KING'S College LONDON

University of London

6CCS3PRJ Final Year
An Evil URL Shortener

Final Project Report

Author: Zhengquan Jared Koh

Supervisor: Dr Andrew Coles

Student ID: 1301574

December 4, 2015

Abstract

This project focuses on the a common security issue based on human error and what possible implications it might have on the user. Information Security today is one of the biggest problems in the world.

Due to its vastly complex nature, there are many components which affects information security and one of the major component is human error. All humans make mistakes. This project aims to design and develop a website that exploits human error which is not looking at the URL (Uniform Resource Locator) after visiting a shortened link.

Originality Avowal

I verify that I am the sole author of this report, except where explicitly stated to the contrary. I grant the right to King's College London to make paper and electronic copies of the submitted work for purposes of marking, plagiarism detection and archival, and to upload a copy of the work to Turnitin or another trusted plagiarism detection service. I confirm this report does not exceed 25,000 words.

Zhengquan Jared Koh

December 4, 2015

Acknowledgements

I would like to thank my supervisor, Dr Andrew Coles, who has provided me with invaluable help and support throughout the project. I would also like to thank my family for their feedback and support and friends who helped me with my survey.

Contents

Chapter 1

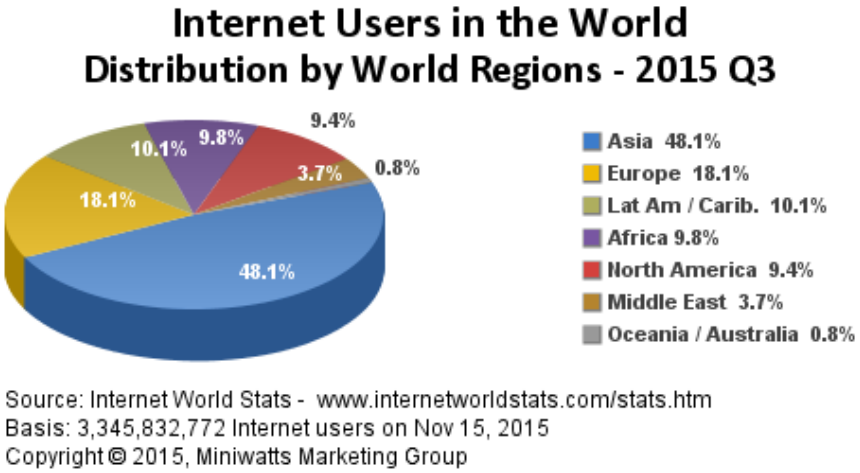
Introduction

1.1 Project Motivation

"The only secure computer is one that's unplugged, locked in a safe, and buried 20 feet under the ground in a secret location... and I'm not even too sure about that one" by Dennis Huges, FBI[1]. Due to amazing technology advancements that has been happening, more and more data are being uploaded to the Internet hence more data are prone to being stolen or unauthorized accessed due to insecure systems or human error.

Security is a big part of today's industry and more companies are investing in security for because they have sensitive information about their customers that should not be of public knowledge.

Figure 1.1



In Relation to Figure 1.1 , We can see that there are approximately 3.35 Billion people in the world who have access to the internet and you could imagine the amount of data that is circulating around World Wide Web and physical systems being used. Its phenomenal. Security on these items are therefore not perfect due to its vastly complex nature and this flaw results in many vulnerabilities within the system or human which eventually leads to unauthorized usage or access to systems or data. 'Security is a not a product, but a process' by Bruce Schneier[2].

1.2 Scope

The scope of my project is to exploit the vulnerability of human error by using social engineering in particular a commonly used service , a Url (Uniform Resource Locator) Shortener , to find how many uniquely different ways i could retrieve sensitive information of the user or unauthorized usage of the user's system.

1.3 Aims and Objectives

The aim and objectives of my project is to develop a website that is used as an URL (Uniform Resource Locator) Shortener , but for malicious purposes such as injecting malicious scripts , false redirection etc. Exploring and developing different scripts and techniques to trick the user in thinking nothing malicious is happening.

