SEVENTH FRAMEWORK PROGRAMME

Information & Communication Technologies
Trustworthy ICT

NETWORK OF EXCELLENCE

A European Network of Excellence in Managing Threats and Vulnerabilities in the Future Internet: *Europe for the World* [†]

Deliverable D5.2: Preliminary Report on Social Networks Security

Abstract: This deliverable discusses security aspects in social networks and how they influence the privacy of participating individuals.

Contractual Date of Delivery	February 2012
Actual Date of Delivery	05.03.2012
Deliverable Dissemination Level	Public
Editor	Christian Platzer
Contributors	All *SysSec* partners
Quality Assurance	Antonis Papadogiannakis, Davide Balzarotti, Evangelos Markatos, Christian Platzer

The *SysSec* consortium consists of:

FORTH-ICS	Coordinator	Greece
Politecnico Di Milano	Principal Contractor	Italy
Vrije Universiteit Amsterdam	Principal Contractor	The Netherlands
Institut Eurécom	Principal Contractor	France
IICT-BAS	Principal Contractor	Bulgaria
Technical University of Vienna	Principal Contractor	Austria
Chalmers University	Principal Contractor	Sweden
TUBITAK-BILGEM	Principal Contractor	Turkey

[†] The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 257007.

Contents

List of Figures

1 Introduction

The purpose of this deliverable is to give an overview on current security issues in social networks. Some of them are well-known and have been dealt with by the research community while others are relatively new. Before immersing into the various subjects, however, it is imperative to define the scope of a *social network* and the semantic behind this term. When referring to social networks in a common sense, it boils down to a single name: *Facebook*. With over 800 million users [10] in December 2011, it is the largest, most widely accepted social network so far. However, it is not the goal of this document to discuss a single platform and its possible weaknesses. Instead, the concept of a social network is discussed in a broader sense with the participants in its center and the surrounding technology as an enabler.

With the ubiquitous availability of internet connectivity, the tendency of human beings to share certain facets of their personal life has led to various technology platforms supporting this impulse. Naturally, *pure* social networks like Facebook and its predecessors are very good examples and can be used as a reference for most case studies. There are, however, various other platforms to consider. A good example are gaming platforms like Steam [66], Origin [68] or BattleNet [65] where users interact, share their latest achievements or simply chat with each other. Other networks such as LinkedIn or Xing focus on more professional participants to help them establish business relationships and maintain them. In fact, a lot of communities reaching from the aforementioned gaming to research communities, already established their own social network to help likeminded individuals to keep in touch.

Finally, there are artificial societies as they exist in massive multiplayer online games (MMOG). What all of these platforms have in common is the fact that they rely on their user's social interactions to function. They only differ in the validity of the presented persona and, from an attacker's point

of view, the asset connected with the person behind that persona. That can be a real name and personal information on Facebook, credit card information on gaming platforms or in-game currency in an MMOG. Security researchers aim to protect those assets by devising new protection mechanisms or identifying previously unseen threats. This task is not always simple and, due to the unpredictable nature of humans and their actions, often challenging.

The key concepts covered in this documents are new forms of attacks against social networks and their user bases. In Chapter 2, a new form of social engineering, especially tailored to social networks, is discussed. One of the culprits when enabling attacks or exploits on a social network is an inadequate authentication method. This is true for traditional, Facebook-like networks as well as gaming platforms. To amend this shortcoming, a novel form of authentication is discussed in Chapter 3. Instead of relying on mechanisms like passwords and security questions, the idea is to directly leverage personal information to authenticate individuals. From a technological perspective, Chapter 4 covers spread functionality of social networks on other sites on the internet. While adding functionality, such features are very prone to badly influence the user's privacy. Efficiently harvesting data from social networks is significant for large-scale studies and in the field of digital forensics. Chapter 5 shows novel techniques to obtain *Social snapshots*, which contain a user's profile information and associated meta-data. From an attacker's perspective, that is the most essential information as it enables targeted exploits like spear fishing and social engineering. Finally, Chapters 6 and 7 deal with uncommon forms of social networks and the human factors involved in operating them. Still, they utilize the same basic concepts and are therefore prone to the same attacks and weaknesses that influence the more common form like Facebook or Google+. Summed up, this document depicts the gross direction of security threats in social networks and how they can be handled presently and in the future.

2

Social engineering attacks in social networks

The large amount of information published, and often publicly shared, by users on their online social network profiles is increasingly attracting the attention of attackers. Attacks on social networks are usually variants of traditional security threats (such as malware, worms, spam, and phishing). These "common" threats are thoroughly discussed in other research papers. The one thing these attacks have in common when used in junction with social networks is their possibility to leverage personal data for a higher impact. Spam, for example, can be directly sent to an interested person, probably with the name of a friend as the sender [107]. Worms and other malware have a higher infection rate because links within a social network are more likely to be clicked [47]. Phishing attacks can be aimed at a narrow category of individuals wit a higher success rate as traditional spam [8]. These attacks are carried out in a different context by leveraging the social networks as a new medium to reach the victims. Moreover, adversaries can take advantage of the trust relationships between "friends" in social networks to craft more convincing attacks by exploiting personal information gleaned from victims' pages. Therefore, most of the attack requires, as a first step, to become friend of the victim. As already mentioned in the introduction, that applies to almost any form of social networks as long as they support some form of "friendship".

In fact, past research has shown that users of online social networks tend to exhibit a higher degree of trust in friend requests and messages sent by other users (e.g., [23, 34]). However, to date, *reverse social engineering* attacks in social networks have not received any attention. Hence, no previous work exists on the topic.

In a reverse social engineering attack, the attacker does not initiate contact with the victim. Rather, the victim is tricked into contacting the attacker herself. As a result, a high degree of trust is established between the victim and the attacker as the victim is the entity that first wanted to establish

a relationship. Once a reverse social engineering attack is successful (i.e., the attacker has established a friend relationship with the victim), she can then launch a wide range of attacks such as persuading victims to click on malicious links, blackmailing, identity theft, and phishing.

In this chapter we present the first user study on how attackers can abuse some of the features provided by online social networks with the aim of launching automated reverse social engineering attacks. In particular, we present three novel attacks, namely, recommendation-based, visitor tracking-based, and demographics-based reverse social engineering. Furthermore, using the popular social networks Facebook, Badoo, and Friendster, we discuss and measure the effectiveness of these attacks, and we show which social networking features make such attacks feasible in practice.

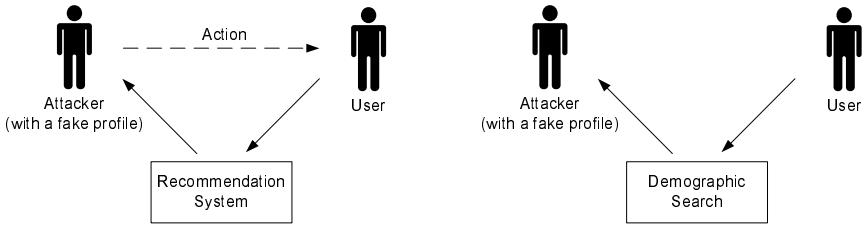
In the recommendation attack, the aim is to exploit the friend recommendations made by the social network to promote the fake profile of a fictitious user to the victim. The hope, from the attacker's point of view, is that the victim will be intrigued by the recommendation, and will attempt to contact the bogus profile that is under the attacker's control. In the visitor tracking attack, the aim is to trigger the target's curiosity by simply browsing her profile page. The notification that the page has been visited may be enough to attract the target to visit the attacker profile. Finally, in the demographic-based attack scenario, the attacker attempts to reach his victims by forging fake demographic or personal information with the aim of attracting the attention of users with similar preferences (e.g., similar musical tastes, similar interests, etc.).

Our findings suggest that, contrary to the common folk wisdom, only having an account with an attractive photograph may not be enough to recruit a high number of unsuspecting victims. Rather, the attacker needs to provide victims with a pretext and an incentive for establishing contact.

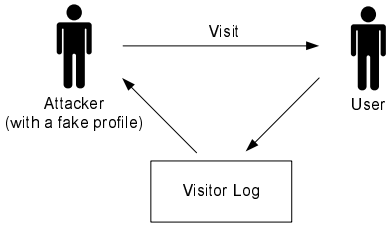
2.1 Reverse Social Engineering in Social Networks

Online social engineering attacks are easy to propagate, difficult to trace back to the attacker, and usually involve a low cost per targeted user. They are well-known threats in which the attacker aims at influencing the victims, and making them perform actions on her behalf. The attacker is typically interested in tricking the victims into revealing sensitive or important information. Examples of these attacks include traditional e-mail hoaxes and phishing, or their more advanced targeted forms, such as spear phishing.

Most online social engineering attacks rely on some form of "pretexting" [94]. That is, the attacker establishes contact with the target, and sends some initial request to bootstrap the attack. This approach, although effective because it can reach a large number of potential victims, has the downside that Internet users are becoming more and more suspicious about



(a) Recommendation Systems (b) Demographic Search



(c) Visitor Tracking

Figure 2.1: Different types of Reverse Social Engineering attacks.

unsolicited contact requests. However, previous work has shown that it is possible to raise levels of trust by impersonating an existing friend of the target (e.g., [34, 73]) or by injecting the attack into existing chat conversations [85].

Reverse Social Engineering (RSE) is a form of social engineering attack that has not yet been reported widely in an online context. RSE is a well-known technique in the hacker community (e.g., [94]) for targeted phone attacks. The attack, in a first step, relies on some form of "baiting" to stimulate the victim's curiosity. In a second step, once the victim's interest is raised, the attacker waits for the victim to make the initial approach and initiate contact. An RSE attack usually requires the attacker to create a persona that would seem attractive to the victim and that would encourage the victim to establish contact. For example, directly calling users and asking them for their passwords on the phone might raise suspicion in some users. In the reverse social engineering version of the same attack, a phone number can be e-mailed to the targets a couple of days in advance by spoofing an e-mail from the system administrator. The e-mail may instruct the users to call this number in case of problems. In this example, any victim who calls the phone number would probably be less suspicious and more willing to share information as she has initiated the first contact.

RSE attacks are especially attractive for online social networks. First, from an attacker's point of view, there is a good potential to reach millions of registered users in this new social setting. Second, RSE has the advantage that it can bypass current behavioral and filter-based detection techniques

that aim to prevent wide-spread unsolicited contact. Third, if the victim contacts the attacker, less suspicion is raised, and there is a higher probability that a social engineering attack (e.g., phishing, a financial scam, information theft, etc.) will be successful.

In general, Reverse Social Engineering attacks can be classified based on two main characteristics:

- *Targeted/Un-targeted*: In a targeted attack, the attacker focuses on a particular user. In contrast, in an un-targeted attack, the attacker is solely interested in reaching as many users as possible. Note that in order to perform a targeted attack, the attacker has to know (or acquire) some previous information about the target (e.g., such as her username or e-mail address).

- *Direct/Mediated*: In a direct attack, the baiting action of the attacker is visible to the targeted users. For example, an attacker can post a message on a public forum, or publish some interesting picture on a website. Mediated attacks, in contrast, follow a two-step approach in which the baiting is collected by an intermediate agent that is then responsible for propagating it (often in a different form) to the targeted users.

In the following, we present three different combinations of RSE attacks within the context of online social networks.

Recommendation-Based RSE [Targeted, Mediated]

Recommendation systems in social networks propose relationships between users based on background, or "secondary knowledge" on users. This knowledge derives from the interactions between registered users, the friend relationships between them, and other artifacts based on their interaction with the social network. For example, the social networking site might record the fact that a user has visited a certain profile, a page, a picture, and also log the search terms she has entered. Popular social networks (e.g., Facebook) often use this information to make recommendations to users (e.g., "*Visit page X*", "*You might know person Y, click here to become her friends*", etc.).

From an attacker's point of view, a recommendation system is an interesting target. If the attacker is able to influence the recommendation system and make the social network issue targeted recommendations, she may be able to trick victims into contacting her. Figure 2.1(a) demonstrates the recommendation system-based RSE attack scenario.

A variation of this attack could be executed in an un-targeted form, where the recommendation system is simply used to attract a high number

of possible victims. In most cases, however, natural restrictions (e.g. maximum friend request/time, maximum friends, maximum messages/day) are a limiting factor for this kind of approach.

Demographic-Based RSE [Un-targeted, Mediated]

Demographic-based systems in social networks allow establishing friendships based on the information in a person's profile. Some social networks, especially dating sites (e.g., Badoo), use this technique as the norm for connecting users in the same geographical location, in the same age group, or those who have expressed similar preferences.

Figure 2.1(b) demonstrates an RSE attack that uses demographic information. In the attack, the attacker simply creates a profile (or a number of profiles) that would have a high probability of appealing to certain users, and then waits for victims to initiate contact.

Visitor Tracking-Based RSE [Targeted, Direct]

Visitor tracking is a feature provided by some social networks (e.g., Xing, Friendster) to allow users to track who has visited their online profiles.

The attack in this case involves exploiting the user's curiosity by visiting their profile page. The notification that the page has been visited might raise interest, baiting the user to view the attacker's profile and perhaps take some action. Figure 2.1(c) outlines this attack method.

2.2 RSE Attacks in the Real-World

In this section, we present three types of real-world RSE attacks that are possible on three different social network platforms: Facebook, Badoo, and Friendster. In particular, we describe a recommendation-based RSE attack on Facebook, a demographic-based RSE attack on Badoo, and a visitor tracking-based RSE attack on Friendster.

Table 2.1 shows the social networks that were used in the experiments, and also describes which kind of RSE attacks are possible against them. Note that not all the combinations are possible in practice. For example, Facebook does not provide any information about the users that visit a certain profile, thus making a visitor tracking attack infeasible. In the rest of this section, we describe the different steps that are required to automate the attacks, and the setup of the experiments we performed.

2.2.1 Influencing Friend Recommendations

A good example of a real recommendation system is Facebook's friend suggestions. During the tests with Facebook, it was observed that Facebook pro-

Type of Attack	Facebook	Badoo	Friendster
Recommendation-Based	✓✠	-	-
Demographic- Based	✓	✓✠	✓
Visitor Tracking-Based	-	✓	✓✠

Table 2.1: RSE attacks on three popular social networks. ✓ indicates that the attack is possible; ✠ indicates that the effectiveness of this attack on the particular social network was demonstrated and measured.

motes the connection of users by suggesting them friends that they probably know. The system computes these suggestions based on common information, such as mutual friends, schools, companies, and interests. This feature is well-known to many social network users. In fact, whenever a user is logged in, she is regularly notified of persons that she may know.

Previous work [32] has shown that Facebook also uses the e-mail addresses a user has queried to identify a possible friendship connection between two users. The premise is that if users know each other's e-mail addresses, they must be connected in some way. Therefore, if an attacker gains access to the e-mail address of a victim (e.g., a spammer who has a list of e-mails at her disposal), by searching for that address, she can have a fake attacker profile be recommended to the victims. In these experiments, it was observed that this technique results in the attacker profile being the most highly recommended profile.

For the first experiment, the data collected for over a year in a previous study on Facebook [32] was used. In this study, a single account was registered that was used to perform a large number of e-mail search queries, using an email list obtained from a dropzone on a machine compromised by attackers. This profile was later recommended to all the queried users as a potential friend. As a result, the test account received thousands of messages and friend requests.

2.2.2 Measuring RSE Effects by Creating Attack Profiles

In the second set of experiments, five different attack profiles were created in three social networks. The profiles were designed with different characteristics to enable us to observe and measure the effects that each characteristic had on the effectiveness of the RSE attacks. That is, it was interesting to determine which features would attract the higher number of potential victims using the recommendation-based, demographic-based, and visitor tracking attacks.

The five attack profiles are shown in Table 2.2. For the profile pictures, popular photographs from Wikipedia were used, licensed under the Creative

Attribute	Prof. 1	Prof. 2	Prof. 3	Prof. 4	Prof. 5
Age	23	23	23	35	23
Sex	Male	Female	Female	Female	Female
*Location**	N.Y.	N.Y.	Paris	N.Y.	N.Y.
Real Picture	Yes	Yes	Yes	Yes	No

Table 2.2: Characteristics of the dummy profiles used in the experiments. (* In Badoo, more popular in Europe, N.Y was replaced with London)

Commons license. All photos represented an attractive male or female, with the exception of Profile 5 for which a synthetic cartoon picture was used.

Table 2.3 shows the number of users targeted in the tested social networks. For example, in the Facebook experiment, a total of 250,000 profiles were targeted, equally divided between the 5 attack profiles. In the demographic-based attack on Badoo, no action was required on behalf of the attacker. Hence, the number of targeted users is not given (i.e., all registered Badoo users could have found and contacted the attacker profile).

Social Network	# of Targets	Total users	Alexia Rank
Badoo	-	73 million	143
Facebook	250,000	800 million	2
Friendster	42,000	8.2 million	643

Table 2.3: Overview of OSNs as well as number of users targeted.

2.2.3 Automating the Measurement Process

During the study, a number of scripts to automate the three attacks and the measurement process on the different social networks were developed.

2.2.3.1 Recommendation-Based RSE on Facebook

As shown in Figure 2.1(a), the recommendation-based RSE attack against Facebook consisted of two parts: First, the target user's profile was probed using an e-mail lookup, and second, the attack accounts were automatically monitored for victims who contacted these accounts based on the friendship recommendation made by Facebook. To validate that the victims really correlate with the previously queries email address, their facebook id's were stored.

For the first part, the "contact import" functionality provided by Facebook and the API provided by Google Mail's address book to automatically search for users by their e-mail addresses were used. The total set of users to query for was broken into smaller sets, and sent to Facebook in multiple

requests, as they have limited the number of e-mail addresses that can be queried using a single request (because of recommendations made in previous work [32]).

In the second part of the experiments, an API was written that allowed us to interact with Facebook to accept friend requests, fetch user profiles, as well as fetch any private message that may have been sent to the attack profiles.

Note that CAPTCHAs in Facebook were only encountered if no heed was paid to the rate limits.

2.2.3.2 Demographic-Based RSE on Badoo

Badoo was used to test the demographic-based RSE attack. Hence, only the attack profiles and a means to automatically monitor incoming connections had to be created. Just like in the recommendation-based RSE attack, any message sent to the attacker profiles was automatically retrieved and collected. Furthermore, as Badoo allows to see which users have visited a profile, this information was also logged.

2.2.3.3 Visitor Tracking-Based RSE on Friendster

Friendster was used to perform the RSE attack based on visitor tracking. As shown in Figure 2.1(c), this attack consists of two parts: First, a visit to the target user's profile was initiated and as a consequence, the system shows to the victim that someone has visited her profile. If the attacker profile is interesting, the victim may choose to contact the attacker. Hence, in a second step, the visits and the incoming messages to the attack profiles were automatically monitored to determine which of the victims came back and initiated contact.

2.3 Experiments

2.3.1 Recommendation-based RSE Attacks

Most of the previously discussed attacks can be implemented in a real-world experiment, depending on the recommendation system the targeted social network uses. In this Section, we discuss experiments conducted on the most prevalent *pure* social networks.

2.3.1.1 Initial Experiment

During the aforementioned study [32], we observed that the used test account to query e-mail addresses was receiving a large number of friend re-

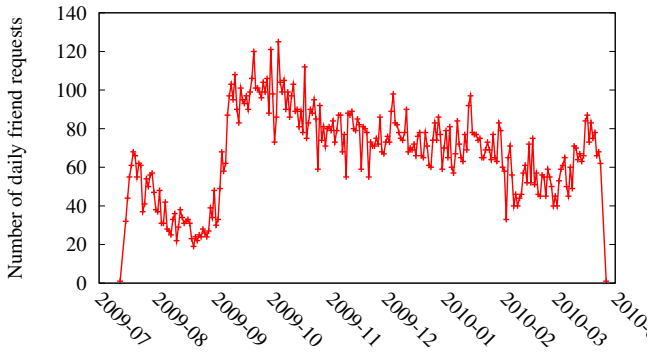


Figure 2.2: Daily number of new friend requests in the initial Facebook experiment

quests. The profile used in this attack was similar to Profile 2 described in Table 2.2.

