# 基于Netflow的网络流量工程

## 实验目的

对网络流量进行识别和统计分析是网络管理非常重要的一环，能搞清网络为什么这么慢、到底谁在BT，甚至能发现谁被入侵控制等。学习相关的技术、工具和标准。

## 实验内容

（0）了解概念：sniffer、network traffic analysis、netflow；

（1）用sniffer软件捕获网卡进出的数据包；

（2）使用网络流量监控/识别/分析软件

（3）实验平台：

Win10/Linux

Wireshark/Ntop

## 实验步骤：

## 基础知识

Sniffer是用于监听网络数据分组的技术。Sniffer软件可以捕获指定网络接口上所有分组，如果是集线器环境或安装在路由器/网关上，则可以监控整个网段的流量。常用的Sniffer软件有Sniffer Pro、Wireshark，以及命令行界面的tcpdump/windump等。按照分组的属性(netflow七元组:源IP,目的IP,源端口,目的端口,协议,接口,服务类型)可以把分组归类，这构成了网络流量管理的基础。对netflow记录数据进行分析和挖掘，可以发现对网络管理有价值的信息，如构成流量的不同协议或应用的组成，网络带宽利用率变化规律等，能帮助解决或解释为什么网络这么慢，为什么服务器或个别机器流量异常大等问题，甚至也能发现入侵、botnet等安全问题。这些信息通常可以用图表直观地展现出来。

