



网络安全 实验报告

学 院： 信息科学与工程学院

专业班级： 信息安全 1401 班

学 号： 0906140125

姓 名： 何文慧

目录

包嗅探和包欺骗.....	1
一、实验背景.....	1
二、实验目的.....	1
三、实验环境.....	1
四、实验内容.....	1
任务 1 写数据包监听程序.....	1
任务 2 包欺骗.....	4
任务 3 嗅探和欺骗.....	8
五、实验心得.....	8

包嗅探和包欺骗

一、实验背景

在网络安全中，数据包嗅探和欺骗是两个重要的概念；他们是在网络通信中的两个主要威胁。能够理解这两个威胁对于理解网络中的安全措施至关重要。有很多包嗅探和欺骗的工具，例如 Wireshark、Tcpdump、Netwox 等。一些这些工具被安全专家以及攻击者广泛使用。能够使用这些工具对于学生来说，是重要的网络安全课程。更重要的是要了解这些工具的工作方式，即，在软件中如何实施数据包嗅探和欺骗。

二、实验目的

让学生掌握包嗅探和欺骗的工具。学生们会一些简单的嗅探和欺骗程序，读取其源代码，对其进行修改和最终深入的了解，对这些程序的技术方面有所了解。在这个实验的最后，学生应该能够编写自己的嗅探和欺骗程序。