Figure 2.2 shows the number of daily friend requests received by the account used in this initial experiment. The graph shows that during the first two months, the account received an average of 45 requests per day, followed by an increase to an average of 75 requests per day for the next 6 months.

The rapid increase in the number of request is the consequence of the cascading effect that commenced when the incoming invitations were accepted. The fact that the account had a large number of friends built up the "reputation" of the used profile. In addition, the account was being advertised by Facebook to new people with whom it shared common friends.

Of the over 500,000 e-mails queried by the decoy profile, it was contacted by over 17,000 users (i.e., 3.3% friend connect rate within 9 months and 0.37% friend connect rate per month). Note that the test account reached both the maximum number of active friend connections and the total number of pending friend requests allowed by Facebook. Furthermore, such an effect is thinkable on any other social network as well as it is primarily caused by human factors.

2.3.1.2 Controlled, In-Depth Experiment

After the success of the initial experiment, a number of controlled, in-depth experiments were started to measure and determine which profile characteristics and social network features affect the success rates of RSE attacks.

To reach this goal, five attack profiles were created on Facebook. For each profile, 50,000 target users were randomly selected and their e-mail addresses looked up (hence, influencing the recommendations made by Facebook). Furthermore, the number of friend-requests, private messages,

and other interaction sent to each attack profile were measured. Figure 2.3 depicts the result of this experiment. The y-axis represents the cumulative number of friend requests or messages for the period represented by the date on the x-axis.

Profiles 2 and 3 were the most successful in terms of the number of friend requests and messages that were received. Both profiles correspond to attractive females who are interested in friendship. Note that there was no correlation with the location of the attack profile (i.e., the location did not influence friend requests). Hence, an initial analysis seems to confirm the general intuition that an attractive female photograph will attract potential victims. In contrast to the other profiles, Profile 5 was the least effective. In this profile, a cartoon character was used as a photograph rather than a real picture. In comparison, Profile 1 performed only slightly better than Profile 5. This profile contained the photograph of an attractive male.

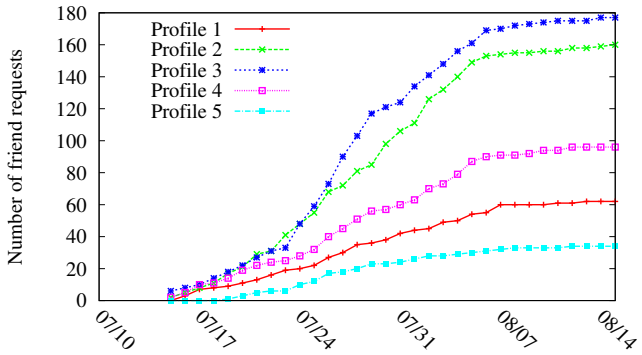
Over the entire month, the most effective profile had a friend connection rate of 0.35% (i.e., in line with the initial experimental profile). The least effective profile instead, had a friend connection rate of only 0.05%.

Although friend connection requests and private messages were the most common form of interaction with a decoy profile, a large number of friend suggestions were received also. Friend suggestions are suggestions made by the victim to other users. Such suggestions are important as they imply that a high level of trust has been achieved between the attacker and the victim. Also, note that over 94% of the messages to the attack profiles were sent after the friend connection requests.

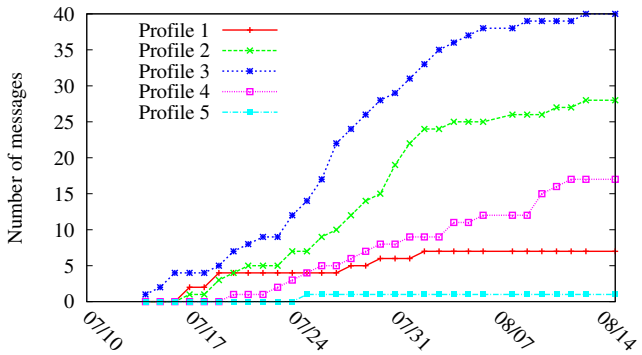
By analyzing the demography of the users who contacted the attack profiles, potential characteristics that make a decoy profile appealing can be identified. In particular, we focused on three fields: relationship status, interested in, and age (Figure 2.4). The y-axis of the figure shows the percentage of friend connection requests that originated from a profile with the respective demographic value (empty values excluded) to the attack profile listed on the x-axis. Young, single users who have expressed interest in "Women" seem to be the easiest victims to attract. In comparison, Profile 1 (the only male profile) received a larger number of friend requests from users who had expressed interest in "Men".

Interestingly, the profile with a cartoon picture was the one to attract the largest number of requests coming from older users (i.e., those who were older than 40). Hence, the experiments show that by carefully tweaking the profile information, it is possible to obtain a higher success rate against a particular group of users.

Finally, the messages that were sent to the different attack profiles were analyzed. To protect the privacy of individuals in the study, user identifiers were removed before processing the messages. After anonymization, only ran word-based statistical analyses on the message contents were used. That is, as a pre-processing step, Porter's stemming algorithm was used on the



(a) Friend connect requests sent to each profile



(b) Messages sent to each profile

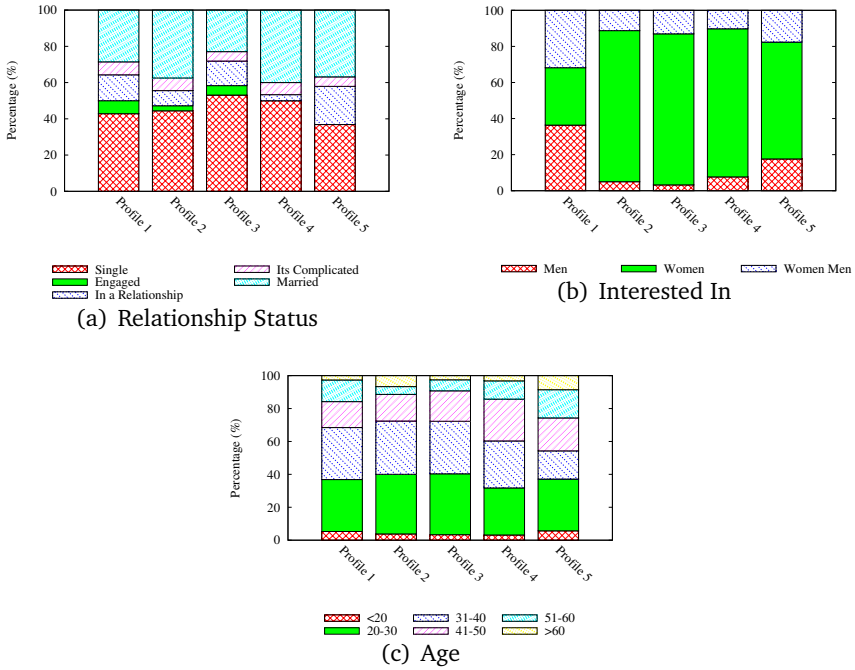
Figure 2.3: Cumulative counts of interactions resulting from reverse social engineering on Facebook.

extracted tokens [102], followed by a count of n-grams (where a single gram is a stemmed token).

Around 10% of the messages mentioned the Facebook recommendation, including 3-grams such as “suggest you as” or “suggest I add”. The analysis shows that some users used the recommendation made by the social network as a pretext to contact the attack profile.

2.3.2 Demographic-based Experiment

For a demographic-based RSE attack, Badoo, a dating oriented socializing system that allows users to meet new friends in a specific area, was targeted. A registered user can list the people who have visited her profile and exchange messages with other users. Figure 2.5 shows the cumulative number of visitors and messages received for each attack profile that was created in the network.



(a) Relationship Status

(b) Interested In

(c) Age

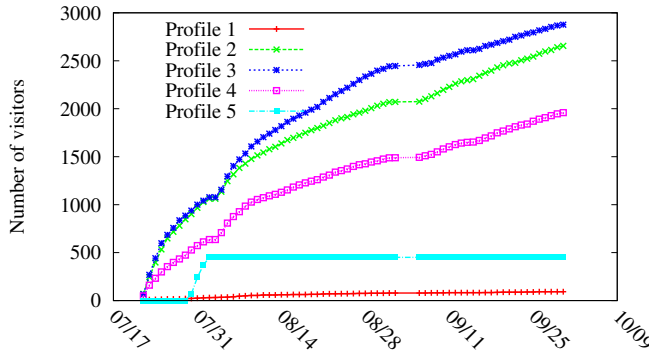
Figure 2.4: Demographic breakdown by Relationship Status, Interested In, and Age for Friend Connect requests on Facebook.

Profiles 2 and 3 were again the most popular, and attracted the most visitors (over 2500 each). These profiles also received the largest number of messages (i.e., more than 2500 each). Because Profile 5 was not using a photograph of a person, it was removed by Badoo from the demographic search after it was visited by 451 users and it received 383 messages. Once again, Profile 1, the attack profile of a male user, received the fewest visits and friend requests.

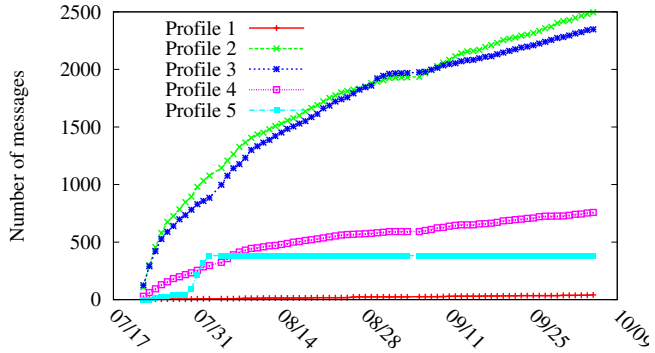
Another measure of how successful an attack profile was, is the percentage of users who decided to send a message after visiting a profile. These figures are over 50% for the two attractive female profiles (Profile 2 and 3), and 44% on average for all attack profiles.

We took a closer look at the demography of the users who contacted us. In the case of Badoo, sending a message is the most concrete form of interest, and one that can easily be exploited (e.g., [34]). Figure 2.6 shows a demographic breakdown by relationship status, what users were interested in, and age. Similar to Figure 2.4, the y-axis shows the percentage of users who sent messages that originated from a profile with the respective demographic value.

Note that Badoo is a site that is geared towards dating. Most of the users who initiate contact express that they are either single, or in an "open rela-



(a) Visitors to each profile



(b) Messages sent to each profile

Figure 2.5: Cumulative counts of interactions resulting from reverse social engineering on Badoo.

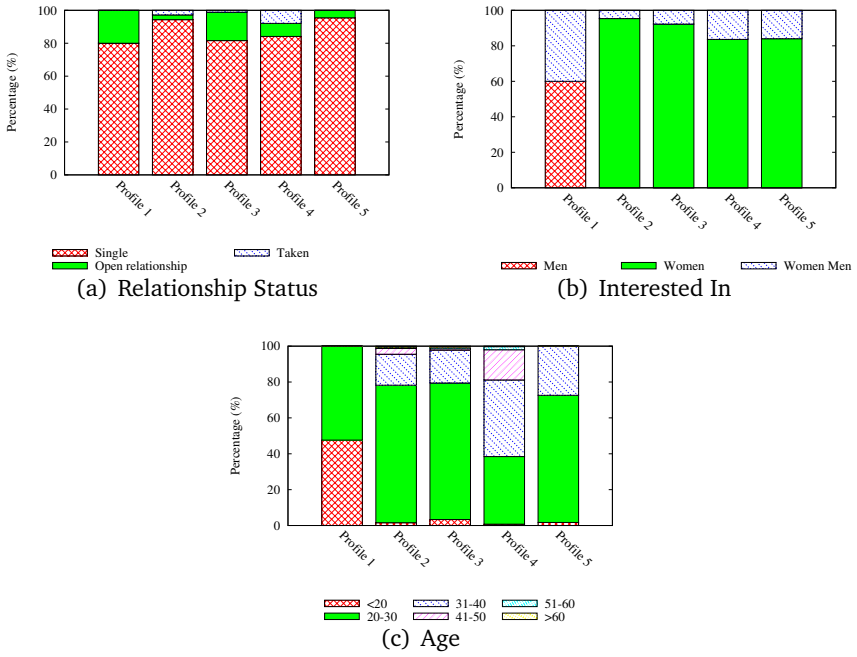
tionship". In general, the attack profiles only attracted users of the opposite gender. The age demographic shows that most of the victims belong to the same age group that the attack profile belongs to. In comparison, there was no correlation of age for contact requests on Facebook.

Another important difference with respect to Facebook was that the location was significant in Badoo. In fact, almost all the messages were sent by people living in the same country as the attack profile.

Finally, the 3-grams analysis for the messages received on Badoo showed that the most popular term was "how are you" occurring over 700 times. Other popular lines included "get to know" and "would you like", "you like" . . . "chat" or "meet".

2.3.3 Visitor Tracking Experiment

In the visitor tracking RSE attack, each of the five attack profiles to visit 8,400 different user profiles in Friendster were used. As it was previously



(a) Relationship Status (b) Interested In

(c) Age

Figure 2.6: Demographic breakdown by Relationship Status, Interested In, and Age for messages on Badoo.

described, on Friendster a user can check which other users have visited her profile.

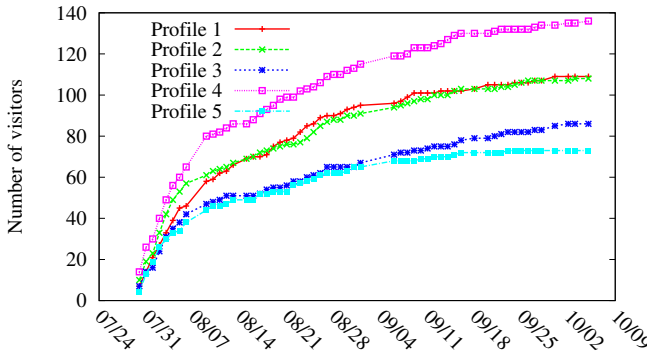
In this experiment, it was tracked which victims visited the attack profiles, and the number of users who sent a friend request was counted. The results of this experiment are shown in Figure 2.7 (the sub-figure 2.7(a) and 2.7(b) represent the number of visitors and number of friend requests sent to the attack profiles).

The number of users who were curious about a visit, and visited us back was consistent with the results of the experiments which was conducted on other social networks (i.e., between 0.25% and 1.2% per month). However, only a few users later sent a friend request or a message.

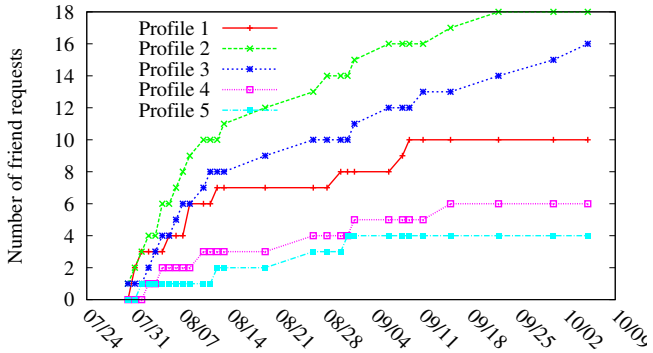
The demographic breakdown for Friendster is presented in Figure 2.3.3. The statistical distributions are similar to the ones obtained in the Facebook experiment, proving the difference in terms of characteristics between friend-oriented and dating-oriented social networks.

2.4 Discussion

In this section, based on the results of the empirical experiments, some insights about the way RSE attacks work in social networks can be distilled.



(a) Visitors to each profile



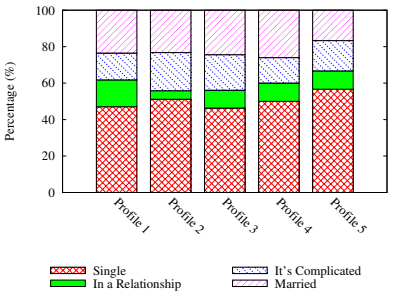
(b) Friend requests sent to each profile

Figure 2.7: Cumulative counts of interactions resulting from reverse social engineering on Friendster.

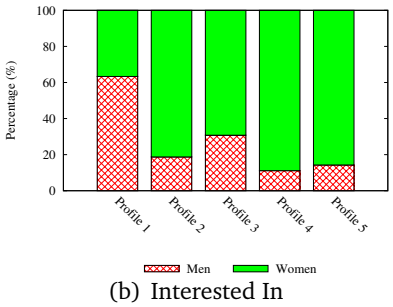
We can summarize these findings in two main points: The importance of having the right profile, and the importance of providing a pretext to the victims.

The first, straightforward, factor which is possible to measure is the impact of the profile characteristics on the overall effectiveness of an attack. The experiments confirm the folk wisdom that using an attractive female photograph is a good choice to attract victims. The success rate of the most successful female profile, in terms of both friend requests and number of received messages, is between 2 and 40 times higher than the worst performing profiles (i.e., the male profile and the profile without a photograph).

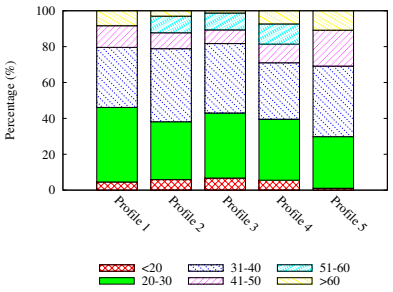
Note that if the objective of the attack is not simply to reach the highest number of users, but to target a specific person, or group, the success rate of the attack can be improved by carefully tuning the profile characteristics. For example, the experiments show that age and location information are decisive in dating sites, while this information is not as critical in more general, friend-oriented, social networks. Also, the results suggest that gender



(a) Relationship Status



(b) Interested In



(c) Age

information is always very important. Hence, a successful reverse social engineering attack should use the opposite sex of the victims in the decoy profile.

The experiments show that the impact of the profile picture is quite uniform in different social networks. For example, we observe that young users are generally more intrigued by attractive photographs, while decoy profiles (e.g., Profile 5) that do not contain the photograph of a real person tend to attract more senior users.

Obviously, even though having a catchy, interesting profile is important, this research shows that there is a second, even more important factor that contributes to the success of the attack: the pretext. The experiments indicate that users need an incentive and a good reason to engage in interaction with a person that they do not know. In other words, users need a good excuse to "break the ice" and motivate the first approach. The differences between the success rates of the attacks on Facebook and Friendster suggest that an incentive or a pretext is critical for reverse social engineering attacks to work in practice. This thought can even be mapped to completely different social structures like online games. Here the incentive to contact a person would even be higher bearing the thought that the whole point of these networks is finding other people to spend time with.

The analysis of the messages received on Facebook support the hypothesis that a recommendation system gives a reason to users to initiate contact. That is, a number of users referenced the Facebook recommendation as a

motivation for their friend request. In contrast, on Friendster, even though the percentage of users that browsed the decoy profiles was consistent with the other social network experiments, very few people moved to the next step and sent a contact message. The reason is that the visitor tracking attack failed to provide a good pretext to the victims.

Note that the demographic experiment on Badoo was also very effective. The reason for this success is that Badoo greatly relies on the demographic search functionality to allow users to find possible contacts. In the case of a dating site, the pretext for establishing contact was the fact itself of living in a close location, or being in the same age group of the victim.

