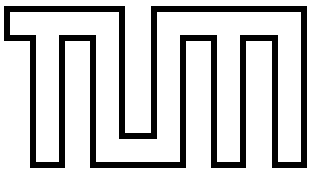
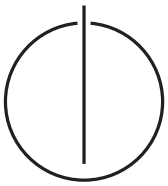


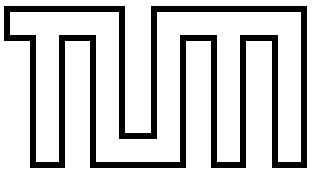
FAKULTÄT FÜR INFORMATIK
TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelorarbeit in Informatik

# On the New Threats of Social Engineering Exploiting Social Networks

Daniel Siegel





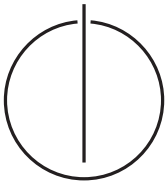
FAKULTÄT FÜR INFORMATIK
TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelorarbeit in Informatik

# On the New Threats of Social Engineering Exploiting Social Networks

# Neue Bedrohungen bei Sozialen Netzwerken im Bezug auf Social Engineering

Author: Daniel Siegel
Supervisor: Univ.-Prof. Dr. Claudia Eckert
Advisor: Dr. Werner Streitberger
Date: August 14th, 2009



I assure the single handed composition of this bachelor thesis only supported by declared resources.

Ich versichere, dass ich diese Bachelorarbeit selbstständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

Munich, August 14$^{th}$, 2009 Daniel Siegel

# Zusammenfassung

Diese Arbeit behandelt eine Studie zu neuen Sicherheitslücken in Sozialen Netzwerken, die durch Social Engineering ausgenutzt werden können. Soziale Netzwerke sind weit verbreitet und werden oft für den privaten aber auch den geschäftlichen Bereich benutzt. Da die Natur eines solchen Sozialen Netzwerkes das Verteilen von Daten ist, stellt sich unweigerlich die Frage, ob diese missbraucht werden können.

Social Engineering ist seit je her eine Bedrohung, die sowohl für Unternehmen aber auch für Privatpersonen eine große Gefahr darstellt und nur sehr schwer zu bekämpfen ist. Bislang war die Informationssuche, auf die ein Social Engineering Angriff aufbaut aber nur mühsam zu bewerkstelligen und barg immer das Risiko für den Angreifer, entdeckt zu werden.

Soziale Netzwerke werden nun in dieser Arbeit analysiert, inwiefern man automatisch an Daten bestimmter Benutzer oder Unternehmen kommen kann um diese für Social Engineering Angriffe zu benutzen. Dafür werden Social Engineering Angriffe sowie Soziale Netzwerke analysiert und drei Beispielangriffe erstellt. Diese werden dann von einem Prototypen implementiert und auf dem Sozialen Netzwerk *Twitter* durchgeführt. Anschließend werden diese Attacken evaluiert und auf ihre Gefährlichkeit hin untersucht. Dabei stellt sich heraus, dass die meisten Informationen, die ein Social Engineering Angriff benötigt, sehr einfach und ohne Mitwissen des Opfers aus einem Sozialen Netzwerk extrahiert werden kann. Neben einem Ausblick auf weitere mögliche Angriffe, werden auch Schutzmöglichkeiten diskutiert.

# CONTENTS

CHAPTER 1

# INTRODUCTION

## 1.1 Problem and Motivation

Social Networks such as Facebook[1], MySpace[2] or Twitter[3] gained millions of members on their platforms over the last years. They are widely used in both private and business networking. Also, they allow individuals to present their own user profile and to share or maintain relationships with other members in the network. There is no doubt that social networks are a useful instrument and pose a communication platform which many people are currently using and even more will be using in the near future [Whi09].

The nature of social networks is about sharing data but frequently users are as well presenting their information to an unknown number of strangers. This does not only raise the question of how useful a social network is for a single user, but also if there

[1] http://www.facebook.com
[2] http://www.myspace.com
[3] http://www.twitter.com

are any drawbacks or risks for an individual or a company. Users often put a lot of sensitive data into their profiles, as for example Brown et al. show [BHI+08]. This data however is stored centrally at the company which provides the network and so the user looses the control over his or her data [Fra08].

Of course there is a relation between the data which the user entered and the actual picture of the user. Following that thought, another person is not only able to extract the data but can obtain new information or interpretations which were not entered by the actual user.

Many studies such as [Fra08, GAHI05] show that this can be a massive intervention into one's privacy. This work however wants to examine whether such information is a security risk and can be used for attacks against a company or an individual. More precisely, it wants to determine how social engineering attacks can be driven against users of social networks, based on data, which is legally accessible on social networks.

Of course, a distinction between employees of a certain company and private users has to be made, as certain information can be harmless to an individual, but a danger to a company [MS03]. For example a phone number of a private person is probably harmless, whilst the phone number of a certain office inside a company could be used for a social engineering attack. In both cases, however, a social engineer saves additional time and effort to get specific information about his victim, if he can find the information needed on a social network. Having to collect the information by hand does not only require extra work, but it also includes the danger of being exposed. A social engineer has to remain undiscovered during the attack, otherwise he would not be able to carry out an attack [MS03].

Especially big corporations spend hundreds of thousands of dollars for the security of their IT infrastructure. An attack on the IT layer would therefore be often very laborious. Hence it is much simpler to bypass the security mechanisms through social engineering, as it is very cheap and does not require any superior technology [WD95]. Commonly, a skilled social engineer does not require anything more than a telephone for an attack of such nature [MS03].

Concretely, this study wants to answer the following research question: How can data of individuals or companies automatically be extracted from social networks and presented in a way that can be used for a social engineering attack. In addition, countermeasures against automatic extraction and social engineering attacks are going to be developed.

## 1.2 Methodology

In this work, a five-step approach was used to characterize useful information, extraction, attacks and countermeasures.

1. Data was obtained from the chosen social network of several users. The data was taken from well known people on the network, as well as less famous. Furthermore, users who were very active such as less active users were observed.
2. The data gathered in the first step was studied and it was determined which kind of attacks could be realistic.
3. Three sample attacks were chosen, which took place in reality in the last years. All of these attacks are described by Mitnick and Simon [MS03]. The chosen attacks were then transferred to the prototype and the chosen social network.
4. The attacks were tested on sample profiles for their feasibility and efficiency.
5. Countermeasures were elaborated to mitigate the risk of such attacks.

## 1.3 Outline of the Thesis

**Chapter 1**: Introduction

This chapter presents an overview of the thesis and introduces the reader to the topic of new threats of social engineering by exploiting social networks.

**Chapter 2**: Related Work

An outline of related researches in the field of social engineering, the attacks and countermeasures will be presented. Common attacks and how companies and individuals can protect themselves against such attacks will also be showcased.

**Chapter 3**: Selection of Social Engineering Attacks

Three social engineering attacks, which happened in reality, were chosen and are presented to the reader. They are analyzed in order to let the prototype repeat the attacks.

**Chapter 4**: Analysis of the Social Network *Twitter*

The study takes a deeper look at the social network *Twitter*. It will show the drawbacks and risks of such social networks, the data which can be extracted automatically and also, how sample attacks could look like. Finally the countermeasures against such attacks are discussed.

**Chapter 5**: Design, Analysis and Implementation of a Prototype

In this chapter, a prototype is developed, which can automatically harvest data and present it in a way that can be used for social engineering attacks.

**Chapter 6**: Evaluation

This chapter examines whether the previously described attacks can be achieved by using the prototype and therefore shows sample attacks. It will analyze the attack and evaluate the risk against a company or an individual.

**Chapter 7**: Conclusion

The thesis finally concludes and draws together the main findings of the study.

## 1.4 Glossary

**API** An application programming interface (API) is an interface in computer science that defines ways to interact with services from libraries, the operating system or the web.

**Atom** The Atom Syndication Format is a format used for web feeds based on the XML Language.

**GET** A method for requesting a specified resource using the HTTP protocol.

**JSON** JavaScript Object Notation (JSON) is a computer data interchange format which uses the JavaScript syntax.

**POST** A method for transferring data to an identified resource using the HTTP protocol.

**REST** Representational State Transfer (REST) is a style for software architecture for distributed systems. The web is built on that style, for example.

**RSS** Really Simple Syndication (RSS) is a feed format to publish web feeds, like blogs, news, audio or video

**URL** A Uniform Resource Locator (URL) specifies how and where an identified resource is available and the methods retrieving it.

**WHOIS** A protocol usable for determining the registrant or owner of a domain name, IP adress or other Internet resources.

**XML** Extensible Markup Language (XML) is a general-purpose specification for creating custom markup languages.

# CHAPTER 2

# RELATED WORK

## 2.1 Social Engineering

In most companies and private networks, the main security focus aims towards security technology, such as firewalls or anti-virus programs and other defense strategies [WD95]. However, companies and private people are often unaware of social engineering which sometimes is a lot more dangerous than the IT-security suggests [Jon04]. Kevin Mitnick, a hacker and one of the most famous social engineers, states that it is often easier to use social engineering to get access to a system than searching for security holes [MS03].

The term social engineering is not easy to describe and there are many definitions available. The new hacker's dictionary [Ray96] defines the term social engineering as follows:

> *Term used among crackers and samurai for cracking techniques that rely on weaknesses in wetware rather than software; the aim is to trick people into*

> *revealing passwords or other information that compromises a target system's security. [. . . ]*

Another way to define the term is that of Microsoft [Mic09]:

> *Social engineering is a way for criminals to gain access to your computer. The purpose of social engineering is usually to secretly install spyware or other malicious software or to trick you into handing over your passwords or other sensitive financial or personal information.*

A more general approach has been undertaken by Mitnick and Simon [MS03], which is one of the most appropriate definitions and will be used in this work:

> *Social Engineering uses influence and persuasion to deceive people by convincing them that the social engineer is someone he is not, or by manipulation. As a result, the social engineer is able to take advantage of people to obtain information with or without the use of technology.*

Besides the various definitions it is also important to understand how a social engineer relates to a hacker in general. Thornburgh proposes [Tho04] that social engineers are not hackers themselves, but are *hacker-enablers*. Following this line of thought, the goal of the social engineer is to gain either physical or digital direct access to information of the target or an information system. Afterwards the social engineer can enable a hacker to access and penetrate the system, delete, alter or steal information and disrupt services [Tho04]. Of course, the social engineer can also be the hacker. Hacking characteristically involves access to systems through technical means, while the social engineer manipulates people to give him access to information that normally would not be available [Jon04]. Furthermore, social engineering uses psychology and theories about the human mindset, for instance what people expect from each other and their natural tendency to be helpful [Jon04].

### 2.1.1 The Social Engineering Life Cycle

Defining the profile of a social engineering attack is quite difficult, as every attack includes people with their behavioural changes over time, just like their mood and

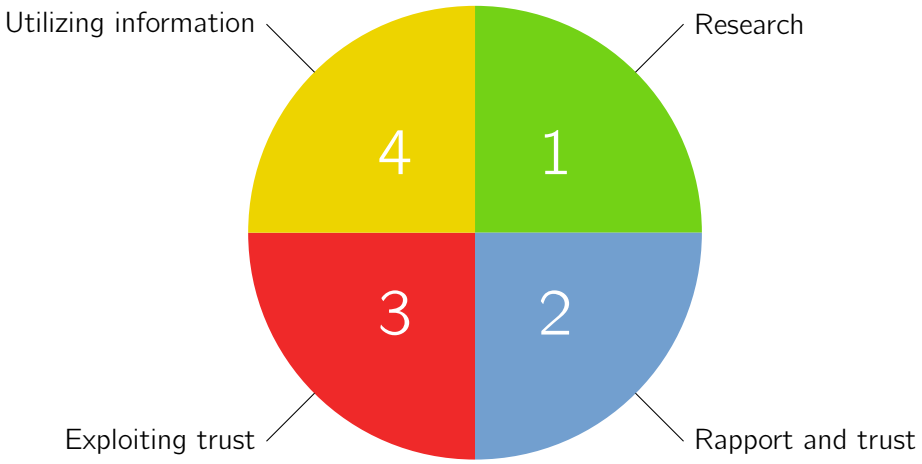



Figure 2.1: A tipical social engineering attack cycle.

other personal or emotional characteristics. However, Mitnick and Simon identified four stages of a social engineering attack [MS03]: research, developing rapport and trust, exploiting trust and utilizing information. A single cycle does not have to be limited to a singular cycle, but can contain several other cycles until the objective is reached [Tho04]. The process itself can therefore be recursive and iterative, depending on the attack and methods used.