三、实验环境

VMware 12.1.1

Ubuntu 12.04

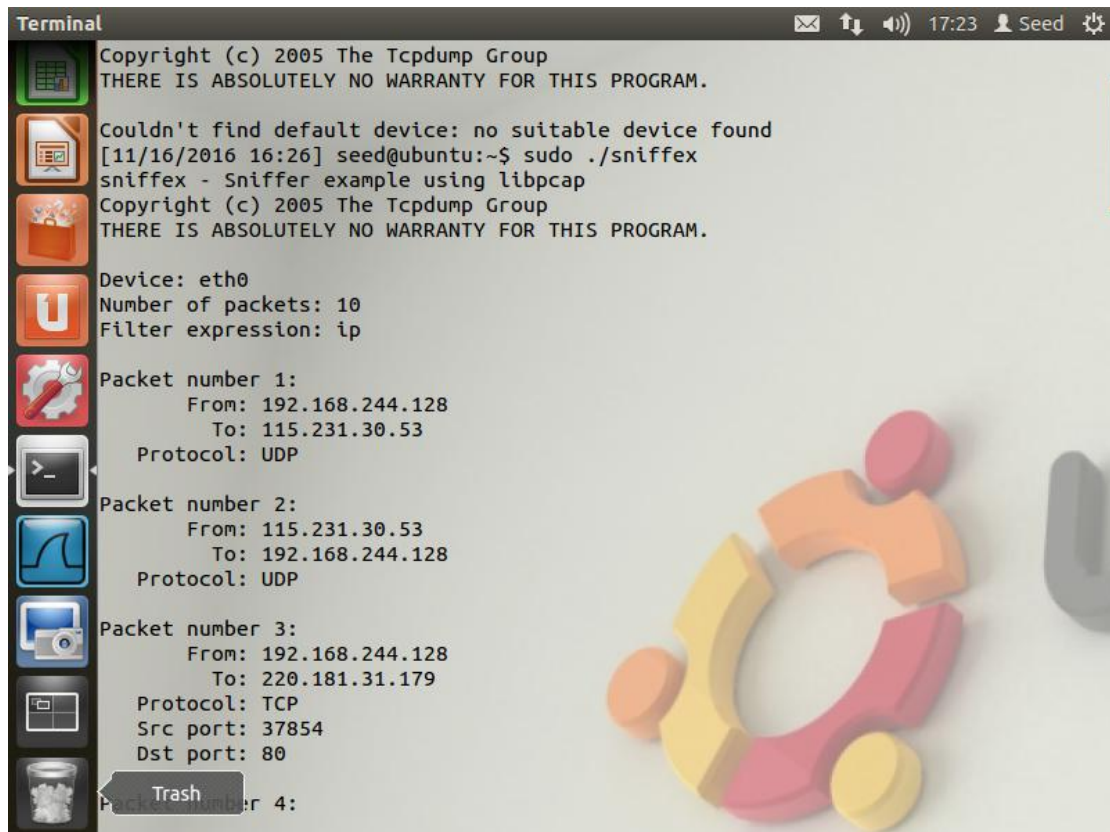
四、实验内容

任务 1 写数据包监听程序

嗅探程序使用 pcap 库可以很容易地编写。利用 pcap，嗅探器的任务就成了在 pcap 库中调用一系列简单的程序。在序列结束时，数据包将被放置在缓冲区中，以进一步处理，这时它们可以被捕获。所有的捕获数据包的细节由 pcap 库处理。Tim Carstens 写了一个教程关于如何使用 pcap 库写的嗅探程序。本教程是在 <http://www.tcpdump.org/pcap.htm>。

任务 1.a: 了解 sniffex。

请从上述教程下载 sniffex.c 程序，编译并运行它。并提供截图证据表明你的程序成功运行并产生预期的结果。



```
Terminal
Copyright (c) 2005 The Tcpdump Group
THERE IS ABSOLUTELY NO WARRANTY FOR THIS PROGRAM.

Couldn't find default device: no suitable device found
[11/16/2016 16:26] seed@ubuntu:~$ sudo ./sniffex
sniffex - Sniffer example using libpcap
Copyright (c) 2005 The Tcpdump Group
THERE IS ABSOLUTELY NO WARRANTY FOR THIS PROGRAM.

Device: eth0
Number of packets: 10
Filter expression: ip

Packet number 1:
  From: 192.168.244.128
  To: 115.231.30.53
  Protocol: UDP

Packet number 2:
  From: 115.231.30.53
  To: 192.168.244.128
  Protocol: UDP

Packet number 3:
  From: 192.168.244.128
  To: 220.181.31.179
  Protocol: TCP
  Src port: 37854
  Dst port: 80

Packet number 4:
```

截图 1 运行 sniffex.c

问题 1: 请用自己的话俩描述对于嗅探程序中必要的调用库的序列。这是一个总结，而不是像教程中的一个详细的解释。

答：库调用顺序以及每个函数作用如下：

`pcap_lookupdev(errbuf)`获取当前系统网络设备

`pcap_lookupnet(dev, &net, &mask, errbuf)`获得指定网络设备的网络号和掩码

`pcap_open_live(dev, SNAP_LEN, 1, 1000, errbuf)`打开网络设备

`pcap_datalink(handle)`检查 MAC 层，以确保处理的是以太网

`pcap_compile(handle, &fp, filter_exp, 0, net)`将用户制定的过滤策略编译到过滤程序中

`pcap_setfilter(handle, &fp)`设置过滤器

`pcap_loop(handle, num_packets, got_packet, NULL)`捕获数据包

`pcap_freecode(&fp)`和 `pcap_close(handle)`关闭会话

问题 2: 为什么需要 root 权限来运行 sniffex？如果没有 root 权限就运行程序到哪一步会失败？

答：由于是底层的系统调用，所以需要 root 权限。否则系统会检测不到网卡。在进行“打开网络接口”这一步时，需要告诉程序我们的网卡接口，或者让程序自己检测。

问题 3: 请开启和关闭嗅探程序的混杂模式。你能证明这种模式在开启和关闭时的区别吗？请描述你如何证明这一区别。

答：一般来说，非混杂模式的嗅探器中，主机仅嗅探那些跟它直接有关的通信，如发向它的，从它发出的，或经它路由的等都会被嗅探器捕捉。而在混杂模式中则嗅探传输线路上的所有通信。在非交换式网络中，这将是整个网络的通信。这样做最明显的优点就是使更多的包被嗅探到。但是，混杂模式也是可被探测到的。

