A Survey of Security and Privacy in Online Social Networks

Ed Novak Qun Li
Department of Computer Science
The College of William and Mary

Abstract

This survey highlights the major issues concerning privacy and security in online social networks. Firstly, we discuss research that aims to protect user data from the various attack vantage points including other users, advertisers, third party application developers, and the online social network provider itself. Next we cover social network inference of user attributes, locating hubs, and link prediction. Because online social networks are so saturated with sensitive information, network inference plays a major privacy role. As a response to the issues brought forth by client-server architectures, distributed social networks are discussed. We then cover the challenges that providers face in maintaining the proper operation of an online social network including minimizing spam messages, and reducing the number of sybil accounts. Finally, we present research in anonymizing social network data. This area is of particular interest in order to continue research in this field both in academia and in industry.

1 Introduction

A social network is a structure made up of actors such as individuals or organizations, and ties between these actors such as interactions, relationships, and connections. Formally, in the literature, this is almost always represented as a graph which we refer to as the "social graph." The nodes of such a graph represents an actor and the edges represent ties between those actors. An online social network is different from a social network; however, the two are often used interchangeably. An online social network is a computer software and hardware system that attempts to model the social networks found naturally in the world. An online social network has a representation of a user (usually a profile) and his or her social links, although other services are often incorporated. Online social networks are usually web based and they almost never perfectly match the underlying social network they are trying to model. Most online social networks today follow the client server architecture that is common on the web. However, distributed social networks are well studied in the literature. In this paper the term "online social network" (OSN) is used to refer to such a system of computer hardware and software. The term "social network" (SN) is used to refer that which is being modeled or approximated by the online social network. Either one can be abstracted using a "social graph."

In any online social network there is a wealth of information about its users embedded in the social graph. This is the focus of the majority of this paper. Specifically, there are two types of information: explicit and implicit. Explicit information is information that is stated by the user on purpose. An example of this might be the birthday that appears on a user's profile page. Explicit information is not necessarily accurate. There is also implicit information. This is information that can be inferred about a user or a community based on explicit information. An example of this is recognizing that a user is connected to many other users that have all stated they are interested in muscle cars on their profiles. It is therefore implied that this user is also interested in muscle cars. Implicit information is also not always accurate. In fact, it is upper bounded in accuracy by the explicit information on which it relies. Interestingly, implicit information is usually quite close to this bound

Much of the information that is typically published by users in an online social network is particularly sensitive. It is because of this sensitive information, both implicit and explicit, that privacy and security concerns are raised. All parties involved face a paradox. More information is necessary to make the OSN thrive. Users, however, maintain their privacy by not publishing personal information about themselves. This problem can be solved by protecting user data and is discussed extensively in section 2. This section covers various solutions for protecting user data from attackers at various vantage points including users with direct access, indirect access, advertising agencies, and the OSN providers themselves.

All of these sections address protection of user data but for an OSN to thrive the proper operation of said social network must also be maintained. In section 5 various threats to the operation of OSNs are discussed. For example, spreading spam messages through an OSN is very common. Additionally, there is a notion of trust in an online social network between users. This trust can be used to verify content and other users within an online social network. This idea can be extended to prevent the viral spread of spam through the online social network. Because many OSNs are web based, general web 2.0 security is a concern. Attackers can also create fake "sybil" accounts that they can utilize to influence the outcome of elections in an OSN.

In order to advance the state of research in online social networks OSN providers need to publish their OSN data. However, this is a task not to be taken lightly. Clearly, this data is high sensitive. Techniques for de-anonymizing social graph data are discussed. These are used to motivate the sophisticated anonymizing schemes discussed at the end.

2 Protecting User Data

In any online social network users are generating a massive amount of data. In this section we focus on data the users create on purpose with the intent of sharing (i.e., explicit information). This includes blog or micro-blog posts, profile information, photos, videos, instant message text, and so on. This does *not* include data such as sentimental relationships with other users, or account creation times. We refer to this type of explicit, "for sharing" data as "content." There are several different vantage points from which an adversary can access user data in a modern online social network.

Other users are a constant threat because in the vast majority of OSNs today anybody (including malicious parties) can sign up for an account and become a member thereby increasing their access to other users' data. The only requirements are a valid email address and the ability to solve a re-captcha. This means the social network provider must provide the users some way to specify who they trust and who they don't trust within the social network. Different providers handle this in different ways with varying levels of detail.

Social network "applications" are web pages that are written by some third party but have API access to social graph data typically only accessible by the online social network provider. Applications are typically opt-in for users. In the case of Facebook users are told explicitly and vaguely what content an application will attempt to access and are given the choice whether or not to use that application.

Advertising agencies provide the vast majority of revenue for online social network providers. They are motivated to, in conjunction with the OSN provider, harvest user data in order to target advertisements at a granularity never before thought possible. Many users do not appreciate having personal information analyzed in order to better target ads. Additionally, it is risky to allow such complete profiles of real world people to be compiled and stored by third party corporations that have little vested interest in keeping this data safe.

The social network provider itself is a threat, because the social network provider has technical access to all the data on the OSN, users must trust that provider and trust in the provider's privacy policy and terms of use. Furthermore, the OSN provider has little motivation for keeping their users' data safe.

As soon as users are publishing content in an online social network, we must be careful about who has access to this content. Fortunately, practical levels of access can clearly be divided into four categories: other users (in section 2.1), third party online social network applications (in section 2.2), third party advertising agencies (in section 2.3), and the online social network provider (in section 2.4). These subsections outline some of the significant research in protecting user data from each of these parties.

2.1 Protection From Other Users

Protecting users from "other users" includes any other user on the social network. We can divide the set of other users into three categories.

Directly Connected Users These are users that have a link between them in the social graph. This means something different in different social networks. In Facebook it means that the two users can view more information on each other's profiles. In some of the literature this simply means the two users have communicated with each other via email.

Unconnected or Indirectly Connected Users These users are two or more hops away from one another. (e.g., friends of friends (FOF) or friends of friends of friends (FOFOF)). This category also includes two users that are in the online social network that have absolutely no relation between them.

General Public The general public has access to information in many online social networks. For example, Twitter makes tweets public by default and Google indexes them.

2.1.1 Direct Access

Many online social network providers allow the users of their social network to make privacy settings. This is the user's first line of defense against malicious users. Some of these privacy setting schemes are simple and straightforward. For example, Twitter allows users to make tweets “private” which are only visible to their followers. This is the only privacy setting they provide and it effectively locks out the *general public* and *unconnected users*. However, most online social network providers like Facebook and LinkedIN subject their users to a dizzying, complex set of privacy controls. The repercussions of the decisions users are forced to make immediately are not fully known until the user is familiar with the online social network in question. These privacy settings must be carefully weighed and experimented with, and yet users are forced to make privacy settings immediately upon joining the social network in order for them to be successful in protecting their own data.

We begin by looking at a paper that analyzes the impact of simply choosing the most restrictive or most private settings possible. In fact, even if users can fully understand their privacy settings they may not be protected from other users in an online social network. In [82], Stutzman and Kramer-Duffield explore the implications of making a profile “friends only” on the Facebook online social network. Only *directly connected users* can view any content the user has posted. The researchers conclude that while such a configuration is effective against other users on the online social network, very few users actually utilize it. However, this researcher points out that this is not a strong defense against a myriad of other attacks from different perspectives including, sybil accounts, advertisers, applications, and the online social network provider itself. Additionally, it stifles prominent features of the online social network like friend discovery.

Rather than simply lock down one’s own profile, several papers have been written that try to make existing privacy settings easier to understand for the user. In [48], the authors attempt to help users better understand their privacy settings by showing them examples of what their own profile would look like from the perspective of other users. In many social networks determining what data is visible and what data is not visible and to whom is a complex task for users. Here lipford et al. implement a system for Facebook that allows users to view their own profile from various perspectives. Because this system is implemented in Facebook users can view their own profile from the perspective of a friend (*directly connected user*), a friend of a friend or a Facebook user (*unconnected or indirectly connected users*), and a non-Facebook user (*general public*). This paper does help users understand the privacy settings that they’ve made but it does not help protect them from various more subtle privacy issues such as neighborhood attacks, and network inference techniques. However, these techniques are out of the scope of this section and are covered in better detail in the OSN de-anonymization section. This paper forces the user to properly sort their friends into the categories described above.

In an attempt to improve upon the situation, we have Wizards, [23], in which the authors design a “privacy wizard” which strives to make it easier for users to specify their privacy intentions. The key observation is that users imply most of their privacy settings and mentally categorize their connections to other users in the online social network as “communities.” For example, users think of groups of relations such as co-workers, friends, classmates, etc. These categories are not bound by the simplistic set of 3 specified above. Rather, the user can create an arbitrary number of them in order to allow proper access to the correct OSN users. In [23], a machine learning technique is used to classify users in the online social network. When a new user initiates the system they are asked to make custom privacy settings for each user as new connections are made. The system uses the manual classifications to make broader categories. It attempts to select the most informative users for manual classification order to minimize the number of users that must be classified manually. The authors leverage the fact that users’ privacy settings for another user usually coordinate with the communities they’re associated with. They assume that the communities will be implied by the online social network data.

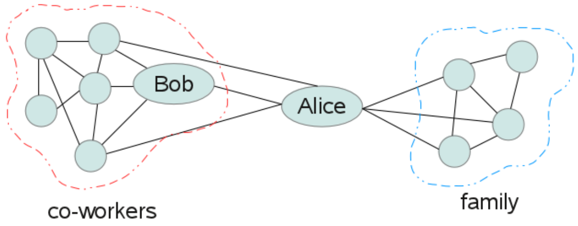


Fig. 1: Alice is a member of two different communities: co-workers and family. After categorizing Bob manually, the system will categorize the rest of the co-workers together because they are strongly connected.

To better understand this concept imagine Alice and, her friend in the OSN, Bob. When Bob tries to access Alice’s profile information there must be some set of rules governing how much of this content he can view. Before Bob has visited Alice’s page Alice is asked to classify Bob explicitly. Alice allows Bob to view everything on her profile except her date of birth (because she is aware of the work in [3]). What has happened here is that Alice has placed Bob into a category (for example Bob might be a co-worker). Alice did NOT tell the system that Bob is her co-worker. She only expressed that Bob should have access to everything but her date of birth. Now imagine Bob is friends with several more people at work. Additionally, most of the people that work at that office are connected in the online social network. We can recognize that Bob is part of a community that Alice is mentally formulating based on who he is friends with and other social network data. We know what Alice wants Bob to access in her profile so we can assume that Alice probably feels similarly about the rest of that community. This, the core-idea of [23] is illustrated in Fig. 1 and in general it works very well except for the occasional outlier.

This is very clever and well designed system for determining a user's privacy settings with minimal effort by the user. However, users are occasionally mis-categorized. Additionally, this design does not protect users from the ever watchful eye of the OSN provider or ad agencies. Instead it allows them to achieve fine grained, explicit privacy settings immediately without having to learn a complex privacy scheme. Indeed, they need only classify a small percentage of their OSN network to achieve powerful results.

In PCO [70], Rahman et al. develop an architecture for privacy protection in online social networks. This is fundamentally different from [82] because it is the implementation of an entirely new privacy architecture rather than analyzing an existing one such as [82] in which we explore the maximum amount of privacy available in the Facebook online social network. The work in PCO, [70], focuses on the idea of exposing different levels of details to different groups of people and at different times of the day automatically.

For example, the user might be watching a movie on TV. When a co-worker asks the social network what the user is doing it might expose to them that the user is relaxing, to his friends it might expose that he is watching TV, and to his parents it might expose that he is watching a movie and perhaps even give the title of the movie! The system, referred to as "PCO" which stands for Privacy-sensitive Context Obfuscation, asks the user to define different granularities. Activities are defined in each granularity (e.g., watching "Monty Python", watching TV, relaxing) and then various attributes of the requester and the user in question determine which granularity is exposed. (e.g., the time the requester is accessing the data, the group the requester is in, the type of activity the requester is accessing, etc).

This system is well designed and is closely related to the google+ circles design which also allows different groups of people different levels of access based on their group membership. However, this is an architecture for privacy and cannot be incrementally deployed. Also, the social network provider has little incentive to use this architecture. Finally, this architecture still requires the user to set up elaborate privacy settings when they first join as well as segregating their friends into various groups, and defining different levels of obfuscation for various activities. All of this and the data is still stored on the social network. This work does little to protect the user from the social network provider itself. However, a system of this design could easily be modified to apply to protecting data from a social network application.

It appears to be that the best solution for users to protect themselves from other users in an online social network is for the online social networks providers to allow the users to classify other users into categories and then provide different access levels for these different categories. In fact, this is similar to how personal information is shared in real social networks.

2.1.2 Indirect Access

One of the more subtle issues in protecting user data from other users is the spread of sensitive, partially private content. The key difference between this and the previous section is malicious users accessing other user content directly vs. indirectly. A malicious user accesses information indirectly when some third party user spreads that information.

Social networks typically try to define some set of rules for the user to define who can view their information and who cannot. Anybody, however, is allowed to publish information. The problem with this is that users that have access to the sensitive, hidden data of another user can simply use their ability to publish to spread that data to users whom are not supposed to have access to it. This can be thought of intuitively as telling a secret. One user knows the secret and may tell select other users that secret. However, any one of the secondary users have the ability to spread the secret further, possibly to people that were never intended to hear the message in the first place. This topic is referred to in the rest of this paper as "leaking." Many social networks accidentally encourage this behavior by providing a built-in mechanism for propagating content. A good example of this is Twitter's re-tweet functionality. Conversely, social networks that attempt to prohibit users from spreading protected content in this way are only fooling themselves. Any user that is allowed to read and allowed to publish will have the ability to spread sensitive information. At the very least, they can read the content from one browser window and re-type it into a second.

In RT @IWantPrivacy [58], the authors address this problem head on. They collect a massive data set from the Twitter social network API that includes over 118 million re-tweets. 4.42 million (3.7%) of these re-tweets are tweets that leak otherwise protected content. In this study "protected" simply means the tweet came from a user that only allows followers to view their tweets. These protected tweets are leaked because the set of followers re-tweet the content thus making the original content publicly available. This paper shows that leaking is a prevalent issue in online social networking.

In [56], Huina Mao et al. study this problem in greater detail. They aim to accomplish three things: recognize what sort of content is dangerous to leak (e.g., divulging medical information), build an automatic classifier for tweets (with categories "dangerous to leak", and "not dangerous to leak"), and characterize who leaks information and how they leak that information, in greater detail than [58]. They determine that there are numerous threats to user privacy latent in the leaking problem. Of their 162,597,208 tweet data set they find 575,689 tweets that divulge vacation information, 21,297 that divulge drinking habits (contain the word "drunk"), and 149,636 that divulge medical information. This paper also explores the idea of information leakage due to the conversational nature of some tweets. In other words users may implicate others by talking amongst themselves about one another on twitter. Conversations, on twitter, are held entirely through re-tweeting.

In [33] the authors take a more broad view of the issue at the level of social networks in general (as opposed to just Twitter). They open by asking if it is possible to minimize vulnerability to information leakage (maximizing privacy) while still being an active member of the social network. Many sites do not provide adequate privacy settings so these researchers instead focus on minimizing friends that jeopardize a user's own privacy. More specifically, they claim that certain friends increase a user's vulnerability more than others. These friends are identified as those that have insufficient privacy settings themselves. Friends should have privacy settings that protect the entire group of friends (community).

