RAMS

Reliability, Availability,
Maintainability, and Safety

This is the Title of my Thesis

Your Name

August 2014

PROJECT / MASTER THESIS

Department of Production and Quality Engineering

Norwegian University of Science and Technology

Supervisor 1: Professor Ask Burlefot

Supervisor 2: Professor Fingal Olsson

Preface

Here, you give a brief introduction to your work. What it is (e.g., a Master's thesis in RAMS at NTNU as part of the study program xxx and...), when it was carried out (e.g., during the autumn semester of 2021). If the project has been carried out for a company, you should mention this and also describe the cooperation with the company. You may also describe how the idea to the project was brought up.

You should also specify the assumed background of the readers of this report (who are you writing for).

Trondheim, 2012-12-16

(Your signature)

Ola Nordmann

Acknowledgment

I would like to thank the following persons for their great help during ...

If the project has been carried out in cooperation with an external partner (e.g., a company), you should acknowledge the contribution and give thanks to the involved persons.

You should also acknowledge the contributions made by your supervisor(s).

O.N.

(Your initials)

Summary and Conclusions

Here you give a summary of your your work and your results. This is like a management summary and should be written in a clear and easy language, without many difficult terms and without abbreviations. Everything you present here must be treated in more detail in the main report. You should not give any references to the report in the summary – just explain what you have done and what you have found out. The Summary and Conclusions should be no more than two pages.

You may assume that you have got three minutes to present to the Rector of NTNU what you have done and what you have found out as part of your thesis. (He is an intelligent person, but does not know much about your field of expertise.)

Contents

Preface i

Acknowledgment ii

Summary and Conclusions iii

1 Security measures against manipulation by bots **2**

1.1 Abstract 2

1.2 Background 2

1.3 Detection 2

1.4 Countermeasures 2

1.4.1 User 3

1.4.2 Institution 4

1.4.3 Service provider 4

1.4.4 Social network 5

1.4.5 Law enforcement and government 5

1.5 Conclusion 6

2 Social Media Attacks **8**

2.1 Hypothesis and methodology 9

2.2 Introduction 9

2.3 Attacks on social media users in general 10

2.4 Specific attack vectors for computers 11

2.4.1 Microsoft Windows 11

2.4.2 Apple OSX 11

2.4.3 Linux distros 11

2.5 Specific attack vectors for smartphones . . . 11
2.5.1 iOS . . . 12
2.6 Social media attacks directly on the given platform . . . 12
2.6.1 User behavioural aspects . . . 13
2.6.2 Hardware and software . . . 13
2.6.3 Social media network trust . . . 13
2.7 Conclusion . . . 14

3 Approaches for Detecting Robots in Social Media **16**

3.1 Introduction . . . 18
3.2 Definition and History of Social Bots . . . 19
3.3 Important Social Media Platforms . . . 20
3.4 Why is Bot Detection in Social Media Important? . . . 21
3.5 Social Bot Detection Approaches . . . 23
3.5.1 Based on Social Network Information . . . 23
3.5.2 Based on Crowd-Sourcing . . . 25
3.5.3 Based on Features . . . 27
3.6 Summary and Outlook . . . 29

4 Web Application vulnerabilities in 2016 **31**

4.1 Introduction . . . 32
4.2 Trending vulnerabilities and consequences . . . 34
4.2.1 Cross-Site Scripting (XSS) . . . 34
4.2.2 SQL Injection . . . 35
4.2.3 Vulnerable JavaScript Libraries . . . 36
4.2.4 Exploit Kits . . . 36
4.3 Countermeasures and mitigation . . . 37

Bibliography **38**

Chapter 1

Security measures against manipulation by bots

1.1 Abstract

In this chapter we will go through some of the countermeasures available against botnets. We will present approaches from different perspectives such as internet service providers, users, social network societies and others. We will where applicable include some examples of where these countermeasures have been used and to what effect. In the end we will go over a conclusion on what can be learned from this and how it may apply to you.

1.2 Background

1.3 Detection

1.4 Countermeasures

There are several different methods to counteract a botnet, depending on the perspective you take. From the perspective of a user some countermeasures are viable, while there are different measures available as a ISP or a webhost. In the following sections we will go through these different perspectives and describe the countermeasures available.

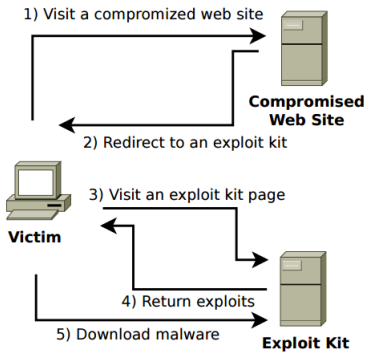

Figure 1.1: Exploit kit infection scheme from [2].

1.4.1 User

As a user it is important to be aware of the risks inherent with the use of certain services and systems, nowadays malicious code can be delivered through a multitude of vectors, from exploit kits which require little to no interaction from the user to phishing emails that try to fool the user into accessing a malicious site or running a attachment [**?**]. Paying attention to what the user is being asked to do and thinking twice if this actually makes sense can thwart many attempts at infecting the users system, many infection campaigns will pretend to be a legitimate request from a known company or service provider, as seen in a Torrentlocker campaign targeting norwegian users in 2015 [1], where the user would recieve a email claiming to be from the norwegian post-office and asking the user to check a package delivery notice.

Another infection vector is the exploitation of vulnerabilities in the software on the users system, either by attempting to get the user to manually execute a seemingly legitimate file that will exploit a vulnerability in the software used to open it or through exploit kits. Exploit kits are tools used to serve as a platform for delivering malicious software for customers to their victims, they exploit vulnerabilities in software such as browsers or browser plug-ins like flash or java [2]. The victims are usually directed to a exploit kit from a legitimate website that has been modified by an attacker to silently redirect the victims browser to a malicious web-server, see figure 1.1. By keeping software such as your browser and its plugins up to date or disabling them if not needed can mitigate or completely remove some of the risks associated with exploit kits and drive-by-downloads. Keeping software such as PDF-readers and other commonly used software updated will ofcourse help.

1.4.2 Institution

As a institution or business there are a few ways to reduce the risk associated with botnets or malware infection, depending on the policy regarding usage of computer resources and responsibility of these, measures may vary from business to business. Keeping software updated is helpfull as mentioned, but application whitelisting is not a new concept [3]. Instead of blacklisting known bad things the way a anti-virus solution would work we instead only allow known good applications to run, we see implementations of this in Apples iPhone where only applications approved by apple are allowed to run or in Windows AppLocker [4] for windows 7 and newer. By limiting the approved applications for a user or a group of users one can limit the attack surface they expose. Whitelisting network protocols is another way to limit the impact a botnet could have in network, by only allowing certain protocols to communicate a potentially infected computer may not be able to communicate with its command and control server. Similar to this is the act of network traffic inspection, rule based IDS or IPS solutions can look for known attributes in the network traffic and either block or alert when a match is found, in cases where the trafficpattern generated by the download of the malicious code of the bot is known, one can stop the infection before it happens.

1.4.3 Service provider

Some botnets use a client-server architecture where several clients, or bots, are controlled by 1 centralized server. To communicate with these servers today where IP-adresses can change often the botnets will use DNS to figure out which IP-adress to contact. Sinkholing is the act where you are deliberately sending the wrong IP-adress in response to certain queries and end up directing the traffic from the original command and control infrastructure of the botnet to a server controlled by someone else. This was used by [5] when they took over control of the Torpig botnet for several days in early 2009. A somewhat controversial case of sinkholing domains associated with malware traffic is the case from 2014 where Microsoft got permission to sinkhole traffic towards several domains belonging to Vitalwerks Internet Solutions, the company behind no-ip, a dynamic dns provider. This was later overturned when the sinkholing proved to affect both legitimate and illegitimate use of the domains [6].

In [7] they describe botnets using IRC-traffic, as a service provider this means that a botnet could be detected by inspecting the traffic from a network and look for known attributes belonging to a botnet, for instance IRC commands. As technology evolves, so does the botnets with them, [5] describe a botnet named Storm, that utilized p2p to communicate. Detecting and stopping traffic from new botnets can be achieved by using honeypots and generating signatures based on observed traffic, this is the objective of Honeycomb project [8].

1.4.4 Social network

As described in [9] the facebook immune system is a system that uses something they call the adversarial cycle(See figure 1.2), where the lifecycle of the immune system is laid out. Defending against a attack has a result that is detected by the attacker which then mutates his attack to counteract the defense mechanism. The goal of the defender is to lengthen the time it takes for the attacker to counter the defense mechanism, this increases the cost of the attacker and thus reduces how profitable attacking the social network is. In the facebook immune system they emphasize targeting features with high cost, due to them being hard to detect or change. This can mean targeting infrastructure such as the hosts IP adress which can force them to relocate or use proxy services or it can for instance mean implementing tasks that cant be easily automated or not automated at all. Captcha [10] is one way to increase the costs of automation, while there exists services that solve captchas for money they cut into already not so great profits [11] and increasing complexity of captchas when detecting suspected malicious automated behaviour could further increase cost for the attacker.

