

中南大学

《网络安全》实验报告

学生姓名_____邹昆池_____

指导教师_____王伟平_____

学 院_____信息科学与工程学院_____

专业班级_____信息安全 1402_____

完成时间_____2016. 12. 25_____

实验一 TCP/IP 攻击

一、实验原理

由于 TCP/IP 协议是 Internet 的基础协议，所以对 TCP/IP 协议的完善和改进是非常必要的。TCP/IP 协议从开始设计时候并没有考虑到现在网络上如此多的威胁,由此导致了形形色色的攻击方法，一般如果是针对协议原理的攻击(尤其 DDOS)，我们将无能为力。

TCP/IP 攻击的常用原理有：

源地址欺骗(Source Address Spoofing)、IP 欺骗(IP Spoofing)和 DNS 欺骗(DNS Spoofing).

路由选择信息协议攻击(RIP Attacks)

源路由选择欺骗(Source Routing Spoofing)

TCP 序列号欺骗和攻击(TCP Sequence Number Spoofing and Attack)

二、实验目的

基于 TCP/IP 协议进行攻击实验,了解 TCP/IP 协议的具体机制。

三、实验环境

三台虚拟机是使用相同的 Ubuntu 版本，以免出现冲突

虚拟机使用信息： VirtualBox 与 Vmware

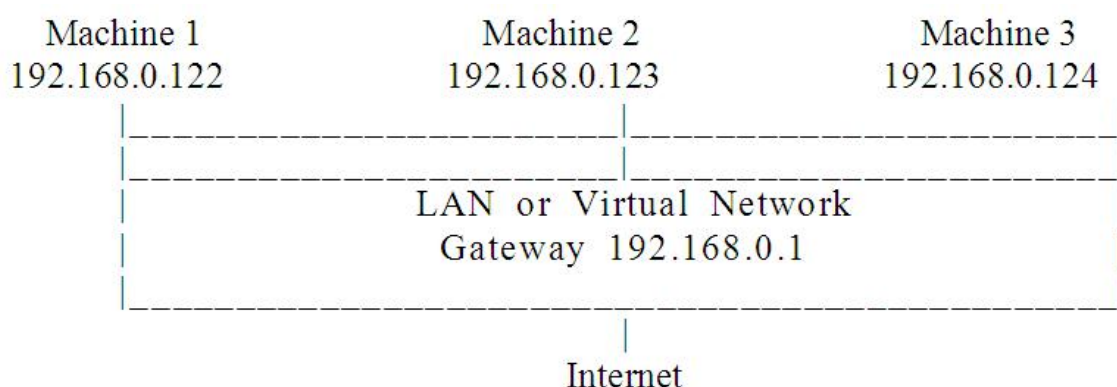
Linux 版本信息： Ubuntu 12.04.2 (Final)

Linux 内核版本： 3.8.0

四、实验步骤

预备工作

这里我们三个同学分别使用三台机器在虚拟机环境下做实验，其中一个用于攻击；另一个用于被攻击；第三个作为观察者使用；我们把三台主机放在同一个 LAN 中，其配置信息参照如下所示（实际在实验过程中有所改动）：



这里我们使用的是用于先前实验的 Ubuntu 系统，并且参照网上教程安装相关的 netwox 工具箱和 Wireshark 工具箱，并且利用它们完成接下来的任务，与此同时三台虚拟机都需要打开 FTP 和 Telnet

服务：

——使用如下命令来完成上述任务

Start the ftp server

```
# service vsftpd start
```

Start the telnet server

```
# service openbsd-inetd start
```

实验任务第一项：ARP 缓存中毒（ARP cache poisoning）

【背景知识】

ARP 缓存是 ARP 协议的重要组成部分。ARP 协议运行的目标就是建立 MAC 地址和 IP 地址的映射，然后把这一映射关系保存在 ARP 缓存中，使得不必重复运行 ARP 协议。因为 ARP 缓存中的映射表并不是一直不变的，主机会定期发送 ARP 请求来更新它的 ARP 映射表，利用这个机制，攻击者可以伪造 ARP 应答帧使得主机错误的更新自己的 ARP 映射表，这个过程就是 ARP 缓存中毒。