This paper assumes that users will not leak sensitive information about one another intentionally but rather that attackers can learn information about one user from another user's content. In this way the information leakage problem is different in this paper than in [56] and [58]. However, the idea that users can spread sensitive content is the same. The difference here is whether or not they do it intentionally. This paper makes a strong assumption that users that are linked in the social network somehow effect the privacy of each other. In reality, this is probably true but the actual result is subtle.

In [54], Kathy Macropol et al. are actually able to predict the flow of information through a social network (leaked, sensitive information or otherwise). There technique is able to determine how many statements are made back and forth between a set or users based on the content of the conversation. The majority of this work is discussed in the link prediction section.

The ML-Model [55] describes social networks in a grander sense. They rightfully recognize that many people have multiple accounts on many different social networks. It is not uncommon for content to flow between social networks through these users, quite the opposite in fact. This paper formalizes such a model of online social networks. Intuitively there are pillars that represent users. Each pillar has several cuts at regular heights. At each height exists a social network. The relationships at the social network at height h are represented by the edges of a graph that exists at that height. These edges connect the pillars. An example can be see in Fig. 2. In this model it is very clear that information might flow across edges between users in a social network but any user may take content from one social network and introduce it in another social network.

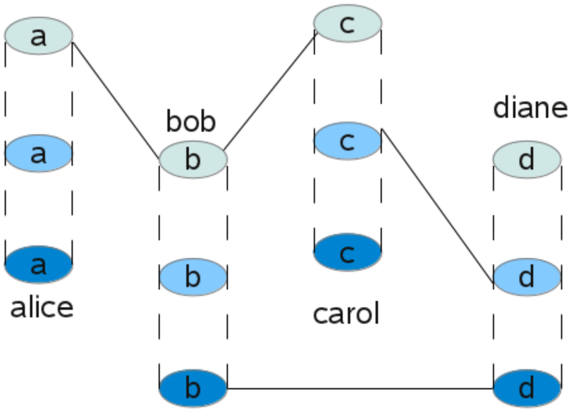


Fig. 2: Example ML-Model Network. Alice can share some content with Bob on the top level social network (e.g., Qzone). This information can then be shared with Diane by Bob using the bottom level social network (e.g., twitter)

This is a much more robust model of online social networking. While this paper does not address information leakage directly it is clear that a model such as this needs to be considered in this vain. Clearly, information can be leaked between social networks and none of the current research addresses this issue. Indeed, many social networks sites make this sort of content spreading streamlined. For example, Stumbleupon makes it easy for users to share content with Facebook friends, Twitter allows for tweets to be automatically posted as Facebook statuses.

The problem of information leakage seems to be fundamental to not just online social networking but generic social networking throughout history. Of course people have always held and spread secrets. In order for social networking to exist users must be able to view and publish content in some capacity. As soon as users attempt to define a subset of other users who can or cannot view their content nothing stops the users in the group from spreading the content to those outside the group. It seems to this researcher that this problem is fundamentally unsolvable without extreme, authoritarian measures.

2.2 Protection From Applications

Online social networks frequently employ some sort of application system. In such a system, third party developers are able to write code that is displayed in the social network and has access to a set of API functions that allow the application being written to access content typically only available to the online social network provider. Applications in all the major online social networks are opt-in however users are not always happy with sharing their personal data with miscellaneous third parties. In the majority of online social networks there is no fine grained control of what personal content the application can access. In other words, applications are opt-in, all or nothing. This subsection reviews some of the methods researchers have attempted in order to improve this situation.

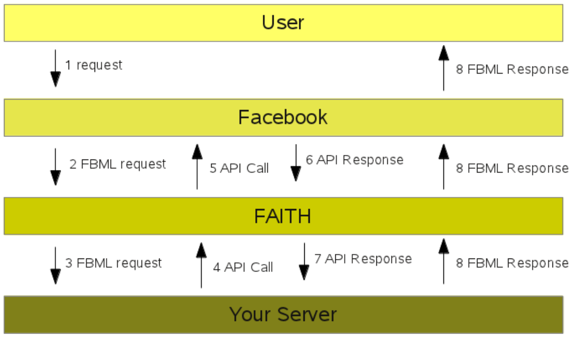


Fig. 3: FAITH Architecture. The FAITH system sits between Facebook and Your Server in order to intercept and verify calls and responses.

In [44], Lee et al. argue that Facebook asks new users to make privacy decisions immediately when their account is created. Obviously it's good that Facebook takes privacy and security concerns seriously but, as we just discussed, new users aren't able to comprehend the full impact of the privacy decisions they're forced to make. This is especially true before they're familiar with the OSN. Additionally, these privacy settings do not have any impact on the ability of third party applications to access the user's data. Lee et al. develop "FAITH" which provides a mechanism for users to expose a different amount of their own data on an application by application basis. They do this by designing a transparent third party server which controls which applications can see which information. Instead of users accessing application data and applications accessing user data directly through the social network, all the API calls are routed through the FAITH server. At this point this server can decide, based on user specifications, what information to expose to this application at this time. A high level diagram of the architecture can be seen in Fig. 3.

The draw-back of this work is that users must opt-in to this service. FAITH has not been widely adopted because third party applications have little or no incentive to use this system. Finally, FAITH still requires users to make elaborate privacy settings to reap the full benefits. This goes against the argument with which they open their paper. This system does provide adequate protection against social network applications when it is used by both the user and the third party app provider.

In [24], Felt and Evans address the same problem as Lee et al. They go on to point out that many applications require less information then they claim they need. Additionally, social graph information (who the user's friends are) is often necessary but rarely mentioned. Felt and Evans design a system called "Privacy By Proxy" which attempts to protect user data from third party social networking applications. In this systems third party applications are given references to user data but not the values themselves. For example, if the application would like to display the birthdays of the friends of the user they would write an html page that contains special tags known only to the social network provider. One tag $< user.friends >$ might reference the set of all friends and another tag $< user.friends[0].birthday >$ might reference a specific friend's birthday. Clearly a more elaborate implementation is straightforward.

It would appear that there is a major security flaw here however. The third party application can simply insert a bit of javascript that reads the content of the page on the client side after it has been loaded and the tags have been replaced and simply send this data back to the application via an XMLHttpRequest. While this hypothesis is untested it would be an interesting research question.

The authors of xBook [78] suggest a slightly different mechanism than the above two. They propose that applications are broken down into modules that have clear input and output. Then, claims can be made about what user data each module has access to, what other websites that module accesses, and where user data is output (if at all). This allows the third party application developer to provide a manifest to the social network provider and user which describes what is done with the user data but does not divulge too many details. It is in the best interest of the application developer to have a well designed application with many modules so that they can simplify their data demands of the user.

In xBook, Singh, Bhola, and Lee describe a horoscope application. This application would require user data such as their birthday. It would also require input from tarrot.com and the current date. If programed correctly the user data can be acquired in one module and the horoscopes can be gathered in another module. This way it is clear that user data is not transmitted to the horoscope site tarot.com. We can now provide more detailed feedback to the user. Rather than simply saying, “this app needs to access your birthday,” we can say to the user, “This application accesses your birthday and tarot.com but it does not transmit your birthday information.” If the application developers do not separate these tasks into separate modules then it is unclear whether or not user data is sent to tarot.com in which case we err on the side of caution and inform the user that it is possible.

This system works very well to protect user information and to provide a finer grain level of privacy choice for the user. It suffers, however, from the same problem that it is opt-in. The online social network provider must choose to implement and enforce such as a system. While satisfying users is a priority, there is no other strong incentive to adopt such a system. In fact, developer time and resource costs both motivate online social network providers to not adopt such a system. However, if they did there would be strong incentive for third party application developers to well modularize their code. It is becoming clear that the only way for privacy concerned users to voice their opinion is to abstain from using online social networks that abuse their role.

2.3 Protection From Advertisers

Because users are publishing a massive amount of data to social networking sites, marketing can be targeted at a granularity never before thought possible. Users can now be fed ads automatically that target their individual interests specifically. Most OSN providers have built their business model on delivering premium, highly focused ad space to advertisers. They determine the specific interests of users based on those users’ own generated content. However, users are not happy with having their data bought and sold at a premium. In this section we discuss works that protect user data from online advertisers.

The first line of defense against advertisements is blocking malicious or adversarial ads. In [75] Sculley et al. attempt to block ads that fit into any of the following categories: counterfeit goods, misleading or inaccurate claims, phishing, arbitrage, inaccurate or deceptive pricing, and malware. Their approach needs to have a low rate of false positives (which would wrongly punishes advertisers), and a low rate of false negatives (which would degrade the user experience). In order to achieve this high accuracy their system evaluates ads (selected randomly) over a large number of models. If the models to not converge to a classification then the ad is subjected to manual inspection.

The models implement a machine learning technique that classifies ads based on several features.

natural language features These are individual terms.

string-based features This feature set tries to target intentional misspellings and typographical manipulation

structural features This refers to the actual layout of the landing page. Where is the ad placed?

landing page-type What type of landing page is this? A blog? A forum? etc.

link-based features What links are there on the landing page. Are there any redirects?

non-text content such as images, videos, or audio.

advertiser account-level features Such as account creation time, and number of ads being run.

policy specific features including a variety of proprietary features that help identify policy violations.

The models are trained individually for different categories. That is, some ads might be considered adversarial on one page and not on another. This makes a lot of intuitive sense in the world of highly focused ad space because we expect ads to sell products related to the content of the page we’re viewing. Adverts that do not follow this simple rule of thumb often are perceived as annoying. Finally, map-reduce is used to train and implement the ad classifier.

If adversarial advertisements can reliably be removed from an online social network than the problem is solved. However, as is typical in any computer security, there is a race between the good and the bad. Defensive techniques do not work well for white hat researchers because it is only a matter of time before malicious researchers discover an exploit or weakness. In [50], Louw, Ganesh, and Venkatakrishnan develop a framework they call “AdJail” for precisely this reason. Because AdJail follows *default deny* behavior, adversaries are forced to make the system behave in the way they want. This fundamentally different from a system that has default allow behavior and the adversary need only make the system malfunction or crash.

In online social networks specifically, ads are placed on the same page as potentially sensitive private user data. Only a simple DOM traversal via basic javascript stands between third party ad agencies and user data. AdJail solves this problem by loading the advertisement in a hidden sandbox window thereby keeping the ad in isolation. From here ad features are analyzed and, depending on a publisher defined policy specification, elements from the ad are forwarded to the page the user is actually viewing. Similarly, user input is forwarded to the ad. In this sandbox, the ad is only exposed to things that the domain (in this case the online social network provider) explicitly exposes to them. This technique is much more robust than [75] because it exemplifies default deny behavior instead of trying to classify all ads. However, [75] places a stronger focus on correctly categorizing ads and as a result, does so much more reliably.

The FCC recently launched a Do-Not-Track initiative on the web. The goal of this initiative is to provide targeted ads to users without revealing user data (e.g. click-streams, user attributes like name and age, or social network connections) to the advertisement agencies. The system places a setting in the browser that can be read by web pages visited by that client. This cookies allows the user to specify that they would like to opt-out of tracking for the purposes of targeted advertising. While the opt-out nature of this system leaves much to be desired (users must opt-out and web sites must choose to implement this feature) this initiative is a step in the right direction. However, researchers Reznichenko et al. point out in [71] that this system does not work in the current, most popular model of ”Pay Per Click” (PPC) advertisements. On a PPC site users visit a page. The page displays an ad from an ad agency. Which ad specifically is chosen by an ad broker. The agencies compete (in auction) for each ad spot. The advertiser pays the site for each click they receive. The broker determines which ads are most likely to be clicked on based on a variety of privacy (and Do-Not-Track) infringing factors. The ads are ranked by the broker based on their click-probability x’s their bid. In this way the advertising agencies are paying the maximum possible.

This paper suggests a system of determining which ads are most probable to be clicked, while still protecting user privacy and the Do-Not-Track initiative by forwarding a collection of the ad agencies bids, broker quality scores, and ads to the client. The client then ranks the ads locally using user information. Because user information is used for ranking but it is not sent to any outside entity this does not violate the Do-Not-Track initiative. The ads are ranked, an ad is chosen for display, and the results are sent back to the broker and ad agency. They also implement this system with ranking done at the broker. In this scenario the client computes a score for each ad, U. These scores are forwarded back to the broker who ranks and selects the ad to be displayed.

This work is very similar to [30] in which researchers Guha, Cheng, and Francis try to strike a middle ground between user privacy and ad targeting with a system they call “Privad”. The difference between these two papers ([71] and [30]) is minimal. Privad implements a dealer. The dealer sits between the client and the broker (party who sells ad space on third party sites). Here the client software supplies information to the dealer whom also receives information from the broker. The dealer decides which ads are displayed to the user based on the attributes of the user and the attributes of the ad. Many many clients submit their information to the broker anonymously who then creates bins. Ads are placed in corresponding bins. Then when clients request ads, they are simply pulled from the bin that that client is associated with. In this way the ad agency cannot determine who exactly the ad was fed to. Instead they only know what type of user the ad was fed to.

Most of the research done in this particular field does not place a very strong emphasis on ensuring that advertisers are able to advertise effectively. The work in [71] and [30] are an exception. In order for online social networking and the web in general to continue to flourish it is obvious that advertising needs to grow as well.

2.4 Protection from the OSN Provider

Very recently much work has gone into protecting user content from the online social network provider. The online social network provider acts as the “Eye of God” in that it can see all data that flows through the network. Currently users sign privacy policies and terms of use agreements with the provider which is their only line of defense. Users are now beginning to realize that they do not want to trust online social network providers with their personal data.

In [4], Anderson et al. design an online social network in the client / server architecture that does not rely on the OSN provider to be trusted. Instead the server simply provides availability. That is, name resolution of members in the social network. The actual content of the social network resides on individuals’ computers spread across the Internet. Imagine two users Bob and Alice want to communicate via the online social network. Alice’s computer needs to connect to Bob’s computer but she does not know Bob’s IP address and he does not have a domain name. In fact, it is very likely that both Bob and Alice were given dynamic IP addresses by their ISPs and that there machines are behind routers. In order to avoid elaborate configuration required for every member of the OSN we instead rely on the OSN servers only for name resolution. Alice will connect to the social network server and ask for Bob by his name. The social network server then gives Alice a path to Bob if Bob is online or, if Bob is not online, Alice will store the message locally and the server will notify Bob about this message the next time he is online and provide a path to Alice for Bob. The actual message content is not stored on or passed through the server at any point. This system has not been widely adopted because there is little incentive for an established social network provider to convert to such a system. In fact, they are incentivized not to build such a system because it would limit their ability to target advertisements. Additionally, online social networks are extremely difficult to start because of the paradox that only social networks with a large population are capable of increasing their population. In spite of this, it is this researcher's opinion that a system of this design is the best solution to the problem of trusting the online social network providers. Unfortunately an online social network of this design does not have the enticing revenue potential that traditional online social networks have.

One of the simplest ways to keep data private in computer science is to use encryption. Many papers have been written that attempt to protect privacy and security of user data on online social networks by leveraging encryption. This brings up many important technical and non-technical challenges. An encryption scheme must be chosen, the relationships in the social network must be defined, there must be a clear mechanism for storing and retrieving encrypted data, encryption keys must be distributed, and it must be clear which data is to be encrypted and which is not. In this section we discuss many of the works that try to solve these issues in building an online social network with encryption to protect user privacy and security.

