

# 网络空间安全综合 课程设计

## 实验报告（六）

学号： 57117137 姓名： 刘康亮

东南大学网络空间安全学院

2020 年 9 月 20 日

## Linux Firewall Exploration Lab

本实验虚拟机网卡设置为桥接网卡，桥接至宿主机无线网卡

### Task1

虚拟机 A 192.168.1.107，虚拟机 B 192.168.1.100

阻断 A 到 B 的 telnet 请求：

```
[09/16/20]seed@VM:~$ sudo ufw deny out on enp0s3 proto tcp to 192.168.1.100 port 23
```

结果如下：

虚拟机 A

```
[09/16/20]seed@VM:~$ telnet 192.168.1.100
Trying 192.168.1.100...
```

宿主机 192.168.1.102：

```
Last login: Thu Sep 17 11:27:01 CST 2020 from 192.168.1.107 on pts/21
```

可以正常访问。

阻断 B 到 A 的 telnet：

```
[09/17/20]seed@VM:~$ sudo ufw deny out on enp0s3 proto tcp to 192.168.1.100 port 23
```

结果如下：

虚拟机 B

```
nie@nie-VirtualBox:~$ telnet 192.168.1.107
Trying 192.168.1.107...
```

宿主机：

```
192.168.1.107 - PuTTY
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Wed Sep 16 23:33:12 EDT 2020 from 192.168.1.100 on pts/19
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

1 package can be updated.
0 updates are security updates.

[09/17/20]seed@VM:~$
```

