On the Feasibility of Inference Attacks by Third-Party Extensions to Social Network Systems

Seyed Hossein Ahmadinejad
Department of Computer Science
University of Calgary
Calgary, Alberta, Canada
shahmadi@ucalgary.ca

Philip W. L. Fong
Department of Computer Science
University of Calgary
Calgary, Alberta, Canada
pwlfong@ucalgary.ca

ABSTRACT

Social Network Systems (SNSs) providers allow third-party extensions to access users' information through an Application Programming Interface (API). Once an extension has been authorized by a user to access data in a user's profile, there is no more control on how that extension uses the data. This raises serious concerns about user privacy because a malicious extension may infer some private information based on the legitimately accessible information. This information leakage is called an *inference attack*. In addition, inference attacks are not only a privacy violation, they could also be used as the building blocks for more dangerous security attacks, such as identity theft. In this work, we conduct a comprehensive empirical study to assess the feasibility and accuracy of inference attacks that are launched from the extension API of SNSs. We also discuss an attack scenario in which inference attacks are employed as building blocks. The significance of this work is in thoroughly discussing how inference attacks could happen in practice via the extension API of SNSs, and highlighting the clear and present danger of even the naively crafted inference attacks.

Categories and Subject Descriptors

K.4.1 [**Computers and Society**]: Public Policy Issues—*Privacy*; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection—*Invasive software*

Keywords

Social networks; Privacy; Inference attacks

1. INTRODUCTION

Third-party applications are a popular feature of SNSs. For instance, there are applications with more than 50M monthly active users on the Facebook platform. Third-party developers host their Facebook applications on their own (untrusted) servers. Such applications then interact with users and access their personal information through an API. Every access request sent by an application via an API call is guarded by a permission check. If the permission is granted by the user, the data will be sent to the application.

Problem definition.

Not all third-party applications are benign. For example, a malicious application may sell user information to marketing companies. As a result, delivering user data via the extension API to such applications puts the data at risk. While this problem has to do with the misuse of *legitimately accessible* information, this work is instead about a more challenging problem: i.e., through the extension API, malicious applications may gain access to private information for which they are not authorized. The following is an example.

Example 1. *In a Facebook user profile is a "*wall*" on which the user or her friends may post status updates, photos, messages, etc. Now, a malicious third-party application asks for the permission to access the wall. Say the permission is granted by the user. The application accesses the user's wall through the extension API, and then scans the wall for a day when a considerable number of birthday greetings were posted. The application can thus infer the user's birthday.*

In the above example, if the individual considers her birthday as private information (and thus should be inaccessible), then the malicious application has effectively inferred some supposedly inaccessible information about the user from the information that is accessible through the extension API.

Note that an application may also have access to some information about an individual outside the SNS. So it might utilize such information in inferring the individual's inaccessible information. We use the term ***SNS API inference attacks*** to refer to the inference of inaccessible information from both accessible information and background information. The term highlights the fact that this work focuses on inference attacks that are launched by extensions to SNSs through the extension API. Our emphasis on the role of third-party extensions in launching inference attacks differentiates us from other related work on generic inference attacks over social network data sets [6, 5, 3].

Significance of the problem.

A naive interlocutor may argue that the above issue has already been addressed by the permission-based access control mechanism, in that third-party extensions cannot access user information without seeking the required permissions. If a user does not trust a third-party application, then she

ASIA CCS'13, May 8–10, 2013, Hangzhou, China.

shall not authorize it. This argument presumes that ordinary users have the necessary information and expertise to judge whether the applications they subscribe to are benign. In reality, most of the third-party applications are developed by developers who are not widely known to the user community. It is therefore not always possible for a user to assess if she can trust an application.

One may also claim that SNS API inference attacks are but another minor privacy violation. We disagree for two reasons. First, analyzing the threats of any security concern must be accompanied by assessing the number of potential victims. When the number of potential victims reaches, for example, 50M, then we are facing a trouble with costly consequences. Popular Facebook applications have 50M monthly users, implying that an inference attack with a meagre success rate of 10% leads to privacy violations of 5M victims.

