Link Privacy in Social Networks

Aleksandra Korolova, Rajeev Motwani, Shubha U. Nabar, Ying Xu
Computer Science Department, Stanford University
{korolova, rajeev, sunabar, xuying}@cs.stanford.edu

***Abstract*— We consider a privacy threat to a social network in which the goal of an attacker is to obtain knowledge of a significant fraction of the links in the network. We formalize the typical social network interface and the information about links that it provides to its users in terms of lookahead. We consider a particular threat in which an attacker subverts user accounts to gain information about local neighborhoods in the network and pieces them together in order to build a global picture. We analyze, both experimentally and theoretically, the number of user accounts an attacker would need to subvert for a successful attack, as a function of his strategy for choosing users whose accounts to subvert and a function of the *lookahead* provided by the network. We conclude that such an attack is feasible in practice, and thus any social network that wishes to protect the link privacy of its users should take great care in choosing the lookahead of its interface, limiting it to 1 or 2, whenever possible.**

I. Introduction

Participation in online social networks is becoming ubiquitous. A major part of the value of participating in an online social network, such as LinkedIn, or a web-service with an online community, such as LiveJournal, for a user lies in the ability to leverage the structure of the *social network graph*. However, knowledge of this social graph by parties other than the service provider opens the door for powerful data mining, some of which may not be desirable to the users.

The motivation for this paper is networks such as LinkedIn, where relationships between users may be sensitive to privacy concerns, and the link information is a valuable asset to the user and to the network owner. In such networks, a user is typically permitted only limited access to the link structure. For example, a LinkedIn user can only see the profiles and friends lists of his friends and the profiles of friends of friends.

Even though each user is given access only to a small part of the social network graph, one could imagine a resourceful adversarial entity trying to stitch together local network information of different users in order to gain global information about the social graph. In this paper we focus on the case in which an attacker, whose goal is to ascertain a significant fraction of the links in a network, obtains access to parts of the network by gaining access to the accounts of some select users. This is done either maliciously by breaking into user accounts or by offering each user a payment or service in exchange for their permission to view their neighborhood of the social network. We describe both experimental and theoretical results on the success of such an attack depending on the type of local neighborhood access permitted by the network owner to each individual user.

Our work is different from the recent work of [1], [2] concerned with user privacy in anonymized social network graph releases. In that setting, a social network owner releases the underlying graph structure after removing all username annotations of the nodes. The goal of an attacker is to map the nodes in this anonymized graph to real world entities. In contrast, we consider a case where no underlying graph is released and, in fact, the owner of the network would like to keep the link structure of the entire graph hidden from any one individual.

II. The Model

A. Goal of the Attack

We view a social network as an undirected graph $G = (V, E)$, where the nodes V are the users and the edges E represent connections or interactions between users. As was discussed in the Introduction, the primary goal of the privacy attack is to discover the link structure of the network. We measure an attack's effectiveness using the notion of *node coverage*, or simply *coverage*, which measures the amount of network graph structure exposed to the attacker.

Definition 1 (Node Coverage): The fraction of nodes whose entire immediate neighborhood is known. We say that a node is *covered*, if and only if the attacker knows precisely which nodes it is connected to and which nodes it is not connected to.

B. The Network through a User's Lens

As mentioned in Section I, LinkedIn allows a user to see all edges incident to oneself, as well as all edges incident to one's friends. An online social network could choose the extent to which links are made visible to its users depending on how sensitive the links are and we quantify such choices using *lookahead*. We say that the social network has lookahead of 0 if a user can see exactly who he links to; it has lookahead 1 if a user can see exactly the friends that he links to as well as the friends that his friends link to. In general, we say that the social network has *lookahead* ℓ if a user can see all of the edges incident to the nodes within distance ℓ from him. Using this definition, LinkedIn has lookahead 1. In terms of node coverage, a lookahead of ℓ means that each node covers all nodes within distance ℓ from it; nodes at distance $\ell + 1$ are *seen* (i.e., their existence is known to the user), but not *covered* (i.e., their connections are not known to the user).

