and Restricted Users. Blocked Users consists of any user on Facebook with whom the user does not want to interact. If a user blocks a current friend, the relationship between them gets unfriended. Restricted Users consists of friends of the user who will be restricted to have access to the user’s public posts only, unless tagged. These two lists are not allowed to be used to express access policy but they are internally handled at the time of each access enforcement.

Table 1. Snapshots of associations formed in a social graph

$\xrightarrow{time}$

t_1	t_2	t_3	t_4	t_5
A $\xrightarrow[\text{OnlyMe}]{\text{authored}} P_{A_1}$	A $\xrightarrow{\text{authored}} P_{B_1}$	A $\xrightarrow[\text{Public}]{\text{authored}} P_{A_1}$	A $\xrightarrow{\text{likes}} P_{F_2}$	A $\xrightarrow{\text{likes}} P_{F_3}$
B $\xrightarrow[\text{Family}]{\text{authored}} P_{B_1}$	B $\xrightarrow[\text{Public}]{\text{authored}} P_{D_1}$	B $\xrightarrow[\text{Friends}]{\text{authored}} P_{B_1}$		
C $\xrightarrow[\text{Friends}]{\text{authored}} P_{C_1}$	C $\xrightarrow{\text{authored}} P_{D_1}$	C $\xrightarrow{\text{authored}} P_{C_1}$	C $\xrightarrow{\text{authored}} P_{B_1}$	C $\xrightarrow{\text{likes}}$ (A $\xrightarrow{\text{authored}} P_{B_1}$)
D $\xrightarrow[\text{FFriends}]{\text{authored}} P_{D_1}$	D $\xrightarrow{\text{likes}} P_{E_1}$			
E $\xrightarrow[\text{Public}]{\text{authored}} P_{E_1}$	E $\xrightarrow{\text{likes}} P_{F_1}$	E $\xrightarrow{\text{likes}} P_{D_1}$	E $\xrightarrow[\text{FFriends}]{\text{authored}} P_{F_3}$	E $\xrightarrow[\text{OnlyMe}]{\text{authored}} P_{E_1}$
F $\xrightarrow[\text{School}]{\text{authored}} P_{F_1}$	F $\xrightarrow[\text{School-E}]{\text{authored}} P_{F_2}$	F $\xrightarrow[\text{University-School}]{\text{authored}} P_{F_3}$	F $\xrightarrow{\text{likes}} P_{E_1}$	F $\xrightarrow{\text{likes}} P_{A_1}$

Assumptions:

at time t_0 (following is the state of different sets)
users = {A, B, C, D, E, F} *objects* = $\{P_{A_1}, P_{B_1}, P_{C_1}, P_{D_1}, P_{E_1}, P_{F_1}, P_{F_2}, P_{F_3}\}$
friendship edges = {(A,B), (B,C), (C,D), (D,E), (E,F), (F,A)} Family$_B$ = {A}
University = {E, F} School = {A, E, F}

at time t_4 (user E has disassociated from list School) School = {A, F}

at time t_5 University = {A, E, F} School = {F}

3.3 Reasoning About Access Control in Facebook w.r.t. Social Graph

To analyze the access control on Facebook, two broad categories of information represented in Facebook becomes very handy: (i) The utility of social graph in Facebook to record relationships and interactions among users and their objects (content). (ii) The way Facebook allows its users to categorize their relationships through intensional, extensional classes and, in turn, using these class names as labels on their respective objects to express access policies.

Table 2. Node reachability in social graph at t_1

At time t_1
• P_{A_1} is accessible to its owner alone
• P_{B_1} is accessible to A, as B has added A to list Family
• P_{C_1} is accessible to B and D [cf. (E_2)]
• P_{D_1} is accessible to all except A [cf. (E_9)]
• P_{E_1} is accessible to all
• P_{F_1} is accessible to A and E

Now let us reason about access control in Facebook with respect to an evolving social graph. Various possible associations that are formed between users and objects are captured through five snapshots depicted in Table 1. (cf. Appendix A) Table 2 describes accessibility of objects to users at time t_1. Object can be operated upon (e.g., to like/comment) by a requester only when the object is accessible. Tables 3, 4, 5 and 6 describe the actions taken or events created by users between time t_2 and t_5.