These experiments demonstrate that reverse social engineering attacks on social networks are feasible if they are properly designed and executed. However, contrary to the common folk wisdom, only having an account with an attractive photograph may not be enough to recruit a high number of unsuspecting victims. Rather, the attacker needs to combine an attractive profile with a pretext and incentive for the victim to establish contact. Recommendation systems such as Facebook's friend suggestions are effective tools for creating such an incentive. Also, we see that profile attributes such as location and age may be the required incentives on dating networks such as Badoo.

2.5 RSE Countermeasures in OSN

Clearly, features that allow social network users to easily make new acquaintances are useful in practice. However, such systems may also be abused to trick users on behalf of attackers. In this section, we list three countermeasures that would increase the difficulty of launching RSE attacks in online social networks.

First, while friend recommendation features are useful, the experiments show that they may pose a risk to users if the attackers are able to somehow influence the recommendation system. Hence, it is important for social network providers to show a potential connection between two users only if there is a strong connection between them. For example, in the case of Facebook, as these experiments show, a simple e-mail lookup does not necessarily indicate that the users know each other. Thus, one could check other information, such as the fact that the users already have some friends in common.

Second, we believe that it is important to closely monitor friendships that have been established in social networks. Benign user accounts will typically send and receive friend requests in both directions. That is, a user may be contacted by people she knows, but she will also actively search and add friends on the network. However, in contrast, a honeypot RSE account (as we described in this chapter) only receives friend requests from other users.

Thus, it may be possible to identify such accounts automatically. An RSE attacker may also send friend requests (to act like a regular user). To detect such attacks, one could find heuristics and thresholds while monitoring the friendships activity, and raise alarms (e.g., anomaly detection).

Third, we believe that CAPTCHA usage also needs to be extended to incoming friend requests. Today, because of the active threats of spamming and social engineering, social network providers may display CAPTCHAs when friend requests are sent to other users. However, no such precautions are taken for messages and friend requests that are received. By requiring to solve a CAPTCHA challenge before being able to accept suspicious incoming friend requests, we believe that RSE attacks would become more difficult. While CAPTCHAs are not the silver bullet in preventing and stopping malicious activity on social networks (e.g., as show in [23, 34]), they do raise the difficulty bar for the attackers.

3 Social-based authentication mechanisms

As discussed in Chapter 2, Online Social Networks (OSNs) are a very attractive target for the cyber miscreants, who have started to harvest social networking profiles' credentials using both technical attacks (e.g., phishing, spear phishing) and social engineering approaches. Interestingly, recent studies [106] have shown that traditional underground economies have shifted their focus from stolen credit card numbers to compromised OSN accounts. A recent study [60] has shown that the vast majority of spamming accounts in social networks are not fake ones created by attackers but from legitimate user accounts that have been compromised. Additionally, new Facebook phishing attacks use compromised accounts to steal credit card information from victims [8].

In the remainder of this chapter we describe an authentication mechanisms born within and inspired by OSNs: the Social Authentication (SA). Specifically, we describe how SA works in practice, using the use case of Facebook, and, because of its high usability, it may lead to a scarce degree of protection.

3.1 Social authentication

Although several effective solutions exist for mitigating purely technical attacks such as brute-force password guessing, it is more difficult to protect OSN users from those attacks that incorporate social aspects. Online banking sites, and recently Google Mail, adopt two-factor authentication mechanisms, where users must present two separate pieces of evidence (or factors) in order to authenticate. The two factors are designed in such a way that it must be very difficult for an adversary to acquire both of them.

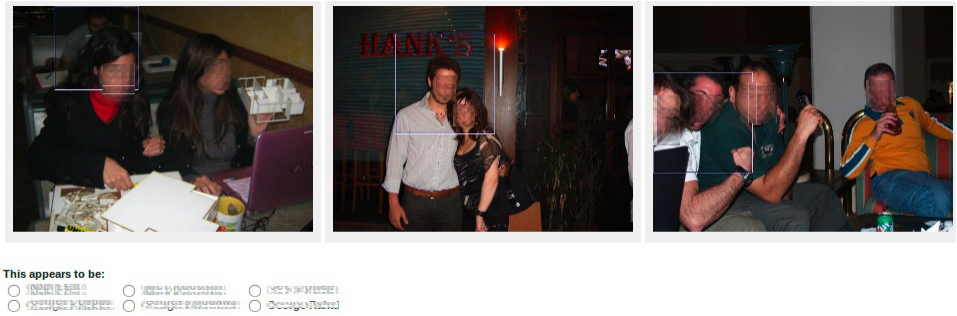


Figure 3.1: Screenshot the user interface of one of the instances of Facebook's SA pages. We pixelized the faces on purpose, for privacy reasons.

In the most common deployment, the two factors comprise something that

(1) the user knows (e.g., a password) and something that

(2) the user possesses (e.g., a hardware token).

Physical tokens turn out to be troublesome for users, who may not always have them with them, and costly for the service employing them.

In an attempt to boost the usability of two-factors authentication mechanisms, Facebook has currently been beta testing its so-called "Facebook Social Authentication"[1], or Facebook SA in short. This mechanism leverages knowledge that the account owner has, but an adversary lacks and cannot easily trick the owner into divulging. Specifically, after the traditional password-based authentication, a Facebook user is asked to name seven friends of his or her, depicted in a series of photos. The key difference with, for example, a hardware token or a confirmation code sent via SMS, is that Facebook users are accustomed to tagging photos of friends. Therefore, Facebook's SA is, in principle, a usable alternative to classic out-of-bound secrets.

Facebook's SA was announced in January 2011 and it was the first instance of an authentication scheme based on the "who you know" rationale: A user's credentials are considered authentic only if the user can correctly tag his friends in a randomly-chosen photo of their own. Given the enormous spread of the Google Plus social network, and the face-detection and recognition service available previously within Google Picasa, it will not be surprising if Google Plus will eventually adopt the same mechanism.

[1] http://www.facebook.com/blog.php?post=486790652130

3.1.1 Scheme

Facebook's SA authenticates users based on the people they know. After the traditional, password-based authentication, the user is prompted with a sequence comprising seven sets of photos. As shown in Figure 3.1, each set of photos in the sequence includes three photos of a specific person chosen randomly from the user's list of friends. Each set also includes six possible suggested answers, which are supposed to help users remember the exact names of his or her friends.

The user must identify the person depicted in each set. The user is allowed to skip (or mistake on) 2 pages, but must correctly identify 5 photos, otherwise the test must be repeated.

3.1.2 Requirements

Facebook triggers its SA based only if certain requirements are met by the user's account.

First, a reasonable *number of friends* must be available. Based on some preliminary experiments, in the case of Facebook, a user must have at least fifty friends (circa). To obtain this information, six distinct fake profiles were created, and the number of friends of these accounts was increased on a weekly basis until the SA could be triggered manually.

Second, each user's friend must be tagged on a reasonable *number of photos* of themselves for the mechanism to have a large pool of photos. For instance, landscape pictures are of little or no help. On the other hand, user-submitted tags are often assigned to funny objects and sometimes there are photos where not all tagged people are present. Although there is no official announcement nor strong evidence to corroborate this, it seems that Facebook makes an effort to select *photos that contain tagged faces* using a face detection algorithm. From a manual analysis on a sample of the SA pages that we collected, we have noticed that Facebook's selection process is imprecise and sometimes photos with no faces are chosen. As a further check, 32,161 distinct photos were processed, belonging to 238 distinct users, using one of the most advanced face detection algorithms and it was found that only 68% of the users had at least one photo with a detected face.

Facebook triggers the SA for user accounts that meet the above requirements, when it considers a login attempt to be suspicious. In practice, this happens when a user attempts to log in from a geographical location from which the account has never been accessed before, or when a user attempts to log in from a device that is not associated with the user account.

3.2 Vulnerabilities of social authentication

The major difference from other two-factor authentication mechanisms such as those that ask for a confirmation code sent via text message, is that Facebook users are less bothered by social authentication, especially because it entails a simple activity that regular users perform on a daily basis: Tagging friends in photos.

However, as noticed recently by other researchers, designing a usable yet secure social authentication scheme is very difficult. More precisely, H. Kim et al. in [78] described a mathematical model that quantifies the degree of risk behind social authentication mechanisms. The practical investigation suggests that this form of authentication has some serious drawbacks even in the wild. First of all, the number of friends and the number of photos per friend can influence the applicability and the usability of the SA. In particular, if a user has a very large number of friends, it may be difficult to identify them, especially when there is little or no actual relationship with such friends. A typical case is a celebrity or a public figure. Another parameter that influences the usability of the SA is the number of photos that depict the actual user, or at least that contain objects that uniquely identify the particular user. Last, but most important, in the remainder of this chapter we show that SA is broken in practice, because the secrets that it relies on (i.e., each user's friends faces) are publicly available for the vast majority of Facebook users.

Although Facebook's SA scheme is considered secure, as suggested in [78], the progresses made by face recognition techniques may threaten the security of these face-based authentication mechanisms. Curiously, M. Dantone and collaborators have shown in [46] that social relationships can be used also to improve the accuracy of face recognition. Later on, A. Acquisti and colleagues at BlackHat USA 2011 [29] went beyond the previous approach and presented a system that can associate names to faces and, thus, retrieve an anonymous' identity solely by using a picture of his or her face. Although no scientific experimentation on real-world data has been made to measure the weakness of social authentication, these works suggest that the face-to-name relation, which is the key intuition behind social authentication, may be exploited further to demonstrate that the scheme is insecure.

3.2.1 Security analysis

Within WP5, POLIMI has been analyzing the social authentication mechanism and has found a series of weaknesses enabling an adversary to carry out a simple yet effective automated attack against this security mechanism. The key idea of social authentication is to employ the unique knowledge compiled when other users are grouped together to form his online friends. The photos of a user's friends or other information about them, individually

or as a whole, is employed to generate security challenges one has to solve, in addition to providing the actual password, to log in under certain circumstances. The strength of this security mechanism lies in the fact that only the user has access to that unique set of information. The intuition behind this research, which is also the key vulnerability of SA, is that such information can also be accessed and leveraged by an adversary to acquire the necessary pieces to solve the security challenges and thereby defeat the social authentication mechanism, given she has already access to the password of the user. To this end, the project partners have been conducting a series of experiments to validate the assumptions about the access an adversary might have to such information. Statistically speaking, the vulnerability identified within this research impacts the majority of Facebook users, because friend lists and published photos are publicly-available by default, and 80% of social network users normally do not change the default privacy settings [28, 83].

4 Social Plugins

After the previous chapters dealing with attack scenarios and defenses tailored to social networks, this chapter discusses the more functional part of such a network and its possibilities to interact with other websites by utilizing so-called plugins. Social plugins enable third-party websites to offer personalized content by leveraging the social graph, and allow their visitors to seamlessly share, comment, and interact with their social circles [7]. For example, Facebook's Like button, probably the most widely deployed social plugin [1], enables users to leave positive feedback for the web page in which it has been embedded, share the page with their friends, and view their friends that have "liked" the page, along with the total number of "likes" from all visitors. Google's "+1" button [12] offers almost identical features to the Like button, while similar widgets are also available from other popular social networking sites such as Twitter and LinkedIn.

Social plugins have also been used for a wide variety of other applications including authentication. For example, instead of a web site implementing its own authentication system with user names and passwords, it may use a *social login* plugin offered by a social networking platform such as Facebook. Additionally to the weaknesses discussed in the previous sections, the plugin itself can put down some additional drawbacks. In theory, this approach to authentication not only saves visitors from the burden of remembering one more password, but also gives them the opportunity to experience a personalized service from the web site based on their preferences and social circle.

Unfortunately, both technologies come with significant compromises in user privacy. Social plugins enable social networking sites to track a growing part of the browsing activity of their members. Social login enables third-party websites to access private information in a user's profile. In this section, we present a detailed assessment of the issues of both social plugins and social login in respect to user privacy.

4.1 Social Plugins

Social plugins offer multifaceted benefits to both content providers and members of social networking sites, a fact that is reflected by the tremendous growth in their adoption over the past two years. Indicatively, as of January 2012, more than two million websites have incorporated some of Facebook's social plugins, while more than 27% of the top 10,000 websites include Like buttons—a percentage three times higher than just one year ago [1]. Unfortunately, as the number of websites that incorporate social plugins increases, so does the erosion of privacy of their visitors.

To personalize the content of a third-party web page, a social plugin connects to the social networking service and transmits a unique user identifier, usually contained in an HTTP cookie, along with the URL of that page. In response, the plugin receives and displays private information relevant only to the individual visitor, typically coupled with other public information. In other words, the social networking service receives detailed information about every visit of its members to any page with embedded social plugins as long as they maintain an active session with the service. Consequently, the larger the number of sites with social plugins, the greater the potential for extensive user tracking. Considering the increasing adoption rate of social plugins, a constantly growing part of its members' browsing history can be precisely tracked.

More importantly, the cookies used in social plugins are linked to user profiles on the social networking site that typically contain the person's name, email address, and other private information. Third-party tracking cookies, as used by advertising networks and traffic analytics services, also aim to track the pages visited by a specific user [71]. In essence, though, they track the pages opened using a particular browser instance running on a device with a given IP address. While this can already be considered as personally identifying information to some extent, in addition to that information, social plugins reveal much more: the browsing history of *individuals*.

At the same time, out of convenience, members of social networking sites practically never log out. The frequency of their visits to the site itself, along with the daily use of affiliated services, such as e-mail, cloud-based office software, or games, keep a session with the site constantly alive [10]. This session state is stored along with the same set of cookies used in social plugins, which are served by the same domain. Thus, the cookies that enable user tracking through social plugins tend to never expire.

The important implications of social plugins to user privacy were identified soon after their release [24, 105], and concerns by diverse user groups and organizations have been intensifying [24, 27]. As avoiding becoming a member of any social networking site is often rather difficult (even users that are not interested in the social aspects of a service can be affected, e.g.,

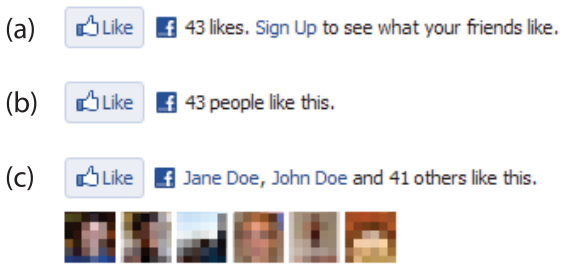


Figure 4.1: Different states of Facebook's Like button for a user that (a) has never logged in on Facebook from this particular browser or is not a member of Facebook (non-personalized view), (b) has previously logged in but is currently logged out (non-personalized view), (c) is currently logged in (personalized view).

Gmail users can still be tracked through Google's "+1" buttons), privacy-conscious users can resort to browser extensions [15, 11, 4, 2, 17, 81] that block user-identifying information from reaching the social networking platform through social plugins. Depending on the subtlety of their approach, ranging from stripping cookies and headers from the plugin's requests to preventing completely the plugin from loading, some or none of its functionality may be preserved. However, as user-identifying information never reaches the social network, these approaches completely disable any kind of content personalization. As an example, for a Like button, even logged in members will be viewing only a non-personalized version that shows just the total number of "likes" for the page (Figure 4.1(a), Figure 4.1(b)), instead of the names and pictures of their friends who have liked the page (Figure 4.1(c)).

4.1.1 Background

Social plugins are provided by the major social networking sites in the form of "widgets" that can be embedded in any web page. Through these plugins, visitors receive personalized information and interact with their social circle in relation to the content of the embedding page. For instance, visitors can convey positive feedback, leave comments, or share the page with their friends or the world, under their social network identity.

Developers can easily incorporate a social plugin in a web page by including an IFRAME element in the page's HTML code. After downloading the page, the user's browser will issue a subsequent HTTP GET request for the IFRAME's source URL to fetch and load the content of the plugin, as shown in Figure 4.2 (step 2). The domain that serves the social plugins is the same as the one that hosts the social networking site itself, and thus all

state information that the browser maintains for the social networking site in the form of HTTP cookies [13] will be transmitted along with the request for the social plugin.

Listing 4.1 shows an actual network capture of an HTTP a GET request for loading an instance of Facebook's Like button, which contains a unique user identifier (`c_user=CURRENT_USER` at line 10) as part of the cookie. The request also contains the URL of the target page that contains the social plugin as an attribute value in the query string of the IFRAME's source URL (`href=TARGET_URL` at line 1). Finally, the Referer [sic] header reports the URL of the embedding page (line 6), which usually matches the target page.

Assuming the user has an active session with the social networking site, e.g., because one of its pages is already loaded in another browser tab, or the user has opted for a "keep me logged in" feature, the site will associate the request with the user's online profile, and respond to the social plugin request with personalized content tailored to that particular user and visited web page (step 3 in Figure 4.2). Otherwise, if the user has never logged in on the social networking site from that particular browser or has never registered at all, the social plugin will display only generic, publicly accessible information for that page, and will usually prompt the user to sign up on the social network.

For instance, Figure 4.1 shows the different modes of the Like button depending on the browser's cookies for the `facebook.com` domain. If the user has not logged in on Facebook using that browser, or does not have an account on Facebook, the plugin displays only the total number of likes and prompts the user to sign up (a). If the user is currently logged in, the plugin displays personalized information, including some of the names and (if the developer has chosen so) the pictures of the user's friends that have liked the page (c). Interestingly, while the user is logged out (b), the plugin does not prompt for sign-up. Depending on how cookies are cleared by the social networking site upon user exit, some user-identifying information may still be present even after logging out.

4.1.2 Privacy Issues

The current approach for the implementation of social plugins, which is followed by all major social networks, requires the transmission of user-identifying information whenever a plugin is to be loaded on a user's browser. Interestingly, even if a web developer customizes a social plugin so as not to show any kind of personalized content, the browser will still append the user's cookies in the HTTP request for that plugin. Consequently, the social network is contacted whenever a user *visits* an external web page that contains a social plugin. This situation poses an increasing threat to the privacy of the members of major social networks, since their visits to third-party websites that contain social plugins can be precisely tracked [105].

```
1  GET /plugins/like.php?app_id=APP_ID&href=TARGET_URL&send=false&
       layout=box_count&width=90&show_faces=false&action=like&
       colorscheme=light&font&height=62 HTTP/1.1
2  Host: www.facebook.com
3  Connection: keep-alive
4  User-Agent: Mozilla/5.0 (Windows NT 5.1) AppleWebKit/535.2 (KHTML,
       like Gecko) Chrome/15.0.874.106 Safari/535.2
5  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q
       =0.8
6  Referer: EMBEDDING_PAGE_URL
7  Accept-Encoding: gzip,deflate,sdch
8  Accept-Language: en-US,en;q=0.8,el;q=0.6
9  Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.3
10 Cookie: datr=DATR; c_user=CURRENT_USER; xs=SESSION_ID
```

Listing 4.1: HTTP GET request for loading a Facebook Like button.

Content providers have been employing social plugins to reach a broader audience and improve the engagement of visitors on their websites. With publishers reporting multifold increases in traffic [25], and the continuous addition of new gestures and social features by the major social networking services [26], it is expected that the explosive popularity of social plugins will only continue to grow. As more sites employ social plugins, the potential for broader user tracking increases. At the time of this writing, with more than 27% of the top 10,000 most visited websites having Like buttons in their pages (as of February 2012) [1], a good part of the daily browsing history of Facebook members is technically available to Facebook.[1] We should stress that the same issue holds for all other major social networking platforms that provide social plugins, including Google and Twitter.