1.4.5 Law enforcement and government

The facilitators of botnets and other malware are in most countries breaking the law with their operation since they are using computer resources without approval of the system owner, it follows that it should be in the interest of law enforcement to stop this activity. Due to the international and borderless nature of the internet enforcing the law on a anonymous person behind a IP adress is not a simple task. International co-operation is necessary to effectively take down the infrastructure of a botnet or prosecute the criminals behind it. The Convention on Cyber-

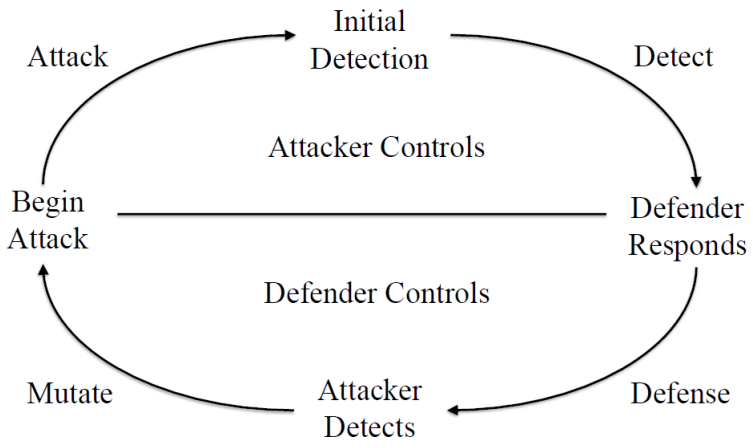

Figure 1.2: The adversarial cycle from [9].

crime in 2001 attempted to target some of the issues with co-operation between countries and promoting a common criminal policy in relation to cybercrime ([12]). The European Cybercrime Centre ([13]) is a part of Europol that focuses on cybercrime, by promoting cooperation between countries and between law enforcement agencies, private sector and academia. An example of the results of cooperation between countries in taking down a botnet is seen when the FBI with the cooperation of law enforcement in other countries shut down and arrested the people and infrastructure behind the GameOver Zeus botnet in 2014 [14, 15].

1.5 Conclusion

Depending on which perspective you look at problem from, your available options are different and by themselves they may not be available or be severely limited due to the specific circumstances surrounding your situation. Cooperation between the different groups may overcome this, it should be in the users interest that his place of work is not bothering or hindering his effectiveness in his daily tasks, just as it is in the internet service providers interest to limit the amount of illegitimate traffic going through its networks. Limiting the feasability of running a botnet can not simply be done by one user, but is a team effort that requires every part of the chain from the normal user on his computer to the business in charge of the network to the social network targeted by the botnet. Increasing the cost while reducing the benefit of running a

botnet will eventually make it economically unattractive to run a botnet for criminal organisations.

Chapter 2

Social Media attacks on mobile devices vs. attacks on computers

Management summary

We have seen an enormous increase in online social media in popularity during the past decade. It has taken over a lot of the networking people tended to do in real life earlier and it makes keeping in touch with a larger network of people a less laboursome task. We tend to trust a friend more than we would any random person on the street and this trust mechanism has been transferred to the cyber space. Cybercriminals, who seem to be mostly concerned with gaining wealth, political high ground, or are ideologically motivated, have shown a keen interest in utilizing these new networking platforms. They come from several different directions, such as attacking via vulnerabilities in hardware, OS, protocols, and apps, but they are also successfully exploiting the trust that exists between members of a social network. In this paper I am looking at the differences between the security topography for social media users, on a mobile device, versus that one would navigate when utilizing a personal computer. I have found quite a few differences and in other areas the situation is quite similar. The conclusion I reached was that it is, at least at present, much safer to use social media on a regular personal computer than on a mobile device, but that it also depends to a great degree how the user behaves security vice. The share of time people spend on social media networks on their mobile device versus on their computer is already 79% vs 21% [11] and still the mobile platform is rapidly gaining terrain. The

mobile platforms are getting better protection, but still they are behind on sophistication level when it comes to security. From what I have seen it looks as if you are an easier target for social media attacks when you are using a mobile device than you are doing the same social media networking on your computer.

2.1 Hypothesis and methodology

My hypothesis is that it is easier to attack social media users via their mobile platform gadgets, than via their use of the same media on computers. I have weighed different factors and gathered information around the subject by reading published material saying something about usage pattern, platform weaknesses, distribution of vulnerabilities, and how the users time is divided between the two platform categories. I have looked at how software and hardware protection is being utilized the two cases and I will in the end try to conclude based upon the information presented. The subject of social has been moving so quickly, behavioural statistics changing drastically from one year to the next. This constant change makes it difficult to obtain standard peer reviewed research with the latest numbers. For this reason, I have partially leaned on numbers obtained from white papers published by some of the big commercial players in the information security market. These papers are recognized by the industry as reliable sources for such information.

2.2 Introduction

Social media attacks are increasingly common. People's usage pattern of social media has also changed dramatically during the past few years. An increasing amount of people's social media interaction is being conducted on mobile hardware platforms. Instead of using Facebook, LinkedIn and other media on the Personal Computer they may or may not still have in their home, they are now using the same media on their mobile devices, such as pads and smartphones. There are also new social media appearing, especially made for these mobile platforms, such as Instagram, which is made to share photos taken with your smartphone on the go. People now spend more time interacting on social media using their smartphone, than they do using

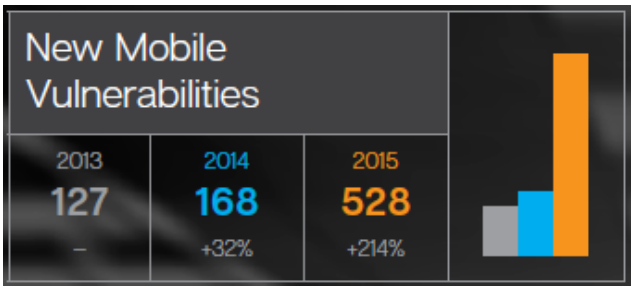

Figure 2.1: New mobile device vulnerabilities per year [10].

their computer. According to comScore 79% vs 21% [11]. Since this is a relatively new trend, that has exploded in the past couple of years, it is interesting to see if the migration from one platform to another has led to a more unsecure situation for the users. I limit the type of attack here to direct attacks on end users, with the intension to extract either labour, money or personal data. Nevertheless, I think that the result says something about the security difference we currently have between these two types of platforms.

2.3 Attacks on social media users in general

Some of the most popular social media providers today are Facebook, LinkedIn, Twitter, Google+, YouTube, Pinterest, Instagram, Vine, Snapchat and Reddit. There are myriads more and new services pop up all the time. While using portable devices to access social media, the user is both a possible target for the same types of attacks that we find being used against computer users and at the same time the limited resources of the mobile devices limits the possibility of using the same advanced mechanisms and software to stay safe [4]. Most attacks on the users of these media are economically motivated. There can be different paths to earning money on a social media attack. The reason for an attack can also be politically motivated or even nation state security concerns. I'll go through a few and compare possibilities on the different platforms. As the mobile devices gets ever more advanced and the number of apps, protocols and uses seem to rise towards the sky, it is no big surprise that the number of vulnerabilities follows suit as seen in Figure 2.1.

2.4 Specific attack vectors for computers

2.4.1 Microsoft Windows

Microsoft Windows has been the most widely used operating system and thereby represent the largest attackable population. It has also been regarded much easier to attack than the other common computer operating systems, because of the way it is built.

2.4.2 Apple OSX

Apples operating system was long regarded as the safest choice of operating system for a computer, but

2.4.3 Linux distros

2.5 Specific attack vectors for smartphones

Android

New types of attacks are coming all the time. Cross platform attack on Android is being performed via Google Play in a web browser on an ordinary computer. When the victim logs on to the Google Play account on a computer, to install apps on a mobile device running Android, malware on the computer can steal the browser cookies for that session and use it to impersonate the user. It is then possible for the attacker to remotely install apps on the victim's Android unit. We can also see that the malware is becoming smarter and the sophistication level is rapidly reaching the level of malware for computers. Example are that they are now both obfuscating the code so as not to be detected by malware protection software that uses signatures, and malware that can check if it is running on a real device or a security company's emulator, thereby avoiding detection. Another weakness with the Android platform is that even if Google pushes out security patches to the makers of all the different devices as fast as they can, many makers take a long time to push these out to their end users [10].

