Exploring TCP Port Vulnerabilities in Metasploitable2 Using Kali Linux
(Continued)

by

Vibhuti Naik
Intern ID - 2046

Exploring TCP Port Vulnerabilities in Metasploitable2 Using Kali Linux

Login Port 513

Description:

The login service, operating on Port 513 (TCP), is part of the legacy "r-commands" suite (specifically rlogin) used for remote terminal sessions on Unix and BSD-based systems. Much like its counterparts (rexec and rsh), it was designed for convenience in trusted environments, allowing users to log into remote hosts without a password if their client machine was listed in the server's .rhosts or /etc/hosts.equiv files. While it provides a terminal interface similar to Telnet or SSH, it lacks any form of encryption. This means that session data, is transmitted across the network in cleartext, making it inherently insecure for modern networking.

Impact:

- **Eavesdropping and Sniffing:** Since Port 513 does not encrypt traffic, an attacker can use a network sniffer to capture everything typed in the session, including usernames and passwords if prompted.
- **IP Address Spoofing:** Because the protocol often relies on "host-based authentication" (trusting an IP address), an attacker can forge their IP address to gain unauthorized access to the server without needing a password.
- **Session Hijacking:** Attackers can intercept an active rlogin stream, allowing them to take over the session or inject malicious commands into the remote terminal.
- **Privilege Escalation:** If an attacker spoofs a trusted administrative host, they can gain root-level access to the target system instantly.

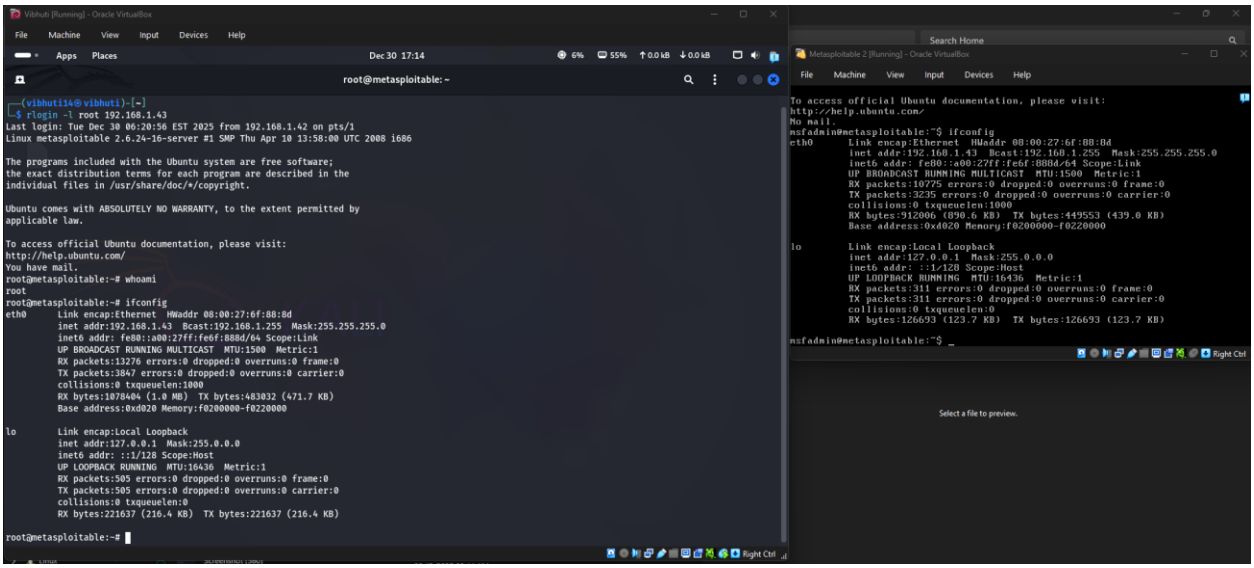
Severity: Critical

Remedial:

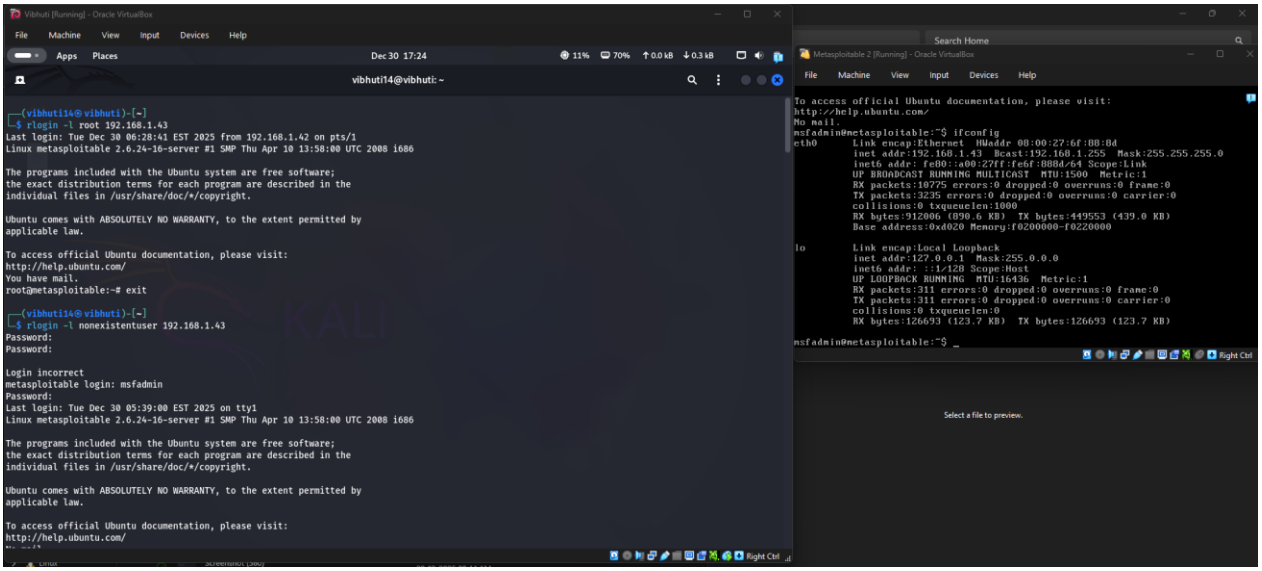
- **Disable the Service:** Shut down the rlogind daemon immediately. This is usually done by disabling the service in inetd or xinetd, or by running: systemctl disable rlogin.socket
- **Migrate to SSH (Port 22):** Replace rlogin with Secure Shell (SSH). SSH provides the same remote login capabilities but protects the session with robust encryption and cryptographic authentication.
- **Firewall Blocking:** Block all inbound and outbound traffic on Port 513 (TCP) at both the network perimeter and on individual host firewalls.
- **Clean Up Trust Files:** Search for and delete any .rhosts (usually in user home directories) or /etc/hosts.equiv files to prevent any remaining legacy services from using insecure trust relationships.

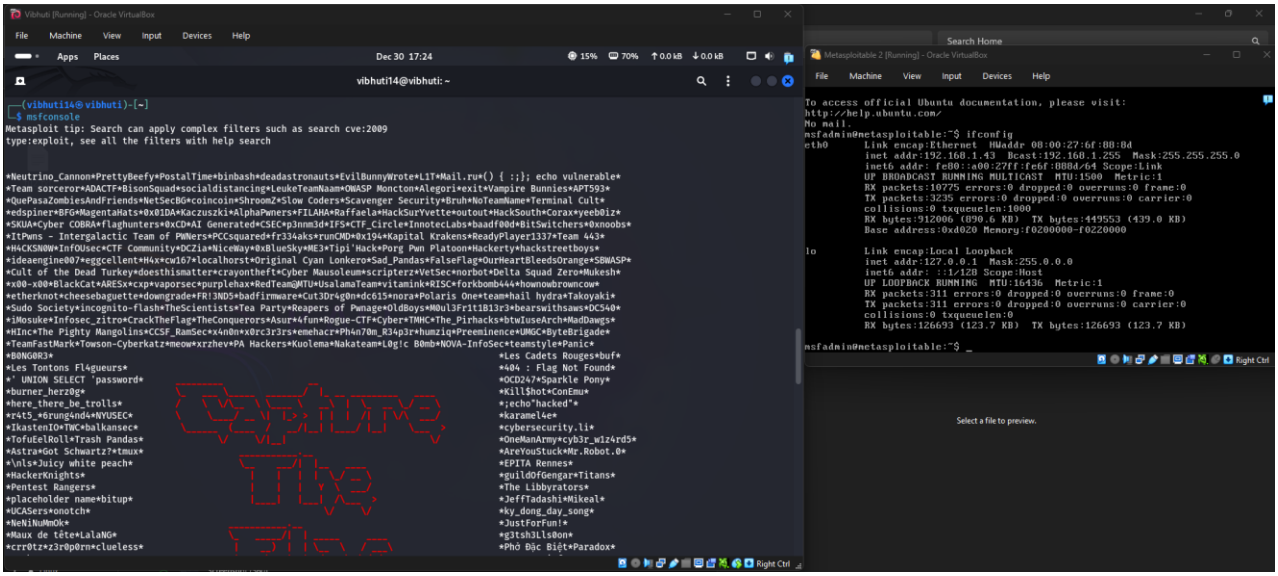
POC:

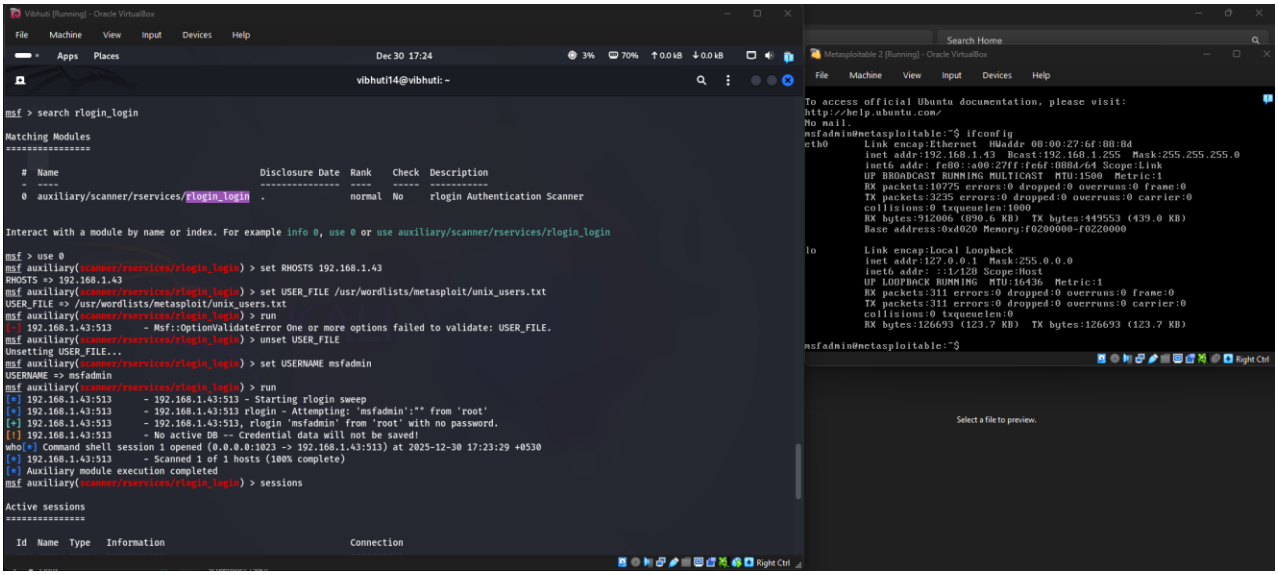
Method 1. Exploiting Weak Trust Relationships ($rhosts$)

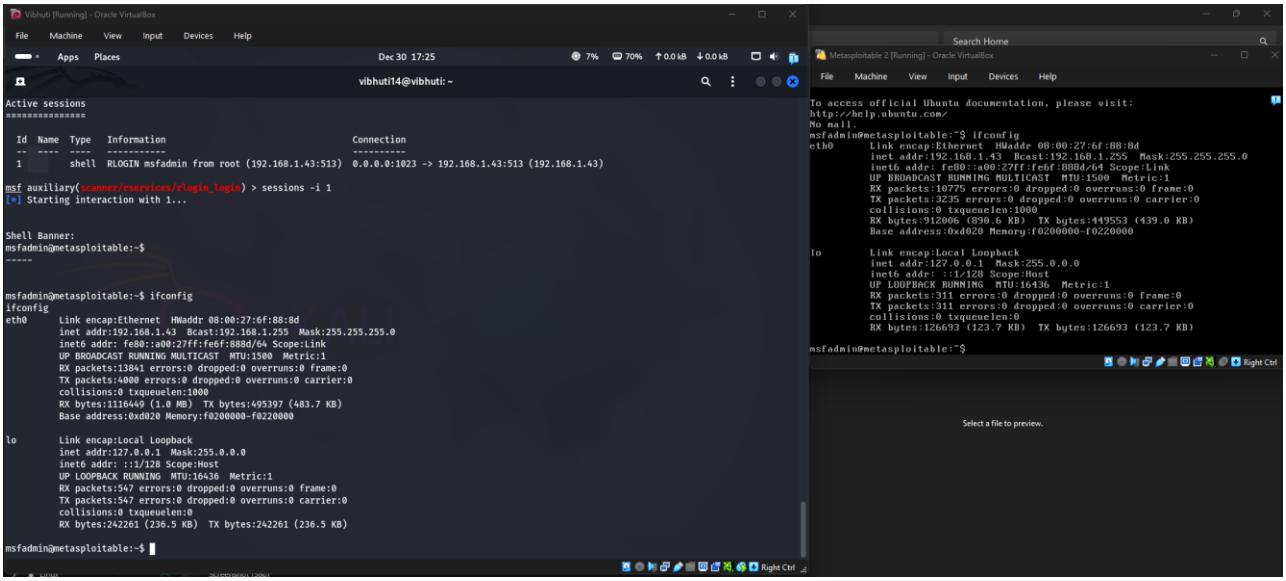


Method 2. Information Leakage and User Enumeration

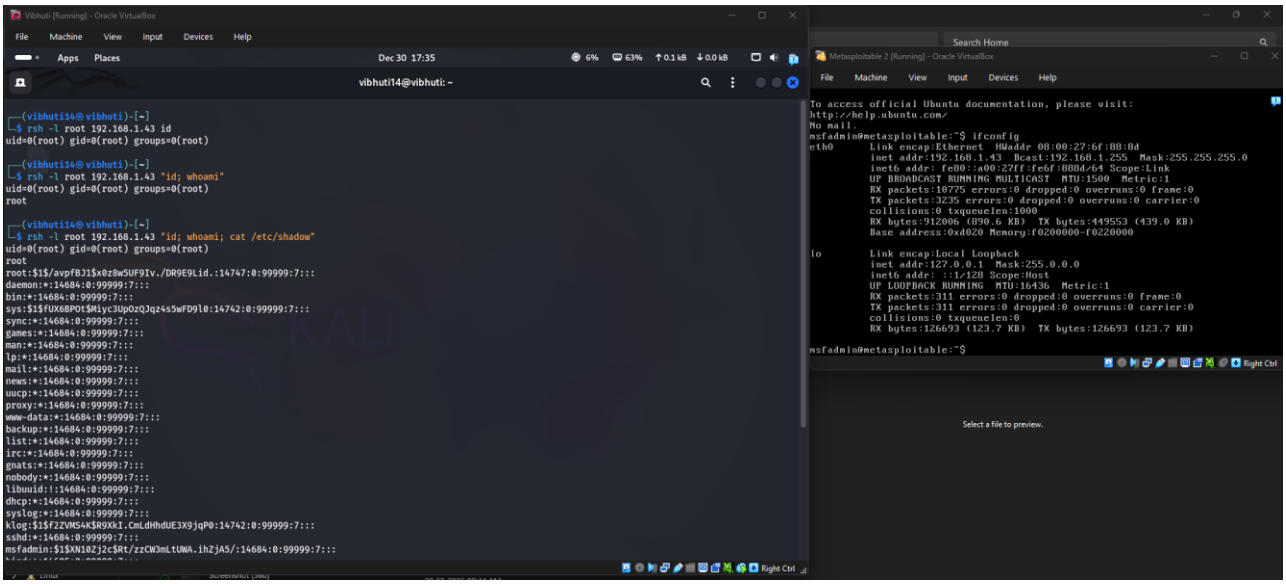








Method 3. Port 514: Remote Shell (rsh) - Trust Exploitation



tcpwrapped Port 514

Description:

Port 514 is traditionally associated with two distinct services depending on the protocol: rsh (Remote Shell) over TCP and Syslog over UDP. When a port scan returns a status of "tcpwrapped," it specifically refers to the TCP service (rsh). This label indicates that the service is protected by a software layer called TCP Wrappers (tcpd). The "tcpwrapped" result means that while the port is technically open and a connection was established, the server immediately closed it because the client's IP address was not authorized in the server's access control files (/etc/hosts.allow or /etc/hosts.deny).

Impact:

- **Information Leakage:** Even if "tcpwrapped," the presence of the port indicates the system is likely running legacy services, marking it as a target for further "r-command" exploits.
- **Credential Sniffing:** If the service is successfully accessed, all session data is sent in cleartext, allowing attackers to harvest passwords and sensitive command outputs.
- **IP Spoofing:** Since rsh often relies on IP-based trust via .rhosts files, an attacker can spoof a trusted IP to bypass authentication and execute remote commands.
- **Syslog Interference:** If Port 514 UDP is exposed, attackers can flood the server with fake log entries, masking their actual malicious activity or filling up disk space (Log Exhaustion).

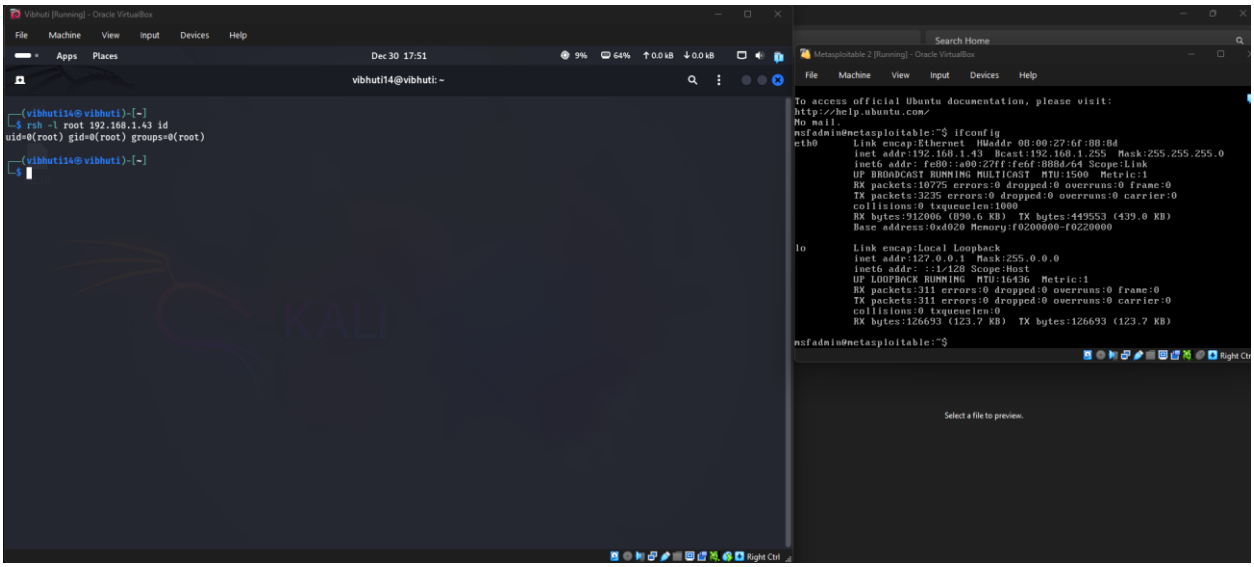
Severity: Critical

Remedial:

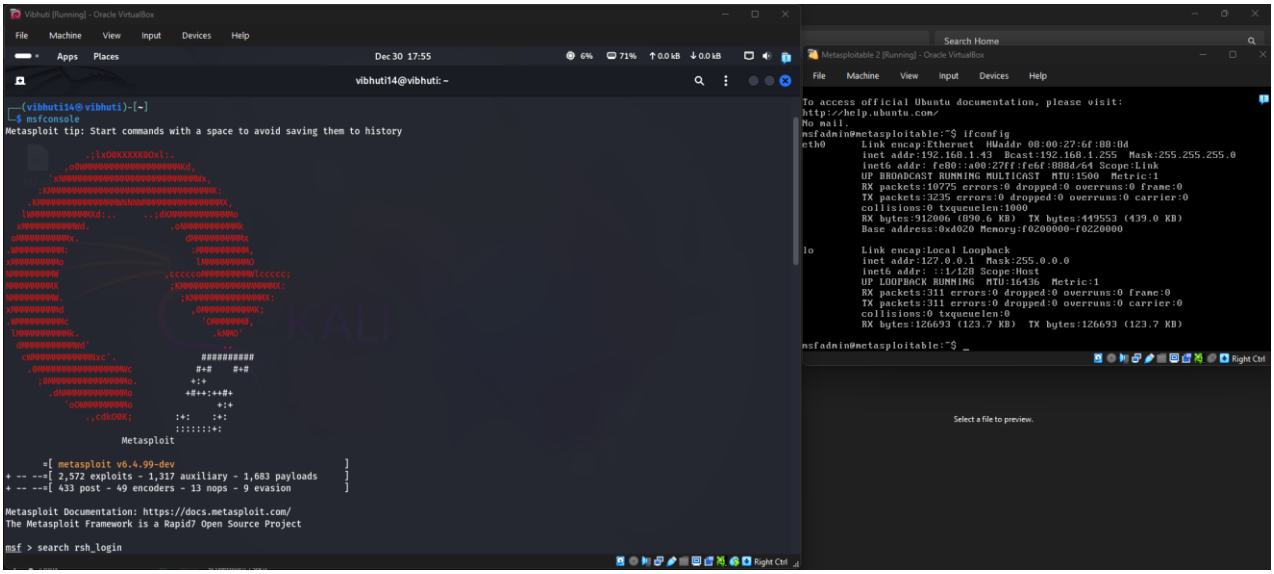
- **Disable rsh/rstatd:** Stop and disable the rsh or shell services immediately. On modern Linux systems, check inetd, xinetd, or systemd: systemctl disable rsh.socket
- **Migrate to SSH (Port 22):** Secure Shell is the modern, encrypted standard for remote command execution. Ensure all automated scripts use SSH keys instead of .rhosts files.
- **Standardize Syslog (Port 514 UDP):** If you are using Port 514 for logging, migrate to Syslog-over-TLS (Port 6514) to ensure log data is encrypted during transit.
- **Restrict via Firewall:** If the service must exist for legacy reasons, use a hardware firewall to restrict Port 514 access to specific, isolated management IPs.