Second, SNS API inference attacks can be employed as building blocks for conducting more dangerous security attacks. For instance, an alternative authentication mechanism is to ask users security questions (e.g., "what is the name of your youngest sibling?"). Answers to these security questions can usually be harvested systematically by launching SNS API inference attacks. The ability to answer a victim's security questions is the first step of identity theft. Therefore, inference attacks could be an initial step in the launching more dangerous attacks. Now, who is best positioned to launch covert inference attacks? The answer is third-party extension developers.

Contributions.

The specific contributions of this work are the following:

- By way of a comprehensive empirical study (Section 3), we develop deep insight into the problem of SNS API inference attacks, and demonstrate the growing threat of such attacks to user privacy.
- We devised an analytical framework for evaluating the risk of SNS API inference attacks. A key component of our framework is the classification of user profiles into four categories from the perspective of an adversary (Section 4). Based on this classification, we devised a scoring technique for assessing the success rate of inference algorithms (Section 6). The insights we gained can inform the design of protection mechanisms for mitigating the threat of SNS API inference attacks. Our analytical methodology can be applied for analyzing SNS API inference attacks not covered in this work.
- Eight (8) realistic inference algorithms are devised for our empirical study, covering a wide variety of inference techniques and inference channels. This diversity of inference techniques improves our understanding of the threat of SNS API inference attacks (Section 5).
- We examine a scenario, namely, identity theft, in which SNS API inference attacks are used as building blocks of more dangerous security attacks (Section 7). We formally model the success rate of this attack, with the success rates of the component SNS API inference attack algorithms as model parameters. We then feed our empirical data into the model to obtain the success rate of the identity theft attack. This modelling exercise offers an innovative means for demonstrating the threat of SNS API inference attacks.

In this work, we mainly focus on Facebook, the most popular SNS. However, our observations regarding Facebook third-party applications also apply to other extensible SNSs (e.g., OpenSocial).

2. SNS API INFERENCE ATTACKS

In a typical SNS, every user owns a profile consisting of attributes such as birthdate, education information, etc. Permission-based authorization schemes allow users to restrict access to their attributes. Therefore, a third-party application that needs to access a user's attributes needs to request the user for granting the required permissions.

A user can also permit an application to access certain information that she does not own. Specifically, given that the required conditions are satisfied, a user can allow an application to access her friend's photo albums. In such cases, the user who runs the application assumes the role of a ***proxy*** through whom the application accesses the friend's profile. An application can access the friend's information if the following conditions are met: (i) the proxy user grants indirect access to the application; (ii) the friend grants permissions to the proxy user for accessing the requested information; (iii) the friend explicitly allows her information to be accessible by applications through proxy users.

Suppose a user u subscribes to a third-party application π in an SNS. As a result, in her interactions with π, the SNS makes some information accessible to π (aka ***accessible information***), given that the required permissions are granted. Such accessible information may contain information about users other than u (e.g., u's friends). Moreover, there might be some personal information about u (aka ***inaccessible information***) that she intends not to share with π. Such an intention is sometimes declared explicitly in her privacy settings, but sometimes implicitly willed by her. π may also have some ***background information*** regarding u. Now a successful SNS API inference attack launched by π against u is defined as follows: *π infers some inaccessible information about u from its background information as well the accessible information it could obtain in its interaction with u from the SNS API. The inferred information must not be inferable solely from π's background information.*

As we want to focus on the role of third-party extensions in launching inference attacks, we give the above definition to emphasize that the user's accessible information through the SNS API must be utilized for inferring information. Note also that π complies with the protection mechanisms of the target SNS, by accessing nothing but the legitimately accessible information. Yet, the protection mechanisms fail to prevent inference of inaccessible information.

The definition above does not cover inference attacks that infer information about friends of the user who runs the application. Our goal is to show that, even by considering only such limited attacks, the damages can still be significant.

3. EMPIRICAL STUDY: DESIGN

This section reports the design of an empirical study we conducted to assess the feasibility of SNS API inference attacks. For the purpose of this study, we identified eight sample inference tasks and devised an inference algorithm for each task. Each inference algorithm takes as input some accessible information, carries some background information that it employs for inference, and infers some targeted in-

accessible information. As mentioned previously, inference attacks can be employed as building blocks for other attacks such as identity theft. We therefore evaluated the success rate of these inference algorithms with this application in mind. Specifically, an attacker is usually allowed to make some α attempts for each security question, before the authentication mechanism blocks off the attacker (where $\alpha > 1$). Therefore, each of our algorithms will return α answers, corresponding to the number of attempts the attacker is allowed to make. We set $\alpha = 4$ in our study.