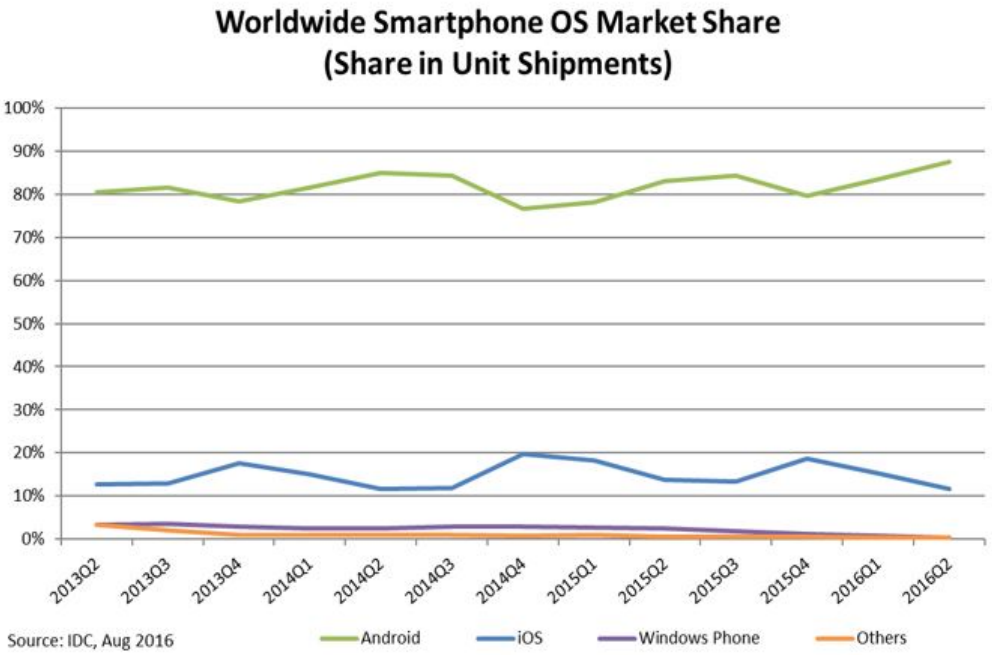

Figure 2.2: According to IDC (IDC, Aug 2016) the market share for the Android smartphone OS in August 2016 was 87.6% And for iOS 11.7%.

2.5.1 iOS

Apples screening of apps, before letting them into their App store, is very strict, so one can generally be more trusting towards apps found there. Attackers of Apple's portable devices, running the iOS operating system, for the most had to rely on finding vulnerabilities or attack a jailbroken device to succeed (unlike Android). This is no longer the case. Several new ways of attacking these devices has recently come to light. Symantec mentions two examples of such attack techniques namely XcodeGhost and YiSpectre which can compromise an iOS device without using vulnerabilities or jailbreaking. The OS distribution in the market means that attackers probably concentrate on Android to get the highest return for their effort. In fact, Kaspersky Lab and INTERPOL has made a report that states that an estimated 98.05

2.6 Social media attacks directly on the given platform

As we have seen there are many ways of attacking social media users/accounts via the platform they are using the media on. There are however a lot of ways to attack via the social media

itself, without depending on vulnerabilities in the hardware or software. The attacks simply rely on the many vulnerabilities found in the users, so to speak. These kind of attacks are usually categorized as

2.6.1 User behavioural aspects

2.6.2 Hardware and software

The way we use these mobile platforms will largely effect how safe we are from attacks. A lot depends on whether the user installs some kind of antivirus/antimalware program on their device and also how the user thinks about security, compare to when using computer. Many users leave the Wi-Fi on all the time. It is convenient to have your device connect automatically to all the networks you use, but this convenience comes at a high cost security wise. It is increasingly easy to attack these users by setting up a portable "Wi-Fi-box" that answers the unknowing smartphone owner's device's call for a named network, saying "Yes, here I am, please connect to me". Such devices can be bought readymade on the Internet; an example is the Wi-Fi Pineapple (https://www.wifipineapple.com/). The other radios in the devices, like Bluetooth and NFC, are also possible entry points for an attacker. We have seen attacks lately using NFC-code stickers containing malicious code or pointing the device user to a malicious website. These stickers can be bought for less than a dollar, programmed via a smartphone and printed with for example "Scan to get 100 free Instagram followers" or something to get people to scan it. An example of a possible attack could be to make the sticker point the user to a mock-up web page that looks like Instagram and since they already think they are there to get more followers they wouldn't react on having to enter their username and password.

2.6.3 Social media network trust

People build their social networks online, for most, as they do in real life. They either do know or at least feel they know the people in their social media network to a certain extent. This again leads to trust. People tend to trust the people in their social media network, just the way they trust their friends and co-workers' in real life. This trust makes the users easy prey to attackers making it look like links and other things has been sent to them by someone in their

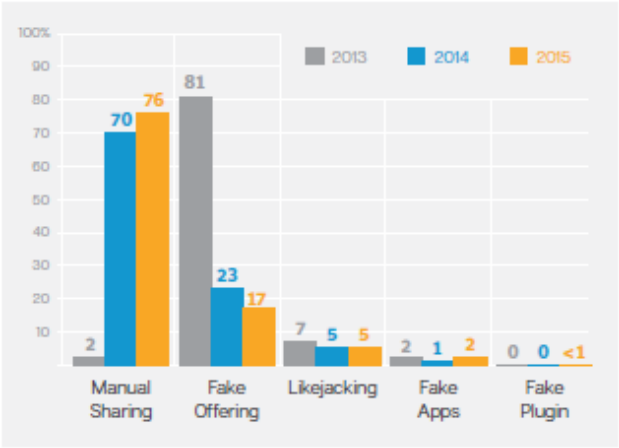

Figure 2.3: Social media network scams by popularity [10].

network. This can be both sent directly as a personal message, shared to all friends or simply liked. The most popular form of social media scam is manual sharing. Manual sharing consists of something that looks great in some way so that users themselves will want to share it with their friends, like for example a cool video or the chance of winning a new car if you share the post. The mechanisms of these scams works the same regardless of platform, since they are relying on the intended functionality of the app or web page. There should not be any difference in the way people

2.7 Conclusion

Based upon the material I have reviewed I would have to conclude that it is much easier to attack social media users via their portable devices, than it would be to attack their computer. The main considerations behind my conclusion are the following: The majority of time spent on social media is spent accessing these on a mobile device and not a computer. Very few people install any form of malware protection on their mobile devices, while a good majority installs malware protection on their computers. The malware protection solutions that exists for mobile devices, are inferior to the systems available to do the same task on a computer. The number of vulnerabilities found for mobile devices per year is rising quickly. People tend to walk around town with Wi-Fi, Bluetooth, and NFC turned on for anyone to exploit. So the advice to the general public should be to at least install some sort of malware protection on all their devices,

don't leave Wi-Fi, Bluetooth or NFC on when it is not in use.

Chapter 3

Approaches for Detecting Robots in Social Media

Management Summary

Social media platforms, like Twitter or Facebook, are a vital part of the Internet of today. Social bots are programs, that automatically take actions on those platforms. Among other things, they are used for a variety of attacks that, for example, aim to simulate a broad natural support for an entity or to cover unpleasant facts or opinions. Because of that, the detection of malicious social bots is an important task.

This chapter focuses on approaches that can be used in order to detect social bots. These approaches can be classified in social network based, crowd-sourcing based and feature based. Each of these classes is discussed in detail.

Social network based approaches model the social media platform into a graph that describes the relationships between all users. Based on those relationships they try to decide whether a user is a bot or a legitimate user. Crowd-sourcing based approaches leverage the social competence of actual humans to decide whether a given user is a bot or not. The decision is based on all visible actions of a user. Approaches that are feature based make use of machine learning, where a system is trained with known data. During this learning phase, it analyzes which features of user activity indicate a real user or a bot. Afterwards, it uses this knowledge to decide on real data.

It is found, that all approaches have their weaknesses and problems. In order to overcome these, it is advisable to combine different approaches. In addition, it is important to conduct further research in this topic, since the number of social bots is to be expected to continue growing. Also, the developers of social bots will develop more and more sophisticated methods to make their bots appear human and defenders should be prepared for this.

3.1 Introduction

Social media is an important part of today's Internet. Of the approximately 3.4 billion internet users, about 2.3 billion, that means almost 70 percent, use social media actively [16]. Especially in order to obtain information about events and the like, social media is often irreplaceable. For some people social media platforms even substitute real life personal contact to a large degree.

With this importance of social media platforms in the lives of so many people, they get more and more interesting for people that want to benefit economically from them. These are for example companies that want to promote their products and services via social channels. However, the problem here is that, due to their size, it is hard to disseminate a certain message on social media platforms. Like in many other areas of today's information technology, big data is a huge topic in social media.

