
University of Portsmouth
BSc (Hons) Computer Science
Second Year

Ethical Hacking (EHACK)
M30239
January 2024 - June 2024
20 Credits

Thomas Boxall
up2108121@myport.ac.uk

Contents

1	Lecture - Introduction to Penetration Testing (2024-01-22)	2
2	Lecture - Information Gathering (2024-01-29)	5
3	Lecture - Exploitation (2024-02-05)	6
4	Lecture - Social Engineering and Decoys (2024-02-12)	8
5	Lecture - Privilege Escalation (2024-02-19)	11
6	Lecture - Misconfigured File Permissions (2024-02-26)	13
7	Lecture - Exam Preperation (2024-03-11)	14
8	Lecture - Web Application Attacks (SQL Injection) (2024-04-15)	15
9	Lecture - Introduction to Binary Exploitation (2024-04-22)	18

Page 1

Lecture - Introduction to Penetration Testing

📅 2024-01-22

🕒 0900

🎓 Tobi

“If you start searching for Vulnerabilities in WordPress, you will find lots”

1.1 Introduction to Ethical Hacking

Ethical Hacking is the process of finding vulnerabilities and reporting them to the correct people so that they can be rectified. Ethical hacking is a core component of the broader thing which is *Cyber Security*, in which we are striving to protect the three core properties: Confidentiality (protecting information from being disclosed), Integrity (protecting information from being modified) and Availability (ensuring access to information when needed).

1.2 Penetration Testing

Penetration Testing is the continuous process of identifying, analysing, exploiting and making recommendations to vulnerabilities. Pen. Testing is often described as a cycle, which follows a very strict plan within a fixed timeframe.

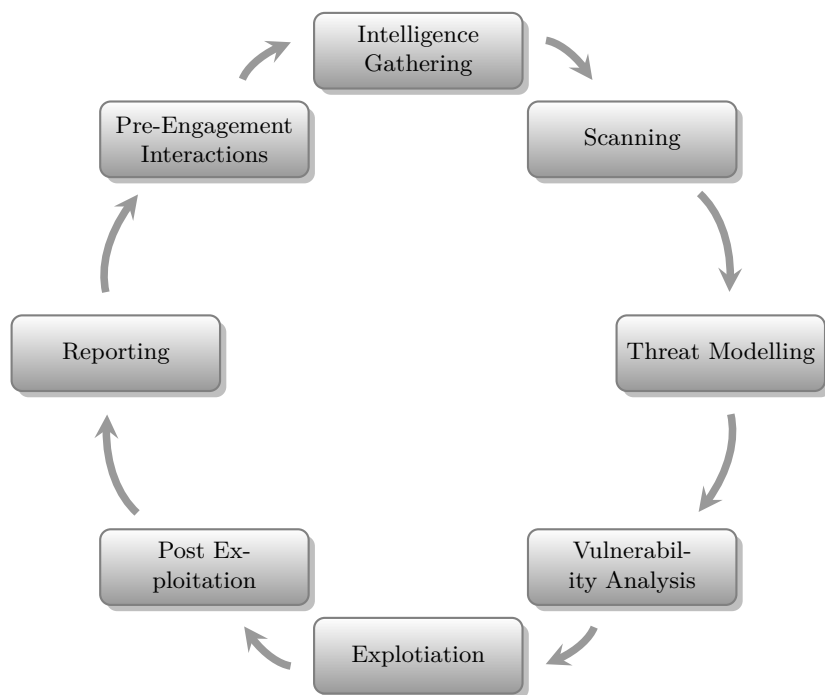


Figure 1.1: Pen. Testing Cycle

There are three types of Pen. Testing:

Black Box where little or no knowledge is disclosed to the pen. tester

Grey Box where some knowledge is disclosed to the pen. tester. They will not be provided full information on anything

White Box where all knowledge is disclosed to the pen. tester

Through Pen. Testing, we actually exploit the vulnerabilities - not just look at them and go “oh, that’s a nice Vulnerability”. Vulnerability assessments can be carried out in a number of places:

Human through human errors, insider threats, social engineering, indifference

Application Functions, storage, memory management, input validation

Host Access Control, memory, malware, backdoor, OS / Kernel

Network Map the network, services, leaks, intercept traffic

1.3 Stages of Penetration Testing

1.3.1 Information Gathering

In the *information gathering* phase, the hacker strives to obtain as much information on the target service / device / company as possible. This could be done through passive methods such as:

- Open Source Intelligence
- Google Dorking
- Social Media Analysis
- DNS Enumeration

Passive methods are methods where as much information as possible is gathered without establishing contact between the pen. tester and the target.