A third-party Facebook application (coded in JavaScript) was developed as an environment for simulation and data collection. Embedded in the application are our inference algorithms that were executed on the participants' profiles. To achieve 95% confidence level and 5% margin of error, we recruited 424 participants (according to [4], 384 participants are required) from 150 universities across North America.

An execution of this application involves the following steps. First, the participant selects a subset of the eight sample inference algorithms to be included in the simulation. Then, the application asks the participant to grant the permissions needed for setting up the accessible information of the selected inference algorithms. The application simulates the accessing of the accessible information, and the running of the inference algorithms. Each algorithm infers up to α answers for each of the inference tasks. Next, for each selected inference task, if the corresponding inference algorithm is able to infer at least one answer, then the answers is presented to the participant, in the order of confidence of the algorithm. The participants are to confirm which answer is the right answer, or to declare that none of the answers is correct. In summary, the execution of an inference algorithm could lead to three possible outcomes: 1) no answer is returned, 2) none of the returned answers is correct, or 3) one of the returned answers is correct. The first two outcomes simply mean algorithm failure.

4. ANALYTICAL FRAMEWORK

In this section, we describe our analytical framework based on which we analyze the behaviour of our inference algorithms. When we assess the success rate of inference algorithms, there is more than one reason for an algorithm to fail. For example, sometimes the data needed by an algorithm to make inference is not available, and at other times the data is available but the algorithm cannot gain access to that data. For each inference algorithm, we classify user profiles based on the level of availability of the information that the algorithm needs to extract from the profile in order to make correct inference. In essence, this classification corresponds to the different reasons for inference failure. For each algorithm, we will define a different success rate when each of the failure conditions is ruled out.

In Facebook, there are two types of access permissions involved in user-application interactions:

1. ***Type-1 permissions***: These are permissions that are granted or denied by the user who is running the application. They include the permissions to access the user's attributes, as well as the permissions to access the attributes of the user's friends.

2. ***Type-2 permissions***: These are permissions that are granted or denied by the friends of the user who is running the application. They include the permissions that a friend grants to the user running the application, as well as the permissions that friends can specify regarding which attributes can be accessed by applications through proxy users[1].

We specify below a scheme for classifying user profiles. Corresponding to each class of user profiles is a predicate.

Definition 1. *A user profile j is* **type-1 accessible** *to algorithm i ($accessible1(i,j) = 1$) iff the type-1 permissions requested by algorithm i have been granted by user j.*

Definition 2. *A user profile j is* **type-2 accessible** *to algorithm i ($accessible2(i,j) = 1$) iff $accessible1(i,j) = 1$ and the type-2 permissions required by algorithm i are granted by at least one of the friends of user j.*

For example, suppose algorithm i, executing on user j's profile, needs to access user j's friends' photo albums. User j's profile is type-2 accessible for algorithm i if photo albums of at least one of her friends is accessible to algorithm i.

Definition 3. *A user profile j is* **available** *to algorithm i ($available(i,j) = 1$) iff $accessible2(i,j) = 1$ and algorithm i could at least return one answer when it is executed on user profile j. In other words, the required data for making a guess is both accessible and available.*

Definition 4. *A user profile j is* **applicable** *for algorithm i ($applicable(i,j) = 1$) iff $available(i,j) = 1$ and the information algorithm i attempts to infer is logically defined for user j. That is, it is not a logical impossibility for algorithm i to infer the target information from the user j's profile.*

For instance, if an individual is single, then it is a logical impossibility for an inference algorithm to infer his/her spouse's name. So this individual's profile is not applicable for this specific algorithm.

We define below a predicate pertinent to the success scoring of an inference algorithm.

- $succeed(i,j,\alpha)$: This binary predicate evaluates to 1 iff, when algorithm i is executed on user profile j, the correct answer of the inference task is among the first α candidate answers returned by algorithm i.