Perhaps the most basic of example of this technique is the "flybynight" system as described in [52]. The authors here have written a Facebook application that helps users send encrypted messages to one another. When the user first uses the application, they create a public / private key pair. The private key is encrypted with a password (also user supplied) and transmitted through Facebook (due to the architecture of Facebook applications) and is stored on the application server. Now when this user, Bob, wants to send a message to another user, Alice, Bob enters message text into the flybynight web form. The flybynight application has a complete list of Bob's friends that have also installed the flybynight application, this includes Alice. The application encrypts Bob's message to Alice using client side javascript with Alice's public key and tags this message with Alice's ID number. Finally the encrypted message is transmitted via Facebook to the flybynight application servers. The next time Alice logs in to flybynight she will be alerted of Bob's message and she can download and un-encrypt the message using her own password-protected private key.

One drawback of this approach is that it relies on client-side javascript to do the encryption which has numerous problems with encryption. The random number generator in javascript is not sufficient to do encryption. The users must trust that the flybynight authors have written javascript that faithfully encrypts the message before sending (and not just sending plain-text or plain-text and dummy cipher-text). In the end this requirement for the user to trust the authors of this paper is not much different from the users trusting Facebook in the first place. While the client has access to the javascript source code, it is laborious and tedious to check on each download that the encryption is performed faithfully.

There are several papers published recently that follow this basic idea of leveraging encryption in order to protect user data in an online social network including [52] [91] [21] and [31]. While these techniques are strong, in that they almost guarantee third parties of any kind cannot access the data in question, they are not streamlined. Their techniques sacrifice user convenience for increased security / privacy. This is a classic trade off often seen in the security research community.

In [53], Luo et al. implement a very similar system they call "facecloak". However, instead of writing a Facebook app that sits inside the Facebook domain. They wrote a Firefox browser extension that is, more comfortably, between the user and Facebook. The authors describe their system as having three phases; setup, encryption, and decryption. During the setup phase, three keys are generated; a master, a personal index, and an access key. The access key is stored locally only for the user to use. The other two keys are distributed to the user's friends (out-of-band). The master key is used to create a symmetric key by a user when they publish data to the social network. This symmetric key is used to encrypt the data being posted. Then viewers can decrypt the data using the publisher's master key. Users tell the system to encrypt a message by proceeding it with a "@@".

While this system is rather clunky, it relies on the Firefox extension heavily rather than Facebook. It is a step in the right direction. This system is much easier to trust because the keys are sent out-of-band and the add-on they wrote helps to automate this process. However, this system (as well as [52]) both relay on client-side javascript to implement encryption. And while either of these systems can ship libraries to fix the poor automatic number generator, they still suffer from the same trust problem as mentioned before.

In a scheme that more readily adapts to online social networks we have Persona [6] by Baden et al. Persona is itself an online social network that leverages attribute-based encryption to enforce privacy and security configurations. Attribute based encryption (ABE) allows users of the social network to organize their friends into groups (e.g. football fans, or co-workers). Several keys are generated in the setup phase: a public key, a master secret key, and a secret key for each friend. These secret keys are based upon the attributes of each corresponding friend. ABE works in such a way that a message encrypted with the structure *football fan* or *co-worker* can be decrypted by any friend that has either *football fan* or *co-worker* in their corresponding secret private key. Additionally, logical operators can be used to create structures of arbitrary length (e.g. *co-worker* and *sibling*, *friend* and not *family*).

Unfortunately ABE encryption is 100 - 1000 times slower than standard RSA encryption and it still requires that keys be exchanged out-of-band. However, many ABE computations can be avoided with clever design (as discussed in the paper). It is clear that ABE is a much better system than any of the standard RSA encryption schemes exemplified in all the previous works mentioned in this section. For no other reason, ABE obviously lends itself to the natural urge for users in an online social network to divide their friends into logical groups. Furthermore, only this technique provides the flexibility that users expect in protecting their data. Persona allows e.g., Bob to send private messages to Alice's friends without revealing to Alice Bob's message or to Bob Alice's friends list. Also, if users desire, they can create one group that contains every person on the social network or, alternatively radical, one group for each user. In this way, Persona and more specifically, ABE encryption are finer grained solutions. Persona has greater flexibility for the user.

We must also consider new actions that the OSN provider might take. In [26], Fredrikson et al. explore the idea of harvesting information about users that is stored in the browser. This idea is very similar to the current implementation of "my location" in many popular browsers. When a web application (e.g., Google maps) wants to know the user's location, the browser has this information and the user is prompted to allow the browser to give the site this information automatically rather than typing it in manually. In [26] we see a system that allows web applications, such as online social networks, access other browser information such as history and bookmarks with explicit user permission. This sort of system, while perhaps very convenient, is very sobering in the light of privacy and security. This paper is motivated by the want of advertising agencies to refine their already spectacularly accurate targeted advertising. Unfortunately user privacy is given little consideration. While the system is opt-in, the specifics about what is being shared and the full repercussions of sharing such data are not yet known. We have seen that the current state of privacy for users in an OSN is bad. Now this paper shows us that research is ongoing that will decrease the privacy of users in the future! This motivates further research on the part of privacy advocates.

2.5 Protecting Location Information

Online social networking has stirred an interesting new trend of users sharing their location information with applications and friends. This allows online social networks to provide targeted, dynamic content based on location information. However, this again raises security and privacy issues because location information is, by its nature, sensitive (users are concerned about physical harm) and time-sensitive. Mobile online social networks make extensive use of user location. A mobile online social network is a network that users access primarily through mobile devices. All of this brings up interesting security and privacy concerns. For example, [109] presents a novel system of automatically categorizing places with tags based on online social network data about those places. A strong imagination is not necessary to envision the sort of malicious activities that might come about with semantically tagged location information about a person. Additionally, techniques like this show us that it is not just other users whom we concern ourselves with when dealing with our location information. Researchers, however good their intentions might be, often create tools like in [109] that can be used maliciously.

Privacy Context Obfuscation (PCO) [70] which has been mentioned previously in this paper does make brief mention of obscuring location information based on parameters such as who is accessing the data, the time of day, and potentially, where the content is being accessed from. For example, users might be able to see the location of other users if and only if they are in the same location. A lot can be inferred from location information, especially if it has some semantic connotation. Using PCO the user simply needs to categorize strangers as not being able to view their location information in order to protect their privacy in a reasonable yet robust way.

Wei, Xu, and Li describe a scheme for sharing location in a privacy preserving way in Mobishare, [98]. This work is an extension of Smokescreen, [19]. In this Mobishare users are able to share their location information with third party applications and other users but the OSN provider nor the location server have complete knowledge of the users' identity and location. This is achieved by splitting location requesters into two groups, strangers and friends. Then, using an encryption scheme to protect location data, this information is transmitted to the location server or the online social network. Using this scheme, the online social network provider cannot learn the location of users and the location server (dealing with only fake ID's) never learns the true identities of the users for which it has location information. A simple diagram of the entities used in this scheme can be seen in Fig. 4.

This system is much more robust than PCO [70] in that it hides location information from the online social network provider itself and the third party location server

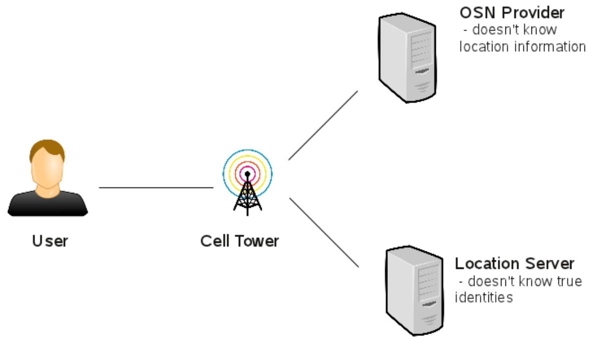

OSN Provider
- doesn't know location information
User
Cell Tower
Location Server
- doesn't know true identities

Fig. 4: Mobishare Architecture. Users send some information to the OSN provider (user ID) and some information to the location server (user location). In this way, the location cannot be linked to the user because neither of the services have enough information.

cannot figure out the identities of the users for which it has location information. In order to place users at locations an adversary would need control of both the location server and the online social network.

In [61] Narayanan et al. design a system for privacy-preserving tests for proximity that does not reveal the identity of the requestor or the requestee. They call this system "Desiderata." In this work the online social network is only used to share public keys. After keys have been shared, users send and receive data directly. In this way the online social network provider does not have access to user location information in any way because information about location is never passed through them. A similar encryption scheme is used in [61] as is used in Mobishare above, [98] to protect user data. In this paper they are able to test if two users are in the same area without each user knowing the other is checking and without either user exposing their current location to the other using ElGamal encryption.

The survey paper [77] attempts to level the playing field and compare many of the various privacy protection schemes in an apples vs. apples comparison. They build a model to represent all of the various ways to attack user location information. More specifically, they have three metrics: attack accuracy, attack certainty, and attack correctness. Here *accuracy* is how confident the attacker is in their own location estimation of another user, *correctness* is how close that estimation is to the correct solution in reality, and *certainty* is the probability that the attacker is correct (over many guesses). This gives a metric for determining if one location privacy protection scheme is better than another. They also show that k-anonymity is not a good measure of location privacy because it only takes into account attack certainty and not the other two values.

3 Network Inference

Network inference is the act of determining the state of some attributes of a network given some other attributes of that same network. Network inference is a major concern for privacy and security in online social networks because it shows that it is possible (even easy in some cases) to determine attributes of users and other information that is not explicitly stated in the data found on the online social network. Futhermore, social networks, unlike most networks, are saturated with senstive, personal information about the users. There are a multiude of things that can be inferred from a social network. In the following subsections we discuss inferring user attributes, identifying hubs, and making link predictions.

3.1 Inferring User Attributes

In [106], Yang, Long, and Smola et al. attempt to determine what users are interested in on an online social network. They do this by using latent factor models. A matrix, $[Y]$ is created in the social network. Each row in this matrix represents a user and each column represents an interest. Each spot in the matrix $Y_{i,j}$ represents how interested in that column that user is. Commonly we can infer this information based on what activities the user participates in, based on social network data such as group membership and text content. The technique of determining some information about a network (or about the nodes of a network specifically) by analyzing given information in the network is referred to as "network inference." In this paper the authors make the critical observation of homophily. This attribute means that users in an online social network are similar to users they are friends with. This paper goes on to apply their network inference of user interests to predict future friendships and interests for users.

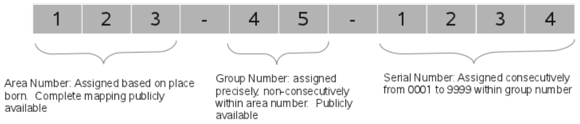

1 2 3 - 4 5 - 1 2 3 4
Area Number: Assigned based on place born. Complete mapping publicly available
Group Number: assigned precisely, non-consecutively within area number. Publicly available
Serial Number: Assigned consecutively from 0001 to 9999 within group number

Fig. 5: SSN Pattern Details

In [3], Acquisti et al. give a detailed account of how, given a limited set of information about a person, they can determine the first 5 digits of that person's social security number with 1 guess 44% of the time. This is due to the systematic nature of social security numbers. Certain social security numbers are given out to people born in certain regions and between certain dates. The details of this pattern can be seen in Fig. 5. This key information is typically found on a user's online social network profile. Without proper protection, anybody will be able to view this information and begin guessing social security numbers following this scheme. Acquisti et al. are able to guess the entire social security number in $< 1,000$ attempts; this is about the same probability as a 3 digit PIN. This is an excellent example of potential identity theft due to information dispersion through an online social network. It is left up to the user to determine if the social network is not only protecting their sensitive information, but recognizing which items of the user's data are sensitive and which are not. Because of the frequency in which users post their home town and birth-date information on their profile, online social networks have created another privacy issue.

In [14], Calandrino et al. perform network inference by using data from recommender systems. A recommender system is commonly employed on many websites especially online auction sites like Amazon.com and ebay.com. On these sites, when a user purchases a new item they are allowed to write a short review of that item and rank it using a 5 star system. The authors point out that because these rating systems form links between items and users there is no possiblitity for an attacker to infer any information and the ratings are publically available. This is also done so that users can see the opinions of others that have purchased certain items.

Calandrino et al. show that these systems actually do reveal sensitive information about a user. They assume every item in the system has a list of associated other items. An attacker can view any item in the system and that item's associated items list. The attacker chooses several items that are associated with the victim (e.g. the attacker knows the victim bought all of these items). Then by monitoring the related items list of all these related items, the attacker can analyze changes. For example, if a new item appears or if an existing item rises in the list it is likely the victim bought that item. However, this technique works best with obscure items. This is because obscure items are more heavily influenced by individual users. The authors point out that when extremely popular items are chosen (e.g. a best seller book) their method only reveals large scale, global community actions.

In [65] we see another example of social network inference, although of a slightly different breed. In this paper many different factors are analyzed to classify users as either "democrat" or "republican." This political affiliation attribute is not explicitly stated in the Twitter online social network. The researchers apply a machine learning algorithm with a training set of known to be either democrat or republican. The system is able to recognize the differences between many attributes of each user and is eventually able to determine, given a new unlabeled user, if that user is more like the republican users it was trained on or more similar to the democratic users it was trained on.

This system takes many factors into consideration including tweet size, tweet vocabulary, sentiment analysis of common political subjects such as Ronald Reagan, various attributes on the user's profile including age, sex, and ethnicity, and finally friend affiliation (assuming the friend has already been labeled democrat *or* republican). These researchers assume that republicans will be friends with other republicans and that democrats are friends with other democrats (homophily). It is more robust than [14], mentioned previously, because it focuses on individual users much better.

Pennacchiotti and Popescu also perform network inference in [65]. In this paper the authors focus on determining if a given user is a democrat or a republican. They train their system on users that are known to be either democrat or republican at first. Attributes of these users are analyzed such as: profile information (e.g., age, sex, and ethnicity), tweeting behaviors (such as average tweets per day, and typical time of tweets); and tweets' linguistic content (such as vocabulary). They are then able to classify new users, live, with precision of 89%, recall of 94%, and F-measure of 91% as either democrat or republican. The authors of this paper mention possible applications of their classification technique including recognizing expert / authoritative users, intelligent web search results, and recommending content or new friends to users. However, the authors don't mention the similar, yet less benign applications such as targeted advertising and stalking.

In [99], Wen et al. are lucky enough to obtain access to a large, unique dataset of more than 30,000 people working at a large scale IT company. They set up a privacy-preserving social sensor network that includes many forms of electronic communication including email, instant messages, and the like. The goal of their research is to study how likely one's interests can be inferred from that person's social connections. They're interested in more than just 1st degree connections as well. In this paper, researchers determine user interests based what is implied by user generated content, and what is explicitly stated on user profiles. Topics are extracted from the user generated content (mostly text data) using latent dire latent dirichlet allocation (L.D.A.).

In [99], we see another scheme that generates a matrix $[Y]$ that has a row for every user and column for every topic. This is very similar to [106]. The spot Y_{ij} is the measure of how interested the *ith* person is in the *jth* topic. Interests are also inferred from the other users in the social neighborhood. Iteratively a user, Bob, is selected from the social network. For every other user a distance is measured (dependent on how much communication occurs between these two users) and the topic interest measures of this second user are weighted and factored into the topic interest measures of Bob. In this way the researchers of this paper also assume homophily.