Alternatively, active information gathering techniques (where the pen. tester establishes contact with the target) could be used:

- Open Ports and Service Enumeration
- Directory Scanning
- Common Weaknesses

1.3.2 Exploitation

After gathering information on the target, then next stage is to exploit and vulnerabilities which have been identified. Commonly this can be done through social engineering & fishing, where illiterate users will handover compromising details unknowingly or through known exploits (such as the wp-google-maps exploit explored during the lecture and practical). The decision as to which exploit to use is quite complex and takes a number of factors into consideration including:

- Reliability
- Complexity
- Detection
- Impact
- Environment
- Cost

1.3.3 Post Exploitation

After an exploit has been exploited, the next stage is to see what can be done with the access gained. Commonly this will be to attempt *privilege escalation* through which a basic user account's permissions are escalated to be higher; or to maintain access - which could be done through keeping a SSH session alive or creating a start up service to open a backdoor. The pen. tester will need to cover their tracks, done through editing logs which in linux are found in the `/var/log/` directory. Finally, the pen. tester will write a report detailing what they have found, how they exploited it and give recommendations on what can be done to close the exploit.


1.4 Defences


There are a number of defences which can be used against hacking:

- Firewalls
- Intrusion Detection Systems
- Intrusion Prevention Systems
- Regular Testing
- Effective Policies
- Regular Effective Training
- Patch Management
- Threat Intelligence

Page 2

Lecture - Information Gathering

 2024-01-29

 0900

 Tobi

2.1 Active vs Passive


There are two approaches which can be taken to Information Gathering: Active and Passive. The difference between the two is about the contact the hacker has with the target; where in active information gathering - the hacker has direct contact with the target (which will lead to potential discovery by the target of the hacker) whereas in passive information gathering - the hacker does not make direct contact with the target (which is less likely to lead to discovery).


Active information gathering makes use of probing the network, social engineering, directory & share scanning.

Passive information gathering makes use of: OSINT, search engines, and physical observations. Passive information gathering also includes activities such as DNS enumeration, which can include using whois, IP address scanning and examining associated devices.

Page 3

Lecture - Exploitation

 2024-02-05

 0900

 Tobi

3.1 Vulnerability Scanning

The process of vulnerability scanning for a system is essentially scanning the open networking ports & scanning through common directory names to see if anything is found. The outcome from a port scan is a banner grab, which allows hackers to be able to identify the services running and therefore identify any potential vulnerabilities.

The outcome from directory scanning would be revealing a hidden file which hasn't been indexed or finding a misconfiguration. Either of these could lead to identifying further information about the target system that may expose a backdoor.

Listed below are some types of vulnerability scans:

Network-Based Scans identify possible network security attacks and vulnerable systems on networks

Host-Based Scans finds vulnerabilities in workstations, servers, or other network hosts and provides visibility into configuration settings and patch history

Wireless Scans identifies rogue access points and validate that a company's network is securely configured

Application Scans detects known software vulnerabilities and mis-configurations in network or web apps

Database Scans identifies the weak points in a database

Ideally corporate environments will have a patch management system, end user protection, intrusion detection systems and intrusion prevention system. Unfortunately, in reality - this is commonly not the case, especially where there is little-to-no investment in IT infrastructure.

3.2 The Trusted Input Problem

A problem which has plagued digital services for as long as they have existed, is the requirement for users to be able to input data into them. Users cannot be trusted with a text-input field and as such we have to treat everything users input as suspicious until we can prove that it isn't.

Software usually relies on interactions with users and other applications, and data & code are executed in the same location. This can lead to: SQL injection, Stack Buffer Overflow, Shell Code Injection, File Inclusion or XSS attacks.

We can identify vulnerabilities in a number of places: where the data is stored, where the data is processed or where the data is transmitted. The first stage to identifying a vulnerability is to find the injection point - which could be done through an information gathering technique.

Page 4

Lecture - Social Engineering and Decoys

📅 2024-02-12

🕒 0900

🎓 Tobi

4.1 Social Engineering

There are two basic concepts to social engineering:

Deception the art of ‘convincing’ people to disclose sensitive / restricted / confidential information (i.e. to disclose their password)

Manipulation undertaking actions that can result in the above (i.e. installing malware or following links in phishing emails)

There are four phases to social engineering.