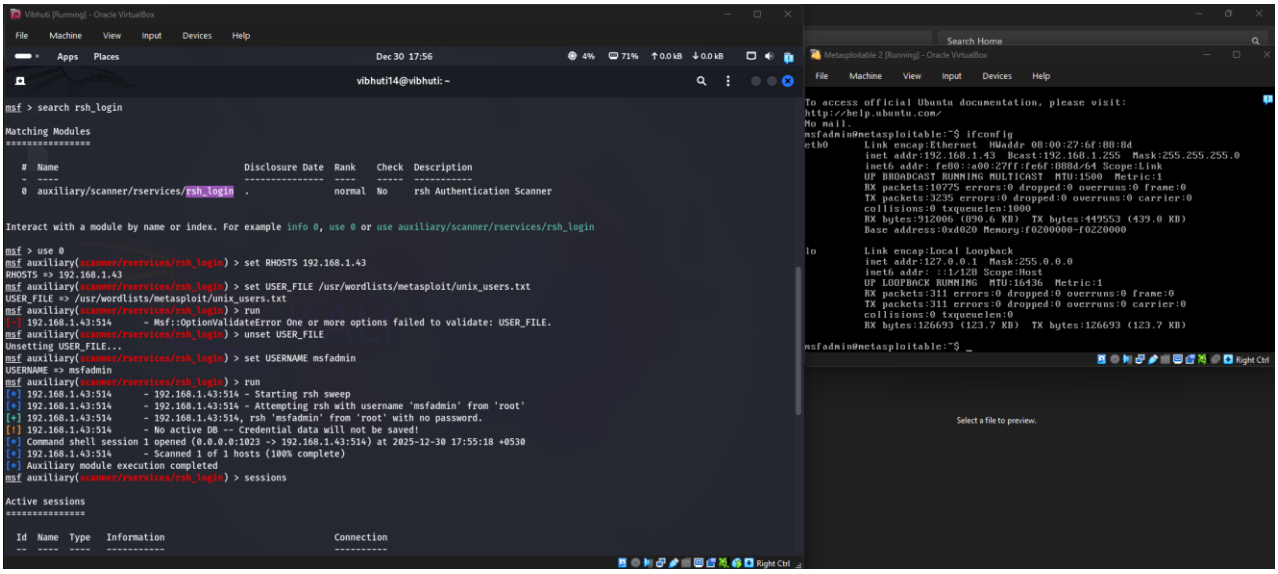
POC:

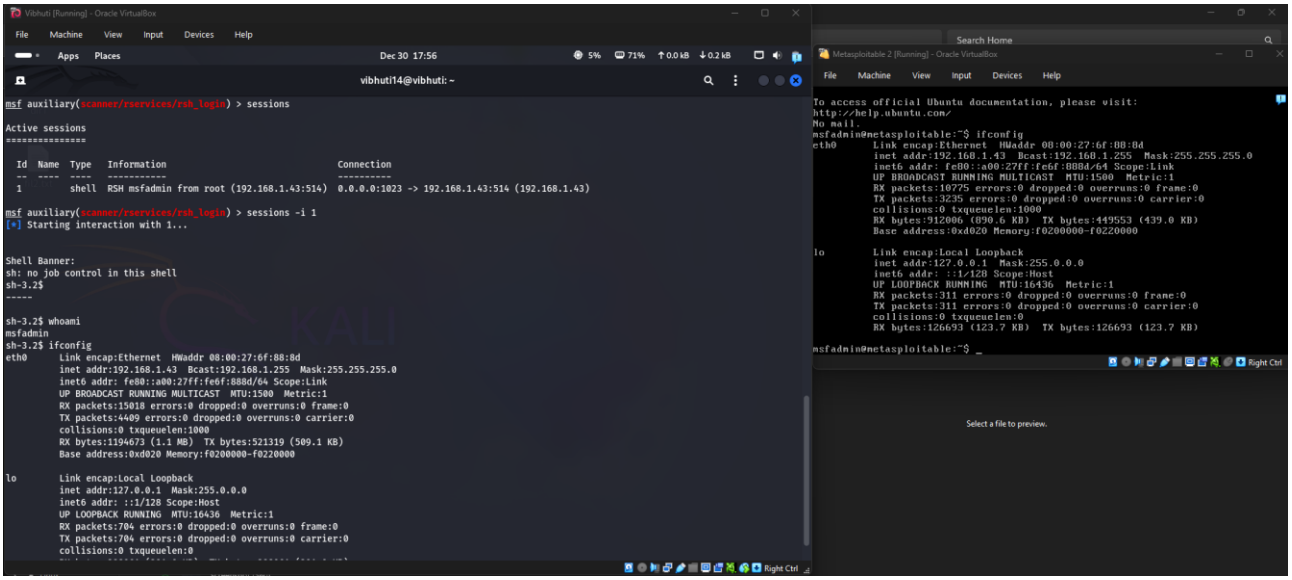
Method 1. The "rsh" Trust Exploitation (Passwordless Root)



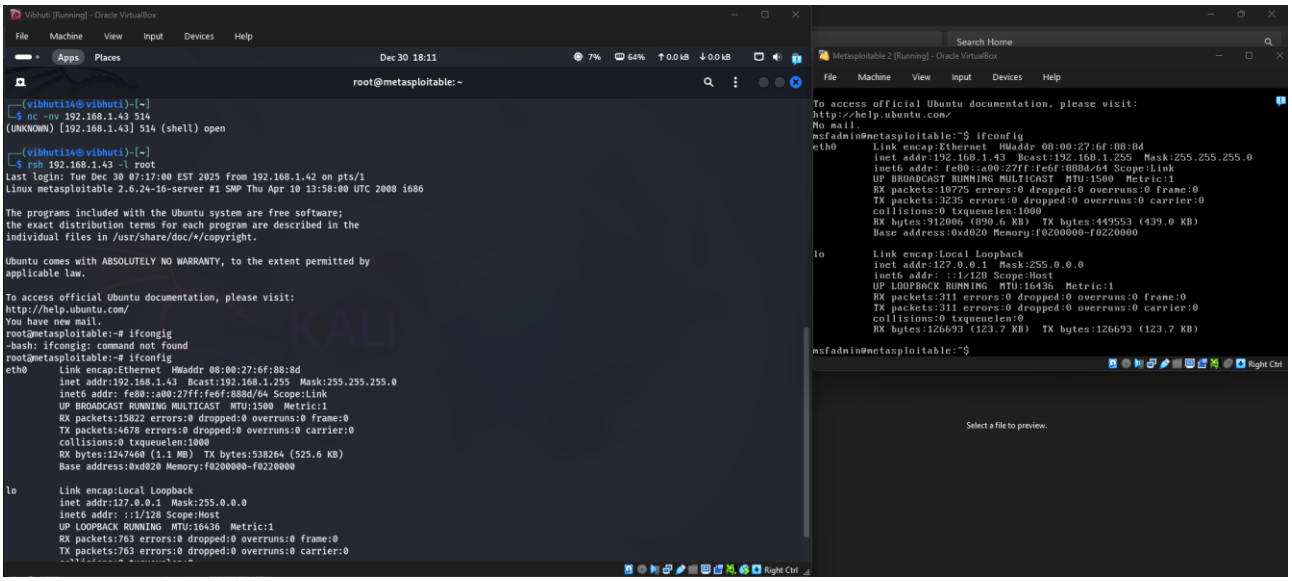
Method 2. Metasploit RSH Login Scanner







Method 3. Using Netcat (nc) to simulate the rshell connection



java-rmi Port 1099

Description:

The Java Remote Method Invocation (RMI) Registry typically operates on Port 1099. It serves as a centralized lookup service—similar to a telephone directory—for Java applications. When a Java server-side object is created, it registers a name and a reference on Port 1099; a remote client then queries this port to find the object and invoke its methods as if they were running locally. Because RMI relies heavily on Java Serialization to pass objects between the client and the server, Port 1099 is inherently sensitive. By default, older versions of Java RMI do not require authentication or encryption.

Impact:

- **Remote Code Execution (RCE):** This is the most critical risk. An attacker can send a specially crafted, malicious serialized object to the registry. When the server attempts to "deserialized" this object, it can trigger the execution of arbitrary commands on the host system.
- **Insecure Deserialization:** Java's native serialization process is often vulnerable to "gadget chains" (using existing libraries like Commons-Collections) that allow attackers to bypass security layers.
- **Information Disclosure:** Unrestricted access to Port 1099 allows attackers to list all registered RMI objects, providing a map of the application's internal structure and available services.
- **Unauthorized Method Invocation:** If the RMI objects themselves are not properly protected, an attacker can invoke administrative or sensitive functions directly via the registry reference.

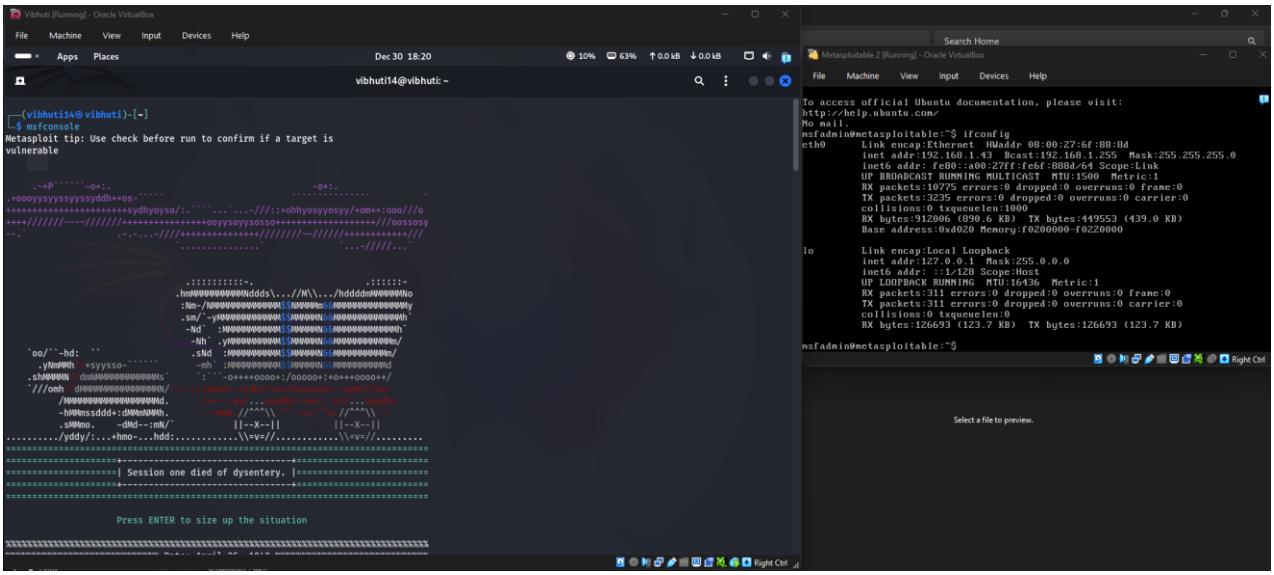
Severity: Critical

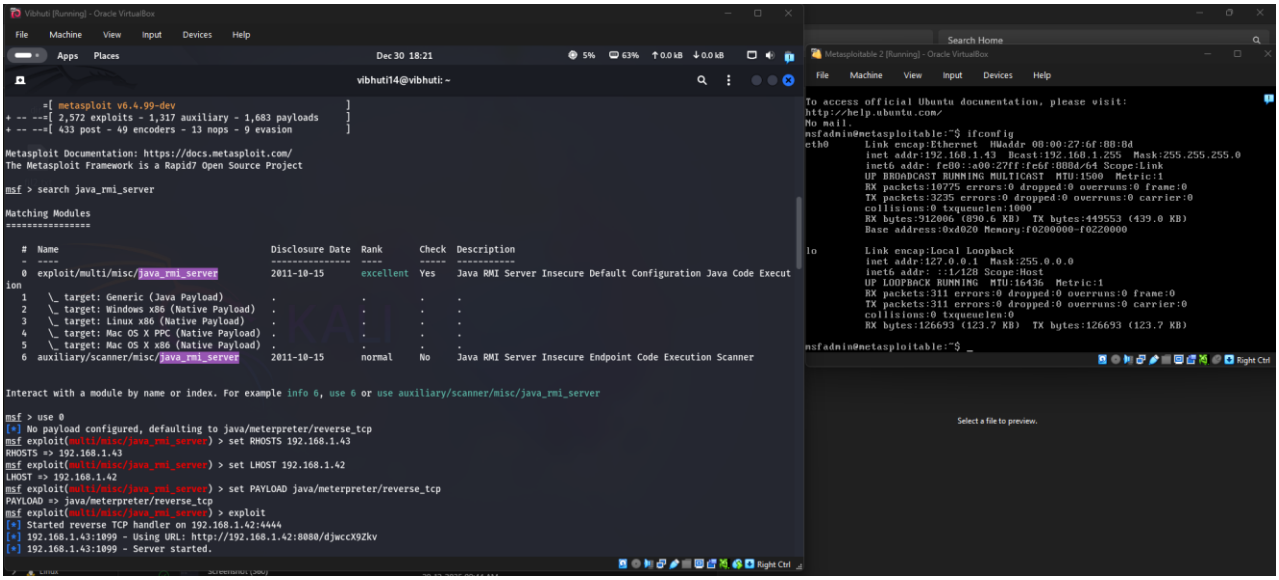
Remedial:

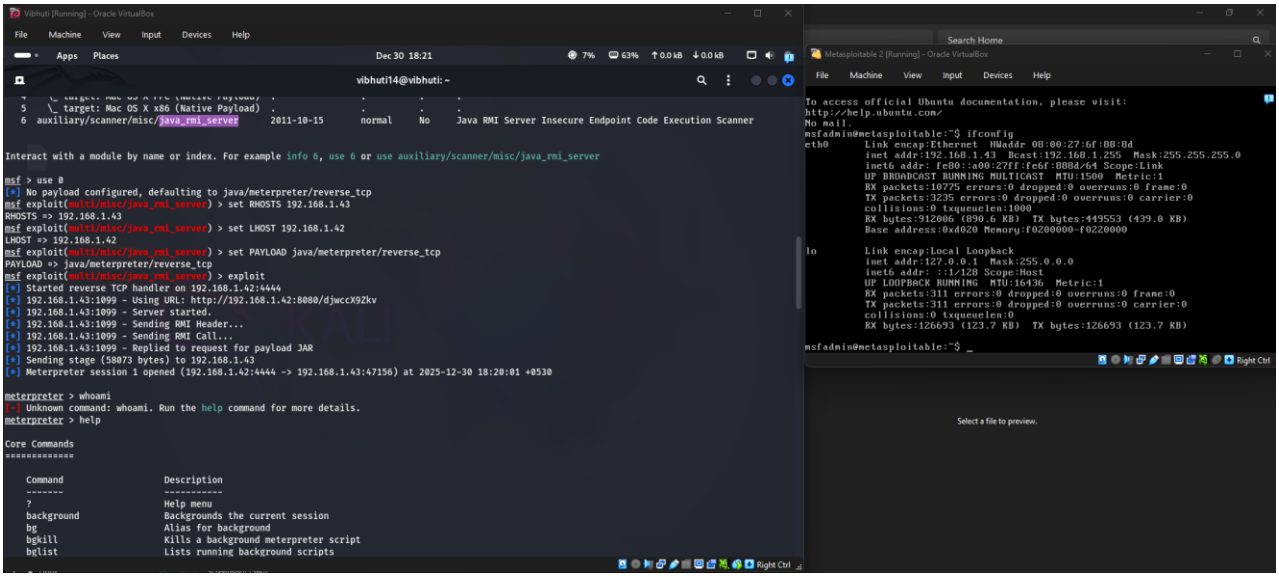
- **Block Port 1099 at the Firewall:** Ensure that Port 1099 is never accessible from the public internet. Access should be restricted strictly to trusted internal application servers via a local firewall (IPTables/UFW) or Security Groups.
- **Enable JEP 290 Filtering:** Use the serialization filtering features introduced in modern Java versions to whitelist only safe classes, preventing the deserialization of malicious gadget chains.
- **Implement SSL/TLS (RMI over SSL):** Configure the RMI Registry to require SSL/TLS for all communications to ensure data is encrypted and to prevent Man-in-the-Middle (MitM) attacks.
- **Require Authentication:** Use a custom RMIClientSocketFactory and RMIServerSocketFactory to implement strong authentication before a client can query the registry.

POC:

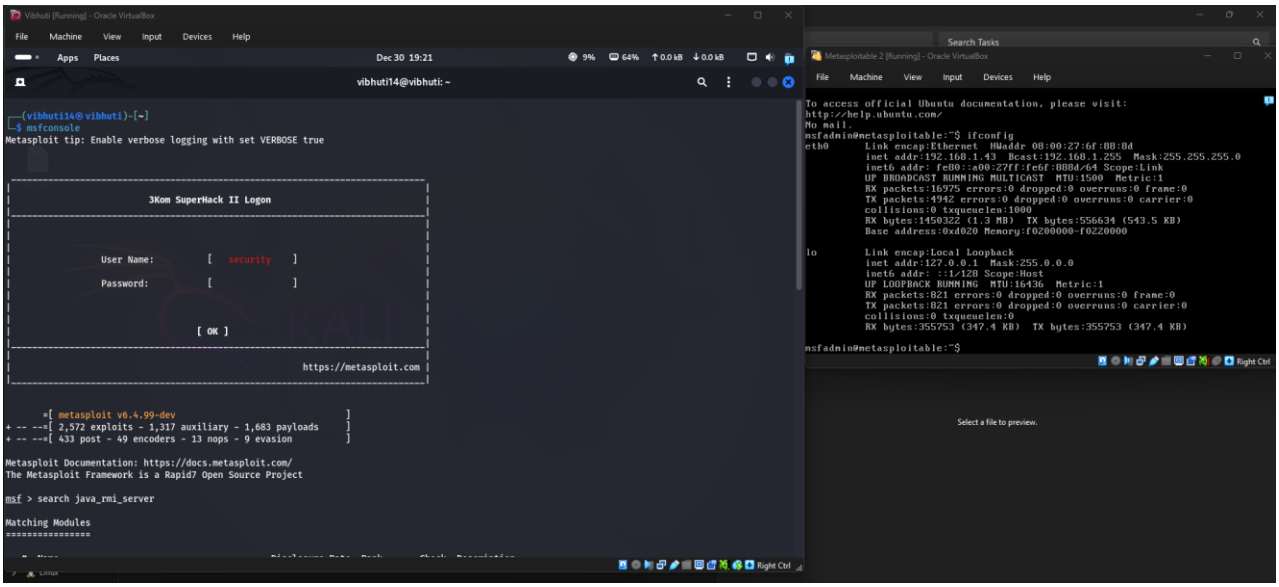
Method 1. Metasploit RMI Default Managed Object Exploit

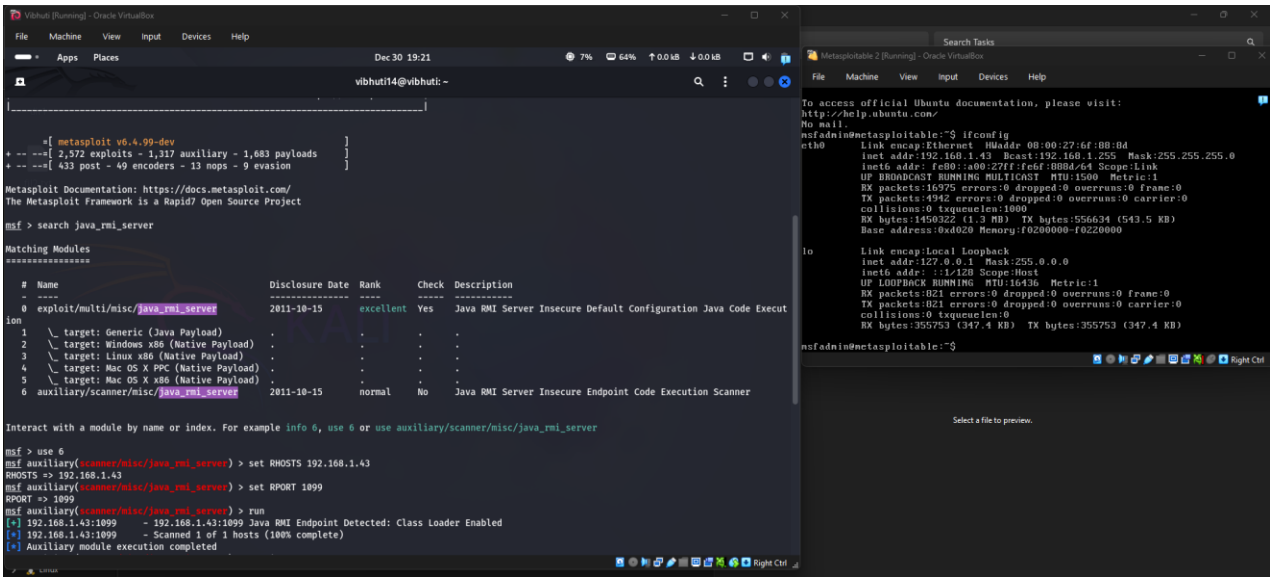




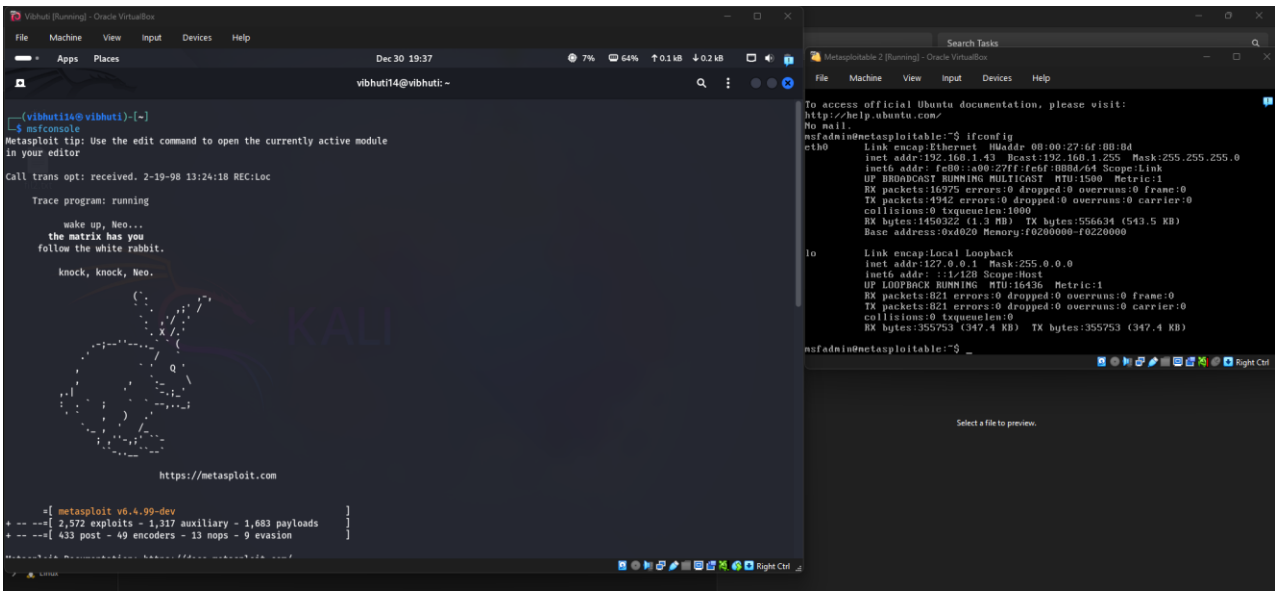


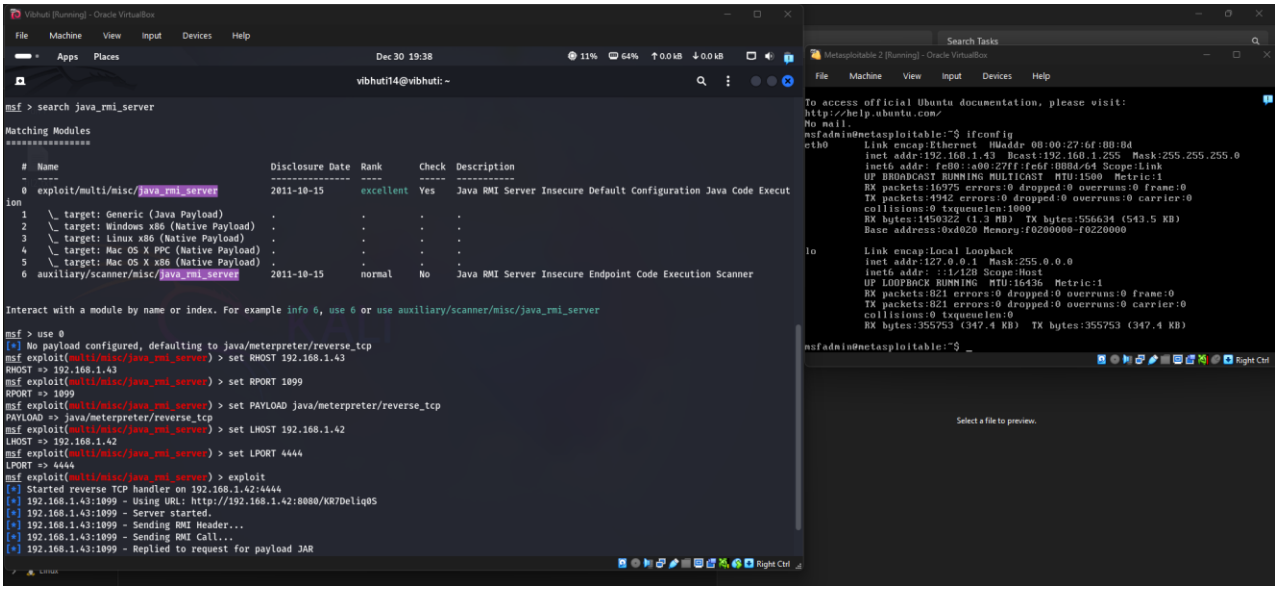
Method 2. Using Metasploit Auxiliary Modules for Vulnerability Scanning

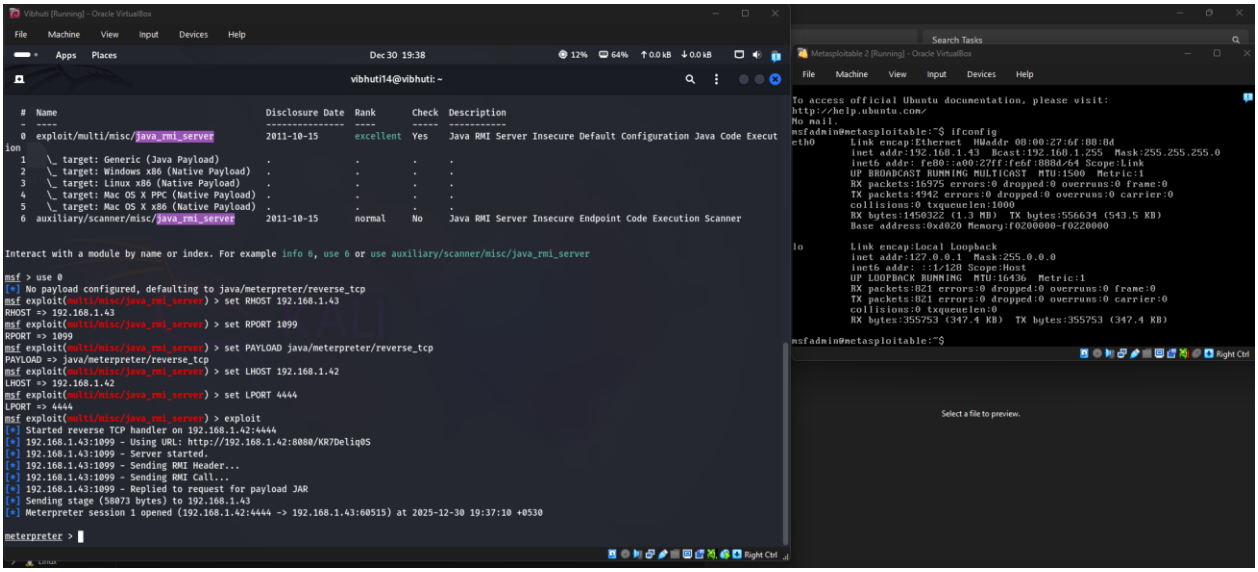




Method 3. Gaining a Root Shell using Metasploit Exploit Module







bindshell Port 1524

Description:

Port 1524 is historically associated with the Ingres lock daemon, but in the context of cybersecurity, it is most infamous as the default port for a bindshell. A bindshell is a malicious backdoor created when an attacker successfully exploits a system and executes a listener—usually a compromised version of /bin/sh or cmd.exe—that "binds" to a specific port. When Port 1524 is open, it typically means the shell is sitting and waiting for an incoming connection. Once a remote user connects to this port via a tool like Telnet or Netcat, they are immediately granted an interactive command-line interface with the privileges of the process that started the shell, often without any requirement for a username or password.