#### 2.1.1.1 Research

This phase consists of several methodologies to gain an extremely high amount of information about the target. Different methods might be used, depending on whether the target is a company or an individual. The ultimate goal of this phase is to gain as much information as possible about the target to develop a relationship, rapport and trust in the next phase. Of course, there are enormous possibilities to get information on companies and individuals, the most known are described by [Jon04, MS03, Tho04], but of course they are not limited to those.

**Corporate Website** The Website of a company might be the first place to start looking out for such information. Lists of employees, company directors, or company brochures is often publicly available.

**Personal Homepage** If the target is an individual, the personal homepage of the target might offer more information than the company website. If the target runs a blog, information about the workplace can be accessed also quite often. This information can then be used to attack the employer.

**Dumpster Diving** This is a technique was and is also used by several intelligence agencies [Liv03]. It includes the search for useful information in the rubbish container of companies and individuals. While this seems a strange technique, a social engineer still can gain much information out of this.

**Phishing** Fake e-mails or websites are set up and sent/accessed by individuals. They both have the goal to make the target believe that they are watching a regular e-mail or website and entering sensitive information. Phishing's description can range from a complete social engineering attack to a simple information gathering.

**Trojan Horses and Other Malware** These tools have the same goal as phishing, however they work automatically and often do not require any user interaction, except for example installing the software. They can be used for other objectives as well, for example a keylogger or a tool which logs the website the target is accessing or e-mails.

**Newsgroups and Mailing Lists** Several employees or individuals are subscribed to public accessible newsgroups and mailing lists. These are often technical groups and lists and therefore expose information, like which system is in use and how it is configured.

**Job Sites** Companies often look to hire new people and many times they have to include sensitive information, if they look for people who have gained experience on that field. Like above, information about the computer systems can also be obtained here.

As social networks are quite new, the existing literature does not count them yet as a typical research possibility for social engineers. Social networks however offer a wide range of the above mentioned possibilities. For example, the corporate website could be replaced if a large amount of the employees use a specific social network. The personal homepage might be replaced by a social network, if enough private data about a person is available on the social network. Social networks also often include newsgroups, job sites and other information on their platform, making it even easier to access a large amount of data about a single person.

#### 2.1.1.2 Rapport and Trust

In this second phase, the social engineer makes contact via several ways of communication. The attacker uses techniques like name-dropping, where he *drops* several names of verifiable employees or individuals to make the target believe the attacker is who he pretends to be. Once the attacker has established himself as authentic, he goes onto the next phase. Jones shows the following physical settings, in which an attack can happen [Jon04]:

**Telephone** The most common type of social engineering attacks are those made by telephone. Almost every person is vulnerable to this type, especially call desks, but also individuals and employees.

**Workplace** This is a quite dangerous type, as an attacker has to gain physical access to the targets workplace. However, there are many possibilities of how a talented attacker can gain legitimate access to a building. Once inside, the attacker can look out for passwords, sensitive documents, do shoulder-surfing or access computers and network, which are left unguarded in the accessed area.

**E-mail** The social engineer can compose an original looking document, either from the company the target works for, another company or a service, the target uses, like a social network.

**Chat** Similar as described above, the attacker connects to the victim directly, forcing the target to install malicious software or exposing sensitive information.

Social networks allow an additional way to create a rapport with the victim, for example the often integrated chat, forum and messaging functions. The rapport could be established on a thinner basis, depending on the method used. However, this could also be a desired effect by the social engineer.

#### 2.1.1.3 Exploiting Trust

Once the connection has been established, the attacker can exploit the trust or better the relationship of reliance by *„weaving a story that plays upon the emotions of the victim“* [Tho04]. Social engineers heavily rely on psychology to exploit their trust. Jones lists the following [Jon04]:

**Authority** This can be a highly effective psychological weapon, for example when impersonating the boss of the company. Orgill et al. have proved by means of a real case study that social engineering attacks are even more effective when the supervisor supports the attack or an attacker impersonates a supervisor [ORBO04].

**Natural Tendency to be Helpful** Psychology shows that it is a natural tendency to be helpful. A social engineer can use this to his advantage. Several examples are shown by Mitnick and Simon [MS03], for example desperate calls in the night to the help desk, or a person trying to get access to the building with many boxes in his hand.

**Liking and Similarity** A personal connection might be helpful for the attacker. Psychology shows that it is natural to like people who are like themselves. If information about hobbies or interests are available, a social engineer might use that information to establish a relationship based on it. Once this connection has been made the victim may feel more responsive to the attacker and therefore might offer more sensitive information.

**Reciprocation** Social interaction is often based on the fact that if someone gives something, one might give him something in return. In social engineering, this

is called *reverse social engineering*. In this case, a social engineer tries to help the victim and in return requests a favour from that person.

**Low Involvement** If a person has low to no involvement in the company or towards a victim, it might have little interest in what the social engineer is asking. Commonly, receptionists, the cleaning crew and similar people are favourable victims.

These constructs can also be exploited by using the information available on social networks. For example, the authority might not exist on a social network, however if the attacker is able to create a large social group around, he surely has a certain authority he can use against the victim. The gained information can even help to exploit the trust, for example if an attacker already knows how helpful the victim is, what the victim likes or dislikes or if he shows low involvement.

#### 2.1.1.4 Utilizing Information

This phase finally utilizes the gained information to either implement a technical attack, a non-technical attack or to use the gained information for a new social engineering attack [Tho04]. The actual attack and the threat it exposes is created in this phase. It is interesting that the attack created here does not limit itself to technical or non-technical attacks, but can also be used to publish sensitive information or other.

### 2.1.2 Threats and Risks

Thornburgh [Tho04] defines a social engineering attack as successful if the attacker achieves his set objective, even if the information itself is either not enough to perform penetration or if the information is considered of not great value by the target. Moreover, every small piece of information can help the attack to become successful and increases the possibility of it's success.

Orgill et al. [ORBO04] suggests that social engineering is an ever-present threat to the security of computer systems, because it does not attack the computer itself, but the human being using the computer. It exploits the natural tendency of humans to trust others, helping them and gain favour.

In an introduction to social engineering, Manske [Man00] offers the following explanation:

> *Successful social engineering attacks give the attacker the means to bypass millions of dollars invested in technical and non-technical protection mechanisms and consulting, completely nullifying security investment, Firewalls, secure routers, PKI, e-mail. . . and security guards are all down the drain.*

The risk of a social engineering attack depends mostly on the value of the gained information [Tho04]. If the gained information cannot help to develop another attack, the impact of this special attack is minimal to non-existent. On the other side, valuable information can lead to a very dangerous attack, especially if the attacker remains hidden and the attack has not been detected throughout his attack. If the attacker is not able to remain hidden, a system penetration is still possible, however subsequent attacks can eventually be prevented. Overall, a social engineering attack can be as dangerous as the information gained. However, non-valuable information can still help the social engineer to acquire more data.

It was shown that a typical social engineering attack is cyclical and therefore can be repeated inside a company or by using several individuals. The target can be penetrated over and over again, until the ultimate goal is reached. Not to mention the different type of goals that can be reached. Thornburgh [Tho04] and Mitnick and Simon [MS03] mention that social engineers take care to keep the target for further attacks and not to *burn* it [Tho04]. They try to remain anonymous and do everything to prevent their detection.

Amongst several others, Mitnick and Simon [MS03], Winkler and Dealy [WD95] and Orgill et al. [ORBO04] show how easy it is to get access to a company. They use several methods, like telephone, surveys and other methods. All were extremely successful and while not being expensive, all of the case studies managed to get

sensitive information in a very short amount of time, despite strong security measures. Additionally all of these did not use very company specific data and the case studies can easily be extrapolated to other attacks at other companies.

### 2.1.3 Countermeasures

Winkler and Dealy [WD95] prove that many of the weaknesses exploited by the attackers were common in most companies. Although they mention that even the best technical mechanisms had not prevented the attacks. The attacker exploited poor security awareness and even if the attackers had not been able to get passwords, they still would have been able to acquire sensitive personal and company information.

Moreover, security operators have to consider the non-technical side of computer security and not assume that for example cryptography is enough to prevent such attacks. In addition, computer professionals believe all too often that computer security fundamentals are known to everyone. Lively [Liv03] hits the point with following: *„Users cannot defend against what they do not know“*. Most of the related work see user awareness and user rewards as an essential part against social engineering, however they agree that there is no way to protect themselves against social engineering attacks.

Mitnick and Simon [MS03] give some indication of how a social engineering attack can be recognized. However, all of these points still can happen in a totally ordinary phone call or dialogue. Even so, they give a good base to recognize attacks.

- Refusal to give a callback number
- Out of ordinary request
- Claim of authority
- Stresses urgency
- Threatens negative consequences
- Shows discomfort when questioned
- Name dropping

Winkler and Dealy [WD95] try to create a more concrete policy with the following points:

**Do not Rely upon Common Internal Identifiers** Some companies use basic authentication methods to authenticate as real employees. Unfortunately the employee number or similar identifiers are commonly used by companies. Therefore it is quite easy to get a list or just some single identifiers and use them for authentication. The authors suggest having a different identifier for computer related activities and not to rely upon such simple identifiers.

**Implement a Call Back Procedure** Often, social engineering attacks can be prevented if employees would have verified the caller identity by calling them back at their proper number, as listed in the company directory. This of course also includes e-mail or instant messaging. Although it creates an inconvenience, but when compared to the potential losses, this is of course justified, as the team shows. Caller ID services or reverse DNS services can be helpful too.

**Implement a Security Awareness Program** An enormous amount of money is often spend by companies on hard- and software security devices. Still, security awareness programs are barely implemented. As stated before, computer professionals cannot assume that security practises, even the simplest, are known to every user. The paper also mentions that a good security awareness program can be implemented for minimal cost and can save a lot of money compared to the losses.

**Identify Direct Computer Support Analysts** In a company, every employee should be familiar with a single well known computer analyst. The analyst on the other side should not have more than 60 users. He ought to be the only person who can contact and should be contacted by the users. The users shall also contact their computer analyst immediately if another person from the computer support contacts them.

**Create a Security Alert System** During the attacks the authors recognized that there was no way to warn other employees if a single employee of the company would have detected them. So, the attack can continue until the attacker's goal

is reached, even if they were detected by some employees. They also mention that a detection would only have improved the attack, as the attackers would have learned what works and what does not.

**Test Security Policies** Many people already test their physical and electronical security devices, however, the human vulnerabilities is often not dealt with. To enable the above mentioned policies, they have to be tested by social engineering attacks for their effectiveness. And still, those policies cannot assure 100% protection against social engineering attacks.

## 2.2 Social Networks

Social networks gained millions of members on their platforms in the last years. They are widely used in both private and business networking. Social networks allow individuals to present their own identity and to share and keep relationships with other members of those services.

As these platforms became so popular in the last years and store an enormous amount of data about each user, the question must be raised what the relationship between the classic social engineering and the new technology of social networks. It also raises the question about new threats and attacks which are possible by either just using social networks as an attack platform or by using social networks as an information pool. However, both together seem to be possible too.