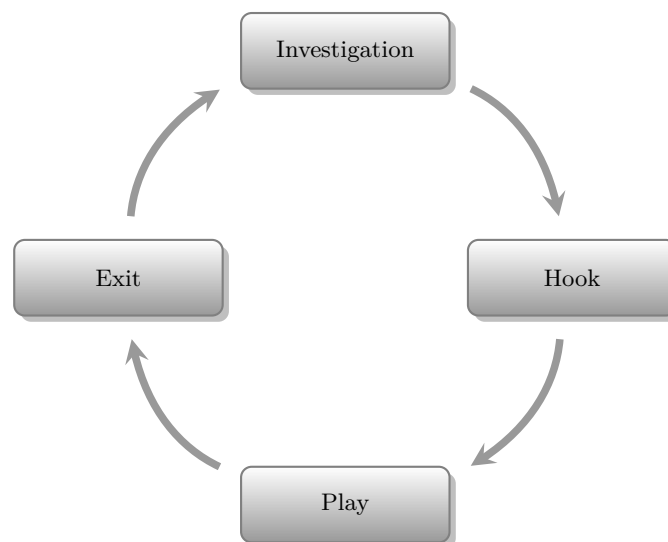


Figure 4.1: Social Engineering Cycle

The first phase, investigation, is primarily concerned with preparing for the attack. This stage involves identifying the victims, gathering background information on them then selecting the attack method. This is highly critical as for the phishing attack to be effective, the user must be convinced that the attack is a real-looking email.

The second stage is deceiving the victim(s) to gain a foothold. This stage involves engaging the target where they are then spun a story. The attacker will ensure that they have control over the interaction, to ensure that they are able to manipulate the target into disclosing the information which the

attacker wants.

The third stage is obtaining the information over a period of time. This stage is where the attacker expands their foothold over the target, then they will execute the attack. The attacker is then able to disrupt business and / or siphoning data from the target.

The final stage of the cycle is closing the interaction, ideally without arousing suspicion. This is a critical stage, especially where the attacker removes all traces of malware and covers their tracks. Ultimately, they are bringing their charade to a natural end.

4.1.1 Why is it Possible to Social Engineer

Humans are vulnerable! Humans have far too much trust in each other, especially where there is an ounce of truth in what the attacker says. For example, assuming an attacker was trying to gain access to an Instagram account - the target has an Instagram account and therefore would be likely to open an email from “Instagram” and following instructions in it.

4.1.2 Examples of Social Engineering Attacks

Spear Phishing Targeting a specific company

Net Phishing Targeting a group of people or random people

Pretexting fabricates a scenario or pretext to steal their victims’ personal information

Blackmail / Intimidation Utilising information / force to compel the victim

Baiting Tricking or enticing the victim to reveal / do something

Tailgating / Piggybacking Involves an unauthorised person following an authorised individual into a restricted area or system

Vishing Using phone calls to scam the victim into divulging information

Smishing Text messages to lure victims into clicking on a malicious link or providing personal information

Impersonation Pretending to be a trusted contact or authority

4.2 Intelligent Password Guessing

Through information gathering, we are able to come up with good guesses as to what the target’s passwords may be. Examples of common passwords include: their date of birth, important dates in their lives, their families date of births. Numerical identifiers (such as dates) may be found in 4 or 6 digit pins. Common wordlists are freely available on the internet, or other active information gathering methods can be used.

However - there are some disadvantages to dictionary attacks. The results are uncertain, there is a change of being detected which could lead to blocking of the attacker / blacklisting of them. It could also lead to a honeypot being found which will lead to the attacker being discovered.

There is also some benefits to trying default passwords of services, a common mistake with configuring new equipment is to not change the admin password from the default - which are commonly the same across the same model.

4.3 Deception Techniques

One of the most commonly used deception techniques is a honeypot. This is a special built device which is designed for attackers to find and attempt to breach. However - when the attacker makes contact with a honeypot, information about the attacker is harvested. Honeypots can manifest as users which are designed to be broken into (honey users); or as entire networks which are vulnerable and designed to be attacked (honey nets).

Another common technique is obfuscation where the the target creates a service, which from the outside looks real, that is designed to deceive the attacker.

It is also common to deceive attackers through switching port numbers, names of plugins or directory names.