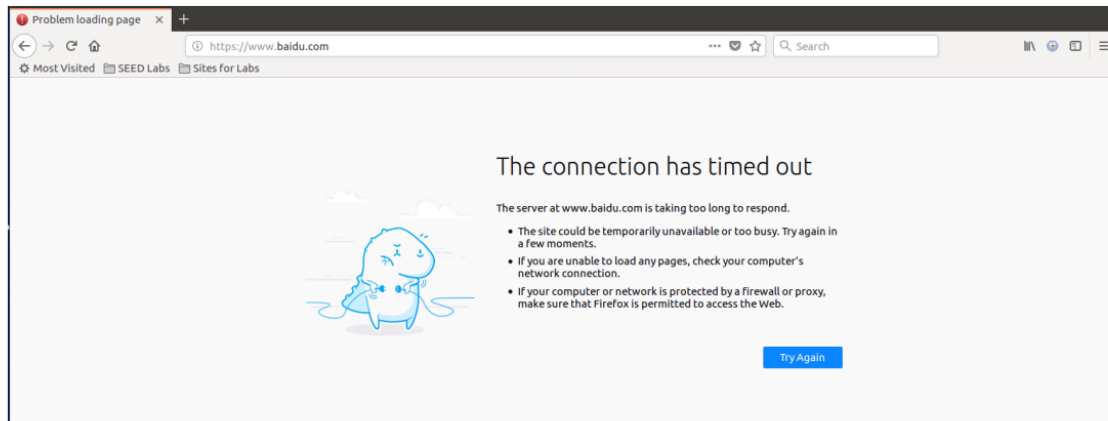
可以正常访问

阻断对百度的访问。通过 dig 命令查到 www.baidu.com 有两个 ip：180.101.49.11 和 180.101.49.12

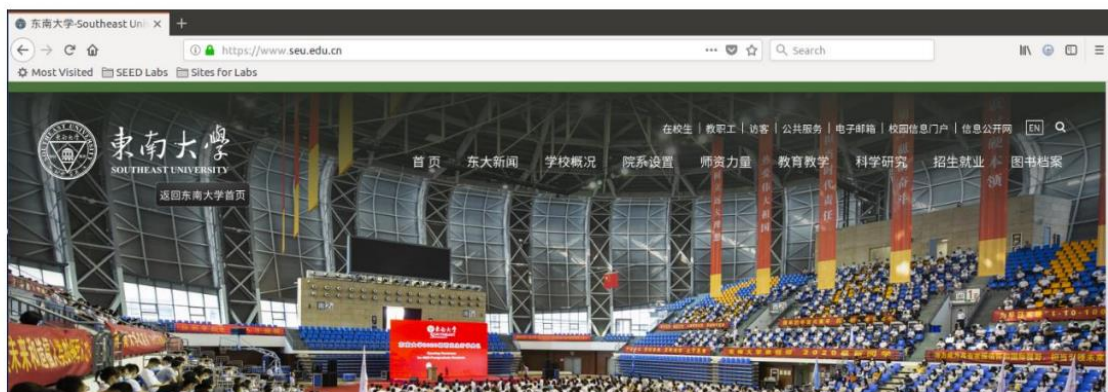
那么我们的命令如下：

```
[09/17/20]seed@VM:~$ sudo ufw deny out on enp0s3 proto tcp to 180.101.49.0/24 port 80
Rule added
[09/17/20]seed@VM:~$ sudo ufw deny out on enp0s3 proto tcp to 180.101.49.0/24 port 443
Rule added
[09/17/20]seed@VM:~$
```

对发向 180.101.49.0/24 整个网段的 80 和 443 端口的包都 deny 掉



其他网站:



可以正常访问。

## Task2

先关掉 ufw

虚拟机 A: 192.168.1.107 即我们使用 netfilter 的机子。

虚拟机 B: 192.168.1.100

规则为如下五条:

Task1 中的三条: 1、禁止 A 到 B 的 telnet; 2、禁止 B 到 A 的 telnet; 3、禁止 A 对百度的访问。

其余两条: 4、B ping A 无响应; 5、禁止 B 对 A 的 web 访问

代码如下:

```

struct iphdr *iph;
struct tcphdr *tcph;
struct icmphdr *icmph;
iph=ip_hdr(skb);
icmph=(void *)iph+iph->ihl*4;
tcph=(void *)iph+iph->ihl*4;
if(((unsigned char *)&iph->saddr)[0]==192&&((unsigned char *)&iph->saddr)[1]==168&&((unsigned char *)&iph->saddr)[2]==1&&((unsigned char *)&iph->saddr)[3]==100
&&iph->protocol==IPPROTO_TCP &&(tcph->source==htons(23)||tcph->dest==htons(23)))
{
    printk("drop1\n");
    return NF_DROP;
}
else if(((unsigned char *)&iph->saddr)[0]==180&&((unsigned char *)&iph->saddr)[1]==101&&((unsigned char *)&iph->saddr)[2]==49
&&iph->protocol==IPPROTO_TCP &&(tcph->source==htons(80)||tcph->source==htons(443)))
{
    printk("drop2\n");
    return NF_DROP;
}
else if(((unsigned char *)&iph->saddr)[0]==192&&((unsigned char *)&iph->saddr)[1]==168&&((unsigned char *)&iph->saddr)[2]==1&&((unsigned char *)&iph->saddr)[3]==100
&&iph->protocol==IPPROTO_ICMP &&icmph->type==8)
{
    printk("drop3\n");
    return NF_DROP;
}
else if(((unsigned char *)&iph->saddr)[0]==192&&((unsigned char *)&iph->saddr)[1]==168&&((unsigned char *)&iph->saddr)[2]==1&&((unsigned char *)&iph->saddr)[3]==100
&&iph->protocol==IPPROTO_TCP &&tcph->dest==htons(80))
{
    printk("drop4\n");
    return NF_DROP;
}
else
    return NF_ACCEPT;

```

这里主要是最核心的判断和处理部分

与示例框架代码一样，只使用了一个 hook。

想要使用 htons 等函数此处需要#include<linux/inet.h>。

不过，我选择的 hooknum 是 NF\_INET\_LOCAL\_IN。那么在这个位置的话，可以只对发给自己的包进行检查。上面的五条规则对应实现方法就是：1、扔掉 B 传给 A 的，源端口是 23 的包；2、扔掉 B 传给 A 的，目的端口是 23 的包；3、扔掉 180.101.49 为源地址前三部分，源端口是 80 和 443 的包（扔掉 http/https 的回复包）；4、扔掉 B 传给 A 的，icmp type=8（请求）的包；5、扔掉源地址为 B，目的端口为 80 的包。（B 发给 A 的 http 请求）

其中，前两条缩在了一个 if 块内。

测试：

第一、二规则

```

[09/18/20]seed@VM:~$ telnet 192.168.1.100
Trying 192.168.1.100...

