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

Chapter 1

Do bots, make click, text and behavior metrics irrelevant?

1.1 What are bots and how are they used?

In general, bots are types of software which run automated tasks(which are repetitive and allows for the program to be faster at doing some tasks than what normal humans would be able to. There are many types of bots, for example gaming bots which can be programmed to efficiently react faster than what a human would be able to on certain events in the game, thus making the humans better players. Others examples include auction bots, which hunt for bargains. Ebay went to court in 2000(Rosencrance (2000)) in order to stop this type of behavior, the federal courts in turn then decided to block Bidders Edge and their bots from accessing Ebays API.

In the US 2010 mid-term election, social bots were reportedly used to influence the support of some candidates while effectively slandering others by tweeting thounsands of tweets which pointed to websites which contained fake news about the other candidate (Ratkiewicz et al. (2011)). This example highlights the potential seriousness of social bots.

It is not only in elections these bots have a great influence, during the aftermath of the Boston bombings it was noted that social media(and bots) had a great effect on getting the message out there(Cassa et al. (2013)), but there was also a negative effect which bots attributed on Twitter by re-tweeting peoples' unverified accusations or even checking out the credibility of the source, causing more hurt than good.

Furthermore bots are used to mimic human activity, in for example service applications which appear to be human interaction but is instead a bot which for example tries to ”help” you based on keywords. Bots are used in many areas, but are probably most known for their maliciousness in areas such as spam, bandwidth-thiefs, scrapers, worms and viruses as well as as nodes in a larger scale bot-nets is used for example for DDOS. These bots can be part of bigger bot-nets in order to for example generate revenue through for example click fraud.

The observers article on fake traffic Holiday (2014) describes the problem and its extent:

> A recent study by comScore found that 54 percent of display ads shown in thousands of campaigns between May 2012 and February 2013 never appeared in front of a human being. Rather, the traffic came from bots. As an advertiser, this would be like buying a billboard you were told was seen by thousands of cars a day only to find out that was because the billboard sat next to the assembly line at a Ford plant. Sure, that’s a lot of cars, but there’s no one in them.
>
> In fact, the system is so broken that, for some publishers, knowingly buying traffic that comes from bots is part of their business model. An anonymous publishing executive, who claimed to be buying up to $35,000 worth of traffic per day, recently told Digiday that for publishers running an arbitrage model, all that matters is profit; quality of traffic does not factor into the equation.
>
> Holiday (2014)

This quote from comScore highlights a important point in today’s society. How does click metrics influence us, and how are they used? Are they useless, can they be used to measure something even though results most likely have been contaminated? and how does this type of behavior influence the buyers decision? These and some others are the question I will answer during this chapter.

I will also look into bots in social media, how they interact and how they spread in order to seem like normal users as well as interact as normal users.

1.2 What is social media statistics, and how is it utilized?

With the rise of social media use, social media marketing got a bigger foothold, allowing marketeers to interact with their buyers to a bigger degree, and it allowed the buyers to interact with their brands. Following this trend, many companies have big social media presences. Social media statistics allows the companies to closely monitor and narrow down their demographic and easily make specific content for a specific demographic. Furthermore, it allows for the company to get free publicity through what is called ”organic reach” (getting exposure through users’ activity in order to generate more clicks from users’ contact) as well as for example paid reach(ads sent to a predefined demographic) Social media statistics allows for all of this to happen, as Singh (2013) mentions:

> Social Media statistics about audience likes and dislikes makes it plausible to employ ”push marketing”-techniques to target audiences with advertisements that are relevant to their interests. Also, parameters such as click through rates or CTR can further quantify the success (or failure) of an advertising campaign.

This type of push marketing, utilizes the information the users’ themselves have provided to the mother service(e.g. Facebook), like for example metadata which gives a certain characteristic which allows them to be targeted by the broad filters which services like e.g. Facebook uses.
Businesses also utilizes what is called pull marketing, which is a little more subtle, e.g. referrals, social competitions(”like and share” and get the *chance* to win X”) these methods can generate traction in social media, and be spread by word of mouth.