## 利用Wireshark分析HTTP协议，IP协议，DNS协议，UDP协议，ARP协议以及Ethernet数据帧

1. HTTP分析

### HTTP GET/response 交互

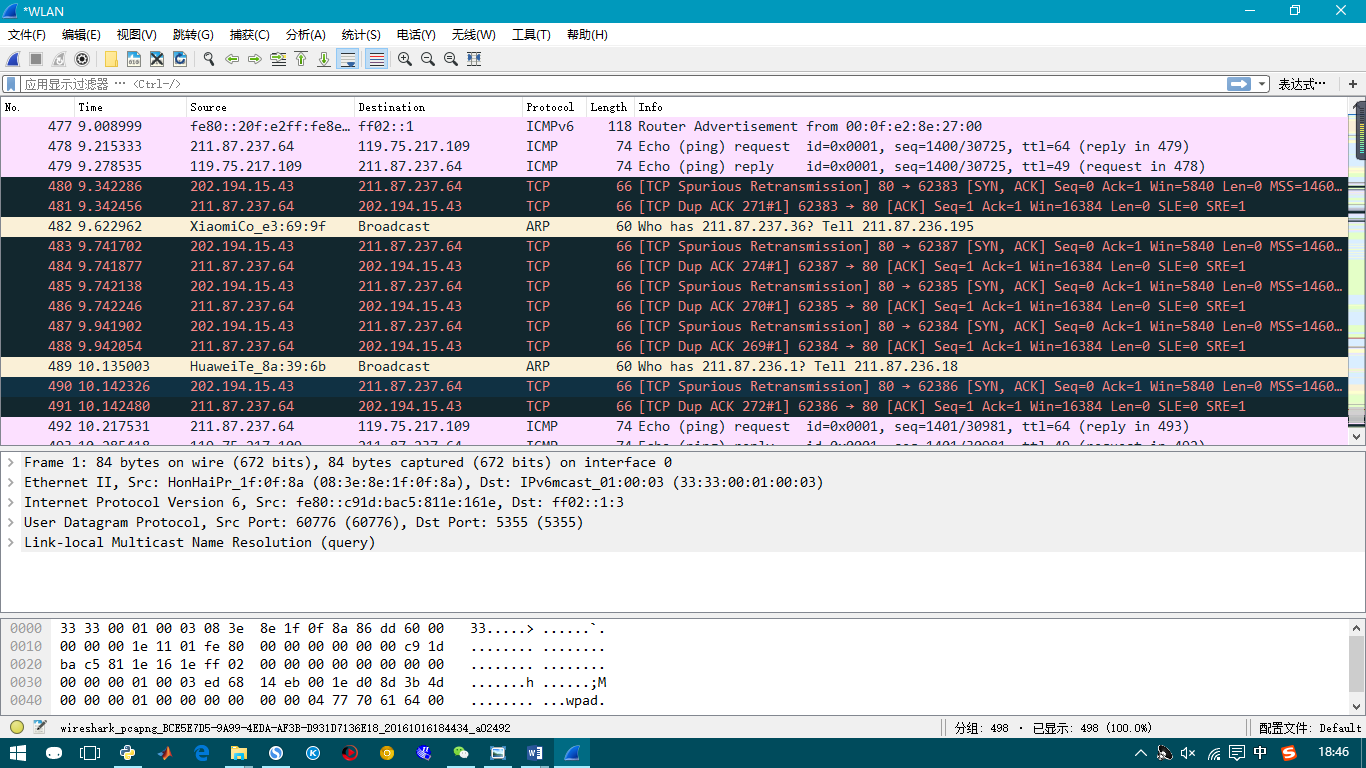
启动搜狗浏览器，然后启动 Wireshark 分组嗅探器。

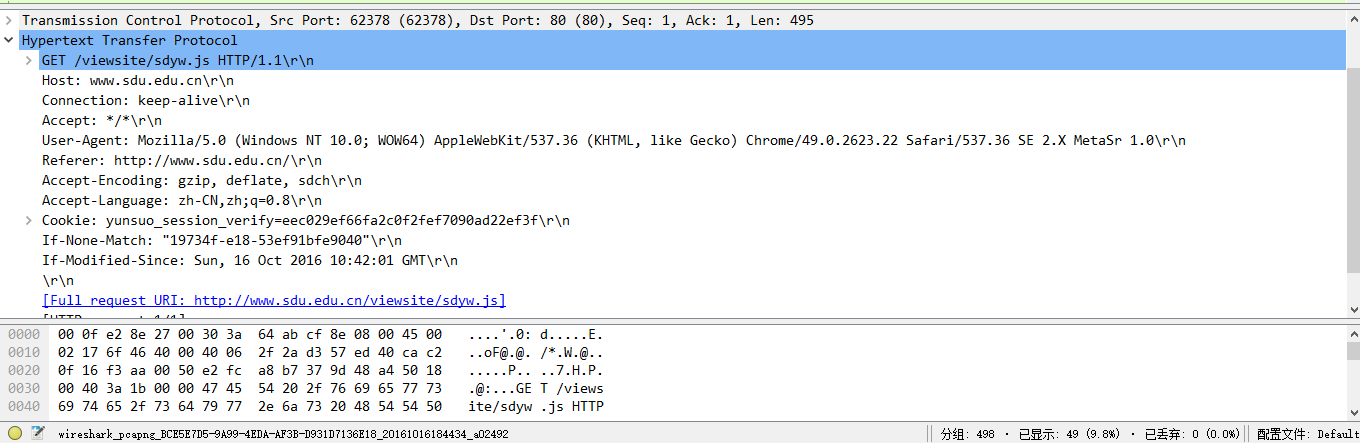
开始 Wireshark 分组俘获。

在打开的搜狗浏览器窗口中输入一下地址 ：<http://www.sdu.edu.cn/>

停止分组俘获。

捕获内容如下：



针对上面捕获的内容，进行分析：

浏览器运行的是 HTTP/1.1。

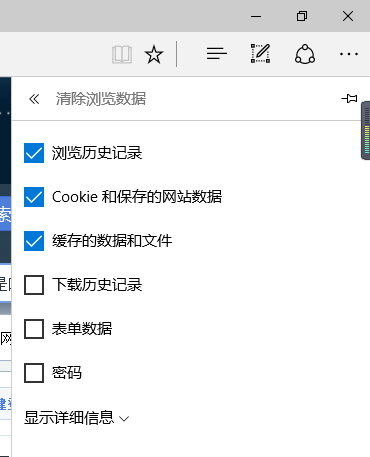