任务 1.b: 写过滤器。

请给你的嗅探程序写过滤表达式来捕捉如下格式的数据包。在你的实验报告中，你需要用截屏来展示这些过滤表达式运行的结果。

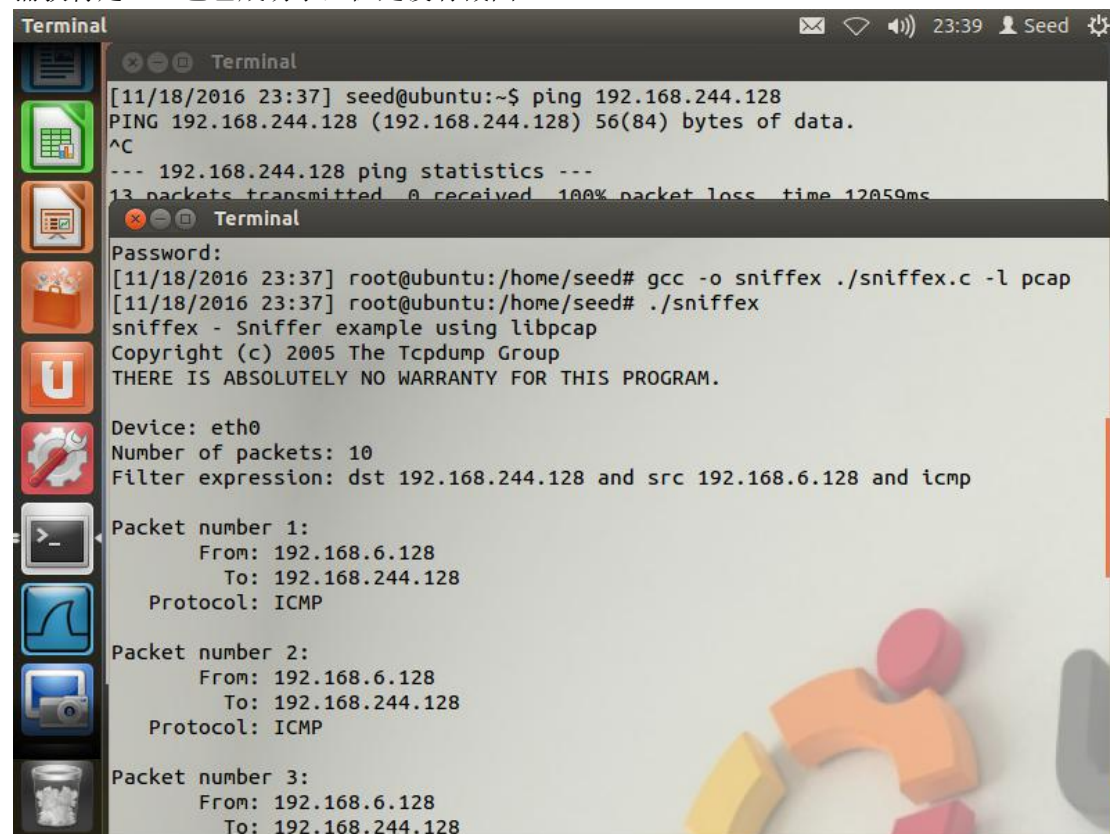
- 捕获两个特定主机之间的 ICMP 数据包。
- 捕获有一个目的端口范围是端口 10 - 100 的 TCP 数据包。

分别修改过滤规则为：

```
char filter_exp[] = "dst 192.168.225.128 and src 192.168.244.128 and icmp";
```

```
char filter_exp[] = "tcp and portrange 10-100";
```

重新编译程序后运行即可得到已经经过过滤的数据包。以下为捕获特定 ICMP 包的截图，捕获特定 TCP 包也成功了，但是没有截图。



截图 2 设置 ICMP 包过滤后发送 ICMP 包的嗅探结果

任务 1.c: 嗅探密码。

请展示，当有人在你监控的网络上使用 Telnet 远程登录时，如何使用 sniffex 来获得密码。您可能需要修改 sniffex.c。你还需要在你的虚拟机启动 telnetd 服务器。如果您使用我们的预建的 VM，telnetd 服务器已经安装；只键入以下命令启动它。

```
$sudo service openbsd-inetd start
```

这个任务我尝试了，但是没成功。远程登录我不是很懂，只试了一下登录本地的 127.0.0.1，因为不知道登录本地是否有包发送，但是开了 wireshark 并没有捕捉相应包。而本机登录虚拟机时提示未开放 23 号端口，由于时间有点紧急，没有来得及调试这个 bug 就开始了下一步，所以最后就很遗憾没有实现该任务。