But most notably, these types of metrics can be abused by bots.

An example of why metrics may be come obfuscated

What happens to social media analytics in a business environment when bots enter the fray? Depending on the function of the bot, its behavior can vary greatly. If the bot is what is called a crawler, the bot will crawl through real peoples feed and try to mimic their behavior, and then crawl and gather more data from other people in the first peoples feed and so on. This type of activity has atleast three types of repercussions, it firstly generates fake interest for other companies' social media analytics, it hides their original intention(liking ”this page”) and it likes ”this page” and generates fake revenue and publicity.

A good real-life example is Narang (2015), which had a set of inter chained accounts posing as real accounts in order to spread their own links about some dubious diet pills. The accounts stole meta data from real accounts. Instead of using compromised accounts to tweet spam links, they were using accounts that impersonated brands and celebrities. Symantec goes on to describe how they defined three types of accounts involved in this scam:

- **Mockingbird**: Used real data from real celebrities for impersonating these individuals
- **Parrot**: Fake accounts using stolen tweets and photographs of real women
- **Egg**: New users with no set avatar

Mockingbirds have the goal of promoting the weight loss tricks. These mockingbird-tweets would get thousands of likes from Parrot-accounts which spiders through the real accounts of the mockingbirds. Parrots then follow any and everyone in the hope that users will follow them back because they are using avatars of attractive women, a tactic that has proven very efficient. The Parrots have real content that they post each day which is fake, an not only the content of the Mockingbird, in order to seem more real. These tweets are usually stolen from real accounts in order to seem real. The parrots will also engage in discussions and post ”reviews” of the diet pills in order to make the diet pills seem more real while the egg accounts just inflate the like counts in order to make the mockingbirds as well as the parrots seem trustworthy. The egg accounts do not post any content, they just follow parrots.

The link provided by the mockingbirds seem real(with pictures of famous people). When

a customer orders a free trial, they register the credit card and subsequently lose their money.
The point of this example is to show how easily for example likes can be misguiding for a company that tries to establish themselves as a brand within social media.

1.3 Do these metrics influence buyers?

As we have seen, metrics can be obfuscated by many different factors; ”click farms”, bots, and other variables which can affect social media analytics. These factors may present in many different ways. In an article entitled ”Beyond likes and tweets: Consumer engagement behavior and movie box office in social media”, Oh et al. (2016), they researched how consumer engagement behavior(CEB) was associated with economic performance based on popular movies released in the US. The study found that CEB in Facebook and YouTube correlated with gross-revenue in opening-week movies. The study further concluded that CEB played a pertinent role in relation to future economic performance, so in some cases pure metrics can have great effects on the consumer itself.

Another good example of this is metrics in movies. In particular ratings on movies and Tv-series through IMDB(Internet movie data base), which is used to share users' opinions on films and tv-series in the form of ratings(top 250 movies, bottom 100) as well as recommendations and critics, both professional and user generated. In a paper entitled ”Judgement devices and the evaluation of singularities: The use of performance ratings and narrative information to guide film viewer choice”(Bialecki et al. (2016)), the researchers noted that the use of metrics such as ratings(moviegoers set a threshold in which served as a ”hurdle” in which movies they wanted to see had to pass). Research also indicated that if a movie had a high rating or were in the top 250 rating-list, it was likely moviegoers would see it because of the ratings themselves. Some subjects noted that the movie did not meet their expectations, but that they had to watch it because of the the ratings:

> *”*I rented this movie on the strength of the ratings and glowing reviews at this site [IMDb]. “Brilliant”,they said. “Dark and beautiful”, they wrote. 8.4 stars. Well, all I can say is, these people must have been on some serious drugs when [they] saw this totally inane movie. . .I give this movie 1 black hole.*”*
>
> Bialecki et al. (2016)