The privacy issues related to the use of HTTP cookies are a well-known problem. Since their introduction in web browsers in 1995, cookies have been extensively used by advertising networks for building user profiles and tracking the browsing activity of users across the web [91]. Although user tracking through social plugins resembles this kind of cross-site tracking through third-party cookies [71], these two types of tracking have two crucial differences.

First, although browser support for blocking third-party cookies, albeit not enabled by default, offers some mitigation against well behaved advertising networks (user tracking can still be achieved through other means, such as other persistent client-side storage mechanisms [75], or by combining a visitor's IP address and browser fingerprint [50]), it does not solve the problem of user tracking through social plugins. Social plugins are served by the same domain—and thus share the same cookies—with the social networking site itself, which users visit directly. This makes most browsers (except Firefox) to consider them as first-party cookies even when they are

[1] At the time of writing, Facebook has more than 845 million active users [10]. Facebook has officially stated that it does not use cookies to track visits to external websites [27].

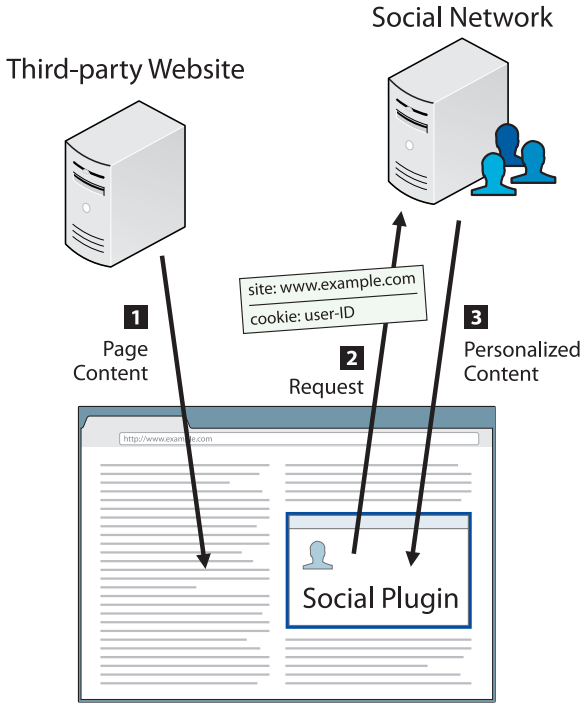


Figure 4.2: Loading phase of social plugins. After a page from a third-party website is fetched (1), the browser requests the content of the IFRAME that contains the social plugin (2). If the user is logged in on the social networking site, the plugin will receive and display personalized information (3). Users are identified (and their visit to the third-party website is tracked) by the social networking site through the HTTP cookies that are included in the request.

encountered in external web pages. Thus, blocking of third-party cookies in general does not block social plugins.

Second, an advertising network uses cookies to track the same user across all affiliate sites that host the network's advertisements, but cannot easily link the derived user profile to the actual identity of the user. In contrast, social plugins use cookies associated with real user profiles that typically contain an abundance of personally identifiable information [84]. In essence, instead of tracking anonymous users, social plugins enable tracking of named *persons* (perhaps with the rare exception of users with fake profiles). Advertising agencies can also potentially associate a user profile with a person's identity by combining information from other sources, e.g., in cooperation with one or more affiliate websites on which users provide contact information for registration. Social networking sites, though, do not have to collude with another party, since they have access to both extensive per-

sonally identifiable information, and to a broad network of sites that host social plugins.

4.1.3 Preventing the Privacy Leaks of Social Plugins

One might think that if users diligently log out of the social networking site, they will be safe from the privacy leaks caused by its social plugins. Unfortunately, this seems a rather daunting task for users that rely daily on Google, Facebook, Twitter, and other popular online services for their personal and professional communication and social interaction activities. To provide convenience for frequent use, these sites follow a single sign-on approach for all offered services, and prompt users to stay logged in indefinitely through "keep me logged in" features, even after closing the browser or switching off the computer. Consequently, users typically remain logged in throughout the whole duration of their online presence, and thus are constantly exposed to tracking through the social plugins of the respective provider. In practice, web sessions on the most popular social networking sites remain active for days or even weeks since the last log in time, unless the user explicitly logs out.

To make matters worse, in some cases even when a user explicitly logs out, the cookies of the social networking site might not be cleared completely, and personally identifying information may still persist [45]. For example, even after logging out of Facebook, a cookie with a unique user identifier (in particular, the `lu` attribute) remains in the browser, enabling features such as pre-filling a returning user's email address in the log in form, or avoiding to unnecessarily prompt existing members to sign up, as shown in Figure 4.1(b).

This situation drives privacy-conscious users to browser extensions [15, 11, 4, 2, 17, 81] that block user-identifying information from reaching the social networking platform through social plugins. For instance, Facebook Blocker [4] removes completely the IFRAME elements of social plugins from visited web pages. In this way, no request is sent towards the social networking platform and the plugins never appear on any third-party web page. Instead of blocking the requests completely, ShareMeNot [17] simply removes the sensitive cookies from the requests made by the social plugin at load time. When a user explicitly interacts with a plugin, e.g., clicks on a Like button, the associated cookies are then allowed to go through, enabling the action to complete normally. Although this approach strikes a balance between usability and privacy, it still completely disables any content personalization.

The Do Not Track HTTP header [3] is an encouraging recent initiative that allows users to opt out of tracking by advertising networks and analytics services. Although currently not supported by any social networking platform, if it were adopted, Do Not Track could allow users to choose whe-

ther they want to opt in for the personalized versions of social plugins or not. This requires social networking platforms to serve the loading phase of the plugins from a different domain than the one on which users are logged in. However, users who opt in for the personalized versions (or who do not opt out, depending on the default setting) can still be tracked.

4.2 Social Login

An emerging trend on the Web is "single sign-on" platforms that allow users to register and log in on multiple websites using a single account and an OAuth-like protocol [16]. Social networking sites, such as Facebook and Twitter, have been in the front lines of this trend, allowing their users to use their social networking site credentials in a plethora of third-party websites. This type of cross-site interaction enables, for instance, third-party websites to authenticate users based on their Facebook (or Twitter) identity. In addition, such sites may add a social dimension to users' browsing experience by encouraging them to "like," share, or comment on certain content using their social network capacity, i.e., automatically post respective favorable messages to their social profile and let their friends know about the site. To enable this social dimension, third-party sites request access and control over a user's information and account.

In other words, these sites request users to authorize web applications specific to the third-party site, or API calls originating from the third-party site, to access and control part or whole of their social profile. Unfortunately, this process may have several disadvantages, including:

Loss of anonymity. Even the simple act of signing on to a third-party website using the Facebook identity sacrifices the anonymous browsing of the user; his social identity usually contains his real name. In most cases it is unclear how this loss of anonymity is necessary for the site's purposes.

User's social circle revealed. Several of these third-party websites install web applications in the user's social profile or issue API calls which request access to a user's "friends." Although having access to a user's friends may improve the user's browsing experience, e.g., for distributed multi-player games, in most cases it is not clear why third-party websites request this information, and how based on it they are going to improve the user's browsing experience.

Loss of track. Once users start to enable a torrent of third-party applications to have access to their personal contacts, they will soon lose control of which applications and sites have access to their personal data, and thus they will not be able to find out which of them may have

leaked the data in a case of a data breach. As a matter of fact, a recent effort [18], which enables users to become aware of third-parties with access to their (private) social information, has been met with surprise regarding the amount of data being exposed and the type of permissions granted.

Propagation of advertisements. Third-party websites may request permission to access and act upon a user's social profile (e.g., upload content to it) even when the user is not accessing the third-party site. Such actions may frequently take the form of explicit or implicit advertisements, not necessarily approved by the user.

Disclosure of users' credentials. Once a large number of applications start receiving credentials to access a user's profile, such credentials may be subject to loss or theft, or accidental leakage. Indeed, recent reports by Symantec suggest that such Facebook applications accidentally leaked access to third parties [19].

Reverse Sign-on Semantics. When a service prompts a user to sign on, he provides his credentials and gains access to data offered by that service. However, in the cases described above, the service is the one being given access to the data of the user, and from that data selects information that may be used to identify or authenticate the user.

Although a user could theoretically deny this social login approach, and the installation of the third-party application, many websites respond to this disapproval usually by diminishing the user's browsing experience significantly, cutting the user off from the largest part of their sophisticated functionality. Although this might be less of a problem if only a handful of third-party websites used this single sign-on mechanism, recent results suggest that more than two million websites have added Facebook social plugins [20]. To make matters worse, popular websites seem to adopt Facebook social plugins even more aggressively. Indeed, as of February 2012, as many as 27% of the top 10,000 most popular websites have adopted Facebook social plugins, a whooping 300% increase compared to May 2010 [1]. If this trend continues, as it appears to be, then it will be very difficult for users to browse a significant percentage of the websites without revealing their personal information.

4.2.1 Background

In this section we provide some background information on the OAuth protocol [16], which is the primary method for implementing single sign-on functionality across multiple websites. We also detail Facebook's single sign-on platform [6], which as of January 2012 is the most popular social login platform with more than 2.5 million websites using it [9].

4.2.1.1 OAuth Protocol

The OAuth or Open Authentication protocol [16] provides a method for clients to access server resources on behalf of a resource owner. In practice, it is a secure way for end users to authorize third-party access to their server resources without sharing their credentials.

As an example, one could consider the usual case in which third-party sites require access to a user's e-mail account so that they can retrieve his contacts in order to enhance the user's experience in their own service. Traditionally, the user has to surrender his username and password to the third-party site so that it can log into his account and retrieve that information. Clearly, this entails the risk of the password being compromised.

Using the OAuth protocol, the third party registers with the user's e-mail provider using a unique application identifier. For each user that the third-party requires access to his e-mail account, it redirects the user's browser to an authorization request page located under the e-mail provider's own domain, and appends the site's application identifier so that the provider is able to find out which site is asking for the authorization. That authorization request page, located in the e-mail provider's domain, validates the user's identity (e.g., using his account cookies or by prompting him to log in), and subsequently asks the user to allow or deny information access to the third-party site. If the user allows such access, the third-party site is able to use the e-mail provider's API to query for the specific user's e-mail contacts. At no point in this process does the user have to provide his password to the third-party site.

4.2.1.2 Facebook Authentication

Facebook's social login platform, known as Facebook Connect [6], is an extension to the OAuth protocol that allows third-party sites to authenticate users by gaining access to their Facebook identity as described in the previous chapter. This is convenient for both sites and users; sites do not have to maintain their own accounting system, and users are able to skip yet another account registration and thereby avoid the associated overhead.

A "login with Facebook" button is embedded in a third-party website, and once clicked, directs the user's browser to a Facebook server where the user's cookies or credentials are validated. Upon successful identity validation, Facebook presents a "request for permission" dialog where the user is prompted to allow or deny the actions requested by the third-party website, e.g., social plugin interactions or access to various information in the user's social profile. However, the user is not able to modify or regulate the third-party website's requests, for instance to allow access to only a part of the information the site is requesting. If the user grants permissions to the site's

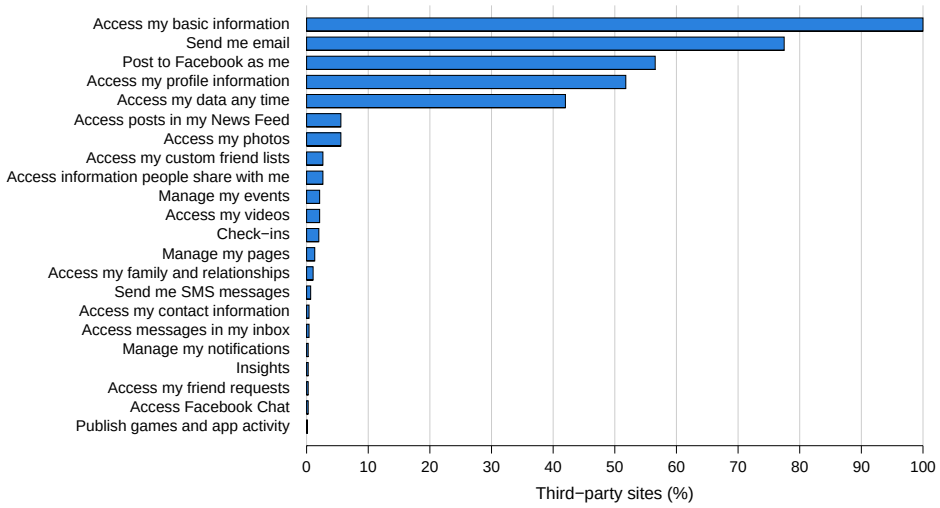


Figure 4.3: Distribution of requested permissions for a set of 755 websites that have integrated Facebook's single sign-on platform.

request, Facebook will indefinitely honor API requests originating from that third-party site, that conform to what the user has just agreed upon.

4.2.2 Social Login vs. User Privacy

To gain a better understanding of the type and extent of the permissions requested by third-party websites through the Facebook Connect mechanism, also known as "login with Facebook," we studied a random sample of 755 sites that have incorporated Facebook's social login platform. Figure 4.3 presents the frequency distribution of the different permissions requested by these websites. A full list of the available permissions can be found through Facebook's developer page [5].

One may notice that all sites request access to a user's basic information. That is the minimum amount of private information a user must disclose, even if a third-party website does not really need all that information. According to its description, the basic information includes the "user id, name, profile picture, gender, age range, locale, networks, user ID, list of friends, and any other information they have made public."

Besides the basic profile information, the administrator of the third-party website may explicitly ask for additional permissions to access more user information or perform certain actions on behalf of the user. For instance, 77% of the studied sites request access to the user's e-mail address, 57% are able to post content on behalf of the user, and more than 42% require to be able to indefinitely access user information even when a user is not using the application.

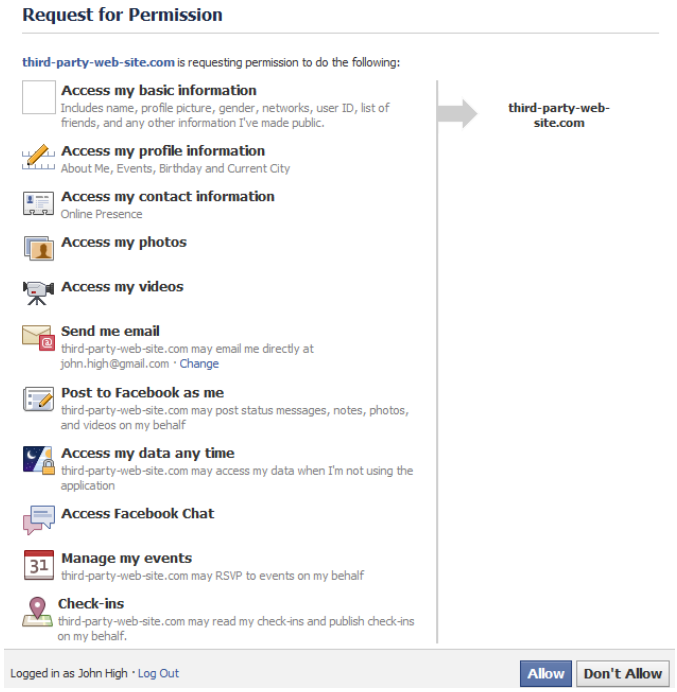


Figure 4.4: Example case of a third-party website requesting permission to access and manage an excessive amount of personal user information. The user can only allow everything or nothing (thus aborting the social login). Any kind of fine-grained control over the permissions is absent.

Moreover, permission to manage Facebook notifications could enable malicious third-parties to hide the misuse of other permissions granted to them. What is more, access to direct messages sent or received and Facebook's real-time chat system, could seriously compromise a user's private communications. Finally, special consideration should be given to permissions that may result in real-world consequences for the user; the ability of a third-party to access information about the user's physical location (*"Check-ins"*) or send SMS messages, which may result in monetary charges.

We argue that in most of the cases the type of permissions and the amount of information requested from the user during social login are more than necessary. Even with benign third-parties, the more personal data being shared, the greater the damage in case of leaks either accidental or as a result of an attack. To give an example, one of the cases in our study is a music band which urges its fans to perform a social login when visiting its site. Although we could not confirm the presence of functionality dependent upon social login, we will give the site the benefit of the doubt. However, its requirements are over the top. It requests access to basic, contact and profile information, photos and videos, to the user's e-mail address and Facebook

chat. Moreover, such access is requested even if the user is not using the site. Finally, it requests the ability to upload content to Facebook on behalf of the user, read and manage the user's events and reports on his physical location. Figure 4.4 is a screenshot of the social login dialog for that site. Its name has been anonymized. Access to all of the user's photos and videos is unjustified as is access to the user's private conversations. Furthermore, the ability to impersonate the user on Facebook is in no way restricted to purposes related to the nature of the third-party. Finally, managing all events and physical location information so it can for instance generate activity related to the band clearly demonstrates the need for fine-grained permissions. Ideally, the third-party would request access to photos tagged with a certain keyword related to the band, manage events and locations with specific prefixes in their names and add a "uploaded by X on behalf of user A" label to content uploaded on Facebook.

Overall, the above study confirms our intuition that the amount, type, and combination of permissions requested by third-party sites can seriously put users in a compromising position. At the same time, user reactions to a recent effort [18] that enables users to become aware of third-party applications and websites with access to their (private) social information, confirm the general request for improved control and better protection over the data one uploads to a social network. Facebook itself acknowledges the issue and, in a slight effort for remedy, offers users the option to anonymize the e-mail address they surrender to third-parties. This option is unfortunately opt-in and enabled by default in rare occasions driven by abuse-related heuristics.

5 Social snapshots

A different aspect from those presented in the last three chapters is discussed here. When dealing with social network research, it is not always feasible to operate directly on live data. To avoid being banned for abusive or overly extensive use, it can be convenient to take a *snapshot* of the network's structure and the participant's relationships at a certain time. All Online Social Networks (OSNs), such as Facebook or LinkedIn, contain sensitive and personal data of hundreds of millions of people, and are integrated into millions of other websites [53] as discussed in the previous chapter. Studies focused on security issues that are associated with OSNs [34, 59, 72, 69, 112] highlight challenges to the security and privacy of social network users and their data.

These, and similar studies, heavily depend on datasets that are collected from the social networking websites themselves, often involving data that is harvested from user profiles. Furthermore, as social networks continue to replace traditional means of digital storage, sharing, and communication, collecting this type of data is also fundamental to the area of digital forensics. For example, data from OSNs have been used successfully by criminal investigators to find criminals and even confirm alibis in criminal cases [44, 108].

While traditional digital forensics is based on the analysis of file systems, captured network traffic or log files, new approaches for extracting data from social networks or cloud services are needed. Interestingly, little academic research aims at developing and enhancing techniques for collecting this type of data efficiently. Despite the growing importance of data from OSNs for research, current state of the art methods for data extraction seem to be mainly based on custom web crawlers. However, this naïve approach seems to have a number of shortcomings:

- High network traffic: The extraction of profile data via traditional web crawling can be regarded as costly with regard to the required network

resources, as it typically incurs a large amount of HTTP traffic and causes a high number of individual network connections. Apart from inherent disadvantages, social networking websites may also choose to block network access for clients that cause high levels of traffic, thus preventing them from harvesting additional data.

- Additional or hidden data: Per definition, web crawlers can only collect data that is accessible on the target website. However, social networks often publish interesting meta-information (e.g. content creation timestamps or numeric identifiers) in other data sources, for example via a developer APIs.
- Maintainability: The structure and layout of websites tend to change unpredictably over time. Additionally, the increasing use of dynamic or interpreted content (for example, JavaScript) leads to high maintenance requirements for custom web crawlers.