The Website should be relatively easy to use. The graphical user interface should be like a normal URL (Uniform Locator Resource) Shortening Service such as bitly[3] or tinyurl[4] in order to deceive others that it is a legit service that is being provided when visited. The user of this service should be aware that this service is malicious and using this at his own cost. I will not hold accountable for any malicious activity that it is being used for.

Requirements for this project will be as followed:

1. To have a functional and graphically alike other Url Shortening Services Website.
2. The user of the shortening services should be aware of its malicious purposes.
3. The structure and flow of the website should be simple and straightforward so it is easy to use. being able to use the feature that is implemented without trouble.

In the next chapter we will discuss the background content for this project and looking into: Web Application Security; Cross Site Scripting; URLs (Uniform Resource Locator); Social Engineering; Ways to Counter XSS.

In the design section we will discuss the design philosophy and system architecture of the

system. Followed by the implementation and testing section which will provide in depth view of implementation of all the website and its malicious features. This will also reveal how we met the requirements and fulfilled deliverables of this project. Lastly, we will discuss conclusion of the project and what we could further explore.

Chapter 2

Background

2.1 Cyber Threats And The Human Error

Cyber Threats have always been constantly rising as more people become technically inclined to hack , more systems are up and more sensitive data is available on the web. The figure below shows you the most common few ways to commit a cyber attack.

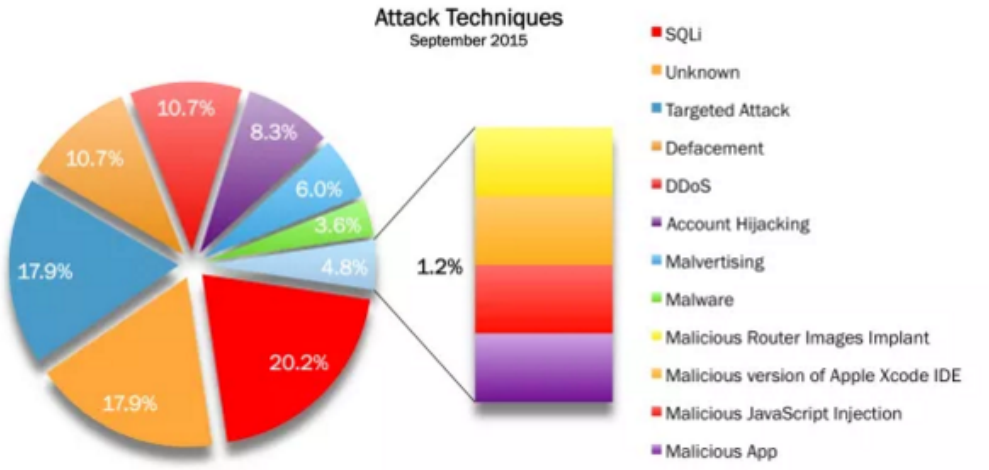


Figure 1.2

There are also different reasons and intentions for people to commit these threats and attacks. The figure below will help us to understand the motivations behind the attacks you the breakdown of these reasons.