Many studies, as for example [Fra08, GAHI05] already show that privacy and information revelation in social networks is a massive problem. The following sections want to take a deeper approach and characterize what actual security issues social networks disclose. These attacks can be characterized as new threats, which did not exist before.

The related work can be divided in two parts: *active* attacks and *passive* attacks, whereas *active* attacks contain attacks which exploit services and users of social networks, like messaging services, birthday invitations and other. *Passive* attacks mean that the attack is driven passively by harvesting data and eavesdropping, commonly

without having access to the network. It is important to note that the active attacks are rather real attacks, whilst the passive attacks are more of an information gathering, on which an attack might follow.

### 2.2.1 Active Attacks

Jagatic et al. [JJJM07] ran a phishing attack against a number of students of the Indiana University, aged between 18 and 24 years. They harvested freely available information on social networks and built a dataset. They then spoofed an e-mail message between two friends on a social network using the first person as a sender and the second as the recipient. The message contained a phishing site clearly marked as a phishing URL, which then asked the recipient to enter his university credentials. By means of a control group, they then sent the same message with an unknown sender. The result showed that about 72% of all students did enter their university credentials on the insecure site, while only 16% of the control group did the same. The authors mentioned that the result was much higher than anticipated, as there were many ways to detect the phishing attack, as for example the non-university URL of the site, a bogus authentication message, the *WHOIS* entry and others. However many students did change the password afterwards and installed anti virus software, as they assumed malware on their computers. This is of course useless, as the attackers have already obtained access.

A more concrete attack was carried out by Brown et al. [BHI+08]. The researchers did get sample data out of a popular group on the *Facebook* social network. They then analyzed the publicly available attributes of each group member, trying to determine how they could be attacked. They described three types of attacks, which they actually carried out on *Facebook*, but are transferable to other social networks as well.

**Relationship-based Attacks** Attacks in this category do only use the relationship information of several users for the attack. No attributes other than the relationship are used. The attack described by Jagatic et al. [JJJM07] falls into this category. As an additional example attack, they spoofed a message notification

that looked like an official one that was sent by an actual friend of the recipient and contained a phishing URL.

**Unshared-attribute Attacks** These are attacks which use attributes of only one user together with the relationship information to carry out the attack. The authors created a sample attack, in which they either sent a birthday greeting card or a birthday invitation on the actual birthday date of a user to him or to his friends, similarly containing a phishing URL.

**Shared-attribute Attacks** As mentioned previously, these attacks use the relationship information, together with attributes, which are shared by both users. As a sample attack they created a message again, containing a link to a photo site where photos of a common event are located. Again, the URL in the message pointed to a phishing site.

The results of the attacks mentioned above are listed in table 2.1, containing the three types of attacks based on profile openness. Furthermore, they estimated that about 85% of the users can be accurately targeted by such attacks and even users with closed profiles or strict privacy settings are almost equally vulnerable.

| | Open profiles | Close profiles | All |
|---|---|---|---|
| Relationship-only attacks | 85% | 84% | 85% |
| Birthday greeting | 74% | 0% | 50% |
| Birthday invitation | 84% | 84% | 84% |

Table 2.1: Results of the case study of Brown et al. [BHI+08]

### 2.2.2 Passive Attacks (based on Gross et al. [GAHI05])

Information revelation is one of the most important entities in passive attacks, as they provide a good information source for further attacks. Gross et al. analyzed *Facebook* and showed potential attacks by just using the information presented on

profile pages [GAHI05]. This paper is one of the first that describes passive attacks in social networks.

The study shows that *Facebook*, and probably other social networks too, provide an enormous amount of information about their users. For example, 90.8% of the profiles of the dataset they acquired contained a picture, 87.8% revealed their birth date, 50.8% their current residence. Furthermore, most of the users convey information like their dating preferences, current relationship status and the name of their partner, political views, interests, jobs and many more. They show that just by viewing profiles, one is able to connect first and last name, their residence and birth date.

Regarding the validity of the data, they found that 89% of all names may likely be real names, whilst 8% did use a fake name. The remaining 3% just disclosed their first name. Even if they are not forced to provide their full names, the vast majority does so. Looking at other data, around 98.5% disclosed their quite identifiable birth date, 61% did provide a thorough identifiable picture of themselves, 19% were semi-identifiable and 8% did provide a group image of altogether 90.8% users who provided a picture.

Furthermore, just a small number actually did change the default privacy preferences and therefore their fully identifiable information is available to every user registered at *Facebook*.

The possibilities of attacks based on the data mentioned above are endless, however the authors outlined a few. Basically, they all state that a large amount of data about a single person is available and accessible without authorization by that person. Possible attacks contain stalking, cyber-stalking, demographics re-identification, face re-identification, theft of social security numbers and identity theft.

The authors claim that there is a large amount of people who are willing to provide large amounts of personal information, while the user's notion relating to privacy risks is often unconcerned. Furthermore, they show that they expose themselves to various risks and make it extremely easy to create records about them. In addition, those risks are not unique to *Facebook*.

# CHAPTER 3

# SELECTION OF SOCIAL ENGINEERING ATTACKS

The previous chapter has shown sample social engineering attacks, the threats it exposes and their countermeasures. The work wants to assign now selected social engineering attacks to social networks and find out what new threats they disclose. Furthermore, the attacks should constitute a part of the demand of the prototype, which will be developed.

All of the following attacks are described by Mitnick and Simon [MS03] and happened in the past.

## 3.1 Phishing Mail ([MS03, pp. 97-100])

The first attack is an attack against a private person, with no affiliation at a company. A phishing mail is used to trick the victim into revealing his username and password of an online payment service. A definition of the term *phishing* is as follows [JJJM07]:

> *Phishing is a form of deception in which an attacker attempts to fraudulently acquire sensitive information from a victim by impersonating a trustworthy entity. Phishing attacks typically employ generic „lures“.*

The enormous threat that phishing attacks can present was already shown in section 2.2.1. The following scenario was therefore chosen because of its relevancy and the fact that just individuals are being attacked.

The victim, a retired insurance salesman named Edgar, received an e-mail from *PayPal*[1], which is a company that offers a fast and secure way to make online payments. Edgar did use *PayPal* often, as he was a collector of antique glass jars. Therefore he used the service several times a week. The e-mail he received during the holiday season 2001 was looking officially from *PayPal*, offering him a requital for updating his *PayPal* account. The message looked like the following [MS03, p. 97]:

> Season's Greetings Valued PayPal Customer;
>
> As the New Year approaches and as we all get ready to move a year ahead, PayPal would like to give you a $5 credit to your account!
>
> All you have to do to claim your $5 gift from us is update your information on our secure Pay Pal site by January 1st, 2002. A year brings a lot of changes, by updating your information with us you will allow for us to continue providing you and our valued customer service with excellent service and in the meantime, keep our records straight! To update your information now and to receive $5 in your PayPal account instantly, click this link:
>
> `http://www.paypal-secure.com/cgi-bin`
>
> Thank you for using PayPal.com and helping us grow to be the largest of our kind!
>
> Sincerely wishing you a very "Merry Christmas and Happy New Year,"
>
> PayPal Team

One might notice several signs in the message that can lead to the belief that something is wrong with the e-mail. For example, the semicolon after the greeting line,

[1] `http://www.paypal.com`

the blemished text about *„our valued customer service with excellent service and in the“* and most noticeable the URL, which does not lead to `http://www.paypal.com` but to a different domain. Edgar clicked on the link, entered the information requested: his name, address, phone number and credit card number. He then waited for the $5 gift to show up, but what showed up was a list of charges he never purchased on his credit card bill.

The victim is not a single one, there are many scams which look like the above. The attack was also prepared properly, as they knew that Edgar was a PayPal customer. If he would not have been, the attack certainly wouldn't have worked.

To make the attack even more realistic and effective, the message could have also been sent to the victim's friends and colleagues, if they were to be a member of the same electronic payment service.

| Information | Required |
|---|---|
| Real name | ✓ |
| E-mail address | ✓ |
| Knowledge of account of an electronic payment service | ✓ |
| E-mail addresses of friends and colleagues | |
| Knowledge, which friend or colleague has an account on an electronic payment service | |

Table 3.1: Overview of the required data of the phishing attack.

## 3.2 Insider Attack ([MS03, pp. 83-89])

Another real case scenario was an attack, which involved stealing source code from a company containing the encryption algorithms and firmware used in the company radio products. The relevant fact is that this was an attack against a company, involving information gathering about the company structure, security devices used inside the

company and having several employees as victims. Due to its manifold character, this attack could be extended to similar attacks which even could require less information.

The attacker, named *Danny* by Mitnick, began his information retrieval on the Internet. He luckily found a years-old message written by an employee of the affected company and posted to a public readable newsgroup. This message contained a signature including the employee's name, his phone number and workgroup. He now had to check if that person still worked for that company. He called the employee who indeed was still working for the same company and manipulated him to reveal the names of the servers the employees used for development work.

In addition, every employee in the company had a small electronic device called *Secure ID* ancillary to their username and password. The attacker managed to get enough pieces of information about the company together to masquerade as a real employee. He now had an employee's name, residence address, phone number, department and employee number, the manager's name and phone number. Furthermore, he knew the servers he needed access for.

Danny now waited for a snow storm in the location of the employee's residence. As it was winter, he did not have to wait very long and he launched the attack. He telephoned the IT department and talked to a computer operator named Roger Kowalski.

The attack went as follows [MS03, p. 86]:

> *Danny:* „This is Bob Billings. I work in the Secure Communications Group. I'm at home right now and I can't drive in because of the storm. And the problem is that I need to access my workstation and the server from home and I left my Secure ID in my desk. Can you go fetch it for me? Or can somebody? And then read off my code when I need to get in? Because my team has a critical deadline and there's no way I can get my work done. And there's no way I can get to the office–the roads are much too dangerous up my way."
> *Roger Kowalski:* „I can't leave the Computer Center."
> *Danny:* „Do you have a Secure ID yourself?."
> *Roger Kowalski:* „There's one here in the Computer Center, we keep one for the operators in case of an emergency."

*Danny:* „Listen, can you do me a big favor? When I need to dial into the network, can you let me borrow your Secure ID? Just until it's safe to drive in."
*Roger Kowalski:* „Who are you again? Who do you work for."
*Danny:* „For Ed Trenton."
*Roger Kowalski:* „Oh, yeah, I know him."
*Danny:* „I'm on the second floor, next to Roy Tucker."

The operator of course was uncomfortable walking into the office desk and looking through other people's belongings. But he was uncomfortable not helping either, so he asked his manager and actually vouched for the attacker. The manager wanted to speak to the attacker personally and the operator gave him the name and phone number.

The attacker then called the manager and explained the same story again, mentioning that his team has a critical deadline. The manager then allowed him to use the *Secure ID* device in the IT department just for the weekend. From now on, Danny just needed to call in and ask the operators to pass him the *Secure ID* token. Furthermore, he was able to get a temporary account to bypass the firewall restrictions, directly from the operator. So, he now had the whole weekend to find a security hole which he found.

## 3.3 The Bank Heist

The last attack was chosen due to the fact that it uses information, which is not available on any social networks or the Internet. Again, this is an attack against a company, including information about several employees, the company structure and the already mentioned secret information.

The bank heist was conducted in 1978 by Stanley Mark Rifkin. It was the largest bank robbery in U.S. history by that time. He stole 10.2 million U.S. dollars through wire transfer just by using a telephone and social engineering techniques.

| Information | Required |
|---|---|
| Colleagues | |
| Name | ✓ |
| Phone number | ✓ |
| Department | ✓ |
| Employee number | |
| Manager | |
| Name | ✓ |
| Phone number | |
| Residence of an employee | ✓ |
| Weather information | ✓ |
| Company and employee structure | ✓ |
| The knowledge, what security devices were used in the company | ✓ |
| The servers names and location | ✓ |

Table 3.2: Overview of the required data of the insider attack.