This chapter introduces a novel method for data collection from social networks that aims to overcome these problems. The approach is based on a hybrid system that uses an automated web browser in combination with an OSN third-party application. Results show that the developed system can be used efficiently to gather "*social snapshots*", datasets which include user data and related information from the social network. Unfortunately, this research is a double-edged sword because it is not only an advantage for researchers and those with benign intent, but also a possibility for an attacker to gather a rich dataset with details information about a multitude of users. How the information is used is, however, not of immediate concern in this chapter. What we discuss here are merely technical aspects.

5.1 OSN data harvesting

The application enables taking a snapshot of a given online social network account including meta-information, a method we termed "social snapshot". Meta-information such as exact timestamps are not available to the user via the user interface of the web application. A social snapshot represents the online social networking activity of a specific user such as circle of friends, exchanged messages, posted pictures etc. Due to the diversity of information available via OSNs we propose a twofold approach: an automated web-browser in combination with a custom third-party application. The social snapshot application is initialized with a user's credentials or authentication cookie. In the following, a custom third-party application is temporarily added to the target account. This application fetches the user's data, pictures, friend list, communication, and more. Information that is unavailable through the third-party application is finally gathered using traditional web-crawling techniques. By automating a standard web-browser and avoiding

aggressive web-crawling the behavior of a human OSN user is simulated, thus minimizing the risk of being blocked by the social networking site.

5.1.1 Social Snapshot Framework

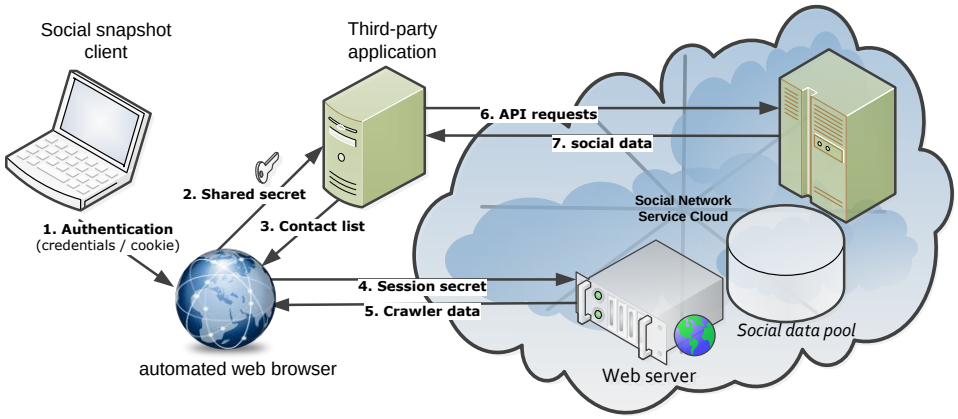


Figure 5.1: Collection of digital evidence through our social snapshot application.

Figure 5.1 shows the core framework of our social snapshot application. **(1)** The social snapshot client is initialized by providing the target user's credentials or cookie. The tool then starts the automated browser with the given authentication mechanism. **(2)** The automated browser adds our social snapshot application to the target user's profile and sends the shared API secret to our application server. **(3)** The social snapshot application responds with the target's contact list. **(4)** The automated web browser requests specific web pages of the user's profile and her contact list. **(5)** The received crawler data is parsed and stored. **(6)** While the automated browser requests specific web pages the social snapshot application gathers personal information via the OSN API. **(7)** Finally the social data collected via the third-party application is stored on the social snapshot application server.

5.1.2 Authentication

In order to get access to the complete content of a target's social network account, social snapshots depend on gathering the initial authentication token. In the following, we outline three scenarios that explain how this initial gathering of the authentication token works and that are representative for real-world use cases.

Consent. This naïve approach requires consent from the person whose social networking profiles are analyzed. A person would provide the rese-

archer temporary access to her social networking account in order to create a snapshot. This would also be the preferred method for academic studies to conduct this research in an ethically correct way and to comply with data privacy laws. This method is used for the evaluation of the proposed application as further described in Section 5.2.

Hijack social networking sessions. The social snapshot application provides a module to hijack established social networking sessions. An attacker would monitor the target's network connection for valid authentication tokens, for example unencrypted WiFi connections or LANs. Once the hijack module finds a valid authentication token, the social snapshot application spawns a separate session to snapshot the target user's account.

Extraction from forensic image. Finally, physical access to the target's personal computer could be used to extract valid authentication cookies from web-browsers. Stored authentication cookies can be automatically found searching a gathered hard drive image or live analysis techniques such as Forenscope [39].

5.1.3 Depth of Information Collection

Starting from a target profile, a number of subsequent elements become available for crawling such as the user's friends, uploaded photos and joined groups. With these elements, again, a number of subsequent elements can be accessed. For example, the single-view page of a photo can contain comments and likes of other users, who do not necessarily have to be direct friends of the owner of the photo. Additionally, users can be tagged in photos. These are all starting points for further crawling. The same applies for groups; A group gives access to the profiles of all group members, photos with users tagged, who are potentially not members of the group, and so forth. Consequently, a social snapshot of a single user does not only obtain the user's data and data of her friends, but its depth can reach a high value. Thus, the depth of the social snapshot is an essential configuration option which controls the social snapshot's extent. Figure 5.2 shows an example of a social snapshot with $depth = 2$. For a given user all of her friends are first fetched, followed by the friend's photos. The single path for photos of the friend's user illustrates the magnitude of available paths and thus data. Defining a specific social snapshot depth enables us to limit the amount of fetched data. The amount of data grows exponentially with social snapshot depth.

It is important to note that the relevance of data is not the same for different elements. For example, tagged users in a photo are most likely in a closer relationship to the owner of the photo than two users that joined the same group, just because of similar interests. Therefore, the social snapshot tool prioritizes element types that suggest higher data relevance and uses

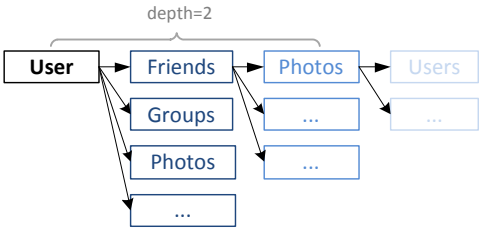


Figure 5.2: Example for elements fetched with social snapshot of depth=2

them as a starting point of each iteration. The prioritization is performed on the basis of predefined priority flags in the third-party application.

5.1.4 Modules

The social snapshot application consists of a number of modules being described in the following. The core modules are the automated web browser and the custom third-party application as outlined in Figure 5.1.

Social snapshot client. The social snapshot client module initializes the data gathering process with a given user's credentials or cookies. Once started, the client first authenticates itself against the target online social network. In the following, the client automatically adds the custom third-party application with the highest possible permissions to the target's account. Information that cannot be retrieved through the third-party application is crawled and parsed by the client. Once all information has been retrieved, the client removes the third-party application and logs out of the given social networking account. The interaction with the social network as well as web-crawling is performed by the Selenium framework [98] being described in the following. The social snapshot client is implemented in Java and the module offers a command line interface.

Automated web browser. The browser module is responsible for the basic interaction with the target online social network. The Selenium testing framework [98] is leveraged to automate the Mozilla Firefox browser. Selenium comes with a command line server that receives Selenium commands. Therefore, the framework can be used to script the behavior of an average user using her Firefox web-browser to surf a social networking website. One initial obstacle had to be overcome, though: cookie authentication with Selenium which was not supported out-of-the-box. The original Java source code of the command line server had to be patched to be able to correctly set HTTP cookies for the cookie authentication mode.

Third-party social snapshot application. The OSN social snapshot application is a third-party application, which sole purpose consists of gathering all possible account data through the target OSN's API. The main design goal of our third-party OSN application is performance, thus multiple pro-

gram threads are used to gather information as quickly as possible. The third-party application can be configured to prioritize specific account data and to download only a predefined set of account artifacts (social snapshot depth).

Hijack. The hijack module is a network sniffer module that collects valid OSN HTTP authentication cookies from sources such as LAN or WiFi connections. The hijack module is built on the basis of Mike Perry's modified libpkt library[99], which works out of the box with LAN, unencrypted WiFi, and WEP encrypted WiFi connections. It offers a command line interface and is implemented in Python.

Digital image forensics. The digital image forensics module matches image files gathered from online social networks with their original source. The goal is to find the pristine image of a compressed picture extracted through our social snapshot application. All images are initially clustered according to their color histograms, rescaled and compressed to the target picture size, and finally matched with pattern recognition techniques. As social networks typically remove meta (EXIF) information of uploaded images this module is helpful in finding the source of collected pictures from OSNs and thus restore information such as the original image creation time, camera model etc.

Analysis. The analysis module is a parser for the results gathered with the data collection modules of the application. It parses the crawled data as well as the information collected through the OSN's API. Furthermore, the analysis module fetches additional content such as photos that are openly available by knowing the URI from online social networks. Finally, it generates a report on the social snapshot data. The analysis module can be used to generate exact timelines of communication, metadata summaries, e.g. of pictures, a weighted graph from the network of friends, or their online communication.

5.2 Results

This section describes the evaluation of the social snapshot application. The generic social snapshot approach is applicable to the majority of today's social networking services. The sole requirement for target social networks is the availability of a developer API or the adaption of our automated browser.

5.2.1 Social Snapshots on Facebook

At the time of writing Facebook is the most popular online social network with a claimed user base of over 800 millions of users [53]. Furthermore, Facebook supports third-party applications and user profiles contain a plethora of information. Thus Facebook was chosen to evaluate the social snapshot

tool. Third-party applications on Facebook have access to account data via the Graph API[52]. Almost the entire account data of Facebook users and their contacts are made available through their API. Facebook solely makes sensitive contact information such as phone numbers and e-mail addresses inaccessible to third-party applications. Hence, the social snapshot client crawls the contact information of Facebook profiles, while all remaining social data is fetched through a custom third-party application. In October 2010, Facebook introduced a download option[54] that enables users to export their account data. Table 5.1 outlines the different profile content ele-

Element	**Download**	**social snapshot**
Contact details	–	✓Crawler
News feed	–	✓Graph API
Checkins	–	✓Graph API
Photo Tags	–	✓Graph API
Video Tags	–	✓Graph API
Friends	name only[a]	✓Graph API
Likes	name only[a]	✓Graph API
Movies	name only[a]	✓Graph API
Music	name only[a]	✓Graph API
Books	name only[a]	✓Graph API
Groups	name only[a]	✓Graph API
Profile feed (Wall)	limited[b]	✓Graph API
Photo Albums	limited[b]	✓Graph API
Video Uploads	limited[b]	✓Graph API
Messages	limited[b]	✓Graph API

[a] No additional information available.
[b] Missing meta-information such as UIDs.

Table 5.1: Account information available through social snapshots compared with Facebook's download functionality.

ments gathered through the social snapshot application as compared with Facebook's download functionality. As shown in Table 5.1, the download functionality only offers a very limited representation of a user's online activity. For example, for a given user's friends, only their ambiguous names are made available and no information on the activity of a given user's friends is included.

5.2.2 Hardware and Software Setup

To test the functionality of the social snapshot application, we developed a third-party application for Facebook based on their PHP Graph SDK. One of the main modifications we performed on their original library was the support for multi-threaded API requests. The third-party social snapshot application for Facebook is thus able to handle a number of predefined API requests simultaneously. The single requests are hereby pushed on a request queue with a specific priority. Hence the third-party application can be configured to, for example, fetch private messages before user comments of a Facebook group. The extent/depth of social snapshots can be further configured as a parameter for our third-party application. We deployed it on a Linux server in the TUV's network.

The third-party application fetches Facebook elements of a given account and stores them as separate JSON files. The separate JSON files correspond to specific requests, whereas the files are named as follows. The first part of the JSON file name is the ID of an API object while the second part specifies the requested connection detail. For instance, "123456789∼friends.request" contains all friends of the object with ID 123456789 formatted as a JSON object. In order to improve the performance of the application, it is configured not to download any videos or photos through the Graph API directly. As the third-party application collects direct links to photos, the digital image forensics module was configured to download photos during the analysis phase. Once the third-party application is finished fetching account data, it creates a tarball containing the social snapshot data.

The social snapshot client was adapted to fetch contact details of given user profiles and automatically add the third-party application to a target account. One particular challenge that had to be overcome was to reliably obtain the list of friends of a given target account. This was hindered by the changing layout of the friend lists as well as Facebook only displaying a random subset of friends at a given time. The solution was to fetch it through the third-party application and send the profile links back to the client. The client generates requests for every friend of the target user and sends them to the Selenium server that automates a Mozilla Firefox browser. The responses from the automated web browser module are parsed by the client and the contact information is extracted with a set of XPath queries. The client finally creates a CSV-file containing the contact information of all users. The client application was deployed in a virtual machine with a standard Ubuntu Desktop that runs the patched Selenium server. The social snapshot analysis module implements both a parser for the fetched JSON Graph API requests as well as for fetched CSV contact details. The analysis module merges the results from the social snapshot client and the third-party application into a single database. The analysis module is implemented in Java.

Furthermore, the digital image forensics module was extended to automatically search a social snapshot for photo links, which it automatically downloads from the Facebook content distribution network. The hijack module did not require any Facebook specific modifications as it simply strips cookies of a given domain from a monitored network connection.

5.2.3 Test Subjects and Setting

Human volunteers were recruited via e-mail, describing our experiment setting. The e-mail contained the experiment instructions and a briefing on how their personal information is going to be stored and analyzed. Furthermore, volunteers were briefed on the ethics of our experiment: no Facebook account data is modified, the social snapshots are stored in an encrypted filecontainer, no personal information is given to third-parties nor published. The invitation to support this first social snapshot evaluation was sent to researchers and students in computer science. Finally 25 people gave their consent to temporarily provide us access to their Facebook accounts. Volunteers temporarily reset their Facebook account credentials, which were used to create a social snapshot of their accounts. Once a social snapshot had been created, the test group was notified to reset their account password.

The third-party social snapshot application was configured to fetch an extensive account snapshot. Experiments showed that 350 simultaneous API requests lead to the best performance results in a series of indicative experiments we conducted beforehand. The third-party application was configured to fetch the following elements recursively:

- Highest priority ($priority = 3$)
  inbox, outbox, friends, home, feed, photos, albums, statuses

- Medium priority ($priority = 2$)
  tagged, notes, posts, links, groups, videos, events

- Lowest priority ($priority = 1$)
  activities, interests, music, books, movies, television, likes

The priority settings ensure that important information is fetched first. Account elements with highest and medium priority are fetched with $depth = 2$ while elements with the lowest priority are gathered with $depth = 1$. Thus a social snapshot of a given user includes for example, her friend's groups, tagged pictures, links etc. but no pictures, comments, etc. are downloaded from her favorite television series. These social snapshot settings imply that not only the target's account is completely fetched but also social data on the targets' friends is collected.

5.2.4 Results on Social Snapshot Performance

Figure 5.3 illustrates the time required by our third-party social snapshot application to snapshot the test accounts through the Graph API. The third-party application required on average 12.79 minutes. Account elements of the test accounts were on average fetched with $93.1kB$ per second.

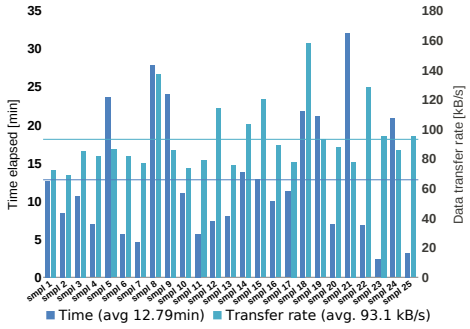


Figure 5.3: Required time and transfer rate of our social snapshot third-party application.

The time required for crawling contact details with our automated web browser is outlined in Figure 5.4. Test accounts have been crawled within 14 minutes on average. The average elapsed time per account corresponds to 3.4 seconds per user profile page.

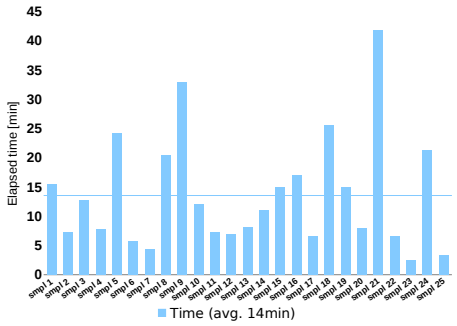


Figure 5.4: Time required for crawling contact details with social snapshot client and automated web browser.

5.2.5 Results on Social Snapshot Completeness

As illustrated in Figure 5.5, the third-party application found and fetched on average $9,802$ Facebook account elements per test subject. The storage size of the fetched JSON files accounted to $72.29MB$ on average. Listing

5.1 shows an anonymized example from the fetched Facebook account elements. The example represents the basic information fetched of the user "John Doe" formatted as a JSON object. This example request also highlights that account data fetched through the Graph API provides a richer information set for further investigations. The standard web interface does not provide information if a user's account is verified nor an update time that is accurate to the nearest second with information on the used time zone.

```
1 {"id":"12345678","name":"
    John Doe",
2 "first_name":"John","
    last_name":"Doe",
3 "link":"http:\/\/www.
    facebook.com\/johndoe",
4 "username":"johndoe","
    birthday":"04\/01\/1975
    ",
5 "hometown":{"id":"","name"
    :null},
6 "quotes":"social snapshot
    your account!.\n",
7 "gender":"male","email":"
    johndoe@example.com",
8 "timezone":2,"locale":"
    en_US","verified":true,
9 "updated_time":"2011-05-15
    T13:05:19+0000"}
```

Listing 5.1: Example of collected JSON element

Compared to data collected via the standard web interface, the social snapshot contains a number of additional information tokens. Most notably for forensic investigation is the availability of exact creation timestamps through the Graph API. We used the image forensic module to download all unique photos in the highest available resolution from the gathered social snapshots. The downloaded photos corresponded to $3,250$ files or $225.28MB$ on average per test account.

Figure 5.6 shows the additional contact details crawled with the social snapshot client. On average, the social snapshot client had to crawl 238 profile sites per test account. For all crawled profile pages our crawler found 22 phone numbers, 65 instant messaging accounts, as well as 162 e-mail addresses on average. After a number of subsequent requests to user profiles of a given account, Facebook replaces textual e-mail addresses with images. This behavior was noticeable with our social snapshot client, whereas on average we fetched 85 e-mail addresses in image form (OCR in Figure 5.6). Due to the fact that Facebook uses e-mail addresses in image form as a web crawler protection method, the fetched images could not be directly parsed.

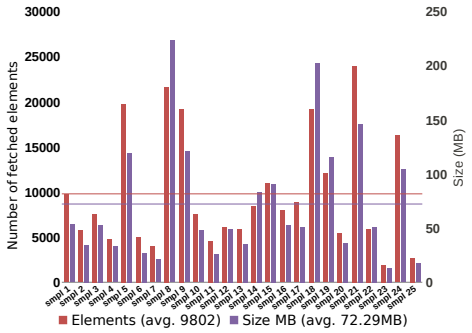


Figure 5.5: Account elements fetched through social snapshot third-party application.

Finally the analysis module was used to verify the integrity of the collected snapshots. Every entry in our fetched contact details CSV files had correspondent entries within the retrieve JSON files. Apart from that no invalid responses were received through the Graph API. We furthermore implemented a mechanism for the analysis module to overcome the obstacle of parsing image e-mail addresses. By providing Facebook's e-mail image creation script the maximum possible font size of 35 instead the default of 8.7, higher resolution versions of the e-mail address pictures could be fetched. Thus GNU Ocrad[58] could be used to resolve these high resolution images into their textual representation. The idea of replacing the default font size with a larger one was first described in [103] and we could successfully verify that the described method still applies.