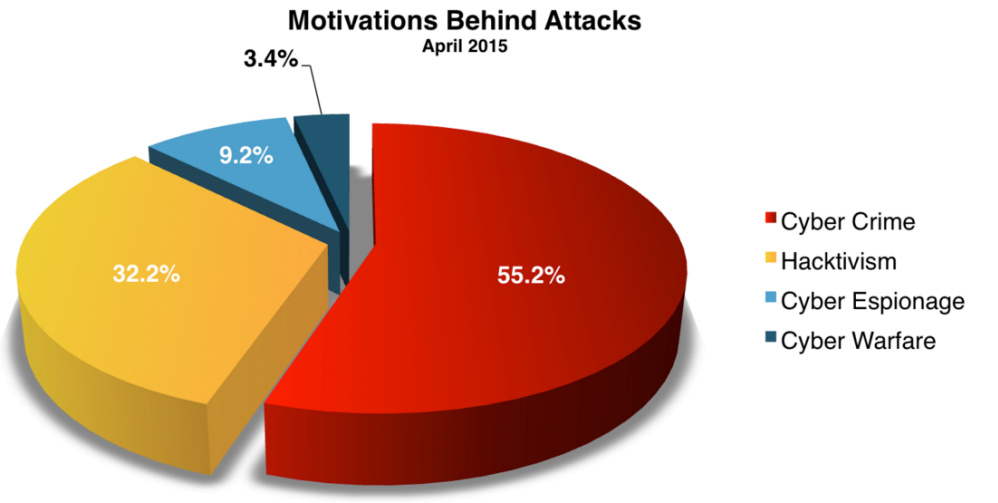


Figure1.3

All humans make mistakes. One of the most intriguing findings from IBM's "2014 Cyber Security Intelligence Index" is that 95 percent of all security incidents involve human error. Many of these are successful security attacks from external attackers who prey on human weakness in order to lure insiders within organizations to unwittingly provide them with access to sensitive information[6]. The human interest factor is also being exploited by attackers and plays a large part in successful security attacks seen today, but it is not always attributed to mistakes made by insiders. Many of these attacks involve social engineering techniques to lure individually targeted users into making mistakes. According to Verizon's "2013 Data Breach Investigations Report," 95 percent of advanced and targeted attacks involved spear-phishing scams with emails containing malicious attachments that can cause malware to be downloaded onto the user's computing device. This gives attackers a foothold into the organization from which they can move laterally in search of valuable information, such as intellectual property[7]. In this project, i did a specific survery (using google forms) on Urls and Url Shorteners and this is result i got.

Figure 1.2 (below) shows that out of 109 random people, Almost a hundred percent of people clicked on a shortened link before. Out of the 108 people whom clicked on the shortened link. A huge 31 percent of them do not notice the URL after clicking it hence in this project i can going to use this human error instances to deceive others that they are using a shortening service, When in fact malicious activities are happening in the back-end.