Another example of research done in this field is an article entitled ”The Influence of Social Media: Twitter Usage Pattern during the 2014 Super Bowl Game” which analyzed how people used twitter during the superbowl event specifically looking at consumer interaction with advertisement. The researchers used data mining in order to find correlations in data which may not be otherwise easily delectable. In the article(Shin et al. (2015)), the researchers asked two research questions: ”How does the overall number of tweets differ between a game day and non-game day?”Shin et al. (2015)) and ”What major topics in commercial related tweets were exchanged during the 2014 superbowl game?”Shin et al. (2015)). The data analysed for research question 1 (How does the overall number of tweets differ between a game day and non-game day?) indicated that Super bowl commercials created a buzz. The research also indicated that tweets about a commercial product accounted for 33% of total commercial tweets generated on superbowl day(february 2) as opposed to 0.64% and 9.48% on january 26 and february 9th. Further, more than half of tweets(37 of 49 tweets) registered mentioned Budweiser products or their commercial.

Even though these examples do not directly prove that consumers act on these type of data, one can draw the conclusion that user interacting might create a buzz and get your brand out there, thus leading to potentially more customers, subconsciously or otherwise.

In the case of IMDB-ratings, they seem to influence what movies people want to see, and therefore revenue of movies which are featured in a popular way. This is backed up in a article entitled ”An empirical investigation of user and system recommendations in e-commerce” about general statistics with e-commerce metrics Lin (2014) which could report that:

- 1% increase in user recommendation volume increases product sales by 0.013%.
- 1% increase in user recommendation valence increases product sales by 0.022%.
- 1% increase in system recommendation strength increases product sales by 0.006%.

1.4 How can the statistics become irrelevant?

So, what happens when these statistics become diluted by social media bots, does SMA become irrelevant? is there grounds to believe that bots dilute commercial interest rather than stimulate it?

An article released in 2013, named ”Is that a bot running the social media feed? Testing the differences in perceptions of communication quality for a human agent and a bot agent on Twitter”Edwards et al. (2014) investigated the claim that humans would not see the difference between a bot, and a human in a single newsfeed on Twitter. They used a sample of 240 undergrad students enrolled in a communication course at a large midwestern research university with subjects ranging from the age of 18 to 39 years old. The researchers then used two treatment groups(the twitterbot and the human twitter agent). Those who chose to participate were given a link to a secure webpage with the study. After consenting to the study, the participants were randomly given one of the two twitter pages. The twitter profiles were designed to appear as information provided by the central for disease control(CDC) about sexually transmitted infections. The pages were identical all except for one important detail - the author; in the twitterbot case it was clearly stated that it was a CDC twitterbot, in the other case it was stated that the author was a CDC scientist. The study concluded that

> The findings demonstrate that Twitterbots can be viewed as credible, attractive, competent in communication, and interactional, and might be an appropriate program to transmit information in the social media environment.
>
> Edwards et al. (2014)

This can mean that humans may not be able to detect the difference between humanoids and bots on the web, and one can therefore draw the conclusion that metrics may not provide the whole picture. In the next section we look at an example as to why.

Chapter 2

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.

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 counter-measurements and discuss why so many sites are vulnerable to these well-known attack surfaces.

2.1 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 Acunetix (2016b). 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 Acunetix (2016a). 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. **More on methods**

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 OWASP (2016a). 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 OWASP (2016b).

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 Acunetix (2016b). 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 Acunetix (2016a).

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 Enterprise (2016).

Symantec Internet Security Threat Report 2016

Through the Global Intelligence Network, Symantec has one of the most comprehensive sources of internet threat data Symantec (2016). 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 Symantec (2016).

2.2 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 Project (2010)Project (2013)Acunetix (2016a). They are all vulnerabilities that arise in the development phase of a web application, and they can all potentially cause significant damage.

2.2.1 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 Kirda (2011). Unlike traditional distributed systems security where access control equals authentication and authorisation, a web application uses the same-origin policy Gollmann (2011). 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 Kirda (2011). 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 Edmunds (2016). 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 Edmunds (2016).

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 SucuriBlog (2014). 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 Edmunds (2016).