Impact:

- **Full System Takeover:** Since a bindshell typically provides a root or administrator-level shell, the attacker gains total control over the operating system, including the ability to read, modify, or delete any file.
- **Bypassing Authentication:** Bindshells are specifically designed to circumvent standard security controls; they do not use the system's legitimate login mechanisms, making them invisible to standard access logs.
- **Malware Installation:** The shell allows for the easy download and execution of further payloads, such as ransomware, keyloggers, or persistent rootkits.

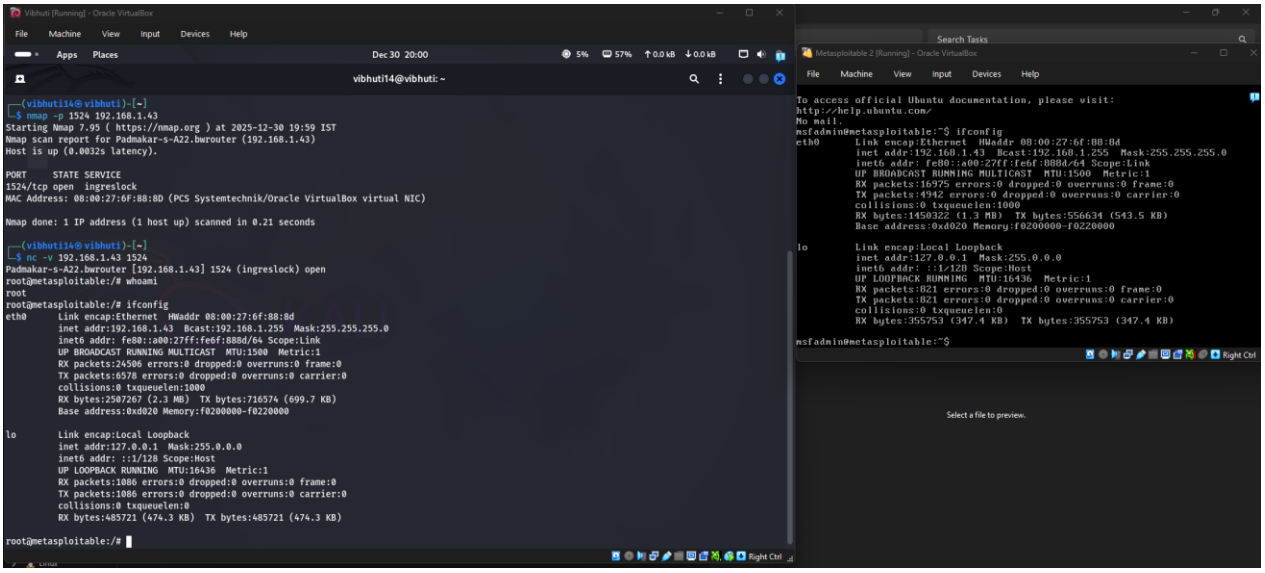
Severity: Critical

Remedial:

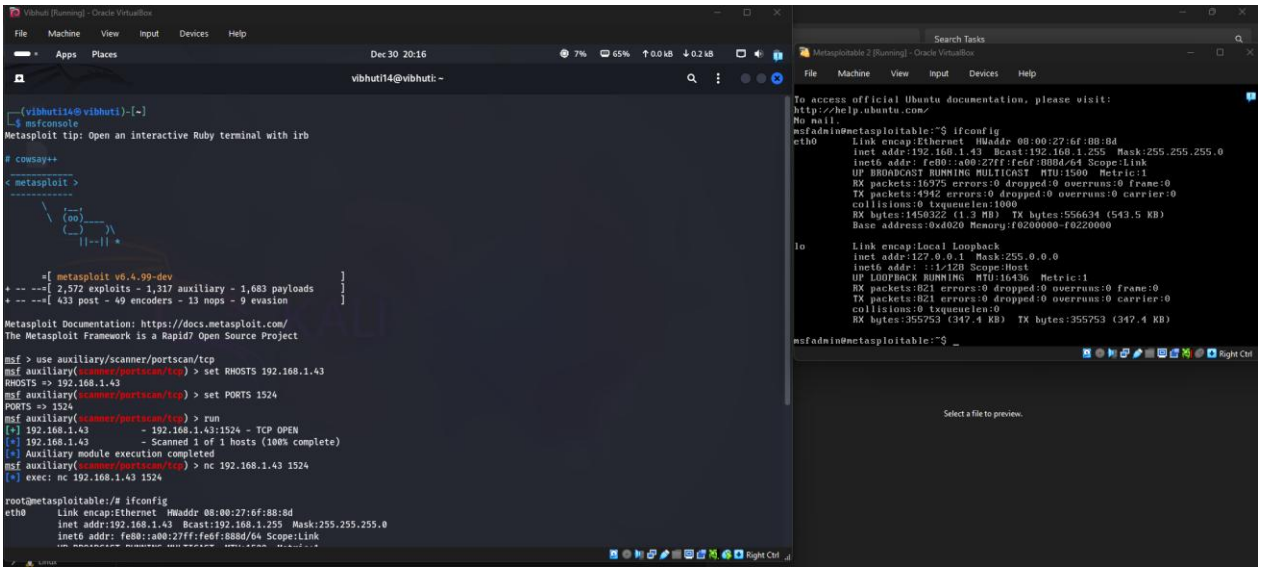
- **Immediate Isolation:** Disconnect the affected system from the network immediately to prevent the attacker from executing further commands or moving laterally to other servers.
- **Identify the Malicious Process:** Use commands like netstat -antp | grep 1524 or lsof -i :1524 to identify the Process ID (PID) associated with the port, then investigate the executable path.
- **Wipe and Reinstall:** Because it is difficult to guarantee that an attacker hasn't installed other hidden rootkits or backdoors, the most secure remediation is to wipe the server and rebuild it from a known-secure backup.
- **Patch the Entry Point:** Identify the original vulnerability that allowed the bindshell to be installed (such as an old version of Metasploit's proftpd exploit which used this port) and apply the necessary security patches.

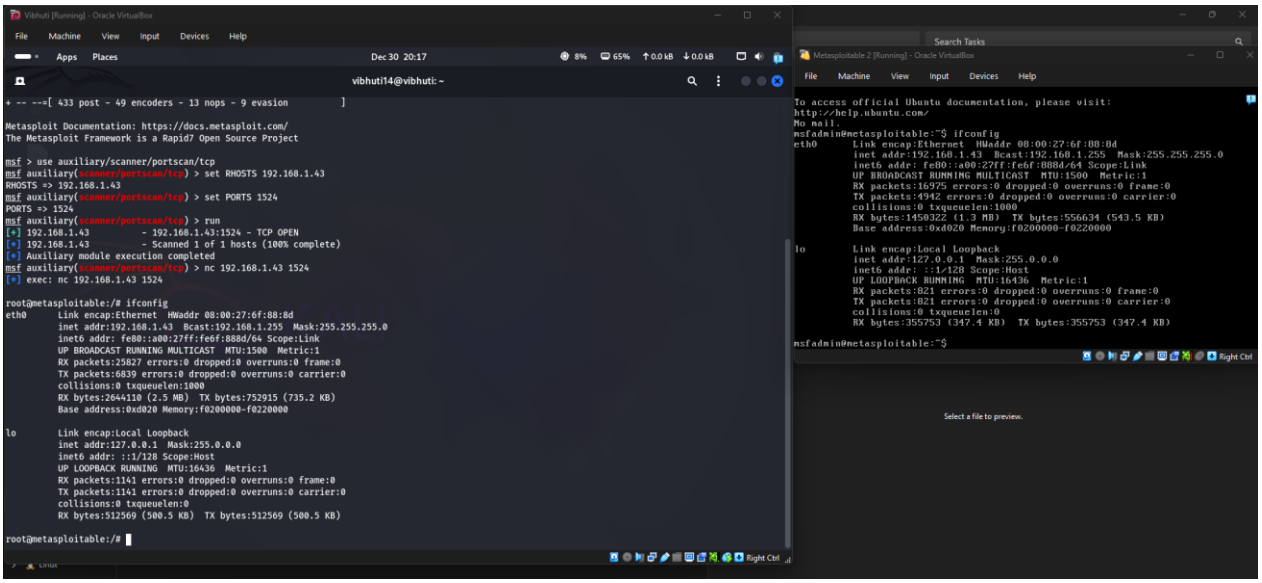
POC:

Method 1. Direct Connection (Netcat/Telnet)

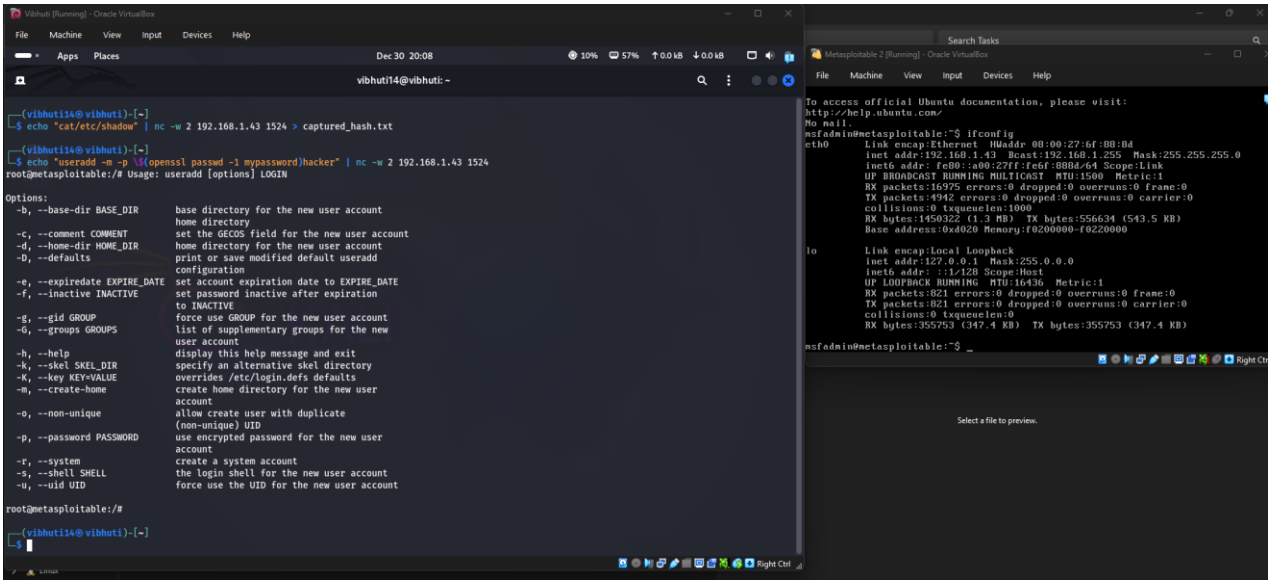


Method 2: Using Metasploit Framework Auxiliary Modules





Method 3. Scripted Command Execution (One-Liners)



nfs Port 2049

Description:

Network File System (NFS) is a distributed file system protocol that allows a user on a client computer to access files over a network as if they were stored on a local disk. Modern versions of NFS (NFSv3 and NFSv4) primarily operate on Port 2049 (TCP/UDP). While it is an essential tool for sharing resources between Linux/Unix systems, NFS was historically built for speed rather than security. It often relies on "trust" between the client and server based on IP addresses or UID/GID (User/Group ID) matching. If Port 2049 is exposed without proper encryption or robust authentication, an attacker can mount the remote file system and gain unauthorized access to every file shared by the server.

Impact:

- **Unauthorized Data Access:** If export permissions are misconfigured (e.g., using no_root_squash or insecure IP ranges), an attacker can mount the drive and read sensitive configuration files, databases, or personal data.
- **Information Leakage:** Attackers can query the port to discover the internal directory structure and export paths, which aids in planning further attacks.
- **Data Manipulation:** With write access, an attacker can modify system files, inject malicious scripts into web directories, or delete critical backups.
- **Privilege Escalation:** By exploiting the no_root_squash option, an attacker with root access on their own machine can act as a root user on the mounted NFS share, allowing them to change file ownerships and permissions on the server.

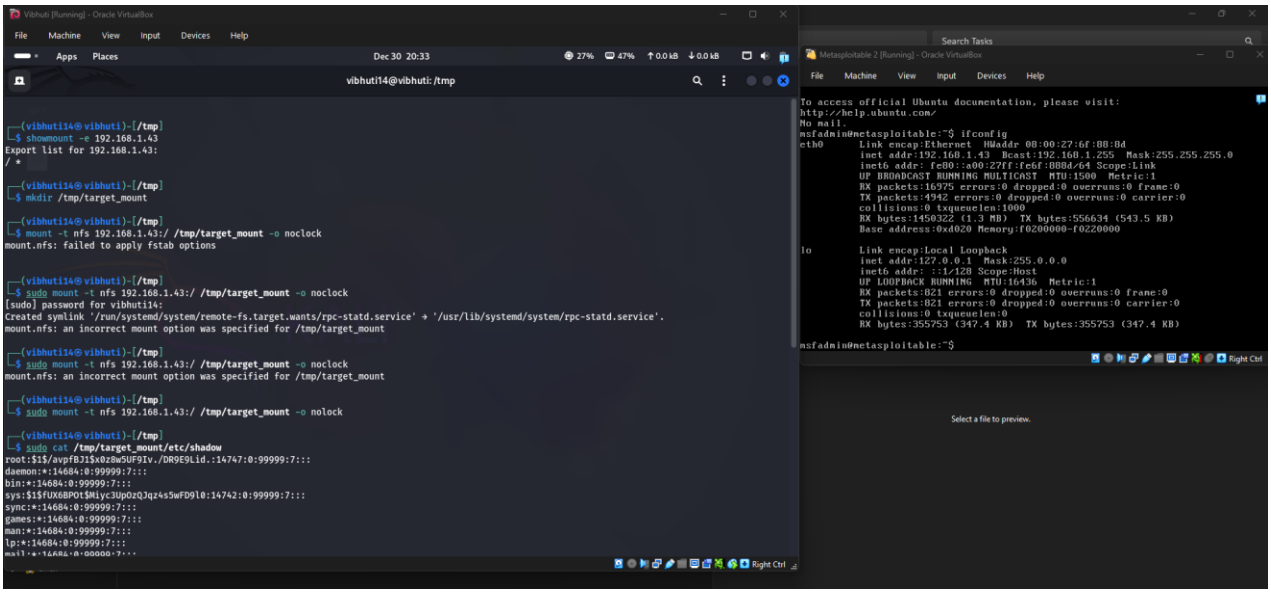
Severity: Critical

Remedial:

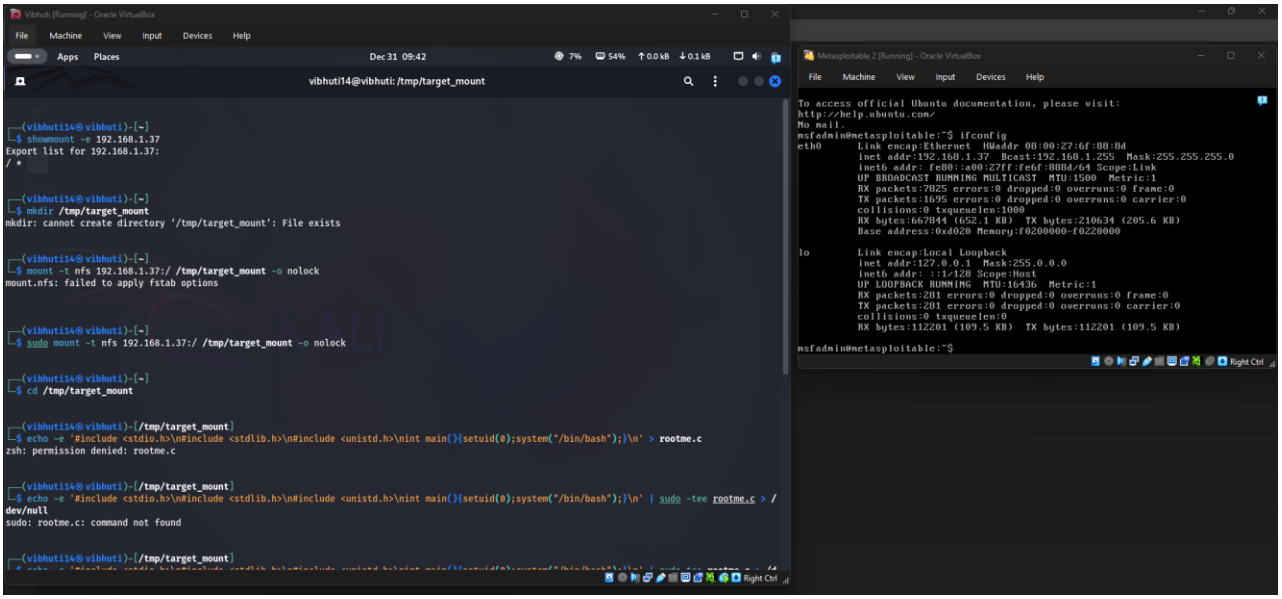
- **Restrict by IP/Subnet:** Ensure the /etc/exports file only allows connections from specific, trusted internal IP addresses rather than using wildcards (*).
- **Firewall Lockdown:** Block Port 2049 at the network perimeter. Only allow NFS traffic within a dedicated, isolated storage VLAN.
- **Use SSH Tunneling or VPN:** If you must access an NFS share over an untrusted network, wrap the connection in an SSH tunnel or a VPN to provide the encryption that native NFS lacks.
- **Mount as Read-Only:** Where possible, export directories with the ro (read-only) flag to prevent unauthorized data modification.