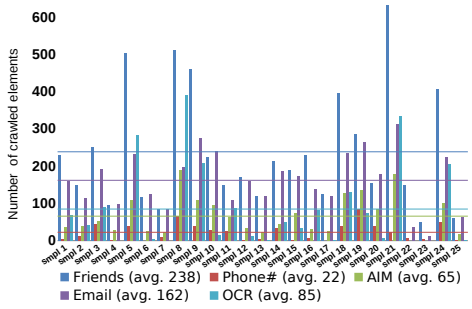


Figure 5.6: Contact details crawled with social snapshot client and automated web browser.

5.2.6 Indicative Cookie Authentication Experiments

We performed a number of indicative experiments to verify our cookie authentication method on Facebook. Both non-persistent as well as persistent cookie authentication is available. Persistent cookies are valid for 30 days in the case of Facebook. The social snapshot tool with the hijack module was successfully tested on a number of non-persistent users over an unencrypted test WiFi network. Furthermore, the social snapshot application was validated with persistent cookies extracted from web browser profile files. In the case of one particular test setting, namely the TUV campus WiFi, as many as 50 valid social networking sessions could be observed within one hour.

5.3 Discussion

The evaluation required on average 9,802 API and 238 HTTP requests to successfully snapshot an entire social networking account in less than 15 minutes. With traditional web-crawling more than 10,000 HTTP requests are necessary to snapshot a single test account. The generated network traffic of traditional web-crawling would have been likely detected and blocked by social networking providers. Moreover, the evaluated approach retrieved the great majority of social networking account data without the requirement of additional parsing and with exact timestamps. During the implementation of our social snapshot techniques, Facebook's web-site layout changed a number of times. Since only contact details were crawled, the parser of our client could be promptly adapted, while the third-party application did not require any changes at all. As Facebook has no review process for third-party applications we could also make the third-party application available straightforward. Third-party applications on Facebook do not even have to appear in their application directory in order to be usable.

Social snapshots could also be used to raise user awareness. Users would run our social snapshot tool and get a report on their account data. Thus, social networking users could sight the magnitude of information that is stored with their social networking providers. We hope that this would help the average social networking user to make better informed decisions on which information they post.

Unencrypted social networking sessions pose a serious security threat. Since HTTPS is not enabled by default on today's social networking services, user sessions can easily be hijacked. Two proof-of-concept tools have been released that make session hijacking of social networking sessions available to the average user. *Firesheep* [37] has been released in October 2010 as a browser extension and at the time of writing is not functioning anymore. *Faceniff* [101] offers a point-to-click interface and supports a number of wireless network protocols. It is an Android application for hijacking

social networking sessions released in June 2011. Both hijacking applications were released in order to create awareness for the problem of insecure social networking sessions. It is trivial however to couple such simple hijacking applications with our social snapshot tool. Thus, attackers could harvest complete account snapshots in an automated fashion. It has been shown [69] that the large amount of sensitive data stored in social networks could be used for large-scale spam attacks via session hijacking.

6 Gaming and other platforms

The technique to take a snapshot as described in Chapter 5 is essentially restricted to those networks that we entitled as being *pure* in the Introduction of this deliverable. In this chapter, we discuss a different form of online community: games. At first glance it may seem that it is out of scope for this document but in fact, online gaming platforms like steam [66] or origin [68] implement a lot of functionality from ordinary SN's. Some examples are: Friend finder, online status, friends list, photos, history and many more.

Let's take a look on the targeted asset from an attackers point of view. On dedicated social network, the target is most probably personal information which can in turn be used to leverage known forms of attacks. On gaming platforms, the asset can be twofold. First, almost all accounts come with attached payment information. A huge difference to Facebook, for example. Recent events have shown that an attack on this data has to be counted with. In November 2011, Gabe Newell, CEO of Valve Software and the Steam online platform posted the following announcement [56]:

> We learned that intruders obtained access to a Steam database in addition to the forums. This database contained information including user names, hashed and salted passwords, game purchases, email addresses, billing addresses and encrypted credit card information. We do not have evidence that encrypted credit card numbers or personally identifying information were taken by the intruders, or that the protection on credit card numbers or passwords was cracked. We are still investigating.
>
> We dont have evidence of credit card misuse at this time. Nonetheless you should watch your credit card activity and statements closely.

> While we only know of a few forum accounts that have been compromised, all forum users will be required to change their passwords the next time they login. If you have used your Steam forum password on other accounts you should change those as well.
>
> We do not know of any compromised Steam accounts, so we are not planning to force a change of Steam account passwords (which are separate from forum passwords). However, it would not be a bad idea to change that as well, especially if it is the same as your Steam forum account password.

Although such an attack can be severe in its impact, the mechanics behind it are not especially surprising. The second, and research-wise more interesting asset is embodied by in-game investments. In the end, the work and effort put into developing a character or advancing a game can and is measured in real-world money.

Online games, and especially multiplayer online games, ran through an astonishing development in the past ten years. With November 2010, the renowned massive multiplayer online game (MMOG) *World of Warcraft* hit the 12 million subscribers mark and is still gaining new users every day [67]. Considering all major games, a total of over 23 million people around the globe participate in multiplayer online games [67]. At the same time, the demand to protect these games and especially the participating players is increasing likewise.

One major problem comes along with almost every MMOG that exists today: *Bots*. Where online games are concerned, the term *bot* usually entitles a program or method that allows a user to play the game partially or completely unattended. At first thought this might not seem like much of a problem, but the impact of bots on the game world and its population can be very serious. Gold farmers, for instance, which produce in-game currency at a far higher rate than ordinary players ever could, alter the prices of traded goods such that casual players cannot afford them anymore. As a result, in-game currency and items are traded via ebay and payed for with real money. At the time of this writing, 5.000 pieces of *World of Warcraft* Gold were worth roughly 20 US Dollars. A sum that many consumers deem worthy to make up for the additional spare time. While automatic gathering of resources, also called farming, is certainly one application of game bots, this particular segment is far better covered by human players in low-wage countries [48]. Besides, participating players are only secondarily affected by gold farmers when they experience raising prices. Furthermore, the game company can introduce money sinks to stabilize these effects.

A far more annoying application of bots is their deployment as level-bots or active team members in battle groups. The restricted action set of these programs and their inability to properly react to its surroundings may

work a dedicated player's last nerve. Currently, human players depend on the game company to take care of bots flooding their servers. To counter this issue, several detection techniques are being researched to distinguish genuine players from game bots by comparing their action sequences.

6.1 Inflation

The motivation for using a bot is simple. It produces an unfair advantage for the player compared to other participants and, therefore, increases the game asset compared to other players. Similar to other games, combat interaction yields the best results in *World of Warcraft*. Killing monsters is the easiest way to gain experience, gold and resources. Once the maximum level is reached, additional items can be earned by engaging in Player versus Player (PvP) combat. This is perilous terrain for bot programs simply because the program's actions (movement and action sequences) are visible to other players of the same team. And these players usually tend to be peeved when two players out of a ten player team are bots. It drastically lowers the chances of winning an encounter, thus causing grief most gamers are not willing to accept. Once a player files a complaint, game companies do not hesitate to suspend the account in question or ban it completely. Unfortunately, spotting a bot based on its visible actions is not trivial. PvP combat tends to be hectic, restricting the time human players can spend watching teammates and judging their actions. Even in the smallest battleground, a bot with a decent movement and action pattern [14, 21] is very hard to distinguish from human players. As a result, every bot comes with some inherent weaknesses which can be used in a detection attempt.

6.1.1 Waypointing

Compared to a human, a gaming bot is very limited in its capabilities to navigate the virtual environment. Modern MMOGs provide a rich, three dimensional game world where movement can occur in every direction. Additionally, certain obstacles and hindrances can exist, that prevent a player from getting there. Consequently, an implementation with random movement patterns is not an option, because it can easily result in a trapped character or an avatar that runs against a wall for hours. To prevent this, a waypoint system is mandatory. The desired path can either be recorded by the player or is shipped with the bot program. When starting the bot, the character traverses more or less the same path, with small variances depending on the environment and the bots sophistication level. This fact is actually a huge limitation because it renders a bot detectable if the traveled path is properly analyzed[95]. In smaller environments, like battlegrounds,

such a detection technique is unfeasible due to the limited time the players spend in the instance and the relatively small size of the environment.

6.1.2 Sequencing

Just like the waypoint system, an element every bot has in common is the reaction to certain events. The main goal still is to kill enemies within the game. The decision which spells to launch or which skills to use to successfully reach this goal is again pre-defined in the bot's configuration. The sum of all abilities used from the point where an enemy is engaged until the enemy (or the own character) dies is entitled as a *combat sequence*. A side-effect of the fixed configuration is, that these sequences are always very similar. Compared to a human player, the bot can make no, or very restricted choices based on the game environment. As a result, the abilities used to kill a single enemy in the game will always be very similar. This is even more blatant in PvP combat, where players use a very large variety of different skills in various combinations to thwart their enemy. Based on this assumption, a behavior-based analysis of the player's actions can be implemented as a countermeasure [100].

6.2 Other approaches for cheat detection

The two previously mentioned weaknesses are the basis for most research conducted in this area. In [42] [43], for instance, a system is introduced which aims to detect bots by tracking their path information and trajectory respectively. Although designed as a general-purpose solution for all kinds of games, it is ultimately restricted to a very narrow category of fast-paced action games, where turning the avatar with the keyboard is not an option.

In [113] [63], the authors propose the use of *captchas* that automated scripts cannot solve. The same principle is of course possible in MMOGs. The analogy would be a maze, where an avatar had to find the exit or overcome certain obstacles. Such an approach would certainly work but the number of players that decide to quit because they were unexpectedly trapped inside a maze while in the middle of PvP combat is one of many reasons why no game company would even consider such a method. In [41], the authors utilize a traffic-analysis approach to identify bots for the game "Ragnarök Online". A major drawback of the proposed method lies in the fact that it is custom-made for that game. The authors try to distinguish traffic generated by the official game client from traffic generated by standalone bot programs through statistical analysis of packet transfer properties. Unfortunately, this approach does not work on modern games, as they mostly implement largely ping-independent command queueing on the client-side.

This fact also renders the method described in [77] ineffective, where the timing of keystrokes is used to distinguish between humans and bots.

A more general view on security in online games is presented by Greg Hoglund and Gary McGraw [64]. In their book, they cover a wide section of game security topics, ranging from the legal issues over bug exploits and hacking game clients to writing bots. Although it provides a good introduction into several gaming security areas, it concentrates mostly on the attacker's point of view and does not provide concrete solutions on how to detect or prevent botting.

Another piece of related work was done by Jeff Yan et al. [114], where the most important cheating methods were analyzed and categorized. This work can be seen as the theoretical foundation for cheat detection. In the end, a bot is just another form of cheating. In [115] the concept was extended and a detection approach for aiming bots was introduced. Other than bots in MMOG's, aimbots are deployed in games where heightened reaction is of advantage, like in first person shooters for example. Time is not a factor for queue-based games, however. Even with a 200ms delay, those games are designed to work without any drawbacks.

6.3 Bot Detection

One approach to detect bots is based on the combat sequence each avatar produces when engaging an enemy. For each fight, a list of actions is extracted. The difference between subsequent combat sequences is measured in terms of their levenshtein distance [87]. Killing each monster with the same combination of skills would therefore result in a distance of 0. The similarity value is calculated as v_i for a particular combat sequence c_i following the formula

$$v_i = \frac{\sum_{j=(i-k)}^{i-1} d_{levenshtein}(c_i, c_j)}{k},$$

with d being the levenshtein edit distance. This method takes an interval of size k before the combat sequence in question and averages the levenshtein distances to the current one (i).

The final goal is, to detect all tested game bots and successfully distinguish them from humans playing the game.

6.3.1 Implementation

To gather the data, the logging facility of a MMOG usually suffices. This logging facility is primarily designed to enable backtracking through fight sequences. It is heavily used by large player communities to decipher who messed up a fight in dungeons where 40 people are playing in a single raid group. A nice side-effect of this logging facility is, that it keeps track of

other player's actions as well. Therefore, it is also possible to simply follow a character and record the produced combat sequence to find out if a bot is playing. For the server-side, it is simply a matter of logging the actions caused by a single character and, therefore, not a challenging task.

6.3.2 Experimental results

In an example implementation based on *World of Warcraft* , a set of combat sequences produced by bots and human players was used to evaluate this approach. In total, the recorded data comprised 485.761 combat actions, whereof 266.318 (55%) were produced by bots. The human traces were collected by a total of 41 individuals over a period of several weeks. The bot traces were produced by a self-written bot program (AltNav), MMOMimic[14], FairPlay[21] and ZoloFighter[22]. During the evaluation phase, the bot trace with the highest variance and the human trace with the lowest variance in combat routines was used, with the only restriction that they have to be of the same class (e.g. Fighter, Mage, etc..). Character classes exhibited different combat schemes, depending on the variety of skills available to them. Therefore, comparing them to each other is not feasible.

To get comparable results, the following steps were executed:

- Each bot was modified to include conditional actions based on player health, pet health, encountered mob etc. The goal was to make the bot behave as much as a human player as possible. It should be noted here, that the majority of bots do not use such a high level of randomization by default. They mostly incorporate a single fight sequence which is simply repeated until either the target or the player dies.
- After recording the bot's actions, it was switched over to a human player with the directive to kill the same monsters and traverse the same route.
- The levenshtein distance was calculated for each combat sequence.
- The result graph was generated to visualize the distances and decide which detection metric to apply, respectively, which evaluation strategy is the most promising.

Figure 6.1 shows a result plot for a hunter played by both, a human and a bot. The trace shows 90 minutes of actual (human) gameplay which resulted in more than 120 distinct combat sequences. Therefore, the time interval between each combat sequence is 45 seconds. The recording shows non-PvP action which happens at a far lower pace than battlegrounds. Here, k was set to 1, thus, only the directly preceding combat sequence was taken into account. For the average curve, 10 concrete distances were considered. This figure clearly shows the different styles of a bot compared to the

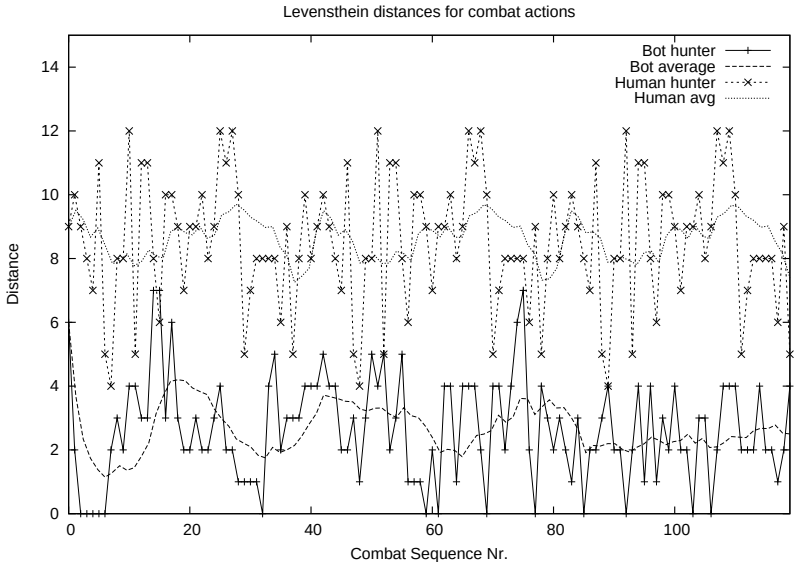


Figure 6.1: Concrete levenshtein, k=1, average=10

human player. For this type of interaction, the human player acts within a levensthein distance interval of $[4, 12]$ with an average of slightly above 8, while the bot acts in an interval of $[0, 7]$ with an average of around 3. Although feasible, a drawback of this method is that once a bot knows how it is calculated, the metric can be dodged by alternating between two completely different combat sequences, which in turn results in very high values. As a countermeasure, the *minimum* levenshtein distance was utilized for a certain interval k. The exact values v_i are processed as

$$v_i = min(d_{levenshtein}(c_i, c_j)) \quad \forall\{j|k \leq j < i; j \epsilon N\},$$

with N being the overall amount of combat sequences that have to be evaluated, and k the range again. Figure 6.2 shows the same sample as before, with the values calculated by this new method. Whenever a combat sequence is repeated within the interval k, it causes the value to be zero. With this formula, lower discrete and averaged values can be expected for humans and bots. Here though, the bot causes a certain amount of zero-passes, a human player does not, which is a good beginning for a detection metric.

Detection metric

In a first approach, the number of subsequent zero-passes caused by the player was used as an evaluation metric. Whenever more than four subsequent values were at zero, an alert was raised, and the player was identified as a bot. With this method, it was possible to detect all bots and all players

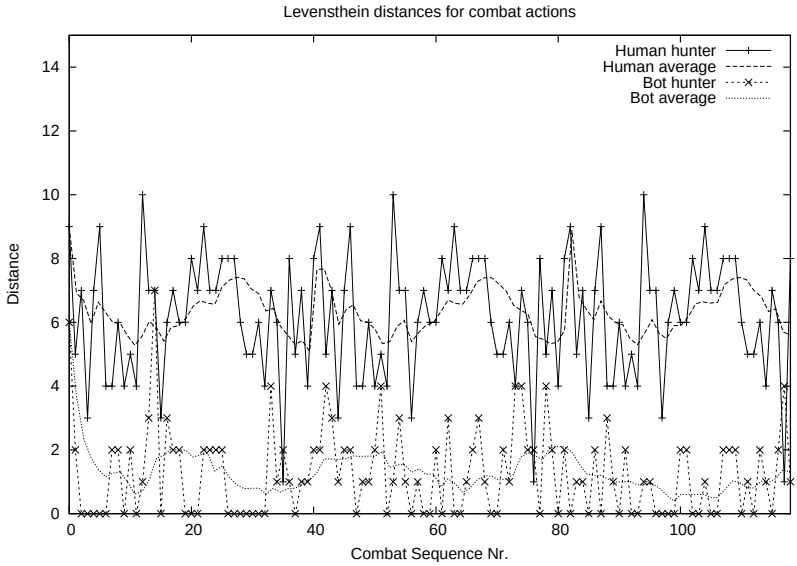


Figure 6.2: Minimum levenshtein, k=5, average=10

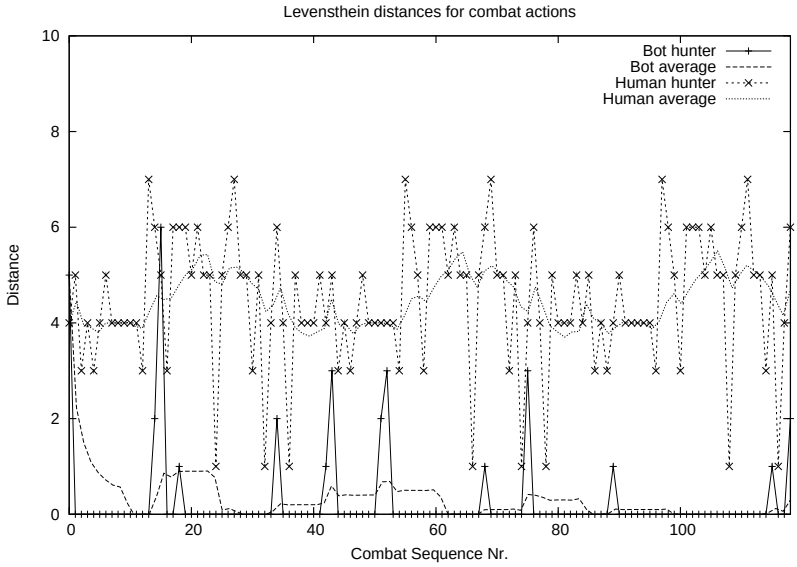


Figure 6.3: Minimum levenshtein, N_k=40, average=10

from the initial test set reliably. During the evaluation, however, a human, playing a Mage, caused one or sometimes even two zero-passes.