所访问的服务器所运行 HTTP 协议的版本号是HTTP/1.1。

浏览器向服务器指出它能接收的语言版本：zh-CN,zh；q=0.8；

我的计算机的 IP 地址：211.87.237.64；服务器 www.sdu.edu.cn  
的 IP 地址是：202.194.15.22。

### HTTP 条件 GET/response 交互

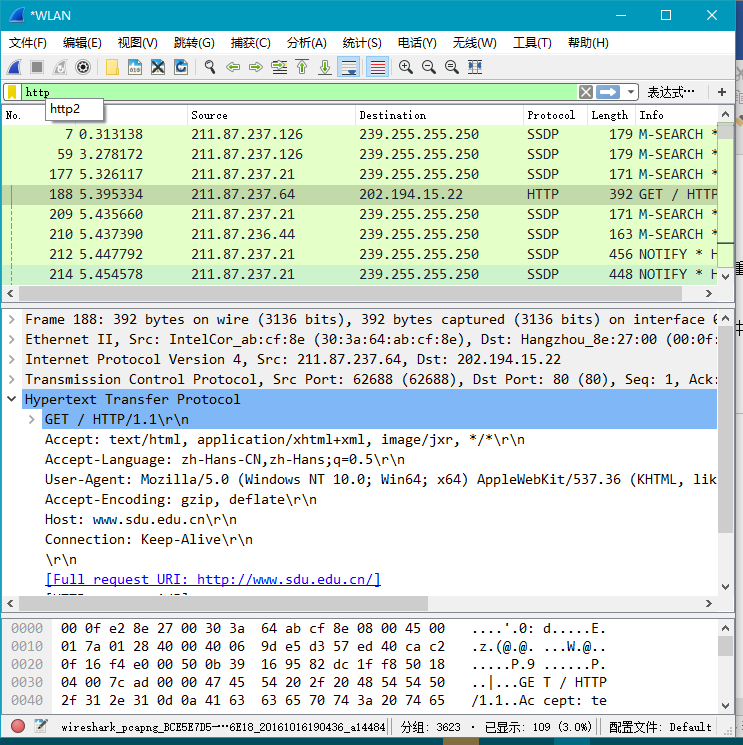
启动Edge浏览器，清空浏览器的缓存



启动 Wireshark 分组俘获器。开始 Wireshark 分组俘获。

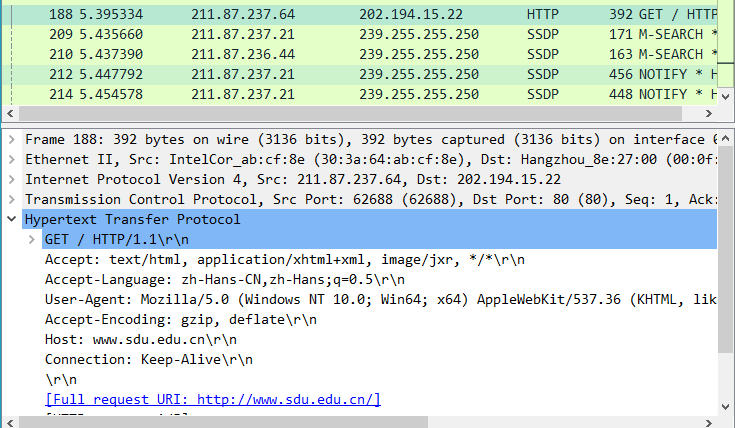
在浏览器的地址栏中输入以下 URL: <http://www.sdu.edu.cn/>,然后在浏览器重新输入相同的 URL 或单击浏览器中的“刷新”按钮。

