

网络安全实验报告

题 目 捕包软件的使用与实现

专 业

学 号

学 生

指 导 教 师 王彦

**一、实验目的**

理解捕包程序捕包过程，可以自己编程捕包并从数据包中解析出需要的信息。

# 二、实验内容

1. 熟练使用 sniffer 或 wireshark 软件，对协议进行还原（能够找访问网页的四元组）；只需要写报告，不需要在实验课检查。
2. 利用 libpcap 或 winpcap 进行编程，能够对本机的数据包进行捕获分析（比如将本机所有数据包的四元组写到指定文件），按照自己的设想撰写需求分析和详细设计。（实验课检查程序）

# 三、实验过程

**（一）使用 wireshark 软件对协议进行还原**

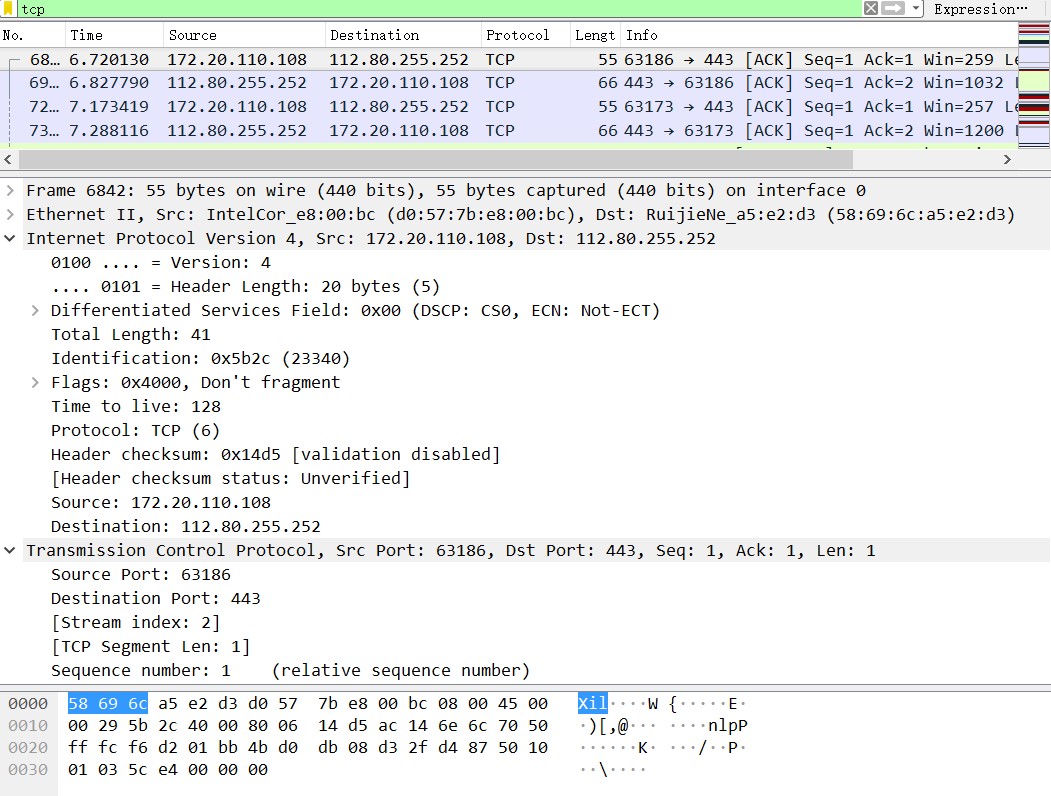
## 实验基本信息：

实验环境：Windows10 x64

WireShark2.6.4

## 1. 捕包并分析四元组

(1) TCP 分析



截图中这个 TCP 数据包，源 IP 为 172.20.110.108，目的 IP 为 112.80.255.252，源

端口为 63186，目的端口为 443。

分析：以太网头部：

前 6 个字节 58 69 6c a5 e2 d3 为目的主机 MAC，往后 6 个字节 d0 57 7b e8 00 bc 为源主机 MAC，

往后 2 个字节为上层协议，0x0800 表示 IPv4 协议；以太网头部结束，现在是 ip 头部：往后 1 个字节 0x45 表示 IP 版本为 4，头部长度为 5，往后 1 个字节为区分服务，0x00 表示默认，