A more passive player would certainly be able to cause several zero-passes in a row and, therefore, cause a false positive. A solution to this problem is, to extend the above formula to:

$$v_i = min(d_{levenshtein}(c_i, c_j)) \quad \forall\{j|j \neq i; j \epsilon N_k\}.$$

N_k has to be chosen such, that it covers the desired time-frame. In Figure 6.3, it was set to 40, resulting in effectively 30 minutes of played time. The result of the formula is, that a zero value is produced whenever the same combat sequence is repeated within this time-frame. Other than before, the zero-passes of the *averaged* curve are counted. This method has a much higher accuracy rating, because the variance a human introduces, causes at least one unique combat sequence out of 10. The threshold for this method was set to four subsequent zero passes for the average curve. For a bot to be detected as such, it has to produce 14 non-unique combat sequences within the given time frame. This threshold is admittedly low, but during the evaluation process, it showed that especially players new to the game, use a limited amount of their abilities at the beginning. With these settings, it was still possible to detect all available bot traces reliably. In a PvP environment, it is far less problematic to identify bots. Even with the first introduced detection approach, the intervals the characters usually operated in, were between $[6, 20]$ with an average of over 14, while the bot acted in an interval of $[0, 4]$. PvP gameplay has the property to be highly interactive. The resulting fight sequences have a high variance because human enemies have a broader reaction spectrum than computer-controlled ones, resulting in a heightened demand of proper reactions. The bot, on the other hand, executes its predefined sequence every time, regardless of its surroundings.

6.4 Impact

Game bots are, of course, not the only harassment some individuals have to endure when indulging in massive multiplayer worlds. Almost the first thing that happens to every game upon release is, that the servers and their communication channels are flooded with spam. What starts with ordinary announcements that can be ignored quickly evolves into targeted (directed) messages to sell specific in-game goods like money or equipment. This issue is, however, easier to control for game companies, because a really anonymous post within the game is nigh impossible. There is always account information connected to a character which can be tracked, suspended or treated likewise. Account and identity theft are another major problem, which are not directly connected to the social network-aspect of games but mostly happen through malware infections (e.g. keyloggers and trojans). Game companies try to mitigate this issue by introducing a two-channel authentication (e.g. with a separate authentication token). For the very narrow field of bots in MMO's, a sequence-based approach could be used to protect them from being exploited by bots. In a proof-of-concept implementation for *World of Warcraft*, it was shown that it is possible to reliably categorize over 485.000 action sequences and tell the difference between automated and human players.

The approach is not limited to *World of Warcraft* but is feasible to every queue-based MMOG provided that the necessary domain knowledge is available. While several solutions to toughen a game against bots are thinkable, the effort to actually implement them is rarely appointed. One reason might be insufficient resources. After all, not every game is as successful as *World of Warcraft*. Competing game developers have a hard enough life with keeping their games free of bugs. For the major players, however, the conclusion is a different one. Having the resources to protect games against bots, but not using them, either means the effects are inside their self-set tolerance zone or *wanted* in some respect. In the end, a *bot-ing* player still is a *paying* player. With a detection mechanism for the gamers themselves, they are given the power to detect cheaters and report them to the company. And user complaints are hard to ignore for game companies and social network providers. If they do, they run the risk of losing customers.

7 Human Factors

As already mentioned in the Introduction, a central element all social networks have in common is the human operating it. Those human factors are important elements that nfluence social network development and bring in their own set of security challenges. One of the key problems is related to the notion of *cyber-physical systems*, which encompasses the broader human-machine interaction paradigm, or *social computing*. Furthermore, the newly introduced Web 2.0 and its successor Web 3.0 are, in fact, strongly oriented towards the end-users. This in combination with the boom of on-line social networks is opening a vast field of research for social networks security and especially the involved human factors.

A good starting point for this chapter is presented in 'Social Computing: Study on the Use and Impact of Online Social Networking' [38], a technical report that outlines the emerging online social networks phenomena peculiarities and problems. In this context, virtual socialization blurring, privacy problems, influence of the young generation and cross-cultural problems are noteworthy key elements. Some progress on these problems is already available in the new Progress Report from December, 2011 of the Digital Agenda for Europe 2020 [49] and in the new project for Global Risks 2012 [57] but they concern the policy level, whilst the research is not so fast progressing.

Basically, the current modern research is practically oriented towards human factor characteristics dynamics like: emotions, privacy, attitude and behavior, extracting data from clickstreams, workloads and psychophysiological analysis of social network users including the *media sharing concept*.

Apart of this, the general concept of *social media* and a discussion about how it is related to Web 2.0 and user-generated content is presented in Kaplan and Haenlein [76]. The authors provide a classification of *social media*, which groups applications currently subsumed under a generalized term into more specific categories by characteristics like collaborative projects,

blogs, content communities, social networking sites, virtual game worlds, and virtual social worlds.

The overall trend is clear. While on-line social networks are undoubtedly a technical implementation with a lot of technical aspects, the human and emotional factor is just as important. So far, our main focus in this deliverable was mostly technical. In this last chapter, we want to shed some light into non-technical research related to this topic.

7.1 Clickstreams, mouse movements and workload based studies

A number of studies on analyzing the workloads of social networks and their graph theoretic properties have been recently performed.

A study of the popular Orkut, MySpace, Hi5, and LinkedIn social networks based on clickstream analysis is given in Benevenuto et al [33]. Key features of the social network workloads, such as how frequently people connect to social networks and for how long, as well as the types and sequences of activities users conduct on these networks have been explored. Additionally, modeling the probability of the Orkut network elements usage was studied. These achieved results based on identifying patterns in social network workloads and social interactions allow a better understanding of social networks users' activities and interactions.

Burke et al [35] explored user motivations for contributing to social networking sites, based on server log data from Facebook. They found that newcomers who see their friends contributing typically share more content themselves. Furthermore, those who were initially oriented towards contribution, receiving feedback and having a wide audience, were also inclined to increased sharing. Chapman and Lahav [40] conducted survey interviews and analysis of web browsing patterns for different nationalities to examine ethnographical differences in the social networks usage.

An attempt to analysis the Twitter social network data by leveraging different users' datasets and their posted content, together with a snapshot of their activities is presented in Kivran-Swaine and Naaman [80]. They consider the social networks as 'social awareness streams' trying to understand the relationship between social sharing of emotion and online social network properties excluding gender and cultural differences. A utilization of discrete emotional scale (including: joy, sadness, fear, trust, anger, disgust, anticipation, and surprise) was performed.

A study of social well-being and social network activity using well-being scales and server logs in the Facebook environment was performed by Burke et al [36]. The study is showing that the augmented social network usage is associated with increased social capital and reduced loneliness. The authors interpret these findings in three ways:

1. people who feel more socially connected gravitate toward technical systems that reify those connections,

2. using sites like Facebook allows people to reinforce fledgling and distant relationships, or

3. there is a positive feedback loop.

The self-presentation and social happiness was also investigated in Kim and Lee's work [79] amongst college students who are Facebook users by using recruitment messages, along with a hyperlink to the survey questionnaire. The study suggests that the number of Facebook friends and positive self-presentation may enhance users' subjective happiness, but this portion of happiness may not be grounded in perceived social support. On the other hand, honest self-presentation may enhance happiness rooted in social support provided by Facebook friends. Implications of the findings are discussed in light of affirmation of self-worth, time and effort required for building and maintaining friendships, and the important role played by self-disclosure in signaling one's need for social support.

In the work of Mislove et al [93] by using Orkut, Flickr, LiveJournal and YouTube the power-law, small-world, and scale-free properties of this phenomena are confirmed. A comparison of the explicit friend relationship network with the implicit network created by messages exchanged on Cyworld's guestbook is studied by Ahn et al [30]. They found similarities in both networks: the in-degree and out-degree were close to each other and social interaction through the guestbook was highly reciprocal.

Liben-Nowell et al [88] analyzed the geographical location of LiveJournal users and found a strong correlation between friendship and geographic proximity. Krishnamurthy et al [82] analyzed the social network Twitter by examining the geographical spread usage of Twitter and user behavior in this environment. Huberman et al [70] showed that Twitter users have a small number of friends compared to the number of followers they declare.

Golder et al [62] analyzed temporal access and social patterns in Facebook. They analyzed the message header exchanged by Facebook users, revealing periodic patterns in terms of messages exchanged on that network. Gjoka et al [61] have studied application usage workloads in Facebook and the popularity of applications. Nazir et al [96] similarly analyzed application characteristics in Facebook, by developing and launching their own applications.

A proposal for using interaction graphs to impart meaning to online social links by quantifying user interactions is given in Wilson et al [111]. They analyzed interaction graphs derived from Facebook user traces and showed that they exhibit significantly lower levels of the *small-world* properties shown in their social graph counterparts. Valafar et al [109] conducted a measurement study of the Flickr network and demonstrated that only

a small fraction of users in the main component of the friendship graph is responsible for the vast majority of user interactions.

The problems of 'digital parenting' and hidden threats for the younger generation and 'non-use attitude' to social networks are discussed in the publications of J. A. Rode [104] and Baker, Psych and White [31].

In her work J. A. Rode [104] starts from the idea of more leisure time for the generation under eighteen spent in the social networks and presents three systematic approaches to setup the parent's strategy for protecting children against different threats. The first approach is tentatively called *security czar*, a single adult experienced in security, privacy, ethics and pedagogy. The second approach is called *self-support households*. Each computer owner in the household is responsible for her own safety. The last approach proposed by the author is *outside-support providers* in which the security needs are satisfied by an external source, outside the household. In each case, monitoring the children's activity is performed with and without specialized software tools. The results from practical implementations of these three approaches have shown a necessity for further research of how children use these technologies in order to organize an effective protection strategy for parents and from a technological side. Some reasons for non-use of the rapidly growing social networks like Facebook and MySpace utilizing self-reporting questionnaires used amongst young teenagers was given in Baker, Psych and White [31]. The initial findings are proclaiming: lack of motivation, poor use of time, preference for other forms of communication, preference for engaging in other activities, cybersafety concerns, and a dislike of self-presentation online, including influence from real friends' usage of social networks.

Different profile privacy aspects like: challenge questions, messaging security, social networks profiles similarities, fake profiles, profiles security guidance errors, health care privacy, biometric authentication and data management problems depending on the human factors are presented in the work of Just and Aspinal [74], Mannan and Oorschot [89], Lavesson and Johnson [86], Zhu et al [118], Egelman et al [51], Williams [110], Zheng et al [117], Nolan and Levesque [97].

The work of Just and Aspinal [74] discusses a case with so called 'challenge questions' a form of social authentication. These questions can be used for recovering forgotten passwords, credentials or as an additional step for authentication in the social networks. Typically, the question and corresponding answers are selected by the users. The problem is that most of the users tend to select questions for which it is relatively easy to guess the answers for an attacker. In the opposite case, the user generates questions with a large number of possible answers and then forgets the right one. Both cases are security issues directly depending on human factors. The authors propose a model based on a survey amongst large numbers of students. The

challenge questions collected are used to enumerate and assess the modes of attacks and to evaluate the effort needed to crack specified classes of questions. Despite the research is in an early stage, the authors believe that the result could be used for building reliable secure solutions in this area.

Mannan and Oorschot [89] propose a solution for restricting and managing the amount of personal data which a user decides to publish on the Web. The authors present a working prototype called *IM-based Privacy-Enhanced Content Sharing* directed to instant messaging system as a separate application, outside of the particular social network implementation, which is not dependent on the hosting system.

Lavesson and Johnson [86] discuss important security and privacy issues related to user profiles in a social network. The profiles contain private and public data that users expose to different parties according to their needs and individual preferences. The authors propose a method for comparing user profiles by measuring the distance between the profiles in Euclidean space, and evaluation of the user privacy settings. As a demonstration, this method is applied to Facebook. The proposed framework and methods are suitable to use within Facebook only. As a result, end users and application vendors will receive detailed warnings which application may threaten the user's privacy or integrity.

Zhu et al [118] discuss problems with fake profiles with convincing and comprehensive details that are far more likely to gather personal data from other users when exchange messages and other information with them, using the psychological method *norm of reciprocity*. The practical experiments with mobile PDAs and different scenarios for reciprocity attacks have shown promising results with applicability in social networks.

Egelman et al [51] deal with well-known design limitations for user privacy and access control settings in Facebook and their impact on user behavior. These limitations can bring over-sharing of personal data, despite the user's desire. Two different techniques are developed to help users minimize such kind of human errors on Facebook. In the first case, a specific guidance about access control limitations is given to a group of the participants. In the second case no such guidance is given and the users are significantly less likely to see or prevent errors while configuring their personal data access rules across one or more overlapped circles of network friends. These two techniques are not directly compared in the study.

Zheng et al [117] describes a method for biometric authentication without specialized and expensive hardware, based on the unique nature of user's mouse movements and clicks. The angle metrics between curves drawn by the mouse are used. The authors claim a better reliability than collecting curve shapes only. When recording the mouse activity is finished, a unique mouse signature or mouse fingerprint is produced for the user by using Support Vector Machines. The signature can be continuously checked and rechecked many times during the user's real work. This way, security

issues like stolen password or private keys, computer abandoned by the authorized person after logon etc., could be avoided with quite low error rate, false positive or false negative authentications when enough mouse movements are recorded and analyzed. An additional advantage is that practically no sensitive user information can be collected at server-side in opposite to the analogous keystroke authentication method, which makes the idea quite interesting for social networks users.

Nolan and Levesque [97] discuss the security and privacy aspects from a human and management viewpoint regarding the so called *buried* or *archaeological data* which exists since decades in the network. The reasons for that included forgotten web links after server reinstallation, mistakes in sharing data, expired but uncleaned caches and many other human errors that are made in our busy schedules. Personal information and useful facts for the attackers may be revealed and extracted from that archaeological data scattered over the net. In general, the authors present different techniques, methods and understandable examples for exploiting the human factors. This problem is extremely important for social networks and opens the question of data ownership that is an ongoing discussion for the Digital Agenda for Europe 2020.

Williams [110] studies the existing solutions in using different forms of social networking for health care. These types of social networks are interesting for attackers because of the presence of very detailed personal data not only health-oriented. Actually, the social networks in health care are the most comprehensive source of personal data. Special attention is paid to privacy and security issues inside such kinds of environment. The identified issues include the high sensitivity of personal health records, the ability for primary and secondary exposed patient data and inquiries and informal friendship relations inside the social network which could be taken as signals by third parties. The author presents an abstract model of the social network as a dynamic graph in which the vertices are binary relationships between the user profiles. In addition, other functions are described, which are necessary for a fully operational social network. Future directions for research and improving the security and privacy in this area are noted in the conclusion.

Finally, a comprehensive set of studies and methodologies concerning social media behavioral aspects analysis from a 'media-sharing' viewpoint is given in the book of H. Vicky Zhao et al [116]. An application of the Game theory for modeling user dynamics and human behavior inside the 'media-sharing' concept in social networks is discussed. In general, multimedia is produced and uploaded or forwarded and shared in the network by the millions of end users. The main focus are human factors, which influence all aspects of large scale 'media sharing' including the security, privacy and intellectual property rights. The authors accentuate on the importance of the human factors and demonstrate that different signal processing methods

can be effectively used in modeling user dynamics producing improvement of the system performance and data safety with many real world examples. Two different types of media-sharing social networks are analyzed: multimedia fingerprinting and peer-to-peer live streaming networks.

7.2 Psychophysiological analysis based studies

In the work of Mauri et al [90] an exploration whether the use of social networks elicits a specific psychophysiological pattern is given. Specifically, signal records of skin conductance, blood volume pulse, electroencephalogram, electromyography, respiratory activity, and pupil dilation have been monitored. The study is based on the valence-arousal discrete emotions space model and the theory of flow dynamics for different user states. Some significant findings for stress, relax and Facebook usage differences concerning the dynamic analysis for specific physiological patterns have been observed. The results are seen from the authors as a key that gives a possibility for explain why social networks are spreading out so successfully.

A cognitive model and continuous approach to evaluate a methodological framework for studies of emotions in the context of IT threats identification for Facebook social network users is given in Minchev and Gatev [92]. The psychometric results showed that extroversion and stability of the studied Facebook users are correlated positively with the intensity of social networks usage. The polyphysiographic monitoring study gave a good starting point for an evaluation of the threats foreseen and modeled by experts.

A noteworthy project conducted in Bulgaria is called the 'Study of the Information Threats and Behavior Dynamics of Social Networks Users from the Internet' [55]. Starting at the end of 2011, this project aims to shed some light into human factors in social networks and how they are affected by emotions. The project maintains a close cooperation with the SysSec consortium and is planned to extend on a 24 month period.

7.3 Discussion

Evidently, human factors play an important role in today's social networks. The open tasks offer a multitude of research fields where a deeper investigation is required. People feel compelled to share their emotions and that is changing their behavior which, in turn, opens additional research questions in the psychology and psychophysiology of humans. From the IT aspect of the social networks this includes problems like privacy, user security, social media and successful e-economics that are currently studied on the basis of users monitoring via different informational sources (e.g. workloads, server

logs, specialized browser applications, click streams and mouse movements, psychological questionnaires, bio data analysis of the central and peripheral nervous systems).

What becomes clear with this overview on related work for social network is, that the field is not a purely technical area. Instead, it directly blends from the IT area into social and human science. Even when devising countermeasures for attacks or security implementations, the human factor plays a major role. A new technology is only accepted when its usable and fit to be used by the majority of human operators no matter how well-devised it may be. Therefore, today's researchers must adopt to the situation and take human emotions, preferences or reluctance into account. A task that can proof to be more difficult than creating a technically sound system.

8 Conclusion

This deliverable discussed key topics of social network security and how these aspects influence the network's participants. The main focus was not on those well-known "common" attack scenarios which affect ordinary computer systems. Instead, the presented problems are all quite specific to social networks and the humans that operate them. That does not mean that these attacks (e.g. Spam, spear fishing, social engineering, keyloggers, etc.) are not feasible in such an environment. These topics are, however, well discussed in other research papers and need not be discussed in this context. Instead, the new possibilities for attackers, like the Reverse Social Engineering from Chapter 2, show that a whole new area of prospects needs to be investigated. The presented attack scenario is highly effective and as of today, no effective countermeasure is available to thwart it.

New functionalities like social plugins to third party sites are also a considerable influence on the structure of today's web sites. It severely affects the user's privacy without them knowing it. The most apparent change when introducing such a feature is an immediate gain. Therefore, we count on these plugins to be ubiquitous also in the near future.

From a research aspect, the task of mapping a social network is made easier by employing snapshotting techniques instead of using live data at all times. Compared with state-of-the-art web crawling techniques, such an approach significantly reduces network traffic, is easier to maintain, and has access to additional and hidden information. Extensive evaluation of these techniques have shown that they are practical and effective to collect the complete information of a given social networking account reasonably fast and without detection from social networking providers. We believe that these techniques can be used in cases where no legal cooperation with social networking providers exists.