This paper has the obvious benefit of being able to track virtually all communication between the users because they worked jointly with a large scale IT company. It is perhaps easy to cast this work aside rationalizing that such a complete data set of user interaction is very rare. With a data set like this of course it's easy to determine user interests. However, the take-away from this work is not that large, detailed, complete datasets contain sensitive, implicit information. This is obvious, instead we should worry about generating such datasets in the first place. Many major social networking providers are working toward a more ubiquitous experience. The niave approach to defense against us users generating such a complete dataset is to use social networking sites less. A more subtle approach is simply to use multiple social networks. It is important to note this includes more traditional social networks like face to face communication, and regular mail.

Up until this point all the papers have made the static assumption that users connected in a social network share similar interests both in real life and latent in their attributes in the online social network. However the situation is not this simple. Rather, relationships (like in real life) grow, change, are born and die in online social networks. In fact all we can say is that there is an auto correlation between attributes of closely connected users. In [42], Fond et al. explore the two sides of the auto-correlation coin. One side of that coin is homophily in which users that are similar develop a relationship in the social network. The other side of that coin is social influence. Social influence is the phenomenon of users adapting the views and characteristics of those that they are already close to. Similar to how an ice cube warms as it cools the water and the water cools as it warms the ice. Users in an online social network both find friends similar to themselves and become more similar to their existing friends. This is an important result that many other papers ignore.

Many of the papers in this section begin to bleed over to network link prediction. In [57], Choudury et al. present a very simple network inference technique in a social network that they build out of a dataset of emails at a major university and from the infamous Enron corporation. In this social network nodes represent users (email addresses) and links are formed when there are mroe than x emails sent between two nodes. The tuning of this parameter is central to this paper and the authors discover the the communities formed vary greatly depending on this threshold x. These communities are what is inferred in this social network.

Their link prediction technique (which also varies greatly based on x) is more sophisticated. They create a vector for each user based on that user's features such as degree, and number of communications with every other user. They then run a linear regression on to predict the future communications between this user and a given other user. The key difference between social network inference and link prediction is that link prediction is inferring the values of what certain attributes of the network will be in the future. Network inference does not make a future prediction of values but rather, it tries to guess the current value of certain network attributes. This is explored in greater detail in the next section.

3.2 Locating Hubs

In the study of online social networks, there is the notion of a "hub" sometimes called an "effector", or a "central node". These are the nodes in the social network that are the most active with the most other nodes. Typically, information given to a central node spreads quickly through many of the other nodes in the social network. This topic is one in which we see very little if any research going into protecting user privacy directly. Instead, we see work in identifying hubs. This is strikingly different from the vast majority of papers covered in this survey up until this point. We should recognize that locating hubs is still relevant in a security and privacy sense. Firstly, this paper assumes that accessing any data that is not explicitly stated and explicitly shared by the user is a privacy concern. Additionally, central nodes must be found in a privacy preserving way if they're to be found at all. That is, we must be able to find hubs without learning their true identity in the social network thus linking them to all sorts of profile and other social graph information.

Research in this area really begins with the tangentially related network community detection [46]. In this work researchers attempt to find communities in the social network, not central nodes. However, it is very likely that each community will have a top effector in it. Their paper is a survey of the various techniques for finding communities including the local spectral partitioning algorithm and the Metis+MQI algorithm. The first research we see in locating central nodes is [43] in which we have the problem formally defined. It might be an interesting area of research to use community detection to evaluate locating hubs. In this paper researchers Lappas et al. have a more theoretical definition of central nodes than in most papers.

Imagine a graph with weighted edges connecting the nodes. Some nodes are active and some are inactive. At each time step each active node will infect the nodes it is

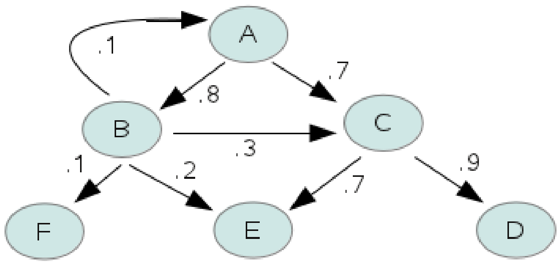


Fig. 6: Top K Effectors: Here we can see that node B has the highest degree but node A is the top effector due to the edge weights.

directly connected to with probability equal to the weight of the edge connecting the two nodes in question. (edge weights here are between 0 and 1). We call the set of active nodes "A" The problem now is: given k, find the set of k nodes such that if they had been the only original active nodes, the resulting active set "A" would be the same size with the same number of time steps as the original set "A". This point is illustrated in Fig. 6 The authors point out that the top k nodes aren't necessarily centrally-located due to edge weights but it is very common that they are.

In [43], Lappas et al. prove that this problem is NP-hard to solve optimally and NP-hard to approximate. However, there are some special cases, for example when the graph is a tree, the problem can be solved in polynomial time with the use of dynamic programming techniques. This is a powerful result which has spurred much research in the field.

Solving this problem has many powerful applications including determining which users to advertise a product to in order to reach the maximum number of users with the minimum amount of cost, locating key members of crime organizations, and finding the most popular users of a social network. We see the application of finding crime leaders in [89]. In this paper the problem is stated differently. In fact, this is not an online social network at all. Instead, a social graph is built from an existing real world social network made up of actors, locations, and resources as nodes. A crime draws hyper-edges between these nodes that connect multiple nodes at once.

Researchers then, because finding the top-k effectors is NP-hard, approximate them. "Approximate" here is used in the traditional sense of the word. They plot the graph and measure several different metrics for nodes; degree, betweeness, closeness, eigenvector centrality, and page rank centrality. Using these measures police can identify (manually) the nodes that have the most effect on the network. Additionally, we can associate high level meaning about these different metrics. For example, a node with high *betweeness* is the "message passer" between many parties in the social network. Also, because nodes of the graph can be places or resources police are able to fight crime in a variety of ways. Rather than trying to catch individuals they can more heavily patrol a certain area or make it more difficult to obtain certain resources.

Another application of locating hubs in online social networks is [16], in which researchers Wei Chen, Chi Wang, and Yajun Wang try to solve the problem of finding a small set of k nodes such that these k nodes have the maximum influence on other nodes under certain cascade models. The application of this problem is to target advertising at these select nodes. In this way we can minimize advertising effort and maximize penetration.

Throughout the paper (after proving the problem is NP-hard) they iterate through several different algorithms that improve upon one another comparing them each to their own heuristic based MIA algorithm. The MIA algorithm begins by finding the maximum influence path using Djkstra's shortest path algorithm. A maximum influence path between u and v is the set of nodes and edges that the influence is most likely (based on edge weights) to pass through to get from u to v. The algorithm then collects the paths that are shorter than some threshold, and adds them to an arborescence (either on the root or end node of the path). This gives us the local influence of each node. This technique is very powerful and outperforms several other greedy techniques. Typically running in a matter of milliseconds where other techniques take upward of an hour. However, it does require the entire social graph as input.

Ilays, Shafiq, Liu and Radha present [36] which does not require the entire social graph. Their paper attempts to identify the top information hubs in the online social network and it is distributed, accurate, and it does not sacrifice user privacy, The intuition behind this work is that highly connected users tend to have more interactions with their neighbors than less connected users.

The technique is based on the eigenvector centrality (EVC) algorithm for determining influence. However, EVC does not perform well in a social network because there are multiple poles or communities. In order to solve this problem Ilays et al. use their technique *PCC* which is defined for each node as the euclidean distance norm from the origin in P dimensional eigenspace. P here is the number of supposed communities. When P=1, PCC is the same as EVC. An interesting area of research would be how to automate the discovery of the parameter P.

This paper is the first to address privacy directly when locating central hubs in online social networks in that their technique does not require a complete social graph in order to find hubs. Rather, as the users interact in the social network communication is sensed passively by the algorithm. The identities of the users and the content of the messages are not known to the algorithm. The consensus of the research community in this specific topic seems to be that privacy comes second to the wishes of the industry. In every other paper the technique involves are complete graph when clearly, most of the information in that graph is unnecessary. This is a dangerous behavior to repeat!

3.3 Link Prediction

Link prediction can be thought of as a special case of network inference. The idea here is that we're trying to infer the future state of the graph based on the current state or based on a collection of past states. In [106], the authors are able to infer user interests based on publicly available information in the Yahoo! Pulse online social network. They use this information to predict users future friends and interests. This system shows that it is possible to take common, public, explicit social network information and determine not only the interests of users but what interests and friends that user will gain in the future. It is important to remember that information can be retrieved from a social network even if that information is not explicitly stated. This even applies to future data! If researchers are to protect users and if users are to protect themselves we must learn what can be inferred from the data that is released, however small or insignificant that data might seem.

In [72], Roth et al. devise a system that is strikingly similar to social network inference as discussed in the previous section although this term is not used. We recognize that this paper is about link prediction because the authors are trying to determine who users will become friends with in the future. They can then take this result and apply it directly in the form of friend suggestions for the user. Because of this application they're able to afford manual false positive detection. That is, users can give feedback directly and explicitly when friends suggested are not a good match.

Their technique is based on email threads. The problem is formulated as follows: user email threads are analyzed then when writing a new email, the system suggests to the user other users that should also receive the email in question based on groups of email recipients in the past. The best application of such a system would probably be a "don't forget Bob" system that alerts the user when they are addressing an email to a group of people and have forgotten one member of the group. The authors propose several methods for determining if a user should be included in a group. One such method takes the list of recipients that the user has already addressed the email (the set $\{S\}$) and a 'new' user C. The implicit group $\{S\} \cup C$ has a certain similarity to $\{S\}$ based on previous communications between the members of the new implicit group and communications between the members of the original group $\{S\}$. If this similarity is above a certain threshold (a tunable parameter) then the friend is included in the list of suggestions. This technique works reasonably well and the authors are able to predict users for group membership with precision and recall of about 80%.

One of the more simple network link prediction schemes can be found in [45] by researchers Leskovec, Huttenlocher, and Kleinberg. In this paper the researchers have a social graph that has positive and negative signs on every edge. Such a network can be found on epinions.com or slashdot.org where users are able to rate the comments of one another thus establishing explicit, fine grained indications of their opinions about one another. A positive relationship has a positive sign and a negative relationship between two users has a negative sign. The problem stated is that one of these edges in the social network has its sign removed and the researchers are tasked with determining what sign it was originally. Although this does not frame the problem like a link prediction problem. It can just as well be thought of as that link having never existed and the researchers are trying to predict whether the two users connected by that edge will have a positive or negative relationship if they are connected in the future.

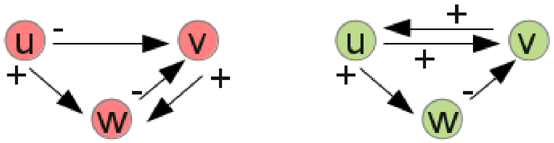


Fig. 7: Two example triads. There are sixteen possible triads because we have directed edges, and binary signs on each edge between 3 nodes.

Their technique works by focusing on the existing "triads" between the two nodes in question u and v. A triad is formed by some third node w that connects u and v. This is illustrated in Fig. 7. Using these triads (of which there are typically several) we can determine if the two are likely to have a positive or negative link. We can see that there are 16 possible triads. The edge between node w and u might be in the direction $w \rightarrow u$ or $u \rightarrow w$ and it might be positive or negative. Accounting for this possibility on both sides of w (toward u and toward v positive and negative) there are $2 * 2 * 2 * 2 = 16$ possible triads. Using these characteristics and examining all triads between u and v we can see if node u has predominately positive incoming links or if v has very few outgoing links at all and so on.

This technique is simple in that it requires the complete network graph but really only relies on a small portion of it to determine a given, unweighted edge. Secondly, simple probability of edges are analyzed. There is no higher level technique or information taken into account; like we will soon see in other papers later in this section. Additionally, the weights are binary which is not very interesting.

In [13], Bonneau et al. saw an opportunity to exploit the Facebook social network. Facebook exposes a small subset of user data in the form of a publicly available profile page. At the time (2009) Facebook showed a random selection of 8 friends on any given user's public profile page. It was assumed that 8 friends was not enough to determine anything significant about the social network or community of that user. The paper proves that 8 friends are enough (hence the title) to make close approximations to common graph metrics including shortest path between nodes, dominating sets, and community detection. They are able to calculate node degree for example because any node with $degree \leq 8$ is apparent because that nodes profile page will have less than 8 friends listed. Any node with $degree > 8$ typically shows up on the profile pages of the nodes in their community. Also, attackers are able to probe the public profile page in question an unlimited number of times, each time getting a different random selection of 8 friends. Additionally, there is no limit to how many different nodes can be probed. While it is impossible to know if every friend has been exposed through the random friend probing. It is simple to probe many times until the random selection begins showing repeated friends. The graph metrics exposed by these friendships can be used to do link prediction. Like, for example, likely future friends based on existing mutual friends.

An application of link prediction in the twitter microblog social network can be seen in [27]. Authors Galuba, Aberer, Chakraborty, Despotovic, and Kellerer attempt to predict which users on twitter will mention a given URL mentioned by a second given user on twitter. Their machine learning technique takes into account the virality of the link so far, the number of followees of the user (those that this user is following) that tweeted the link in question, and the likelihood that the user in question tweets any links at all. This paper exhibits $50 - 60\%$ recall and a $\leq 15\%$ false positive rate. This paper is a classic example of link prediction in an online social network. However it requires complete knowledge of the social network before hand and a modest training period for the machine learning algorithm.

Macropol and Singh attempt to use actual message content to predict information dissemination in [54]. In this papers the authors are working with a data set of email conversations. Cleverly, they split the data set up into individual threads. Each email thread is a set of emails that transpired between a (usually small) set of individuals in the social network. In this context a node is a user and an email is a directed edge between that user and the recipient. They run the messages of a thread through LDA after some light pre-processing to determine the topic of that particular thread. Again, because the data set is divided into threads we get the topics of individual threads not the topic of the entire data set which is so broad it becomes meaningless. They assume that topic and conversation length (measured in number of email responses) are correlated. Intuitively, this is like saying, "When we talk about this topic we typically have 40 or 50 response emails back and forth between the two of us." Using this information we can predict that certain topics will generate longer email threads between these users than other topics.

This paper is different than those previously mentioned because they're using message content to predict future links. Using latent information like this is much more robust than lower level metrics and is the future of research in online social network link prediction.

4 Distributed Social Networks

Many researchers have turned to the idea of a distributed social network in the face of privacy concerns about online social network providers harvesting user data. One example of this is [4] in section 2.4. At a glance, this seems to be a fantastic solution to the problem because it removes the online social network provider almost entirely. In a distributed social network there is no central store of user data. Rather, each user's machine holds that user's information. The only centralized service performed is lookup. However, upon closer inspection it is clear that distributed online social networks have several technical security and privacy challenges. This section outlines those problems and highlights some of the research that has gone into solving them.

In [12], we see an early, basic implementation of distributed online social network. Users store their data locally and encrypted. When a user generates a new piece of data it is public / private encrypted and the user specifies who can access this data. The public encryption key is spread among the users who are allowed access and the user posting the information. The keys are split into enough parts such that the posting user has 50% and the authorized users collectively have the remaining 50%. Any one user needs only 50% of the key parts in order to decrypt the actual data. Either they can get them from the user directly or from each of the other users in the authorized group. This system leaves much to be desired. For example, the system is robust against one node failure only. Additionally, the simple nature of data storage (user data is stored locally) means that every user must be running a dedicated server in order to achieve any kind of usability.