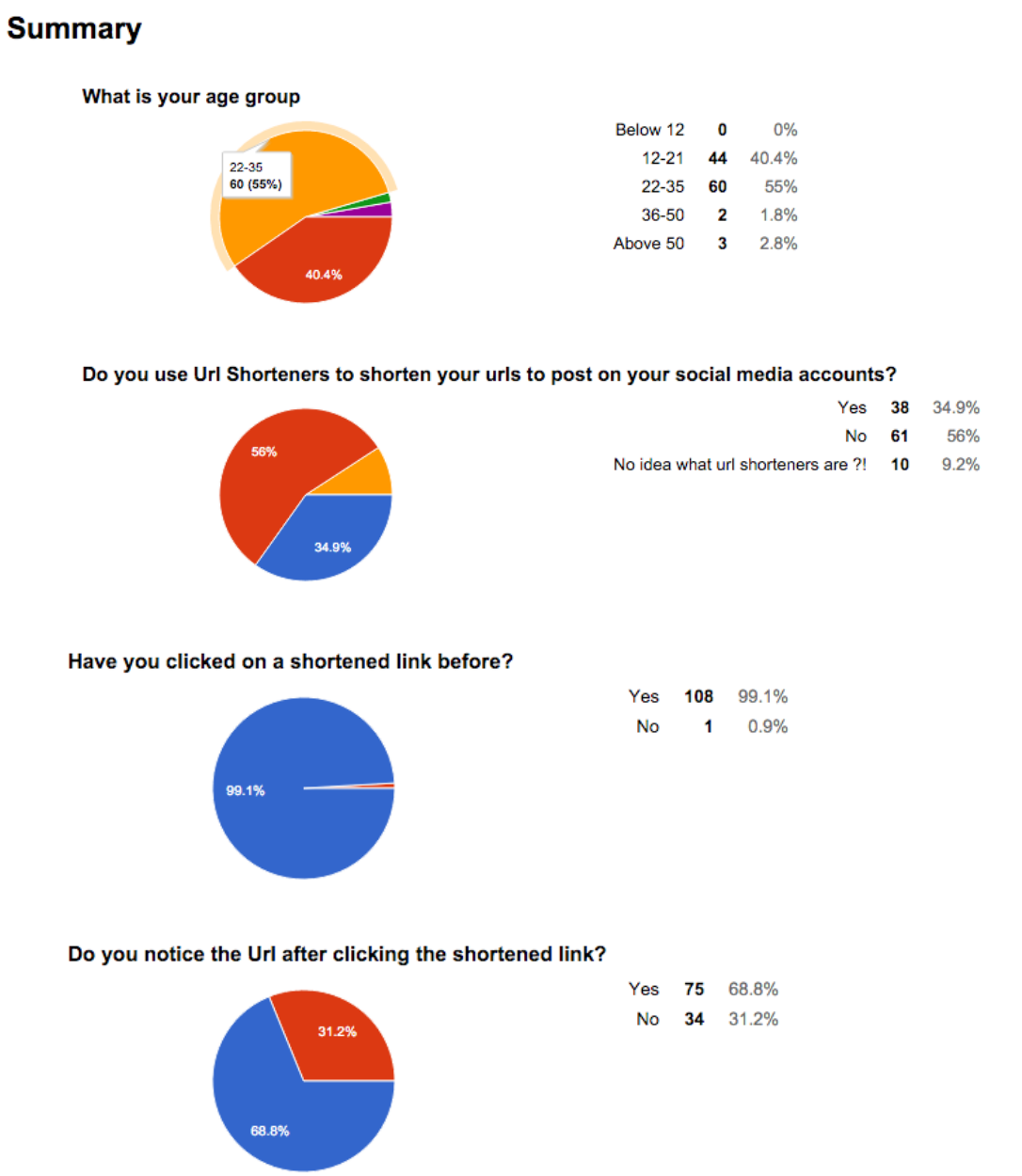


Figure 1.4

2.2 Cross Site Scripting XSS

2.3 Url Shorteners

URL is an acronym for Uniform Resource Locator and is a reference (an address) to a resource on the Internet[7]. People use URLs to access a certain resource on the World Wide Web and sometimes it could get 'unnecessary' long. There are problem that arises with long links.

Firstly , with the rise of usage of mobile internet , it is cumbersome to text a long link and your phone might have a certain limit on the amount of characters you can use in a single text mesasge. Thus it will hinder you from sharing the link with your friends.

Secondly , When a long link is being shared on social media, people often view them as 'spammy' It creates displeasure to their sight because all the see are random characters and symbols most of time. Chances of them clicking it is lower and have been proven.

This is where an URL Shortener comes in. An URL shortener is an online application that converts a regular URL into an abbreviated format. What it does is to generate a key and stores it in its database. In relation to this key is a value , which is going to be the long link that is provided by the user of the service. A generated URL which consists of the host name and the key respectively is then presented to the user and it is the shortened link which is supposed to be shared among the user's friends. Upon clicking of this link, the system would redirect you to the original website that the user intended for its audience to see.

There are many benefits in using a URL shortener. 1. It helps you keep under character limits. Websites like twitter only allows you to tweet 140 characters. Having a shortened link not only allows you to be under the limit but also enables you to add more content to the tweet which improve social media marketing.

2. They promote sharing but making the links more managable. Having shortened links makes it easier to type , both in computers and mobile phones. It encourages to share the link more indirectly which affects the page visits to the website as well.

3. Url shorteners could provide useful features. It could track and compile click data or provide you with a useful interface which allows you to share the page or see a comment made on the page. Take PageTweet for a reference example.