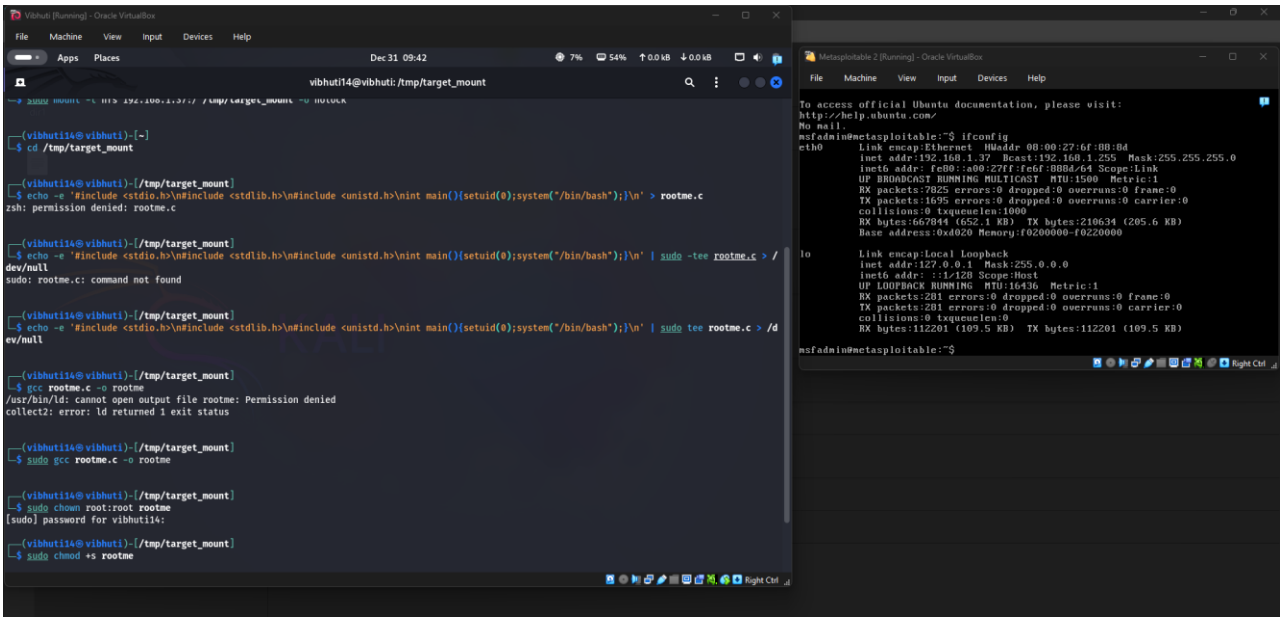
POC:

Method 1. Root File System Mounting (Unauthorized Access)

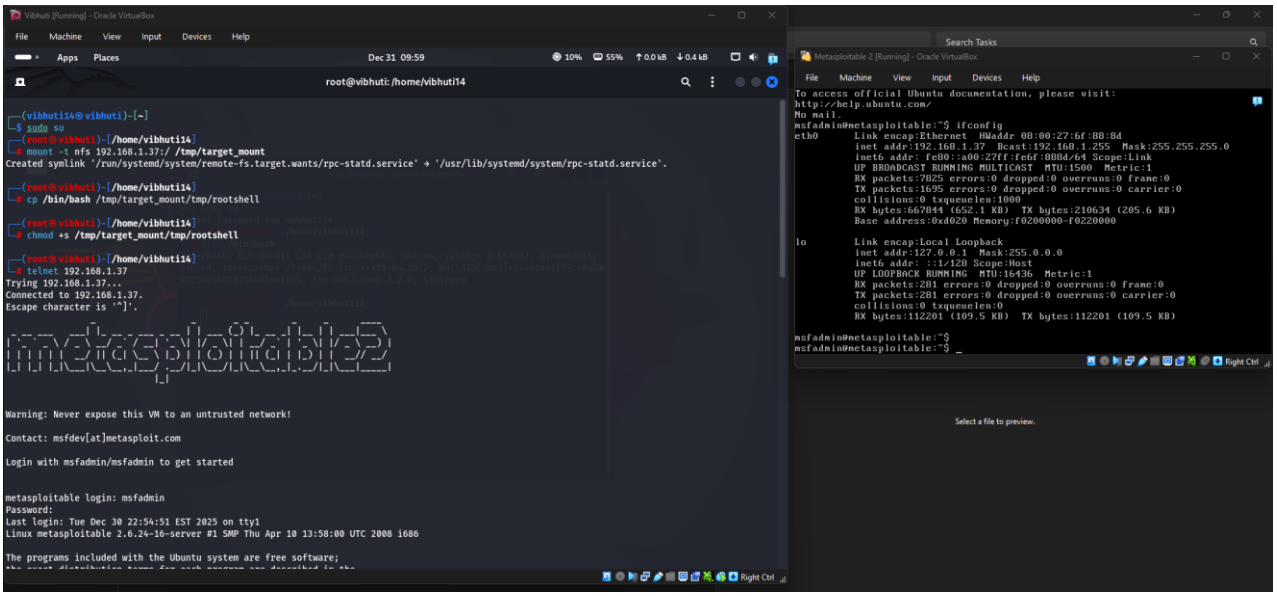


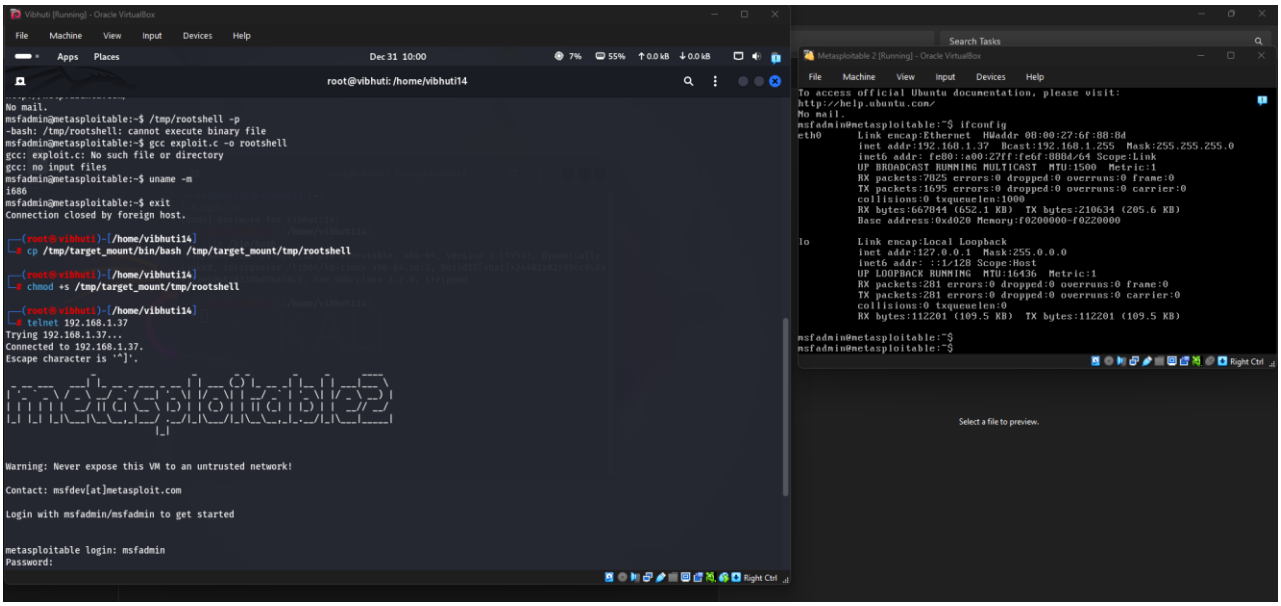
Method 2. Privilege Escalation via no_root_squash

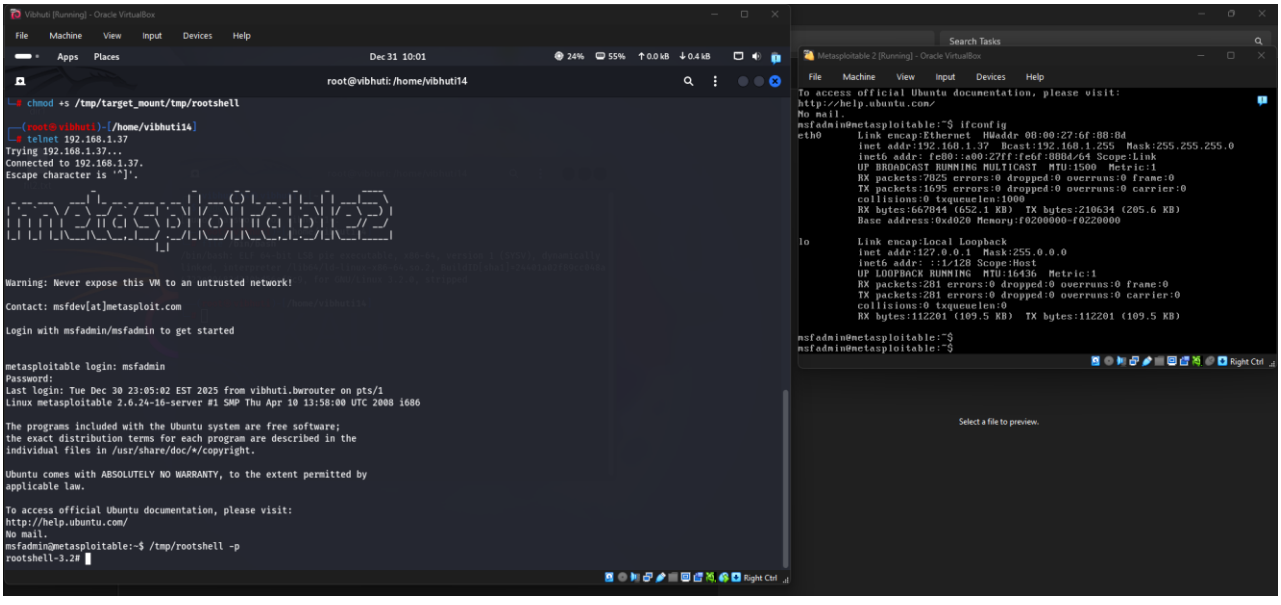




Method 3. SUID Binary Creation (Instant Privilege Escalation)







ftp Port 2121

Description:

Port 2121 is a non-standard port frequently used as an alternative to the default FTP Port 21. It is commonly employed in two scenarios: first, by developers or administrators who wish to run an FTP server without root/administrative privileges (as ports below 1024 often require elevated permissions); and second, as a form of "security through obscurity" to hide the service from simple botnets that only scan standard ports. Despite being moved to a different number, the underlying protocol remains the same File Transfer Protocol.

Impact:

- **Cleartext Exposure:** Moving the service to Port 2121 does not automatically add encryption; credentials and data are still vulnerable to packet sniffing.
- **Bypassing Firewalls:** Some internal firewalls or egress filters may be configured to monitor Port 21 for suspicious activity but might ignore Port 2121, allowing unauthorized data movement to go unnoticed.
- **False Sense of Security:** Administrators may mistakenly believe that using a non-standard port makes the service "hidden," leading them to neglect strong password policies or encryption.
- **Tool Incompatibility:** Some security monitoring tools and Application Layer Gateways (ALGs) are hardcoded to inspect Port 21; they may fail to inspect traffic on 2121, potentially breaking active/passive mode transitions or missing malicious payloads.

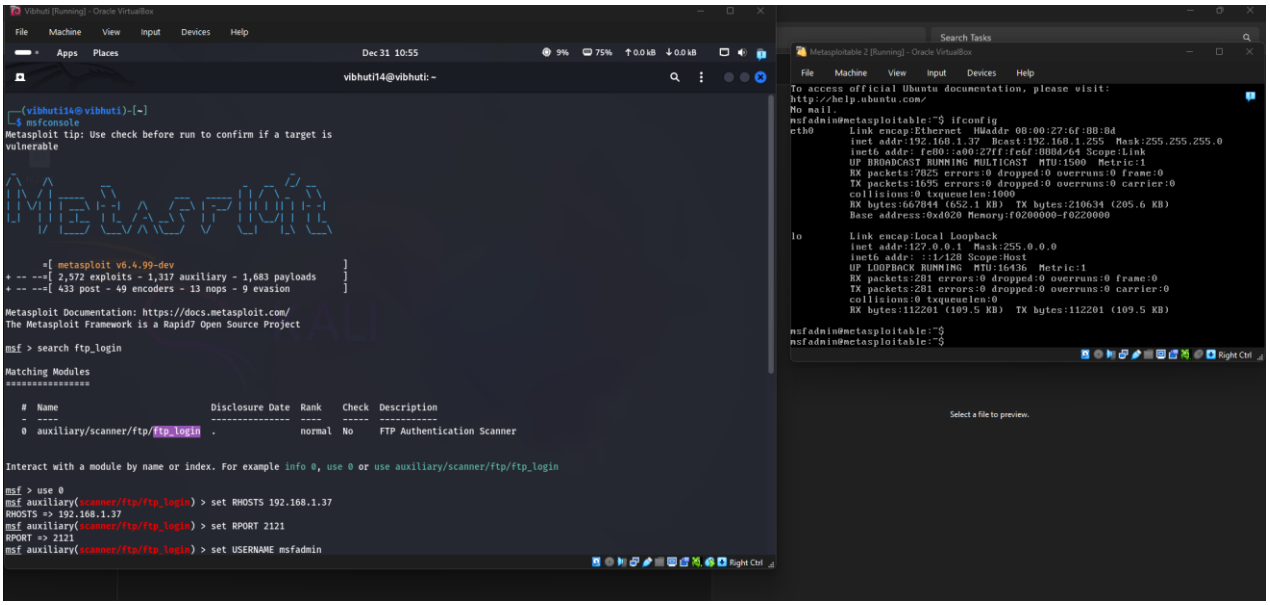
Severity: High

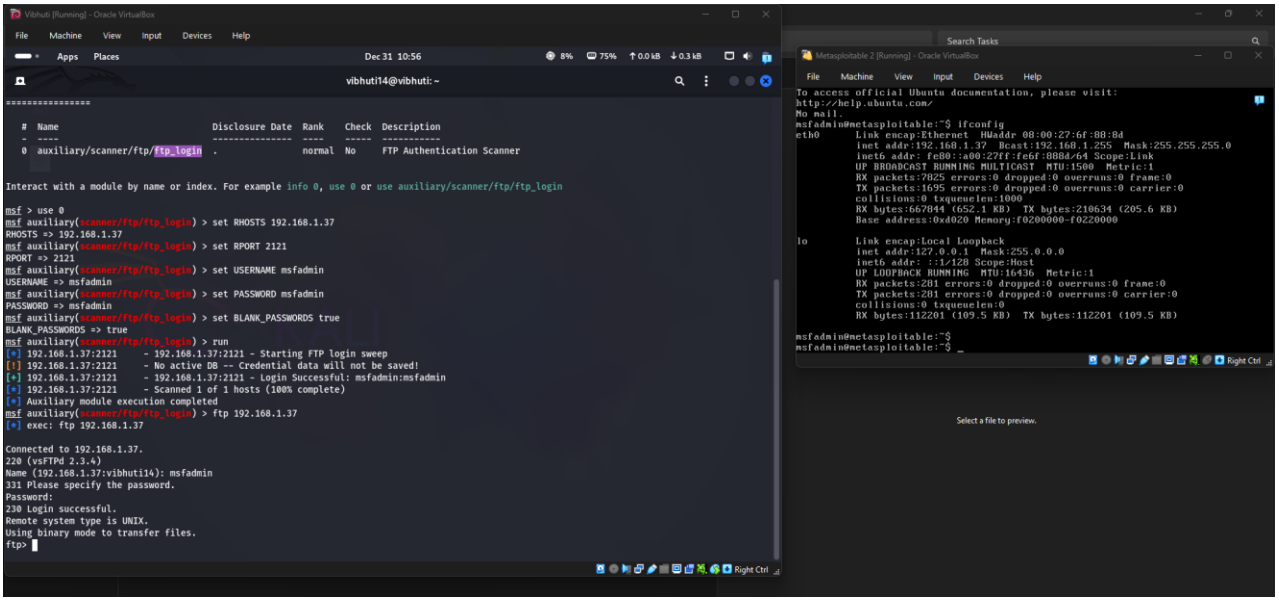
Remedial:

- **Upgrade to SFTP (Port 22):** The most effective remedy is to stop using FTP altogether and switch to SFTP, which provides native encryption for both authentication and data.
- **Enforce Explicit FTPS:** If you must use Port 2121, configure the server to require STARTTLS. This ensures the connection is upgraded to a secure, encrypted tunnel before any credentials are exchanged.
- **Implement IP Whitelisting:** Since Port 2121 is often used for specific administrative tasks, restrict access at the firewall level to only allow specific, trusted IP addresses.
- **Monitor for Non-Standard Traffic:** Configure Intrusion Detection Systems (IDS) to inspect traffic on Port 2121 specifically for FTP-based signatures and brute-force patterns.
- **Use Strong Authentication:** Ensure that all accounts accessible via Port 2121 use complex passwords and that anonymous access is strictly disabled.

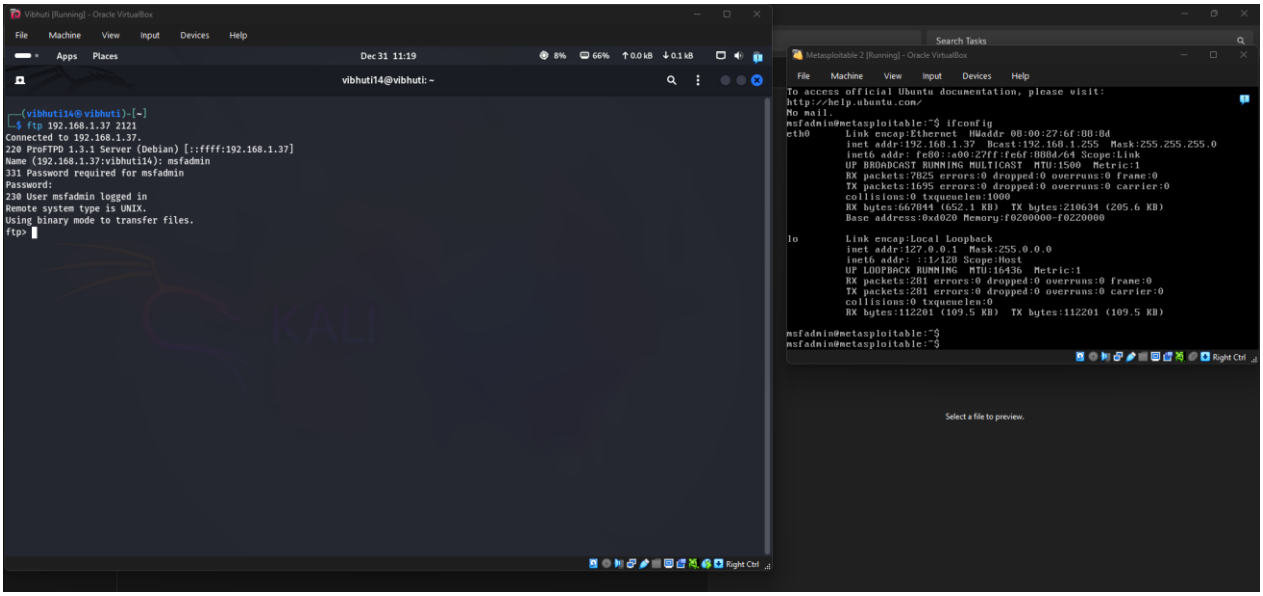
POC:

Method 1. Brute-Force Login with auxiliary/scanner/ftp/ftp_login

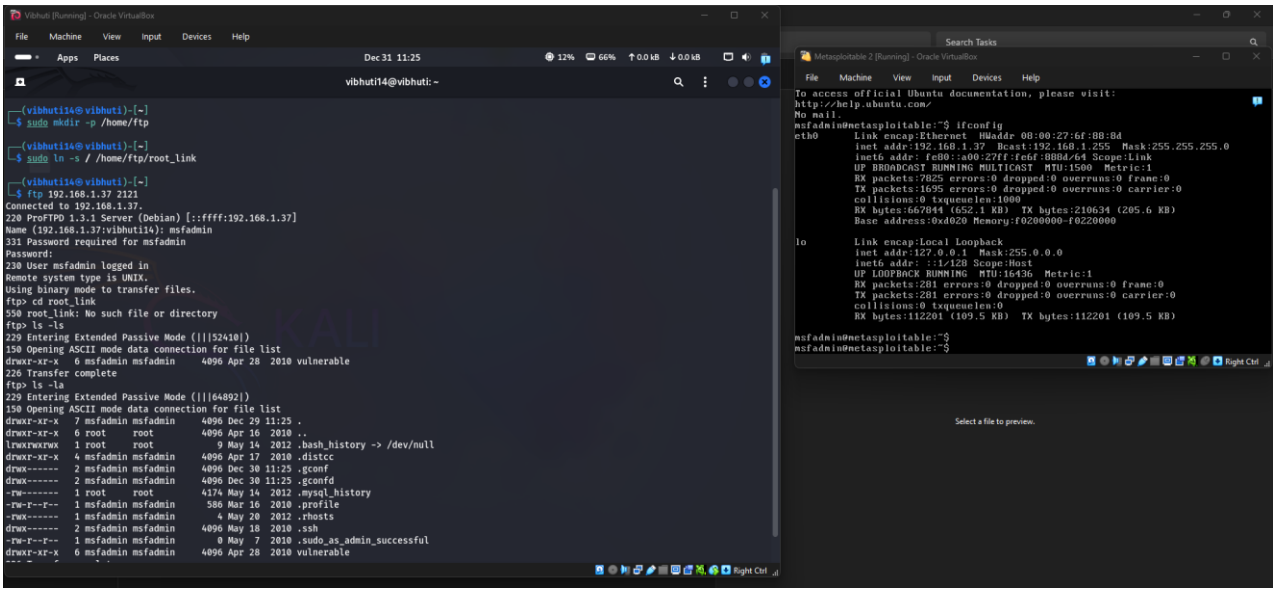




Method 2. Manual Login with Default Credentials



Method 3. Symlink Attack (Directory Traversal)



mysql Port3306

Description:

Port 3306 is the default port for MySQL and MariaDB, the most widely used relational database management systems in the world. It serves as the primary communication gateway between the database server and its clients, such as web applications, data analysis tools, or administrative consoles. When an application needs to store or retrieve data, it initiates a TCP connection to this port to send SQL queries and receive result sets. While modern MySQL versions support encrypted connections via TLS, many legacy configurations still transmit data in cleartext or rely on weak authentication methods. Because this port is the direct entrance to a company's "crown jewels"—its data—it is a constant target for attackers seeking to steal sensitive information or deploy ransomware.

Impact:

- **Data Breach & Exfiltration:** Successful unauthorized access allows an attacker to dump entire databases, exposing customer records, financial data, and intellectual property.
- **SQL Injection Escalation:** If an application is vulnerable to SQL injection, an open Port 3306 can allow an attacker to connect directly to the database, bypassing application-level security controls.
- **Brute-Force & Credential Stuffing:** Since Port 3306 is well-known, it is subject to non-stop automated attempts to guess "root" or "admin" passwords.

- **Privilege Escalation:** Exploiting misconfigured "User Defined Functions" (UDF) through an open port can sometimes allow an attacker to gain command-line access to the host operating system.

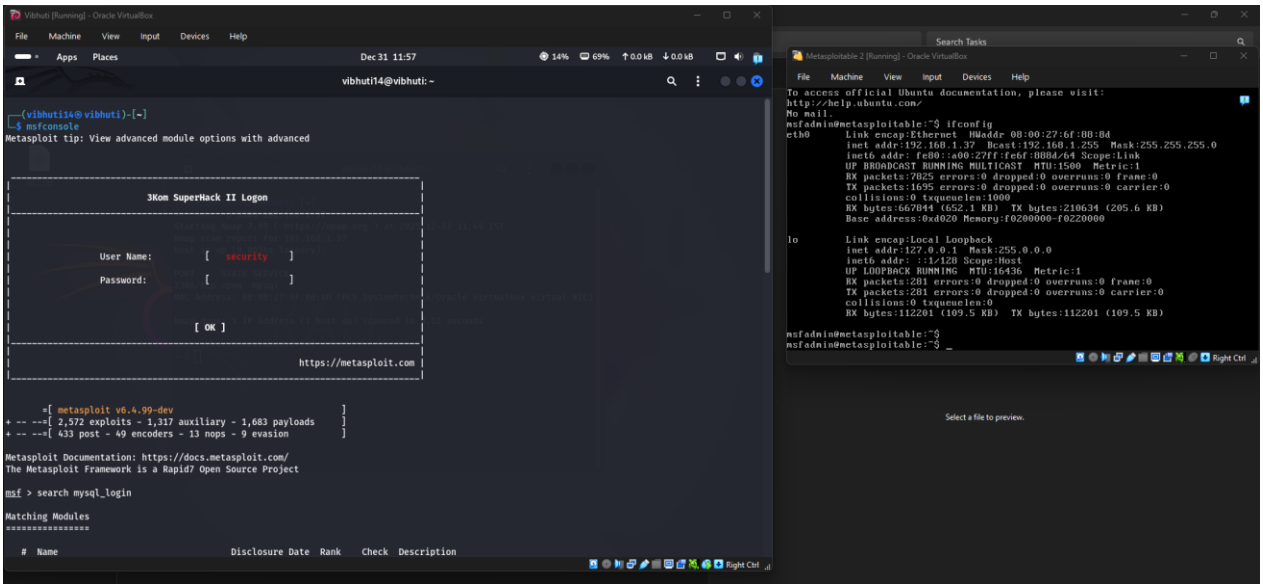
Severity: Critical

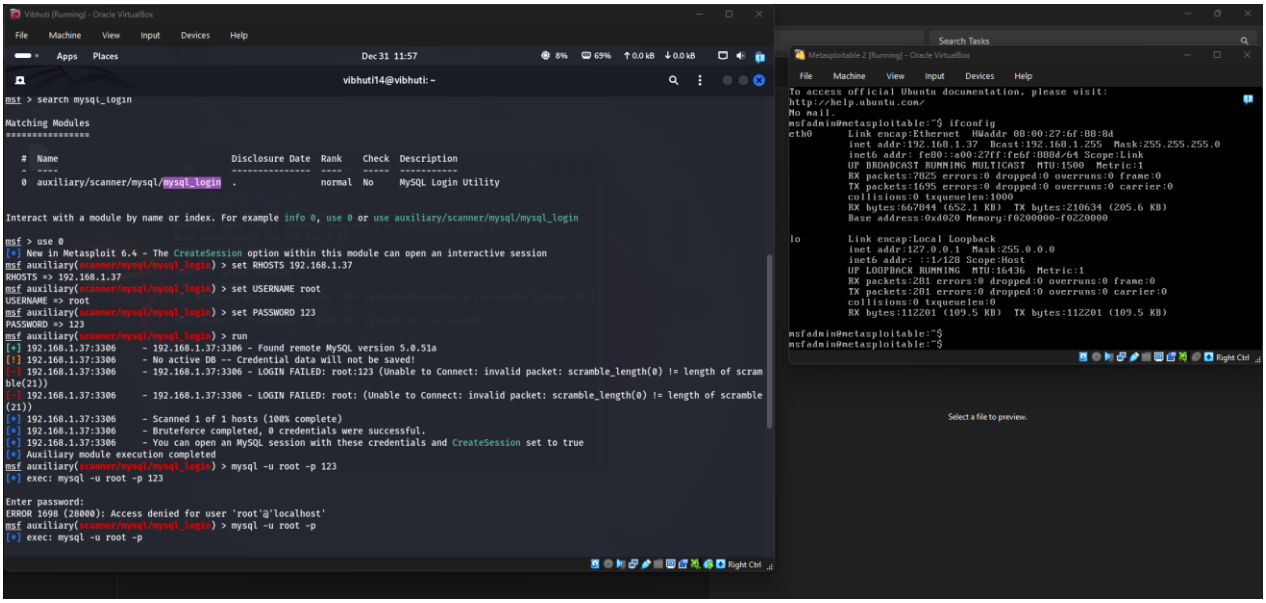
Remedial:

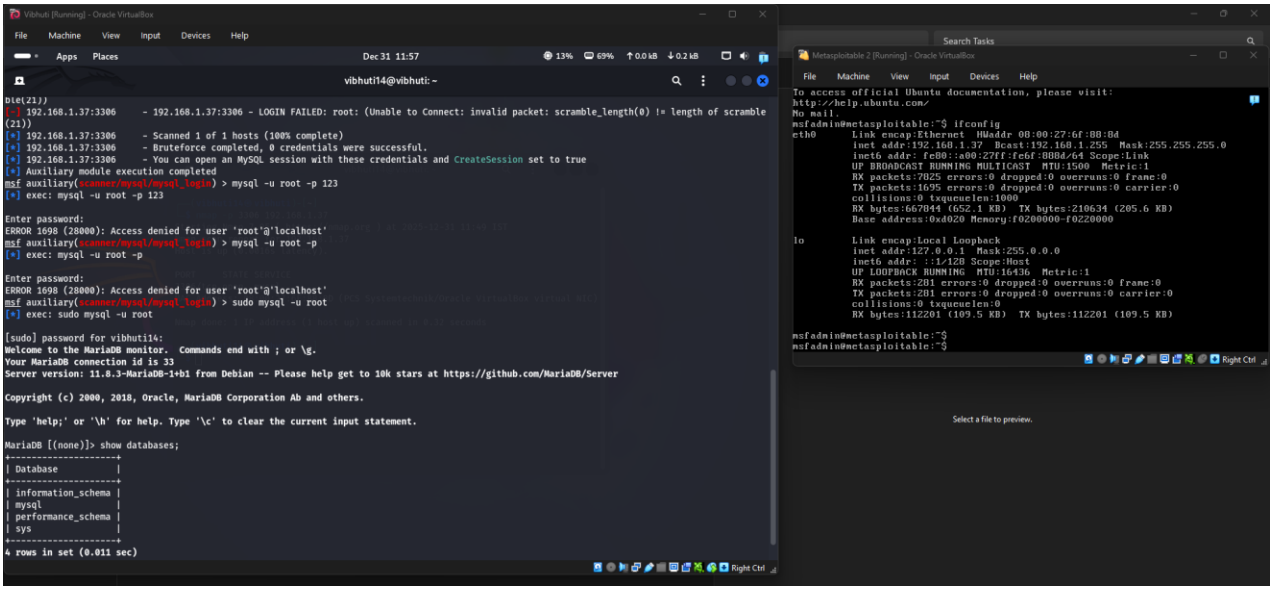
- **Bind to Localhost:** If the database is only used by a web application on the same server, configure MySQL to listen only on 127.0.0.1 by editing the my.cnf file.
- **Restrict Access via Firewall:** Never expose Port 3306 to the public internet. Use a firewall to restrict access to specific application server IPs or require a VPN/SSH Tunnel for remote administration.
- **Disable the Root Remote Login:** Ensure the root user can only log in from localhost. Create specific, limited-privilege users for remote applications.
- **Change the Default Port:** Moving the service to a non-standard port (e.g., 3307 or a random high port) can reduce the volume of automated bot attacks, though it is not a substitute for real security.
- **Implement Strong Password Policies:** Use complex passwords and consider using plugins that support Multi-Factor Authentication (MFA).

POC:

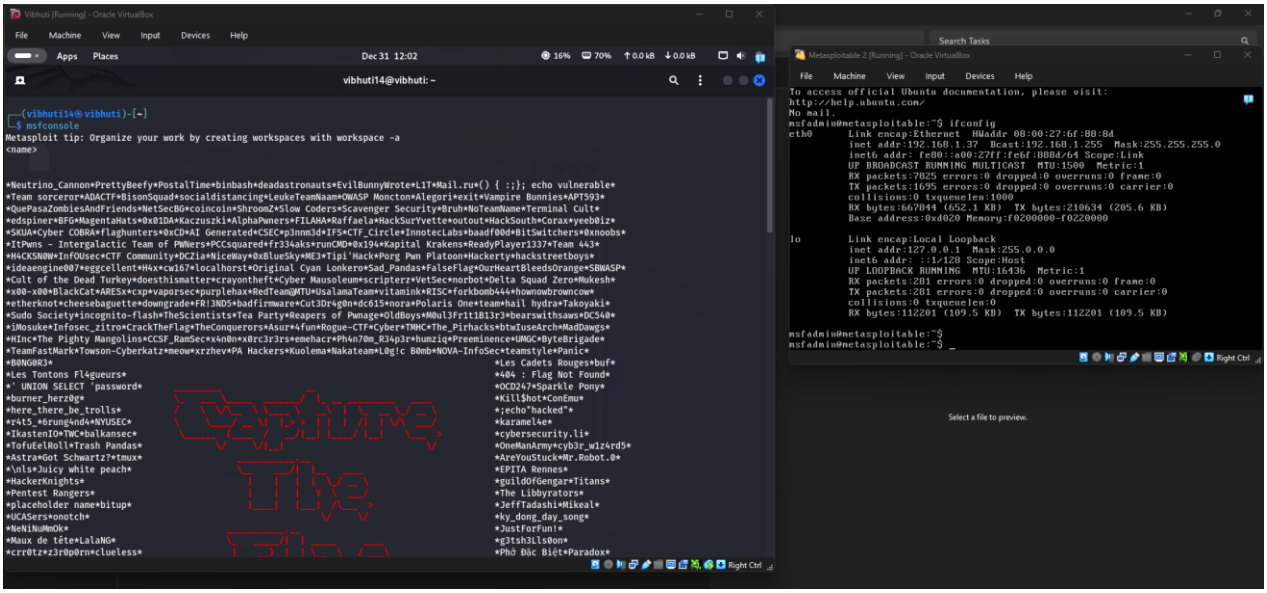
Method 1. Root Login with Empty Password (Misconfiguration)

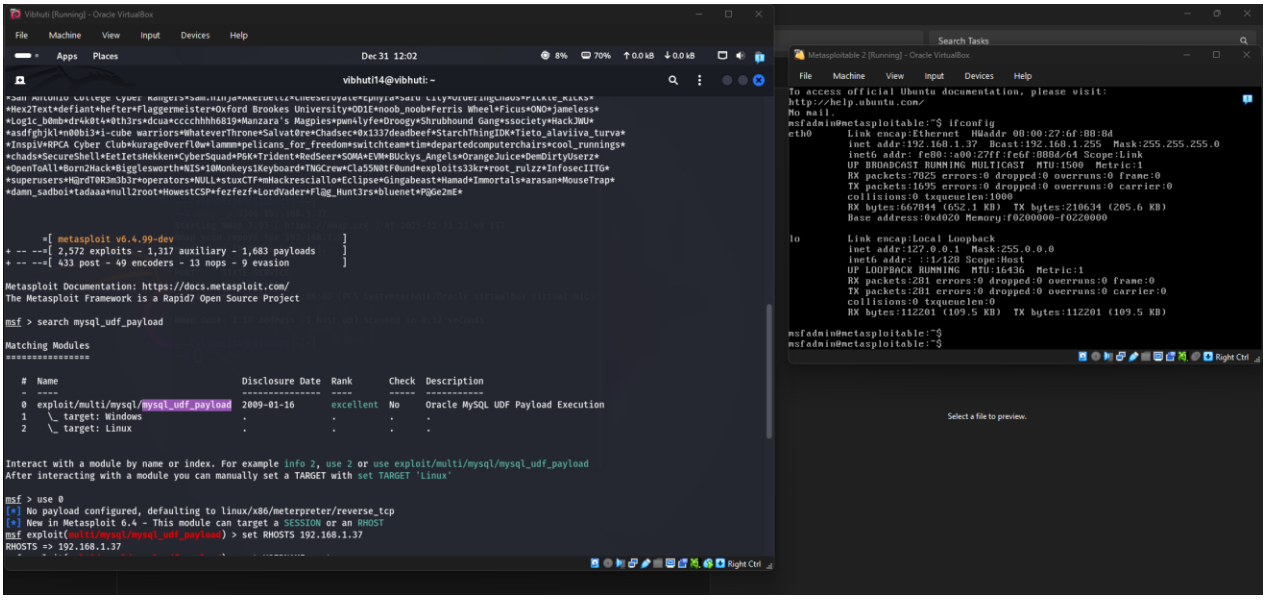


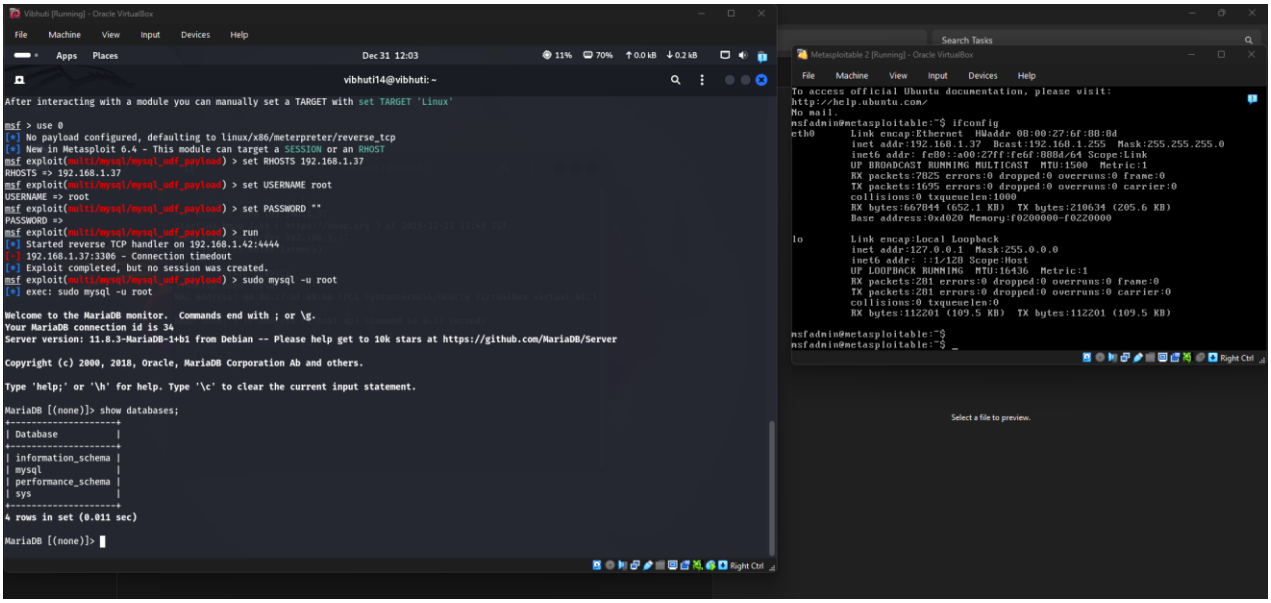




Method 2. Remote Code Execution via User Defined Functions (UDF)







postgresql Port 5432

Description:

Port 5432 is the standard port used by PostgreSQL, a powerful, open-source object-relational database system. This port serves as the primary endpoint for all client-server communication, where the database engine listens for incoming connections from web applications, business intelligence tools, and database administrators. When a connection is established, Port 5432 handles the authentication handshake and the subsequent exchange of SQL queries and data results. While PostgreSQL is renowned for its advanced security features, such as robust Role-Based Access Control (RBAC) and native support for SSL/TLS encryption, an improperly hardened Port 5432 remains a high-value target for attackers aiming to gain access to sensitive organizational data.

Impact:

An exposed or poorly secured Port 5432 can lead to several severe security outcomes:

- **Unauthorized Data Access:** Attackers can attempt to bypass authentication or exploit weak credentials to query, download, or delete entire database tables.
- **Brute-Force Attacks:** Since 5432 is a well-known port, it is frequently targeted by bots attempting to guess passwords for default accounts like postgres.
- **Denial of Service (DoS):** An attacker can flood the port with connection requests, exhausting the server's connection pool and rendering the database unavailable to legitimate applications.
- **Man-in-the-Middle (MitM) Sniffing:** If SSL is not enforced, SQL queries—which may contain PII or sensitive business logic—are transmitted in cleartext and can be intercepted.

Severity: Critical

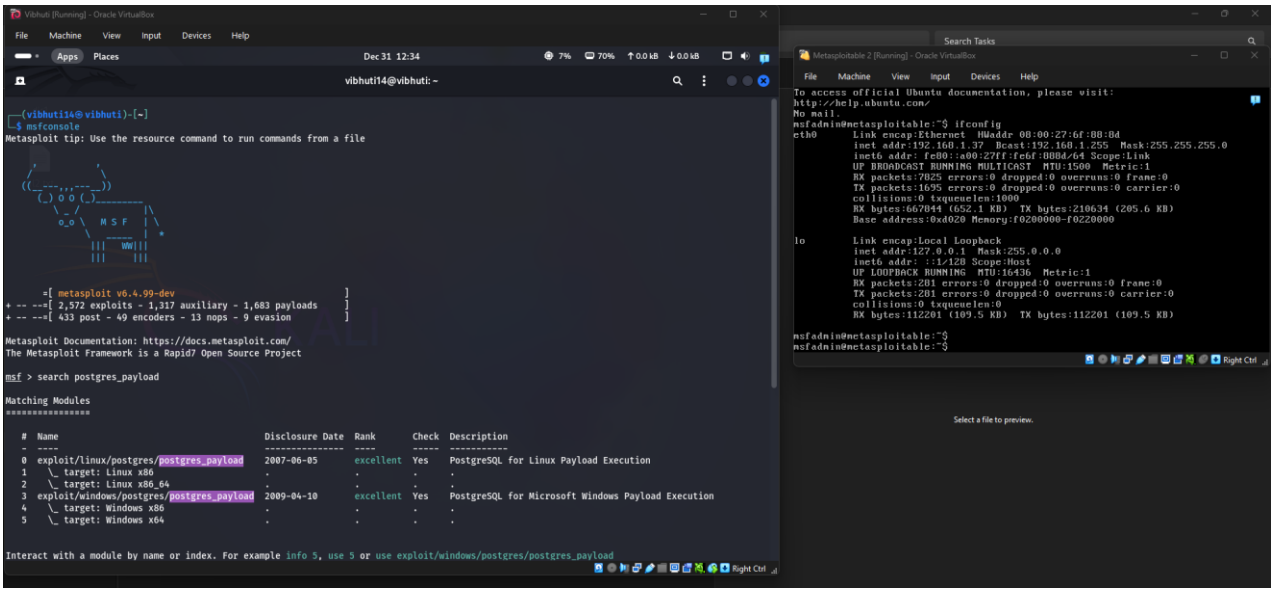
Remedial:

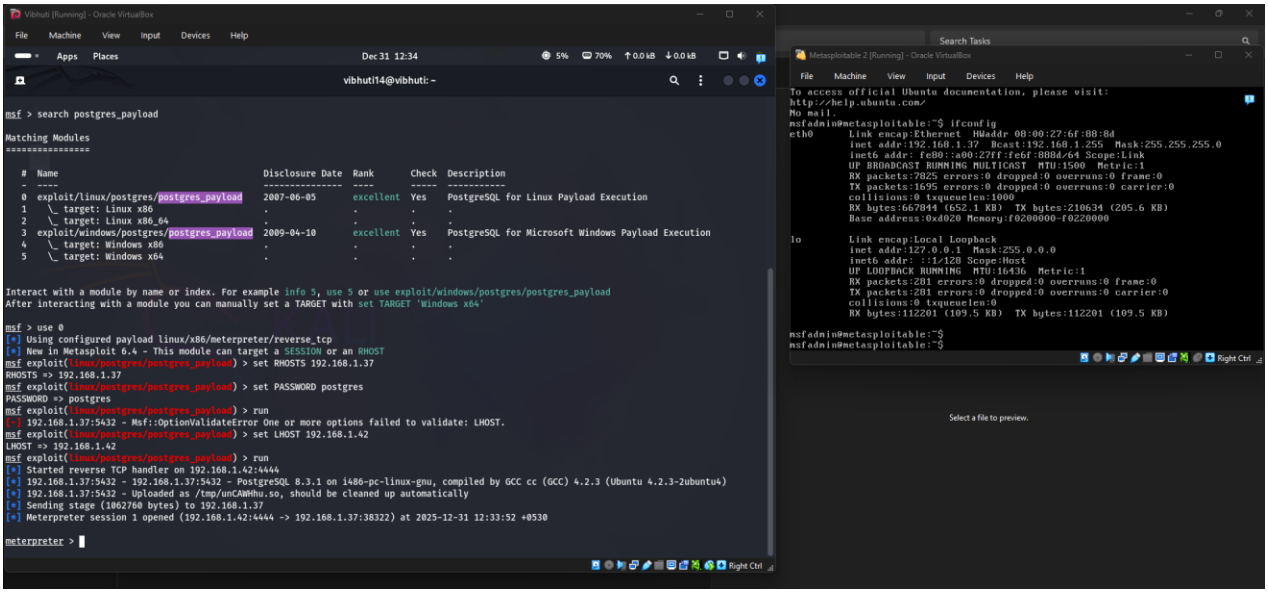
To secure your PostgreSQL instance, follow these best practices:

- **Modify pg_hba.conf:** Use the Host-Based Authentication (HBA) file to strictly limit which IP addresses or subnets are allowed to connect to which databases. Avoid using 0.0.0.0/0.
- **Bind to Specific Interfaces:** In postgresql.conf, set the listen_addresses parameter to localhost or a specific internal private IP, rather than allowing it to listen on all available network interfaces.
- **Enforce SSL/TLS:** Set ssl = on in the configuration and require clients to use encrypted connections by using the hostssl record type in the pg_hba.conf file.
- **Firewall Isolation:** Use a network firewall or Cloud Security Group to ensure Port 5432 is never reachable from the public internet. Access should only be granted via a secure VPN or SSH Tunnel.
- **Rename or Change the Port:** While not a primary security measure, changing the port from 5432 to a non-standard number can reduce the noise from automated internet scanners.

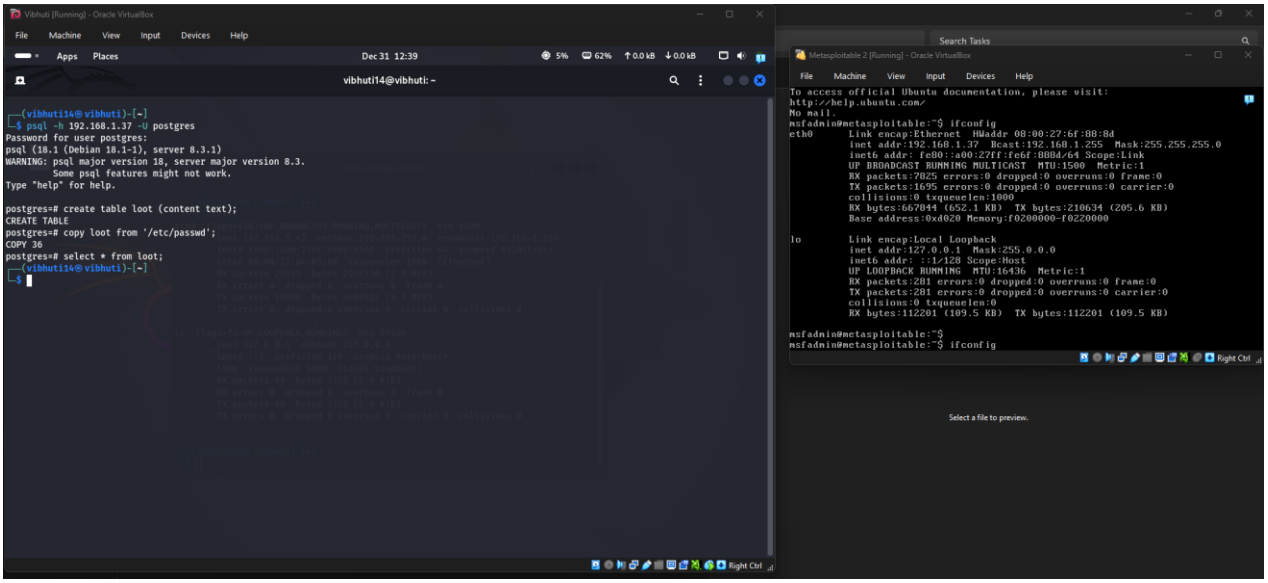
POC:

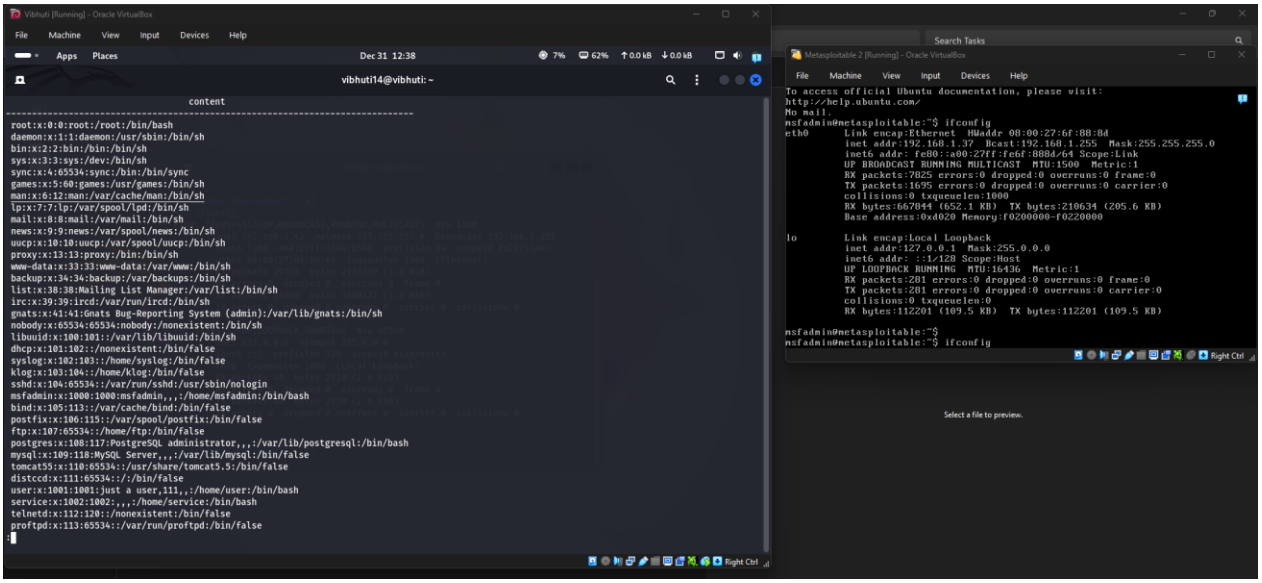
Method 1. Remote Code Execution via Linux "Shared Object" (UDF)