The problem of node availability is properly addressed in [1]. Here authors Abbas et al. design a system that maintains links in the face of low reliability with the help of a gossip protocol. This protocol is simple in the common case. Alice wants to become Bob's friend. Alice sends a request to Bob, Bob decides (via user input) to accept or deny the friend request and sends a reply or deny back to Alice. However, if Bob is not online during the request it is retried every five minutes for one day and then once every 24 hours for one week. Unfortunately this work does not address privacy or security of such a social network directly and it is rather naive.

Together these first two papers outline what is necessary to build a distributed social network. A more robust, system is designed in [37] that addresses privacy in a distributed social network directly. "OneSwarm" is a system that enables privacy in peer to peer data sharing applications like bittorrent. Currently, bittorrent is not 100% anonymous because users connect to each other directly (via IP address). These connections can be passively observed by anyone sharing or downloading some data. The IP addresses can be traced back to individual computers or at the very least internet service providers' clients. The current best solution is to use proxy services like TOR to mask one's IP address. However, this comes at a performance cost because packets need to be routed to some number of random hosts through the proxy. OneSwarm offers a privacy solution that has better performance that these proxies.

The basic approach of OneSwarm is to apply the proxy only when searching for content and only internally through other users of the network. In this way the content can be found anonymously (via random node hopping) and the content supplier can determine if the searcher has adequate permissions to access these files. Once this has been decided the peers can connect directly. What they achieve is the ability of users to search the network to see which users have the content they're looking for without exposing their own public IP address. This system is much more robust than the previous two mentioned. However, bittorrent is a liberal interpretation of a social network. Users do little socializing and do not know each other personally. Perhaps with a new user authorization feature like the one described in OneSwam this might change. Finally, it would seem that this does not protect users from exposing that they are downloading content. It may be possible to identify a user in this social network based on their download history. Furthermore, if much of the previous download history of a user is known. It might be simple to recognize that user when they search (anonymously) for similar content even though their IP address is masked with a proxy. It is, after all, likely that a given user will perform many searches especially if they are very new to the social network.

Creating a privacy preserving distributed social network is a daunting task. Backes et al. develop an API framework in [5]. This work defines the task abstractly thus making the task of implementing much easier. They describe a security API for a distributed social network through extensive use of cryptography. The idea of this is that any existing distributed social network can design their system with these API calls so that they can focus on more important social networking issues and features such as distributed content sharing, performance, and friendship in the face of node failure. By using these API calls the social network can guarantee complete user anonymity. This includes the seemingly fundamental issue that users expose themselves when they querying other nodes for information.

This system is very similar to Persona [6] which is mentioned in the Protection from the OSN Provider sub section. Persona uses attribute based encryption as well as standard public / private key encryption. The system in [5] works using only public / private key encryption which cuts down on the computational time, compared to Persona, as well as the complexity of distributing and managing so many keys. In order to protect user privacy when making queries pseudonyms are used. For example, if B wants to become friends with A. B sends it's own pseudonym which only B and existing friends of B know is linked to B. The API provides a *handle* function which takes a Mask from a user (this can be the user's real identity, the user's pseudonym, or the user's relation). The function returns a list of handles to A's data. This is not the data itself but merely a reference to the data with meta-data for names. (e.g. "A's Birthday" instead of "10-26-90").

The API also includes a *getResource* function takes a Mask and a resource handle. This function only returns if the Mask corresponds to a user that has access to that handle. When it does return, it returns the actual content at that handle. There is also a *putResource* function and a 2nd degree friend function that both work in a similar, straightforward way to their respective counterparts. Finally, there is a simple *getFriends* function that takes a mask and returns a list of that mask's friends that accept indirect connections. Basic session key encryption is used to protect the messages that travel across the network when any of these API calls are made by any of the nodes in the social network. Zero-knowledge proofs are used in the paper to show that users are who they say they are without revealing who they are. This last paper is by far the most robust and cleanly designed system reviewed in this section.

5 Maintaining Proper OSN Operation

For the majority of the paper thus far we have focused on the techniques that adversaries are using to expose user data as well as the schemes benevolent researchers have presented that protect against these threats. Both sides of this coin have a myriad of possibilities. However, this is not the only privacy or security concern in online social networks. The providers, users, and advertisers all have a vested interest in making sure that the OSN in functioning properly so that it can thrive. If the social network is not succeeding then users do not gain any enjoyment or convenience, advertisers cannot market their products, and OSN providers cannot make money. This section outlines solutions and challenges that arise in maintaining the proper operation of an online social network.

5.1 Trust

In online social networking there is a notion of "trust" which we can define in the traditional sense: "reliance on the integrity, strength, ability, surety, etc., of a person or thing; confidence." Trust is a key concept in online social networks. It can be leveraged and applied in many different ways. For example we saw in [21] in section 2.4 that the users of a social network often cannot trust the online social network provider but they can trust their own social network connections. We'll see this theme repeatedly. We can use this notion of trust to maintain the operation of an online socail network.

The most common application of trust in an online social network is that of spam detection / removal as seen in [29], and [34]. In [105] the authors detect spam in social bookmarking websites. Social bookmarking sites are social networking sites that allow users to tag and share web pages (bookmarks). However, there is a strong incentive for sites to pollute the social network with phony, malicious tags on rival content or phony, intriguing tags on their own content in order to increase page visits to make their ad space more valuable. [105] exemplifies the spam problem in an online social network. However, they do not use trust in order to detect spam. Another example of spam in an online social network is [29]. Researchers here show that 8% of the 25 million URLs that have been posted on twitter point to malicious sites.

LENS [34] is example of a system that uses trust in an online social network in order to prevent transmission of spam email. Hameed et al. accomplish this by ensuring that email can only come from direct relations in the social network of the recipient. This should logically cover the vast majority of email a user receives. For the less common emails that come from outside a user's social network pre-designated gatekeepers are used to vouch for the legitimacy of the sender. Gatekeepers can vouch only for nodes inside their social circle and gatekeepers are selected based on recommendations from 2nd tier social circles of the receiver (Friends of Friends) recursively. This can be seen in Fig. 8

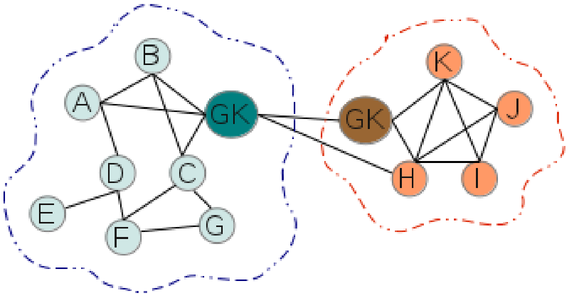


Fig. 8: Gatekeepers (GKs) are used to forward messages outside of a users social circle.

In [7], Bamberger et al. investigate trust in vehicular networks. In this scenario users are passengers in a vehicle. The vehicles pass information to and from one another such as traffic reports and upcoming points of interest. Because the cars are doing much of the information passing automatically they actually make up the social network. However, in a completely open system it is trivial for malicious users / vehicles to spread false or malicious information throughout the network. In this paper a system is devised to determine whether or not to trust information from specific vehicles. Firstly, we have the notion of an agent's opinion about a situation. This opinion is expressed as a vector of evidence $\overrightarrow{E}$ for or against the situation being true. Each $e_i \in \overrightarrow{E}$ is a floating point value. Positive numbers indicate that a situation is likely (from the agents point of view), negative numbers indicate that a situation is unlikely. Magnitude expresses the strength of the evidence. We create a vector of situations $\overrightarrow{S}$, such that each spot in that vector is the agent's opinion $\overrightarrow{E}$ (vector of evidence) of that situation occurring.

We can then combine our opinion of a situation (e.g. a new stop sign exists at a certain intersection) with the opinions received from others. If the evidence is interdependent we use the average fusion, if the evidence is inter-independent we use the cumulative fusion of the vectors. After the agent knows with a great amount of certainty (either from experiencing the situation's outcome or from many other trusted entities) it can evluate all reports from other agents as truthful or not.

Many e-commerce websites have online reviewing or content rating systems that allows users who have purchased an item or content to write a review and rate that object. This is extremely useful in quickly recognizing scams or low quality items and other malicious activity on the site. However, marketing departments have caught on to this trend and now pay employees to go to popular third party websites that sell their merchandise, purchase the items in question, and them leave immaculate positive reviews in order to skew the statistics and convince future users to buy. The authors of [18] design a system call the Trust Antecedent Factor (TAF) model which allows users to see which of the reviewers they should trust and which they cannot.

In this system there are three entities, users (have not purchased an item), contributers, and raters (have purchased an item). A single person can, at times, play multiple roles. The system expects users (raters) to rate objects that were created by contributors. Raters can trust / dis-trust contributors but they must do so explicitly. Additionally, secondary users (a user that is not rating an item directly) can trust / dis-trust raters but they must also do it explicitly. The authors actually predict the future trust / distrust between secondary users, raters, and contributors based on the assumption that good ratings implicate trust and bad ratings implicate distrust. Their model makes heavy use of probabilistic sampling. This system can be used by raters to determine whether or not they can trust the new objects of a contributor and whether or not they can trust the ratings of other users. The paper is very theoretical and it is often difficult to get a firm grasp on certain details. For example, it is unclear how many ratings are necessary before accurate trust predictions can be made.

In [8], the authors suggest that trust in an online social network can be derived from attribute disclosure. That is, they make the assumption that users trust those that they have released the most information about themselves too. Their model, in more detail, says that as users share more sensitive information with other users the trust between those users grows. Trust is measured as the sum of all the weights of the shared data items normalized over the total number of shared items. The weights are determined by how sensitive the data in question is. This last paper is an example of network inference, the topic of the next section.

5.2 Sybil Attacks

Sybil attacks are a very important part of maintaining the proper operation of an online social network. A “sybil attack” is an attack in which the attacker subverts the reputation system of a network by creating (usually multiple) pseudonymous entities. In the context of an online social network, there are many ways to subvert the reputation system including sending spam, leveraging sybil friendship to map or de-anonymize the social network, influencing elections of content rating systems (e.g., digg.com, reddit.com), and simply impersonating another user to slander them.

Whanau [47] is an example of classic sybil problem in networking. That problem is that of a distributed hash table in the face of sybil attacks. Suppose there is some node in the distributed system u that needs to know the value of index y in the distributed hash table. However, node u does not know the section of the hash table that contains value y. Instead it forwards the request on to a node that is closer to that section of the hash table. This forwarding process is repeated until the real value of y is found and the recursion is unwound until the original node u has the value of y. This works well if you assume that none of the nodes are malicious (sybil nodes). In the case of sybil nodes some system must be devised in order to avoid misinformation being spread while still maintaining performance and minimizing network traffic.

While the sybil problem mentioned in this paper is not specifically an online social network instance of the problem, it is not hard to imagine this problem adapted to an online social network or for online social networking techniques like the following to be used in foiling this attack.

All Your Contacts Are Belong To Us [11], exhibits an example of another threat that sybil accounts pose, this time specific to online social networks. Here the authors explore the technical challenges for an attacker trying to steal identity information from other users in an online social network. The idea here is that the attacker picks a user (Bob) whom they have access to a relatively large amount of personal information. Perhaps this person is a friend of the attacker outside the social network. The attacker then creates a sybil account that closely resembles Bob and uses this account to connect to people that would normally friend Bob. This attack (referred to as profile cloning) is very effective but relies on some social engineering to convince friends of Bob to connect with the sybil account.

The authors suggest this entire process could be automated by scraping information about users from the social network. When users are found that reveal a relatively large amount of personal information, they are targeted for cloning. The current strongest defense against this type of automated attack is CAPTCHAs which defends against automation but not against manual profile cloning.

The authors of [11], however, do incorporate a method to break CAPTCHAs. Specifically, they begin be drawing a point half way between every letter in the word. They then approximate this line using a third degree polynomial. The goal of this is to unbend or un-distort the word in the CAPTCHA. A third degree polynomial is used because of typical irregularities in English letters. (e.g. P draws a higher point than g). Then, each pixel column (hopefully corresponding to a letter) is moved up or down so that the third degree polynomial becomes a straight line. Finally, this new translated image, is run through several different imagemagik filters. Terreract is run on each of the outputs from the individual filters which outputs letter approximations of the input.

It is very common for sybil accounts to display interesting and unique connectivity characteristics because it is difficult for the attacker to gain connections for their sybil accounts such that they create a community that is similar to that of a real account. Specifically, while the sybil accounts may be strongly connected to one another, there is typically a small cut between the real nodes of the system and the sybil nodes. This means that there are few links between sybil account communities and non-sybil accounts at all. The reason for this is two fold. Firstly, the attacker does not have in depth knowledge of the social network graph as a whole so it is difficult to make connections that closely resemble the characteristics those of other neighborhoods. Secondly, it is a difficult social engineering task to convince users to make a direct connection with a sybil account. Indeed, maintaining a single, real account in an online social network can be a time-consuming task! All of the papers following in this section take advantage of the small cut attribute of sybil communities in some way.

In [111], Yu et al. take advantage of this small cut observation. They leverage a random route which is slightly different from a random walk of the social graph. In a random walk, a node is entered from some edge, and that same node is exited upon a randomly chosen (with uniform probability among edges) edge. In a random route the output edge is determined by the input edge at each node. As a result, two nodes following the same random route might enter the same node on the same edge. In this case they would both leave on the same edge as well. It is possible these two routes converge but once they have entered the same node on the same edge, they will not diverge.

The system these authors devise, called "sybilguard", starts from a known, non-sybil verifier node and an unknown, suspect node. Random routes are generated for each. Because sybil nodes are strongly connected and there is a small cut between sybil communities and normal users it is very unlikely that the routes will intersect, even if only at one point unless they are either both sybil or both non-sybil nodes. This system works very well but it will stand to be improved upon two more times.

In [110], the same group of researchers improve upon there own techniques with a system they call "sybillimit." They acknowledge that there previous approach was computationally expensive and they ease this burden while tightening their sybil finding standards. They do this by running several random routes that are each shorter rather than one long random route. Additionally, they determine the (now shorter) random route length by starting at length = 1 and doubling that length until a certain percentage of pre-selected benchmark nodes K have been verified. The trick is to select mostly nodes for K that we already know are likely to be non-sybil starting with definitely known to be non-sybil and gathering more based on runs of sybillimit.

To improve again on this idea, Wei et al. propose "sybildefender" [97]. This paper improves yet again on this core random route algorithm. These authors suggest that we can take advantage of the small cut property to identify sybil communities instead of single sybil nodes at a time. More specifically, they implement [110] as is and then simply perform a new random route from the now known sybil node. It is extremely likely that the nodes we visit on this route are also sybil nodes because sybil communities are often strongly connected and there is a small cut to non-sybil nodes which means it is very unlikely that the random tour takes us into normal nodes.