4.4 Why Are We Targeted?

- Trust
- Most companies don't properly train (or re-train) employees
- Information / assets are readily available
- Larger companies may well have disparate units / sites with differing policies
- Many organisations still lack policies
- Many organisations' employees aren't aware of the policies when they are in existence

Page 5

Lecture - Privilege Escalation

📅 2024-02-19

🕒 0900

🎓 Tobi

5.1 Introduction

Privilege escalation is the process of changing the user which the attacker has gained access through to either a user of the same access level (horizontal privilege escalation); or alternatively to a user of a higher access level (vertical privilege escalation). Horizontal privilege escalation can sometimes be referred to as 'lateral' privilege escalation - this term is commonly used in a Windows Environment.

5.2 Stages of Privilege Escalation

There are a few stages to privilege escalation, which will all commonly be used in privilege escalation. Ultimately, the hacker is aiming to understand as much about the system as possible - with the aim to find an exploit and use it.

5.2.1 Who Am I?

When access is gained to a system, the attacker will have a role within the system. This will not be obvious to them, so they will want to identify what this role is, what user rights they have and what they can do. They would then want to attempt to spawn a shell as this is much more comfortable for them to use than the TTY shell which they will probably have spawned into.

- The `whoami` command gives the username of the current user.
- The `id` command gives the user and group information for the specified user (which is the current process when unspecified)
- The `sudo -l` gives information about what commands can be run by the current user; it does require their password however.

An example of where Privileges have been badly messed up is in Windows XP where running Command Prompt as an administrator would elevate a basic users privilege.

5.2.2 What Is My Environment?

After identifying who the user is, they can then begin to identify what environment they have landed in. This will include things such as identifying the OS details and the Linux Kernel version. Both of these may be exploitable. The hacker then may identify what programming languages and tooling is available on the system, for example `gcc` or `Python`. The hacker would then also be attempting to identify how files can be uploaded to the system.

- The `uname -a` command will display the kernel version
- The `/etc/os-release` file (which can be displayed using `cat`) contains more information about the operating system, including it's flavour.

5.2.3 How do Users use the System?

The next stage is to identify how users are using the system, as this may lead to some identification of what the system is used for and who else has access to it. It may also be pertinent to identify the history of commands used, any profiles or cached data. This may also lead to identifying if there is a directory service enabled and if it is possible to enumerate other users.

5.2.4 Sensitive Information

From here - it is possible to begin to identify what sensitive information has been left laying around where it shouldn't have. It may be possible to identify usernames / groups; passwords; database files; keys (for example SSH private keys); configuration files / scripts; hidden files; or shared directories / files.

Sensitive files should always be restricted to the admin / root user, and most definitely not made public.

5.2.5 Processes and Services

From identifying what processes and services are running, it is possible to further identify what the device is doing. There may be some processes which are run by root, meaning that if these can be exploited then they will be exploited by root. As we have seen throughout this module, known exploit databases should be checked against all installed software to attempt to identify vulnerable software installed. There may also be hidden files, shared directories or files or configuration files which are stashed somewhere.

5.2.6 File Permissions / Access Control

Through looking at what data the user can read, what they can write to and what they can execute - it is possible to further identify permissions on the system. From reviewing who owns what, it's possible to identify how badly configured the file permissions are. It may be possible to identify 'sticky bits' which are user ownership access right flags and are assigned to files and directories. There may also be SUID (set user ID, where you run as the owner not the user who started it) and GUID (group user ID, where it is run as the group not the user who started it) configured. There may also be other filesystems mounted which can lead to further vulnerable points.

5.2.7 Networking

Through looking at IP addresses, MAC Addresses and available interfaces - it is possible to identify what networks the target device is connected to. It can also be worth looking at DHCP, proxies, the default gateway, routes, DNS entries, etc. As all of this together will help to build out a map of what access this device has to. From this - it may be possible to identify internal firewalls, what services are open, who is communicating with the target. There is a possibility of sniffing packets, ARP poisoning or even port forwarding / tunnelling.

5.2.8 Automating the Process

There's obviously lots of different stages and processes to go through, and tools to use to effectively follow through privilege escalation. However - there are tools which can automate the process for hackers. These tools may not use all the techniques, just some of them and importantly, they might be noisy and make cleaning up difficult.

Page 6

Lecture - Misconfigured File Permissions

📅 2024-02-26

🕒 09:00

🎓 Tobi

Can bring in a single A4 sheet of paper (double sided) to the exam. It must be handwritten.

6.1 Files

Within Linux, everything is a file. We can find files using OSINT (Google Dorking, etc) however locally on a Linux system - we have three commands we can use:

- `find` searches for files in a directory
- `locate` finds files by name, quickly
- `which` locates a command

We are also able to exploit the `PATH` variable by forcing programs to run our version of certain binary files. The command `echo $PATH` shows what files are currently in the path.