任务 2 包欺骗

当一个普通用户发出的数据包，操作系统通常不允许用户自己组建协议头（如 TCP，UDP，和 IP 报头）。操作系统将设置大部分的领域，而只允许用户设置其他领域，如目的 IP 地址，目的端口号等。但是，如果用户有 root 权限，他们可以将数据包头设置为任意字段。这就是所谓的包欺骗，它可以通过原始套接字完成。原始套接字给程序员对数据包结构的绝对控制，允许程序员构建任何任意的数据包，包括设置头字段和有效载荷。使用原始套接字是相当简单的，它涉及到四个步骤：（1）创建一个原始套接字，（2）设置套接字选项，（3）构造数据包，（4）通过原始套接字发送数据包。有许多在线教程，可以教你如何使用原始套接字编写 C 编程。我们已经把一些教程与实验室的网页联系起来了。请阅读它们，并学习如何写一个包欺骗程序。我们展示了一个这种程序的简单骨架。

```
int sd;
struct sockaddr_in sin;
char buffer[1024]; // You can change the buffer size

/* Create a raw socket with IP protocol. The IPPROTO_RAW parameter
 * tells the sytem that the IP header is already included;
 * this prevents the OS from adding another IP header.
 */
sd = socket(AF_INET, SOCK_RAW, IPPROTO_RAW);
if(sd < 0) {
    perror("socket() error"); exit(-1);
}

/* This data structure is needed when sending the packets
 * using sockets. Normally, we need to fill out several
 * fields, but for raw sockets, we only need to fill out
```

```

* this one field
*/
sin.sin_family = AF_INET;

// Here you can construct the IP packet using buffer[]
// - construct the IP header ...
// - construct the TCP/UDP/ICMP header ...
// - fill in the data part if needed ...
// Note: you should pay attention to the network/host byte order.

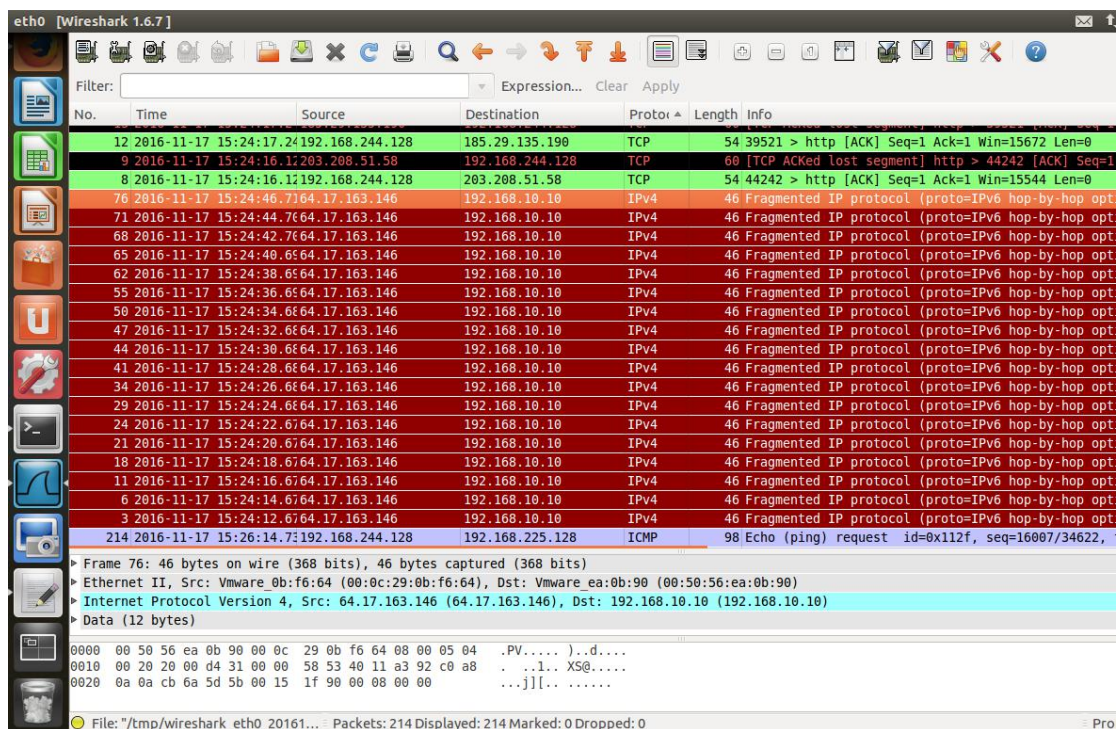
/* Send out the IP packet.
* ip_len is the actual size of the packet.
*/
if(sendto(sd, buffer, ip_len, 0, (struct sockaddr *)&sin, sizeof(sin)) < 0) {
    perror("sendto() error"); exit(-1);
}