There are other variations on the type of access that a user can have to the social graph structure. For example, some networks allow a user to see the shortest path between himself and any other user. We ignore these additional options in our discussion, while noting that the presence of any of them simplifies the task of discovering the entire link structure.

In addition to the connection information, a typical online social network also provides a search interface, where people can search for users by username, name or other identifying information such as email and affiliation. The search interface returns usernames of all users who satisfy the query, often with the numbers of friends of those users, i.e., the degrees of the nodes corresponding to those users in the social network graph, G. LinkedIn is an example of a social network that allows such queries and provides degree information. We formalize the various aspects of social network interfaces that may be leveraged by attackers to target specific user accounts below:

- *neighbors(username, password, ℓ)*: Given a username with proper authentication information, return all users within distance ℓ and all edges incident to those users in the graph G;
- *exists(username)*: Given a username, return whether the user exists in the network;
- *degree(username)*: Given a username, return the degree (number of friends) of the user with that username. Note that *degree(username)* implies *exists(username)*;
- *userlist()*: Return a list of all usernames in the network.

A social network might expose some or all of these functions to its users, either explicitly or implicitly. LinkedIn provides *neighbors(username, password, ℓ)* for $\ell = 0$ or 1, but not for $\ell > 1$; it also provides *exists(username)* and *degree(username)*. Most social networks do not expose *userlist()* directly; however, an attacker may be able to generate a near complete user list through other functionalities provided by the network such as fuzzy name search.

C. Possible Attack Strategies

Recall that each time the attacker gains access to a user account, he immediately covers all nodes that are at distance less than the lookahead distance ℓ enabled by the social network, i.e., he learns about all the edges incident to these nodes. Thus by gaining access to user u's account, an attacker immediately covers all nodes within distance ℓ of u. Additionally, he gets to learn about the existence of (”sees”) all nodes within distance $\ell+1$ from u. We call the users to whose accounts the attacker obtains access *bribed* users.

A natural question that arises is how does an attack's success or attained node coverage vary depending on the strategy followed for picking the users to bribe. We list the strategies we study in the decreasing order of information needed for the attacker to be able to implement them and study the success of attacks following these strategies in Section III and IV.

Greedy: From among all users in the social network, pick the next user to bribe as the one with the maximum “unseen” degree, i.e. its degree according to the *degree(username)* function minus the number of edges incident to it already known to the adversary.
Requires: *neighbors(username, password, ℓ), degree(username), userlist()*;

Highest-Degree: (abbreviated as **Highest**) Bribe users in the descending order of their degrees.
Requires: *neighbors(username, password, ℓ), degree(username), userlist();*

Random: Pick the users to bribe uniformly at random.
Requires: *neighbors(username, pwd, ℓ), userlist();*

Crawler: From among all users already seen, pick the next user to bribe is the one with the maximum unseen degree.
Requires: *neighbors(username, pwd, ℓ)*;

III. Experimental Results

In this section we present experimental results from the application of the four strategies discussed in Section II-C to real world social network data. At a high level, our experiments explore the fraction, f, of nodes that need to be bribed by an attacker using the different bribing strategies in order to achieve $1-\varepsilon$ node coverage of a social network with lookahead ℓ. Our experimental results show that the number of users an attacker needs to bribe in order to acquire a fixed coverage decreases exponentially with increase in lookahead. In addition, this number is also fairly small from the perspective of practical attack implementation, indicating that several of the attack strategies from Section II-C are feasible to implement in practice and will achieve good results.

We felt that bribing LinkedIn users with a goal of recovering the network's structure would be inappropriate as a research exercise, so we used the LiveJournal friendship graph, whose link structure is readily available, instead as a proxy. We crawled LiveJournal using the friends and friend-of listings to establish connections between users and extracted a connected component of $572,949$ users. The obtained LiveJournal graph has an average degree of 11.8.

We denote the fraction of nodes bribed by f, and the coverage achieved by $1-\varepsilon = \frac{number\ of\ nodes\ covered}{n}$.