This is one of the reasons, why social robots, in the form as we know them today, have been developed. As software robots are often simply called bots, we will use the term social bot or simply bot in the following synonymously for social robot. The propagation of such bots has reached a remarkable high level. In 2014, the social media platform Twitter stated in a report to the United States Securities and Exchange Commission that up to 8.5% of their active user base may be social bots[17]. We will see in the upcoming chapters, that they are used to spread certain messages, to hide others or simply to give the impression of a much bigger support for a person, thing or belief than there actually is. Social bots are also used to directly attack other, actual users, for example to steal their personal data.

First of all, we will take a look at the definition and history of social bots, in order to clearly define what we are talking about and where it originates from. We will then introduce the online social media platforms that are important for this chapter, namely Twitter, Facebook and Renren, and give a short overview about their characteristics relevant here. Afterwards, we will motivate why social bot detection is actually important. In the main part of this chapter, we will introduce several social bot detection approaches and give examples for them. In the last section we will finally summarize our findings and give an outlook about the further developments in this area.

3.2 Definition and History of Social Bots

This section will introduce the term social bot formally and give a short overview about the beginning and the development of this topic.

In order to be able to discuss social media bot detection, we need a clear understanding of what social bots actually are. For that, we use the definition given by Ferrara et al. in their article The Rise of Social Bots:

> "A social bot is a computer algorithm that automatically produces content and interacts with humans on social media, trying to emulate and possibly alter their behavior." [18]

A social bot is therefore basically any software that acts as a user, or something similar, on social media platforms. As the broad definition suggests, they can appear in many shapes.

The root of of social bots, can be found in the Turing test, developed by Alan Turing in 1950 [19]. It involves three parties, two of them are human and one is a computer program. While one human is having a conversation with the software, it is the task of the other human to identify the program by looking only at the conversation. If he is not able to do so, the software is passing the Turing test. This led to the development of a lot of so called chatbots, which just aimed to appear as human as possible in a conversation.

A rather famous and often cited example for such a chatbot is ELIZA, introduced by Joseph Weizenbaum in [20]. It mimicked a psychotherapist and showed that, at least some kind of, communication between a human and a computer is possible.

Since then, a lot of things have changed. Today, bots are a lot more than bare entertainment or proof of concept. With the triumph of the Internet and especially social networks like Facebook and Twitter, the possible use cases for social bots have increased dramatically. While they were initially mostly used to simply post content, today they are able to credibly interact with each other and even humans [21, 22]. As we will see in the next section, nowadays' bots are used to spread messages, for marketing and a lot more.

3.3 Important Social Media Platforms

In this section we want to give a short overview about the social media platforms, or online social networks (OSNs), that are relevant for this chapter. This is necessary in order to better understand the problems and attacks that will be presented later on. However, we will describe those networks only as detailed as necessary in this context, since a thorough explanation would go beyond the scope of this chapter.

The first and probably most important OSN that needs to be mentioned here is Twitter[1]. Twitter is a microblogging network which allows its users to broadcast short messages, so called tweets. Tweets are delivered to anyone that follows the tweeting account. Following is hereby a one-way process, that does not need any confirmation by the followed account. To see the tweets of a certain account, following is not necessary though. Thus, tweets can be regarded as public. Furthermore, tweets can be retweeted. Retweets can be seen as a forwarding of a tweet in order to disseminate it: all followers of the retweeting account get the retweeted message. Tweets can be tagged with hashtags, that associate them with a certain topic or an ongoing discussion.

Another OSN that is important for this chapter is Facebook[2]. As on Twitter, it is possible to publish messages on Facebook. However, the visibility of those messages, or posts, can be adjusted. Usually, posts are only visible to users that are friends with the posting account. A friendship is here, in contrast to follow-relationship on twitter, bilateral: a user can send a friendship request to another account which has to accept it in order for the relationship to be built. Another important difference to Twitter is, that Facebook accounts respectively profiles usually contain a lot more personal information than Twitter accounts do. Like it was the case with posts, the visibility of Facebook profile pages can be adjusted and usually they are only visible to accounts that are friends.

The last platform we want to mention shortly is Renren[3], which is a chinese OSN directed mainly at college students. Like Facebook it uses the concept of friendships and highly personalized profiles. It also allows to post messages that can be seen by friends.

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

3.4 Why is Bot Detection in Social Media Important?

Another point that is important to be looked at before we go into detail about the actual detection methods is, why social media bot detection is important at all. In this section we will argue, why this is an important topic, especially today.

As already said above, social media platforms are nowadays an important component in the life of many people. For some, they even substitute real personal contact sometimes. Thus, it is important to ensure, that humans are still able to estimate what is happening in their social media surrounding and what can be seen as trustworthy – and what cannot. While the challenge of verifying facts or news from the Internet is not new, social bots add a new factor to this challenge. They can make a non-trustworthy source look trustworthy by simulating that it is highly popular.

An example for this, is the so called astroturfing. In an astroturfing campaign, an attacker tries to give the impression of a broad grassroot support for a certain person, belief or political position. A known political astroturfing case are for example the 2010 Massachusetts senate elections in the US. During these, a small number of Twitter social bot accounts produced a large amount of tweets that contained a link to a website that smeared one of the candidates. The tweets also mentioned users that had shown the same political position beforehand and thus were likely to retweet. In a few hours the smearing website link spread rapidly what was even noticed by the Google search engine. That way, the website got promoted to the top search results for the name of the smeared candidate [23].

A similar approach can also be used to distract from potentially inconvenient or just unwanted facts or opinions is called smoke screening. By just flooding the platform with information, it aims to draw attention away from the unwanted topic to a topic that suits the attacker more. The flooding information can hereby even be related to the topic that is to be screened. It just focuses on a component that is more pleasant to the attacker and withholds the inconvenient part. An example for this is given by Abokhodair et al. [24] in the context of the syrian civial war. They analyzed a social botnet on Twitter from April until December 2012. It made heavy use of this technique in order to cover up news about the civil war in Syria.

Another, more direct, example for an attack scenario is described by Boshmaf et al. [21]

They operated a network of social bots on the social media platform Facebook and successfully tried to build as many relationships as possible with real users. After this was done, they were able to extract data from the profiles of those real users that were not publicly available, like for example mail addresses or phone numbers. While they show that operating such a botnet for the sole purpose of data extraction might not be efficient for an attacker, they argue that this data can be used for more advanced attacks afterwards.

Besides all these specific attacks, which are carried out by bots built for this purpose, there are also simple negative effects from social bots that have not necessarily been built for an attacking purpose. The information in social media is often used by various entities. An example for this is the area of emergency response. By analyzing the information streams of social media platforms it is possible to estimate emergency situations and to take proper actions for decision makers. Cassa et al. [25] for example show, that information about the Boston Marathon attacks in 2013 was more than five minutes earlier available on Twitter than the public health alerts, even though there were already many first responders present on this event. Another area where information from social media platforms is leveraged is the stock market. By monitoring the general mood on Twitter, for example, it is possible to predict the market trend to a certain degree [26]. A case can be made that this is also already done by traders. When in 2013 for example the Twitter account of the Associate Press was hacked and it posted rumors about a terrorist attack, the stock market crashed significantly [18]. It is therefore not hard to see that bot-caused information distortion on social media platforms – that does not even necessarily needs to be intended malicious – can have severe impact. Some social bots are built just to retweet and if they retweet false information or rumors they help to make this information popular. This happened for example also after the former mentioned Boston Marathon attacks, where false accusations were spread by such social bots [27].

Furthermore, social bots are often used by public persons in order to appear more popular and thus to gain influence or by companies in order to promote their products on social media platforms [28]. Here, social bot detection is necessary for ordinary users as well to be able to distinguish between real and bought support.

3.5 Social Bot Detection Approaches

In this chapter we want to introduce several techniques for detecting social bots. Based on Ferrara et al. [18] we distinguish three detection approach classes.

The first category of detection approaches is based on social network information. They are also called graph-based, since they map users and their relations into a graph and then try to identify bots in the hereby obtained social network by means of graph theory.

Afterwards we will discuss crowd-sourcing based social bot detection approaches. They use actual humans to detect bots, assuming that the human ability to notice details in communication will make this an easy task.

The last category we want to elaborate on are detection approaches based on behavioral features. Mechanisms that make use of this approach try to observe behavioral patterns that are typical for humans and social bots. By doing so they aim to distinguish bots from human users.

In the following sub sections, we will go into detail about each of these three approaches and illustrate them using real detection systems. It has to be noted though, that the borders between the individual approaches is not always very clear, so that some examples could also be mentioned in a different category. Many schemes for example, use some kind of graph theory. However, we try to categorize them by their core traits.