这样的后果，要么使主机发送 MAC 帧到错误的 MAC 地址，导致数据被窃听；要么由于 MAC 地址不存在，导致数据发送不成功。

【操作流程】

当前网络配置情况如下面所示：

【备注】

这里我们的 IP 由于使用相同虚拟机配置，一开始是一样的，可是经过一系列调整之后，我们实现了在同一子网内并且 IP 不同，同时可以相互 Ping 成功！

同时，由于只要是一个子网内，那么虚拟机的 IP 是可以在规定范围内任意修改的，为了方便起见，我们同意把 3 个 IP 改成连续相邻的 3 个 IP 地址（从 122 到 124）

虚拟机 1 号：

MAC: 08:00:27:ea:ab:26

IP: 192.168.0.122

虚拟机 2 号：

MAC: 08:00:27:c6:57:e5

IP: 192.168.0.123

虚拟机 3 号：

MAC: 08:00:27:5f:ae:ef

IP: 192.168.0.124

我们是基于这样的原理：

通常情况下，如果 2 号机要向 3 号机请求建立 Telnet 链接，2 号机就会需要知道 3 号机的 Mac 地址是多少，于是它会通过广播 ARP 请求来寻求回应；

3 号机如果发现 ARP 广播的 IP 地址恰好是自己 IP，那么它就会发送 ARP 相应包，通知请求方也就是 2 号机自己（3 号机）的 MAC 地址。

在此基础上，两者建立好连接并且开始进行通信！

——所截得的最终效果示意图如下所示：

	Time	Source	Destination	Protocol	Length	Info
47	38.288914	CadmusCo_c6:57:e5	Broadcast	ARP	42	Who has 192.168.0.124? Tell 192.168.0.123
48	38.289505	CadmusCo_5f:ae:ef	CadmusCo_c6:57:e5	ARP	60	192.168.0.124 is at 08:00:27:5f:ae:ef
49	38.289514	192.168.0.123	192.168.0.124	TCP	74	60736 > telnet [SYN] Seq=0 Win=14600 Len=0
50	38.289685	192.168.0.124	192.168.0.123	TCP	74	telnet > 60736 [SYN, ACK] Seq=0 Ack=1 Win=
51	38.289733	192.168.0.123	192.168.0.124	TCP	66	60736 > telnet [ACK] Seq=1 Ack=1 Win=14600
52	38.290136	192.168.0.123	192.168.0.124	TELNET	93	Telnet Data ...
53	38.290179	192.168.0.124	192.168.0.123	TCP	66	telnet > 60736 [ACK] Seq=1 Ack=28 Win=1448

现在 1 号机伪造 3 号机的 ARP 应答，写上自己的 MAC 地址，这样的话，当 2 号机向 3 号机建立 Telnet

链接时，由于 ARP 缓存中 IP 为 192.168.111.139 对的 MAC 地址是 1 号机的，所以 2 号机无法发送数据到 3 号机，Telnet 失败。

【攻击过程示意图】

1 号机伪造 3 号机的 ARP 应答并且进行周期性的广播，实验截图如下

```
Title: Periodically send ARP replies
Usage: netwox 80 -e eth -i ip [-d device] [-E eth] [-I ip] [-s uint32]
Parameters:
-e|--eth eth          ethernet address {08:00:27:EA:AB:26}
-i|--ip ip            IP address {192.168.0.122}
-d|--device device    device for spoof {Eth0}
-E|--eth-dst eth      to whom answer {0:8:9:a:b:c}
-I|--ip-dst ip        to whom answer {5.6.7.8}
-s|--sleep uint32     sleep delay in ms {1000}
--help2              display full help
Example: netwox 80 -e "08:00:27:EA:AB:26" -i "192.168.0.122"
Example: netwox 80 --eth "08:00:27:EA:AB:26" --ip "192.168.0.122"
```