```

```

nie@nie-VirtualBox:~$ telnet 192.168.1.107
Trying 192.168.1.107...

```

AB 互相无法使用 telnet

第三规则：

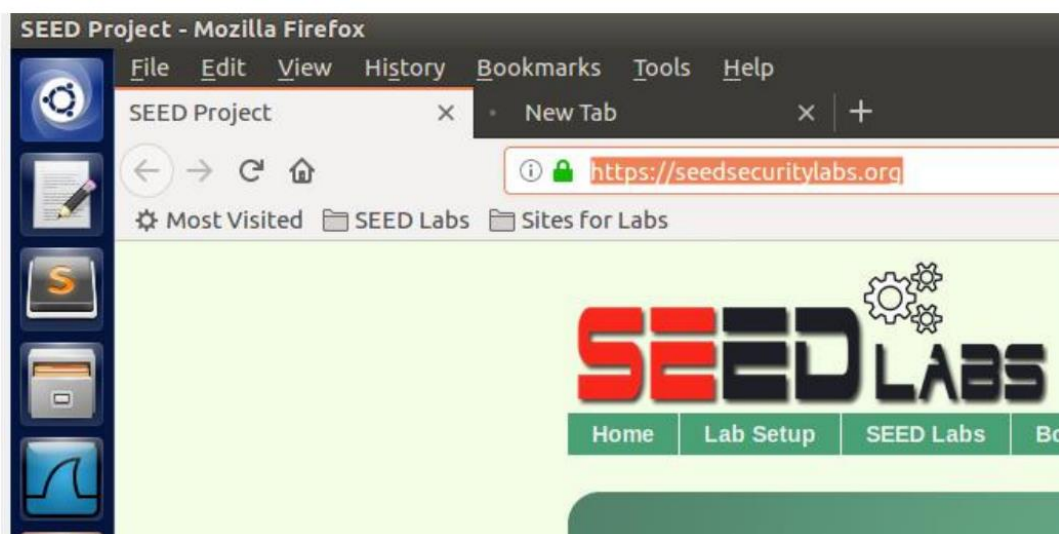
虚拟机 A 上尝试访问百度

Inspector Console Debugger Style Editor Performance Memory Network Storage											
All HTML CSS JS XHR Fonts Images Media WS Other Persist Logs Disable cache Filter URLs											
Sta...	Meth...	File	Domain	Cause	Type	Transfer...	Size	0 ms	80 ms	160 ms	240 ms
GET	/		www.baidu.com		document						

无回应



seed 主页可以访问:



第四规则:

B ping A;

```
nie@nie-VirtualBox:~$ ping 192.168.1.107
PING 192.168.1.107 (192.168.1.107) 56(84) bytes of data.
^C
--- 192.168.1.107 ping statistics ---
11 packets transmitted, 0 received, 100% packet loss, time 10227ms
nie@nie-VirtualBox:~$
```

A ping B:

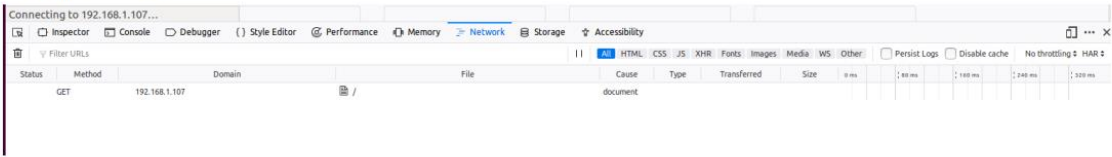
```
[09/18/20]seed@VM:~$ ping 192.168.1.100
PING 192.168.1.100 (192.168.1.100) 56(84) bytes of data.
64 bytes from 192.168.1.100: icmp_seq=1 ttl=64 time=0.386 ms
64 bytes from 192.168.1.100: icmp_seq=2 ttl=64 time=0.304 ms
64 bytes from 192.168.1.100: icmp_seq=3 ttl=64 time=0.337 ms
64 bytes from 192.168.1.100: icmp_seq=4 ttl=64 time=0.184 ms
^C
--- 192.168.1.100 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3069ms
rtt min/avg/max/mdev = 0.184/0.302/0.386/0.077 ms
[09/18/20]seed@VM:~$
```

