# Title Page

**Research into common website hacks**and the implementation solutions to generate a usable test template

Table of Contents

[Title Page 1](#_Toc436502596)

[Abstract 3](#_Toc436502597)

[Chapter 1: Introduction 4](#_Toc436502598)

[Chapter 2: Web Security and the origins of Hacking 5](#_Toc436502599)

[Chapter 3: Law and Ethics of hacking 6](#_Toc436502600)

[3.1 Introduction 6](#_Toc436502601)

[3.2 The three hats 6](#_Toc436502602)

[3.3Ethical Hacking 6](#_Toc436502603)

[3.4 Skill levels 6](#_Toc436502604)

[3.5 Fog of Law 7](#_Toc436502605)

[Chapter 4: Top Ten hacks 8](#_Toc436502606)

[4.1 Introduction 8](#_Toc436502607)

[4.2 SQL Injection 8](#_Toc436502608)

[4.2.1 SQL Error Based Injection 8](#_Toc436502609)

[4.2.2 Blind SQL Injection 8](#_Toc436502610)

[4.2.3 Solutions 9](#_Toc436502611)

[4.3 Social engineering 10](#_Toc436502612)

[4.3.1 Common social engineering attacks 10](#_Toc436502613)

[4.3.2Phishing attempts 10](#_Toc436502614)

[4.4 Preventions 10](#_Toc436502615)

[4.4 Cross site scripting attacks 11](#_Toc436502616)

[4.5: Broken Authentication and Session management attacks 12](#_Toc436502617)

[4.6: Click Jacking Attacks 13](#_Toc436502618)

[4.6: DNS Cache Poisoning 14](#_Toc436502619)

[4.7: Symlinking – AN INSIDER ATTACK 15](#_Toc436502620)

[4.8: Cross Site Request Forgery Attacks 16](#_Toc436502621)

[4.9: Remote code Execution Attacks 17](#_Toc436502622)

[4.10: DDOS Attack – Distributed denial of Service Attack 18](#_Toc436502623)

[Chapter 5: Methodology 19](#_Toc436502624)

[5.1 Review the research undertaken 19](#_Toc436502625)

[5.2 State the Research Question 19](#_Toc436502626)

[5.3 Indicate the Methodology 19](#_Toc436502627)

[6. Design 19](#_Toc436502628)

[Chapter 7: Implementation 22](#_Toc436502631)

[Chapter 8: Data Analysis/Synthesis 23](#_Toc436502632)

[Chapter 9: Findings & Conclusions 24](#_Toc436502633)

[References 25](#_Toc436502634)

# Abstract

Research and implementation the top ten commons hacks and the possible solutions. This is will then be turned into a procedure which can be used by any website designer.

# Chapter 1: Introduction

With the growth of the internet it has become an important part of society and life. Its uses have gone well beyond the stage of sending and retrieving information it now caters for entertainment, travel, online purchasing and banking. This however has increased dangers of breeches and information being accessed by some else anywhere in the world. The internet is an uncontrollable place that we must take precautions as users and as developers of web applications that this paper will show the most basics of hacks and how to prevent them.Chapter 2: Web and the origins of Hacking

The World Wide Web was born in 1989 by sir Tim Berens when he laid out his vision for what the web would be in a document called “Information Management: A Proposal.”

In October of 1990 Tim had written the three fundamental technologies that remain the foundation of today.

* HTML: Hyper Test Markup Language. The formatting language for the web.
* URI: Uniform Resource Identifier. An address that is unique and used to identify to each website also known as URL.
* HTTP: Hypertext Transfer Protocol. Allows for the retrieval of linked resources from across the web.

As the web began to grow Tim realised in order for it to reach full potential the web had to be free and not in any way belong to him this lead to revolutionary ideas that are still used today.

(webfoundation, 2008)

As the web grew and more advance uses were created this brought some risks and sparked the age of Hacking.

The term hacker was a positive term in the early 1960s where to be a hacker was to have great skill at pushing programs beyond what they were designed for this was when computers were large machines that need to be stored in controlled environments, computer programmers created “hacks” so that they had shortcuts to help improve performance.

In the 1980s the term took dark tern with the more nitrous hackers using computers and the internet for their own questionable gains. In 1983 one of the first high profiled arrest against hackers, FBI arrest six teenagers from Milwaukee kwon as the “414” due to the area code they were accused of breaking into more than 60 computers including those of Memorial Sloan-Kettering Cancer Canter and Los Alamos National Laboratory.

1986 congress passes the computer Fraud and Abuse Act and the Electronic Communications act. Robert Morris in **1988** releases a self-replicating virus on the Internet designed to exploit security holes in UNIX systems. The virus eventually infects more than 6,000 systems - roughly one-tenth of the Internet's computers at the time - and virtually shuts down the entire network for two days.

Morris is arrested for releasing the virus and is sentenced to three years’ probation, 400 hours of community service and a $10,000 fine.