2.2.2 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 Bisson (2005). Databases often contain sensitive consumer or user information, 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 Acunetix (2016a).

2.2.3 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 Acunetix (2016a).

2.2.4 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 Preuss and Diaz (2011). 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 Kotov and Massacci (2013). 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 Preuss and Diaz (2011). The exploit kits inject shellcode to the vicim process to download a independent malware executable to the victim's hard drive and executes it Preuss and Diaz (2011). These kits has become a competative market and range from several hundreds to over a thousands dollar on the black market Preuss and Diaz (2011), and therefore also have sophisticated code and sometimes even zero-day exploits.

2.3 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 Bisson (2005). 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. **More on how to store passwords**

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. **More on JS Libs**

Bibliography

Acunetix (2016a). Acunetix Web Application Vulnerability Report 2016. https://d3eaqdewfg2crq.cloudfront.net/resources/acunetix-web-application-vulnerability-report-2016.pdf.

Acunetix (2016b). Combating the web security threat.

Bialecki, M., O'Leary, S., and Smith, D. (2016). Judgement devices and the evaluation of singularities: The use of performance ratings and narrative information to guide film viewer choice. *Management Accounting Research*, pages –.

Bisson, R. (2005). Sql injection. *ITNOW, Oxford Journal*, 47:25.

Cassa, C., Chunara, R., Mandl, K., and Brownstein, J. (2013). Twitter as a sentinel in emergency situations: Lessons from the boston marathon explosions. *PLoS Currents*, (JUNE).

Edmunds, B. (2016). *Safe Defaults, Cross-Site Scripting, and Other Popular Hacks*, pages 41–47. Apress, Berkeley, CA.

Edwards, C., Edwards, A., Spence, P. R., and Shelton, A. K. (2014). Is that a bot running the social media feed? testing the differences in perceptions of communication quality for a human agent and a bot agent on twitter. *Computers in Human Behavior*, 33:372 – 376.

Enterprise, H. P. (2016). Security research cyber risk report 2016.

Gollmann, D. (2011). *Problems with Same Origin Policy*, pages 84–85. Springer Berlin Heidelberg, Berlin, Heidelberg.

Holiday, R. (2014). Fake traffic means real paydays.

Kirda, E. (2011). *Cross Site Scripting Attacks*, pages 275–277. Springer US, Boston, MA.

Kotov, V. and Massacci, F. (2013). *Anatomy of Exploit Kits*, pages 181–196. Springer Berlin Heidelberg, Berlin, Heidelberg.

Lin, Z. (2014). An empirical investigation of user and system recommendations in e-commerce. *Decision Support Systems*, 68:111 – 124.

Narang, S. (2015). Uncovering a persistent diet spam operation on twitter.

Oh, C., Roumani, Y., Nwankpa, J. K., and Hu, H.-F. (2016). Beyond likes and tweets: Consumer engagement behavior and movie box office in social media. *Information & Management*, pages –.

OWASP (2016a). About.

OWASP (2016b). Top ten project.

Preuss, M. and Diaz, V. (2011). Exploit kits - a different view.

Project, O. W. A. S. (2010). OWASP Top 10 - 2010. https://www.owasp.org/index.php/Top10#OWASP_Top_10_for_2010.

Project, O. W. A. S. (2013). OWASP Top 10 - 2013. https://www.owasp.org/index.php/Top10#OWASP_Top_10_for_2013.

Ratkiewicz, J., Conover, M., Meiss, M., Goncalves, B., Flammini, A., and Menczer, F. (2011). Detecting and tracking political abuse in social media.

Rosencrance, L. (2000). Federal judge blocks web bot from tapping into ebay.

Shin, H., Byun, C., and Lee, H. (2015). The influence of social media: Twitter usage pattern during the 2014 super bowl game. *International Journal of Multimedia and Ubiquitous Engineering, 2015*, vol 10(3):109–118.

Singh, A. (2013). Social media and corporate agility. *Global Journal of Flexible Systems Management*, 14(4):255–260.

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

Symantec (2016). Internet security threat report 2016.