Table 3. Events and actions at t_2

At time t_2
• A comments on object P_{B_1}
• B shares object P_{D_1} as Public
• C comments on object P_{D_1}
• D likes object P_{E_1}
• E likes object P_{F_1}
• F creates object P_{F_2}, accessible to A

Table 4. Events and actions at t_3

At time t_3
• A changed access policy of P_{A_1} from Only Me to Public, so it becomes accessible to all
• B changed access policy of P_{B_1} from Family to Friends, so it becomes accessible to A and C
• C comments on her own post
• E likes P_{D_1}
• F creates object P_{F_3} with with custom policy in which "Share with = University" and "Do not share with = School". The object is not accessible to anyone. University and School are social lists and they will be evaluated in the context of user F. Therefore University$_F$ = {E} and School$_F$ = {A,E}. Though user E is part of "Share with" field of this custom policy, he will not get access to P_{F_3} because "Do not share with" is evaluated before "Share with" (cf. Fig. 2)

Table 5. Events and actions at t_4

At time t_4
• A likes object P_{F_2}
• C comments on object P_{B_1}. This object was not accessible to C until t_2
• E disassociates from list School
• E shares P_{F_3} with FFriends. Though user A is a member of user E's FFriends, she will not get access to P_{F_3} because it is governed by its owner's base policy University-School.
• F likes object P_{E_1}

Table 6. Events and actions at t_5

At time t_5
• A disassociates from list School
• A likes object P_{F_3} as it is now accessible, since she has associated herself with list University and is no more part of list School
• C likes the comment made by A on P_{B_1}
• E changes the access policy of P_{E_1} from Public to Only Me
• F likes object P_{A_1}

4 Analysis of Privacy-Preservation in Facebook Through User Specified Policies/Actions

Issue of privacy comes to fore as soon as an unintended observer observes an information and learns something more, which later could be associated with that subject under observation. The observed information need not necessarily be "personally identifiable information" as defined in prevalent definitions of privacy [13,21]. We believe that users trust Facebook as they voluntarily divulge [19] personal information, during profile creation and thereupon, to Facebook. In this section we analyze some of the instances of privacy violations in Facebook using the analysis done in Sects. 2 and 3. All of these breaches have been validated using Facebook as of May 2017.

Our analysis of Facebook policies using its access control is elaborated using a hypothetical scenario captured in Table 1. In this table, each row represents a user's actions in chronological fashion. Therefore, we use C.t_4 to denote an action of user C at time t_4. All other actions, of every user, up to time t_4 is the environment/status of social graph w.r.t. C.t_4. Thus, an action should be analyzed in context with its current environment. Note that, *since social graph is a co-creation by its users, an individual has little or no control over the environment in which he/she is operating. An action/setting that seems privacy-preserving can later be compromised by a change in environment.* This will become clear as we navigate through the scenarios.

Nonrestrictive Change in Policy of an Object Risks Privacy of Others. Consider user action A.t_2 in context with environment trace B.t_1, B.t_3. User B has changed policy of his object P_{B_1} from Family to Friends. Since user A is member of B's Family, through action A.t_2 user A has authored a comment on P_{B_1}. The environment at time t_2 ensured that only Family members of B had access to object P_{B_1}. Nonrestrictive change in policy from Family to Friends on P_{B_1} gets enforced on all of its dependent nodes (*comment, reply*) and fields (*like*). *User A's comment is exposed to friends of B without A's consent.*

Restrictive Change in Policy of an Object Suspends Others' Privileges. Consider user action E.t_5 in context with two other events in environment D.t_2, F.t_4. User E has changed policy of her object P_{E_1} from Public to Only Me. Prior to policy change, users D and F have liked the object. When a query on "likes of a user" is made to social graph at time t_4, object P_{E_1} will be listed in the reply. Restrictive change in policy from Public to Only Me on P_{E_1} gets enforced on all of its dependent nodes (*comment, reply*) and fields (*like*). A "likes of user D" query will not list object P_{E_1} at t_5, neither user D/F will be able to disassociate themselves from the object under control of user E. The like edges to P_{E_1} from D and F will only be accessible again on social graph when user E makes a nonrestrictive policy change on P_{E_1}. Assume P_{E_1} is a sensitive post to which some users have liked/commented. *A restrictive change in policy over P_{E_1} locks out users from updating/retracting their own comments or likes.*

At a later point in time, user E can divulge list of users associated with her post from the past.