往后 2 个字节为总长度，0x0029 = 41，往后 2 个字节为 id，值为 0x5b2c（23340），

往后 2 个字节为标志位+片偏移，值为 0x4000，

往后 1 个字节为 ttl，值为 0x80 = 128，

往后 1 个字节为上层协议，0x06 表示 TCP 协议，往后 2 个字节为头部校验和，值为 0x14d5，

往后 4 个字节为源 ip 地址 0xac146e6c，转换为 10 进制就是 172.20.110.108，往后4个字节为目的ip地址0x7050fffc，转换为10进制就是112.80.255.252；

ip 头部结束：现在是 TCP 头部：

往后 2 个字节为源端口 0xf6d2 = 63186，往后 2 个字节为目的端口 0x01bb = 443，往后 4 个字节为 seq 0x4bd0db08，往后 4 个字节为 ack 0xd32fd487，

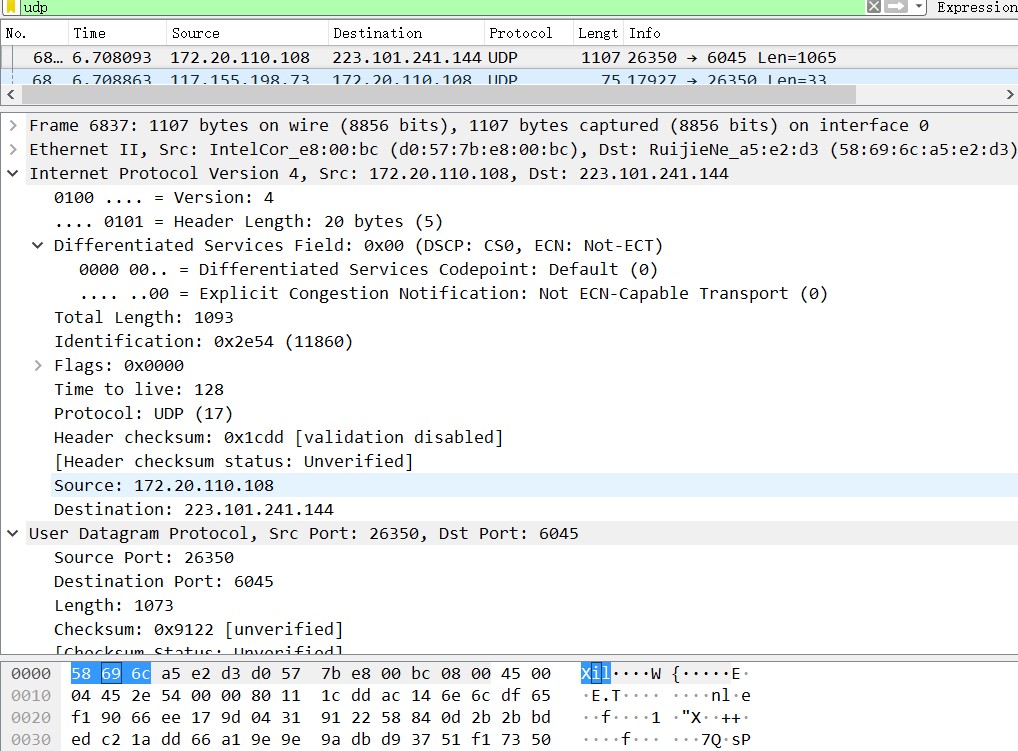
往后 1 个字节为头部长度 0x50 = 80（其实是 4 位），

往后 1 个字节为标志位（其实是 6 位），

往后 2 个字节为窗口大小 0x0103 = 259，往后 2 个字节为校验和 0x5ce4，

往后 2 个字节为紧急指针 0x0000； TCP 头部结束，接下来是数据。

(2) UDP 分析



截图中这个 UDP 数据包，源 IP 为 172.20.110.108，目的 IP 为 223.101.241.144，

源端口为 26350，目的端口为 6045。

分析：

以太网头部和 ip 头部与 TCP 协议基本相同，直接分析 UDP 头部，从第三行第三个字节开始：

2 个字节源端口 0x66ee = 26350，

2 个字节目的端口 0x179d = 6045，

2 个字节总长度 0x0431 = 1073， 2 个字节校验值：0x9122；

后面全都是数据。

**（二）利用 libpcap 编写捕包软件实验基本信息：**

实验环境：Ubuntu16.04 x64

编程语言：C 语言

## 1. 需求分析

本程序需要运用 libpcap 来捕获本机数据包，并获取数据包中的四元组，将其展示给

用户。

程序功能：

(1)捕获本机数据包（可以自定义过滤条件）；

(2)逐层解析数据包，获得 IPv4 数据包的源 ip、目的 ip、源端口、目的端口；

(3)将上述四元组写入文件（每次运行程序都新生成一个文件）。

## 2. 环境配置

(1)从官网下载安装包：m4-1.4.18.tar.gz

bison-3.3.tar.gz flex-2.6.4.tar.gz

libpcap-1.9.0.tar.gz

(2)对以上安装包逐个使用以下命令来解压安装包：

tar -zxvf xxx.tar.gz

(3)进入解压后的文件夹后执行以下命令来安装库文件：

./configure

make

make install

(4)配置好环境后，可正常编译，但运行时提示

error while loading shared libraries: libpcap.so.1: cannot open shared object