As a result of the Morris virus, the federal government forms the Computer Emergency Response Team. Based at Carnegie Mellon University in Pittsburgh, its mission is to investigate attacks on computer networks. (Devitt, 2001)

# Chapter 3: Law and Ethics of hacking

## 3.1 Introduction

Hacking has been around for over a century according to sptimes.com. The most famous first hacker was John Draper in the early 1970s in which he made a long distance phone call for free by using a whistle that made a precise tone and blew it into a telephone that tells the phone system to open a line. He discovered the whistle as a prize in a children’s cereal. He later earns the title “Captain Crunch”. (Triguax, 2000)

## 3.2 The three hats

They are three main categories each represented by a different colour hat.

Black Hat Hackers – “Evil hackers” (Guy, 2010)

These are the most infamous epitomised by Hollywood in movies like “Blackhat” staring Chris Hemsworth. These hackers use hacking for personal gain ranging from financial to causing chaos. (Triguax, 2000).

White Hat Hackers – “Angels of the hacking world” (Guy, 2010)

These hackers only hack for the good of the system they try not to damage anything but find the vulnerabilities in doing so make the system stronger. (Triguax, 2000)

Grey Hat Hackers – “Most of us” (Guy, 2010)

In between good and bad they hack systems to prove it can be but later go on to notify the owner. They have a great understanding of the system but want to see if it can be pushed beyond its means whether they have done this with consent of the owner or not this why they are placed as grey hat.

## 3.3Ethical Hacking

White hat hackers go also by ethical hackers. These are what companies hire to carry out penetration testing. This technique helps to determine how secure the company’s system really is. It is a necessary business service. The dilemma with ethical hackers face is how far the white hat hacker goes to find breeches in the system and if they would go beyond the agreed upon actives set by the company. Example social engineering is a technique used by hackers to trick people into giving confidential information. This means that the ethical hacker has logging in credentials that belong to a user and has obtained them using illicit methods. The hacker and company may have breached data protection legislations.

When the ethical hacker gives there report they may not be aware or more important care that they have broken protection laws. Therefor there may be only grey hat hackers. (Triguax, 2000)

## 3.4 Skill levels

They are three main skills level to hacking regardless if you are a white, black or grey hat.

Computer science level

The highest level of hacking this is where hackers understand how networking work, operating systems work protocols and can program to create viruses, malware, botnets etc.

Technician level

This were hackers use hacking tools this where you don’t develop virus software but simply using what is currently already out there through a graphic interface.

Script kitty

Is lowest skill in hacking generally non-technical people where an individual uses scripts be they visual basic scripts malicious code created by others and use them to attack networks, computer systems and deface websites. (Guy, 2010)

## 3.5 Fog of Law

In today’s society privacy and information privacy is a major issue. The owner of the website is not necessary same person who owns the data. For example the owner of a server does not have full rights to client data. This why precautions need to be taken have discloser signed by both the owner and if possible the client or employee so that it is clear that the hack is going on. (Guy, 2010)

# Chapter 4: Top Ten hacks

## 4.1 Introduction

Microsoft, Twitter, NBC, WordPress and Drupal all have in common is they all have been hacked. This chapter covers the top ten most common threats to websites as shown by

(Bhowmick, 2013)

## 4.2 SQL Injection

Injection attacking occurs when there are flaws are in the database, libraries or even the operating system itself. Employees can open what seem to be credible files with hidden commands or “injections”. In doing so they have allowed hackers to access the database to the system. From here the hackers have access to all the data in the database like admin­\_username, users\_passwords. (Bhowmick, 2013)

Finding a weakness or vulnerability in the database layer of an application by using SQL codes is known as SQL injection. This technique tricks the database in allowing unauthorised access to the hackers to the database. (Dorai & Kannan, 2011)

### 4.2.1 SQL Error Based Injection

Error based injection takes advantages over the poor implementation of error handling in the application. When the application returns an error it the hacker finds a way to return useful information along with the error message. (stackexchange.com, 2015)

Following example shows how using UION command to gather all the table names as you see an error message has bene returned back to the hacker.

***www.bh.com/subcat.php?id=null Union all select group\_concat(table\_name) from information\_schema.tables where table\_schema=database()—***

Fig 1: result of SQL injection.

### 4.2.2 Blind SQL Injection

This is when the web application is vulnerable but the results are not visible to the attacker. The results very depending on the logical statement injected into the legitimate SQL statement for the current web page. The hacker checks for vulnerabilities by injecting Malicious SQL codes and if the server returns an Error message. The attacker than interrupts the error message and performs a successful SQL injection attack. The program accepts the data from the client and executes the queries without validating the input. This attack can be time consuming as a new statement must be crafted for each bit recovered many tools are available to automatically run this attack. (Dorai & Kannan, 2011)

Example of Content based Hack

http://newspaper.com/items.php?id=2

This URL sends the following query to the database.

SELECT title, description, body FROM items WHERE ID = 2

The attacker may then try to inject a query that returns 'false':

http://newspaper.com/items.php?id=2 and 1=2

Now the SQL query should looks like this:

SELECT title, description, body FROM items WHERE ID = 2 and 1=2

If the web application is vulnerable to SQL Injection, then it probably will not return anything. To make sure, the attacker will inject a query that will return 'true':

http://newspaper.com/items.php?id=2 and 1=1

If the content of the page that returns 'true' is different than that of the page that returns 'false', then the attacker is able to distinguish when the executed query returns true or false.