Rifkin was working for a company under contract developing a backup system for the Security Pacific National Bank and more specific for the wire transfer system. As he was working inside the wire transfer room, he had the possibility to learn the process of bank wire transfer. As the bank changed the required transfer code daily, most employees did write the code down, in order to not have to remember it every day. Then he tried to adept the names, room numbers and employee numbers. One day he additionally memorized the daily code and executed a wire transfer over a callbox outside of the bank. Rifkin transfered over 10 million U.S. dollars to a bank in Switzerland by just using the information he already had gained and social engineering techniques.

| Information | Required |
|---|---|
| Employee of the wire transfer room | |
| Name | ✓ |
| Employee number | ✓ |
| Employee outside the wire transfer room | |
| Name | ✓ |
| Department | ✓ |
| Employee number | ✓ |
| Phone number of wire transfer room | ✓ |
| Current wire transfer code | ✓ |

Table 3.3: Overview of the required data of the phishing attack.

Then he picked up the money in Switzerland, changed it into diamonds and flew back to the United States. When he tried to sell them in the U.S., the FBI quickly caught him.

# CHAPTER 4

# ANALYSIS OF THE SOCIAL NETWORK *TWITTER*

The chapter takes a closer look at a concrete social network in order to apply the previous described attacks. Of course there are many social networks, which can be used for harvesting data, however, the study wants to concentrate on a big social network, which is widely used by employees and individuals.

The requirements for the choice were, as already mentioned, a large number of users, content, which is created by the users themselves and which is relevant for them and an API, which can be utilized by the prototype for harvesting the data. Two big social networks fulfilled the requirements: *Facebook* and *Twitter*. Both are widely employed, have a large number of users, contain a lot of content created by the users and an API which can be accessed by a *REST* interface.

*Facebook* however seems to be more restrictive than *Twitter* when it comes to allowing an application to use their API, as each application must have an application

key supplied by *Facebook*. Furthermore, the *Facebook* API[1] is quite big and does not support the harvesting of data that much.

The *Twitter* API[2] on the other side is constructed for getting and setting data, which supports the development of a prototype. Furthermore, using the large number of messages written by each *Twitter* user can tell a lot more than just using static fields, e.g. interests or similar. This will be shown later in this chapter. At last, *Twitter* was chosen due to many already existing programs and platform bindings[3].

*Twitter* is a popular social networking and micro-blogging service that enables users to post messages and to let other users read those messages. The term *micro-blogging* describes a form of communication that consists of brief messages in text form, which then can be sent by a variety of ways, e.g. instant messages, mobile phones, e-mail or other [JSFT07].

Micro-blogging is relatively new, though already widely used and provided by services like *Twitter*[4], *identi.ca*[5], *Jaiku*[6] and others. It is a faster way to communicate compared to other means of communication, like blogging [JSFT07]. It requires less time to think and write a message and this of course implements much quicker *write rate*. This is of course one of the main differences between blogging and micro-blogging, where an author spends several minutes or up to hours to think up and write a blog post. As it now takes less time to think and write a message, the frequency increases and a micro-blogger may post several messages a day.

*Twitter* is one of the most popular micro-blogging services [JSFT07] and currently exhibiting more than a million users. The accurate number unfortunately is not available. However, the estimation is about 1.4 million users in 2008 [KGA08] and more than 1.78 million users in 2009 [Whi09].

[1] `http://wiki.developers.facebook.com/index.php/API`
[2] `http://apiwiki.twitter.com/Twitter-API-Documentation`
[3] `http://apiwiki.twitter.com/Libraries`
[4] `http://www.twitter.com`
[5] `http://identi.ca`
[6] `http://www.jaiku.com`


Figure 4.1: An example Twitter Homepage, featuring several *Twitter* messages, account information and the users following this account.

The social network limits the length of the messages to 140 characters. The messages are called *updates* or *tweets*. The scopes of those messages range from events, news, daily life and other interests [JSFT07]. Of course, other services like instant messaging also offers a way to communicate such information, however microblogging allows to do this publicly.

Users may choose to make their messages visible to all users (even those not logged onto *Twitter*) or to just make them available to *friends*. If the messages are marked as public, they will be displayed in the public timeline which can be accessed by the URL `http://twitter.com/public_timeline`. A *friend* is a relation inside the *Twitter* platform and allows a user to *follow* the messages from other members who are added as a *friend*. Users who are not friends to another user can still follow the user who they are not friends with, but are then called a *follower*. The friend relation is not forced to be mutual, but can be single-way too.

The main types of user interactions are daily chatter, conversations, sharing information and reporting news [JSFT07]. This of course leads us to question, which information can be dangerous and which can be used by a social engineer.

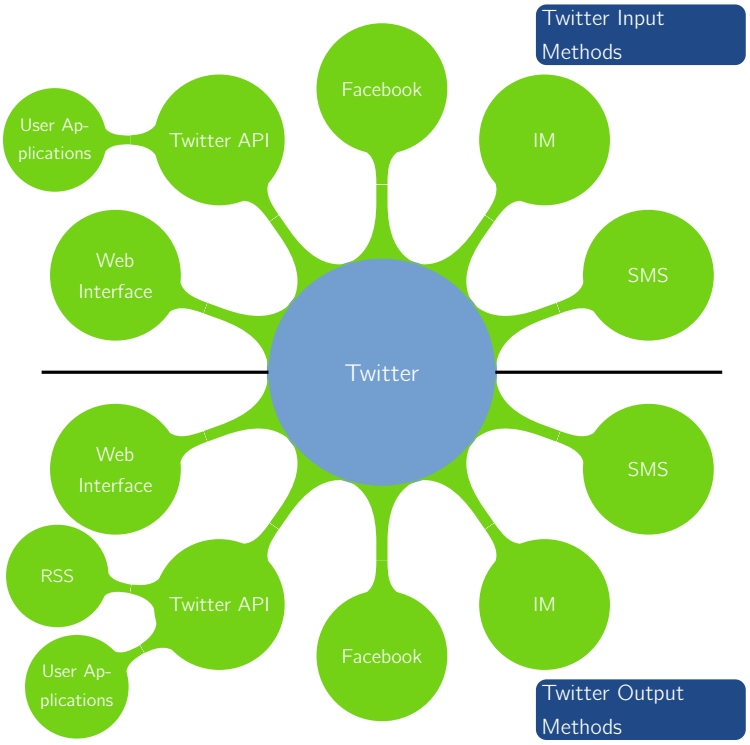



Figure 4.2: *Twitter* input and output methods, on the basis of [KGA08].

A sample *Twitter* profile page is shown in figure 4.1. It features several elements, which will be discussed in the next section.

*Twitter* messages can be sent and received by a variety of methods, all listed in figure 4.2. Several of those are or were discontinued for a certain amount of time, but either re-enabled or provided by a third party company.

## 4.1 Twitter Profile Data

As already mentioned, a *Twitter* profile page consists of a series of elements, which can be used for harvesting data. The complete profile data, which can be accessed is described in appendix A. A social engineering attack does not require all elements, however those needed are described in section 4.3.

## 4.2 Message Classification

In order to see, what and how messages can be useful for an attack, a social engineer might be interested in a classification of the messages of the victim. Java et al. describe a classification of the messages on the *Twitter* platform [JSFT07]. After they gained a dataset, the author and his team tried to categorize each message. The results were the following:

**Daily Chatter** Most users on *Twitter* post about what they are currently doing or similar things. Of course, they are answering the question *„What are you doing?“*, which can be found over the input field of a message. This is the most common use of the *Twitter* platform.

**Conversations** There are two ways to communicate with another user: one is by public messaging, which features the @ symbol like stated above, the other one is private messaging, which uses the D character. About one eighth of all posts in their dataset contained a public conversation and about 21% of the users in their dataset used this type of messaging [JSFT07]. Of course, there is no data available about private messages.

**Reporting News** Another big piece of the cake is done by users who report or comment news and events. Some of them are automated and post weather reports, other are eyewitnesses and report events in their location.

**Sharing Information/URLs** Many users also share information and websites and therefore utilize services, like *TinyURL*[7] or *tr.im*[8]. According to Java et al., about 13% of all posts in the dataset contain at least one URL [JSFT07].

## 4.3 Relevant Data and Security Risks for Individuals and Companies

### 4.3.1 The Challenge of Extracting Data Automatically

The *Twitter* social network gives access to a *REST* API, which can be used by calling either `twitter.com` or `search.twitter.com`. The first one is used for account settings and methods, posting updates and several other options related to users, their relationships and updates. The latter one is utilized for searching for certain updates of users or users themselves. The API supports *JSON*, *XML*, *RSS* and *Atom* as output formats, although just *JSON* is supported on every API method.

Most API methods require a username and a password, however this is not a problem in itself, as a *Twitter* account can be created within minutes by using an e-mail address. The real challenge however is the limit of 150 API calls per hour. If a user is logged in, he has 150 calls, which he can utilize on API calls. If the user is not logged in, i.e. an anonymous user, the IP address is used for tracking him. To gather a very deep information visualisation of a user, 150 calls are not enough, as for example many users have more than 150 friends. However, there are three ways around this problem: The first one is to just use the important data, leaving a deep information visualisation of e.g. friends aside, the second is to create many accounts and then change the accounts during acquiring the data. The last one is not to log into *Twitter* and acquiring the data anonymously, while changing the IP address after 150 calls. However, some API methods require the user to be logged in. A compromise between these methods might be the best option, e.g. acquiring as much data as possible about

[7] `http://tinyurl.com/`
[8] `http://tr.im/`

a single user and then repeat the same for several other users who are in contact with him and might be interesting for an attack.

### 4.3.2 Ontology and Classification of the Data

The *Twitter* API offers a wide range of methods, admittedly in the first place, the attacker is mostly interested in gaining information and not that much in using the API methods to do actions. Additionally the attacker is mostly interested in a certain person or group, therefore just the methods, which allow gathering data about a certain user or group make sense for this work. For a social engineering attack, the following data is interesting.

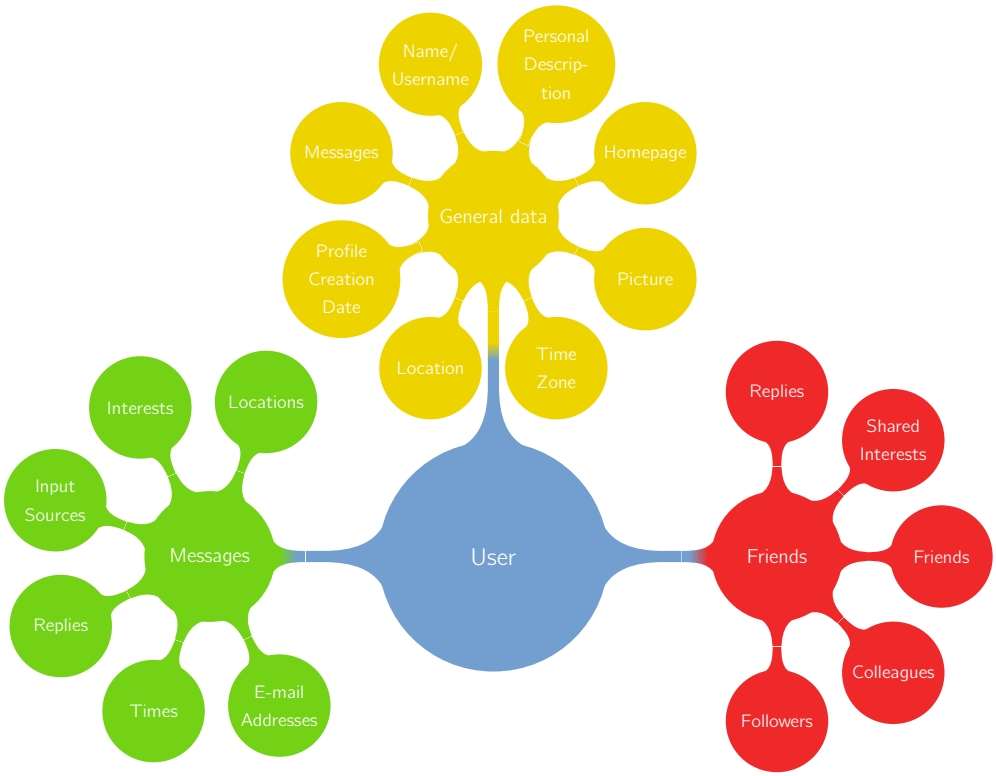



Figure 4.3: Graphical representation of the data classification.