An interesting aspect of the discussed topics is their convergence towards the research roadmap presented in Deliverable *D4.1: First Report on Threats*

on the Future Internet and Research Roadmap. In this document, the importance of social networks are counted among the top-tier threats as perceived by the research community. In 4.2.2, privacy-related issues are discussed as one of these concerns. The research and discussion presented in this document under 4 directly follows up on these problems, acknowledging the relevance of the roadmap. Section 8.3 in D4.1 covers the problem of detailed user data being available for targeted attacks. One of these attacks is the Reverse Social Engineering Attack in Chapter 2. An obvious reason for the tight correlation between research roadmap and this document is the limited timeframe between these deliverables. There is only so much that can change in six months. Discussing the difference of the forecast and perceived research directions will be more interesting in the upcoming deliverables.

What remains to be said is, that even though there seem to be a lot of shortcomings in today's social networks, the big crash has yet failed to appear. In our opinion the reason for this fact is that a system based on human interaction is highly adaptive and hard to automatically exploit. Furthermore, the same humans put enough pressure on the service provider where technical shortcomings are concerned. As a result, severe problems are fixed in a relatively short time, while the community finds work-arounds for minor or just mildly annoying problems. This very property might also provide the necessary resources to encounter future challenges.

Bibliography

[1] BuiltWith - Widgets Distribution. http://trends.builtwith.com/widgets.

[2] Disconnect. http://disconnect.me/.

[3] Do Not Track - Universal Web Tracking Opt Out. http://donottrack.us/.

[4] Facebook Blocker. http://webgraph.com/resources/facebookblocker/.

[5] Facebook Developers - Permissions. https://developers.facebook.com/docs/reference/api/permissions/.

[6] Facebook for Websites. https://developers.facebook.com/docs/guides/web/.

[7] Facebook Plugins. http://developers.facebook.com/docs/plugins/.

[8] Facebook Security Phishing Attack In The Wild. http://www.securelist.com/en/blog/208193325/Facebook_Security_Phishing_Attack_In_The_Wild.

[9] Facebook Statistics. https://www.facebook.com/press/info.php?statistics.

[10] Facebook Stats. http://www.facebook.com/press/info.php?statistics.

[11] Ghostery. http://www.ghostery.com/.

[12] Google +1 button. http://www.google.com/+1/button/.

[13] HTTP state management. http://www.ietf.org/rfc/rfc2109.txt.

[14] *MMOMimic*. Last accessed: 15. June 2011.

[15] NoScript. http://noscript.net/.

[16] OAuth. http://oauth.net/.

[17] ShareMeNot. http://sharemenot.cs.washington.edu/.

[18] Start 2012 by Taking 2 Minutes to Clean Your Apps Permissions. http://mypermissions.org/.

[19] Symantec Official Blog - Facebook Applications Accidentally Leaking Access to Third Parties. http://www.symantec.com/connect/blogs/facebook-applications-accidentally-leaking-access-third-parties.

[20] WebProNews - Million Sites Have Added Facebook's Social Plugins Since f8. http://www.webpronews.com/2-million-sites-have-added-facebooks-social-plugins-since-f8-2010-09.

[21] *WoW Bot:FairPlay-Bot*. Last accessed: 22. July 2011.

[22] *WoW ZoloFighter Bot, discontinued 15.04.2009*. Last accessed: 15. June 2011.

[23] Sophos Facebook ID Probe. http://www.sophos.com/pressoffice/news/articles/2007/08/facebook.html, 2008.

[24] An Open Letter to Facebook CEO Mark Zuckerberg, June 2010. https://www.eff.org/files/filenode/social_networks/OpenLettertoFacebook.pdf.

[25] Facebook + Media - The Value of a Liker, Sept. 2010. https://www.facebook.com/note.php?note_id=150630338305797.

[26] 5 ways facebooks new features will fuel social shopping, Sept. 2011. http://mashable.com/2011/09/29/facebook-social-shopping/.

[27] Re: Facebook's response to questions from the Data Inspectorate of Norway, Sept. 2011. http://www.datatilsynet.no/upload/Dokumenter/utredningeravDatatilsynet/FromFacebook-Norway-DPA.pdf.

[28] A. Acquisti and R. Gross. Imagined Communities: Awareness, Information Sharing, and Privacy on the Facebook. In G. Danezis and P. Golle, editors, *Privacy Enhancing Technologies*, volume 4258 of *Lecture Notes in Computer Science*, chapter 3, pages 36–58. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2006.

[29] A. Acquisti, R. Gross, and F. Stutzman. Faces of facebook-or, how the largest real id database in the world came to be. Draft available online at http://www.heinz.cmu.edu/~acquisti/face-recognition-study-FAQ/acquisti-faces-BLACKHAT-draft.pdf, Aug. BlackHat USA 2011.

[30] Y.-Y. Ahn, S. Han, H. Kwak, S. Moon, and H. Jeong. Analysis of topological characteristics of huge online social networking services. In *International World Wide Web Conference*, 2007.

[31] R. Baker, M. Psych, and K. White. In Their Own Words: Why Teenagers Don't Use Social Networking Sites. In *Cyberpsychology, Behavior, and Social Networking, Vol. 14, Number 6*, 2011.

[32] M. Balduzzi, C. Platzer, T. Holz, E. Kirda, D. Balzarotti, and C. Kruegel. Abusing Social Networks for Automated User Profiling. In *Recent Advances in Intrusion Detection*, pages 422–441. Springer, 2010.

[33] F. Benevenuto, T. Rodrigues, M. Cha, and V. Almeida. Characterizing User Behavior in Online Social Networks. In *Internet Measurement Conference, Chicago*, 2009.

[34] L. Bilge, T. Strufe, D. Balzarotti, and E. Kirda. All Your Contacts Are Belong to Us: Automated Identity Theft Attacks on Social Networks. In *18th International Conference on World Wide Web (WWW)*, 2009.

[35] M. Burke, C. Marlow, and T. Lento. Feed me: Motivating newcomer contribution in social network sites. In *ACM CHI*, 2009.

[36] M. Burke, C. Marlow, and T. Lento. Social Network Activity and Social Well-Being. In *ACM Conference on Human Factors in Computing Systems, Atlanta, Georgia, USA*, 2010.

[37] E. Butler. Firesheep. Online at http://codebutler.com/firesheep, oct 2010.

[38] R. Cachia. *Social Computing: Study on the Use and Impact of Online Social Networking - Technical Report*. JRC, IPTS, EUR 23565 EN, 2008.

[39] E. Chan, S. Venkataraman, F. David, A. Chaugule, and R. Campbell. Forenscope: A framework for live forensics. In *Proceedings of the 26th Annual Computer Security Applications Conference*, pages 307–316. ACM, 2010.

[40] C. N. Chapman and M. Lahav. International ethnographic observation of social networking sites. In *ACM CHI Extended Abstracts*, 2008.

[41] K.-T. Chen, J.-W. Jiang, P. Huang, H.-H. Chu, C.-L. Lei, and W.-C. Chen. Identifying mmorpg bots: a traffic analysis approach. In *ACE '06: Proceedings of the 2006 ACM SIGCHI international conference on Advances in computer entertainment technology*, page 4, New York, NY, USA, 2006. ACM.

[42] K.-T. Chen, A. Liao, H.-K. K. Pao, and H.-H. Chu. *Game bot detection based on avatar trajectory*. Entertainment Computing-ICEC, 2009.

[43] K.-T. Chen, H.-K. K. Pao, and H. Chang. Game bot identification based on manifold learning. In *Proceedings of the 7th ACM SIGCOMM Workshop on Network and System Support for Games*, pages 21–26. ACM, 2008.

[44] CNN. Facebook status update provides alibi. Online at http://cnn.com/2009/CRIME/11/12/facebook.alibi/index.html, nov 2009.

[45] N. Cubrilovic. Facebook fixes logout issue, explains cookies. http://nikcub.appspot.com/facebook-fixes-logout-issue-explains-cookies.

[46] M. Dantone, L. Bossard, T. Quack, and L. V. Gool. Augmented faces. In *IEEE International Workshop on Mobile Vision (ICCV 2011)*, 2011.

[47] R. Dhamija, J. D. Tygar, and M. Hearst. Why phishing works. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, CHI '06, pages 581–590, New York, NY, USA, 2006. ACM.

[48] J. Dibbell. The life of the chinese gold farmer. *Life*, 2007.

[49] EC. *Digital Agenda for Europe Annual Progress Report 2011*. EC, 2011.

[50] P. Eckersley. How unique is your web browser? In *Proceedings of the Privacy Enhancing Technologies Symposium (PETS)*, 2010.

[51] S. Egelman, A. Oates, and S. Krishnamurthi. Oops, I did it again: mitigating repeated access control errors on facebook . In *CHI '11 Proceedings of the 2011 annual conference on Human factors in computing systems*, 2011.

[52] Facebook. Graph API. Online at https://developers.facebook.com/docs/reference/api/.

[53] Facebook. Statistics of Facebook. Online at http://www.facebook.com/press/info.php?statistics. Accessed April 20th, 2011.

[54] Facebook. The Facebook Blog: Giving You More Control. Online at https://blog.facebook.com/blog.php?post=434691727130, oct 2010.

[55] Facebook Profile of Project DMU03/22. Study of the Information Threats and Behavior Dynamics of Social Networks Users from the Internet, DMU03/22, Bulgarian Young Scientists NSF Grant 2011-2013. Online at http://www.facebook.com/pages/Human-Factor-Analysis-in-Social-Networks/282560151814614, feb 2012.

[56] Forbes. Steam hacked. Online at http://www.forbes.com/sites/danielnyegriffiths/2011/11/10/steam-hacked-newell-watch-your-credit-card/, nov 2011.

[57] W. E. Forum. *Global Risks 2012, Technical Report*. World Economic Forum, 2012.

[58] FSF. Ocrad - The GNU OCR. Online at http://www.gnu.org/software/ocrad/.

[59] H. Gao, J. Hu, C. Wilson, Z. Li, Y. Chen, and B. Zhao. Detecting and characterizing social spam campaigns. In *Proceedings of the 10th annual conference on Internet measurement*, pages 35–47. ACM, 2010.

[60] H. Gao, J. Hu, C. Wilson, Z. Li, Y. Chen, and B. Y. Zhao. Detecting and characterizing social spam campaigns. In *Proceedings of the 10th annual conference on Internet measurement*, IMC '10. ACM, 2010.

[61] M. Gjoka, M. Sirivianos, A. Markopoulou, and X. Yang. Poking Facebook: characterization of OSN applications. In *ACM SIGCOMM WOSN*, 2008.

[62] S. Golder, D. Wilkinson, and B. Huberman. Rhythms of social interaction: messaging within a massive online network. In *ICCT*, 2007.

[63] P. Golle and N. Ducheneaut. Preventing bots from playing online games. *Comput. Entertain.*, 3(3):3–3, 2005.

[64] G. Hoglund and G. McGraw. *Exploiting Online Games: Cheating Massively Distributed Systems (Addison-Wesley Software Security Series)*. Addison-Wesley Professional, 2007.

[65] http://eu.battle.net/. Battle.net. Last accessed: 10.02.2012, 2 2012.

[66] https://steamcommunity.com/. The steam gaming community. Last accessed: 10.02.2012, 2 2012.

[67] http://www.mmodata.net/. Mmodata charts version 3.3. Last accessed: 14.06.2011, 6 2011.

[68] http://www.origin.com/. Origin. Last accessed: 10.02.2012, 2 2012.

[69] M. Huber, M. Mulazzani, E. Weippl, G. Kitzler, and S. Goluch. Friend-in-the-middle attacks: Exploiting social networking sites for spam. *Internet Computing*, 2011.

[70] B. Huberman, D. Romero, and F. Wu. Social networks that matter: Twitter under the microscope. In *First Monday*, 2009.

[71] C. Jackson, A. Bortz, D. Boneh, and J. C. Mitchell. Protecting browser state from web privacy attacks. In *Proceedings of the 15th International World Wide Web Conference (WWW)*, 2006.

[72] T. Jagatic, N. Johnson, M. Jakobsson, and F. Menczer. Social phishing. *Communications of the ACM*, 50(10):94–100, 2007.

[73] T. N. Jagatic, N. A. Johnson, M. Jakobsson, and F. Menczer. Social phishing. *Commun. ACM*, 50(10):94–100, 2007.

[74] M. Just and D. Aspinall. Personal choice and challenge questions: a security and usability assessment. In *SOUPS '09 Proceedings of the 5th Symposium on Usable Privacy and Security*, 2009.

[75] S. Kamkar. Evercookie. http://samy.pl/evercookie/.

[76] A. Kaplan and M. Haenlein. *Users of the world, unite! The challenges and opportunities of Social Media*. Business Horizons, 53, 59-68, 2010.

[77] H. Kim, S. Hong, and J. Kim. *Detection of Auto Programs for MMORPGs*, volume 3809. Springer Berlin, 2005.

[78] H. Kim, J. Tang, and R. Anderson. Social Authentication: Harder than it Looks. In *Proceedings of the 2012 Cryptography and Data Security conference*.

[79] J. Kim and J.-E. R. Lee. The Facebook Paths to Happiness: Effects of the Number of Facebook Friends and Self-Presentation on Subjective Well-Being. In *Cyberpsychology, Behavior, and Social Networking, Vol. 14, Number 6*, 2011.

[80] F. Kivran-Swaine and M. Naaman. Network Properties and Social Sharing of Emotions in Social Awareness Streams. In *ACM Conference on Computer Supported Cooperative Work, China*, 2011.

[81] G. Kontaxis, M. Polychronakis, and E. P. Markatos. SudoWeb: Minimizing information disclosure to third parties in single sign-on platforms. In *Proceedings of the 14th Information Security Conference*.

[82] B. Krishnamurthy, P. Gill, and M. Arlitt. A few chirps about Twitter. In *ACM SIGCOMM WOSN*, 2008.

[83] B. Krishnamurthy and C. E. Wills. Characterizing privacy in online social networks. In *Proceedings of the first workshop on Online social networks*, WOSN '08, pages 37–42, New York, NY, USA, 2008. ACM.

[84] B. Krishnamurthy and C. E. Wills. On the leakage of personally identifiable information via online social networks. *SIGCOMM Comput. Commun. Rev.*, 40, 2010.

[85] T. Lauinger, V. Pankakoski, D. Balzarotti, and E. Kirda. Honeybot, your man in the middle for automated social engineering. In *LEET'10, 3rd USENIX Workshop on Large-Scale Exploits and Emergent Threats, San Jose*, 2010.

[86] N. Lavesson and H. Johnson. Measuring profile distance in online social networks . In *WIMS '11 Proceedings of the International Conference on Web Intelligence, Mining and Semantics*, 2011.

[87] W. I. Levenshtein. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10:707–+, Feb. 1966.

[88] D. Liben-Nowell, J. Novak, R. Kumar, P. Raghavan, and A. Tomkins. Geographic routing in social network. In *PNAS*, 2005.

[89] M. Mannan and P. C. van Oorschot. Privacy-enhanced sharing of personal content on the web. In *WWW '08 Proceedings of the 17th international conference on World Wide Web*, 2008.

[90] M. Mauri, P. Cipresso, A. Balgera, M. Villamira, and G. Riva. Why Is Facebook So Successful? Psychophysiological Measures Describe a Core Flow State While Using Facebook. In *Cyberpsychology, Behavior, and Social Networking, Vol. 14, Number 12*, 2011.

[91] L. I. Millett, B. Friedman, and E. Felten. Cookies and web browser design: toward realizing informed consent online. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, 2001.

[92] Z. Minchev and P. Gatev. Psychophysiological Evaluation of Emotions due to the Communication in Social Networks. In *Scripta Scientifica Medica, Vol 43, No. 3*, 2011.

[93] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee. Measurement and analysis of online social networks. In *ACM IMC*, 2007.

[94] K. Mitnick, W. L. Simon, and S. Wozniak. *The Art of Deception: Controlling the Human Element of Security*. Wiley, 2002.

[95] S. Mitterhofer, C. Platzer, C. Kruegel, and E. Kirda. Server-side bot detection in massively multiplayer online games. *IEEE Security & Privacy*, pages 29–36, 2009.

[96] A. Nazir, S. Raza, and C.-N. Chuah. Unveiling Facebook: a measurement study of social network based applications. In *ACM IMC*, 2008.

[97] J. Nolan and M. Levesque. *Hacking human: data-archaeology and surveillance in social networks*. ACM SIGGROUP Bulletin - Special issue on virtual communities Homepage archive, Volume 25, Issue 2, 2011.

[98] OpenQA. Selenium wep application testing system. Online at http://seleniumhq.org/.

[99] M. Perry. CookieMonster: Cookie Hijacking. Online at http://fscked.org/projects/cookiemonster, aug 2008.

[100] C. Platzer. Sequence-based bot detection in massive multiplayer onlien games. In *Proceedings of the 8th International Conference on Information, Communications and Signal Processing (ICICS 2011)*. IEEE, 2011.

[101] B. Ponurkiewicz. Faceniff. Online at http://faceniff.ponury.net/, jun 2011.

[102] M. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[103] N. A. Rahman. Scraping facebook email addresses. Online at http://www.kudanai.com/2008/10/scraping-facebook-email-addresses.html, aug 2008.

[104] J. A. Rode. Digital parenting: designing children's safety. In *BCS-HCI 09 Proceedings of the 23rd British HCI Group Annual Conference on People and Computers: Celebrating People and Technology*, 2009.

[105] A. Roosendaal. Facebook tracks and traces everyone: Like this! http://ssrn.com/abstract=1717563.

[106] A. Shulman. The underground credentials market. *Computer Fraud & Security*, (3):5–8, March 2010.

[107] G. Stringhini, C. Kruegel, and G. Vigna. Detecting spammers on social networks. In *ACSAC*, 2010.

[108] The New York Criminal Law Blog. Criminal found via Facebook. Online at http://newyorkcriminallawyersblog.com/2010/03/assault-criminal-who-was-found-via-facebook-is-back-in-ny.html, mar 2009.

[109] M. Valafar, R. Rejaie, and W. Willinger. Beyond friendship graphs: a study of user interactions in Flickr. In *ACM SIGCOMM WOSN*, 2009.

[110] J. Williams. Social networking applications in health care: threats to the privacy and security of health information . In *SEHC '10 Proceedings of the 2010 ICSE Workshop on Software Engineering in Health Care*, 2010.

[111] C. Wilson, B. Boe, A. Sala, K. P. N. Puttaswamy, and B. Y. Zhao. User interactions in social networks and their implications. In *ACM EuroSys*, 2009.

[112] G. Wondracek, T. Holz, E. Kirda, and C. Kruegel. A Practical Attack to De-Anonymize Social Network Users. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2010.

[113] R. V. Yampolskiy and V. Govindaraju. Embedded noninteractive continuous bot detection. *Comput. Entertain.*, 5(4):1–11, 2007.

[114] J. Yan and B. Randell. A systematic classification of cheating in online games. In *NetGames '05: Proceedings of 4th ACM SIGCOMM workshop on Network and system support for games*, pages 1–9, New York, NY, USA, 2005. ACM.

[115] S. Yeung, J. C. Lui, J. Liu, and J. Yan. Detecting cheaters for multiplayer games: theory, design and implementation[1]. *Consumer Communications and Networking Conference, 2006. CCNC 2006. 3rd IEEE*, 2:1178–1182, Jan. 2006.

[116] H. V. Zhao, W. S. Lin, and K. J. R. Liu. *Behavior Dynamics in Media-Sharing Social Networks*. Cambridge University Press New York, 2011.

[117] N. Zheng, A. Paloski, and H. Wang. An efficient user verification system via mouse movements . In *CCS '11 Proceedings of the 18th ACM conference on Computer and communications security*, 2011.

[118] F. Zhu, S. Carpenter, A. Kulkarni, and S. Kolimi. Reciprocity attacks . In *SOUPS '11: Proceedings of the Seventh Symposium on Usable Privacy and Security*, 2011.