第五规则:

宿主机访问虚拟机 A 的 web 服务:



虚拟机 B 访问：



无响应。

dmesg 中也可以看到程序中输出的提示信息：

```
[ 3541.242535] drop2
[ 3541.493812] drop2
[ 3546.864886] drop2
[ 3593.885854] drop3
[ 3594.897223] drop3
[ 3595.921002] drop3
[ 3596.945810] drop3
[ 3597.969344] drop3
[ 3598.992960] drop3
[ 3600.017570] drop3
[ 3601.041686] drop3
[ 3602.065173] drop3
[ 3603.089703] drop3
[ 3604.113545] drop3
[ 3737.003523] drop4
[ 3795.698767] drop4
[ 3795.950184] drop4
[ 3796.721685] drop4
[ 3796.977040] drop4
[ 3860.193574] drop4
[ 3860.235725] drop4
```

```
[ 3012.401804] drop1
[ 3016.429262] drop1
[ 3024.434120] drop1
[ 3034.835484] drop1
[ 3035.856886] drop1
[ 3037.877610] drop1
[ 3040.565393] drop1
[ 3042.097187] drop1
[ 3050.301670] drop1
[ 3338.110486] drop2
[ 3338.245308] drop2
[ 3338.412052] drop2
[ 3338.561329] drop2
[ 3339.059402] drop2
[ 3339.125319] drop2
[ 3339.250468] drop2
```

### Task3

关掉/卸载之前所有防护措施

设置防火墙的虚拟机为 192.168.1.107

我们将题目中的访问 facebook 改为访问 baidu

程序如下：

```
unsigned int hook_func(void *priv, struct sk_buff *skb,
const struct nf_hook_state *state)
{
    struct iphdr *iph;
    struct tcphdr *tcph;
    struct icmphdr *icmph;
    iph=ip_hdr(skb);
    icmph=(void *)iph+iph->ihl*4;
    tcph=(void *)iph+iph->ihl*4;
    if(iph->protocol==IPPROTO_TCP &&tcph->dest==htons(23))
    {
        printk("drop1\n");
        return NF_DROP;
    }
    else if(((unsigned char *)&iph->daddr)[0]==180&&((unsigned char *)&iph->daddr)[1]==101&&((unsigned char *)&iph->daddr)[2]==49
&&iph->protocol==IPPROTO_TCP &&(tcph->dest==htons(80)||tcph->dest==htons(443)))
    {
        printk("drop2\n");
        return NF_DROP;
    }
    else
        return NF_ACCEPT;
}

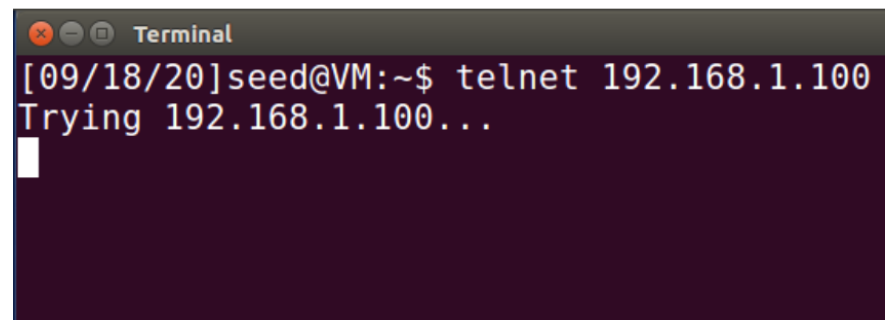
/* Initialization routine */
int init_module()
{
    /* Fill in our hook structure */
    nfhoout.hook = hook_func; /* Handler function */
    nfhoout.hooknum = NF_INET_LOCAL_OUT; /* First hook for IPv4 */
    nfhoout.pf = PF_INET;
    nfhoout.priority = NF_IP_PRI_FIRST; /* Make our function first */
    nf_register_hook(&nfhoout);
    return 0;
}

/* Cleanup routine */
void cleanup_module()
{
    nf_unregister_hook(&nfhoout);
}
```