Once this has been verified, the only limitations are privileges set up by the database administrator, different SQL syntax, and the attacker's imagination. (OWASP, 2013)

### 4.2.3 Solutions

There are many ways to set up preventions to for SQL injection they all can fall under three levels web server level, code level and SQL Firewall.

#### 4.2.3.1 Code Level

The main and essential prevention of SQL injection is by writing secured source code therefore making intrusion very difficult. This can be done by escaping in PHP generally we escape characters in PHP by replacing single quote (‘) in a parameter by double quote to make it valid SQL statement. Developers use mysql\_real\_escape\_string () function for escaping special characters it adds backslashes to the query before sending it to the database. (Dorai & Kannan, 2011) This method works but is “Clunky” (Scott, Hacking Websites with SQL Injection - Computerphile, 2013) it can still be bypassed if any small mistakes in code for example if you forget the command real than the web application id still unsecure. To get over this we use prepared statements. “It’s a hack on top of a Hack” (Scott, Hacking Websites with SQL Injection - Computerphile, 2013).

Example

SELECT \* FROM USERS WHERE Username =?

This takes the command and treats it very carefully and stores it in the database.

## 4.3 Social engineering

Social engineering is a non-technical method of hacking. It entails on human interaction where people give away vital information. It is the one of the major threats to face companies. (Bhowmick, 2013)

Criminals use social engineering because it is easier to exploit natural inclination than it is to technically gain information. (Criddle, What is Social Engineering? Examples and Prevention Tips, 2015) This entails general hackers pretending to be as some they are not. This can be phone calls pretending to be technical support asking for admin privileges to email posing as PayPal or a bank telling you that account is at risk and you must click the link and enter your credentials. (Guy, 2010)

### 4.3.1 Common social engineering attacks

Email from a friend: If a hacker gets access to all user’s email than they have all of the users contacts. From here they could send emails containing, a link that has malware that lets the attacker gain control over the victims computer this can be done using downloads or they go straight for financial gain and ask for money under false pretences. (Criddle, What is Social Engineering? Examples and Prevention Tips, 2004)

### 4.3.2 Phishing attempts

Typically a phisher sends an email, IM, Comment or text message that comes from a legitimate source. These messages contain the following scenarios or story

* The message may explain there is a problem and that you are to verify information. There may be a link to a site that looks familiar and everything looks legitimate this because the hacker has set up this site to be exactly as the site and they want you to act before you think. This the methodology for all phishing scams.
* The message may notify you are a winner the most notorious scam it can be calls emails or even by post telling the victim you have received a prize and in order to do so you must enter personal details to claim it.
* A message asking for help preying on people kindness theses phishes ask for aid for charity. (Criddle, What is Social Engineering? Examples and Prevention Tips, 2004)

### 4.3.3 Preventions

* Slow down do not be so quick to react this is what the scammer is hoping for if they use high pressure terms than be sceptical and never let urgency influence your careful review. (Criddle, What is Social Engineering? Examples and Prevention Tips, 2004)
* Use of email validation tool (more research)

## 4.4 Cross site scripting attacks XSS

“Cross site scripting is the number one vulnerability on the web today” (Scott, Hacking Websites with SQL Injection - Computerphile, 2013).

“Part of the big three injection, denial of service DOS and Cross site scripting” (Scott, Hacking Websites with SQL Injection - Computerphile, 2013).

Also known as XSS attack occurs when an application URL “get request” or file packet to the web browser window and bypassing the validation process. Once the script is triggered its deceptive property makes users believe that the compromised page is legitimate.

For example www.example.com/abcd.html has XSS script in it, the user might see a popup window asking for their credit card info and other sensitive info. (Bhowmick, 2013)

This is normally done using Java script. Java script sits in the middle of the Html doing most if not all the background operations. This makes java script very powerful but also very dangerous. For example an online bank if you had java script that when a user enters there bank account number the java script sent it to a third party than log in the user.

Example of how a hacker uses it

Imagine we have a google search in it we enter valid java script code.

GOOGLE

<script> //Java script</script>

This means the one user will execute the java script and the web page will do what the hacker wants. In this case this only affects the one instance but it can be more malicious with code being attached to the sites. (Scott, Cracking Websites with Cross Site Scripting - Computerphile, 2013)

### 4.4.1 Solutions

It is very easy for this hack to become possible if the developer makes a mistake were user input is needed. Escaping (&lt ;) is used if the user enters in the following<script> it is stored as just characters and not taken in as code. (Scott, Cracking Websites with Cross Site Scripting - Computerphile, 2013)

## 4.5: Broken Authentication and Session management attacks

The user authentication system of a website is weak, hackers can take full advantage. Authentication systems invoice passwords, key management, sessions IDs and cookies that can allow a hacker to access your account from any computer as long as it is valid. If a hacker can exploit the authentication and session management system, they can assume the users identity. (Bhowmick, 2013)

### 4.5.1Solutions

To find out if your website is vulnerable to this attack ask yourself these questions.