3.5.1 Based on Social Network Information

A term that is often used in combination with detection of bots by using social networks is sybil or the sybil attack. It was presented as a thread to distributed systems by John R. Douceur [29]. In the specific context of social media platforms, when conducting a sybil attack, an attacker creates a large amount of fake identities in a system to the point where these identities make up a considerable fraction of the system's whole user base. When this is achieved, the attacker can influence the whole system and control its contents to a certain degree. A sybil, sybil node or sybil account is therefore simply one of the fake entities, or depending on the attack's architecture, just a social bot. It is not hard to see that social bot detection can, more specifically, be viewed as a defense against the sybil attack.

The general proceeding of bot detection approaches based on social network is rather sim-

ple. They map the user base of the OSN they aim to defend into a social graph, where a node is corresponding to a user and an edge between two nodes exists if there is a specific kind of relationships between the two respective users on the platform. The nodes can hereby be distinguished in sybil nodes, respectively bots, and non-sybil nodes, respectively legitimate users. The goal of the detection approach is now, to identify whether a given node is a sybil or not [30].

There are a number of proposed social network based sybil detection schemes like for example SybilGuard [31], SybilInfer [32] or SumUp [33]. While they all have different assumptions and use varying algorithms to achieve their goal, Viswanath et al. [30] show that, at a high level, all of them work by them same principles.

Basically they can be viewed as graph partitioning algorithms, which partition a given graph into multiple disjoint subgraphs. As already mentioned above, ideally two subgraphs are assembled, one that contains only sybil nodes and one that contains only non-sybil nodes. Since a clear distinction is often hard to make, the approaches basically assign a rank to each node and decide afterwards, depending on several parameters, which ranks are classified as sybil and which as non-sybil. It is thus obvious, that the ranking algorithm is crucial for the whole scheme. Although, of course, the ranking algorithms for the different detection schemes are different, they generally have in common, that they base their rating on how tightly connected the respective node is to a known trusted node. Thus, they work by detecting local communities of nodes, in other words closely connected groups of nodes [30].

It is not hard to see, that these algorithms are therefore easy to deceive. If an attacker is able to establish so called attack edges, connections between his sybil nodes and non-sybil nodes which are connected to a trusted community, it gets significantly harder to identify the bots. A common assumption is that these attack edges are hard to create [31], which means, that legitimate users tend to not establish social network connections to social bots. However, Boshmaf et al. show [34] that this assumption is to be questioned. They tried to infiltrate Facebook with a large number of sybil accounts and to establish connections to real users. The average acceptance rate of their friendship requests came to about 20% and could be increased to 80% depending on how many indirect friendships between the sybil and the user already existed [34].

A well known example for a bot detection approach that is based on social network informa-

tion is the Facebook Immune System (IMS) [35]. This system aims to defend the social media platform Facebook and its users not only against sybil attacks but also to prevent spam, malware distribution, phishing and so on. To achieve this goal, it takes a lot more actions than the above described general approach for social network based detection schemes. The IMS runs checks on every action performed on Facebook in realtime in order to give attackers as less time as possible to accomplish their goals and to react. It classifies these actions according to predefined policies and makes it thereby possible to judge them and the corresponding users.

An example for this could be a newly created Facebook account, that sends a lot of friendship requests. A legitimate user starts sending friendship requests usually mainly to people he knows and vice versa, that is, people that are likely to accept his friendship request. If a lot of these requests are now declined, this could be a indication for the system that the sending user might not be legitimate. The IMS also makes heavy use of machine learning, to which we will come back in later on, and generates training data automatically in order to adapt to the fast changing attacks and user behavior.

3.5.2 Based on Crowd-Sourcing

A rather straight forward approach for social bot detection is based on crowd-sourcing. In contrast to the above described social network information based approaches, a connection between bots and legitimate users is not a problem for schemes based on this approach – at least not a direct one. The basic idea of crowd-sourcing based social bot detection is, to engage actual human workers to study user profiles and subsequently to decide whether it belongs to an actual user or a sybil.

Wang et al. [36] present a study on the effectiveness of this approach and introduce a sample for a crowd-sourcing based social bot detection system. For their tests they use data from Renren, China's most popular social media platform, Facebook US and Facebook India. They subdivide their human investigators, or simply testers, in expert workers, and crowd-sourced workers. Each of the testers is presented a number of social media profiles and has to decide, whether it is a real one, or a sybil. While the experts achieve a detection rate at about 90%, the crowd-sourced workers perform not as well individually. If their single votes are aggregated and used for a majority decision though, the results can be significantly increased. If this is done for

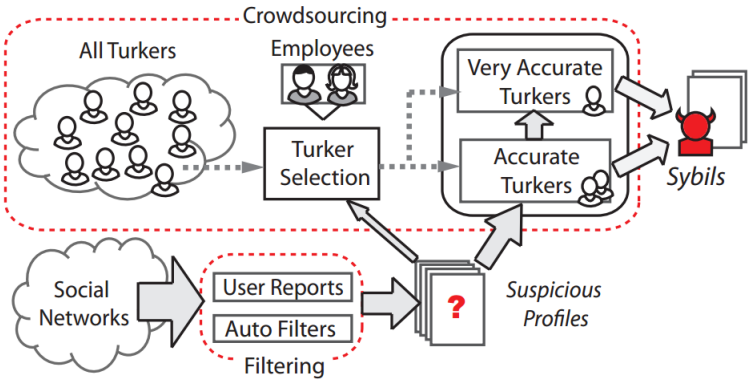

Figure 3.1: A crowd-sourced sybil detection system introduced by Wang et al. [36].

the expert workers, their results can be increased even more, too. A very desirable result of their studies is also that the false positive rate, that is the amount of profiles that are falsely classified as sybils, is in all groups very close to zero. This ensures, that the probability that legitimate users are accused to be social bots, what will probably offend them, is very low.

Wang et al. conclude, that it is very hard for sybil creators, to assemble social bots, respectively profiles, that are able to pass a "social turing test" and that crowd-sourcing based approaches can perform very well. They proceed with introducing a general practical system that is illustrated in figure 3.1 [4].

The system works like this: The social network that is to be defended generates suspicious profiles. This can either happen through explicit reports from users or through automatically applied filters that detect abnormal behavior. The so obtained profiles are then checked by crowd-sourced workers, which are subdivided in accurate and very accurate. To establish this differentiation, some of the suspicious profiles are mixed with some profiles that are known to be sybils. They are presented to all workers and their results allow a classification and to filter out unreliable workers. In the actual checking process, the profiles are first presented to the accurate workers which make a majority vote. If their decision is not clear, the profiles are presented to the very accurate workers which once again make a majority vote. The authors claim, that a system like this is very reliable and cost effective.

Although such a social bot detection approach based on crowd-sourcing seems very promis-

[4] Turkers are in this case the former mentioned crowd-sourced workers.

ing at first sight, several problems become obvious on closer examination. First of all, this approach works optimal if every newly created profile can be reviewed in the above described way. However, young social media platforms will probably not be able or willing to pay for this bot detection scheme. A factor that may influence this could also be, that those platforms probably do not have huge issues with the social bot problem in their early stages. Once the platform has grown and the issue arises though, their is usually already a huge user base that has to be reviewed retrospectively. This is not optimal and may take a considerable amount of time.

Another point that must be taken into consideration is that this approach might not be suitable for all types of social media platforms. While it might be well suited for platforms like Facebook or Renren, where profile pages can be customized a lot and contain usually plenty of information, other networks, like for example Twitter could be less appropriate for this approach. Generally it can be said that this detection approach is highly based on the information given in the users' profiles. If the information contained in the profiles is low, human investigators will probably not be able to distinguish as precisely as in the tests of Wang et al.

The last and probably most difficult issue that has to be mentioned is connected to this. Human investigators that are charged with distinguishing between bots and legitimate users need to have access to the profiles of the users. While those profiles are sometimes publicly available anyways, detailed profiles, which are more interesting for this approach, are, as discussed above, usually only visible for certain users. It is not hard to see the privacy issue that arises here, especially when keeping in mind, that the investigators are – at least to a great extent – only crowd-sourced workers, that cannot be supervised as easily as ordinary employees [18].

3.5.3 Based on Features

The last detection approach for social bots is based on features. Features hereby mean observable characteristics of posted information, profiles and possibly anything else that can be associated to the account in question. Schemes that make use of this approach often use machine learning algorithms. These are algorithms that are able to make decisions based on learned patterns. They basically consist of two phases. In the first phase, the so called training phase, the algorithm processes training data which is labeled with the decision that should be made on this piece of data. By processing a lot of training data, the algorithm learns the features that are

indicators for a certain decision. In the second phase, real data is processed and the algorithm is able to make decisions based on the learned data. Thus, schemes that make use of this detection approach can be compared to anomaly based intrusion detection systems. In the social bot detection topic, the decision being made is usually whether a specific account is controlled by a bot or a human. Since it is not easy to make a clear decision in many cases, there are often multiple decision options that each express a degree of certainty.