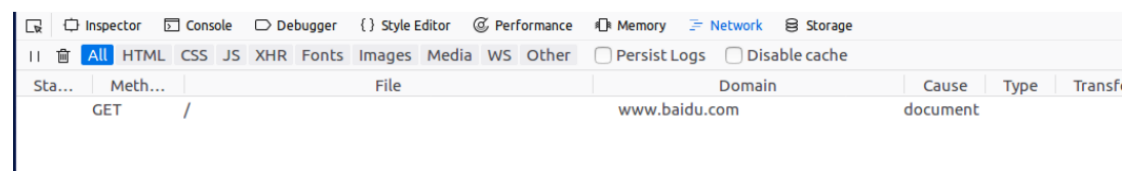
将 hooknum 改为 NF\_INET\_LOCAL\_OUT

阻断所有 telnet 直接检查 tcp 端口即可

阻断对百度的访问将 http 请求丢弃即可。



无法与 192.168.1.100 建立 telnet 连接



无法访问百度

### 3.a

我们要在设置防火墙的虚拟机 A 192.168.1.107 上，通过与虚拟机 B 192.168.1.104 建立 ssh 连接，实现与 C 192.168.1.100 建立 telnet 连接

先与 192.168.1.104 按照实验指导建立连接：

```
[09/18/20]seed@VM:~$ ssh -L 8000:192.168.1.100:23 nie@192.168.1.104
nie@192.168.1.104's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.15.0-76-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

 * Kubernetes 1.19 is out! Get it in one command with:

   sudo snap install microk8s --channel=1.19 --classic

   https://microk8s.io/ has docs and details.

Last login: Fri Sep 18 17:53:51 2020 from 192.168.1.107
nie@nie-VirtualBox:~$ ls
Desktop Documents Downloads Music Pictures Public Templates Videos
nie@nie-VirtualBox:~$
```

再按照实验指导，与 192.168.1.100 建立 telnet（另启终端窗口）：

```
[09/18/20]seed@VM:~$ telnet localhost 8000
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Ubuntu 16.04.6 LTS
nie-VirtualBox login: nie
Password:
Last login: Fri Sep 18 17:55:46 CST 2020 from 192.168.1.104 on pts/19
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.15.0-45-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 个可升级软件包。
0 个安全更新。

New release '18.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

nie@nie-VirtualBox:~$ ls
eclipse          lab3             php-7.0.4.tar.gz  server.key       模板  音乐
examples.desktop main              Qt                t1.pcap          视频  桌面
freetds-1.1.40   main.cpp         qtcreator-4.13.0  tfile            图片
freetds-1.1.40.tar.gz main.o           server.crt         t.pcap           文档
javacode         php-7.0.4        server.csr         公共的          下载
```

这两台主机名和用户名重合了，不过老师可以看用户主目录下文件不一致来区分。

在看 wireshark：

当在执行 telnet 终端窗口中输入字符'c'时：

1655	2020-09-18 06:21:01.991277209	192.168.1.107	192.168.1.104	SSHv2	102 Client: Encrypted packet (len=36)
1656	2020-09-18 06:21:01.991806444	192.168.1.104	192.168.1.100	TELNET	67 Telnet Data ...
1657	2020-09-18 06:21:01.992088710	192.168.1.100	192.168.1.104	TELNET	67 Telnet Data ...
1658	2020-09-18 06:21:01.992203728	192.168.1.104	192.168.1.100	TCP	66 58808 → 23 [ACK] Seq=1206995326 Ack=3778006493 W
1659	2020-09-18 06:21:01.992322392	192.168.1.104	192.168.1.107	SSHv2	102 Server: Encrypted packet (len=36)
1660	2020-09-18 06:21:01.992330780	192.168.1.107	192.168.1.104	TCP	66 45554 → 22 [ACK] Seq=44441345 Ack=4038725785 Win