***Share* Operation is Privacy-Preserving.** Consider user action B.t_2 in context with environment D.t_1. Since user B is member of D's "Friends of friends" list, object P_{D_1} appears on his *Newsfeed*, which he shares by action B.t_2. Sharing an object of others creates a local node and it is allowed to specify access policy on this new node under the control of sharer. This shared node can have comments, likes as any other object node created by the sharer. However the sharer cannot increase or decrease the accessibility sphere of the original object (on social graph) which it has shared. Original access policy of the base object continues to be carried along with the object in its shared form. Thus, the intended reach of the object by its original author is always enforced. *Restrictive or non-restrictive change in policy on base object reflects wherever the object exists on social graph in a shared form. This is similar to the notion of capability lists* [15] *except that instead of user carrying the capability list, the object is carrying it.*

Policy Composition Using Intensional Labels Is Not Privacy-Preserving. Consider user actions F.t_3, E.t_4, A.t_5 in context with social list "School" at t_4 and t_5. Through action in F.t_3, user F has created an object P_{F_3} with a custom policy University-School. Here the intention of the user is to make the object available to his friends from University but not from his School. According to the state of social graph at time t_3 nobody gets access to object P_{F_3} because University $\subseteq$ School. At time t_4, user E disassociates herself from social list School and thus could get access to P_{F_3}. Through action in E.t_4, user E shares P_{F_3} with access policy as School. As shared objects carry their access policy wherever they exist on social graph, user A will not receive access to P_{F_3} due to E.t_4. Whereas by disassociating herself from social list School, user A will have access to P_{F_3} at time t_5 as shown in A.t_5. *Disassociation from a social list allows users to bypass the privacy/access intention of a custom policy when an intensional label is part of "Do not share with" field. Similarly, association with a social list allows users to bypass the privacy/access intention of a custom policy when an intensional label is part of "Share with" field. Note that "Friends of Friends" is also an intensional label.*

***Like, Comment* Operations are Not Privacy-Preserving.** Consider user actions D.t_2 and F.t_4 in context with environment event E.t_1. On Facebook, *List of Friends* is an object of user profile. In its privacy settings, Facebook allows to choose intended audience for this object. We assume all users in this scenario have set their audience to "Only Me" for this object. The intention behind such a setting is not to let the profile visitors know who their friends are, except their mutual friends. However, the way Facebook works, *Newsfeed* of a user is supplied with relevant content from user's social circle. With a high probability friends' posts appear in *Newsfeed* to which a user may interact by making a comment or like. These interactions get recorded on social graph. *When a user interacts with objects with access policy set to Public, those interactions also become public. Social graph allows queries to public content.* For example, https://fb.com/search/FBID-Alice/photos-commentedon returns all the "photo" type of objects on which Alice has commented. Similarly, /photos-liked returns all photos liked by Alice. *For a typical user, these queries return objects from their friends. Any user of Facebook can make these queries to social graph for any other user of Facebook.* For a complete list of search attributes please refer Facebook API page [8].

5 Is There a Way to Preserve the Intentions of Policies?

Having seen scenarios of breach of policies, the question is: is it possible to adhere to the intended privacy in the policies without compromising too much on the performance or is it possible to consider policy streamlining to avoid such breaches? Due to lack of space (cf. [20]), we shall briefly discuss some of the possible approaches below.

Intercept and Resolve Intensional Labels at the Time of Object Creation. Association and disassociation with intensional label of type social list is unverified, unsupervised and easy. Whereas associating with Friends of Friends label of a user is relatively tough but possible. To restrict users from bypassing access control when intensional label is part of access policy, one may think of resolving intensional labels into its prevalent member set and use list of member IDs as explicit custom policy. To automate this process we are experimenting with a browser extension that intercepts, resolves intensional labels and writes equivalent custom policy on-the-fly.