A well known example for this kind of social bot detection is the Bot or Not? system, presented by Davis et al. [37]. It is a publicly available[5] system that allows a feature based social bot detection for accounts on the social media platform Twitter. Bot or Not? was trained with about 31000 account samples of both, human and social bot accounts. For classification, it makes use of more than 1000 features that can be assigned to six feature-classes. In the following we want to mention these classes shortly:

- **Network** features reflect on the spreading of information, for example citations of different users or the like.
- **User** features take the information given in the actual account into consideration. Metadata, that is basically data about data, is especially relevant in this feature.
- **Friends** features are about the social relationships of the account in question.
- **Temporal** features analyze issues like the rate in which content is produced and the like.
- **Content** features are about wording characteristics in the texts produced by the relevant account.
- **Sentiment** features reflect on emotional aspects and are obtained by using specific algorithms [37].

If a Twitter account name is entered in the Bot or Not? system, it checks all of those features and afterwards decides, based on the former mentioned training data, whether this account is a human or bot account.

[5] http://truthy.indiana.edu/botornot

Another social bot detection approach that can be classified as, feature based relies on detecting the coordinated behavior between social bots. Schemes that make use of this approach focus generally on the above described network, content and temporal features and often incorporate some graph theory as well. They usually aim at detecting attacks which try to distract from unpleasant facts or to distribute fake facts, like in the above described astroturf attack. An example for such a scheme is SynchroTrap, presented by Cao et al. [38]. Again, comparable to anomaly based intrusion detection systems, SynchroTrap makes use of clustering algorithms. These algorithms try to group values, that are similar in some aspects into distinct sets.

The system monitors all user activity on a social media platform over an extended time period and monitors aggregated user actions. Then, the pairwise similarity between actions is determined and a hierarchical clustering algorithm is used to group users that show similar behavior at approximately the same time. If a cluster is growing too big, it can be assumed as malicious, since legitimate users tend to behave diverse [38].

An issue of this detection method is though, that it is for example not able to detect bot accounts that show a mixture of bot and human behavior. Thus, the detection systems have to be developed further in order to keep up with the fast evolving social bots and their evermore sophisticated behavior [18].

3.6 Summary and Outlook

In this chapter, we firstly introduced the term social bot, defined it and looked at the history of robots in social media. We also gave a really short overview about the OSNs that are most important for this chapter and motivated why social bot detection is important after all. Mainly, we discussed social bot detection approaches. We distinguished hereby social network information based, crowd-sourcing based and feature based approaches.

Social network based approaches model the analyzed OSN into a graph where users correspond to the vertices and relationships between the users correspond to the edges of the graph. Afterwards, an algorithm tries to make a decision whether a given node is a social bot or an actual user. A problem these systems have to cope with is, that they tend to assume, that bots can't establish relationships to actual, what emerged to be false.

The basic idea of crowd-sourcing based approaches is, to engage actual humans to analyze given user profiles and to decide whether they belong to an actual user or a social bot. We discussed a paper which analyzes this approach thoroughly and introduces a general practical system. One of the issues this approach has to handle is that some social networks users don't have very detailed profiles what makes crowd-sourced bot detection very hard.

Feature based social bot detection approaches observe characteristics of OSN users and information published by them in order to decide – or give a tendency – whether a given account is a bot or not. Often, they make use of machine learning algorithms.

After analyzing the different detection approaches, it becomes apparent that each has its own problems and a combination of different approaches may yield good results. Observing the social graph and behavioral features of accounts together is for example a promising strategy. Ferrara et al. [18] bring in the Renren sybil detector [39, 40] as a good example for these combined approaches.

However, it has to be noted that not all robots in social media have to be evil necessarily. Some social bots provide useful services that many users do not want to miss. It is just important, that users are able to distinguish between other humans and bots for example in order to differentiate whether content they see is actually as popular as it seems.

It is safe to assume that the number of social bots will increase even more and that their methods and behaviors will get even more sophisticated. We expect an arms race between bot herders and bot detection mechanisms, like it was – and still is – the case with malware and malware detection software.

Chapter 4

Common vulnerabilities and attack surfaces for Web Applications in 2016

Management summary

The web stack has evolved to serve feature rich and dynamic experiences within the browser by expanding on the legacy applications. The majority of cyber attacks are done at web application level, and most of the vulnerabilities are well known and documented - but alot of web applications fail to mitigate or counteract them.

In this chapter several reports from security companies and non-profit organizations was analyzed to find some of the most common web application vulnerabilities with the possibility of severe consequences. The chapter focuses on Cross-Site Scripting, SQL Injection and vulnerable JavaScript Libraries to explain how they work and how they can harm the companies and their visitors/clients. It briefly looks at the rising usage of Exploit Kits, and how these vulnerabilities can compromize the web application and make it a part of an Exploit Kit infrastructure. It continues to look at possible mitigation techniqes and countermeasurements and discuss why so many sites are vulnerable to these well-known attack surfaces.

4.1 Introduction

Over the last years web application hacking has increased, and as many as 75% of cyber attacks are done at web applications level or via the web [41]. The introduction in the Acunetix report states that 55% of web applications has atleast one high-severity vulnerability - and thats up 9% since 2015.

The web stack has evolved to serve feature rich and dynamic experiences within the browser by expanding on the legacy applications. This has widened the attackers oppertunities, and the amount of vulnerable applications on the web is ever increasing [42]. In recent years an alarming number of high profile web service providers has lost huge amounts of personal data, including emails and weakly hashed passwords.
This chapter aims to find the most common vulnerabilities and attack surfaces for web applications in 2016, what the consequences can be, what measures can be taken, and hopefully a hint to why so many web applications are still vulnerable.

This chapter will focus on the application code vulnerabilities that might arise from poor programming and vulnerable third-party libraries in the application itself. A combination of the results of four induvidual information security reports will be examined to find a few vulnerabilities that occur often and are of high severity. Three of the reports are from 2015/2016, and the fourth is OWASP Top 10 from 2013.

OWASP Top 10

Open Web Application Security Project is a international charitable non-profit organization, and is a colaberation between a varity of security experts around the world [43]. The primary aim of the OWASP Top 10 is to raise awareness for web application security, and so far the report has been released twice, in 2010 and 2013 [44].

Acunetix Web Application Vulnerability Report 2016

Acunetix is a company spacializing in automated tools to scan servers for vulnerabilities, and have many high profile private and public companies as clients [41]. They released an annual

report on statistics from their scans throughout the period of 1st April 2015 to 31st March 2016. Acunetix has gathered, aggretated and analyzed data from over 61.000 scans over a two year period, and are in a great position to observe trends in the field [42].

Hewlett Packard Enterprise Security Research Cyber Risk Report 2016

Hewlett Packard provide a broad view of the 2015 threat landscape, based on industry-wide data and a focused look at open source, mobile and IoT [45].

Symantec Internet Security Threat Report 2016

Through the Global Intelligence Network, Symantec has one of the most comprehensive sources of internet threat data [46]. It's made up of more than 63.8 million attack sensors in over 157 countries and territories through a combination of Symantec products and services [46].

4.2 Trending vulnerabilities and consequences

This section will introduce the three common high-severity vulnerabilities that this chapter will focus on. By examinin the reports listed in the Introduction, the choice has fallen on Cross-Site Scripting (XSS), SQL Injections and Vulnerable JavaScript libraries. All of the threats described here have consistently been reported as the most common vulnerabilities in web applications for several years, and 2016 is no different [47][48][42]. They are all vulnerabilities that arise in the development phase of a web application, and they can all potentially cause significant damage.

4.2.1 Cross-Site Scripting (XSS)

If an attacker is able to submit malicious HTML such as JavaScript to a dynamic web application, they are able to execute a Cross-Site Scripting, or XSS, attack [49].
Unlike traditional distributed systems security where access control equals authentication and authorisation, a web application uses the same-origin policy [50]. This policy is exploited by a Cross-Site Scripting attack, when the vulnerable site is viewed by a victim the malicious content seems to come from the trused site, and the attacker can steal cookies, session identefiers and other sensetive information that the web site has access to [49]. Cross-Site Scripting can be viewed as two categories, Persistent and Non-persistent attacks. In a persistent attack the attacker is able to store some malicous code in the database or a file that are rendered as part of the website for future visitors [51]. In a non-persistent attack the attacker uses parameters in the url to inject malicious code to the server, potentally deleting or disclosing sensitive information [51].

Consider the following scenario: A user connects to *trusted.com* where they are a registered and authenticated user. In *trusted.com/forum* a malicious user has posted a seemingly innocent post, but managed to append some unsanatized JavaScript code at the end of the post. The JavaScript snippet is not visually rendered by the browser, but executed in the background. The JavaScript snippet then sends the cookie of the victim to *malicious.com/save?cookie=* and append the cookie for storage.