The assumption that there is a small cut between sybil nodes and non-sybil nodes in a social network is addressed directly in [96]. This paper attempts to compare the various sybil identification techniques that leverage online social networking. They conclude that many of the techniques work reasonably well but they all depend on the small cut attribute and that sybil networks are strongly connected. They argue that typical attackers may wise up to this technique and make adjustments to how they attack the network. Specifically, an attacker could create sybil accounts that represent real (perhaps famous) people and friend only non-sybil accounts. In this way systems like sybildefender [97] would almost certainly mis-identify the sybil accounts because there would not be a small cut between sybil nodes and non-sybil nodes and the resulting sybil communities would no longer be strongly connected. Indeed, it's likely that sybil communities would not exist.

Perhaps sybil nodes are not inherently evil. For example, a user might create multiple identities on a social network in order to maintain a higher level of privacy. A user might use one account to communicate with co-workers and another account to communicate with family that are considered more privileged. A third account may be used to post content related to controversial political or religious views. In this context, privacy is extremely important and creating a sybil account seems rational. In [60], [92], [66], and [95] we see work in the idea of sybil tolerant schemes. A sybil tolerant scheme is one in which sybil accounts are allowed to exist in the network. Instead, the network is designed in such a way that those accounts won't be able to adversely effect the operation of that network.

For example, in SumUp [92] users vote on user generated content in an OSN. A malicious user may generate many sybil accounts and use them all to vote on a single piece of content in order to skew the real vote ranking. In a social network such as this, recognizing sybil accounts often can't be done until it's too late and the sybil accounts have already skewed the results of some election. Instead authors Tran et al. tolerate sybil accounts in their network and simply make it nearly impossible to use them to mass vote in this way.

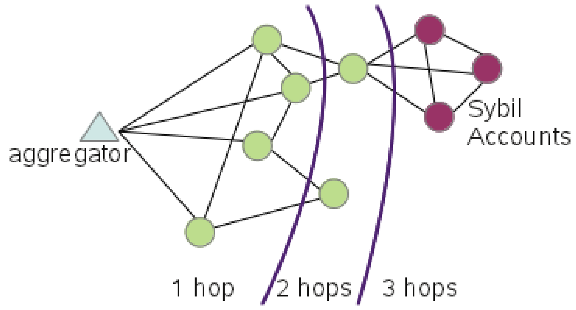


Fig. 9: Sybil accounts are often in "pockets" created by the small cut property. As a result the edges that make the small cut will be congested with votes unlike edges between legitimate users.

Their scheme works by randomly selecting users to be "vote collectors" in the network. These vote collectors are responsible for counting the nodes. Votes are aggregated to the vote collectors across the edges that make connections between users in the online social network. Because sybil accounts have the small cut property it is likely that they will be many hops from the aggregator (as seen in Fig. 9) and, that the edges closer to the sybil accounts (particularly the edges that are the small cut set) will be congested with vote traffic. Using this knowledge, the authors are able to limit the voting rate of sybil accounts by amortizing a synthetic cost of voting across all the users. By recognizing also that most elections (even extremely popular elections) only garner vote participation of less than 1%. Using this knowledge the authors can give edges an allowable number of votes to traverse over. Edges closest to the aggregator get the highest number of allowable vote traversals and those edges further out get less. In this way sybil accounts are limited in the number of votes they can submit to the aggregator.

This work culminates in [95] in which the authors recognize that the schemes in [60], [92], and [66] are all very similar and all have a common computational intensity problem. Specifically, the aggregator nodes are selected randomly which means they are often far from one another. The work in [95] solves this problem by finding aggregator nodes that are close to both the sender and the receiver.

5.3 Web 2.0 Security

There are few works in the literature that address security directly as opposed to privacy in online social networks. There are several reasons for this. Firstly, it might just be that security in online social networks is not a well defined term. In fact this is very likely considering the nearly ubiquitous use of the (very similar) term "privacy." In light of this we'll begin this section with a clear definition of both. Privacy, in the abstract, is the state or condition of being free from being observed or disturbed by other people. Security, in the abstract, is the state of being free from danger or threat. It would appear that in online social networks these two are the same. Users cannot be physically harmed by other users. The only way then to cause harm in this context is to observe information about that user and then use it in some malicious way. So, what is the difference between security and privacy in online social networks?

We shall re-define privacy and security in the context of online social networks so they are more discrete. Privacy is protecting information through technical means that an adversary legally has access to. Security is the protection of information through legal and technical means that the adversary does or should not have access to legally. Using this definition of both we can forge ahead. In fact, there is little security specific to online social networks that does not also apply to web 2.0 web sites in general because online social networks are almost always web based.

Several potential privacy and security issues that arise when building an online social network are outlined in [112] by Zhang et al. These researchers first define the core features a system needs to have in order to be considered an online social network. An online social network needs to allow users to create / cancel accounts, mutable user profiles, and the ability to upload / edit user generated content. The authors go on to discuss the security issues that designers face in implementing these features. Should users have to submit their real names or (effectively anonymous) usernames? Should user profiles be publicly available? What architecture should be choosen, distributed or client-server? Where can user data be safely stored? The consequently intricate details of these decisions are covered in the paper. This paper also categorizes attackers into two types; inside attackers (those that utilize a legitimate account) and outside attackers.

In [9], researchers Bau et al. survey a collection of implementations of a tool called a "black box web vulnerability tester" The idea of a tool like this is to recognize security issues in web applications. The tools recognize a wide variety of over 100 different vulnerabilities that the authors categorize down to 6 distinct types: cross site scripting (xss), SQL injection (SQLi), session management vulnerabilities, cross-site request forgeries, ssl/server configuration errors, and information leakage (through source code or error message disclosure). These categories do cover a very wide section of vulnerabilities on the web but they are not specific to online social networking sites. Any site that accepts user generated content is susceptible to such attacks.

An interesting area of research would be allowing users to determine if sites are protected against these various. This paper focuses on the feasibility and usefulness of these automated black box testers for the site maintainers. However the maintainer has no obligation or motivation to use these black box testers in the first place, much less publish the results! Instead it is up to users to select which social networks they want to use and security should be high on the list of important attributes when making such a choice.

Another interesting tool is discussed in [94]. The problem is as follows. Many web developers, when writing a form in html, use javascript to sanitize the input before it is posted to some server side script for processing. This is becoming more and more common due to AJAX where much of the computation that is typically server side, is now done by javascript on the client side in order to improve response times for the user on that site. The issue here is that an attacker can download the source code from the site, edit the javascript so as to avoid any security enforcement and then use the page to submit to the site. This can be done rather easily with moderate programming and technical skills by writing a browser extension or using an existing one like grease-monkey. In fact, Mozilla Firefox recently released a "scratch-pad" feature that allows users to run arbitrary javascript in the context of any web page they are viewing.

In order to stop this type of attack (which is classified as a source code exposure attack) the authors of this paper propose a technique in which the client side javascript is replicated and ran simultaneously on the server. The output of both the client side replica and the server side replica are compared and any discrepancy should indicate client side tampering. The draw back of this approach is obviously overhead. For the client directly there is no induced overhead. However the client will have to wait for the server to do the same computation that they themselves have already doen and then for the server to make the comparison before finally receiving a response.

Additionally, javascript should never be used to enforce any security policy in the first place. As a result all of the major server side scripting languages can protect themselves against any attack that the javascript was being used to protect against. This technique would be useful if we could enforce policies using javascript to lessen the the burden on the server (and increase performance for the client) but this method shoots itself in the foot by replicating the computation on the server side and negating any performance gain that would have been incurred! Why not just enforce security on the server side in a server side scripting languages in the first place?

Instead of replicating the client computation on the server side authors Stamm, Sterne, and Markham present [81] in which they introduce a new security layer. They refer to this layer as a "Content Security Policy." The idea behind this new layer is to remove many of the features from web programming that introduce security issues in the first place. Examples of features removed include html element onclick values (which are replaced with event handler functions), inline scripting (the <script> tag), and the eval() function in javascript. A set of user specified "policies" are put in place. These policies allows users to disable any other feature they feel threatens the security of their site. Specifically the 'src' value in html tags is a common culprit of cross site request forgeries so it is often placed in the user's privacy policy.

This new tool focuses on removing things from various web development languages that cause security issues. In the community it is already generally accepted not to use many of these features. It stands to reason that any developer who is specifying the policy for CSP must be aware of the security issues of the certain functions he or she would be removing. However if that were the case the developer could just as easily avoid their use in the first place.

In addition to malicious activities like sybil accounts and general web 2.0 exploits, authors Gao et al. point out that attackers also infect online social networks with malware, [28]. The first worm to have significant penetration is called "Koobface." In reality, the OSN is not infected. Rather, individual client machines are infected and the malware uses the OSNs that the client is active on, to propogate itself. Although the malware author assumes that the client uses specific online social networks, this propogation method is quite effective because the malware can take advantage of social phishing in order to bait victims into clicking malicious URLs. This is covered in detail in [104]. The authors of this paper propose a scheme for detecting OSN worms. A variety of other methods in which malware can spread through a social network is discussed in a survey by Dolvara Gunatilaka [32]. For example, URL shorteners are used extensively by attackers in order to obfuscate the true location of a link. In the context of an online socail network shortened URLs are trusted more readily by users because those URLs are coming from trusted friends.

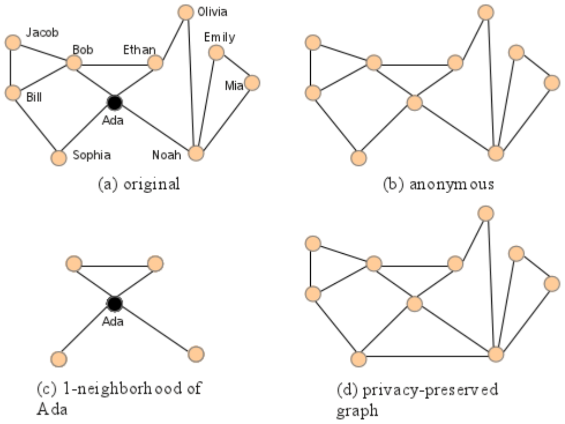
6 OSN Anonymization and De-Anonymization

In order for research to continue in industry and especially in academia online social network providers must publish their data. However, as we have seen throughout this survey, the explicit and implicit information contained in the social graph is extremely sensitive. OSN providers must have a way to publish these data sets without negative consequences for the users involved. This is known as anonymization. The practice of reversing this process (taking an anonymized data set and identifying the nodes) is de-anonymization.

Anonymization and de-anonymization are well studied in software engineering. Specifically, large software systems are written that interact with a database. The software testing needs to be outsourced for efficiency reasons but the software cannot be tested without the database because they are so closely intertwined. Additionally, the data in the database contains sensitive information that cannot be publicized. The area is well studied with solutions typically involving replacing key data values in the database so that it can be publicized safely. For example, the most naive approach is to simply replace unique entity identifies with unique pseudonyms.

The main goal of any anonymization effort is to make the entities in question indistinguishable from some number of other entities. This metric is known as k-anonymization or a k-anonymous dataset. A data set is said to be k-anonymous if, given any entity in the data set, this entity is indistinguishable from at least k other entities. If this is done properly an attacker with unrestricted access to the database will not be able to identify any single entity. This is a powerful metric but it does not apply in a straight forward way when dealing with a graph instead of relational database because there is much implied in the structure latent in the graph. The details of what can be derived from this data is discussed in the previous two sections 3 and 3.3. In the remainder of this section many of these ideas will be re-hashed with anonymization or de-anonymization in mind.

One technique that translates directly from anonymizing a relational database to anonymizing a graph data-set is generalization. We see this technique applied in the PCO paper mentioned earlier [70]. Here the user tells the system to generalize their attributes based on who is requesting information, the time of day, and several other factors. This works fine when protecting data from a fellow user but it does not do any form of anonymization unless many users are using the same generalizations and their names (or labels) have been removed. With a system like this in place, an adversary with a more complete view of the social network can still perform a neighborhood attack.



(a) original

(b) anonymous

(c) 1-neighborhood of Ada

(d) privacy-preserved graph

Fig. 10: a: The original social graph. b: The social graph anonymized (names removed). c: 1-hop neighborhood graph of Ada. d: edge added to anonymize Ada. The graphs in d and b differ in only one edge. If this edge is not added to graph d then the 1-hop neighborhood graph of Ada and Bob would be different and Ada would be easily recognizable by an attacker with this knowledge.

Neighborhood attacks are addressed in [113] by authors Zhou and Pei. In a neighborhood attack an adversary has an anonymized copy of the social network. That is, a graph with edges and vertices but no labels on the vertices. The adversary also knows a small bit of information about the graph around the person they're trying to identify or locate in the graph. For example they might know the person in question has three neighbors (vertices connected directly by one edge) and that two of those neighbors are they, themselves neighbors. It is not hard to imagine a graph where the identity of this person might be obvious. The paper presents a novel technique for anonymizing the social network graph in such a way that it is immune to neighborhood attacks. An example of this can be seen in Fig. 10.

The technique works by first extracting the neighborhoods of the nodes we're anonymizing. Then, we group the nodes that have similar neighborhoods and incrementally add edges to the nodes of each group until we have achieved isomorphisms between the neighborhoods of these groups. In Fig. 10(d) we can see that an edge was added between Harry and Irene making Ada indistinguishable from Bob. They both have two pairs of friends. Because testing two graphs for isomorphism is an NP-hard problem, the researchers propose a novel coding technique so that isomorphisms can be recognized in polynomial time because of their encoding. Several more techniques for anonymization through edge addition are discussed in [39]. The authors of this paper show that all of the techniques in the literature are NP-hard. Fortunately, because social networks rarely contain nodes that have greater then a few thousand edges this is not a massive obstacle. It is however, a large obstacle.

A de-anonymization attack is outlined in [35] by researchers Henderson et al. In this paper the neighborhood attack is augmented with local node data. They assume the adversary has access to two types of data. First, local (node-based) features such as degree of a node, and, second, neighborhood (ego-based) data such as which nodes a node is connected to and which nodes those nodes are connected to. The system outputs the third type of data known as "regional features" which the authors claim exposes behavioral. This data can be thought of intuitively as what type of nodes the node is question is connected to (e.g. rich people, democrats, etc). This is fundamentally different from the node label and in some way is much more useful information. The only reason labels are valuable at all is because it is presumed the label can connect the entity to even more information. Indeed, identification is the act of associating information. This technique does rely on social influence or homophily and it is very high level and theoretical.

In search of a more realistic, easier to grasp attack we find [100] which details in a more pragmatic way how to attack or de-anonymize a social network. The authors of this paper assume the adversary knows that the user in question is a member of a certain group. The adversary can easily write a bit of javascript and html that can place a link to that group's page, or pages that are closely related to that group. If, when the user visits this adversarial code the links appear purple then the adversary knows that they exist in the user's browser history. This technique of determining what is in the client's browser history is known as a "web browser history stealing attack" or "history stealing." While this is not a theoretically strong attack it is pragmatic. To defend against this attack users need to simply clear their history regularly, not view content written by the adversary, or edit their browser configuration so visited links are not purple in order to thwart the attacker. However simple this may be, in practice users rarely behave in this way.

A third type of attack that is completely different from neighborhood attacks and the hybrid mentioned in [35] is the attribute disclosure attack like the one described in [17]. Here, as in [35], the authors recognize that the actual labels of the nodes is often unimportant and what attackers are really after is to be able to assign a set of attributes to a specific node in the network (as opposed to just the label). If this can be achieved the attributes of the node (such as node neighborhood etc) can be associated with original attribute set. This is called an attribute disclosure attack. The paper [17] presents an anonymization technique that hopes to protect against attribute disclosure attacks. Their technique is based on the intuition that node communities or neighborhoods need to be similar to the network as a whole in terms of their attributes. For example, a neighborhood of car enthusiasts or caffeine junkies stand out and are significantly different from the surrounding neighborhoods. This makes it easier for attackers to locate users in the graph. Names are often not unique, especially in an online social network. There is, however, likely only one or two car enthusiast groups in towns of moderate size. Using this information we can more easily locate users in a social network.