* Are user credentials weak (e.g. stored using hashing or encryption)?
* Can credentials be guessed or overwritten through weak account management functions (e.g. account creation, change password, recover password, weak session IDs)?
* Are session IDs exposed in the URL (e.g. URL rewriting)?
* Are session IDs vulnerable to session fixation attacks?
* Do session IDs timeout and can users log out?

If you answer yes to any of these questions then your site is vulnerable to a hacker. (Bhowmick, 2013)

(DEMO by OWSAP Mantra)

https://www.youtube.com/watch?v=o1WVx6eYE-M

## 4.6: Click Jacking Attacks

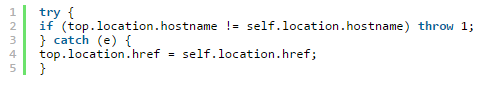
Clickjacking also known as UI Redress Attack is when a hacker uses multiple opaque layers to trick a user into clicking the top layer without them knowing. Hence the attacker is hijacking clicks that are not meant for the actual page, but for a page where the attacker wants you to be. For example, using a carefully combination of stylesheets .iframes and text boxes a user could be led to enter there password for their bank or there identity but actually typing into an invisible frame that controlled by the attacker. (Bhowmick, 2013) This can be used for simple reasons such as getting like that a user did not knowingly gave. <https://www.youtube.com/watch?v=4k4HhKhmwUQ>

### 4.6.1 Solutions

X-frame Options: Include x-frame options to your HTTP header in all your webpages. This will prevent your site from being placed within a frame by most web browser except Firefox.

It is a simple solution although has its cons. It needs to access a webserver configuration and scripting language on the server.

Framebuster JavaScript: The web designer detects by Java Script if their site has been placed in a frame and reload it main browser window.



This can be hacked but intern it can be hacked back. There are many things to consider when using this solution for example what if a user has java script turned off.

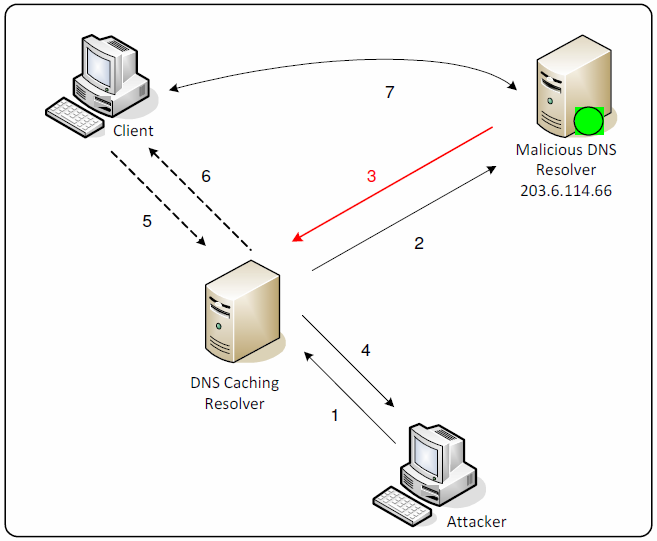
Move elements on your page: When a hacker frames your site they are blind. Your browser knows where to render a button but it won’t be possible foe an attacker to check where this button is.

If you put you button on different places on different pages than it’s hard for the attacker to trick users where to click. This can however agitate legitimate users.

Require additional action: Require your users to mark a checkbox, fill in a password

## 4.6: DNS Cache Poisoning

DNS means Domain name server this when a user looks for a site and using cache it receives the site much faster. DNS Cache poisoning involves old cache data that a user may not think they still have been toxic. Also known as DNS spoofing hackers can identify vulnerabilities in a domain name system, which allows them to divert traffic from legit servers to fake website or server. This from of attack can spread and replicate itself from on DNS server to another. (Bhowmick, 2013)



### 4.6.1 Solution

Query id is what is sent from the client to the server for a website the hacker can guess the id number. Randomize the id instead of in order id send randomized numbers this could be still guest as its on 16 bit number.

Keep your resolver private and protected. If you operate your own resolver, its usage should be restricted to users on your network to help prevent its cache being poisoned by hackers outside your organization. It should not be open to external users. You can check for open resolvers on your network using The Measurement Factory's online tool.

Configure it to be as secure as possible against cache poisoning. Protections built in to DNS software to protect against cache poisoning include adding variability to outgoing requests, to make it harder for a hacker to get a bogus response accepted. Possible ways of doing this include:

* using a random source port (instead of UDP port 53)
* randomizing the query ID
* Randomizing the case of the letters of the domain names that are sent out to be resolved. (That's because name servers will treat example.com and ExaMPle.com the same when it comes to resolving the IP address, but it will reply using the same case as the original query.)