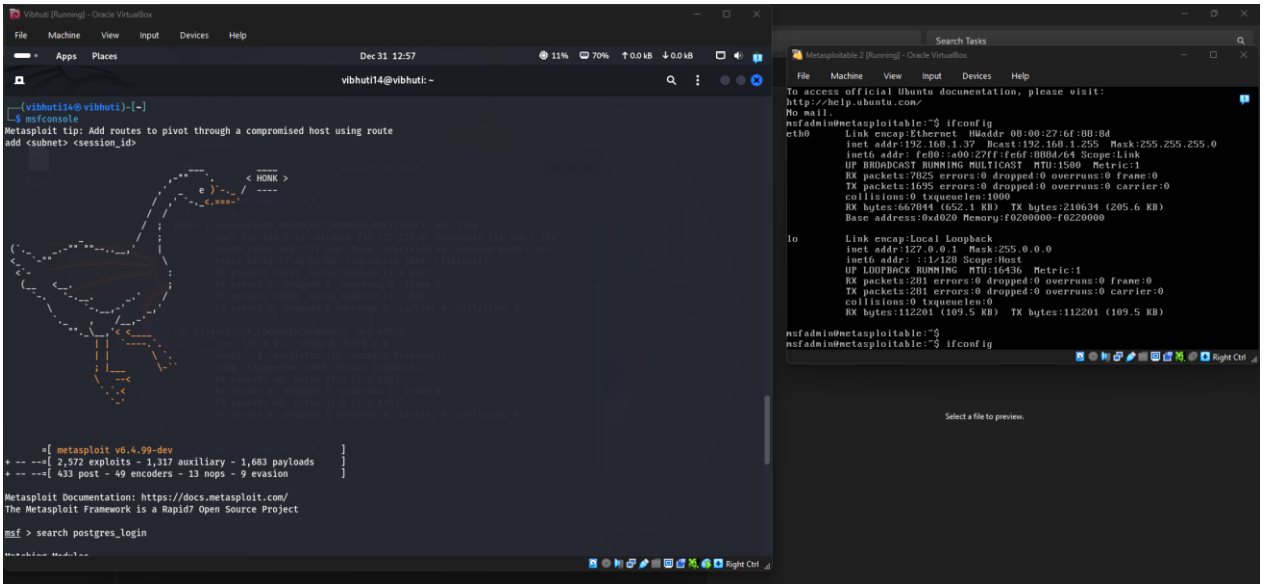


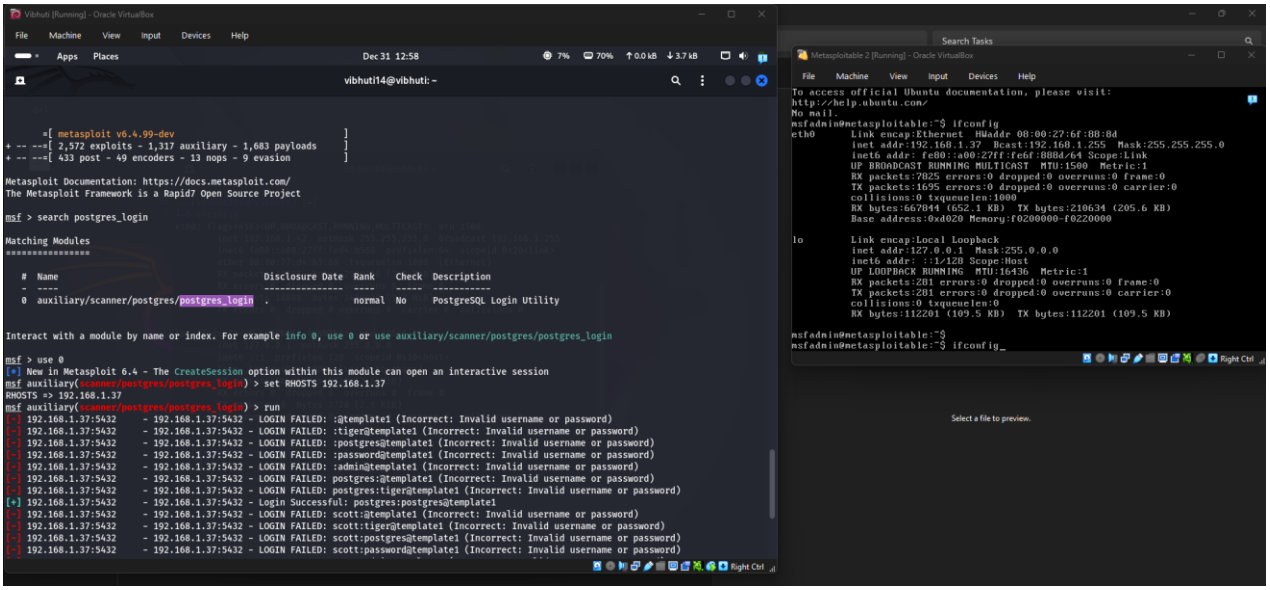
Method 2. File System Access via COPY FROM PROGRAM

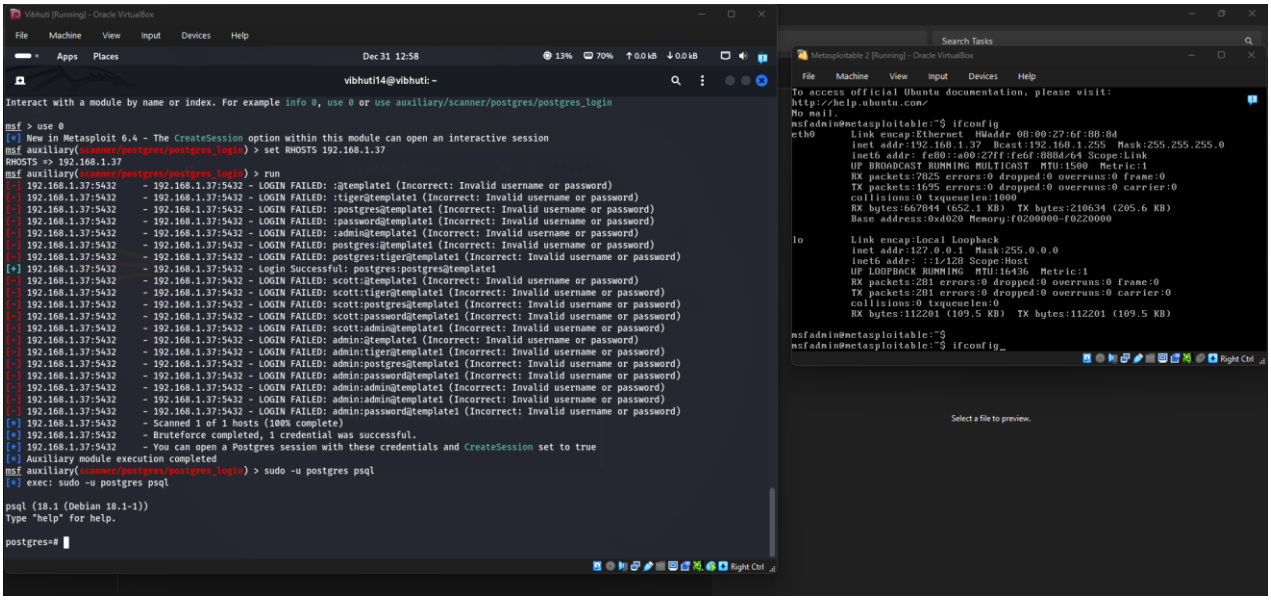




Method 3. Automated Brute-Force & Enumeration







vnc Port 5900

Description:

Virtual Network Computing (VNC) is a graphical desktop-sharing system that typically uses Port 5900 to allow remote control of another computer. It works on the Remote Frame Buffer (RFB) protocol, where the server transmits the display to the client and the client sends keyboard and mouse events back to the server. Unlike SSH, which is text-based, VNC provides a full GUI experience, making it popular for remote technical support and server administration. However, standard VNC is notoriously insecure; it often lacks robust encryption for the actual data stream, and many older implementations only encrypt the initial password exchange, leaving the entire visual session vulnerable to interception

Impact:

- **Full Graphical Control:** A successful exploit gives an attacker "eyes on" the system, allowing them to see exactly what a logged-in user sees and interact with any open application.
- **Credential Theft:** Since the session data is often unencrypted, attackers can capture keystrokes (keylogging) as the user types passwords into websites or local applications.
- **Session Hijacking:** Attackers can "sniff" the RFB traffic and inject their own mouse and keyboard events into an active session.
- **Cleartext Exposure:** Many VNC versions transmit the desktop image in plain text; anyone on the network path can essentially "watch" the remote user's screen in real-time.

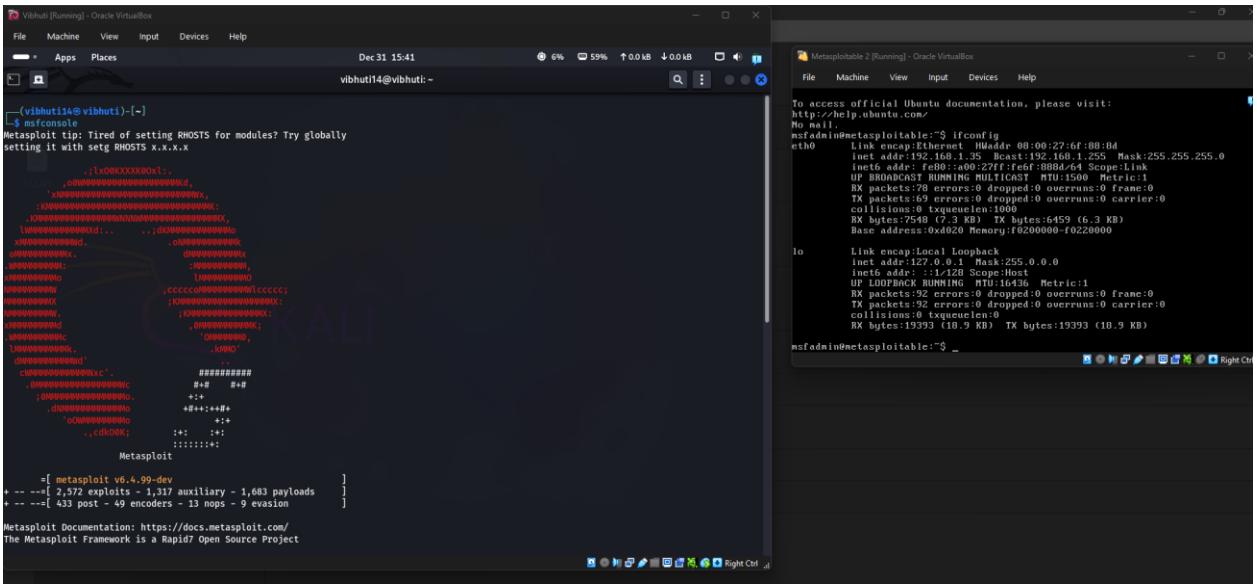
Severity: Critical

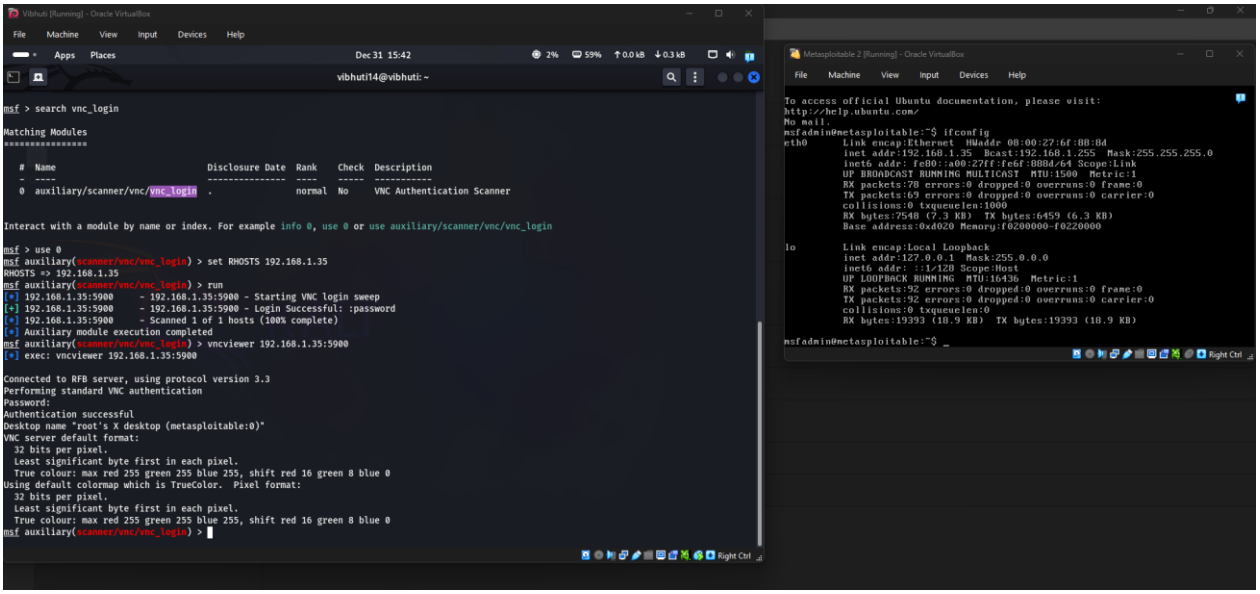
Remedial:

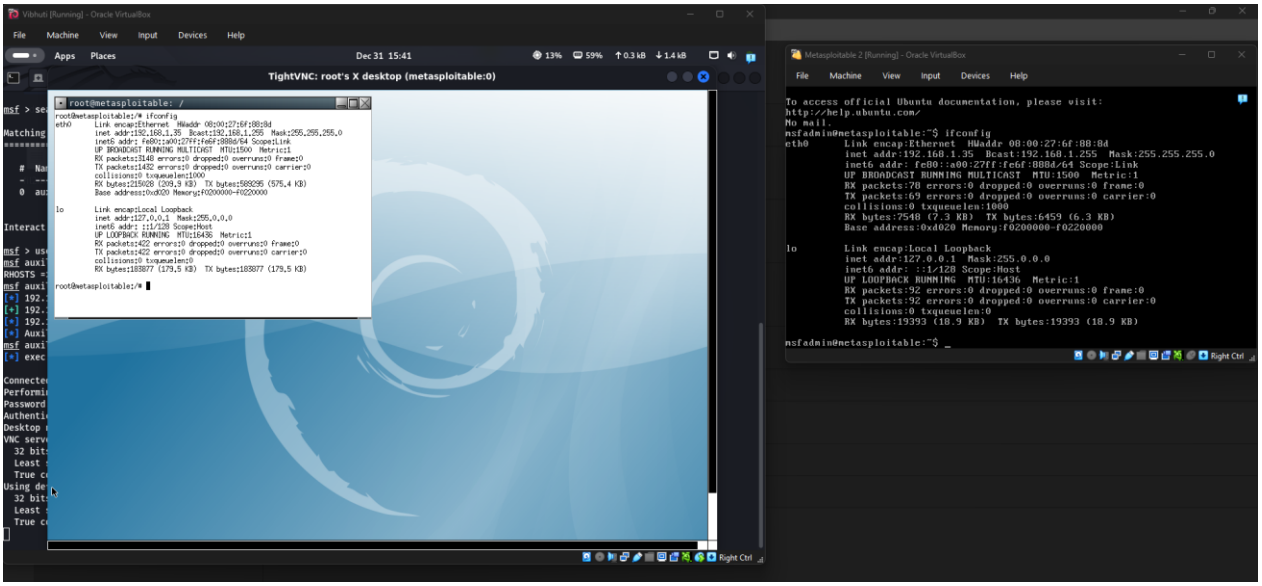
- **Tunnel VNC over SSH:** Never expose Port 5900 directly. Close the port on your firewall and use an SSH Tunnel (Local Port Forwarding) to wrap the VNC traffic in a secure, encrypted pipe.
- **Restrict Access to VPN:** Only allow VNC connections from within a secure Virtual Private Network (VPN) or specific internal IP addresses.
- **Enable Strong Encryption:** If you must use VNC, ensure you are using a version that supports VNC Authentication with TLS/SSL (like RealVNC or TigerVNC).
- **Enforce Complex Passwords:** Move beyond the standard 8-character VNC password and use system-level authentication (PAM) if supported by your VNC server.
- **Disable "View Only" or "Shared" Sessions:** Configure the server to disconnect idle sessions and prevent multiple users from connecting to the same screen simultaneously unless required.

POC:

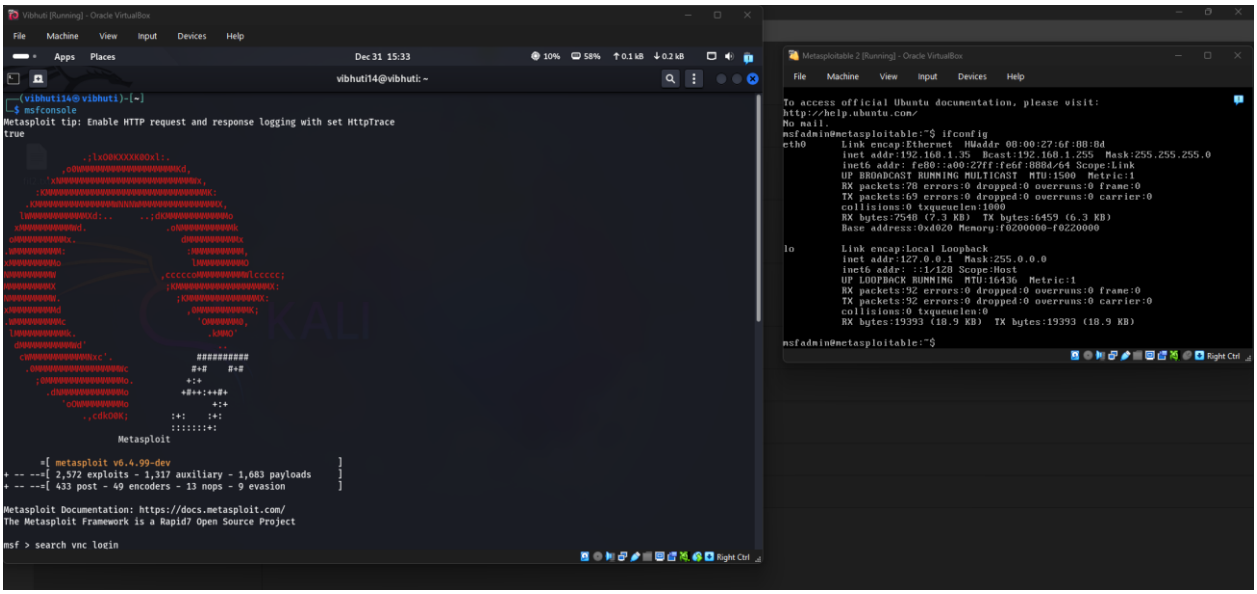
Method 1. VNC Password Brute-Force (The "password" Exploit)

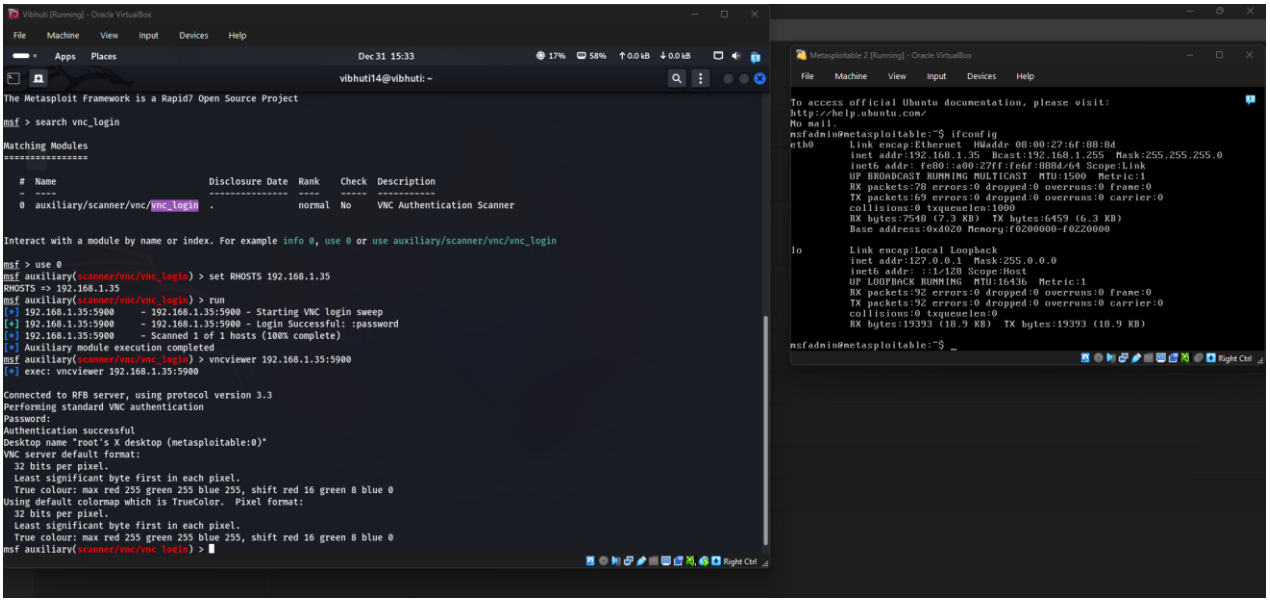


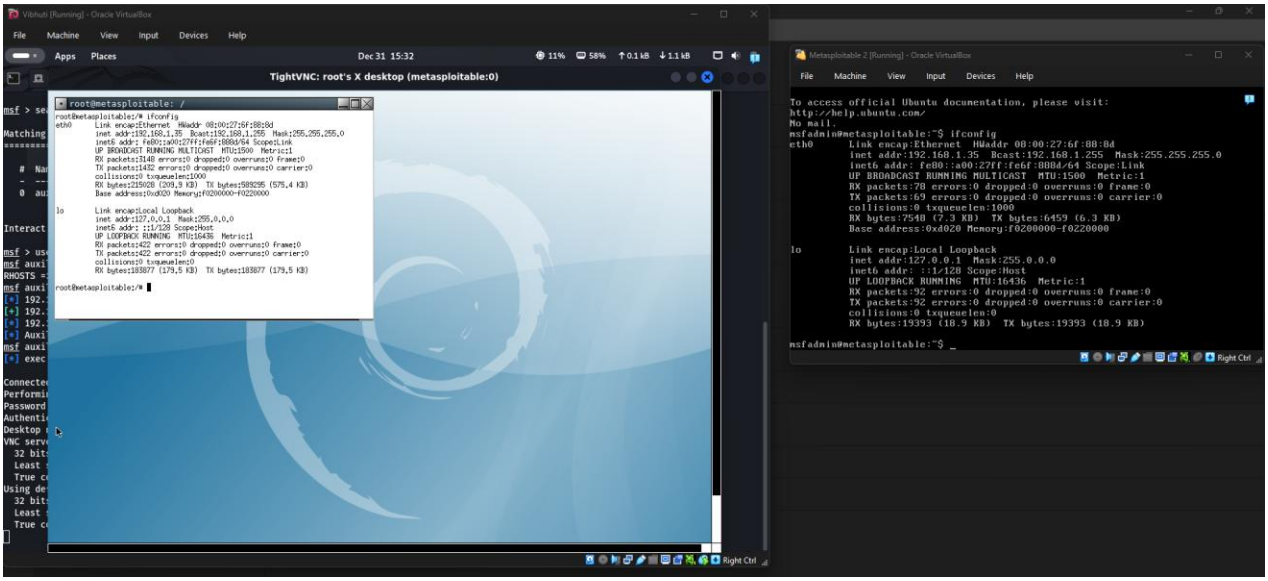




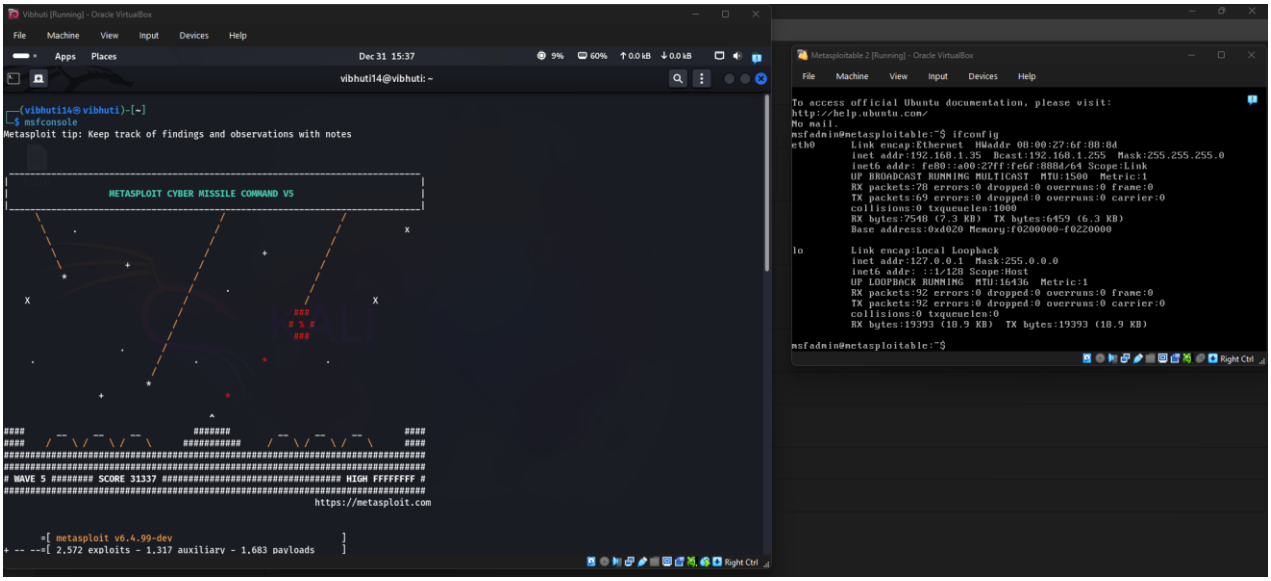
Method 2. VNC Authentication Bypass (CVE-2006-2369)