Moving on to more network anonymization techniques authors Gundecha, Barbier, and Liu show in [33] that users can protect themselves from network de-anonymization attacks. Although it is not stated explicitly, the idea of recognizing which of your friends or social connections is vulnerable can help protect the user herself. Vulnerability is somewhat contagious in this context. A wide open and interesting area of research is studying what users can do to recognize if they are susceptible to de-anonymization attacks and if their friends are giving them away. Although this paper does not address anonymization or de-anonymization specifically, it is quite clear that the concept of vulnerable friends making you yourself more vulnerable quite clearly translates over to this field.

With all of these sophisticated de-anonymization techniques and the lack of an easy way to determine if a graph is susceptible to them, authors Sala et al propose a novel technique for online social network graph anonymization in [74]. Rather than starting with the graph as-is and modifying it to make the data and the users anonymous they propose that a fake graph be generated the matches the real graph in various (graph, statistical, or other miscellaneous) metrics. Throughout the paper they discuss various ways of creating a graph that matches a second in certain metrics e.g. average degree. This approach brings to mind another interesting research opportunity describing a framework for how social networking sites can produce a second dummy graph that matches the real graph with the proper desired metrics. In this way, researchers can request data be formulated for their research and supply the social network with the metrics they intend on using in their research. This way there is very little threat of users being de-anonymized accidentally through the publishing of research paper or maliciously. Alternatively, researchers might request a set of metrics from the online social network, then use those metrics to create the dummy data set themselves. In either case, sensitive information cannot be leaked regardless of what the researchers do because the information in question never leaves the hands of the OSN provider.

7 Conclusion

Privacy and security are important topics in many areas of computer science. They are of particular interest specifically in the area of online social networks because of the sensitive data involved. Never before has there been a single collection of personal, identifiable, sensitive, and volunteered data like we have now with online social networks. The convergence of this data is extremely dangerous. For example, hometown and birth-date are all that is necessary to determine one's social security number with more than reasonable accuracy and these are often both readily available on a user's profile in an online social network.

The bulk of this survey paper is a careful compilation of recent research that tries to protect user data in section 2 Firstly, we discuss the different types of vantage points from which information can be accessed. The broadest threat, other users, is discussed in detail in section 2.1, Protection from Other Users. The remaining vantage points: protection from OSN applications, protection from advertising agencies, and protection from the actual online social network provider themselves show that once user data has been publish in an online social network it is susceptible to a wide array of attackers. Various literature shows techniques ranging from elaborate encryption schemes to architecture changes. User location is an issue special enough to garner it's own sub section in the paper because it introduces the threat of physical harm. As a result, users and researchers alike consider location to be much more sensitive information.

Network inference and it's effects are discussed in section 3. Here we are reminded that much can be deduced about a person given a limited set of social graph information. Particularly, we see schemes in which researchers are able to deduce political views, user interests, and positive or negative emotions between based on small subsets of the social graph. Network inference marks a turning point in this survey from solution type research to problem type research. Locating hubs or top effectors, in section 3.2 is a special case of network inference. This research is important and relevant because of the continuing trend of viral marketing. Viral marketing is especially prevalent in social networking and it is at the users' expense. Link prediction, another special case of network inference where future attributes are inferred, is discussed in section 3.3. Here we see researchers predict new friendships, new purchases, and new conversation lengths between users. The key difference between network inference and link prediction is that network inference determines the current state of some unknown attribute in the social graph. Link prediction, is determines the future state of currently known attributes.

Online social network providers must do more than produce a service one time and then allow users access. Rather they must maintain the proper operation of that service over time. This theme is discussed in section 5 beginning with the notion of trust in an online social network. We see from the literature that users in a social network trust one another; often times, better than those users might trust the actual OSN provider. This trust can be leveraged to remove sybil accounts and filter spam messages from the network. Sybil accounts are discussed next in more detail. We see a number of papers with schemes based on the small cut between sybil accounts and other communities in an OSN. This comprises most of the research in maintaining OSN operation. What is remaining is web 2.0 security in general, because online social networks today are almost always web based. We see several schemes that attempt to measure the vulnerability of a given web application and one survey paper that compares various black box web vulnerability testing suites.

Social network providers need to anonymize their social networks for research and development both in academia and in industry. In the final section of this survey, section 6, we see that this raises many technical challenges due in part to the sophisticated network inference and link prediction techniques discussed in the proceeding sections.

References

[1] S.M.A. Abbas, J.A. Pouwelse, D.H.J. Epema, and H.J. Sips. A gossip-based distributed social networking system. In *Enabling Technologies: Infrastructures for Collaborative Enterprises, 2009. WETICE '09. 18th IEEE International Workshops on*, pages 93 –98, 29 2009-july 1 2009.

[2] Alessandro Acquisti and Ralph Gross. Imagined communities: Awareness, information sharing, and privacy on the facebook. In *Privacy Enhancing Technologies*, pages 36–58, 2006.

[3] Alessandro Acquisti and Ralph Gross. Predicting social security numbers from public data. *Proceedings of the National Academy of Sciences*, 106 (27):10975–10980, 2009.

[4] Jonathan Anderson, Claudia Diaz, Joseph Bonneau, and Frank Stajano. Privacy-enabling social networking over untrusted networks. In *Proceedings of the 2nd ACM workshop on Online social networks*, WOSN '09, pages 1–6, New York, NY, USA, 2009. ACM.

[5] Michael Backes, Matteo Maffei, and Kim Pecina. A security API for distributed social networks. In *NDSS*, 2011.

[6] Randy Baden, Adam Bender, Neil Spring, Bobby Bhattacharjee, and Daniel Starin. Persona: an online social network with user-defined privacy. In *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*, SIGCOMM '09, pages 135–146, New York, NY, USA, 2009. ACM.

[7] W. Bamberger, J. Schlittenlacher, and K. Diepold. A trust model for intervehicular communication based on belief theory. In *Social Computing (SocialCom), 2010 IEEE Second International Conference on*, pages 73 –80, aug. 2010.

[8] G. Barbian. Assessing trust by disclosure in online social networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 163 –170, july 2011.

[9] Jason Bau, Elie Bursztein, Divij Gupta, and John Mitchell. State of the art: Automated black-box web application vulnerability testing. In *Security and Privacy (SP), 2010 IEEE Symposium on*, pages 332 –345, may 2010.

[10] María-Victoria Belmonte, Manuel Díaz, and Ana Reyna. Coalitions and incentives for content distribution over a secure peer-to-peer middleware. In *Proc. of the third international conference on advances in P2P systems. AP2PS 2011. November 20-25, 2011. Lisbon, Portugal*, pages 71–78. IARIA, December 2011.

[11] Leyla Bilge, Thorsten Strufe, Davide Balzarotti, and Engin Kirda. All your contacts are belong to us: automated identity theft attacks on social networks. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 551–560, New York, NY, USA, 2009. ACM.

[12] Angela Bonifati, Hui (Wendy) Wang, and Ruilin Liu. Spac: a distributed, peer-to-peer, secure and privacy-aware social space. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, pages 1953–1954, New York, NY, USA, 2010. ACM.

[13] Joseph Bonneau, Jonathan Anderson, Ross Anderson, and Frank Stajano. Eight friends are enough: social graph approximation via public listings. In *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems*, SNS '09, pages 13–18, New York, NY, USA, 2009. ACM.

[14] J.A. Calandrino, A. Kilzer, A. Narayanan, E.W. Felten, and V. Shmatikov. "you might also like:" privacy risks of collaborative filtering. In *Security and Privacy (SP), 2011 IEEE Symposium on*, pages 231 –246, may 2011.

[15] Ruichuan Chen, Eng Keong Lua, and Zhuhua Cai. Bring order to online social networks. In *INFOCOM, 2011 Proceedings IEEE*, pages 541 –545, april 2011.

[16] Wei Chen, Chi Wang, and Yajun Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 1029–1038, New York, NY, USA, 2010. ACM.

[17] S. Chester and G. Srivastava. Social network privacy for attribute disclosure attacks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 445 –449, july 2011.

[18] Freddy Chong Tat Chua and Ee-Peng Lim. Trust network inference for online rating data using generative models. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 889–898, New York, NY, USA, 2010. ACM.

[19] Landon P. Cox, Angela Dalton, and Varun Marupadi. Smokescreen: flexible privacy controls for presence-sharing. In *Proceedings of the 5th international conference on Mobile systems, applications and services*, MobiSys '07, pages 233–245, New York, NY, USA, 2007. ACM.

[20] Justin Cranshaw, Eran Toch, Jason Hong, Aniket Kittur, and Norman Sadeh. Bridging the gap between physical location and online social networks. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, Ubicomp '10, pages 119–128, New York, NY, USA, 2010. ACM.

[21] L.A. Cutillo, R. Molva, and T. Strufe. Safebook: A privacy-preserving online social network leveraging on real-life trust. *Communications Magazine, IEEE*, 47(12):94 –101, dec. 2009.

[22] Munmun De Choudhury, Winter A. Mason, Jake M. Hofman, and Duncan J. Watts. Inferring relevant social networks from interpersonal communication. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 301–310, New York, NY, USA, 2010. ACM.

[23] Lujun Fang and Kristen LeFevre. Privacy wizards for social networking sites. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 351–360, New York, NY, USA, 2010. ACM.

[24] Adrienne Felt and David Evans. Privacy protection for social networking platforms. In *Web 2.0 Security and Privacy 2008 held in conjuction with the 2008 IEEE Symposium on Security and Privacy*. IEEE, 2008.

[25] P.W.L. Fong. Preventing sybil attacks by privilege attenuation: A design principle for social network systems. In *Security and Privacy (SP), 2011 IEEE Symposium on*, pages 263 –278, may 2011.

[26] M. Fredrikson and B. Livshits. Repriv: Reimagining content personalization and in-browser privacy. In *Security and Privacy (SP), 2011 IEEE Symposium on*, pages 131 –146, may 2011.

[27] Wojciech Galuba, Karl Aberer, Dipanjan Chakraborty, Zoran Despotovic, and Wolfgang Kellerer. Outtweeting the twitterers - predicting information cascades in microblogs. In *Proceedings of the 3rd conference on Online social networks*, WOSN'10, Berkeley, CA, USA, 2010. USENIX Association.

[28] Hongyu Gao, Jun Hu, Tuo Huang, Jingnan Wang, and Yan Chen. Security issues in online social networks. *IEEE Internet Computing*, 15(4):56–63, 2011.

[29] Chris Grier, Kurt Thomas, Vern Paxson, and Michael Zhang. @spam: the underground on 140 characters or less. In *Proceedings of the 17th ACM conference on Computer and communications security*, CCS '10, pages 27–37, New York, NY, USA, 2010. ACM.

[30] Saikat Guha, Bin Cheng, and Paul Francis. Privad: practical privacy in online advertising. In *Proceedings of the 8th USENIX conference on Networked systems design and implementation*, NSDI'11, pages 13–13, Berkeley, CA, USA, 2011. USENIX Association.

[31] Saikat Guha, Kevin Tang, and Paul Francis. Noyb: privacy in online social networks. In *Proceedings of the first workshop on Online social networks*, WOSN '08, pages 49–54, New York, NY, USA, 2008. ACM.

[32] Dolvara Gunatilaka. A survey of privacy and seucrity issues in social networks, November 2011.

[33] Pritam Gundecha, Geoffrey Barbier, and Huan Liu. Exploiting vulnerability to secure user privacy on a social networking site. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 511–519, New York, NY, USA, 2011. ACM.

[34] S. Hameed, Xiaoming Fu, Pan Hui, and N. Sastry. Lens: Leveraging social networking and trust to prevent spam transmission. In *Network Protocols (ICNP), 2011 19th IEEE International Conference on*, pages 13 –18, oct. 2011.

[35] Keith Henderson, Brian Gallagher, Lei Li, Leman Akoglu, Tina Eliassi-Rad, Hanghang Tong, and Christos Faloutsos. It's who you know: graph mining using recursive structural features. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 663–671, New York, NY, USA, 2011. ACM.

[36] M.U. Ilyas, M.Z. Shafiq, A.X. Liu, and H. Radha. A distributed and privacy preserving algorithm for identifying information hubs in social networks. In *IEEE INFOCOM 2011 Proceedings*, pages 561 – 565, April 2011.

[37] Tomas Isdal, Michael Piatek, Arvind Krishnamurthy, and Thomas Anderson. Privacy-preserving p2p data sharing with oneswarm. In *Proceedings of the ACM SIGCOMM 2010 conference on SIGCOMM*, SIGCOMM '10, pages 111–122, New York, NY, USA, 2010. ACM.

[38] P. Joshi and C.-C.J. Kuo. Security and privacy in online social networks: A survey. In *Multimedia and Expo (ICME), 2011 IEEE International Conference on*, pages 1 –6, july 2011.

[39] B. Kapron, G. Srivastava, and S. Venkatesh. Social network anonymization via edge addition. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 155 –162, july 2011.

[40] Vassilis Kostakos, Jayant Venkatanathan, Bernardo Reynolds, Norman Sadeh, Eran Toch, Siraj A. Shaikh, and Simon Jones. Who's your best friend?: targeted privacy attacks in location-sharing social networks. In *Proceedings of the 13th international conference on Ubiquitous computing*, UbiComp '11, pages 177–186, New York, NY, USA, 2011. ACM.

[41] Balachander Krishnamurthy, Phillipa Gill, and Martin Arlitt. A few chirps about twitter. In *Pro-*

ceedings of the first workshop on Online social networks, WOSN '08, pages 19–24, New York, NY, USA, 2008. ACM.

[42] Timothy La Fond and Jennifer Neville. Randomization tests for distinguishing social influence and homophily effects. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 601–610, New York, NY, USA, 2010. ACM.

[43] Theodoros Lappas, Evimaria Terzi, Dimitrios Gunopulos, and Heikki Mannila. Finding effectors in social networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 1059–1068, New York, NY, USA, 2010. ACM.

[44] Ruaylong Lee, R. Nia, J. Hsu, K.N. Levitt, J. Rowe, S.F. Wu, and Shaozhi Ye. Design and implementation of faith, an experimental system to intercept and manipulate online social informatics. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 195 –202, july 2011.

[45] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. Predicting positive and negative links in online social networks. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 641–650, New York, NY, USA, 2010. ACM.

[46] Jure Leskovec, Kevin J. Lang, and Michael Mahoney. Empirical comparison of algorithms for network community detection. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 631–640, New York, NY, USA, 2010. ACM.

[47] Chris Lesniewski-Laas and M. Frans Kaashoek. Whanau: a sybil-proof distributed hash table. In *Proceedings of the 7th USENIX conference on Networked systems design and implementation*, NSDI'10, Berkeley, CA, USA, 2010. USENIX Association.

[48] Heather Richter Lipford, Andrew Besmer, and Jason Watson. Understanding privacy settings in facebook with an audience view. In *Proceedings of the 1st Conference on Usability, Psychology, and Security*, pages 2:1–2:8, Berkeley, CA, USA, 2008. USENIX Association.