Anonymize Unsafe Operations via a *Proxy* node. Information leakage on *List of Friends* object has serious security implications. By knowing friends of a user an attacker creates profiles similar to friend of the user and launch a spear phishing attack. Therefore, it is important to prevent this leakage *at least* when the object's policy is set to Only Me. Collecting likes on objects is one of the important input Facebook relies on for *Newsfeed* content and targeted advertising. Without forgoing these objectives, Facebook can introduce a special node to its social graph called *Proxy* node. All like and comment operations of users with privacy setting as Only Me for *List of Friends* object should be rerouted through this *Proxy* node on social graph.

Treat Comment Nodes Similar to Nodes with Access Policy. A comment is a node on social graph of Facebook. However comment type of nodes have shared ownership as long as the post on which comment is made is accessible to the commenter. Once the owner of the post changes post policy to a restrictive one, commenter's ownership is suspended until the node policy is reverted. Facebook can extend the treatment of post type of nodes to comment type of nodes.

6 Related Work

The access control mechanism of Facebook is ad-hoc and hybrid therefore it is different from standard, prevalent lattice-based access control models [22] like Mandatory Access Control (MAC), Discretionary Access Control (DAC), Role-Based Access Control (RBAC), Capability Systems [15] etc. Facebook's access control mechanism is based on the topology of the underlying volatile social graph. Content and identities are represented by nodes in the graph and edges represent their relationships with other nodes. An access is granted to a requester (a node on the social graph) only if there exists a path to the node being accessed and the requester fulfils the access control specified on the node by its owner. Access control in Facebook involves a subtle element of delegation similar in RBAC [1,6] in the midst of discretionary access control [11,16]. Access control mechanism of Facebook is a function of communication history among users (e.g., existence of friendship is necessary for certain policies), which has parallels with characterizations presented in [23] and works presented in [18] has a security and privacy conformance model based on labels controlled by independent domains.

Owing to the difficulty for users to fully comprehend the privacy consequences of adjusting their privacy settings or re-adjusting their relationships; Fong et al. [10], presented a paradigm for access control in Facebook-style SNSs and also discussed the possibility of overcoming Sybil attacks [9] using the conditions (prevalent on Facebook in year 2009) needed for satisfying Denning's principle of privileged attenuation (POPA); it easily follows from our illustrated privacy breaches that POPA cannot be preserved in the current setup of Facebook. In [5], a rule-based access control mechanism for Facebook-style SNSs is presented. The mechanism relies on relationship inputs (node's type, trust level of relationships, etc.) from underlying social graph. Social graph of Facebook, as of today, does not provide a normal user the attributes required to write rules that are presented in it.

In works on web transparency and accountability [17], it has been argued that vast amounts of profile tracking leads to various breaches of policies and hence there is a need for regulations on privacy while profile tracking. In [2], the authors present an inference, from social graph queries about friendship, that eight friendship links are enough to construct a "public view" of a user. Works highlighting the importance of privacy to friendship links are presented in [12,14]. In future, privacy will be *the most* important parameter for any application's acceptability, especially so for applications dealing with social networks [19].

7 Conclusion

Through our experiments on Facebook, we have discovered certain user actions and configurations that undo the privacy guarantees set by the user. We presented our findings with the help of a hand-crafted scenario and proposed plausible mitigation techniques for privacy-preservation. Our mitigation techniques in the existing access control model would pave way for a general purpose access control model for global-scale social applications. It is of interest to note that there is a serious impact on privacy due to App ecosystem of Facebook. While the impact of cross-domain access can be analyzed through several techniques like the one explored in [18], the aspect just pointed out need further exploration. Assuming there will be a regulation for privacy [19], one would need to understand how compliance of regulations on the policy settings can be ensured.

Acknowledgement. The work was carried out as part of research at ISRDC (Information Security Research and Development Center), supported by 15DEITY004, Ministry of Electronics and Information Technology, Govt of India.

A Appendix

Figure 3 shows instance of the graph at time t_1. Each *user* node of the graph has authored an object. Their access policies are marked on respective $(user, object)$ edges. Therefore, at time t_1, following are the objects introduced to the social graph. Accessibility to these objects is evaluated according to their respective labels as described in Table 2.