#### 4.3.2.1 General Data about the User

*Twitter* offers some static fields, which can be altered by the user, but remain mostly the same over the year. Some other information is created automatically and still accessible by other users.

Most of the fields described in appendix A are already usable for gaining a penetrative overview of a person.

**Name** Gives the real name of the user, which can be used for phishing mails or social engineering attacks, where a name is needed.

**Username** Many users have the same username on different services, like e-mail or other social services. This can be used for some types of attacks.

**Description** This description might be useful for gathering other information, such as workplace or interests.

**Homepage** A personal homepage might give additional information about the user.

**Picture** Shows how the person might look like.

**Time Zone** Gives an idea, where the person lives or works.

**Location** Together with the Time Zone, an attacker now could actually define his work place and home more precise.

**Profile Creation Date** Tells the attacker since when the user is on *Twitter*. That might be a good indication, whether he also uses other services and since when.

**Messages** The Message count could tell how computer active a person is or how interested he is in a certain topic.

**Followers** An interpretation of the followers count may say how famous he is and how much effect he could have on other people.

**Friends** While the number of friends is not that interesting, the friends can be used for attacks too, as most of them are real friends or colleagues.

#### 4.3.2.2 E-mail Addresses

As already described earlier, the e-mail addresses of the users are not visible throughout *Twitter*, though the addresses can be gathered in most cases. Brown et al. for example described that an e-mail address can be constructed out of a naming scheme of given data, like the username or the real name [BHI+08]. Another possibility is to look through the messages for anything that looks like an e-mail address. Quite often users write messages, about contacting them at a special address or that they changed to a new one.

#### 4.3.2.3 Times

An analysis of the time to write messages can be a good instrument to track down working hours and working days or periods the user is using a computer. Additional hours, the user is not online can be tracked too, for example when the user is on holiday.

#### 4.3.2.4 Replies and Friends

*Twitter* itself does not publish any user's friends, however it promotes followers and users who follow the candidate. With this data however, the *hidden* network can be identified, which basically is a graph that only consists of bidirectional connections. This is the network, the user is actually friends with. By analyzing the replies of messages, one can understand, with who of the friends, the user has most contact. This gives a quite good representation of the users friends or colleagues.

#### 4.3.2.5 Input Sources

The *Twitter* input sources give an overview, what methods the user utilized to post updates. For example, if he is using a *Twitter* client that is only available for a certain operating system, this could be exploited. If the source is mostly SMS, the location

is getting even more important, as location tracking could be done in this case. Also there could be a relation between the input method and the computer experience the user has, if he is using external clients or not.

#### 4.3.2.6 Interests

In most messages, the users represent a certain interest in a single topic. This is either done by marking the topic explicitly with a hash sign or by just using text to describe the interest.

#### 4.3.2.7 Locations

Also, many messages include information about the current location, for example if they arrive at the workplace, at conferences or other. This data can be filtered by analyzing the messages. A promising project that could be used here is *TwitterData*[9] which marks keywords as key value pairs and actually promotes keys like *location*, *lat* and *long* for location, latitude and longitude positions.

#### 4.3.2.8 Text Search

Finally a text search can be used to get further information about a certain topic the user wrote about.

## 4.4 Threats and Risks

All of the above mentioned fields and data, create a threat to the user himself and also to other users who are in contact with him. The threat and risk does not directly come from the information gained, but from the fact that every piece of information a social engineer can bring together, builds an even better base or starting point for

[9] `http://twitterdata.org/`

a social engineering attack [MS03]. This means that the information itself, does not create a threat, however, the social engineering attack does. Even with useless information at first glance, an attack can be driven. For example, if a user shares his location and what he is doing job-wise, his workplace and position can easily be found.

There are various scenarios, which are possible using different data. One can create a quite expletive ontology about a person just with the fields and data mentioned above, further attacks are supposable.

Another matter is the almost hidden information retrieval possibility. If a social engineer has to create a contact to his victim or to other people who are in contact with the victim over the telephone or e-mail, the attacker could be detected. In this scenario however, only the provider of the social network can track the information retrieval. If the attacker now does not stand out in relation to other retrievals, for example normal clients, which are accessing the social network, there is probably no way to detect the attacker. This is a very vital matter for a successful social engineering attack.

The prototype, which is described in the next chapter, will make use of exposed data, building a fact sheet about the user and people who are in contact with each other. This is then used to create a social engineering attack, showing that attacks based on data of social networks are already possible.

## 4.5 Countermeasures

The nature of a social network, especially *Twitter* is sharing data. Therefore it is difficult to encounter attacks and there has to be a trade-off between utility and security [BHI+08]. Given also, that most data come out of analyzing the messages, there are not many ways to exclude the risk. However, there are various methods to minimize the likeliness of a social engineering attack, based on the data published on *Twitter*.

### 4.5.1 Protection for the User

The first way to minimize the risk, would be to protect the updates, which basically means that only users who are approved by the author may see them. This however lacks in two facts: first, the fields described in section 4.3.2.1 are still visible. Second, the attacker could just become a friend with the victim and gain access to the messages.

Another, possibly extreme, way would be not to publish sensitive information about the user himself, his workplace, friends and other. This is of course quite hard to accomplish, as the whole idea about *Twitter* is to share information. Even, if a user would accomplish this, lots of information would be still easily accessible, like the time analysis, his friends, colleagues and much more.

At last, a user could not utilize a social network like *Twitter*. This certainly would exclude the chance of being a victim in a social engineering attack, based on the data found on *Twitter*. However, everyone should decide if this solution is suitable.

### 4.5.2 Protection through Twitter

Having a look on the whole social network, there are not many solutions, in which *Twitter* could minimize the risk either.

One possibility would be to require each user to be logged in to view the messages of other users. This however would detach the *Twitter* social network from the outside and probably not the right way for a commercial oriented company. Anyhow, if that would be the case, an account could easily be made and so there wouldn't be any surplus.

At last, the social network could limit the API calls and track them more precisely, so that an automatic extraction is no more possible or at least made harder. But to decide, which data is harmless and which could be used for an attack is very difficult, as both fetch almost the same data. The only difference could be the fact that an attacker is interested more into a single user, while an unoffending *Twitter* client might

be more interested in many users. Although, if there would be a way to accomplish this, this is probably the only countermeasure, which really can hinder an attacker of gaining information.

# CHAPTER 5

# DESIGN, ANALYSIS AND IMPLEMENTATION OF A PROTOTYPE

This chapter is going to introduce the design, analysis and implementation of the prototype, which implements the analysis of the previous chapter.

## 5.1 Design Goal

The goal is to develop a prototype, which can cull and evaluate data of one or more users of the *Twitter* social network. The produced data should then be assigned to real people, in order to create a fact sheet for a social engineer. It should be shown that it is easy to file employees of a certain company or a private person, just by using legally available data, put online by the individuals themselves. The filed data should then enable a social engineer to do social engineering attacks against the weakest point. Furthermore, the attacks, which are analyzed in chapter 3, can be repeated by using the prototype.

The prototype developed for this work allows to find individuals either by their username or by their real name. Then it evaluates and displays every possible information, which can be gathered legally, based on the analysis in section 4.3.2. The data is saved locally, with the possibility to update it.

## 5.2 Used Programming Languages, Tools and Libraries

For the prototype, the *Python*[1] programming language was used. It is a dynamic object-oriented programming language that offers strong support with many tools and comes with a large amount of standard libraries. *Python* allows rapid prototyping and has built in many libraries, used for example to call the *Twitter* API.

The plotting was done using the 2D plotting library *matplotlib*[2], which produces production quality figures and is easy accessible through the *Python* programming language.

## 5.3 Design and Implementation of the Prototype

The prototype was developed in a modular approach, to allow easy extensibility. As the *Twitter* API is in constant change, this helps to keep the prototype working and allows to extend it for new data sources.

Each information source is developed as a plugin. The plugin downloads the required data, parses it and prepares an output, from which a fact sheet gets generated.

The structure of the prototype is outlined in figure 5.3. The starting point is the `Prototype` module, which holds the starting values, loads the plugins, runs them and puts together the output. To allow further searches for people, the main module holds several options. The prototype currently supports the following:

[1] `http://www.python.org/`
[2] `http://matplotlib.sourceforge.net/`

- Search and analyze a user. If the username passed is not found, a search is launched.
- Search users by using specific keywords
- Search users in a specific location
- Search users in a specific range

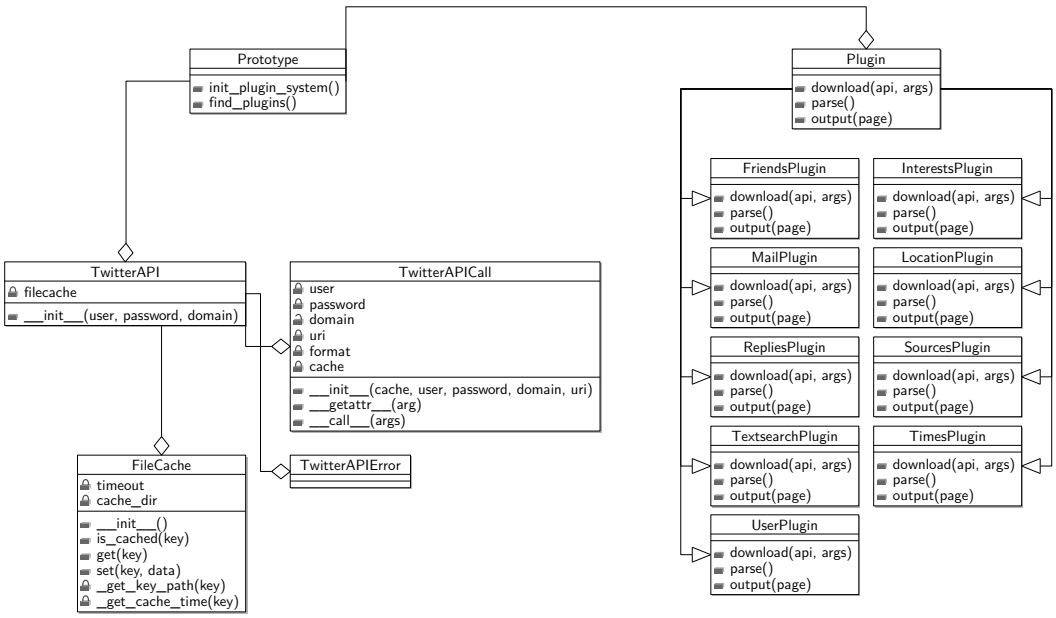



Figure 5.1: UML class diagram of the developed prototype.