Manage your DNS servers securely. When it comes to your authoritative servers, you need to decide whether to host them yourself or have them hosted at a service provider or domain registrar. "No one cares about your security as much as you do, so we advise hosting and managing yourself -- if you have the skills to do so," says Brenton. "If you don't have those skills, then of course it is better to get someone else to do it for you. It's not just a matter of expertise, but also of scale because many organizations need to have DNS servers in three or four places around the world. (Rubens, How to Prevent DNS Attacks - eSecurity Planet, 2013)

## 4.7: Symlinking – AN INSIDER ATTACK

A symlink is a special file that points to a hard link on a mounted file system. A symlinking attack occurs when a hacker positions the symlink in such a way that the user or application that is being used thinks it is using the right file but it is not. The endpoint file is an output that has been modified which is directed to a septic location on the users system the modifications could be overriding corrupting or even changing permissions. In different variations of symlinking attack a hacker may be able to control the changes to a file grant themselves advance access, insert false information or expose sensitive information. (Bhowmick, 2013)

### 4.7.1 Solution

## 4.8: Cross Site Request Forgery Attacks CSRF

The third biggest hack out of the major three, lesser known but still can have a long lasting damage effect if not prepared for. Cross site request forgery attack happens when a user is logged into their account of a site the hacker sends them a forged HTTP request to collect their cookie information.

The cookie remains valid as long as the attacker or user stays logged in to the account which is why it is important to logout of every session so that it is terminated immediately. In other cases the hacker can generate requests to the application and the system will not be able to differentiate between the hacker and user.

Example of CSRF

<http://example.com/app/transferFunds?amount=1500&destinationAccount=4673243243>

<img src=”<span style=”color: red;”>http://example.com/app/transferFunds?amount=1500&destinationAccount=attackersAcct#</span>” width=”0″ height=”0″ />

In this case the hacker creates a request that will transfer money from a user’s account, and then embeds this attack in an image request or iframe stored on various sites under the attacker’s control.

(Bhowmick, 2013)

Another example we have a poorly made bank application.

You’re Account Other Account

Transfer

€100

Button

GO

Once we hit go it send the request and sends it as a post.

As long as we are valid authentication and have pass security checks, the problem is that the bank forum is public. If the hacker creates a fake web page and puts the bank forum on their site and sends a request €1000 and all the details like account and hides it on the web page. (Scott, Cross Site Request Forgery - Computerphile, 2013)

Here our hacker has created a blog.

At the right side hidden are the prefilled details for the bank.

When the user writes a comment and hits go the request does not go to the blog but to the bank site. It sees the authorized details and ignores the blog info and sends the transaction through.

It can even become more malicious if a user just loads the page the transaction of a €1000 can go through automatically in the background because the site hasn’t checked where the form has come from. (Scott, Cross Site Request Forgery - Computerphile, 2013)

When a reader of the b Hidden form

**The world’s greatest BLOG**

**Text………………………………………………………………………………….**

Account: 0221555444

€1000

Comment…

GO

### 4.8.1 Solution

In http hypertext transfer protocol has the refer header this prevents this problem so when a request is sent it checks to see if the site sending the request matches the refer header. However if the user have privacy settings like ad blocker that prevents the refer header being sent it can annoy users.

A better way to deal with this problem is we can use the one time key. This is a random one time string of characters created each time the form is created so every request sent must have this token. The hacker cannot know this token and all the requests sent by the world’s greatest blog to the bank system ignores it. (Scott, Cross Site Request Forgery - Computerphile, 2013)

## 4.9: Remote code Execution Attacks

Nothing on a computer can run unless code is running all programs must have that runs in the background so that we can use the system. If the system or application is poorly made than an attacker can change the execution code and remotely doing what they please with your system.

A remote execution attack is a result of either a server or client side security weaknesses. Vulnerable components may include libraries directions on a server that haven’t been monitored frameworks and other software modules. Applications that use these components are always under attack through things like scripts malware and small command lines that retrieve information. (Bhowmick, 2013)

### 4.9.1 Solution

## 4.10: DDOS Attack – Distributed denial of Service Attack

Distributed denial of service attack is where a server or machine is unavailable to its users. The hacker can than proceed to compromise the entire website or a specific function for their own gain.

The goal of DDOS is to temporally or completely disrupt the running system. This most common example is sending tons of URL requests to a website or a webpage in a very small amount of time. This will cause bottlenecking at t the server side because the CPU runs out of resources.

Denial of service attack are considered violations of the internet architecture boards Internet proper use policy also violate the acceptable use policies of virtually all Internet service providers. It can cause in loss of business, money or even cause death. (Bhowmick, 2013)

Distributed Denial of service is specific in that a hacker could use other people unknowingly download a virus to their system which would send the URL request to a company and hold the system at ransom.

When this hack began the internet was much smaller and easier to over flow but now due faster connections and bigger bandwidth, an improved version of this hack called Amplified Denial of service ADOS. (Scott, The Attack That Could Disrupt The Whole Internet - Computerphile, 2014)

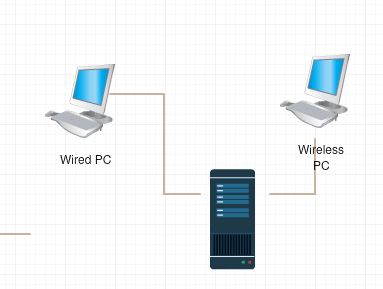
### 4.10.1 ADOS Amplified Denial of service

To explain this we have to talk about two protocols TCP and UDP

TCP is most of the web works it is a two way hand shake when a packet is sent another is sent back all in the right order.