停止 Wireshark 分组俘获，在显示过滤筛选说明处输入“http”,分组列表子窗口中将只显示所俘获到的 HTTP 报文。

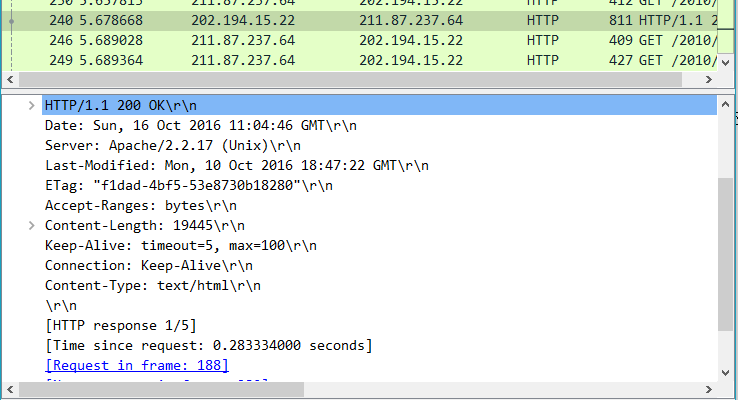


##### 思考分析：

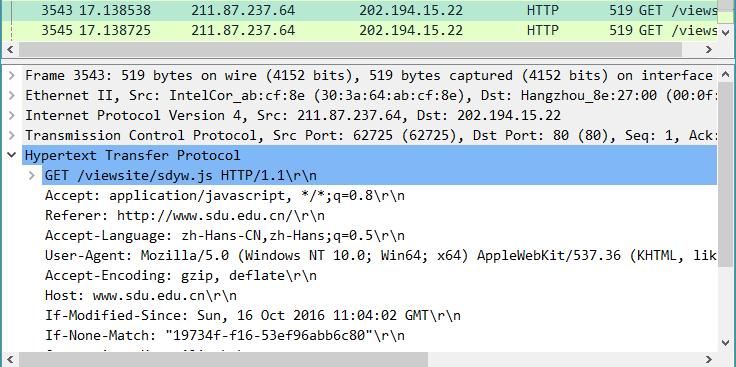
浏览器向服务器发出的第一个 HTTP GET 请求的内容，  
在该请求报文中，没有一行是： IF-MODIFIED-SINCE



分析服务器响应报文的内容，服务器明确返回了文件的内容，因为在HTTP响应报文的实体主体部分，包含了文件内容：



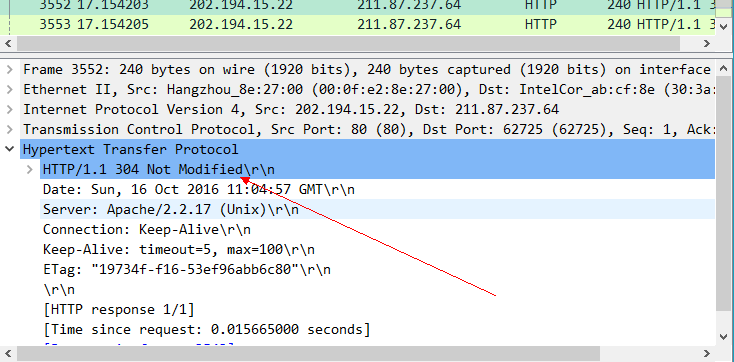
浏览器向服务器发出的较晚的“HTTP GET”请求，在该请求报文中有一行是： IF-MODIFIED-SINCE（这个HTTP请求报文是条件get请求报文）。



含义：这个首部行告诉服务器，仅当指定日期（就是该首部行的值）之后修改过所请求对象才发送该对象，如果没有修改，则服务器所发送的响应报文中不包含该对象。

服务器对较晚的 HTTP GET 请求的响应中的 HTTP 状态代码是304 Not Modified。服务器没有明确返回文件的内容。

因为缓存的对象版本是最新的，包含该对象只是对带宽的浪费。而HTTP 状态代码是304 Not Modified，他告诉缓存器可以使用缓存的对象，向请求的浏览器发送该对象的拷贝（参考《计算机网络》中文第四版73页，有修改）。