After choosing the option, the API object is created, passing the username and password. If the username and password is empty, the prototype connects anonymously to the *Twitter* API.

```
38    twitter = TwitterAPI(conf.username, conf.password)
```

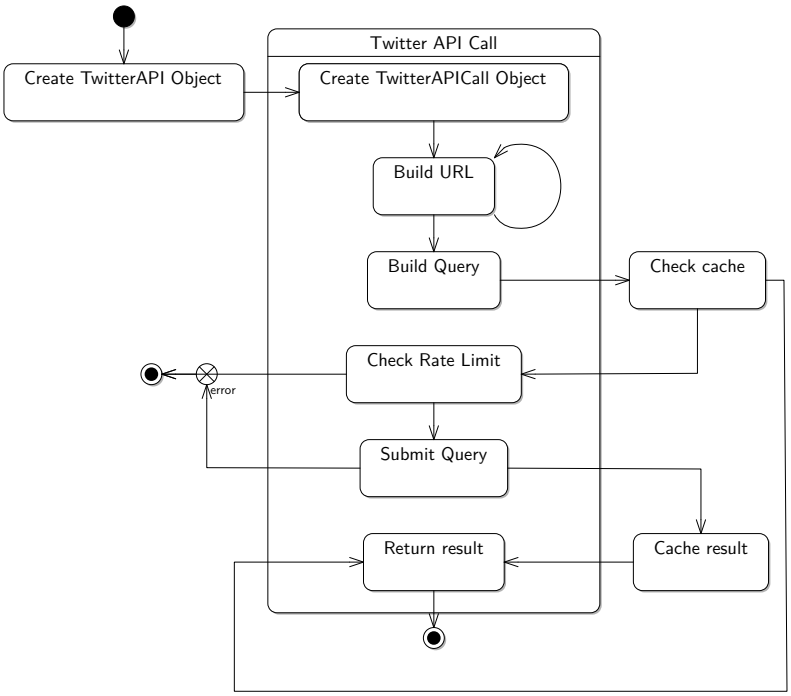



Figure 5.2: UML state machine diagram of an *Twitter* API call.

First, the API object gets created, then it calls the `__init__()` method of the `TwitterAPICall` object. This means that the prototype has an object of the `TwitterAPI` object, which then creates a new `TwitterAPICall` object each time called. At the same time the `FileCache` object is created and passed on, which simply stores the values downloaded into files.

```
152  class TwitterAPI(TwitterAPICall):
153
154    def __init__(self, user=None, password=None, domain="twitter.com"):
155      filecache = FileCache()
156      TwitterAPICall.__init__(self, filecache, user, password, domain, "
             ")
```

When the `TwitterAPICall` object is created, it fills the default values with those passed by the `TwitterAPI` object.

```
55  class TwitterAPICall(object):
56
57    def __init__(self, cache, user=None, password=None,
58                 domain="twitter.com", uri=""):
59      self.user = user
60      self.password = password
61      self.domain = domain
62      self.uri = uri
63      self.format = "json"
64      self.cache = cache
```

The `__getattr__()` method, which is a Python default method, is called every time an attribute of an object is accessed. The prototype exploits this behaviour for the *Twitter* API. For example, if the prototype wants to call the API function `/users/show`, it just calls the method `twitter.users.show()`. The previous mentioned function then just removes the non-existing attribute and appends it as an argument to a new `TwitterAPICall` object.

```
66    def __getattr__(self, arg):
67      try:
68        return object.__getattr__(self, arg)
69      except AttributeError:
70        return TwitterAPICall(self.cache, self.user, self.password, self
              .domain,
71                              self.uri + "/" + arg)
```

If there is no attribute left, the object function is called, where the prototype exploits the Python intern function `__call__` to actually connect to the *Twitter* API and download data. The actual downloading is quite trivial. The method just determines if a POST or GET HTTP request is needed, puts together the username and password, if given, builds the request URL. Then, it looks if the query is already

cached and if there are API calls available. If so, it does the actual call to the *Twitter* API. The result is then cached by using the `FileCache` object.

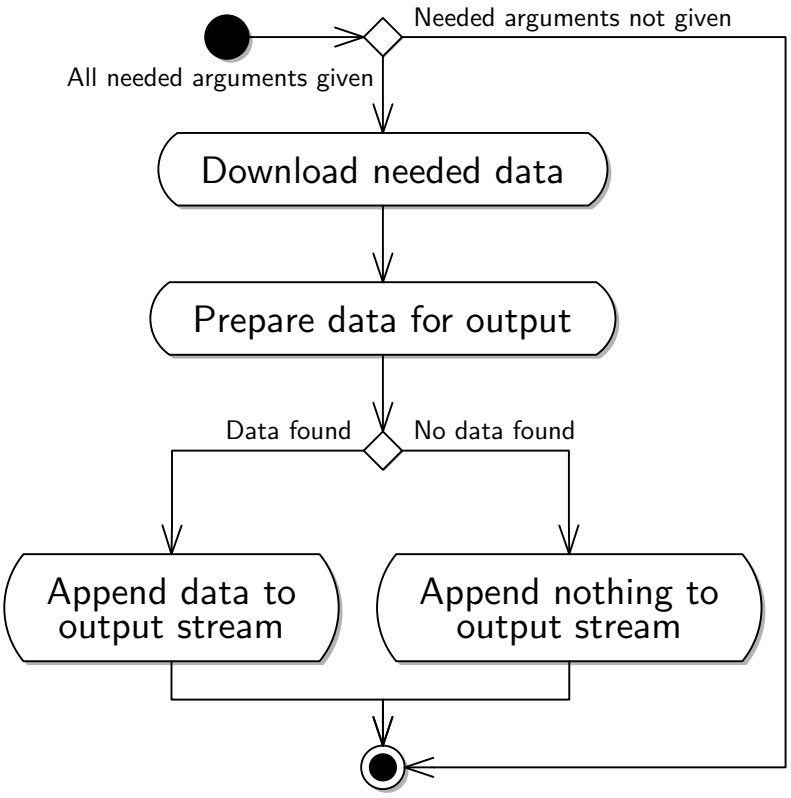



Figure 5.3: Activity diagram of a generic plugin.

Next, the plugin system is initialized, which simply imports the plugin files by using the Python method `__import__()`. The list `PLUGINS_ENABLED` just holds the plugins, which should get loaded, run and displayed.

```
20  def init_plugin_system():
21    if not PLUGIN_PATH in sys.path:
22      sys.path.insert(0, PLUGIN_PATH)
23    for plugin in conf.PLUGINS_ENABLED:
24      __import__(plugin)
```

All plugins are inheritances or subclasses of the `Plugin` object, which just holds the methods `download()`, `parse()` and `output()`. Those methods have to be implemented by each plugin. Figure 5.3 shows, how a single plugin works.

```
8   class Plugin(object):
9
10    def download(self, api, args):
11      pass
12
13    def parse(self):
14      pass
15
16    def output(self, page):
17      pass
```

Now, each plugin has to be addressed somehow. As it was dynamically loaded, there is no structure, which holds the plugin itself yet. However, the `Plugin` objects knows which subclasses or children were loaded. The method `find_plugins()` returns a list of the loaded plugins. If a plugin was not created yet, it creates the specific plugin object.

```
27  _instances = {}
28
29  def find_plugins():
30    result = []
31    for plugin in Plugin.__subclasses__():
32      if not plugin in _instances:
33        _instances[plugin] = plugin()
34        result.append(_instances[plugin])
35    return result
```

The prototype can now call the three methods `download()`, `parse()` and `output()` of each plugin. The output gets merged and written to a file.

If a new plugin should be added, a new plugin object has to be created, which implements the above mentioned methods. Each API call gets cached and therefore

it is no overhead if several modules need the same data to work with. The API object just downloads the required data once and then just passes the cached data.

```
1  import sys
2  import os
3
4  from plugin import Plugin
5
6
7  class TestPlugin (Plugin):
8
9    def __init__(self):
10     self.data = None
11
12   def download(self, api, args):
13     if (args and "id" in args):
14       print "args with id"
15     else:
16       print "args without id"
17
18   def parse(self):
19     print "parsing"
20
21   def output(self):
22     print "output"
```

# CHAPTER 6

# EVALUATION

This chapter will try to evaluate the new threats disclosed by the prototype. First, some sample attacks are driven and then analyzed. In the end, this chapter will try to extrapolate the possibility of further social engineering attacks.

## 6.1 Demonstration of Attacks Using the Prototype

This section wants to demonstrate how the prototype, which was developed in the previous chapter, can assist a social engineering attack. Therefore, the attacks previous described in chapter 3 are repeated together with the aid of the prototype.

## 6.2 Phishing Mail

The attack itself was already described in section 3.1. It will now be driven against a sample profile on *Twitter* using the prototype. The attack is visually represented in figure 6.3. Following data is needed:

- Real name of the victim
- E-mail address of the victim
- The Knowledge that the victim is a customer of an electronic payment service, like *PayPal*, *Amazon* or *eBay*

The attacker now simply starts off with a simple keyword search, as he is not interested in a specific person (at least for now). Therefore he runs:

```
1   $ python prototype.py -k "paypal"
```

This opens a new website on `http://search.twitter.com`, which shows the attacker people who quoted the word *paypal* in one of their messages. The attacker now picks a person with the username *petersample* who writes

> did really some transactions over paypal this week, works really great!

As the attacker now has an individual he can attack, he is interested in more information. Therefore he runs:

```
1   $ python prototype.py -u "petersample"
```

Now this produces a fact sheet about Peter Sample. As the attacker already knows that the victim is using *PayPal*, he just needs the real name and the e-mail address. The output of the prototype is displayed in figure 6.1. The attacker now knows the real name. The e-mail address was also found by the prototype and is displayed in figure 6.2. Since every data needed for the phishing mail is acquired, the attacker can begin to send his e-mails out. As described, it is also possible to include Peter

Sample's friends, if they are using *PayPal* too. This is simply done by redoing the attack on Peter Sample's friends, which the prototype also outputs.

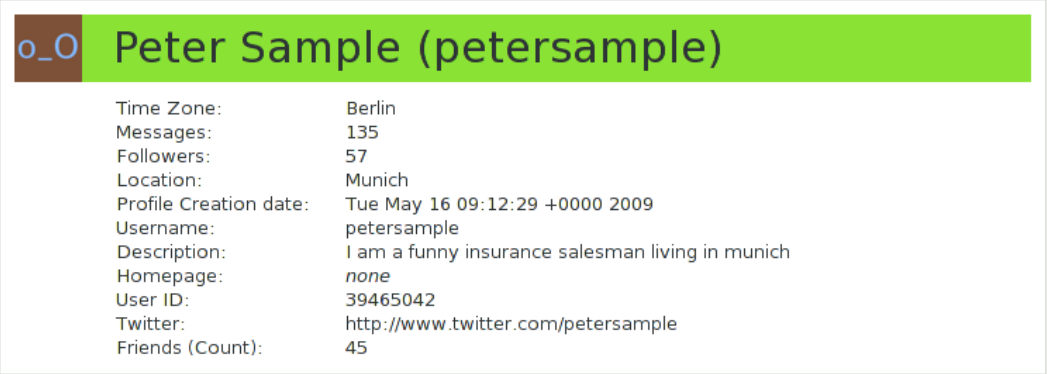



Figure 6.1: Prototype output: general information about a user.


Figure 6.2: Prototype output: e-mail address was found.

## 6.3 Insider Attack

In the attack described in section 3.2, the attacker starts again with a keyword search, as he wants to find an employee of Sample Company Inc. Quite quickly he finds *petersample* who seems to work for one particular company. He launches an information retrieval about that user and gets an output like in figure 6.4.