UDP it sends data at a rapid pace with no real concern on the order of packets as long as they get there this used in Voice over ip (skype).

There is a flaw in certain implementation of UDP is that a hacker can spoof the return address which means I can be someone else. This is not a problem due to we can only send a small amount of data a time however NTP the network time protocol. This manages all the clocks synced on laptops, tablets and phones. There is a serious design flaw in this protocol and that is the command MONLIST. It sends the details of the last 600 people who have request the time from that computer back. This means the attacker can send can send a request using that command to the time server; spoof the victim’s ip address and send an unmanageable amount of data to that computer. (Scott, The Attack That Could Disrupt The Whole Internet - Computerphile, 2014)



Time Server

206\* The amount of data

## 4.10.2 Solution

There is no real solution to this hack websites can hire a company a block these attacks at the network layer. This way campaigns to shut down the relay just like the email relays that sent emails to people every time a user event on their site. The same must be done for DDOS attack a recommended vendor that a company or website might use www.a10networks.com.

(Scott, The Attack That Could Disrupt The Whole Internet - Computerphile, 2014)

# Chapter 5: Methodology

## 5.1 Research Undertaken

The research in this thesis has indicated that these are the most commons hacks that exist to date some have evolved into more update and deadly attacks others still happen when they shouldn’t.

This project will implement the ten most common hacks as provided by defencely.com the following hacks will be implemented to the test site.

Injection attacks known as one of the big three if a hacker gets access to the database than there is nothing the hacker cannot do. From the research undertaken the project will apply two types of injection Error and Blind injection to prove this hack a SQL database, there will be used two types of SQL used to prove this 5.0 and 4.0 SQL databases. This is because Blind injection only works with SQL version 4.0. This will show a website that has weak to no level of security can be breeched.

Social engineering a non-technical method the project will investigate into phishing emails and if it is possible to use software to remove dangerous email, how spam folders are used and current standards that are in place to protect against social engineering. This is the most popular technique and is one the most current threats to date.

Cross site scripting the number one hack and also part of the big three mostly done in java script due to the power if not fully protected from it can cause long lasting damage. How this will be implemented is valid java script will be entered into a login page and all data entered in the current session will be sent to an external source before allowing the user to login. This will show the power of java script and how dangerous it can be.

Distributed denial of Service Attack sending multiple packets to a server or user and congesting the system this is what could be feared as the hack that could break the web this will be implemented by sending packets across the network to the local server and shutting down the site.

## 5.2 Research Question

To do each hack and prove it can be done than find a solution and apply it, finally document the hack and solution for the procedure.

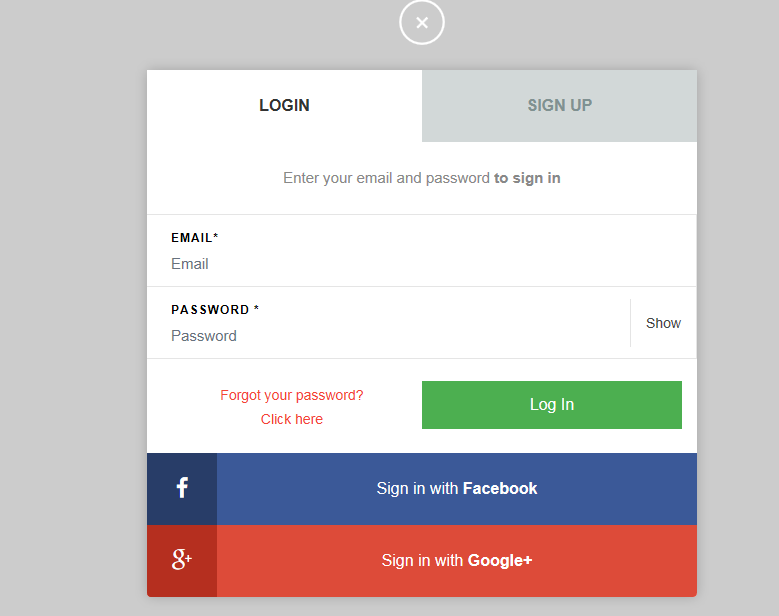
# Chapter 6. Design

This is a formal design which includes best practices for your selected software development methodology. You might cover design documentation including:

(most of your design documents should be presented prior to Christmas;

many may not be completed)





# Chapter 7: Implementation

This project will have an agile approach, with the process divided into a number of sprints each sprint will have an achievable goal that have a two week duration. It has been agreed that time is allocated to a sprint as follows:

1 week = 12hrs (4hrs lab work + 8hrs project work).

Each sprint = 2 weeks in duration.