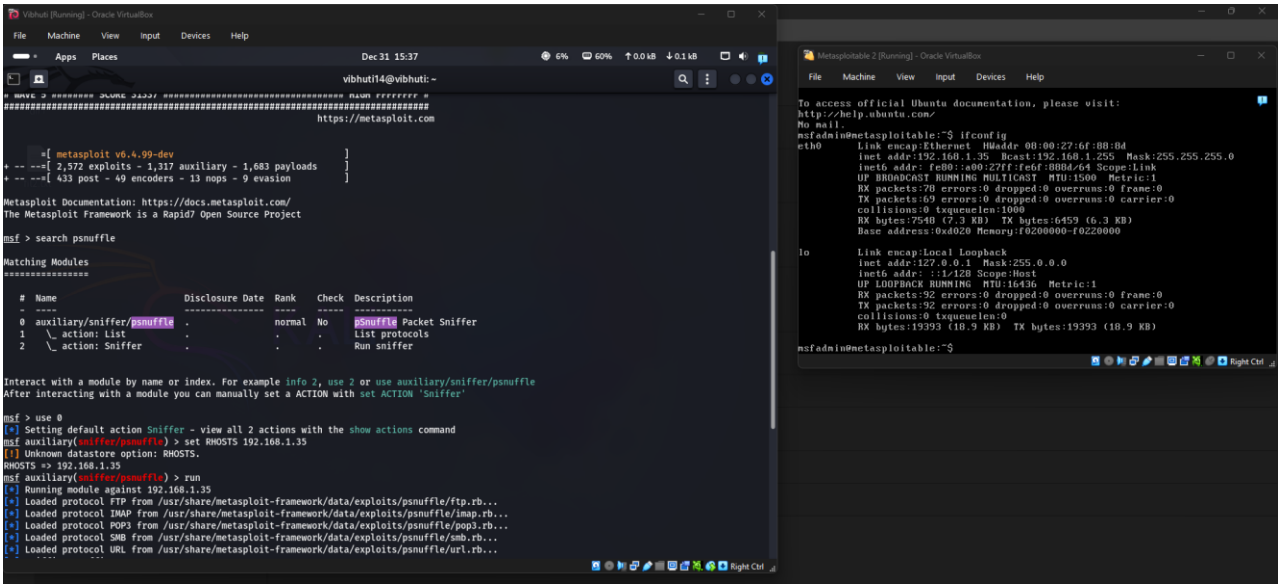


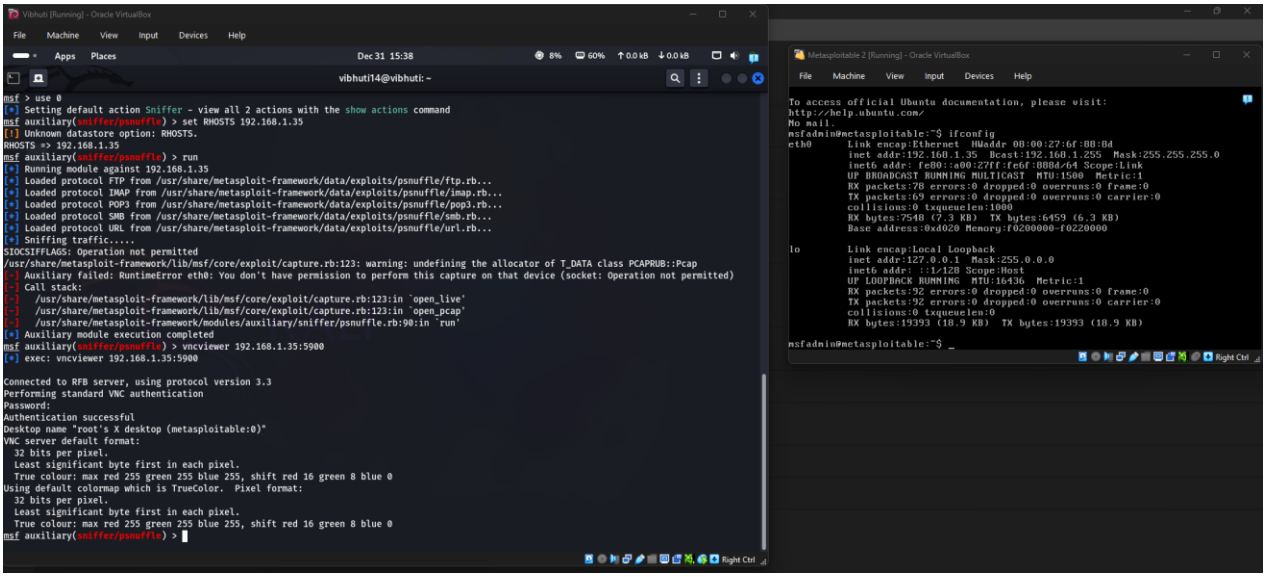


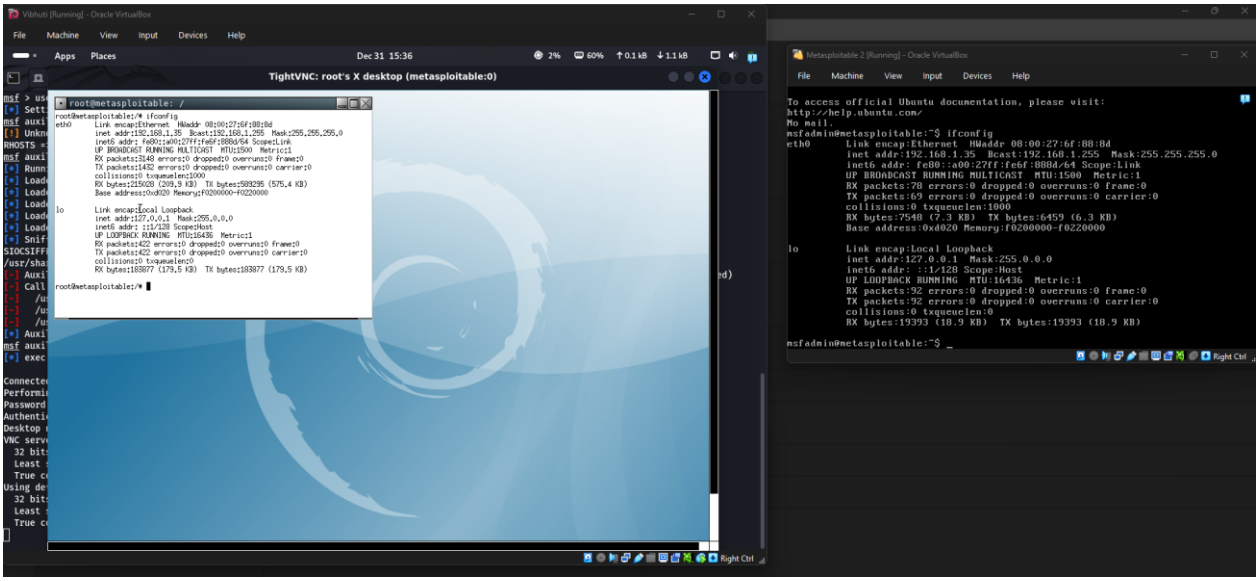


Method 3. VNC Session Hijacking & Keystroke Sniffing









X11 Port 6000

Description:

The X Window System, commonly known as X11, is a windowing system for bitmap displays, prevalent on Unix-like operating systems. It uses a client-server architecture where the X Server listens for connections from X Clients over Port 6000. Because X11 was designed for local network transparency in an era before modern security threats, it lacks native encryption and robust authentication. By default, an X Server listening on Port 6000 may allow any remote client to connect and interact with the display if access controls like xhost are misconfigured.

Impact:

- **Keystroke Logging:** An attacker who connects to an open X11 port can capture every keystroke typed by the local user, including passwords and sensitive messages.

- **Screen Capturing:** The protocol allows remote clients to take screenshots or record the active display, leading to a total loss of visual privacy.
- **Unauthorized Input Injection:** Attackers can send artificial mouse clicks and keyboard events, effectively "driving" the computer to open terminals, delete files, or download malware.
- **Information Disclosure:** Attackers can query the properties of all open windows, discovering what applications are running and the titles of active documents.

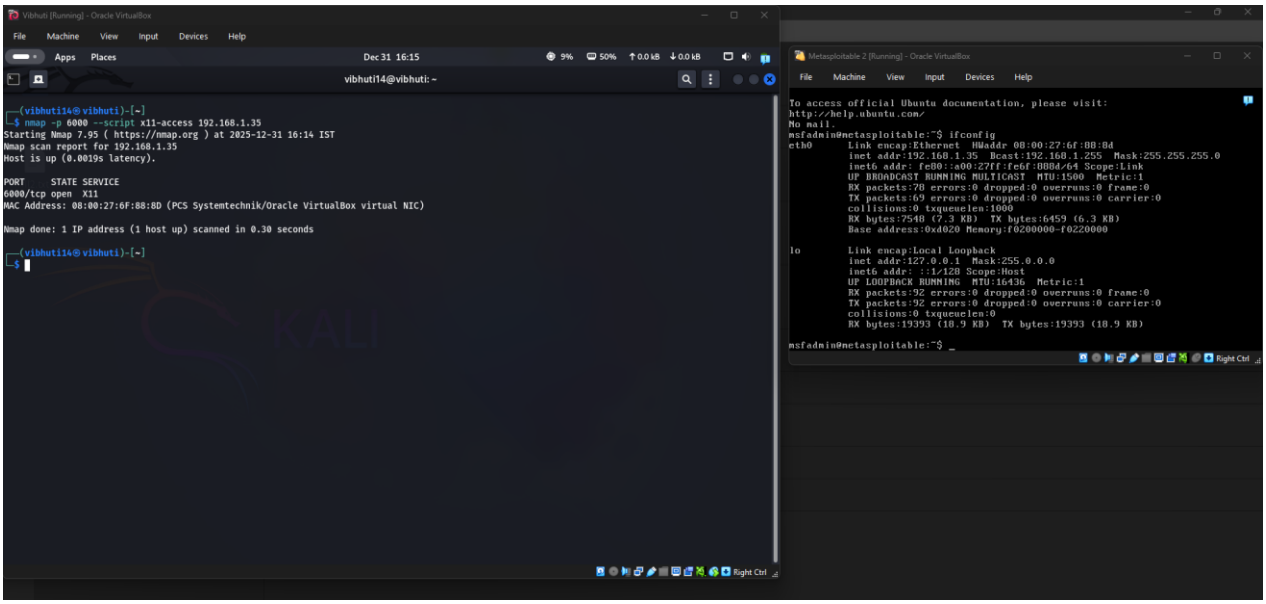
Severity: Critical

Remedial:

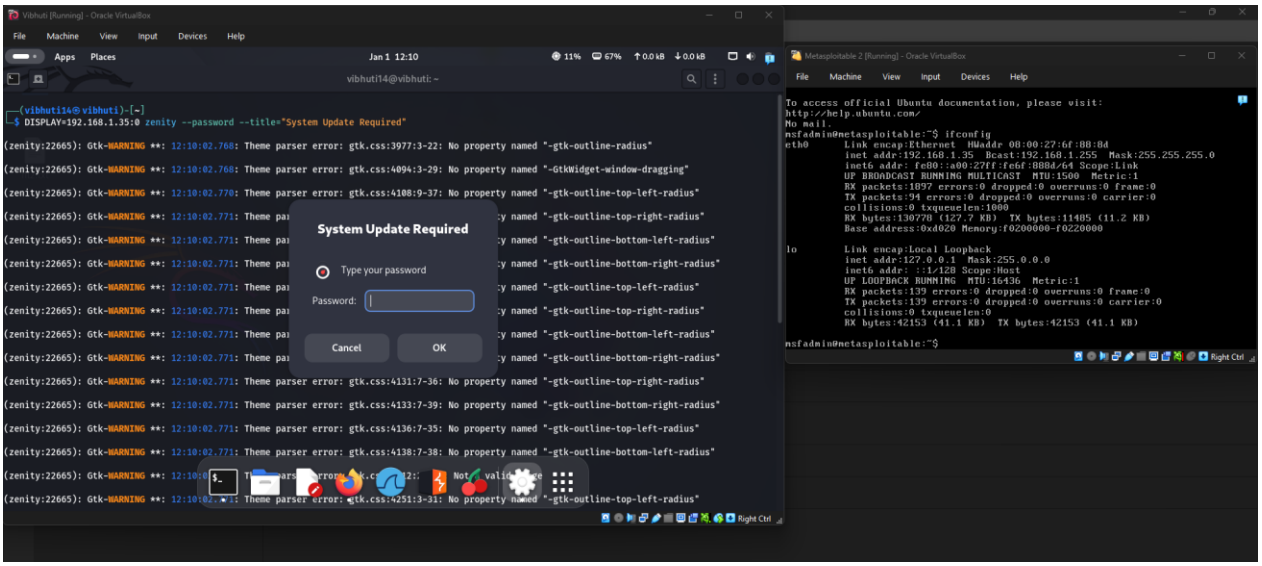
- **Disable TCP Listening:** Modern X servers should be started with the -nolisten tcp flag. This prevents the server from opening Port 6000 entirely, forcing clients to connect via local Unix sockets.
- **Use SSH Forwarding:** Instead of opening Port 6000, use SSH X11 Forwarding (ssh -X or ssh -Y). This tunnels X11 traffic through an encrypted SSH connection, providing security and bypassing the need for an open port.
- **Block Port 6000 at the Firewall:** Ensure that your host-based and network firewalls block all incoming traffic on Port 6000 through 6063 (the range used for multiple displays).
- **Restrict to Localhost:** If the service must be network-accessible, ensure it is bound only to the loopback interface (127.0.0.1) so it cannot be reached from the external network.

POC:

Method 1. Banner Grabbing and Open Access Verification (Nmap)



Method 2. Unauthorized Application Launching (Remote Display)



irc Port 6667

Description:

Internet Relay Chat (IRC) is one of the oldest protocols for real-time text messaging and group communication, traditionally operating on Port 6667. It uses a client-server model where users connect to a central server to join "channels" and exchange messages. Because the protocol was designed in the late 1980s, standard traffic on Port 6667 is unencrypted. Messages, nicknames, and even operator passwords (used to manage channels) are sent in plain text. While largely replaced by modern platforms like Slack or Discord, IRC remains active in technical communities and, unfortunately, is a favorite tool for cybercriminals to manage malicious networks.

Impact:

- **Cleartext Interception:** Since Port 6667 lacks encryption, any attacker on the network path can read private conversations and capture "Opers" (operator) passwords using packet sniffing.
- **Botnet Command & Control (C2):** Hackers frequently use IRC servers to send instructions to infected computers (bots). If an internal machine is communicating over Port 6667, it is often a sign of a malware infection.
- **DDoS Target:** IRC networks are historically prone to massive Distributed Denial of Service (DDoS) attacks, which can destabilize the network hosting the IRC server.
- **IP Disclosure:** By design, many IRC servers reveal the IP addresses of connected users to others in the same channel, making users vulnerable to direct attacks or swatting.

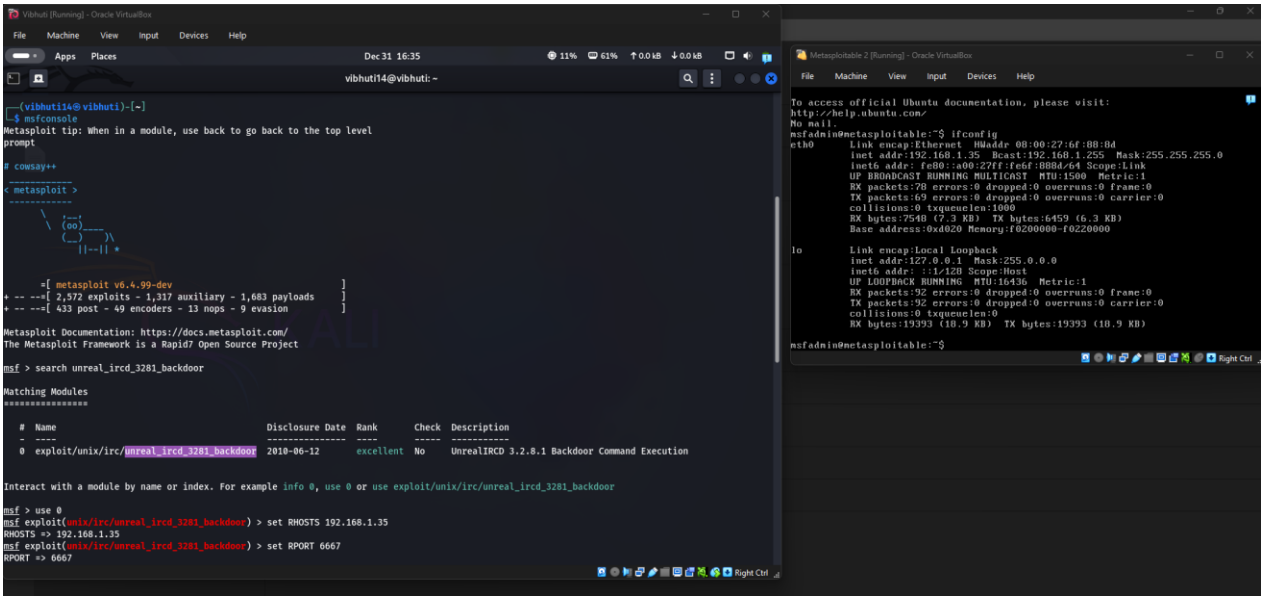
Severity: High

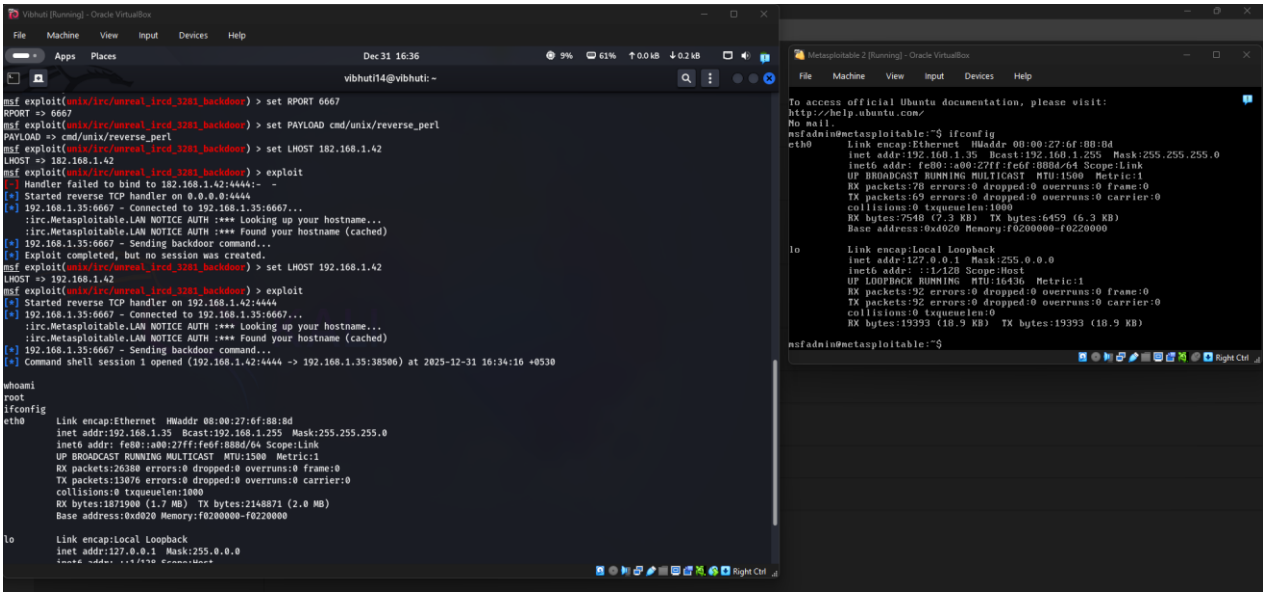
Remedial:

- **Switch to IRC over TLS (Port 6697):** If you must use IRC, ensure you connect via Port 6697, which uses SSL/TLS to encrypt the communication, protecting your data from eavesdropping.
- **Block Port 6667 at the Perimeter:** Most organizations should block outbound traffic on Port 6667 at the firewall to prevent internal infected machines from communicating with external botnet controllers.
- **Use a VPN or Bouncer (ZNC):** To protect your IP address, use a VPN or an IRC Bouncer. A bouncer stays connected to the server for you, masking your actual location.
- **Enable SASL Authentication:** Use Simple Authentication and Security Layer (SASL) to securely identify yourself to the server before you even join a channel.