file: No such file or directory

百度发现是因为没有自动生成链接文件，于是在/usr/lib/文件夹下输入命令：

ln -s libpcap.so.1.9.0 libpcap.so.1

以上命令可生成 libpcap.so.1 文件，但此时仍不能正常运行，还需要将这个文件的地址加入配置文件中，具体做法为执行命令：

locate libpcap.so.1

发现 libpcap.so.1 的路径为/usr/lib/x86\_64-linux-gnu，执行命令： sudo vi /etc/ld.so.conf

打开配置文件，在末尾添加/usr/lib/x86\_64-linux-gnu，保存后退出，再执行命令： sudo ldconfig

这样就编译器就可以正常链接到库了，关于 libpcap 的环境配置也就完成了。

## 3. 数据结构设计

由于是逐层解析以太网数据帧，所以需要准备至少三种数据结构：以太网数据帧头、

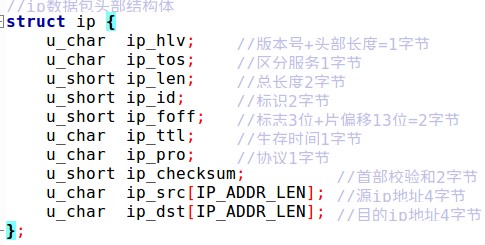
IPv4 数据报头、传输层报文头。具体如下：

数据结构的定义原则：1 字节数据定义为 u\_char，2 字节数据定义为 u\_short，其他 2 的倍数字节的数据（MAC 地址和 IP 地址）定义为 u\_char 数组（TCP 的序列号和 ack 定义为 u\_int，因为它们的表现形式就是一个数字，但地址我们通常是一个一个字节分开解析的）。

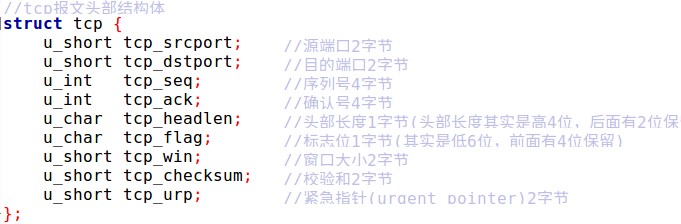
1. 以太网数据帧头



1. IPv4 数据报头



1. TCP 报文头



1. UDP 报文头



## 4. 回调函数设计

## 5. 主函数设计

第一步：使用 pcap\_findalldevs()函数来获取网络设备。此步不用 pcap\_lookupdev() 是因为官方并不推荐使用这个函数，有时第一个位置的网卡是一个虚拟网卡，用它进行下面的步骤会出现错误。

第二步：使用 pcap\_open\_live()函数来获得捕包描述字，由于只需要捕获本机数据包所以设置为非混杂模式。

第三步：如果有过滤条件的话（作为程序运行参数读入），设置过滤条件。

第四步：生成本次捕包的 txt 文件（格式如“capture\_yyyy\_mm\_dd\_hh\_mm\_ss.txt”， capture 后是生成文件的时间），并写入过滤条件和标题栏。

第五步：使用 pcap\_loop()函数和回调函数 ethernet\_callback()来循环捕包。

## 6. 编译运行

使用下列命令编译：

gcc pcap.c -Wall -lpcap -o pcap

其中-Wall 参数指打印所有警告信息，-lpcap 用来链接 pcap 库。

使用下列命令运行：

**sudo** ./pcap

捕包程序在运行时需要 root 权限，否则无法正常打开。

# 四、实验结果

1. **使用 wireshark 软件对协议进行还原**
2. **利用 libpcap 编写捕包软件**

以下为捕包时文件中部分输出：

# 五、心得体会