Activities per sprint should include where applicable:

* Research
* More design detail.
* Coding – adhere to recognised coding standards.
* Review Sprint and update plan and backlog items.
* Update thesis

|  |  |  |  |
| --- | --- | --- | --- |
| Sprints | Status | Start Date | End  Date |
| Sprint 1 – Set up Xamp install PHP driver and a version control. | Done | 7/12/15 | 14/12/15 |
| Sprint 2- Implement SQL injection Hack & solution and document the outcome. | Done | 14/12/15 | 19/12/15 |
| Sprint 3 - Implement cross site scripting attack Hack & solution and document the outcome. | Done | 1/02/16 | 12/02/16 |
| Sprint 4 - Clickjacking Attacks Hack & solution and document the outcome. |  | 15/2/15 | 21/2/15 |
| Sprint 5- Cross site request forgery attacks |  | 15/2/16 | 21/2/16 |
| Sprint 6 – Create word Procedure |  | 22/2/16 | 26/2/16 |
| Sprint 7 - DNS cache poisoning |  |  |  |
| Sprint 8 - Symlinking |  |  |  |
| Sprint 9 - Remote code execution attacks |  |  |  |
| Sprint 10 – Continue word Procedure |  |  |  |
| Sprint 11 - Broken authentication and session management attacks |  |  |  |
| Sprint 12 – DDOS attack – distributed denial of service attack |  |  |  |
| Sprint 13 - Social engineering attacks |  |  |  |
| Sprint 14 – Finish word Procedure |  |  |  |

## Sprint Plan

### Sprint 1:

|  |  |  |  |
| --- | --- | --- | --- |
| Sprints | Status | Start Date | End  Date |
| Sprint 1 – Set up Xamp install PHP driver and a version control. | Done | 7/12/15 | 14/12/15 |

|  |  |  |
| --- | --- | --- |
| **Task Number** | **Details** | **Status** |
| 1 | Research into work environments | Complete |
| 2 | Install work environment | Complete |
| 3 | Install PHP driver | Complete |
| 4 | Research into version control | Complete |
| 5 | Install and set up chosen version control | Complete |

The aim of this sprint was to set up the work environment. Possible ways of hosting the test websites where looked in too such as in a house server, virtual machine or cloud hosting. XAMP was chosen it is used to host a database and all websites locally. PHP drivers allow to code php files that can run on Mozilla Firefox. Through extensive research into many version controls GitHub was chosen as it was best suited to the project.

#### Issues

* Problem with my antivirus shields blocking the Xamp process.
* Xamp ran into difficulty with finding an available process id.
* Finding a way to implement the hacks in a safe environment.
* Version control has conflicting problems with pushing files from the XAMP folder.

#### Resolutions

* Turn off shields.
* Set a given Process id.
* Using XAMP as a local server and move files from the htcdocs to the c drive.

### Sprint 2:

|  |  |  |  |
| --- | --- | --- | --- |
| Sprints | Status | Start Date | End  Date |
| Sprint 2- Implement SQL injection & Solution and write up the procedure | Done | 14/12/15 | 19/12/15 |

|  |  |  |
| --- | --- | --- |
| **Task Number** | **Details** | **Status** |
| 1 | Research into tutorials of how to demonstrate SQL injection | Complete |
| 2 | Look at the most common practice to fix the attack | Complete |
| 3 | Implement a website that is connected to a database | Complete |
| 4 | Attack the website | Complete |
| 5 | Document the attack | Complete |
| 6 | Create a second instance of the site and apply the chosen solution. | Complete |

This sprint is the first hack on defencely.com top ten hacks. SQL injection shows how poor coding in the application layer can cause unauthorised access to a websites database. This hack is proved by implement a simple input forum connected to a database then by enter SQL code into the bar a table was deleted from the database shows the worst case scenario from this hack.

#### Issues

* Finding a website to perform the hack.
* Syntax of the actual hack.
* Connecting to the database through PHP
* Find a proved upon fix.

#### Resolutions

* Implanted a small piece of a website that would prove the hack does work.
* Followed tutorials on SQL injection.
* Followed tutorials on how to connect php

### Sprint 3:

|  |  |  |  |
| --- | --- | --- | --- |
| Sprints | Status | Start Date | End  Date |
| Sprint 3 - Implement cross site scripting attack Hack & solution and document the outcome. | Done | 1/02/16 | 12/02/16 |

|  |  |  |
| --- | --- | --- |
| **Task Number** | **Details** | **Status** |
| 1 | Research into tutorials of how to demonstrate XSS | Complete |
| 2 | Look at the most common practice to fix the attack | Complete |
| 3 | Implement a website that is connected to a database | Complete |
| 4 | Attack the website | Complete |
| 5 | Document the attack | Complete |
| 6 | Create a second instance of the site and apply the chosen solution. | Complete |

This sprint looks at cross site scripting or XSS which is the insertion of unsanitised code. For this a comments box was created that pushed up comments to the web page. Code is entreated in to the comment box starting with non- critical as h1 tags to inserting java script for retrieving static cookie information.

#### Issues

* Finding a website to perform the hack.
* Syntax of the actual hack.
* Connecting to the database through PHP
* Find a proved upon fix.

#### Resolutions

* Implanted a small piece of a website that would prove the hack does work.
* Followed tutorials on SQL injection.
* Followed tutorials on how to connect php

# Chapter 8: Data Analysis/Synthesis

Explain the data that you’ve collected.

If your data is not what was predicted, then conduct another literature review and see if other researchers obtained results similar to yours.

Obtaining an unexpected result does not lower your grade! Simply stand-over your findings.

Discuss other approaches with your supervisors.