Filter:			Expression...	Clear	Apply
Time	Source	Destination	Protocol	Length	Info
1 0.000000	CadmusCo_ea:ab:26	Broadcast	ARP	60	192.168.0.124 is at 08:00:27:ea:ab:26
2 1.000596	CadmusCo_ea:ab:26	Broadcast	ARP	60	192.168.0.124 is at 08:00:27:ea:ab:26
3 2.002241	CadmusCo_ea:ab:26	Broadcast	ARP	60	192.168.0.124 is at 08:00:27:ea:ab:26
4 3.004137	CadmusCo_ea:ab:26	Broadcast	ARP	60	192.168.0.124 is at 08:00:27:ea:ab:26
5 4.005290	CadmusCo_ea:ab:26	Broadcast	ARP	60	192.168.0.124 is at 08:00:27:ea:ab:26
6 5.005723	CadmusCo_ea:ab:26	Broadcast	ARP	60	192.168.0.124 is at 08:00:27:ea:ab:26
7 6.006637	CadmusCo_ea:ab:26	Broadcast	ARP	60	192.168.0.124 is at 08:00:27:ea:ab:26
8 7.006618	CadmusCo_ea:ab:26	Broadcast	ARP	60	192.168.0.124 is at 08:00:27:ea:ab:26
9 8.006734	CadmusCo_ea:ab:26	Broadcast	ARP	60	192.168.0.124 is at 08:00:27:ea:ab:26
10 9.008130	CadmusCo_ea:ab:26	Broadcast	ARP	60	192.168.0.124 is at 08:00:27:ea:ab:26
11 10.008663	CadmusCo_ea:ab:26	Broadcast	ARP	60	192.168.0.124 is at 08:00:27:ea:ab:26
12 11.009601	CadmusCo_ea:ab:26	Broadcast	ARP	60	192.168.0.124 is at 08:00:27:ea:ab:26
13 12.010164	CadmusCo_ea:ab:26	Broadcast	ARP	60	192.168.0.124 is at 08:00:27:ea:ab:26

分析：

可以看出 3 号机的 MAC 被修改成了 1 号机的 MAC 了。此时 2 号机向 3 号机发起 Telnet，发现 telnet 工作不正常，超时！

实验任务第二项：ICMP 重定向攻击

背景知识：

ICMP 重定向信息是路由器向主机提供实时的路由信息，当一个主机收到 ICMP 重定向信息时，它会根据这个信息来更新自己的路由表。由于缺乏必要的合法性检查，如果一个黑客想要被攻击的主机修改它的路由表，黑客就会发送 ICMP 重定向信息给被

攻击的主机，让该主机按照黑客的要求来修改路由表。

PS: 由于在上一个任务中我们三台机器使用的那三个 ip 和网关无法连接外网，在考虑本次实验任务的需求之后将 IP 做出如下修改：

1 号机：

IP: 192.168.1.113

2 号机：

IP: 192.168.1.114

3 号机：

IP: 192.168.1.115

在一般情形中，如果从 3 号机远程访问 **202.112.154.16**（一个校园网址），则 3 号机就会先把请求数据包发送至默认网关，默认网关会转发数据包到 **202.112.154.16**

Filter:				Expression...	Clear	Apply
No.	Time	Source	Destination	Protocol	Length	Info
109	22.800945	192.168.1.115	202.112.154.16	ICMP	74	Echo (ping) request id=0x0b40, seq=27/
110	22.801012	192.168.1.115	202.112.154.16	ICMP	74	Echo (ping) request id=0x0b40, seq=28/
111	22.801254	192.168.1.115	202.112.154.16	ICMP	74	Echo (ping) request id=0x0b40, seq=29/
112	22.801329	192.168.1.115	202.112.154.16	ICMP	74	Echo (ping) request id=0x0b40, seq=30/
113	22.801397	192.168.1.115	202.112.154.16	ICMP	74	Echo (ping) request id=0x0b40, seq=31/
114	22.801465	192.168.1.115	202.112.154.16	ICMP	74	Echo (ping) request id=0x0b40, seq=32/
115	22.801801	192.168.1.115	202.112.154.16	ICMP	74	Echo (ping) request id=0x0b40, seq=33/
116	22.804449	202.112.154.16	192.168.1.115	ICMP	74	Echo (ping) reply id=0x0b40, seq=18/
117	22.804470	202.112.154.16	192.168.1.115	ICMP	74	Echo (ping) reply id=0x0b40, seq=19/
118	22.804474	202.112.154.16	192.168.1.115	ICMP	74	Echo (ping) reply id=0x0b40, seq=20/
119	22.804476	202.112.154.16	192.168.1.115	ICMP	74	Echo (ping) reply id=0x0b40, seq=21/
120	22.804479	202.112.154.16	192.168.1.115	ICMP	74	Echo (ping) reply id=0x0b40, seq=22/
121	22.804481	202.112.154.16	192.168.1.115	ICMP	74	Echo (ping) reply id=0x0b40, seq=23/
122	22.804484	202.112.154.16	192.168.1.115	ICMP	74	Echo (ping) reply id=0x0b40, seq=24/
123	22.804487	202.112.154.16	192.168.1.115	ICMP	74	Echo (ping) reply id=0x0b40, seq=25/