A twitter tool called TweetDeck had a serious Cross Site Scripting vulnerability in 2014 that

made it possible to tweet unsanitized input [52]. This was a self retweeting tweet, with a small red heart as the only visible content. Lets look at the code as it was injected:

```
<script class="xss">$(‘.xss‘).parents().eq(1).find(‘a‘).eq(1).click()
;$(‘[data-action=retweet]‘).click();alert(‘XXS in TweetDeck‘)</script>
```

If we examine the code in this attack, it starts out by opening a *script* tag and giving it a css class called *xxs* to be able to use it as an reference. The $ refers to jQuery, a popular JavaScript Library, and uses the class previously created to locate the parent containers. Then it picks the second parent out of the list of parents, which in this case is the tweet box. It then continues to find all the link tags and again select the second one, which in this case is the retweet button - and clicks it. This in turn opens a popup to confirm the retweet and the next jQuery tag clicks on the comfirm tweet button, or *[data-action=retweet]*. It finally promts the user with a JavaScript alert saying *"XXS in TweetDeck"* and closes off the script tag.

Now this is a fairly innocent attack, and you could argue that this attacker did this to either show of their skills or alert the twitter developers of their error - or even both. There is no reason to alert the user if you want this to go unnoticed as long as possible, so it's safe to assume that this vulnerability could have caused alot more damage if the attacker wanted to.

Both of these previous examples are so-called persistent attacks. The malicious code is stored on the server to render for all visitors. An example of a non-persistent attack could be a page taking page number as a parameter in the url, where the attacker can send malicious code as a parameter - and then somehow get the victim to visit that url while authenticated [51].

4.2.2 SQL Injection

Structured Query Language (SQL) is the language enabeling applications to talk to most relational databases. In a web application scenario SQL is used for creating, reading, updating, and deleting, in addition to searching and filtration. It is often connected to forms and other dynamic user input functionality, and this is where most SQL-Injection attacks come into play. If an attacker is able to inject code that the server treats as SQL code, the attacker can gain access to the underlying database [53]. Databases often contain sensitive consumer or user informa-

tion, and result in violations such as identity theft, loss of confidensial information, fraud and even system control.
This could possibly work on any website or web application that uses a SQL-based database, and is therefore one of the oldest, most prevalent and dangerous vulnerabilities [42].

4.2.3 Vulnerable JavaScript Libraries

JavaScript has become the number one client-side language for user-friendly content display, and does often communicate with backend languages through Asynchronous Javascript And Xml (AJAX). A library in programming terms is a pre-written set of code to ease the development of applications, and is used by most dynamic websites today. One of the challenges that arises from this approach is that when a vulnerability is exposed by an attacker, a large number of sites suffer from the same vulnerability. 27% of the sampled targets in Acunetix Web Application Vulnerability Report 2016 used vulnerable JavaScript libraries within their application [42].

4.2.4 Exploit Kits

Vulnerabilities compromising a site could lead to the site being a part of an Exploit Kit infrastructure. Exploit Kits are server applications made to deliver malware istead of web content [54]. They are not a web application vulnerability in themselves, but a tool used for drive-by-download attacks through a multitude of vulnerabilities in browsers and browser addons packaged together in a kit [55]. The reason exploit kits get a mention here in this report is that they can be a result of the three vulnerabilities mentioned above, for example an <iframe> in a XSS or even a SQLInjection or a JavaScript based redirect [54]. The exploit kits inject shellcode to the vicim process to download a independent malware executable to the victim's hard drive and executes it [54]. These kits has become a competative market and range from several hundreds to over a thousands dollar on the black market [54], and therefore also have sophisticated code and sometimes even zero-day exploits.

4.3 Countermeasures and mitigation

Losing sensetive information, about your users or your own systems, is a serious skratch in the reputation of your online enterprise. Maybe even worse is beeing part of an Exploit Kit infrastructure, putting all your visitors in harms way for drive-by downloading of malware.

Even though both SQL Injection and XSS attacks almost as old as the utilazation of databases and javascript in web applications, it still is a major source of vulnerabilities today.

The basic forms of attacks on both SQL Injection and XSS are easy to pervent, simply by constraining and sanitizing the data. Constrain the length, type, range and possibly format of any input data, and then sanitize the remaining input by escaping characters that could be interpered as code [53]. For a query parameter that set the page number in a search funcion, this data should be a integer in the range between 1 and the total number of pages. In a input box in a form, for instance a email, this data should only be accepted if it is a combination of letters and numbers on the format xxx@xxx.xx. Characters like > and < should never be stored as is, but rather changed to their html equvalent or some other form of rewrite **Find some sources**. To mitigate the damage from a SQL Injection attack, it's important to do ragularly backups of the database and store passwords using strong and computationally heavy hashing functions with salting.

This raises the question: why are so many web applications still vulnerable to these types of attacks today? My thoughs on the subject, that in no way is scientificly validated, is that there is two major reasons for this: 1. The web application scene is filled with developers that not nessesarly have a formal education in programming or web development, thus lack the basics like security. 2. An increase in frameworks and content management systems usage silently teaches the developers that the security is taken care of in the core of the framework or CMS, so that when they do create a plugin or special feature they forget to handle these risks themselves.

The JavaScript Libraries on the other hand is harder to pervent, the easiest solution here is to stay on top of all libraries the site is using. Update or remove the library as soon as a bug or vulnerability is found. Another thing that is worth mentioning is that JavaScript libraries could be used in XSS attacks aswell, as seen with jQuery in the Twitter example above.

Bibliography

[1] NSM. Ny bølge! Falske Posten Norge e-poster i omløp; 2015. Https://nsm.stat.no/om-nsm/tjenester/handtering/norcertvarsler/norcert_varsler2015/postenrw1/. Available from: https://nsm.stat.no/om-nsm/tjenester/handtering/norcertvarsler/norcert_varsler2015/postenrw1/.

[2] Kotov V, Massacci F. In: Jürjens B Janand Livshits, Scandariato R, editors. Anatomy of Exploit Kits. Berlin, Heidelberg: Springer Berlin Heidelberg; 2013. p. 181–196. Available from: http://dx.doi.org/10.1007/978-3-642-36563-8_13.

[3] Mansfield-Devine S. The promise of whitelisting. Network Security. 2009;2009(7):4 – 6. Available from: http://www.sciencedirect.com/science/article/pii/S1353485809700856.

[4] Microsoft. Windows AppLocker;. Https://technet.microsoft.com/en-us/library/dd759117(v=ws.11).aspx. Available from: https://technet.microsoft.com/en-us/library/dd759117(v=ws.11).aspx.

[5] Stone-Gross B, Cova M, Cavallaro L, Gilbert B, Szydlowski M, Kemmerer R, et al. Your Botnet is My Botnet: Analysis of a Botnet Takeover. In: Proceedings of the 16th ACM Conference on Computer and Communications Security. CCS '09. New York, NY, USA: ACM; 2009. p. 635–647. Available from: http://doi.acm.org/10.1145/1653662.1653738.

[6] Microsoft-Corporate-Blogs. Microsoft takes on global cybercrime epidemic in tenth malware disruption; 2014. Http://blogs.microsoft.com/blog/2014/06/30/microsoft-takes-on-global-cybercrime-epidemic-in-tenth-malware-disruption/#sm.00000dcwsbrzn9crgt7lpqod3cjdj.

Available from: http://blogs.microsoft.com/blog/2014/06/30/microsoft-takes-on-global-cybercrime-epidemic-in-tenth-malware-disruption/#sm.00000dcwsbrzn9crgt7lpqod3cjdj.

[7] Cooke E, Jahanian F, McPherson D. The Zombie Roundup: Understanding, Detecting, and Disrupting Botnets. Usenix; 2005. Available from: https://www.usenix.org/legacy/event/sruti05/tech/full_papers/cooke/cooke.pdf.

[8] Kreibich C, Crowcroft J. Honeycomb: Creating Intrusion Detection Signatures Using Honeypots. SIGCOMM Comput Commun Rev. 2004 Jan;34(1):51–56. Available from: http://doi.acm.org/10.1145/972374.972384.

[9] Stein T, Chen E, Mangla K. Facebook Immune System. In: Proceedings of the 4th Workshop on Social Network Systems. SNS '11. New York, NY, USA: ACM; 2011. p. 8:1–8:8. Available from: http://doi.acm.org/10.1145/1989656.1989664.

[10] Von Ahn L, Blum M, Hopper NJ, Langford J. Captcha: using hard AI problems for security. EUROCRYPT. 2003;p. 294–311.

[11] Boshmaf Y, Muslukhov I, Beznosov K, Ripeanu M. Design and analysis of a social botnet. Computer Networks. 2013;57(2):556 – 578. Botnet Activity: Analysis, Detection and Shutdown. Available from: http://www.sciencedirect.com/science/article/pii/S1389128612002150.