We propose here four different success rates for evaluating the effectiveness of an inference algorithm in targeting the different classes of user profiles. Table 1 shows the formulas for the success rates of algorithm i. Here, N is the total number of user profiles on which algorithm i was executed. For every profile class and for every α, we define a different success rate. For example, $P^i_{ava}(\alpha)$ is the ratio of available profiles for which algorithm i successfully returned the right answer in at most α attempts. Note that the four success rates are totally ordered: i.e. $P^i_{acc1}(\alpha) \leq P^i_{acc2}(\alpha) \leq P^i_{ava}(\alpha) \leq P^i_{app}(\alpha)$. Note also that the success rates in Table 1 are conditional probabilities.

5. INFERENCE ALGORITHMS

This section presents a brief description of the inference strategy in the eight sample inference algorithms. In all of

[1]Recall from Section 2 that when an application accesses a friend attribute, both type-1 and type-2 permissions are needed.

Success rate	Formula
Type-1 Accessibility	$P^i_{acc1}(\alpha) = \frac{\sum_{j=1}^{N} succeed(i,j,\alpha)}{\sum_{j=1}^{N} accessible1(i,j)}$
Type-2 Accessibility	$P^i_{acc2}(\alpha) = \frac{\sum_{j=1}^{N} succeed(i,j,\alpha)}{\sum_{j=1}^{N} accessible2(i,j)}$
Availability	$P^i_{ava}(\alpha) = \frac{\sum_{j=1}^{N} succeed(i,j,\alpha)}{\sum_{j=1}^{N} available(i,j)}$
Applicability	$P^i_{app}(\alpha) = \frac{\sum_{j=1}^{N} succeed(i,j,\alpha)}{\sum_{j=1}^{N} applicable(i,j)}$

Table 1: Success rate formulas for algorithm i

our algorithms, the victim is the participant: i.e., the user who runs the application. If a participant does not grant the type-1 permissions required for executing the selected inference algorithms, no data point is collected for this participant. In other words, all the data points we collected correspond to type-1 accessible profiles.

We devised inference algorithms that use different types of information as the basis of inference, and resemble real security questions. Our inference algorithms are described in the following. Note that in the description of the algorithms, the basis of inference that is accessed in the victim's profile for inferring her inaccessible information is underlined.

1. Birthday (`birthday`): A day when the participant receives a considerable number of birthday greetings on her wall is inferred as her birthday.
2. Partner's First Name (`partner`): A user who has the highest number of appearances in the participant's photo albums (specially albums with captions containing "marriage", "wedding", etc.), and is of the opposite gender, is inferred to be the partner of the participant. By partner we mean spouse, boy friend or girl friend.
3. Favorite Author (`author`): An author who authors the majority of the books on the participant's list of favorite books is inferred to be her favorite author.
4. Favorite Movie Genre (`genre`): A genre that accounts for the majority of the movies on the participant's list of favorite movies is inferred to be her favorite movie genre.
5. Youngest Sibling's First Name (`sibling`): Access the participant's friends' family information. The youngest friend who has listed the participant in her family information as a sibling is inferred to be the youngest sibling. If no friend has identified the participant to be a sibling, the youngest friend in the participant's friends list who shares the same family name as the participant is inferred to be her youngest sibling. In both cases above, if the friends' birthdays are not accessible, break tie randomly.
6. Hometown (`hometown`): The participant's hometown is inferred to be in the same town as her high school is. If the participant does not list her high school in her education information, then her college or university are used for inference.
7. Oldest Friend (`oldestF`): Access the participant's and her friends' education information. A friend who went to the same high school as the participant did and, is in around the same age as the participant is inferred to be her oldest friend.
8. Political View (`polView`): Access the participant's family information. The prominent political view of the participant's close relatives (e.g., spouse) is inferred to be the political view of the participant. If `polView` could not access the political view of the participant's relatives, then the political views of all friends are consulted.

We do not claim that the above algorithms are particularly sophisticated, but rather we aim at showing even such simple algorithms can yield unintentional information disclosure.

6. ANALYSIS

In this section, we apply the analytical framework of Section 4 to the data points we collected from our participants.

6.1 Classification of User Profiles

Figure 1 depicts the distribution of the various classes of profiles for each algorithm. All profiles are type-1 accessible for all algorithms, i.e., if participant j does not grant the type-1 permissions requested by algorithm i, then algorithm i will not be executed on participant j's profile.