Now he knows the real name, the company and department Peter Sample works for, the workgroup inside the company and has an approximate idea where the victim could live. A text search about phone specific terms, produces the output shown in

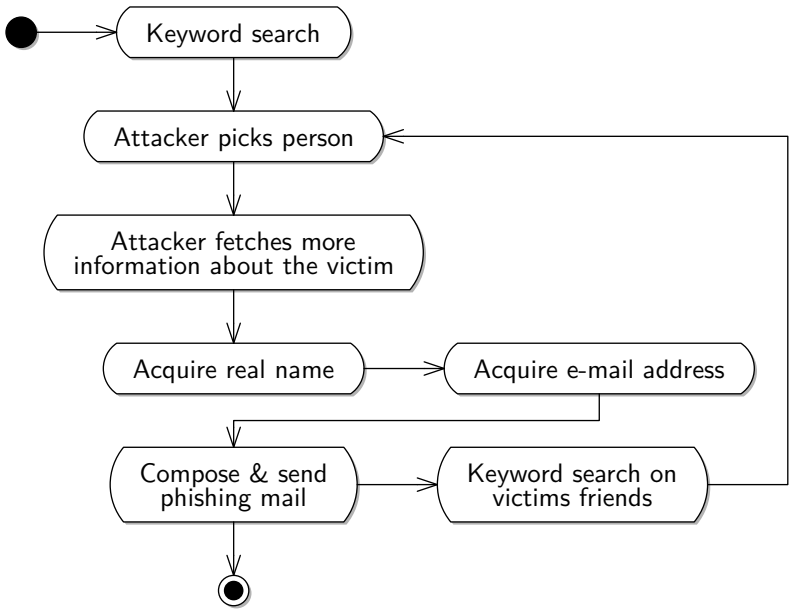



Figure 6.3: Activity diagram of the phishing attack.

figure 6.5. The employee number is trickier to get, as one must rely that the employee posts this number in a message. In this scenario, this is not the case and the social engineer has to call Peter Sample at his phone number.

The prototype also shows the times Peter Sample mostly writes his updates, as shown in figure 6.7, and therefore is most probably available and not having holiday for example. That was also done in the original scenario, however this time much less information is needed, as most data can already be retrieved from the social network. Next, he needs the residence of Peter Sample, which the prototype also supports and is displayed in figure 6.6.

The use of security devices and server names is either determined by the phone call mentioned above or by a text search. A text search works quite good, as many people write some message about private details, which are not working at the moment, such as a broken connection to their development server or that they dislike special security devices. This data can be harvested as the phone number before.

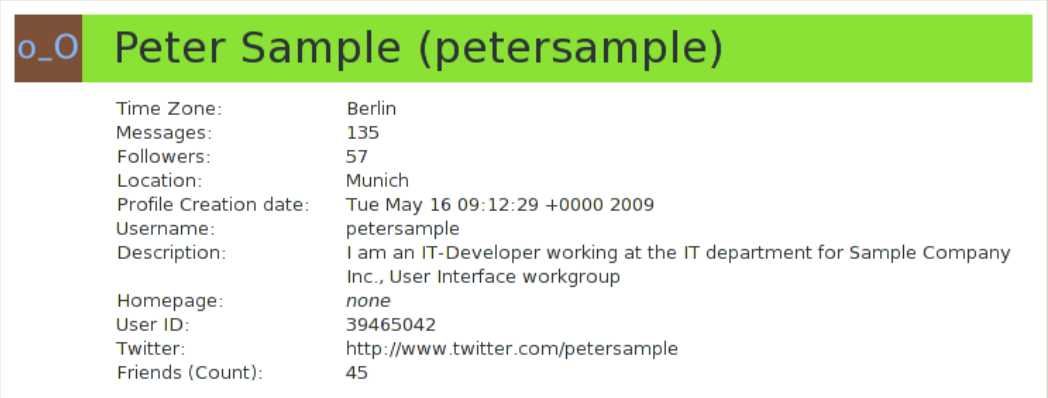



Figure 6.4: Prototype output: general information about an employee of Sample Company Inc.

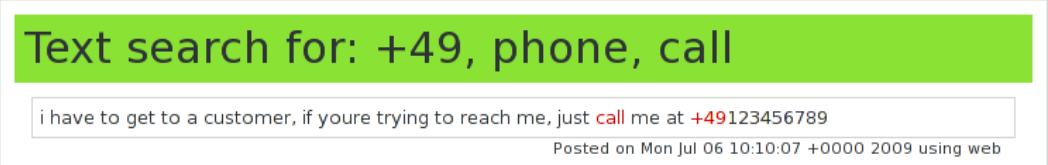



Figure 6.5: Prototype output: text search about phone specific terms.

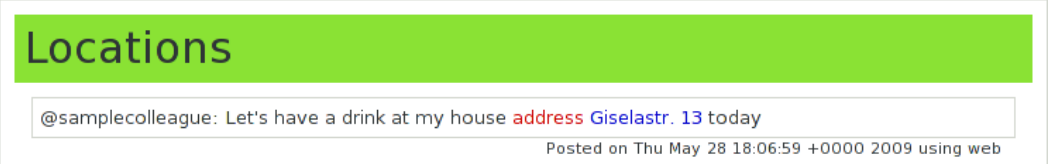



Figure 6.6: Prototype output: the residence is revealed.

The company structure, like the managers name and colleagues of the employee can be determined by analyzing the hidden friends network and the replies the employee made. By further analyzing the friend's networks, replies and applying the same data retrieval as above, a quite detailed company and employee structure can be determined.

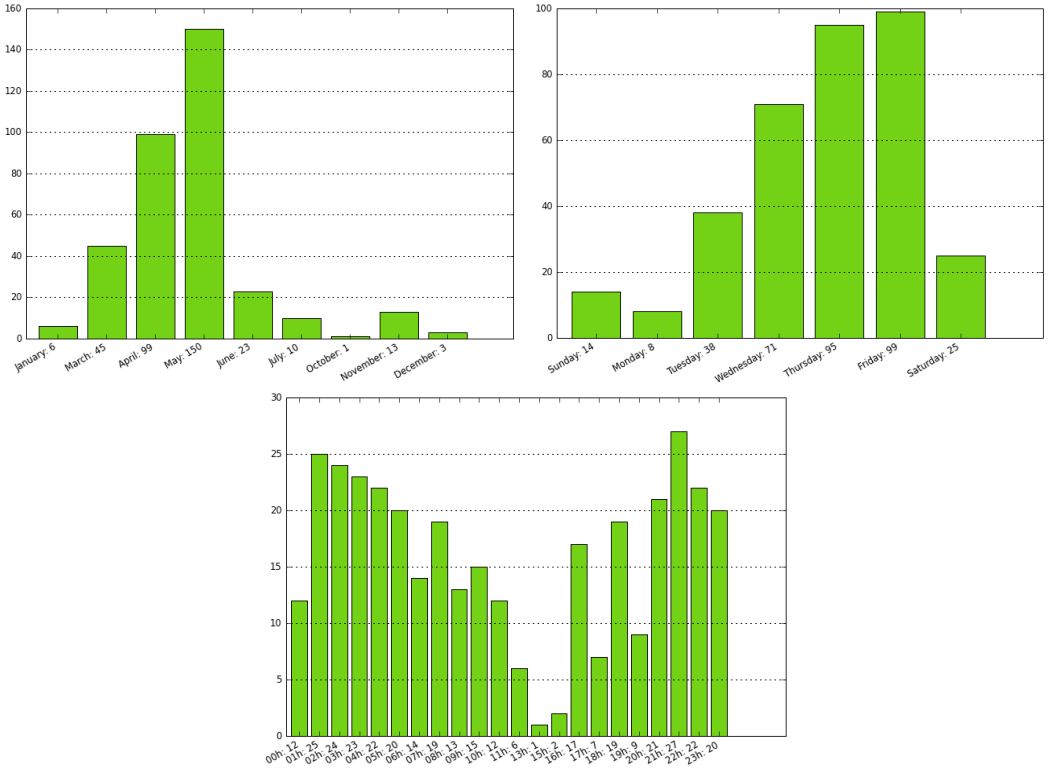



Figure 6.7: Prototype output: the average times a user writes his updates: months, week, day average.

As in the original scenario, the attacker now just has to wait for a snow storm in the area of the residence of the employee, which is easy to check, as he has the residence address.

## 6.4 The Bank Heist

The scenario described in section 3.3 is quite different from the others, as it requires data, which normally is not or at least should not be published. While the first part of the scenario, like names, phone numbers and departments are possible to extract either by using the prototype or by directly exercising social engineering attacks, the

daily wire transfer code is not published. While it still might be possible for a social engineer to get the code, it is not possible to do that over a social network like *Twitter*, as this sensitive data is not published. Though, the prototype can still save a lot of effort, for example by determining the structure of the company or analyzing company employees.

## 6.5 General Feasibility and Extrapolation of Attacks

Attacks and information retrieval are possible with the prototype. That was shown in the above scenarios. One big barrier though remains private and sensitive data, at least if a user of a social network identifies specific data as sensitive. However many users do not and post e-mail addresses, residence addresses, phone numbers, passwords and further more.

Automatically harvesting this data makes the life of a social engineer definitely easier, as he just has to execute the attack and is no more constrained to do an information research beforehand. When it comes to sensitive data, a prototype like the one introduced, can still help to gain data, like company information or user specific information.

With it's plugin architecture, the prototype is easily extendible and more data and analysis can simply be done. For this prototype the *Twitter* social network was chosen, however other famous social networks are possible too. Putting together more social networks and possibly search engines too, one can get an even more complete picture of a victim.

Most of the effort put into the prototype was to do text mining. Though, there is also a lot of optimization possible regarding text analysis. However, the prototype was able to gain much more information, than a simple or even a more complex text search engine could do, for example a web search engine. The prototype offers a wide range of analysis possibilities, all suited for special social engineering attacks. This was shown in the previous sample attacks.

The prototype did get all data legally and can operate almost anonymously, as he won't be tracked, with the small amount of data it fetches. Though it has a big effect and can lead to very dangerous attacks, which can be carried out relatively easy.

## 6.6 New Threats

Critical attacks based on social engineering techniques are possible, that was demonstrated in this work. The question one should pose here is, what new threats this work did discover. As already stated before, the threat is not generated from the possibility to gather information, it is brought forth by social engineering attacks, which are based on such information. The work showed that it is possible to gather large amount of sensitive data of several users fast and almost anonymously. This means that the research phase of a social engineering attack can be almost replaced and done automatically, depending on the attack scenario. By fetching the data automatically and without knowledge of the victim, the attacker can prepare the attack while remaining hidden and not exposing himself to the danger of being discovered.

Taking this approach one step further, automatic harvesting and attacks could be a future scenario. An attacker could define some social engineering attacks, like the phishing mail attack, which was shown before and let them run over a social network. As there is no way to stop such an attack, this would have an enormous impact. Brown et al. [BHI+08] for example were able to create a context-aware phishing mail attack inside a social network, which was extremely successful. This scenario however is not just restricted to phishing mail attacks, but is open to all social engineering attacks, where the attacker no longer chooses a suitable attack for a victim, but the victim chooses the attack by himself, by just exposing sensitive data.

# CHAPTER 7

# CONCLUSION

This work began with the problem and motivation of exploiting social networks using social engineering. It defined the main goal of the work, which is to give an answer to how data of individuals or companies can be automatically extracted from social networks and presented in a way that it can be used for a social engineering attack. Furthermore, countermeasures against automatic extraction and social engineering attacks were analyzed and developed.

A methodology of how to characterize useful information, extraction and countermeasures was defined. This determined the further analysis and the constitution of the work.

Then it tried to define social engineering, the attacks and it's countermeasures. Therefore the social engineering life cycle was introduced, which outlines a social engineering attack and is constituted of four phases: research, developing rapport and trust, exploiting trust and utilizing information. A single cycle of course is not limited to a single cycle, but can contain several other cycles until the objective

is reached. The research phase consists of several methodologies to gain as much information as possible about the victim, which can be used to develop rapport and trust in the next phase. A social engineer will make contact to the victim using the information obtained in the phase before. This trust, which was established is now exploited to procure the information needed, which then is used in the last phase. Also, the threats and risks created through social engineering were described. To obviate social engineering attacks, countermeasures were demonstrated, however there is no way to be absolutely shielded against such attacks. While data retrieval of social networks usually does not count to the methods of the research phase of a social engineering attack, it was shown, how social networks can replace or enhance several *classic* methods. Furthermore, not just the data retrieval, which was described as *passive attacks*, was analyzed, but also attacks, which are exploiting a social network directly and are known as *active attacks*.