### IP 分析

通过分析执行 traceroute 程序发送和接收到的 IP 数据包，研究 IP 数据包的各个字段，并详细研究 IP 分片。

A. 通过执行 traceroute 执行捕获数据包

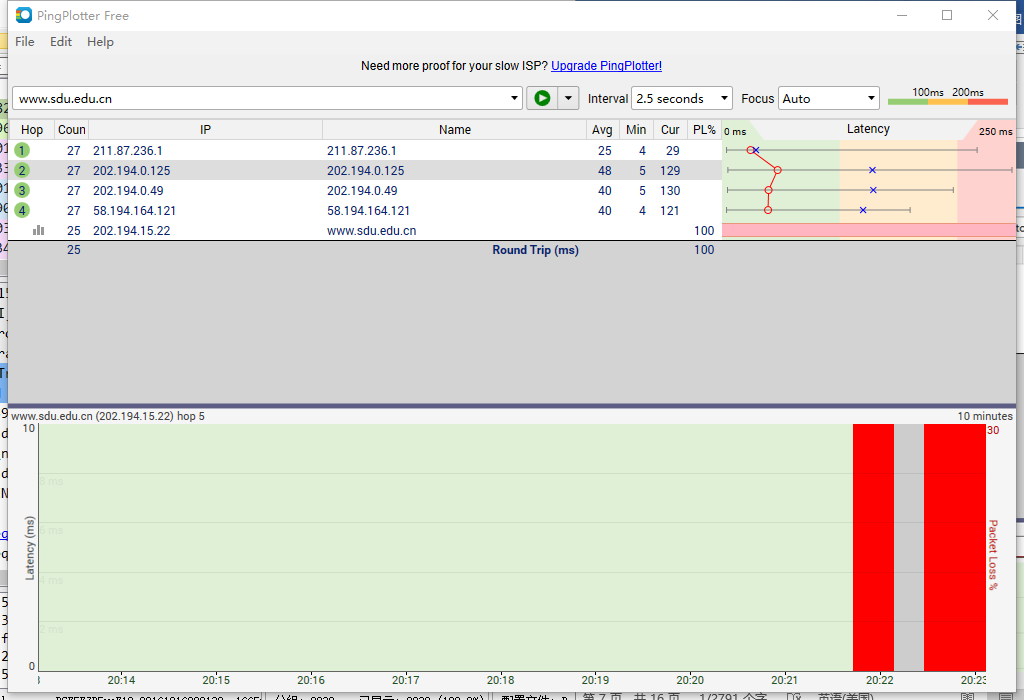
为了产生一系列 IP 数据报，我们利用 traceroute 程序发送具有不同大小的数据包给目的主机 X。源主机发送的第一个数据包的 TTL 设位 1，第二个为 2，第三个为 3，等等。每当路由器收到一个包，都会将其 TTL 值减 1。这样，当第 n 个数据包到达了第 n 个路由器时，第 n 个路由器发现该数据包的 TTL 已经过期了。根据 IP 协议的规则，路由器将该数据包丢弃并将一个 ICMP 警告消息送回源主机。

在 Windows 自带的 tracert 命令不允许用户改变由 tracert 命令发送的 ICMP echo 请求消息（ ping 消息）的大小。一个更优秀的 traceroute 程序是 pingplotter，下载并安装 pingplotter。ICMP echo 请求消息的大小可以通过下面方法在 pingplotter 中进行设置。 Edit->Options->Packet，然后填写 Packet Size(in bytes， default=56)域。