Only three algorithms require type-2 permissions: `sibling`, `polView` and `oldestF`. Thus, all profiles are vacuously type-2 accessible for the rest of the algorithms. The `sibling` algorithm has a secondary inference rule that serves as a fall back when the required type-2 permissions are not in place. That is why to `sibling` all profiles are type-2 accessible. In the case of `oldestF`, the required type-2 permissions happen to be always granted in our empirical data due to the Facebook default privacy settings. A similar reason is behind the low number of type-2 accessible profiles for `polView`.

The higher number there are available profiles, the easier the algorithm can find an answer. For example, only 34.5% of user profiles were available for `author`, whereas the figure is more than doubled for `hometown`. It means the chance that a user has at least one school added to her profile is much higher than the chance that she has at least one entry in her list of favorite books. As a result, the number of potential victims of `hometown` is larger than that of `author`.

Applicability is a more realistic metric for evaluating vulnerability of a profile to inference algorithms. For instance, despite `partner` returns at least one answer for 73.2% of the participants, only 30.2% of the profiles are applicable, i.e., owners of the 69.8% of profiles are single and do not have a significant other. Hence, it is impossible for `partner` to find the right answer in such cases. Analyzing this class of profiles gives us valuable insights on how many users are potentially vulnerable to an inference attack. Figure 1 illustrates that except for `polView`, all the other algorithms have quite a high number of applicable profiles.

6.2 Success Rates of Inference Algorithms

6.2.1 Success rate

Figure 2 depicts the success rates of the inference algorithms as computed using the formulas in Table 1 when $\alpha = 1$. The result shows surprisingly high success rates for some of our algorithms, even though their designs are relatively straightforward. For example, the success rate of `birthday` is 77.4% even for type-1 accessible user profiles. This means, for 77.4% of the users who grant type-1 permissions to `birthday` (i.e., permission to access wall posts), their birthdays can be successfully inferred. In addition, if