A. Comparison of Strategies

For each strategy proposed in Section II-C, we calculate f as a function of $1-\varepsilon$ by averaging across multiple runs the fraction of nodes that need to be bribed using that strategy in order to achieve the desired coverage of the LiveJournal graph. The relative performance of the four strategies in a lookahead 2 graph is presented in Figure 1.

The **Highest-Degree** bribing strategy yields the best performance, i.e., to achieve a fixed coverage of $1-\varepsilon$, **Highest** needs to bribe fewer nodes than any other strategy. **Greedy** and **Crawler** perform equally well - their $f(1-\varepsilon)$ curves are almost overlapping. Not surprisingly, **Random** performs the worst out of all the strategies.

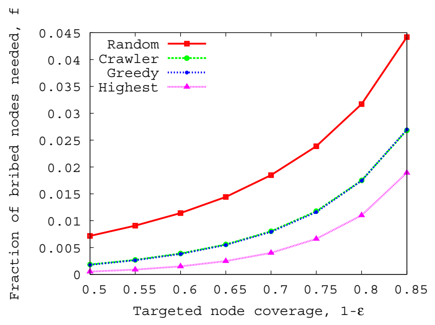
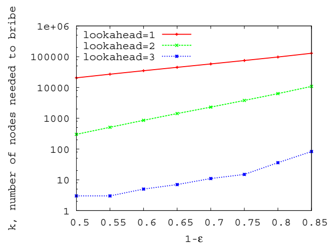


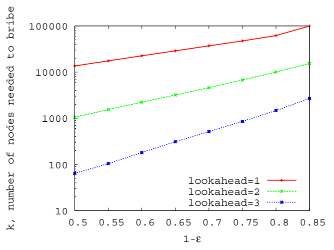
Fig. 1. **Comparison of attack strategies on LiveJournal data.** We plot the fraction of bribed nodes against node coverage with lookahead 2. The two lines for **Crawler** and **Greedy** are almost overlapping.

B. Dependence on Lookahead

The performance of all strategies substantially improves with increase in lookahead. Figure 2 shows that the number of nodes that need to be bribed in order to achieve fixed coverage of LiveJournal decreases exponentially with an increase in lookahead (forming a line in log plot). These experiments confirm our hypothesis that while none of the strategies are a truly feasible threat at lookahead 1, some of them become feasible at lookahead 2, and all of them become feasible at lookahead 3. For example, in order to obtain 80% coverage of the $572,949$-user LiveJournal graph using lookahead 2 **Highest** needs to bribe $6,308$ users, and to obtain the same coverage using lookahead 3 **Highest** needs to bribe 36 users – a number of users that is sufficiently small given the size of the network, and feasible to bribe in practice.



(a) **Highest**



(b) **Crawler**

Fig. 2. **Effect of lookahead on LiveJournal data.** The figures show the number of nodes needed to bribe to achieve $1-\varepsilon$ coverage with different lookaheads using **Highest** and **Crawler** respectively. The y axis is log scale.

IV. Theoretical Analysis for Random Power Law Graphs

In order to be able to extrapolate our results experimentally observed on the LiveJournal graph to a setting with an arbitrary social network graph, and because the degree distribution of social networks is commonly considered to be close to power law ([3], [4]), we performed a theoretical analysis of the performance of two of the bribing strategies from Section II, **Random** and **Highest-Degree**, on graphs with power law degree distribution. We use the configuration model in [5], which generates, uniformly at random, a graph that satisfies a given degree distribution, for our analysis.

Let n be the total number of nodes in G, α $(2 < \alpha \leq 3)$ be the power law parameter, and let d_0 and d_{max} be the minimum and maximum degree of any node in the graph, respectively. First, we generate the degrees of all the nodes $d(v_i), i = 1, \ldots, n$ independently according to the distribution $Pr[d(v_i) = x] = C/x^{\alpha}, d_0 \leq x \leq d_{max}$, where C is the normalizing constant. Second, we consider $D = \sum d(v_i)$ minivertices which correspond to the original vertices in a natural way and generate a random matching over D. Finally, for each edge in the matching, we construct an edge between corresponding vertices in the original graph. This gives us a random graph with the given power law degree distribution. The graph has been shown to be connected almost surely. Following common practice, we cap d_{max} at $\sqrt{n}$.