▶ Frame 111: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)

▶ Ethernet II, Src: CadmusCo_c6:57:e5 (08:00:27:c6:57:e5), Dst: Tp-LinkT_62:1c:be (b0:48:7a:62:1c:be)

▶ Internet Protocol Version 4, Src: 192.168.1.115 (192.168.1.115), Dst: 202.112.154.16 (202.112.154.16)

▶ Internet Control Message Protocol

攻击：

现在 1 号机以默认网关的名义向 3 号机发送 IMCP 重定位信息，通知 3 号机，默认路由的地址已经改为 1 号机的 IP。同时为了让 1 号机能够转发数据包，需要对 1 号机进行转发数据包的设置，可用下面的命令实现：

```
sudo sysctl net.ipv4.ip_forward=1
```

此时 3 号机再次访问 202.112.154.16 时，它会首先把数据包发送至新的路由 1 号

机，再由 1 号机来转发数据包，从而达到 Machine 3 的路由表被更新的效果：

Filter:				Expression...	Clear	Apply
No.	Time	Source	Destination	Protocol	Length	Info
144	17.060051	192.168.1.115	202.112.154.16	ICMP	74	Echo (ping) request id=0x0bcc, s
145	17.060065	192.168.1.115	202.112.154.16	ICMP	74	Echo (ping) request id=0x0bcc, s
146	17.060134	192.168.1.115	202.112.154.16	ICMP	74	Echo (ping) request id=0x0bcc, s
147	17.060149	192.168.1.115	202.112.154.16	ICMP	74	Echo (ping) request id=0x0bcc, s
148	17.060443	192.168.1.115	202.112.154.16	ICMP	74	Echo (ping) request id=0x0bcc, s
149	17.060698	192.168.1.115	202.112.154.16	ICMP	74	Echo (ping) request id=0x0bcc, s
150	17.062177	202.112.154.16	192.168.1.115	ICMP	74	Echo (ping) reply id=0x0bcc, s
151	17.062350	202.112.154.16	192.168.1.115	ICMP	74	Echo (ping) reply id=0x0bcc, s
152	17.062770	192.168.1.115	202.112.144.246	DNS	87	Standard query PTR 16.154.112.202
153	17.063236	202.112.154.16	192.168.1.115	ICMP	74	Echo (ping) reply id=0x0bcc, s
154	17.063250	202.112.154.16	192.168.1.115	ICMP	74	Echo (ping) reply id=0x0bcc, s
155	17.064474	202.112.154.16	192.168.1.115	ICMP	74	Echo (ping) reply id=0x0bcc, s
156	17.064545	202.112.154.16	192.168.1.115	ICMP	74	Echo (ping) reply id=0x0bcc, s
157	17.064549	202.112.154.16	192.168.1.115	ICMP	74	Echo (ping) reply id=0x0bcc, s
158	17.064552	202.112.154.16	192.168.1.115	ICMP	74	Echo (ping) reply id=0x0bcc, s
159	17.064555	202.112.154.16	192.168.1.115	ICMP	74	Echo (ping) reply id=0x0bcc, s
▶ Frame 151: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)						
▶ Ethernet II, Src: CadmusCo_ea:ab:26 (08:00:27:ea:ab:26), Dst: CadmusCo_c6:57:e5 (08:00:27:c6:57:e5)						
▶ Internet Protocol Version 4, Src: 202.112.154.16 (202.112.154.16), Dst: 192.168.1.115 (192.168.1.115)						
▶ Internet Control Message Protocol						