[49] Bin Liu, P. Terlecky, A. Bar-Noy, R. Govindan, and M.J. Neely. Optimizing information credibility in social swarming applications. In *INFOCOM, 2011 Proceedings IEEE*, pages 556 –560, april 2011.

[50] Mike Ter Louw, Karthik Thotta Ganesh, and V. N. Venkatakrishnan. Adjail: practical enforcement of confidentiality and integrity policies on web advertisements. In *Proceedings of the 19th USENIX conference on Security*, USENIX Security'10, pages 24–24, Berkeley, CA, USA, 2010. USENIX Association.

[51] Tom Lovett, Eamonn O'Neill, James Irwin, and David Pollington. The calendar as a sensor: analysis and improvement using data fusion with social networks and location. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, Ubicomp '10, pages 3–12, New York, NY, USA, 2010. ACM.

[52] Matthew M. Lucas and Nikita Borisov. Flybynight: mitigating the privacy risks of social networking. In *Proceedings of the 7th ACM workshop on Privacy in the electronic society*, WPES '08, pages 1–8, New York, NY, USA, 2008. ACM.

[53] Wanying Luo, Qi Xie, and U. Hengartner. Facecloak: An architecture for user privacy on social networking sites. In *Computational Science and Engineering, 2009. CSE '09. International Conference on*, volume 3, pages 26 –33, aug. 2009.

[54] K. Macropol and A. Singh. Content-based modeling and prediction of information dissemination. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 21 –28, july 2011.

[55] M. Magnani and L. Rossi. The ml-model for multilayer social networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 5 –12, july 2011.

[56] Huina Mao, Xin Shuai, and Apu Kapadia. Loose tweets: an analysis of privacy leaks on twitter. In *Proceedings of the 10th annual ACM workshop on Privacy in the electronic society*, WPES '11, pages 1–12, New York, NY, USA, 2011. ACM.

[57] Brendan Meeder, Brian Karrer, Amin Sayedi, R. Ravi, Christian Borgs, and Jennifer Chayes. We know who you followed last summer: inferring social link creation times in twitter. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 517–526, New York, NY, USA, 2011. ACM.

[58] Brendan Meeder, Patrick Gage Kelley, Jennifer Tam, and Lorrie Faith Cranor. Rt @iwantprivacy: Widespread violation of privacy settings in the twitter social network. In *Web 2.0 Security & Privacy*, W2SP, 2010.

[59] F. Ming, Fai Wong, and P. Marbach. Who are your friends? a simple mechanism that achieves perfect network formation. In *INFOCOM, 2011 Proceedings IEEE*, pages 566 –570, april 2011.

[60] Alan Mislove, Ansley Post, Peter Druschel, and Krishna P. Gummadi. Ostra: leveraging trust to thwart unwanted communication. In *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, NSDI'08, pages 15–30, Berkeley, CA, USA, 2008. USENIX Association.

[61] Arvind Narayanan, Narendran Thiagarajan, Mugdha Lakhani, Michael Hamburg, and Dan Boneh. Location privacy via private proximity testing. In *NDSS*, 2011.

[62] Nam P. Nguyen, Thang N. Dinh, Ying Xuan, and My T. Thai. Adaptive algorithms for detecting community structure in dynamic social networks. In *INFOCOM*, pages 2282–2290, 2011.

[63] Soeren Preibusch Nicola Jentzsch and Andreas Harasser. Study on monetising privacy. Technical report, enisa, European Network and Information Agency, february 2012.

[64] Sai Teja Peddinti and Nitesh Saxena. On the limitations of query obfuscation techniques for location privacy. In *Proceedings of the 13th international conference on Ubiquitous computing*, UbiComp '11, pages 187–196, New York, NY, USA, 2011. ACM.

[65] Marco Pennacchiotti and Ana-Maria Popescu. Democrats, republicans and starbucks afficionados: user classification in twitter. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 430–438, New York, NY, USA, 2011. ACM.

[66] Ansley Post, Vijit Shah, and Alan Mislove. Bazaar: strengthening user reputations in online marketplaces. In *Proceedings of the 8th USENIX conference on Networked systems design and implementation*, NSDI'11, Berkeley, CA, USA, 2011. USENIX Association.

[67] Alexei Pozdnoukhov and Christian Kaiser. Spacetime dynamics of topics in streaming text. In *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Location-Based Social Networks*, LBSN '11, pages 8:1–8:8, New York, NY, USA, 2011. ACM.

[68] Krishna P. N. Puttaswamy and Ben Y. Zhao. Preserving privacy in location-based mobile social applications. In *Proceedings of the Eleventh Workshop on Mobile Computing Systems &#38; Applications*, HotMobile '10, pages 1–6, New York, NY, USA, 2010. ACM.

[69] Nader Mohamed Racha Ajami, Noha Ramadan and Jameela Al-Jaroodi. Security challenges and approaches in online social networks: A survey. *IJCNS International Journal of Computer Science and Network Security*, 11(8), 2011.

[70] F. Rahman, M.E. Hoque, F.A. Kawsar, and S.I. Ahamed. Preserve your privacy with pco: A privacy sensitive architecture for context obfuscation for pervasive e-community based applications. In *Social Computing (SocialCom), 2010 IEEE Second International Conference on*, pages 41 –48, aug. 2010.

[71] Alexey Reznichenko, Saikat Guha, and Paul Francis. Auctions in do-not-track compliant internet advertising. In *Proceedings of the 18th ACM conference on Computer and communications security*, CCS '11, pages 667–676, New York, NY, USA, 2011. ACM.

[72] Maayan Roth, Assaf Ben-David, David Deutscher, Guy Flysher, Ilan Horn, Ari Leichtberg, Naty Leiser, Yossi Matias, and Ron Merom. Suggesting friends using the implicit social graph. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 233–242, New York, NY, USA, 2010. ACM.

[73] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 851–860, New York, NY, USA, 2010. ACM.

[74] Alessandra Sala, Lili Cao, Christo Wilson, Robert Zablit, Haitao Zheng, and Ben Y. Zhao. Measurement-calibrated graph models for social network experiments. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 861–870, New York, NY, USA, 2010. ACM.

[75] D. Sculley, Matthew Eric Otey, Michael Pohl, Bridget Spitznagel, John Hainsworth, and Yunkai Zhou. Detecting adversarial advertisements in the wild. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 274–282, New York, NY, USA, 2011. ACM.

[76] B. Sharifi, M.-A. Hutton, and J.K. Kalita. Experiments in microblog summarization. In *Social Computing (SocialCom), 2010 IEEE Second International Conference on*, pages 49 –56, aug. 2010.

[77] R. Shokri, G. Theodorakopoulos, J. Le Boudec, and J. Hubaux. Quantifying location privacy. In *Security and Privacy (SP), 2011 IEEE Symposium on*, pages 247 –262, may 2011.

[78] Kapil Singh, Sumeer Bhola, and Wenke Lee. xbook: redesigning privacy control in social networking platforms. In *Proceedings of the 18th conference on USENIX security symposium*, SSYM'09, pages 249–266, Berkeley, CA, USA, 2009. USENIX Association.

[79] M. Sirivianos, Kyungbaek Kim, and Xiaowei Yang. Socialfilter: Introducing social trust to collaborative spam mitigation. In *INFOCOM, 2011 Proceedings IEEE*, pages 2300 –2308, april 2011.

[80] A. Sridharan, Yong Gao, Kui Wu, and J. Nastos. Statistical behavior of embeddedness and communities of overlapping cliques in online social networks. In *INFOCOM, 2011 Proceedings IEEE*, pages 546 –550, april 2011.

[81] Sid Stamm, Brandon Sterne, and Gervase Markham. Reining in the web with content security policy. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 921–930, New York, NY, USA, 2010. ACM.

[82] Fred Stutzman and Jacob Kramer-Duffield. Friends only: examining a privacy-enhancing behavior in facebook. In *Proceedings of the 28th international conference on Human factors in computing systems*, CHI '10, pages 1553–1562, New York, NY, USA, 2010. ACM.

[83] Bob Sullivan. Survey: Consumers say id theft down 28%, February 2011.

[84] Jinyuan Sun, Xiaoyan Zhu, and Yuguang Fang. A privacy-preserving scheme for online social networks with efficient revocation. In *INFOCOM, 2010 Proceedings IEEE*.

[85] Gayatri Swamynathan, Christo Wilson, Bryce Boe, Kevin Almeroth, and Ben Y. Zhao. Do social networks improve e-commerce?: a study on social marketplaces. In *Proceedings of the first workshop on Online social networks*, WOSN '08, pages 1–6, New York, NY, USA, 2008. ACM.

[86] Karen P. Tang, Jason I. Hong, and Daniel P. Siewiorek. Understanding how visual representations of location feeds affect end-user privacy concerns. In *Proceedings of the 13th international conference on Ubiquitous computing*, UbiComp '11, pages 207–216, New York, NY, USA, 2011. ACM.

[87] Karen P. Tang, Jialiu Lin, Jason I. Hong, Daniel P. Siewiorek, and Norman Sadeh. Rethinking location sharing: exploring the implications of social-driven vs. purpose-driven location sharing. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, Ubicomp '10, pages 85–94, New York, NY, USA, 2010. ACM.

[88] Shaojie Tang, Jing Yuan, Xufei Mao, Xiang-Yang Li, Wei Chen, and Guojun Dai. Relationship classification in large scale online social networks and its impact on information propagation. In *INFOCOM, 2011 Proceedings IEEE*, pages 2291 –2299, april 2011.

[89] M.A. Tayebi, L. Bakker, U. Glasser, and V. Dabbaghian. Locating central actors in co-offending networks. In *2011 International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 171 –179, July 2011.

[90] Eran Toch, Justin Cranshaw, Paul Hankes Drielsma, Janice Y. Tsai, Patrick Gage Kelley, James Springfield, Lorrie Cranor, Jason Hong, and Norman Sadeh. Empirical models of privacy in location sharing. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, Ubicomp '10, pages 129–138, New York, NY, USA, 2010. ACM.

[91] Amin Tootoonchian, Stefan Saroiu, Yashar Ganjali, and Alec Wolman. Lockr: better privacy for social networks. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, CoNEXT '09, pages 169–180, New York, NY, USA, 2009. ACM.

[92] Nguyen Tran, Bonan Min, Jinyang Li, and Lakshminarayanan Subramanian. Sybil-resilient online content voting. In *Proceedings of the 6th USENIX symposium on Networked systems design and implementation*, NSDI'09, pages 15–28, Berkeley, CA, USA, 2009. USENIX Association.

[93] P. Turek, A. Wierzbicki, R. Nielek, A. Hupa, and A. Datta. Learning about the quality of teamwork from wikiteams. In *Social Computing (SocialCom), 2010 IEEE Second International Conference on*, pages 17 –24, aug. 2010.

[94] K. Vikram, Abhishek Prateek, and Benjamin Livshits. Ripley: automatically securing web 2.0 applications through replicated execution. In *Proceedings of the 16th ACM conference on Computer and communications security*, CCS '09, pages 173–186, New York, NY, USA, 2009. ACM.

[95] Bimal Viswanath, Mainack Mondal, Krishna P. Gummadi, Alan Mislove, and Ansley Post. Canal: Scaling social network-based Sybil tolerance schemes. In *Proceedings of the 7th European Conference on Computer Systems (EuroSys'12)*, Bern, Switzerland, April 2012.

[96] Bimal Viswanath, Ansley Post, Krishna P. Gummadi, and Alan Mislove. An analysis of social network-based sybil defenses. In *Proceedings of the ACM SIGCOMM 2010 conference*, pages 363–374, New York, NY, USA, 2010. ACM.

[97] Wei Wei, Fenguan Xu, Chiu C. Tan, and Qun Li. Sybildefender: Defend against sybil attacks in large social networks. In *2012 Proceedings of IEEE Infocom.*

[98] Wei Wei, Fengyuan Xu, and Qun Li. Mobishare: Flexible privacy-preserving location sharing in mobile online social networks. In *IEEE Infocom mini-conference*, Orlando, FL, March 2012.

[99] Zhen Wen and Ching-Yung Lin. On the quality of inferring interests from social neighbors. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 373–382, New York, NY, USA, 2010. ACM.

[100] Gilbert Wondracek, Thorsten Holz, Engin Kirda, and Christopher Kruegel. A practical attack to de-anonymize social network users. In *Proceedings of the 2010 IEEE Symposium on Security and Privacy*, SP '10, pages 223–238, Washington, DC, USA, 2010. IEEE Computer Society.

[101] Shaomei Wu, Jake M. Hofman, Winter A. Mason, and Duncan J. Watts. Who says what to whom on twitter. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 705–714, New York, NY, USA, 2011. ACM.

[102] Rongjing Xiang, Jennifer Neville, and Monica Rogati. Modeling relationship strength in online social networks. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 981–990, New York, NY, USA, 2010. ACM.

[103] Xinyu Xing, Yu-Li Liang, Hanqiang Cheng, Jianxun Dang, Sui Huang, Richard Han, Xue Liu, Qin Lv, and Shivakant Mishra. Safevchat: detecting obscene content and misbehaving users in online video chat services. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 685–694, New York, NY, USA, 2011. ACM.

[104] Wei Xu, Fangfang Zhang, and Sencun Zhu. Toward worm detection in online social networks. In *Proceedings of the 26th Annual Computer Security Applications Conference*, ACSAC '10, pages 11–20, New York, NY, USA, 2010. ACM.

[105] Hsin-Chang Yang and Chung-Hong Lee. Post-level spam detection for social bookmarking web sites. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 180 –185, july 2011.

[106] Shuang-Hong Yang, Bo Long, Alex Smola, Narayanan Sadagopan, Zhaohui Zheng, and Hongyuan Zha. Like like alike: joint friendship and interest propagation in social networks. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 537–546, New York, NY, USA, 2011. ACM.

[107] Xiwang Yang, Yang Guo, and Yong Liu. Bayesian-inference based recommendation in online social networks. In *INFOCOM, 2011 Proceedings IEEE*, pages 551 –555, april 2011.

[108] Zhimin Yang, Boying Zhang, Jiangpeng Dai, A.C. Champion, Dong Xuan, and Du Li. E-smalltalker: A distributed mobile system for social networking in physical proximity. In *Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on*, pages 468 –477, june 2010.

[109] Mao Ye, Dong Shou, Wang-Chien Lee, Peifeng Yin, and Krzysztof Janowicz. On the semantic annotation of places in location-based social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 520–528, New York, NY, USA, 2011. ACM.

[110] Haifeng Yu, P.B. Gibbons, M. Kaminsky, and Feng Xiao. Sybillimit: A near-optimal social network defense against sybil attacks. In *IEEE Symposium on Security and Privacy, 2008.*, pages 3 –17, May 2008.

[111] Haifeng Yu, Michael Kaminsky, Phillip B. Gibbons, and Abraham Flaxman. Sybilguard: defending against sybil attacks via social networks. In

Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications, SIGCOMM '06, pages 267–278, New York, NY, USA, 2006. ACM.

[112] Chi Zhang, Jinyuan Sun, Xiaoyan Zhu, and Yuguang Fang. Privacy and security for online social networks: challenges and opportunities. *Network, IEEE*, 24(4):13 –18, july-august 2010.

[113] Bin Zhou and Jian Pei. Preserving privacy in social networks against neighborhood attacks. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 506 –515, april 2008.