有这样 6 个包：

根据 IP 应该是

第一个包，虚拟机 A 将字符发给 B，

中间三个包：虚拟机 B 与 C 的 telnet 交互，与普通 telnet 交互时一致，把字符发过去，字符再传回来用于显示，然后 ack 确认收到。

最后两个包，B 把该字符传回 A 以用于显示，A ack 确认收到

本地环回卡中也会有三个包，看内容类似于 telnet 的三个包交互的过程，应该是 telnet 所在终端与 localhost: 8000 的交互过程



1	2020-09-18 06:26:09.158235098	127.0.0.1	127.0.0.1	TCP	67 54382 → 8090 [PSH, ACK] Seq=2225712873 Ack=
2	2020-09-18 06:26:09.161935250	127.0.0.1	127.0.0.1	TCP	67 8090 → 54382 [PSH, ACK] Seq=3235834876 Ack=
3	2020-09-18 06:26:09.161946749	127.0.0.1	127.0.0.1	TCP	66 54382 → 8090 [ACK] Seq=2225712874 Ack=32358

其实就能使用telnet的目的而言,直接在与B的ssh交互终端中执行命令telnet 192.168.1.100 (即终端套终端)即可。

### 3.b

首先在加载动态模块情况下:

Inspector Console Debugger {} Style Editor Performance Memory Network Storage									
All HTML CSS JS XHR Fonts Images Media WS Other Persist Logs Disable cache									
Sta...	Meth...	File	Domain	Cause	Type	Transfer...	S		
GET	/		www.baidu.com	document					

无法访问百度

根据实验手册设置后执行命令:

```
[09/18/20]seed@VM:~/.../task3$ ssh -D 9000 -C nie@192.168.1.104
nie@192.168.1.104's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.15.0-76-generic x86_64)

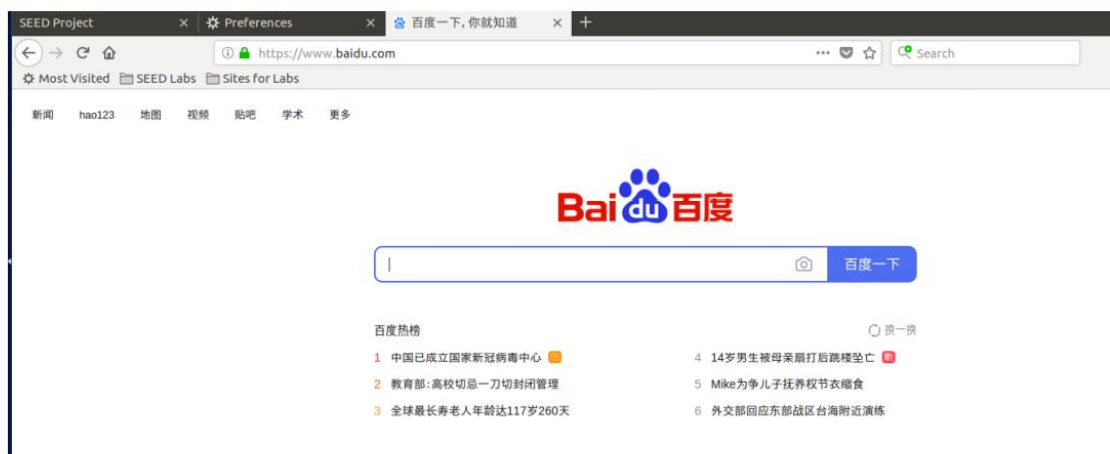
* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:       https://ubuntu.com/advantage

* Kubernetes 1.19 is out! Get it in one command with:

    sudo snap install microk8s --channel=1.19 --classic

https://microk8s.io/ has docs and details.

Last login: Fri Sep 18 17:59:04 2020 from 192.168.1.107
nie@nie-VirtualBox:~$
```



浏览器可访问百度了。

192.168.1.104	180.101.49.11	TCP	74 52242 → 443 [SYN] Seq=720978145 Win=65535 Len=0 MSS=1460 SACK
180.101.49.11	192.168.1.104	TCP	74 443 → 52242 [SYN, ACK] Seq=3586809469 Ack=720978146 Win=8192

Wireshark 可以看到虚拟机 B 对百度发起请求,当然这之前有大量 A 与 B 的通信,都是加密报文,之后则是 A 与 B, B 与百度报文的交杂。

断开 ssh 后,浏览器显示



## The proxy server is refusing connections

Firefox is configured to use a proxy server that is refusing connections.