实验任务第三项：SYN 泛洪攻击

背景知识：

SYN 攻击是一种 DoS (Denial of Service) 攻击，在这种攻击中黑客向被攻击者的 TCP 端口发送很多 SYN 请求，但是黑客并不是想完成三次握手协议，而是使用伪造的 IP 地址或者只进行三次握手协议中的第一次握手。因为 SYN 数据包用来打开一个 TCP 链接，所以受害者的机器会向伪造的地址发送一个 SYN/ACK 数据包作为回应，并等待预期的 ACK 响应。每个处于等待状态，半开的链接队列都讲进入空间有限的待处理队列。由于伪造的源地址实际上并不存在，所以将那些等待队列中的记录删除并完成建立 TCP 连接所需的 ACK 响应用于不会到来，相反每个半开的连接一定会超时，这将花费一段比较长的时间。

只要攻击者使用伪造的 SYN 数据包继续泛洪受害者的系统，受害者的待处理队列将一直处于满员，这使得真正的 SYN 数据包几乎不可能到达系统并打开有效的 TCP 连接。

攻击:

将 3 号机 (IP: 192.168.1.127 端口: 23) 作为 Telnet 服务器, 2 号机 (IP: 192.168.1.1 作为 Telnet 客户端, 去连接 Telnet 服务器。

```
Welcome to Ubuntu 12.04.2 LTS (GNU/Linux 3.2.0-43-generic-pae i686)

* Documentation:  https://help.ubuntu.com/

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

正常情况如上图所示

下图为 3 号机的 23 号端口进行洪泛攻击:

```
inet 地址:192.168.1.113 广播:192.168.1.255 掩码:255.255.255.0
inet6 地址: fe80::a00:27ff:feea:ab26/64 Scope:Link
inet 地址:127.0.0.1 掩码:255.0.0.0
inet6 地址: ::1/128 Scope:Host

Title: Synflood
Usage: netwox 76 -i ip -p port [-s spoofip]
Parameters:
-i|--dst-ip ip          destination IP address {5.6.7.8}
-p|--dst-port port      destination port number {80}
-s|--spoofip spoofip    IP spoof initialization type {linkbraw}
--help2                display full help
Example: netwox 76 -i "5.6.7.8" -p "80"
Example: netwox 76 --dst-ip "5.6.7.8" --dst-port "80"
```

利用# netstat -na |grep “:23” 查看 3 号机的端口 23 的待处理队列后, 继续
下图为 2 号机的 Telnet 客户端无法再次连接 3 号机:

```
Trying 192.168.1.115...
```

分析:

即使被洪泛攻击, 3 号机也可以提供 Telnet 服务。

在 sysctl -w net.ipv4.tcp_syncookies=0(即关闭 SYN cookies)的情况下, 3 号机没有抵抗洪泛攻击的保护机制, 2 号机的 Telnet 客户端连接不上 3 号机, 所以会一直处于 Trying……的状态。如果打开 SYN cookie (用命令: sysctl -w net.ipv4.tcp_syncookies=1), 则即使在被洪泛的情环境下, 3 号机也可以有效的提供 Telnet 服务:

实验任务第四项：S 对 Telnet 和 SSH 的 TCP RST 攻击

背景知识：

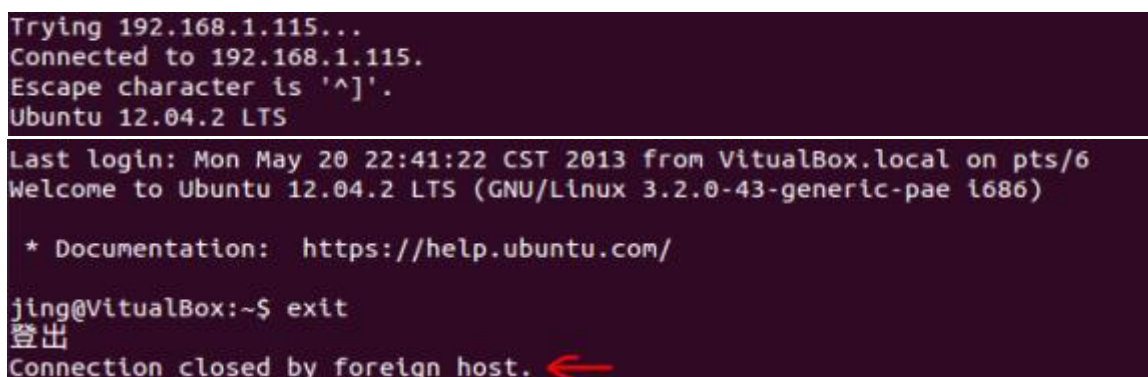
TCP RST 攻击：这种攻击只能针对 tcp、对 udp 无效。 RST 用于复位因某种原因引起出现的错误连接，也用来拒绝非法数据和请求。如果接收到 RST 位时候，通常发生了某些错误。TCP RST 攻击可以终止两个被攻击主机之间的 TCP 连接。

攻击：

当 2 号机器的 Telnet 客户端和 3 号机器的 Telnet 服务器之间建立 Telnet 连接之后，通过向 Telnet 客户端发送 TCP RST，就可以实现终止两者之间的 TCP 连接。

2 号机和 3 号机先建立 Telnet TCP 连接，同时 1 号机向 2 号机发送相应的 TCP TST 包

下图为 2 号机的 Telnet 断开连接的示意图：