[12] CoC. Convention on Cybercrime;. Https://www.coe.int/en/web/conventions/full-list/-/conventions/treaty/185. Available from: https://www.coe.int/en/web/conventions/full-list/-/conventions/treaty/185.

[13] Europol. European Cybercrime Centre(EC3);. Https://www.europol.europa.eu/ec3. Available from: https://www.europol.europa.eu/ec3.

[14] FBI. GameOver Zeus Botnet Disrupted; 2014. Https://www.fbi.gov/news/stories/gameover-zeus-botnet-disrupted. Available from: https://www.fbi.gov/news/stories/gameover-zeus-botnet-disrupted.

[15] FBI. Department of Justice Provides Update on GameOver Zeus and Cryptolocker Disruption; 2014. Https://www.fbi.gov/news/pressrel/press-releases/department-of-justice-provides-update-on-gameover-zeus-and-cryptolocker-disruption. Available from: https://www.fbi.gov/news/pressrel/press-releases/department-of-justice-provides-update-on-gameover-zeus-and-cryptolocker-disruption.

[16] Insights S. Global social media research summary 2016; 2016. http://www.smartinsights.com/social-media-marketing/social-media-strategy/new-global-social-media-research/.

[17] gov S. United States Securities and Exchange Commission - Form 10-Q - Twitter, Inc.; 2016. https://www.sec.gov/Archives/edgar/data/1418091/000156459014003474/twtr-10q_20140630.htm.

[18] Ferrara E, Varol O, Davis C, Menczer F, Flammini A. The rise of social bots. arXiv preprint arXiv:14075225v3. 2015;.

[19] Turing AM. Computing machinery and intelligence. Mind. 1950;59(236):433–460.

[20] Weizenbaum J. ELIZA—a computer program for the study of natural language communication between man and machine. Communications of the ACM. 1966;9(1):36–45.

[21] Boshmaf Y, Muslukhov I, Beznosov K, Ripeanu M. Design and analysis of a social botnet. Computer Networks. 2013;57(2):556–578.

[22] Hwang T, Pearce I, Nanis M. Socialbots: Voices from the fronts. interactions. 2012;19(2):38–45.

[23] Mustafaraj E, Metaxas PT. From obscurity to prominence in minutes: Political speech and real-time search; 2010.

[24] Abokhodair N, Yoo D, McDonald DW. Dissecting a social botnet: Growth, content and influence in Twitter. In: Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing. ACM; 2015. p. 839–851.

[25] Cassa CA, Chunara R, Mandl K, Brownstein JS. Twitter as a sentinel in emergency situations: lessons from the Boston marathon explosions. PLOS Currents Disasters. 2013;.

[26] Bollen J, Mao H, Zeng X. Twitter mood predicts the stock market. Journal of Computational Science. 2011;2(1):1–8.

[27] Gupta A, Lamba H, Kumaraguru P. $1.00 per rt # bostonmarathon # prayforboston: Analyzing fake content on twitter. In: eCrime Researchers Summit (eCRS), 2013. IEEE; 2013. p. 1–12.

[28] Stringhini G, Wang G, Egele M, Kruegel C, Vigna G, Zheng H, et al. Follow the Green: Growth and Dynamics in Twitter Follower Markets. In: Proceedings of the 2013 Conference on Internet Measurement Conference. IMC '13. New York, NY, USA: ACM; 2013. p. 163–176. Available from: http://doi.acm.org/10.1145/2504730.2504731.

[29] Douceur JR. The sybil attack. In: International Workshop on Peer-to-Peer Systems. Springer; 2002. p. 251–260.

[30] Viswanath B, Post A, Gummadi KP, Mislove A. An analysis of social network-based sybil defenses. ACM SIGCOMM Computer Communication Review. 2010;40(4):363–374.

[31] Yu H, Kaminsky M, Gibbons PB, Flaxman A. Sybilguard: defending against sybil attacks via social networks. In: ACM SIGCOMM Computer Communication Review. vol. 36. ACM; 2006. p. 267–278.

[32] Danezis G, Mittal P. SybilInfer: Detecting Sybil Nodes using Social Networks. In: NDSS. San Diego, CA; 2009. .

[33] Tran DN, Min B, Li J, Subramanian L. Sybil-Resilient Online Content Voting. In: NSDI. vol. 9; 2009. p. 15–28.

[34] Boshmaf Y, Muslukhov I, Beznosov K, Ripeanu M. The socialbot network: when bots socialize for fame and money. In: Proceedings of the 27th Annual Computer Security Applications Conference. ACM; 2011. p. 93–102.

[35] Stein T, Chen E, Mangla K. Facebook immune system. In: Proceedings of the 4th Workshop on Social Network Systems. ACM; 2011. p. 8.

[36] Wang G, Mohanlal M, Wilson C, Wang X, Metzger M, Zheng H, et al. Social turing tests: Crowdsourcing sybil detection. arXiv preprint arXiv:12053856. 2012;.

[37] Davis CA, Varol O, Ferrara E, Flammini A, Menczer F. Botornot: A system to evaluate social bots. In: Proceedings of the 25th International Conference Companion on World Wide Web. International World Wide Web Conferences Steering Committee; 2016. p. 273–274.

[38] Cao Q, Yang X, Yu J, Palow C. Uncovering large groups of active malicious accounts in online social networks. In: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security. ACM; 2014. p. 477–488.

[39] Wang G, Konolige T, Wilson C, Wang X, Zheng H, Zhao BY. You are how you click: Clickstream analysis for sybil detection. In: Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13); 2013. p. 241–256.

[40] Yang Z, Wilson C, Wang X, Gao T, Zhao BY, Dai Y. Uncovering social network sybils in the wild. ACM Transactions on Knowledge Discovery from Data (TKDD). 2014;8(1):2.

[41] Acunetix. Combating the Web Security Threat; 2016. Available from: http://www.acunetix.com/company/.

[42] Acunetix. Acunetix Web Application Vulnerability Report 2016; 2016. https://d3eaqdewfg2crq.cloudfront.net/resources/acunetix-web-application-vulnerability-report-2016.pdf.

[43] OWASP. About; 2016. Available from: https://www.owasp.org/index.php/About_OWASP.

[44] OWASP. Top Ten Project; 2016. Available from: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project.

[45] Enterprise HP. Security Research Cyber Risk Report 2016; 2016. https://ssl.www8.hp.com/us/en/ssl/leadgen/secure_document.html?objid=4AA6-3786ENW.

[46] Symantec. Internet Security Threat Report 2016; 2016. https://resource.elq.symantec.com/LP=2899?inid=symc_threat-report_istr_to_leadgen_form_LP-2899_ISTR21-report-main.

[47] Project OWAS. OWASP Top 10 - 2010; 2010. https://www.owasp.org/index.php/Top10#OWASP_Top_10_for_2010.

[48] Project OWAS. OWASP Top 10 - 2013; 2013. https://www.owasp.org/index.php/Top10#OWASP_Top_10_for_2013.

[49] Kirda E. In: van Tilborg HCA, Jajodia S, editors. Cross Site Scripting Attacks. Boston, MA: Springer US; 2011. p. 275–277. Available from: http://dx.doi.org/10.1007/978-1-4419-5906-5_651.

[50] Gollmann D. In: Christianson B, Malcolm JA, Matyas V, Roe M, editors. Problems with Same Origin Policy. Berlin, Heidelberg: Springer Berlin Heidelberg; 2011. p. 84–85. Available from: http://dx.doi.org/10.1007/978-3-642-22137-8_11.

[51] Edmunds B. In: Safe Defaults, Cross-Site Scripting, and Other Popular Hacks. Berkeley, CA: Apress; 2016. p. 41–47. Available from: http://dx.doi.org/10.1007/978-1-4842-2120-4_5.

[52] SucuriBlog. Serious Cross Site Scripting Vulnerability in TweetDeck – Twitter; 2014. https://blog.sucuri.net/2014/06/serious-cross-site-scripting-vulnerability-in-tweetdeck-twitter.html.

[53] Bisson R. SQL injection. ITNOW, Oxford Journal. 2005;47:25. Available from: http://itnow.oxfordjournals.org/content/47/2/25.full.pdf+html.

[54] Preuss M, Diaz V. Exploit kits - a different view; 2011. Available from: https://securelist.com/analysis/publications/36342/exploit-kits-a-different-view/.

[55] Kotov V, Massacci F. In: Jürjens J, Livshits B, Scandariato R, editors. Anatomy of Exploit Kits. Berlin, Heidelberg: Springer Berlin Heidelberg; 2013. p. 181–196. Available from: http://dx.doi.org/10.1007/978-3-642-36563-8_13.