Next, three social engineering attacks were analyzed in-depth, in order to apply them to the chosen social network afterwards. The attacks themselves were analyzed, but also the information needed and how they could be applied to a social network. The first attack is aimed towards an individual and should represent this group of attacks. The other two are aimed against companies, both using data, which often is not easy available and has to be constructed by the social engineer. The last one, however requires sensitive data, which is not supposed to be put online.

Then, the *Twitter* social network was chosen as the social network, the attacks should be applied on. It was dissected, while keeping the main focus on the risks this social network exposes and the relevant data, which could be used for social engineering attacks. It was determined, how automatic extraction of data could actually work and the danger of being detected, which is practical not existent. Further the data, which could be extracted was classified in several groups, which then could be used for the attacks. Again the threats and risks of this concrete information retrieval was discussed and the countermeasures against automatic extraction were illustrated. However, there is no way to exclude the threat and risk. All countermeasures however are able to decrease the chance of being a victim of a social engineering attack based on data of a social network.

Having the requirements and the analysis done, the prototype, which can automatically extract data from the *Twitter* social network was designed and implemented. The prototype was developed to quickly adapt itself on the fast changing *Twitter* API, but also on new information sources or different representations. It is able to extract data from the social network and present it in a *fact sheet*, which then can help a social engineer to replace the research phase by just using the prototype and not exposing himself to the victim.

At last the prototype was used to drive three sample social engineering attacks. The attack against an individual was successful and showed that similar attacks are possible too. The second attack was successful too, however including the possibility of having to run another social engineering attack to gain specific data. Though, this attack can be described as successful too, as most of the data required was obtained and a second attack is not always needed. Only the last attack failed, which required sensitive information, which should not be possible to find on a social network. But of course, the prototype managed to retrieve most of the data about the company.

The evaluation showed that using the prototype would make the life of a social engineer much easier, as in most cases, he can use the prototype for gathering information about his victim. This also dwindles the chance of being detected, which could baffle the social engineering attack.

As the attacks driven before already evidence, both targeted groups are vulnerable to such attacks. Regarding to individuals, there is a wide range of attacks possible sprawling from online attacks, such as phishing or identity theft, to even *offline* attacks. Most social engineering attacks, which involve the knowledge of a certain piece of information, seem to be possible by exploiting the information found a social network. And even delicts, which do not involve social engineering seem to be possible. For example as an extreme case not related to social engineering, even burglary would be possible, if a picklock has the information that the victim is on holiday and nobody is in his home.

Concerning companies as the target of the described attacks, the companies themselves are very vulnerable too, as the work showed. The company structure as well

as sensitive data could be extracted from the social network. Combining the automatic extraction with a social engineer who is able to gain even more information and to establish a relationship to several employees is a very dangerous instrument. It was shown that especially large companies were already extremely insecure. Having a method for the automatic extraction of company related information, this just increases the risk of an social engineering attack.

Combining both groups, could result in a devastating result for the victim. An attacker may drive some attacks against several individuals who are related to a specific company. Then he could exploit them to get access at the company, to execute his attack. This is probably the worst case scenario for a company.

The prototype was kept very extensible, in order to easily attach new information sources or different representation of data. Furthermore, it has to be said that the possibilities of text analysis, especially on a social network like *Twitter* is not for a long time bailed out yet. Spending more effort on that would result into a even more detailed fact sheet, usable for even more social engineering attacks and especially for more detailed planned attacks. It could also be extended to support more networks, or even using some search engines, in order to gather even more data about several individuals. This also, would result in a more detailed achievement.

One scenario that will probably be more present in the next years, could be an automatic extraction and attacking framework. As this work has already shown how to automatically extract data out of a social network, it is only a matter of time until somebody will use that data for predefined social engineering attacks. The example of an automated phishing mail was mentioned before in this work.

While social networks become more famous and attractive every day, the security of such networks does not have to be left aside. Much work is still needed in this area, until people can use those networks safely, without having the fear of becoming a victim of a social engineering attack.

# Appendix

# APPENDIX A

## *TWITTER* PROFILE DATA

**Name** This fields tells the real name of a user. It is limited to 20 characters. Further, it is required to enter a name, albeit a bogus name is also accepted. The field is required.
*Example: Peter Sample*

**Username** The username is presented with this field. A username is needed to create an URL, which is of the form `http://www.twitter.com/[username]`. Therefore only letters, numbers and underscore are allowed. The maximum length is set to 15 characters. The field is required.
*Example: petersample*

**User ID** The user ID which is handed out automatically. The sequence is not known, however it is mostly agreed that the user ID is not given out sequentially.
*Example: 1234567*

**E-mail** Though not visible publicly, a *Twitter* user has to provide an e-mail address. Even if an e-mail address is not provided, it can be build out of a naming scheme

and the already given data, like the username and the real name [BHI+08]. If the user is an employee of a certain company, this might be even easier, as companies often have certain naming schemes. The field is required.
*Example: petersample@example.com*

**More Info URL** The URL is used to link a visitor of a profile to further websites, like the blog of the owner. It can link to any site on the Internet. The shown component of the URL is 17 characters. Except XSS prevention, *Twitter* does not rewrite the URL and port numbers after the TLD, spaces, German umlauts and UTF-8 characters were accepted.
*Example:* `http://www.petersample.com/`,
`http://www.peter sample.com:8080/äöüΓΛΣΨ.htm`

**One Line Bio** A short sentence can be shown on the profile page, where a user can describe himself. It is limited to 160 characters.
*Example: I am a computer science expert and work for example.com*

**Location** The location will also be shown on the profile page and is limited to 30 characters.
*Example: Munich*

**Picture** A user can insert a picture of himself, although it is not required to do so. Moreover, if a user publishes an image, it does not have to show himself, but can also be anything else.

**Profile Creation Date** The date the profile was created.
*Example: Fri Nov 02 00:17:11 +0000 2007*

**Following** Describes a list of users the profile owner is *following*. To see the list, one has to be logged in.

**Followers** This gives us a list of users on the *Twitter* network who are *following* the profile owner. To see the list, one has to be logged in.

**Friends** These are users who have a bidirectional connection. That means, that user A is following user B and vice versa.

**Favorites** A user can mark his messages or the messages from other users as a favourites, which displays a yellow star beneath them. A user's favorite messages can be viewed by visiting `http://twitter.com/favorites?user=[username]`.

**Messages** A message is composed by the actual message text, which is limited to 140 characters, the time, when the message was written and by which mean. A message is identified by a unique ID, e.g. 1767572233 and can be accessed by `http://twitter.com/[username]/status/[messageID]`.

*Twitter* also offers special commands, which can enhance a message or turn a *Twitter* into a query tool.

**@username + message** A message can be sent directly to another person, though still visible publicly.
*Example: @petersample yes, you are totally right!*

**D username + message** In contrast to the command above, this message is entirely private and not visible by other users.
*Example: D petersample this is a private message for peter!*

**WHOIS username** retrieves information of any public user on the *Twitter* platform. Currently, this command returns the real name, the date since the user has an account on *Twitter*, the one line bio, the website and the location.
*Example: WHOIS petersample*
*Answer: Peter Sample, since May 2009. bio: I am a computer science expert and work for example.com. location: Munich web: `http://www.petersample.com/`*

**GET username** gets the last message this user posted.
*Example: GET petersample*
*Answer: petersample: This is my latest message (1 day ago)*

**NUDGE username** sends a note to the user, reminding him to post a message.
*Example: NUDGE petersample*
*Answer (sent to petersample): You've been nudged! [my-username] wants*

*to know what you're doing. Reply to this message to update your Twitter friends.*

**FAV username** marks the last message of username as a favorite
*Example: FAV petersample*

**STATS** returns the number of followers and how many users the account itself is following.
*Example: STATS*
*Answer: followers: 135 following: 47*

**INVITE phone number** sends an invitation SMS to the phone number
*Example: INVITE 123 456 7890*

# BIBLIOGRAPHY


[BHI+08] Garrett Brown, Travis Howe, Micheal Ihbe, Atul Prakash, and Kevin Borders. Social Networks and Context-Aware Spam. In *CSCW '08: Proceedings of the ACM 2008 conference on Computer supported cooperative work*, pages 403–412, New York, NY, USA, 2008. ACM.

[Fra08] Fraunhofer Institut für Sichere Informationstechnologie SIT. Privatsphärenschutz in Soziale-Netzwerke-Plattformen. Technical report, Fraunhofer Institut für Sichere Informationstechnologie SIT, Rheinstraße 75, 64295 Darmstadt, Germany, 2008.

[GAHI05] Ralph Gross, Alessandro Acquisti, and H. John Heinz III. Information Revelation and Privacy in Online Social Networks. In *WPES '05: Proceedings of the 2005 ACM workshop on Privacy in the electronic society*, pages 71–80, New York, NY, USA, 2005. ACM.

[JJJM07] Tom N. Jagatic, Nathaniel A. Johnson, Markus Jakobsson, and Filippo Menczer. Social Phishing. *Commun. ACM*, 50(10):94–100, 2007.

[Jon04] Chris Jones. Social Engineering: Understanding and Auditing. *SANS Institute*, 2004.

[JSFT07] Akshay Java, Xiaodan Song, Tim Finin, and Belle Tseng. Why We Twitter: Understanding Microblogging Usage and Communities. In *WebKDD/SNA-KDD '07: Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, pages 56–65, New York, NY, USA, 2007. ACM.

[KGA08] Balachander Krishnamurthy, Phillipa Gill, and Martin Arlitt. A Few Chirps About Twitter. In *WOSN '08: Proceedings of the first workshop on Online social networks*, pages 19–24, New York, NY, USA, 2008. ACM.

[Liv03] Charles E. Lively. Psychological Based Social Engineering. *SANS Institute*, 2003.

[Man00] Kurt Manske. An Introduction to Social Engineering. *Information Systems Security*, 9(5):1–7, 2000.

[Mic09] Microsoft Inc. What is Social Engineering? `http://www.microsoft.com/protect/computer/basics/social.mspx`, April 2009. Last accessed, July 2009.

[MS03] Kevin D. Mitnick and William L. Simon. *The Art of Deception: Controlling the Human Element of Security*. John Wiley & Sons, Inc., New York, NY, USA, 2003.

[ORBO04] Gregory L. Orgill, Gordon W. Romney, Michael G. Bailey, and Paul M. Orgill. The Urgency for Effective User Privacy-education to Counter Social Engineering Attacks on Secure Computer Systems. In *CITC5 '04: Proceedings of the 5th conference on Information technology education*, pages 177–181, New York, NY, USA, 2004. ACM.

[Ray96] Eric S. Raymond. *The New Hacker's Dictionary (3rd ed.)*. MIT Press, Cambridge, MA, USA, 1996.

[Tho04] Tim Thornburgh. Social Engineering: The "Dark Art". In *InfoSecCD '04: Proceedings of the 1st annual conference on Information security curriculum development*, pages 133–135, New York, NY, USA, 2004. ACM.

[WD95] Ira S. Winkler and Brian Dealy. Information Security Technology? ...Don't Rely on It. A Case Study in Social Engineering. In *SSYM'95: Proceedings of the 5th conference on USENIX UNIX Security Symposium*, Berkeley, CA, USA, 1995. USENIX Association.

[Whi09] Dan Whitworth. Twitter Growth Explodes in a Year. `http://news.bbc.co.uk/newsbeat/hi/technology/newsid_7948000/7948092.stm`, March 2009. Last accessed, July 2009.