So how dangerous can a URL shortener be? Never seeing the full link of what you click can be devasting. Most people do not notice what they click. Due to web technology , it is often easy to hide scripts in the links themselves and cover it up by using a url shortener. And from the statistics we see in the introduction. The odds of the user looking at the redirected link is pretty low or even if they notice the link, they do not know what it means. In This project i aim to use a URL shortener to redirect you to a webpage that looks exactly like the website that you entered to be shorten.

2.4 Social Engineering

Due to human nature, we are often misled to click a link by how the cover photo looks or the title of the article that leads to the link. These fields can be exploit by evil users by specially crafting a title or inputting specific photo to draw the attention of the user to the link and eventually clicking the link to find out more. These techniques are called social engineering. By studying behavioral patterns and trends of people or certain individuals and exploit that vulnerability for the beneficial gain of the attacker. It is now one of the greatest threat that most organizations today encounter. Security has always been a two way relationship, Not only the System must be secure but the user too. We will chat more on how we could prevent things like social engineering or other attacks generally.

2.5 Ways to counter cyber attacks based on human error

There are definitely different ways to counter these attacks. Both the developer of a website and the user of the internet. Lets talk about the everyday user first.

Secure Browsing in this context could mean many different things.

1. Always read the URL to make sure there is no javascript injection.

2. Downloading and using a plugin that has noscript functionality.

3. Visit trusted websites. (only those that start with Https:// , which means they have a SSL certificate).

4. Use incognito mode for cacheless browsing.

5. Downloading and using a trusted pop up blocker.

6. Always read before clicking allow when applications ask for access to your data.

7. Never use Single sign on.

It is always good to safe on the internet!

Secure Coding should be an important step for a developer to take. Protecting your user's data should be part of your system! There are generally two ways in which we could practice secure coding

1. Input Validation Input Validation is generally one of the key features that we should look at when developing a website. SQL injection and Cross site scripting exploit this feature. Being able to identify the attacks when being entered is a huge advantage , as prevention is better cure. Not allowing the attack to get to the code of the server side is essential when it comes to secure coding.

2. Output Sanitization Output Sanitization is the process which prevents stored Cross Site Scripting from happening, allow the user to be infected/affected by the malicious script that is already in the server. For the malicious script to be in the server, it has to get through the input validation. This process is being used the web application has generating its output from the input and before giving it to the user of the web application.

3. Obfuscation Obfuscation in a computing aspect is rendering your code to be an unreadable standard for humans. Thus attackers would not be able to duplicate what is hidden behind the code.

Chapter 3

Requirements & Specification

This chapter is to give detailed requirements and specifications with suitable figures revealing what should be required of the program and specifying use cases.

Having known the requirements and specifications it would give us a better understanding on how we should design the system in order to fit the use cases.

3.1 Requirements

Requirements are what your program should do

Requirements represent the application from the perspective of the user, or the business as a whole.

3.1.1 Functional Requirements

General

- C1: The website should be able to generate a shorten link by a click of a button.
- C2: The website should be able to generate a unique key, this key would be stored in a database together with the original link for queries.
- C3: The website should be able to copy and re-create all files pertaining to the inputted URL in the server.
- C4: The generated webpage should look exactly like the original link webpage.
- C5: Interacting with the generated webpage should not redirect you to the original webpage but around own system.

- C6: The website should allow you to select an attack.

DDoS Attack

- C7: For the DDoS attack, The website should allow you to input a target/victim of the attack.

Illegal Website/Cryptography

- C8: For the illegal website, the website should allow you to type in the secret password OR have a generated key as a password to redirect you if you edit the link.
- C9: The generated public key should be made known to the individual who wants to access the illegal website.

Remote Tracking

- C10: For remote tracking , the website should allow you to visit a different website to view the google maps.
- C11: Data for the remote tracking should be stored in a database for future reference.

RickRoll

- C12: For RickRoll, the user should not be able to stop the music from playing.