Figures 4, 5 and 6 show the state of social graph at time t_2, t_4, and t_5, respectively.

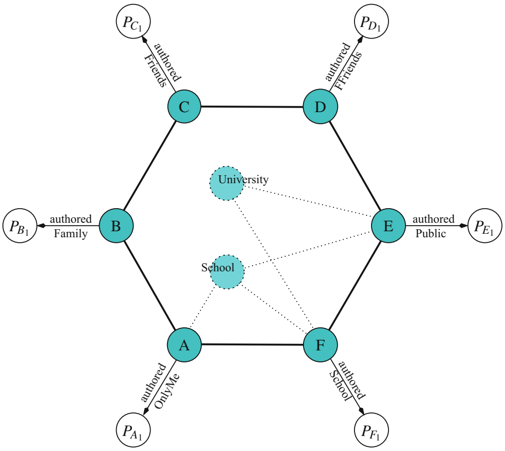


Fig. 3. Social graph at time t_1

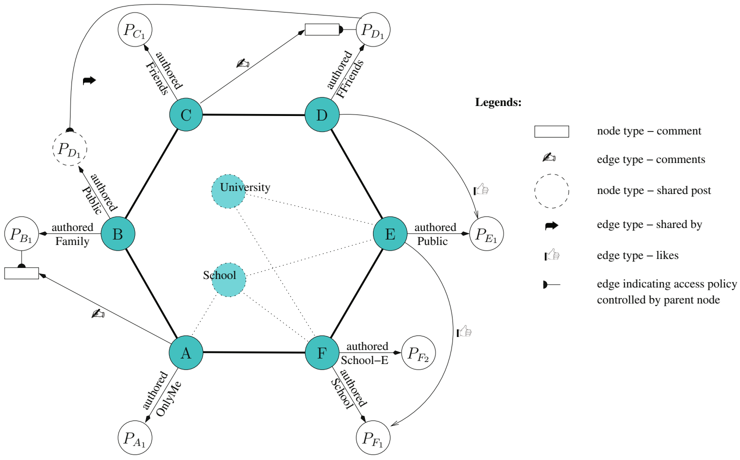


Fig. 4. Social graph at time t_2

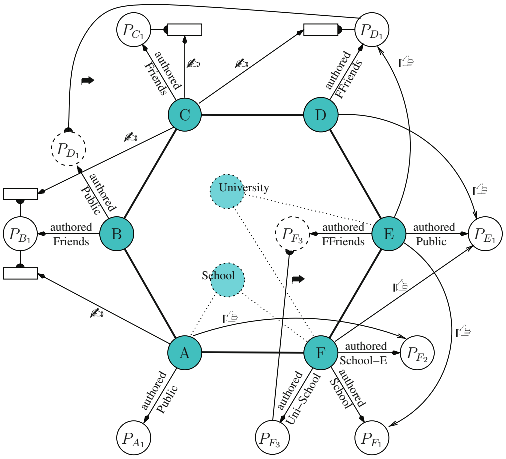


Fig. 5. Social graph at time t_4

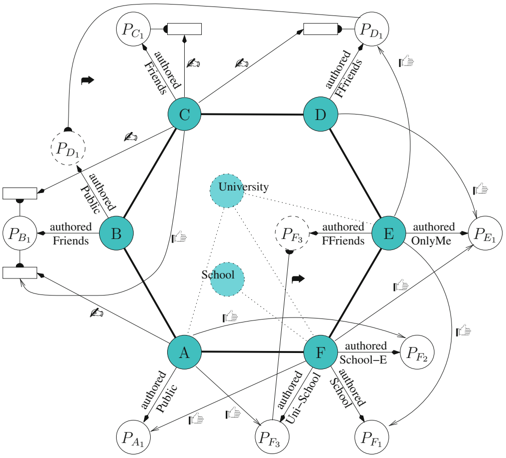


Fig. 6. Social graph at time t_5

References

1. Barka, E., Sandhu, R.: Framework for role-based delegation models. In: Proceedings of the 16th Annual Computer Security Applications Conference, p. 168. IEEE Computer Society (2000)
2. Bonneau, J., et al.: Eight friends are enough: Social graph approximation via public listings. In: 2nd EuroSys Workshop on Social Network Systems, SNS 2009, pp. 13–18. ACM (2009)
3. Boyd, D.M., Ellison, N.B.: Social network sites: Definition, history, and scholarship. J. Comput.-Mediated Commun. **13**(1), 210–230 (2007)