```

任务 2.a: 写一个包欺骗程序。

你可以写你自己的包程序或下载一个。你需要提供的证据（例如，Wireshark 数据包跟踪）向我们展示你的程序成功地发送伪造的 IP 数据包。

这个任务，我从 <http://www.tenouk.com/Module43a.html> Advanced TCP/IP - THE RAW SOCKET PROGRAM EXAMPLES 网页教程中 copy 了一段构造 UDP 数据包的程序，并在 Ubuntu 中运行，一边开启了 Wireshark，成功捕捉到了返回的数据包。

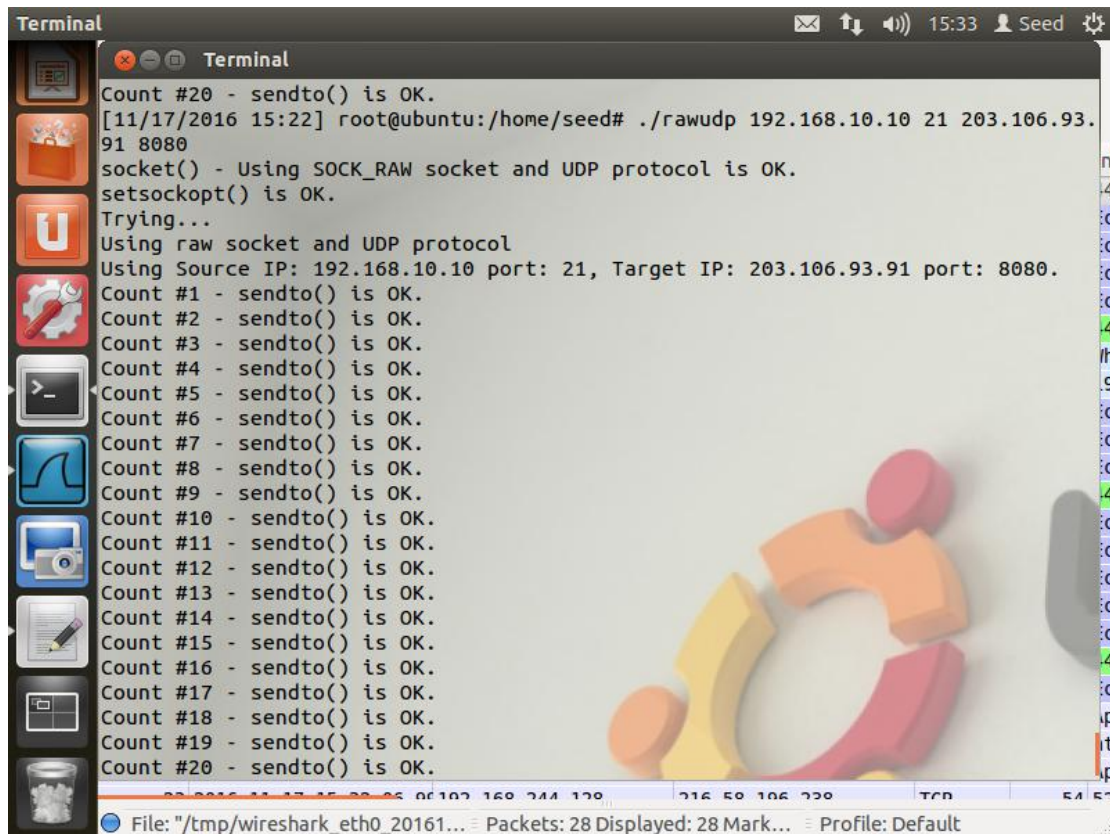


截图 3 wireshark 跟踪到的返回的数据包

任务 2.b: 欺骗 ICMP 回送请求。

欺骗 ICMP 回送请求数据包到另一台机器的地址（即，使用另一个机器的 IP 地址作为源 IP 地址）。这包应该发送到 Internet 上的远程机（机器必须在线）。你应该把你的 Wireshark 的，所以如果你的欺骗是成功的，你可以看到来自远程机器传回来的回应。