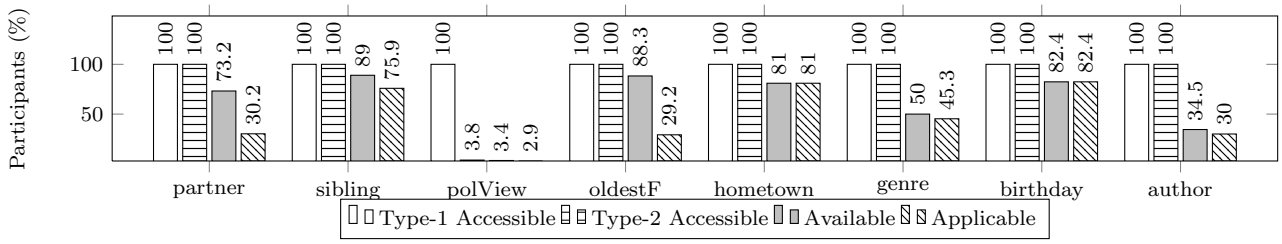

Figure 1: Classification of participants' profiles

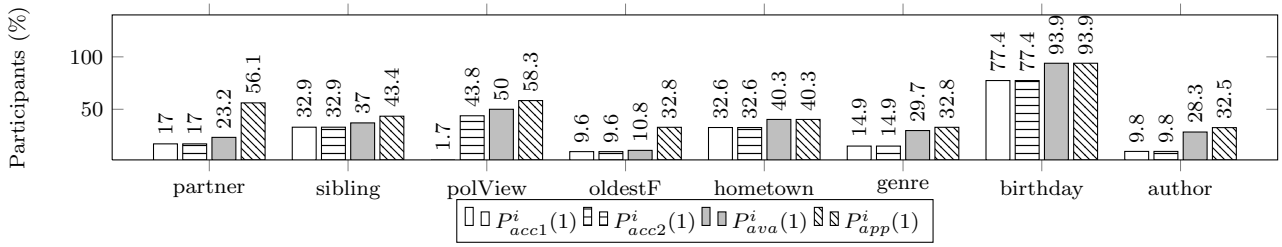

Figure 2: Success rates of inference algorithms when $\alpha = 1$

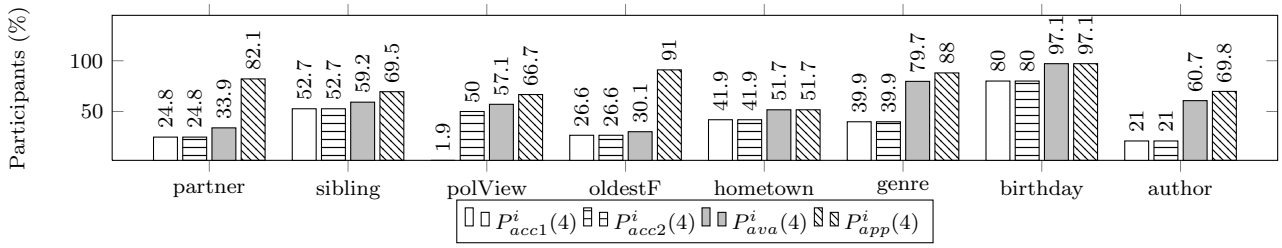

Figure 3: Success rates of inference algorithms when $\alpha = 4$

there is at least one birthday greeting on a user profile, then the success probability ($P^i_{ava}(1)$) is even higher, at 93.9%.

The `polView` had the lowest number of available profiles (3.4%), but $P^i_{ava}(1)$ is a moderate 0.5. This implies, although it is unlikely that an application can access and find the required information for inferring a user's political view, if such an access could be gained and the data is present, then the probability of success is 0.5. This moderately high success rate compared to the small number of available profiles for `polView` shows that the inference strategy behind this algorithm is a reasonable one, but the conditions for the availability of data is rarely satisfied.

6.2.2 Unexpected disclosure

Prior to the experiment, there were a number of algorithms that we expected a low success rate. One of them was `partner` because its inference strategy did not appear for us to be universally applicable. However, the experiment shows that if a user has a partner, algorithm `partner` can identify his/her partner with a non-trivial probability ($P^i_{app}(1)$) of 56.1%.

The difference between $P^i_{ava}(1)$ and $P^i_{app}(1)$ in `partner` is significant (23.2% versus 56.1%) due to the large difference between the number of its available and applicable profiles. The algorithm `partner` has a large number of available profiles because finding at least one photo containing a tagged user of the opposite gender is not difficult. But many of our participants were single. As a result, the number of applicable profiles is much lower. The high applicability success rate of `partner` confirms the effectiveness of its simple inference strategy, contrary to our original expectation.

Although information about one's partner is sensitive on its own, revealing an individual's ex-partner is an even more dangerous kind of privacy violation. People typically hide information about their former relationships. Based on the feedbacks we received from the participants, in at least 7% of cases when all inferred answers by `partner` were wrong, the participant's ex-partner (e.g., ex-wife) was identified instead. More importantly, a few participants told us that algorithm `partner` has identified a person whom they are looking forward to date. Identifying such information through other information sources is not straightforward. In other words, inference attacks through SNS extension API could result in accessing highly sensitive information that are not easy to achieve through other adversarial techniques.

6.2.3 Multiple attempts

By increasing the value of α from one to four, some algorithms showed much higher success rates (Figure 3). For example, $P^i_{app}(4)$ for `genre` increases to 88% (compared with $P^i_{app}(1) = 32.8\%$). On the contrary, success rate of some algorithms like `birthday` and `hometown` do not change significantly: i.e., they are likely to either return the right answer in their first attempt or fail.

7. INFERENCE ATTACKS AS BUILDING BLOCKS: IDENTITY THEFT

Inference attacks are not only privacy violations, they are building blocks for launching other security attacks. For instance, an individual who wants to register for an online banking access account is usually asked to select an alternative authentication mechanism. One of such mechanisms is security questions. The idea is that, a user first configures k supposedly private questions out of a set of N questions prepared by the service provider, together with their answers, and then when it is necessary, she is challenged to answer those k security questions (e.g., *"What is the first name of your youngest sibling?"*). Inference attacks can be employed by adversaries to find the answers to such security questions. An adversary who knows an individual's username, claims that she forgot her password. The pre-configured security questions will now be presented to the adversary. Then, the adversary launches inference attacks against the victim to find the answers to these security questions. As SNS users disclose a vast amount of private information in their profiles, SNS API inference attacks launched by third-party applications could yield an alarming success rate in identity theft attacks, as we shall see in the following.