6.2 File Permissions

Linux file permissions are split into three categories - Users, Groups and Others. It is possible to control the permissions of reading, writing to, and executing a file for each of these three groups.

The SUID (Set User Identity) bit gives special permission for the user access level and always executes as the user who owns the file. This is a thing to keep an eye out for as it is possible to execute a command as root if these permissions have been incorrectly configured. Similarly, the Set Group Identity allows a file to be executed as the group owner.

The sticky bit restricts file deletion at a directory level.

Page 7

Lecture - Exam Preperation

📅 2024-03-11

🕒 0900

🎓 Tobi

7.1 Exam Preparation

- The exam will cover content from week 1 through 5 (inclusive); there will be some vulnerabilities from week 1 and 2, however.
- The ultimate aim of the exam is to break the machines to Root access, and there will be a number of questions on Moodle which we will need to answer, some of these will be flag-style and some will require research on the machines and providing an answer.
- The ‘playbook’ should contain details on some of the OWASP Top 10
- Some geolocation will be expected in the exam
- Filtered internet access will be provided, we will be provided with a list of sites which we can access
- We will be breaking a webserver. There might be *other* stuff running on it, however it will definitely be a webserver
- The webserver might be running WordPress, it might not.
- We will be provided with the *rockyou* wordlist. If an attack using a wordlist is taking a long time then we should assume that it will not provide us the answer.
- If a hashcat is taking a long time, we should try other things while it is running

Page 8

Lecture - Web Application Attacks (SQL Injection)

📅 2024-04-15

🕒 0900

🎓 Tobi

8.1 The Web Server

The *server* software or hardware satisfies client requests over the World Wide Web. Web Servers will use port 80 (for http traffic) and port 443 (for https traffic). It's primary function is to serve HTML content to users. Web Servers can also host databases, and may support server-side scripting. They may also be used to monitor or perform other activities.

Common examples of web server software includes:

- Apache
- Nginx Server
- Cloudflare Server
- Microsoft IIS
- LiteSpeed
- Google Servers

8.2 Web Applications

A web application works through the client making a HTTP request to the server. The server receives this then processes it and returns a HTTP response. The client may be a web browser, a crawler, or a tool.

Unsurprisingly - it's possible to execute an attack on a web application. On the client side (HTML, CSS, JS, etc) this can result in compromise to users including users' access to the server. On the server side (events which occur on the server, eg PHP, Python, Database, Authentication) this can compromise the server.

8.3 Database

A relational database is an extremely common method used to store data, they are commonly used on web servers to store database for web applications. A relational database stores records in tables, which are linked together. Users which can access the database have permissions applied to them - these can either be role based or applied to individual users.

8.3.1 SQL Injection Attacks

An SQL Injection Attack is an attack in which you trick the interpreter into executing unintended commands or accessing data by sending untrusted data as part of a command or query.

Commonly - the attack vector is any source for inputting data, environment variables, parameters, users, GET and POST requests.

The steps to perform a SQL injection attack are outlined below:

1. Find all sources for sending data
2. Understand how that data is interpreted and processed
3. Fuzz (Manually, Burp Suite - Intruder, SQLMap, WFUZZ)

There are a number of different types of SQL Injection Attacks:

In-band SQLi Data is retrieved using the same channel that is used to inject the SQL code. This is simple, union, or Error-based.

Inferential SQLi (Blind SQLi) Data is not transferred via the web application and the attacker is unable to see the result of an attack. This is time-based.

Out-of-band SQLi Data is transferred using a different channel, such as an email.

8.3.2 SQL Injection In Practice

In the first lab, we saw that *WP Google Maps* was vulnerable to SQL Injection - through the WP Rest Route. As we can see in the URL below, the `fields` value is `id` - this is where we can add our SQL Injection Command:

```
http://loan.atlas.local/?rest_route=/wpgmza/v1/markers&filter={}&fields=id
```

There are a few options for how we test that the URL is vulnerable to SQL Injection

- `*-` is a catch-all situation. This will attempt to retrieve all
- `1=1` The answer should be `true` and return 1
- `1=0` The answer should be `false` and return 0

An alternative method is to use *Union Based SQL Injection* - where the SQL UNION operator (allows two SELECT statements to be concatenated together) is appended to the command the server executes. in practice, the threat actor may inset a string like so to the vulnerable input field:

```
' UNION SELECT username, password FROM users--
```

This would then get added into the Server's SQL code like so:

```
SELECT name, description FROM products WHERE category = 'Gifts  
' UNION SELECT username, password FROM users--
```

NB: the injected SQL is shown on the second line, in reality this would be a single string which gets executed.