```
Trying 192.168.1.115...
Connected to 192.168.1.115.
Escape character is '^]'.
Ubuntu 12.04.2 LTS

Last login: Mon May 20 22:41:22 CST 2013 from VirtualBox.local on pts/6
Welcome to Ubuntu 12.04.2 LTS (GNU/Linux 3.2.0-43-generic-pae i686)

 * Documentation:  https://help.ubuntu.com/

jing@VirtualBox:~$ exit
登出
Connection closed by foreign host. ←
```

分析：

一开始 2 号机和 3 号机器之间通过 Telnet TCP 连接，在经过由 1 号机发给 2 号机的 TCP TST 数据包后，从上图箭头处可以看出 2 号机的 telnet 的客户端被断开。

(1) 实验任务第五项：对视频流应用程序的 TCP RST 攻击

该次实验任务的原理与其实与上一个实验任务是相似的，但由于我们小组的虚拟机里没有安装视频流的应用，这里没有贴出相应的操作流程，但其实是同理可得的！

(2) 实验任务第六项：ICMP Blind Connection-Reset and Source-Quench 攻击

背景知识：

为了达到利用 ICMP 数据报进行 TCP/RESET 攻击的目的，黑客们常发送一个 ICMP 报文给通信双方，暗示他们双方的 TCP 通信端出现硬件错误，连接必须终止。

攻击：

下图为 2 号机至 3 号机上的 Telnet 服务器的示意图：

```
Trying 192.168.1.115...
Connected to 192.168.1.115.
Escape character is '^]'.
Ubuntu 12.04.2 LTS

Last login: Mon May 20 22:51:50 CST 2013 from VirtualBox-2.local on pts/2
Welcome to Ubuntu 12.04.2 LTS (GNU/Linux 3.2.0-43-generic-pae i686)