这个任务，我还是从 <http://www.tenouk.com/Module43a.html> Advanced TCP/IP - THE RAW SOCKET PROGRAM EXAMPLES 网页教程中 copy 了一段构造 ICMP 数据包的程序，并在 Ubuntu 中运行，一边开启了 Wireshark，但是这次运行了很久却没有捕捉到了返回的数据包，可能是 ping 的 IP 地址不存在或者屏蔽 ping 的原因。



截图 4 ICMP 包欺骗

任务 2.c: 欺骗以太网帧。

欺骗一个以太网帧。设置 01:02:03:04:05:06 作为源地址。为了告诉系统该包已经有以太网报头，你需要使用以下参数创建原始套接字：

```
sd = socket(AF_PACKET, SOCK_RAW, htons(ETH_P_IP));
```

在构建数据包时，`buffer[]`的开始现在应该是以太网头。

由于对编写此类程序比较不熟悉，我先是参考了一段网上的代码，但是在编译时出错了，网上查询发现是指针和结构体的错误，但是我发现没法修复这个 bug，且对 Linux 下 C 程序的编写也不是很熟悉，所以不知道怎么改，这个任务并没有成功完成。

问题。请回答以下问题。

问题 4： 你可以设置 IP 数据包长度字段为任意值，而不管实际的包有多大吗？

答：不能。可以设置 ip 包长度在一定范围内，但是不能小于 ip 包最小长度也不能大于最大长度。但设置的值不能与实际包长度不等。

问题 5： 使用原始套接字编程，你必须要计算 IP 报头的校验吗？

答：需要。传输数据的时候不可能保证在传输的过程中不会出现差错（传输介质并不是

理想的), 而校验和可以看看发过来的包是否正确。原始套接字是包含了 IP 头和 TCP 头的数据, 所以需要校验和。因为本身 TCP/ip 的协议就是需要有校验和的。不过平时用的 socket 是系统处理了这一切, 看起来是透明的。

问题 6: 为什么你需要 root 权限来运行使用原始套接字的程序? 如果没有 root 权限执行, 程序哪里会失败?

答: 原始套接字可以读写内核没有处理的 IP 数据包。原始套接字只能由有 root 权限的人创建。没有 root 权限时, 程序创建原始套接字时即失败。

任务 3 嗅探和欺骗

在这个任务中, 你将包嗅探和包欺骗技术实现以下嗅探和欺骗程序。你需要在同一局域网有两台虚拟机。你可以从 A 虚拟机 ping B 虚拟机的 IP 地址 X。这将产生一个 ICMP 回送请求报文。如果 X 是存在的, ping 程序将收到一个回复, 并显示出响应。你的嗅探和欺骗程序运行在通过数据包监听局域网的 B 虚拟机上, 每当它看到 ICMP 回送请求, 无论目标 IP 地址是什么, 你的程序应该立即发送一个应用包欺骗技术的应答包。因此, 无论机器 X 是否是存在的, ping 程序将总是收到一个答复, 表明 X 是存在。你需要写这样的程序, 并且在你的报告中包含截屏, 表明你的应答包起作用了, 请在报告中附上代码。

该任务由于我的电脑太渣, 开不起两台虚拟机的原因, 只能无奈放弃了。

五、实验心得

和上一个 Heartbleed 实验相比, 这个实验遇到了很多问题, 甚至有两个小任务和一个大任务没有完成。可能是由于积累不够的原因, 对 C 语言的 pcap 编程和原始套接字编程以前完全没有涉及, 虽然可以理解包嗅探和包欺骗的原理, 但是在实际运用中发现有很多问题没有解决, 只能很多都使用的现有的代码。这之间也查询了很多资料, 也算是解决了一小部分的问题。希望在以后的学习生活中可以解决这些问题。

通过本次实验我了解到包嗅探和包欺骗的原理, 并尝试进行了相应的程序, 感觉自己收获了很多。