- Check the proxy settings to make sure that they are correct.
- Contact your network administrator to make sure the proxy server is working.

### Task4

关掉之前所有防护措施

在虚拟机 A(192.168.1.107)新的 netfilter 模块如下:

```
unsigned int hook_func(void *priv, struct sk_buff *skb,
const struct nf_hook_state *state)
{
    struct iphdr *iph;
    struct tcphdr *tcph;
    struct icmphdr *icmph;
    iph=ip_hdr(skb);
    icmph=(void *)iph+iph->ihl*4;
    tcph=(void *)iph+iph->ihl*4;
    if(iph->protocol==IPPROTO_TCP &&tcph->dest==htons(22))
    {
        printk("drop1\n");
        return NF_DROP;
    }
    else if(((unsigned char *)&iph->saddr)[0]==192&&((unsigned char *)&iph->saddr)[1]==168&&((unsigned char *)&iph->saddr)[2]==1&&((unsigned char *)&iph->protocol==IPPROTO_TCP &&(tcph->dest==htons(80)||tcph->dest==htons(443)))
    {
        printk("drop2\n");
        return NF_DROP;
    }
    else
        return NF_ACCEPT;
}

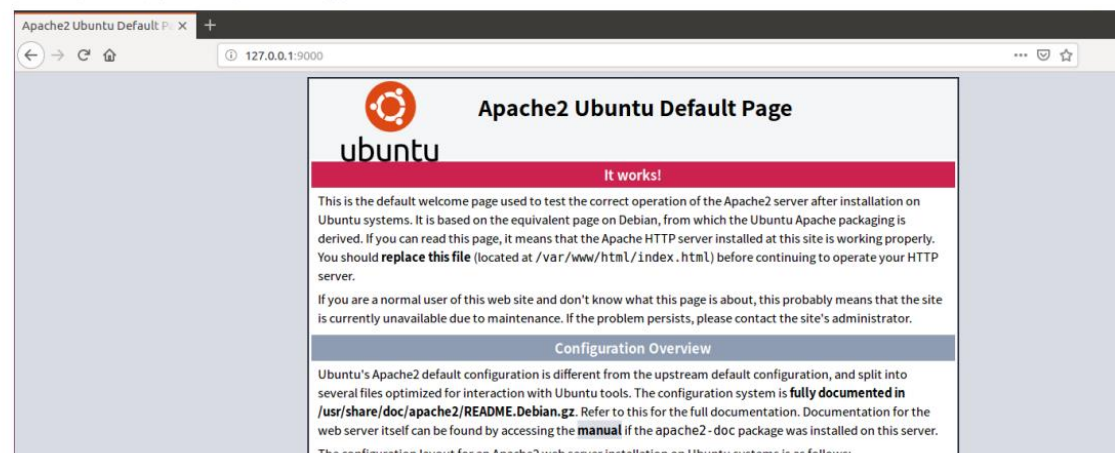
/* Initialization routine */
int init_module()
{ /* Fill in our hook structure */
    nfhoout.hook = hook_func; /* Handler function */
    nfhoout.hooknum = NF_INET_LOCAL_IN; /* First hook for IPv4 */
    nfhoout.pf = PF_INET;
    nfhoout.priority = NF_IP_PRI_FIRST; /* Make our function first */
    nf_register_hook(&nfhoout);
    return 0;
}
```

禁止所有外连内的 ssh 服务, 禁止虚拟机 B(192.168.1.100)访问 web。

在虚拟机 A (192.168.1.107) 中使用如下命令:

```
ssh -R 9000:127.0.0.1:80 nie@192.168.1.100
```

然后在虚拟机 B 浏览器中输入 127.0.0.1:9000



拿到虚拟机 A 的网页