(not required prior to Christmas)

# Chapter 9: Findings & Conclusions

# References

Bhowmick, S. (2013). *10 MOST POPULAR WAYS HACKERS HACK YOUR WEBSITE*. Retrieved 8 2015, from http://defencely.com/blog/10-popular-ways-hackers-hack-website/

Botha, L. (2011). *Step-byStep SQL Injection Attack (HQ)*. Retrieved 11 05, 2015, from https://www.youtube.com/watch?v=7H358PrFagc

Criddle, L. (2004). *What is Social Engineering? Examples and Prevention Tips*. Retrieved 11 28, 2015, from Webroot.com: www.webroot.com/ie/en/home/resources/tips/online-shopping-banking/secure-what-is-social-engineering

Criddle, L. (2015). *What is Social Engineering? Examples and Prevention Tips*. Retrieved October 19, 2015, from Webroot.com: http://www.webroot.com/ie/en/home/resources/tips/online-shopping-banking/secure-what-is-social-engineering

Devitt, M. (2001, June 18). *A Brief History of Computer Hacking*. Retrieved 15 07, 2015, from Dynamicchiropractic.com: http://www.dynamicchiropractic.com/mpacms/dc/article.php?id=18078.

Dorai, R., & Kannan, V. (2011). Journal of International Commercial Law and Technology. *SQL Injection-Dtabase Attack Revolutuion and Prevention, 6*(4), 8.

Guy, E. t. (2010, 12 12). *Introduction to Hacking*. Retrieved 10 17, 2015, from YouTube: https://www.youtube.com/watch?v=yGIHjTmTFfA

Innovation, S. (n.d.). *www.securityinnovation.com.* Retrieved 10 07, 2015, from https://www.securityinnovation.com/uploads/sample%20report%20-%20test%20plan.pdf

Khoury, A. (2009). *How to Build a Community Website Part 1 Social Network Production Tutorials*. Retrieved 08 2015, from https://www.youtube.com/watch?v=aj2ugj0Tzz8&list=PL5DF2FC61F89A44E6&index=1

Kingsley, A. (n.d.). *27 Free CSS HTML Login Form Templates*. Retrieved 09 2015, from http://designscrazed.org/css-html-login-form-templates/

MiniME23qq. (2012). *[TuT] How To Hack A Website Using SQL Injection* . Retrieved 11 05, 2015, from https://www.youtube.com/watch?v=7H358PrFagc

OWASP. (2013, August 26). *Blind SQL Injection - OWASP*. Retrieved 11 2015, 28, from Owasp.org: https://www.owasp.org/index.php/Blind\_SQL\_Injection

Rouse, M. (2011). *WhatIs.com*. Retrieved 10 30, 2015, from http://whatis.techtarget.com/definition/KLOC-thousands-of-lines-of-code

Rubens, P. (2010). *10 Ways to Prevent or Mitigate SQL Injection Attacks*. Retrieved 10 2015, from http://www.enterprisenetworkingplanet.com/netsecur/article.php/3866756/10-Ways-to-Prevent-or-Mitigate-SQL-Injection-Attacks.htm

Rubens, P. (2013, 12 02). *How to Prevent DNS Attacks - eSecurity Planet*. Retrieved 12 13, 2015, from Esecurityplanet.com: http://www.esecurityplanet.com/network-security/how-to-prevent-dns-attacks.html

Scott, T. (2013, October 23). *Cracking Websites with Cross Site Scripting - Computerphile*. Retrieved from YouTube: https://www.youtube.com/watch?v=L5l9lSnNMxg

Scott, T. (2013, 12 3). *Cross Site Request Forgery - Computerphile*. Retrieved 12 2015, 09, from Youtube: https://www.youtube.com/watch?v=vRBihr41JTo&list=PL96C35uN7xGLux5q2c4P\_IqbKF11-pfsR&index=10

Scott, T. (2013, October 9). *Hacking Websites with SQL Injection - Computerphile*. Retrieved 11 28, 2015, from youtube.com: https://www.youtube.com/watch?v=\_jKylhJtPmI

Scott, T. (2014, 03 14). *The Attack That Could Disrupt The Whole Internet - Computerphile*. Retrieved 12 2015, 09, from YouTube: https://www.youtube.com/watch?v=BcDZS7iYNsA&index=5&list=PL96C35uN7xGLux5q2c4P\_IqbKF11-pfsR

stackexchange.com. (2015, June 21). Retrieved 11 28, 2015, from Security.stackexchange.com: http://security.stackexchange.com/questions/92137/can-anybody-explain-error-based-sql-injection-attacks

Sutherland, I. (2014). *Cyber Crime Thriller Author - Ian Sutherland*. Retrieved 10 15, 17, from Ian Sutherland: http://ianhsutherland.com/2013/05/02/ethical-hacking/

Triguax, R. (2000). *A history of hacking*. Retrieved October 2015, 17, from Sptimes.com: http://www.sptimes.com/Hackers/history.hacking.html

webfoundation. (2008). *History of the Web*. Retrieved 12 07, 2015, from World Wide Web Foundation: http://webfoundation.org/about/vision/history-of-the-web/