 * Documentation:  https://help.ubuntu.com/

2 packages can be updated.
0 updates are security updates.
```

下图为由 1 号机发送 3 号机收到的伪造数据包：

tcp	0	0	127.0.0.1:53	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:21	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:23	0.0.0.0:*	LISTEN
tcp	0	0	127.0.0.1:631	0.0.0.0:*	LISTEN
tcp	1	0	192.168.1.115:54875	91.189.89.144:80	CLOSE_WAIT
tcp	0	0	192.168.1.115:42510	192.168.1.115:23	TIME_WAIT
tcp	0	0	192.168.1.115:23	192.168.1.114:46202	ESTABLISHED
tcp6	0	0	:::1:631	:::*	LISTEN

【分析】
从第 31 行可知，1 号机向 3 号机发出的伪造数据包，但和 3 号机上的 Telnet 连接并没有因此而中断，可能是这种安全漏洞在实验所用的 Ubuntu 系统上已经被修复。

实验任务第七项：TCP 报文劫持

背景知识

会话劫持利用了 TCP/IP 工作原理来设计攻击。TCP 使用端到端的连接，即 TCP 用（源 IP，源 TCP 端口号，目的 IP，目的 TCP 端号）来唯一标识每一条已经建立连接的 TCP 链路。另外，TCP 在进行数据传输时，TCP 报文首部的两个字段序号（seq）和确认序号（ackseq）非常重要。序号（seq）和确认序号（ackseq）是与所携带 TCP 数据净荷（payload）的多少有数值上的关系：序号字段（seq）指出了本报文中传送的数据在发送主机所要传送的整个数据流中的顺序号，而确认序号字段（ackseq）指出了发送本报文的主机希望接收的对方主机中下一个八位组的顺序号。因此，对于一台主机来说，其收发的两个相临 TCP 报文之间的序号和确认序号的关系为：它所要发出的报文中的 seq 值应等于它所刚收到的报文中的 ackseq 的值，而它所要发送报文中 ackseq 的值应为它所收到报文中 seq 的值加上该报文中所发送的 TCP 净荷的长度。

攻击

2 号机 Telnet 到 3 号机，实验在 1 号机上劫持 2 号机到 3 号机上的 Telnet 报文。

下图为在 1 号机上的 wireshark 中显示了 2 号机与 3 号机的连接：

Filter:				Expression...	Clear	
No.	Time	Source	Destination	Protocol	Length	Info
9	0.167338	192.168.1.115	192.168.1.114	TELNET	78	Telnet Data ...
10	0.167458	192.168.1.114	192.168.1.115	TCP	66	38429 > telnet [
11	0.167856	192.168.1.115	192.168.1.114	TELNET	105	Telnet Data ...
12	0.168148	192.168.1.114	192.168.1.115	TCP	66	38429 > telnet [
13	0.168312	192.168.1.114	192.168.1.115	TELNET	146	Telnet Data ...
14	0.168516	192.168.1.115	192.168.1.114	TCP	66	telnet > 38429 [
15	0.168845	192.168.1.115	192.168.1.114	TELNET	69	Telnet Data ...
16	0.168849	192.168.1.114	192.168.1.115	TELNET	69	Telnet Data ...
17	0.208677	192.168.1.115	192.168.1.114	TCP	66	telnet > 38429 [
18	0.214176	192.168.1.115	192.168.1.114	TELNET	69	Telnet Data ...
19	0.214520	192.168.1.114	192.168.1.115	TELNET	69	Telnet Data ...
20	0.214710	192.168.1.115	192.168.1.114	TELNET	86	Telnet Data ...
21	0.251120	192.168.1.114	192.168.1.115	TCP	66	38429 > telnet [
▶ Frame 10: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)						
▶ Ethernet II, Src: CadmusCo_5f:ae:ef (08:00:27:5f:ae:ef), Dst: CadmusCo_c6:57:e5 (08:00:27:c6:57:e5)						
▶ Internet Protocol Version 4, Src: 192.168.1.114 (192.168.1.114), Dst: 192.168.1.115 (192.168.1.115)						
▶ Transmission Control Protocol, Src Port: 38429 (38429), Dst Port: telnet (23), Seq: 28, Ack: 13, Len: 0						
0000	08 00 27 c6 57 e5 08 00 27 5f ae ef 08 00 45 10	..'.W... '_....E.				
0010	00 34 fb 00 40 00 40 06 bb 7d c0 a8 01 72 c0 a8	.4..0.0. .}...r..				
0020	01 73 96 1d 00 17 2d f9 be 65 b7 fb 28 6c 80 10	.s....-. .e..(l..				
0030	07 21 68 6c 00 00 01 01 08 0a 00 11 44 de 00 10	.!hl....D...				

五、实验总结

通过这一次内容丰富并且工作量巨大的实验，我们小组对基于 TCP/IP 的攻击有了更加深刻甚至可以说是比较新的认识，对它们各自的机制、攻击特点、相互之间可能存在的联系以及它们差别所在等等细节问题有了新的看法、认识，也有了一些专属于我们小组自己的解决方案。

这次实验，让我们至少意识到了以下这样一个事实：

TCP/IP 协议在设计之初仅考虑了成本和实现功能，并没有过多考虑安全因素。因此 TCP/IP 协议栈中提供了大量的起关键作用的信息和指令，但是这些信息和指令的执行缺乏认证机制，能够方便地伪造。这也就为如此之多的 TCP/IP 攻击提供了可能。