4. Bronson, N., Amsden, Z., et al.: TAO: facebook's distributed data store for the social graph. In: USENIX ATC 2013, pp. 49–60 (2013)
5. Carminati, B., Ferrari, E., Perego, A.: Enforcing access control in web-based social networks. ACM TISSEC **13**(1), 6:1–6:38 (2009)
6. Crampton, J., Khambhammettu, H.: Delegation in role-based access control. Int. J. Inf. Secur. **7**(2), 123–136 (2008)
7. Curtiss, M., Becker, I., Bosman, T., et al.: Unicorn: a system for searching the social graph. Proc. VLDB Endow. **6**(11), 1150–1161 (2013)
8. Facebook: Graph API Overview (2017). https://developers.facebook.com/docs/graph-api/overview/
9. Fong, P.W.L.: Preventing sybil attacks by privilege attenuation: a design principle for social network systems. In: IEEE Symposium on Security and Privacy, pp. 263–278 (2011)
10. Fong, P.W.L., Anwar, M., Zhao, Z.: A privacy preservation model for facebook-style social network systems. In: Backes, M., Ning, P. (eds.) ESORICS 2009. LNCS, vol. 5789, pp. 303–320. Springer, Heidelberg (2009). doi:10.1007/978-3-642-04444-1_19
11. Graham, G.S., Denning, P.J.: Protection: principles and practice. In: Proceedings of the Spring Joint Computer Conference, AFIPS 1972, 16–18 May 1972, pp. 417–429. ACM (1972)
12. Hangal, S., Maclean, D., Lam, M.S., Heer, J.: All friends are not equal: using weights in social graphs to improve search. In: 4th SNA-KDD Workshop. ACM (2010)
13. International Association of Privacy Professionals: What is privacy? (2017). https://iapp.org/about/what-is-privacy/
14. Jernigan, C., Mistree, B.: Project 'Gaydar' Computes Orientation. In: CACM (2009). http://cacm.acm.org/news/42191-project-gaydar-computes-orientation
15. Levy, H.M.: Capability-Based Computer Systems. Digital Press, Bedford (1984)
16. Li, N., Tripunitara, M.V.: On safety in discretionary access control. In: 2005 IEEE Symposium on Security and Privacy (SP 2005), pp. 96–109, May 2005
17. Narayanan, A., Reisman, D.: The princeton web transparency and accountability project. In: Cerquitelli, T., Quercia, D., Pasquale, F. (eds.) Transparent Data Mining for Big and Small Data. Studies in Big Data, vol. 11, 45–57. Springer, Cham (2017)
18. Narendra Kumar, N.V., Shyamasundar, R.K.: Dynamic labelling to enforce conformance of cross domain security/privacy policies. In: Krishnan, P., Radha Krishna, P., Parida, L. (eds.) ICDCIT 2017. LNCS, vol. 10109, pp. 183–195. Springer, Cham (2017). doi:10.1007/978-3-319-50472-8_15
19. Patil, V.T., Shyamasundar, R.K.: Privacy as a currency: un-regulated? In: 14th International Conference on Security and Cryptography, SECRYPT 2017 (2017, to appear)
20. Patil, V.T., Shyamasundar, R.K.: Social networks and collective unravelling of privacy. Technical report, ISRDC, IIT Bombay (2017). http://isrdc.iitb.ac.in/reports/isrdc-tr-2017-rks-vtp-sns-privacy.pdf
21. Renaud, K.G.D.: Privacy: Aspects, definitions and a multi-faceted privacy preservation approach. In: 2010 Information Security for South Africa, pp. 1–8, August 2010
22. Sandhu, R.S.: Lattice-based access control models. Computer **26**(11), 9–19 (1993)
23. Schneider, F.B.: Enforceable security policies. ACM TISSEC **3**(1), 30–50 (2000)