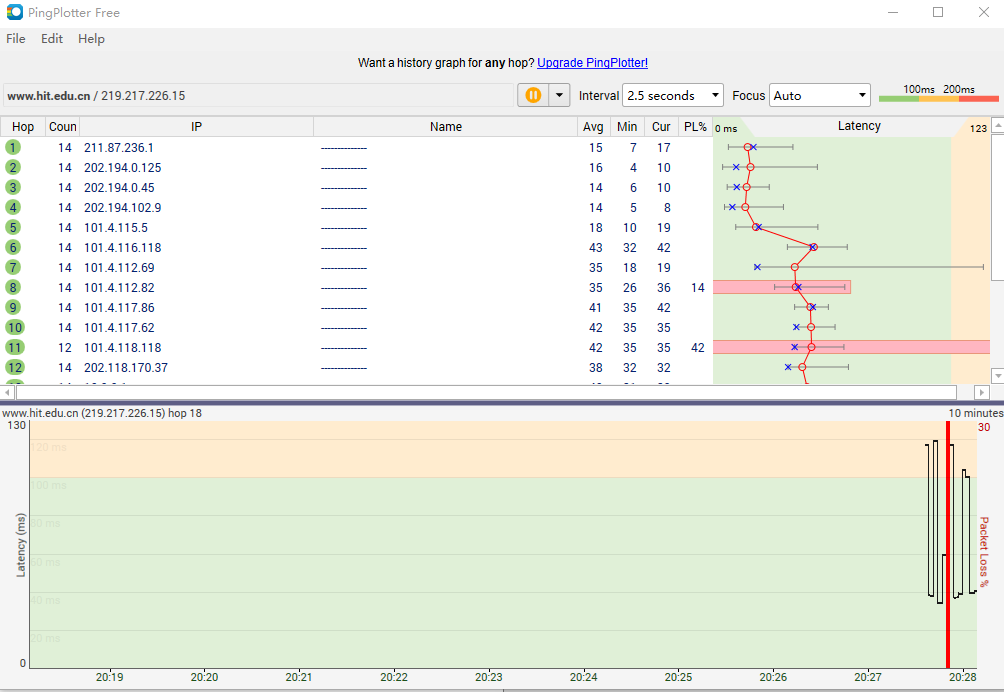
实验步骤：

（ 1） 启动 Wireshark 并开始数据包捕获

（ 2） 启动 pingplotter 并“Address to Trace Window”域中输入目的地址。在“# of times to Trace”域中输入“3”，这样就不过采集过多的数据。 Edit->Options->Packet，将 Packet Size(in bytes,default=56)域设为 56，这样将发送一系列大小为 56 字节的包。 然后按下“Trace”按钮。得到的 pingplotter 窗口如图所示。



去其他网络（比如哈尔滨工业大学官网）



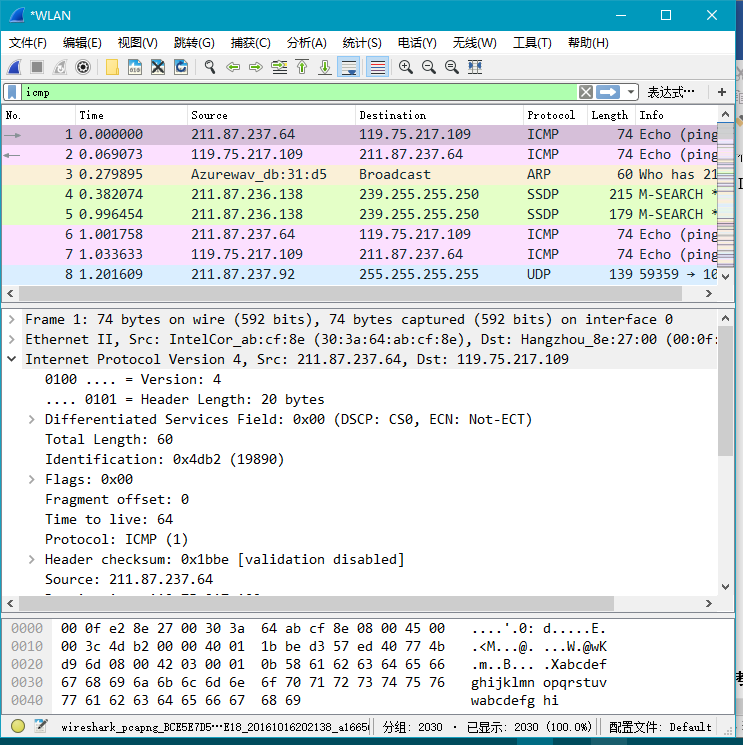
（ 1） Edit->Options->Packet，然后将 Packet Size(in bytes,default=56) 域改为 2000，这样将发送一系列大小为 2000 字节的包。然后按下“Resume”按钮。