A. Analysis for Lookahead = 1

Theorem 1: If an attacker bribes $\frac{-\ln \varepsilon_0}{d_0} n$ nodes picked according to the **Random** strategy, then he covers at least $n(1 - \varepsilon - o(1))$ nodes with high probability, where $\varepsilon = \sum_{d=d_0}^{\sqrt{n}} \varepsilon_0^{d/d_0} C d^{-\alpha}$.

Theorem 2: If an attacker bribes $\left(\frac{-\ln \varepsilon_0}{d_0}\right)^{\frac{\alpha-2}{\alpha-1}} n$ nodes according to the **Highest-Degree** strategy, then he covers at least $n(1 - \varepsilon - o(1))$ nodes with high probability, where $\varepsilon = \sum_{d=d_0}^{\sqrt{n}} \varepsilon_0^{d/d_0} C d^{-\alpha}$.

Compare the two strategies: to cover a certain fraction of the nodes, an attacker needs to bribe much fewer nodes when using the **Highest-Degree** bribing strategy than when using the **Random** bribing strategy. For example, when $\alpha = 3$, if an attacker bribes an f fraction of the nodes with the **Random** strategy, then he only needs to bribe an f^2 fraction of the nodes using the **Highest-Degree** strategy to attain the same coverage. On the other hand, the bad news for an attacker targeting a social network that provides only lookahead of 1 is that even if he has the power to choose the highest degree nodes for an attack, a linear number of nodes will need to be bribed in order to cover a constant fraction of the graph.

B. Analysis for Lookahead > 1

When lookahead $\ell > 1$, our heuristic analysis shows that when $\alpha = 3$, using **Random** strategy, f needs to be approximately $\frac{-\ln \varepsilon_0}{d_0 b^{\ell}}$ to attain $1-\varepsilon$ coverage, where ε and ε_0 satisfy the equation in Theorem 1 and b is of the order $\ln n$; using **Highest-Degree** strategy, the attacker needs to bribe approximately $f = (\frac{-\ln \varepsilon_0}{d_0 b^{\ell}})^2$ of users. Our analysis can be easily extended to any $2 < \alpha \leq 3$.

Our analysis shows that the number of nodes needed to be bribed to cover a constant fraction of the network decreases exponentially with increase in lookahead ℓ. For example, with lookahead $\ell = \ln \ln n$, bribing a constant number of nodes is sufficient to attain coverage of almost the entire graph, making the link privacy attacks on social networks with lookahead greater than one truly feasible.

We generated synthetic data using the above model, and the simulation results match well with our theoretical analysis (for both $\ell = 1$ and $\ell > 1$). This confirms that our analysis is close to being tight. The detailed analysis can be found in [6].

Acknowledgement

Supported in part by NSF Grant ITR-0331640, TRUST (NSF award number CCF-0424422), and a grant from Google.

References

[1] L. Backstrom, C. Dwork, and J. Kleinberg, "Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography," in *16th international conference on World Wide Web*, 2007.
[2] M.Hay, G. Miklau, D. Jensen, P. Weis, and S. Srivastava, "Anonymizing social networks," *Technical Report*, 2007.
[3] S. Wasserman and K. Faust, "Social network analysis," 1994.
[4] Y. Medynskiy and J. Kaye, "Characterizing livejournal: Sccs, liverank, and six degrees," *Cornell Information Science Technical Report*.
[5] W. Aiello, F. Chung, and L. Liu, "A random graph model for power law graphs," *IEEE Symposium on Foundations of Computer Science*, 2000.
[6] A. Korolova, R. Motwani, S. Nabar, and Y. Xu, "Link privacy in social networks," *Technical Report*.