In the following, we estimate the probability of success for an identity theft attack when the above authentication strategy is used by the service provider. We assume that the type-1 permissions required for launching inference attacks are indeed granted. Let Z be the set of questions supported by the service provider. Assume every subset of k questions is equally likely to be adopted by a user. The probability that an adversary can answer all k questions is the following:

$$\pi_k(Z) = \sum_{S \in [Z]^k} \frac{\prod_{i \in S} P^i_{acc1}(4)}{\binom{N}{k}} \quad (1)$$

where $[Z]^k$ is the set of all size-k subsets of Z, and $N = |Z|$. We use $P^i_{acc1}(4)$ in (1) because a user is typically allowed to make more than one attempt to answer his security questions. Note that we assume the successful execution of an algorithm is independent from the other algorithms.

If Z contains exactly the 8 inference algorithms that we designed (i.e., $N = 8$) and $k = 3$, then the value of $\pi_k(Z)$ equals 0.04, i.e. for 4% of the potential victims, all three selected security questions can be answered correctly. The impression that this success probability is low and thus the expected number of victims is low is deceptive. One should be reminded that there are applications in Facebook with around 50M, 28M, or 15M monthly active users. With a success probability of 4%, an application with 15M users means an adversary can successfully impersonate 600,000 users by first launching inference attacks, and then following up with identity theft attacks. This analysis testifies to the unfortunate effectiveness of SNS API inference attacks as a stepping stone for dangerous security attacks.

8. RELATED WORK

Inference attacks in social networks is a new research problem. [6, 5, 3] proposed inference techniques that mainly use two types of information as the basis of inference: (1) friendship information, and (2) group membership. The main idea is that the value of an attribute in an individual's profile is likely to be the same as its value in the majority of her friends' profiles and/or her fellow group members' profiles. Such works assume they have full access to the social network data set (i.e., profiles), and then they suppose half of the profiles are public and the other half are private. However, they do not specify a realistic mechanism (crawling, SNS API, etc.) through which certain profiles come to become visible (or private) to the adversary. On the contrary, we make the more realistic assumption that an application only has access to a profile as well as the information that is accessible via that profile.

Inference attacks can use different inference channels and target different types of users. In [1], inference attacks are classified along two dimensions: (1) inference channel, and (2) victim. Knowing all possible inference channels and all potential victims of inference attacks is of great use to researchers for proposing the required protection mechanisms.

Felt and Evans [2] proposed the first work on protecting SNS users against threats specifically posed by third-party applications. But, they did not relate their work to inference attacks. In other words, they present a protection mechanism, called privacy-by-proxy, to prevent third-party applications from accessing original information in user profiles.

9. CONCLUSION

In this work, we took the first step to understand the feasibility of SNS API inference attacks, as well as to assess their privacy impacts. Future work includes the redesign of SNS APIs that would mitigate the threat of inference attacks.

10. ACKNOWLEDGMENTS

This work is funded in part by a Google Research Award, stipends from ISSNet – an NSERC Strategic Research Network, and an NSERC Discover Grant.

11. REFERENCES

[1] S. Ahmadinejad, M. Anwar, and P. Fong. Inference attacks by third-party extensions to social network systems. In *Proc. of IEEE 9th International Conference on Pervasive Computing and Communications Workshops*, pages 282 –287, 2011.

[2] A. Felt and D. Evans. Privacy protection for social networking APIs. *Web 2.0 Security and Privacy*, 2008.

[3] J. He, W. Chu, and Z. Liu. Inferring Privacy Information from Social Networks. In *Intelligence and Security Informatics*, volume 3975 of *Lecture Notes in Computer Science*, pages 154–165. 2006.

[4] J. Kotrlik and C. Higgins. Organizational research: Determining appropriate sample size in survey research appropriate sample size in survey research. *Information Technology, Learning, and Performance Journal*, 19(1):43, 2001.

[5] W. Xu, X. Zhou, and L. Li. Inferring privacy information via social relations. In *Proc. of IEEE 24th International Conference on Data Engineering Workshop*, pages 525–530, 2008.

[6] E. Zheleva and L. Getoor. To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles. In *Proc. of the 18th international conference on World wide web*, pages 531–540, 2009.