The *time-blind SQL injection* technique is called “blind” because the attacker cannot see information from the database directly. Rather it relies on the response behaviour of the server to reconstruct the database information. This may be, for example, instructing the SQL Server to wait for 10 seconds causing the browser to hang, awaiting the page to complete loading - therefore indicating that the attack has worked. There is a subtype of Bind SQLi where the query's success is determined by the

database server's response time.

These attacks will work by the attacker sending SQL queries to the database, which if true, cause the database to delay the response. The time delay helps the attacker guess if the result of the query is true or false. For example:

```
IF (SELECT user FROM users WHERE username='admin' AND SLEEP(10)) --
```

These can be automated using SQLMAP and WFUZZ.

8.4 Injection Attack Prevention

To protect against injection attacks - developers need to validate all input, filtering out potential attacks in a process called *sanitization*. Developers can also utilise parametrised queries, stored procedures and output escaping. It is also possible to use whitelist server-side input validation, as well as using Database controls like LIMIT to prevent mass disclosure. It is also possible to use Web Application Firewalls, IDS / IPS to filter out traffic containing SQL Injection before it reaches the server.

However - there are a number of ways in which detection tools can be bypassed:

- Fuzzing
- Encoding
- Encryption
- Internal Components

Page 9

Lecture - Introduction to Binary Exploitation

📅 2024-04-22

🕒 0900

🎓 Tobin

9.1 Trusted Input

As we have already seen, anywhere where users of an application or service can enter text - is potentially vulnerable. Depending on the type of input field, it may be vulnerable to SQL injection or to command injection.

9.2 Finding Files

When we are doing exploitation, we first have to do some information gathering. A key step of this is to work out what files exist, what permissions they have and therefore if any are vulnerable.

To find files with the SUID bit set:

```
find / -perm -4000 or find / -perm -u=s -type f
```

To find files with the SGID bit set:

```
find / -perm -2000 or find / -perm -g=s -type f
```

To find files with the SUID and SGID bit set:

```
find / -perm -6000 or find / -perm -u+g=s -type f
```

See Misconfigured File Permissions lecture for more information on file permissions and finding files.

9.3 Binary Files

Once we have compiled a file, it gets turned into binary code - this will either be in 32 bit or 64 bit. Under Windows, compiled files will be in the **exe** format and under Linux, they're in **elf** format.

9.3.1 Loading to Memory

When the code is prepared to be executed, it is loaded into Memory. Registers will get set with values which pertain to the execution of the code. Registers are quickly accessible locations on the computer's processor. Memory will contain the program code. The stack will be ready to accept values from registers and from the CPU during execution.

Registers can either be 8-bit, 32-bit or 64-bit. There are a few different types which are used for different purposes:

- General registers store data, pointers or indexes in Memory
- Control registers are used to control the CPU
- Segment registers

There are a number of different types of data register - these each have a specified purpose of storing a specific data item during execution:

- Accumulator stores arithmetic instructions
- Base Register stores indexed addressing
- Count Registers stores counts in iterative operations

Pointer registers contain pointers to addresses in memory:

- Instruction Pointer stores the offset address of the next instruction to be executed
- Stack pointer provides offset value within the program stack
- Base Pointer references a parameter within a subroutine

9.3.2 Buffer Overflow

A buffer, or data buffer, is an area of physical memory storage to temporarily store data. A challenge is experienced when the data entered into the buffer extends beyond the intended location. This will cause the program to crash and can be used to run unintended functions or to run inputted code.

There are two different types:

Stack-Based overflows a buffer onto the call stack

Heap-based attacks target data in the open memory pool known as the heap

The stack is a portion of memory which is used to save temporary data; including function calls, local variables, and function parameters.

9.3.3 Vulnerable Functions

There are a number of vulnerable functions:

- `gets`
- `strcpy`
- `scanf`
- And more!

These C functions are potentially vulnerable to attacks.

9.4 Defences

There are a number of defences which can be used against Binary Exploitation:

- Address Space Layout Randomization (ASLR)
- Data Execution Protection
- Structured Exception Handling

- Patch device / software
- Safe Programming
- Validating Data
- Least Privilege