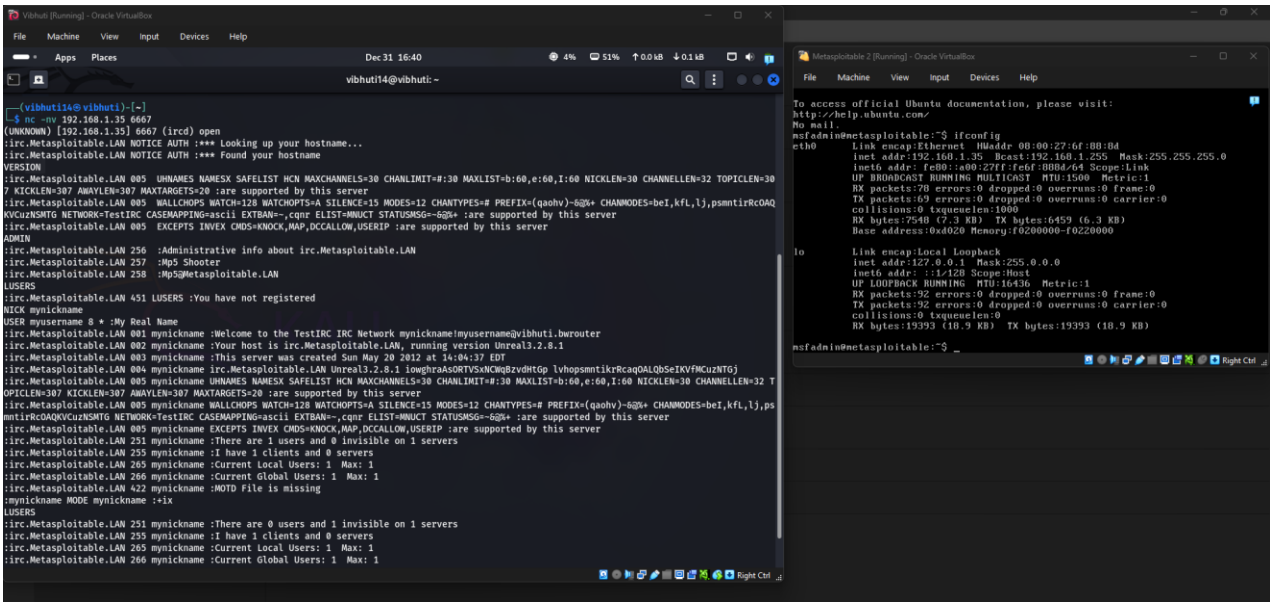
POC:

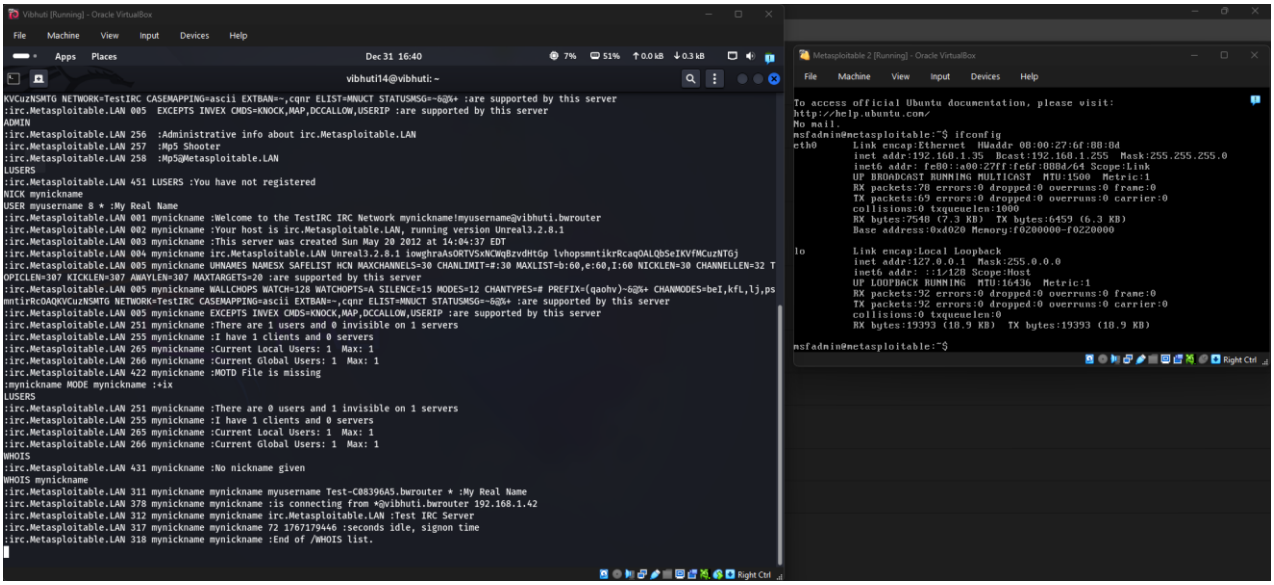
Method 1. The UnrealIRCd Backdoor (Remote Code Execution)



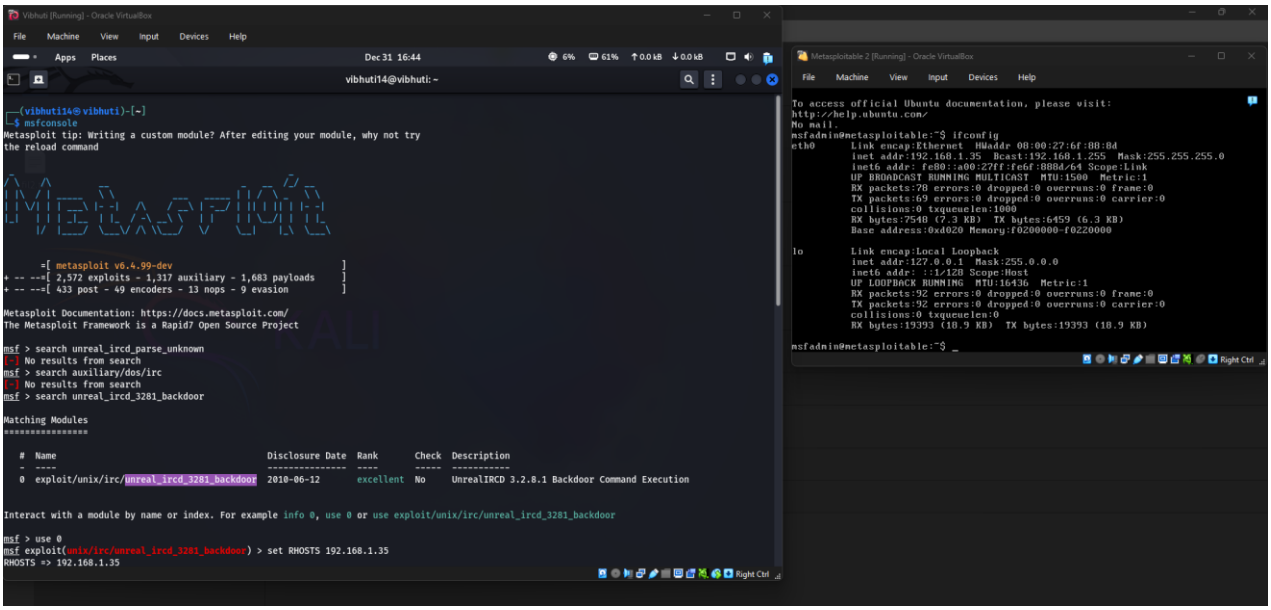


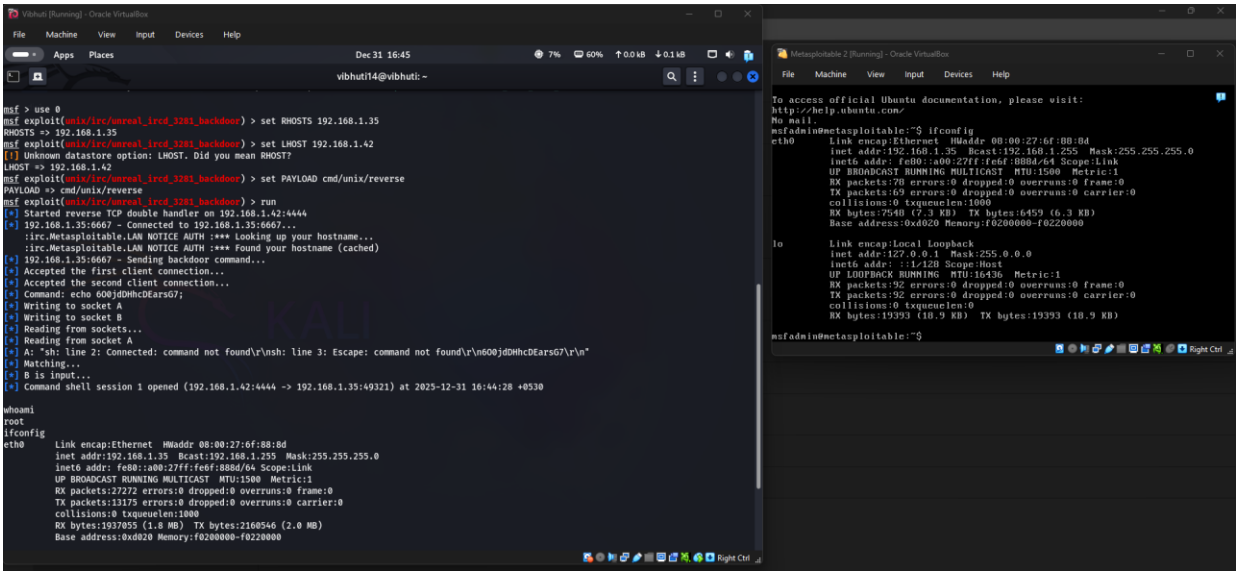
Method 2. IRC Information Gathering and User De-cloaking





Method 3. IRC Service Denial (DoS via Malformed NICK)





ajp13 Port 8009

Description:

The Apache JServ Protocol version 1.3 (AJP13) typically operates on Port 8009. It is a binary, packet-oriented protocol designed to facilitate communication between a front-end web server and a back-end Java application server. Because AJP is a binary protocol, it is faster and more efficient than HTTP for proxying requests, as it avoids the overhead of parsing text-based headers. However, AJP was built with the assumption that the web server and the application server are located within a trusted, secure network.

Impact:

- **Ghostcat Vulnerability (CVE-2020-1938):** This is the most significant impact. An attacker can use the AJP protocol to read any file within the web application directory or, in some cases, achieve Remote Code Execution (RCE).
- **Request Smuggling/Spoofing:** An attacker can craft malicious AJP packets to spoof request attributes, such as the remote IP address or the authenticated user, potentially bypassing application security filters.
- **Information Disclosure:** Attackers can query the port to gain details about the back-end Java environment, version numbers, and internal file paths.
- **Sensitive Data Theft:** Since AJP traffic is usually unencrypted, any attacker on the local network can intercept the binary stream to steal session cookies or personal user data.

Severity: Critical

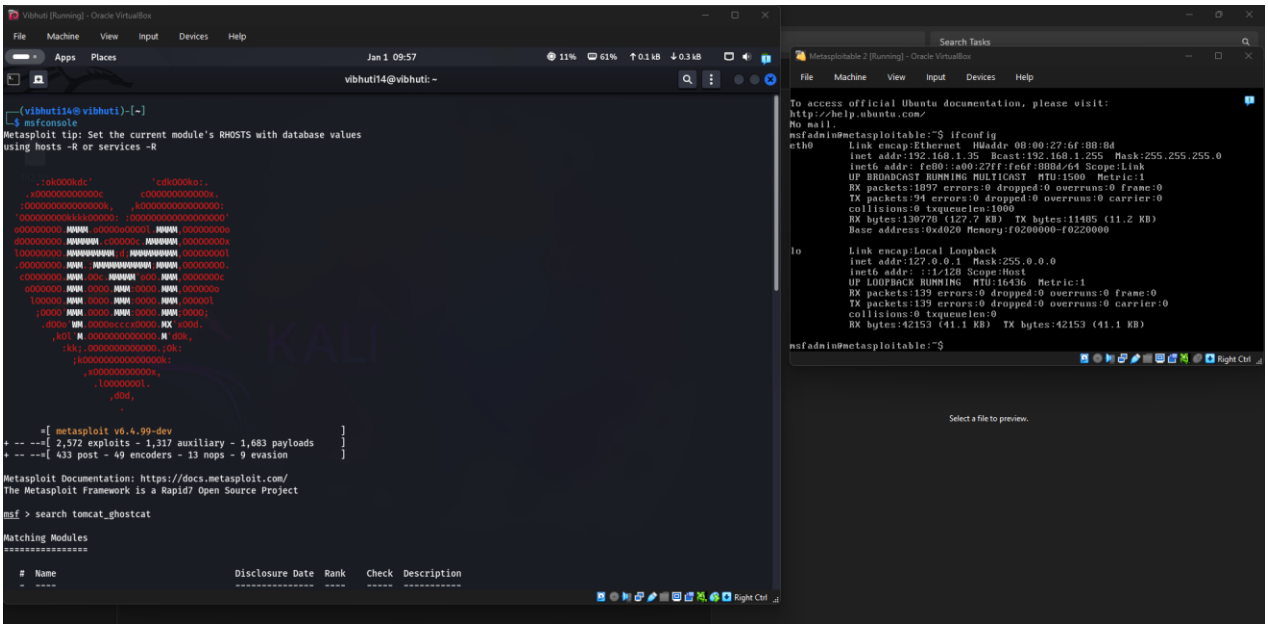
Remedial:

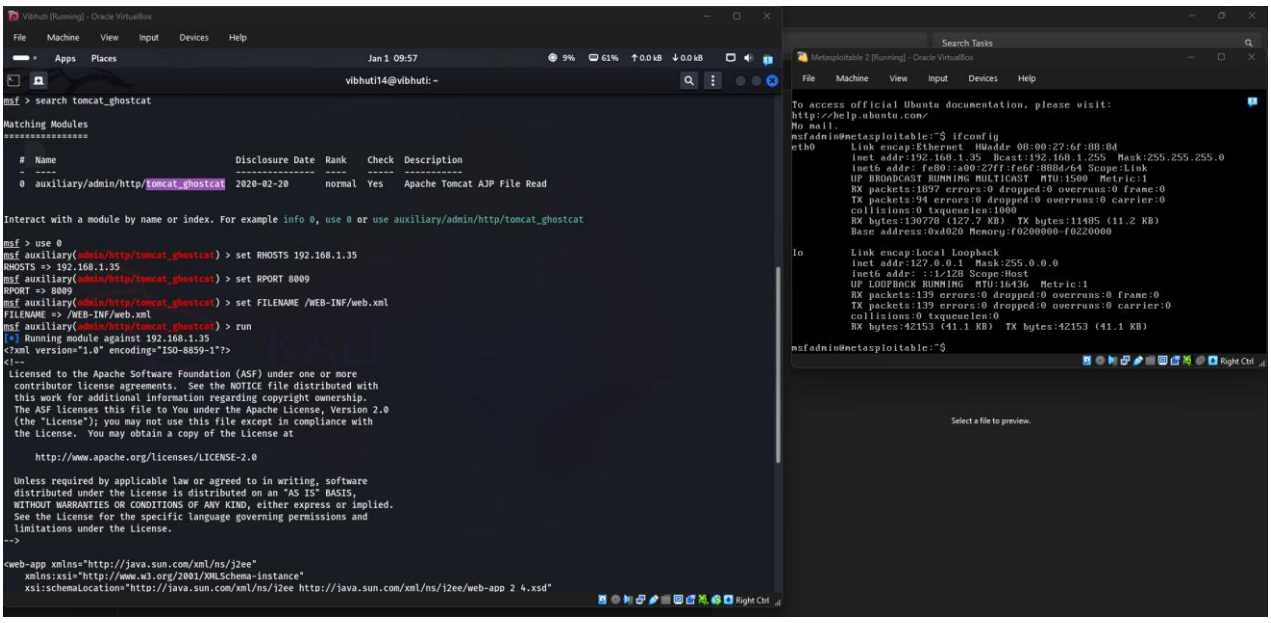
- **Disable AJP if Not Needed:** Many modern setups use mod_proxy_http instead of AJP. If you aren't specifically using the AJP connector, comment out or remove the <Connector port="8009" protocol="AJP/1.3" ... /> line in Tomcat's server.xml.

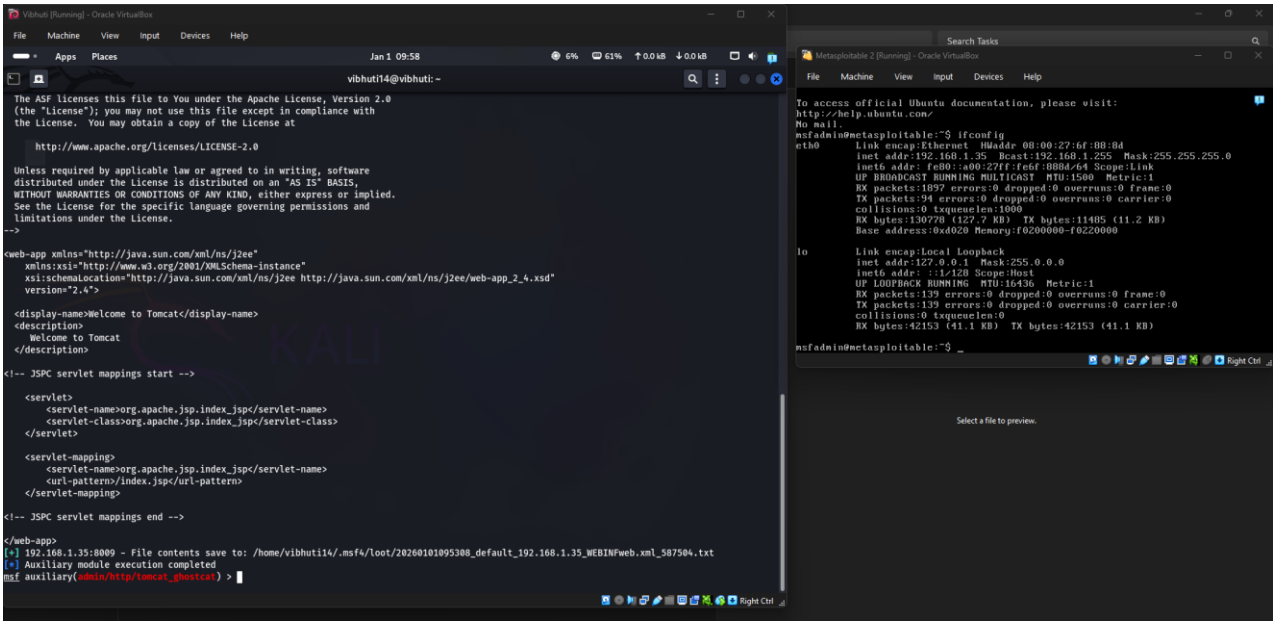
- **Implement the 'requiredSecret' Attribute:** Ensure that the AJP connector is configured with a strong, unique secret (password) that the front-end web server must provide to establish a connection.
- **Enforce Firewall Rules:** Strictly block Port 8009 at the network firewall. Only the IP address of the front-end load balancer or web server should be permitted to communicate with this port.
- **Update Tomcat:** Ensure you are running a version of Tomcat released after February 2020, which includes the patch for the Ghostcat vulnerability and enforces more secure default settings for AJP.

POC:

Method 1. The "Ghostcat" Vulnerability (LFI/Read-Only)







http Port 8180

Description:

Port 8180 is a non-standard port commonly used as an alternative for HTTP traffic, most notably serving as the default management port for the JBoss/WildFly application server or as a secondary instance for Apache Tomcat. Developers often use it to avoid conflicts with the standard Port 80 or Port 8080 when running multiple web services on a single host. Because it is an HTTP port, it typically handles unencrypted traffic. While its non-standard nature provides a minor layer of "security through obscurity" by avoiding the most basic automated scanners, the services listening on this port are often administrative in nature or development environments that may not have the same rigorous security hardening as production systems.

Impact:

- **Cleartext Data Exposure:** Like Port 80, traffic on 8180 is unencrypted. Login credentials for management consoles or sensitive application data can be captured via packet sniffing.
- **Access to Administrative Consoles:** Since 8180 is often used for JBoss or Tomcat management, an attacker who gains access may find an exposed "Manager" or "Admin" dashboard, which can lead to full server takeover.
- **Shadow IT and Configuration Drift:** Use of non-standard ports often bypasses corporate security monitoring tools that are only configured to inspect standard web ports (80/443).

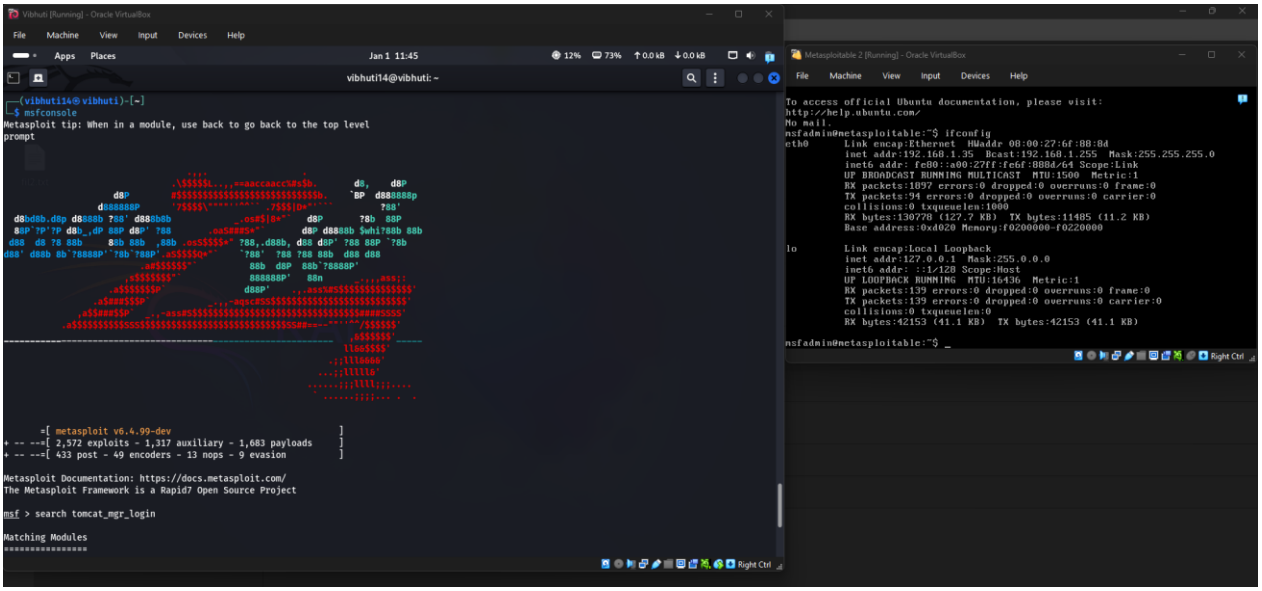
Severity: High

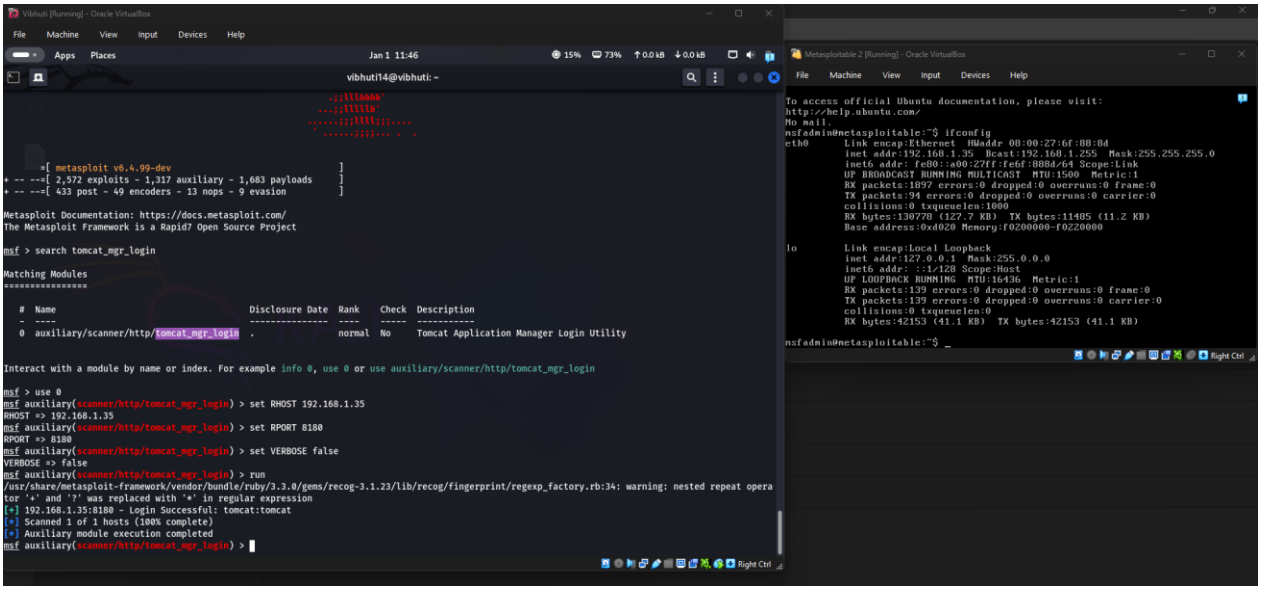
Remedial:

- **Implement TLS/SSL:** Transition the service to a secure port (like 8443) or use a reverse proxy (Nginx/Apache) to wrap the Port 8180 traffic in HTTPS.
- **Restrict Access (IP Whitelisting):** Use a firewall to ensure Port 8180 is only accessible from specific administrative IP addresses or through a secure VPN.
- **Secure the Management Console:** If the port is hosting a JBoss or Tomcat console, ensure that default credentials are changed and that Role-Based Access Control (RBAC) is strictly enforced.
- **Disable If Unused:** If the secondary instance or the development environment is no longer required, shut down the service to reduce the attack surface.
- **Regular Patching:** Keep the underlying application server (Tomcat, WildFly, etc.) updated to the latest version to protect against known vulnerabilities like Remote Code Execution (RCE).

POC:

Method 1. Manager Console Brute-Force (Default Credentials)





Method 2. Exploiting the /manager/text Interface (Scripted Deployment)