Keylogging

- C13: For Keylogging, the user should not be notified that his data is being recorded.
- C14: The data retrieved from the keylogging should be stored in an accessible place for for future reference for the attacker.

3.2 Specifications

The specifications are how you plan to do it. The specification represents the application from the perspective of the technical team. Specifications and requirements roughly communicate the same information, but to two completely different audiences. In specifications we will also discuss more depth of each task and the priority of each task.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum ante mi, ultrices id erat ac, malesuada dapibus nisl. Ut mattis suscipit ex, et ullamcorper neque posuere non. Suspendisse at luctus mi. Sed id est odio. Vestibulum varius nibh at ligula tincidunt tempus.

Integer sem risus, convallis porttitor lorem nec, eleifend ultrices tellus. Donec ac ex finibus, lobortis enim sed, pharetra eros. Nullam in nisl congue, porttitor lorem vitae, finibus magna. Mauris posuere volutpat ornare. Sed vulputate, sapien non suscipit porttitor, libero quam egestas ex, tincidunt viverra mi urna in ex. Vestibulum viverra fringilla odio id pulvinar. Vivamus nunc justo, interdum id lacus id, condimentum consectetur tellus. Vivamus scelerisque mattis felis, vel lacinia urna ullamcorper ut. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Suspendisse lectus nibh, vestibulum ac eleifend ut, gravida ut quam. Donec eu libero ut orci elementum malesuada.

3.3 Use Cases & Flow

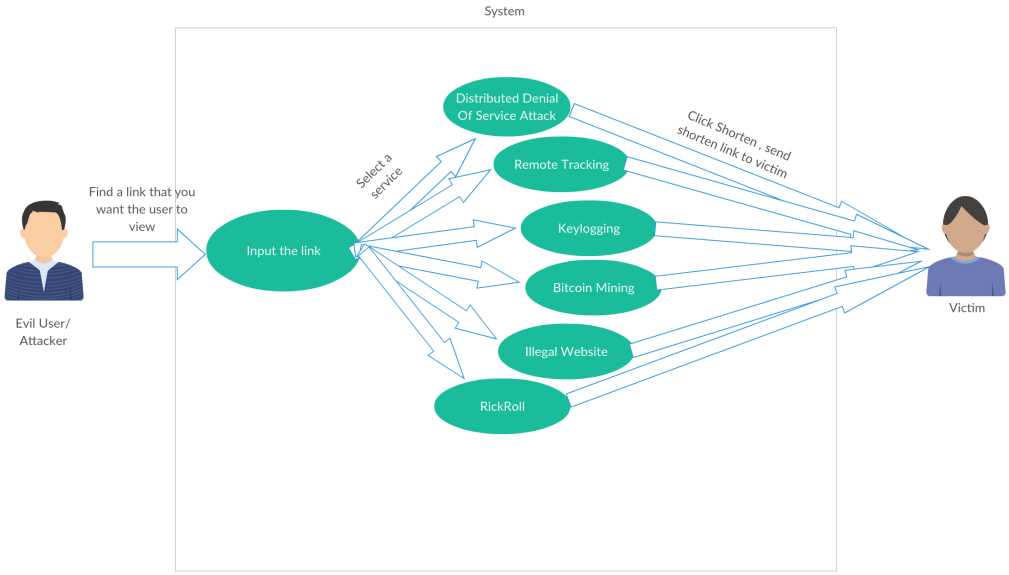


Figure 1.5

Appendix A

References

1. http://www.internetworldstats.com/stats.htm
2. https://www.schneier.com/essays/archives/2000/04/the$_process_of_secur.html$
3. https://securityintelligence.com/the-role-of-human-error-in-successful-security-attacks/
4. http://www.slideshare.net/ibmsecurity/2014-cyber-security-intelligence-index
5. https://docs.oracle.com/javase/tutorial/networking/urls/definition.html
6. http://www.hackmageddon.com/2015-cyber-attacks-timeline-master-index/
7. Web Application Handbook