（ 2） 最后，将 Packet Size(in bytes,default=56)域改为 3500，发送一 系列大小为 3500 字节的包。然后按下“Resume”按钮。

（ 3） 停止 Wireshark 的分组捕获。

B. 对捕获的数据包进行分析

（ 1）在捕获窗口中，能看到由主机发出的一系列ICMP Echo Request包和中间路由器返回的一系列ICMP TTL-exceeded消息。 选择第一个你的主机发出的ICMP Echo Request消息，在packet details窗口 展开数据包的Internet Protocol部分，如图所示



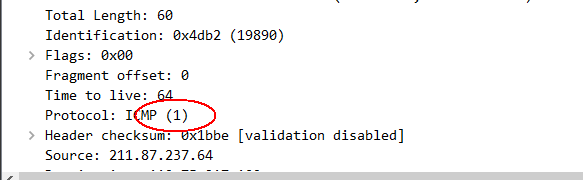
###### 思考分析

⮚主机的IP地址是什么？

211.87.237.64

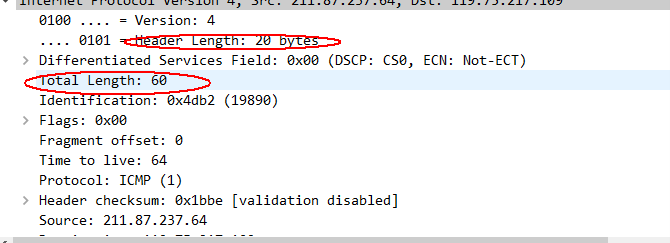
⮚在IP数据包头中，上层协议（ upper layer）字段的值是什么？

是1



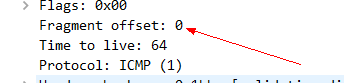
⮚该IP头有多少字节？该IP数据包的净载为多少字节？并解释你是怎样确定该IP数据包的净载大小的？

Ip头有20字节，该ip数据包的净载为60-20=40字节，因为由下面的信息知该ip数据包的总长为60字节，总长减去头部即为净载。

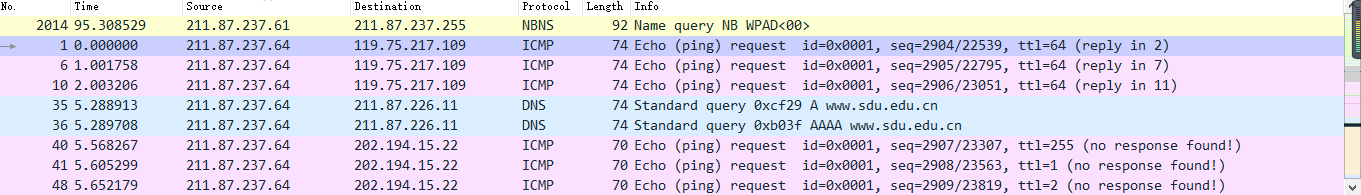


⮚该IP数据包分片了吗？解释你是如何确定该P数据包是否进行了分片

没有分片，有Fragment offest为0知，没有分片。



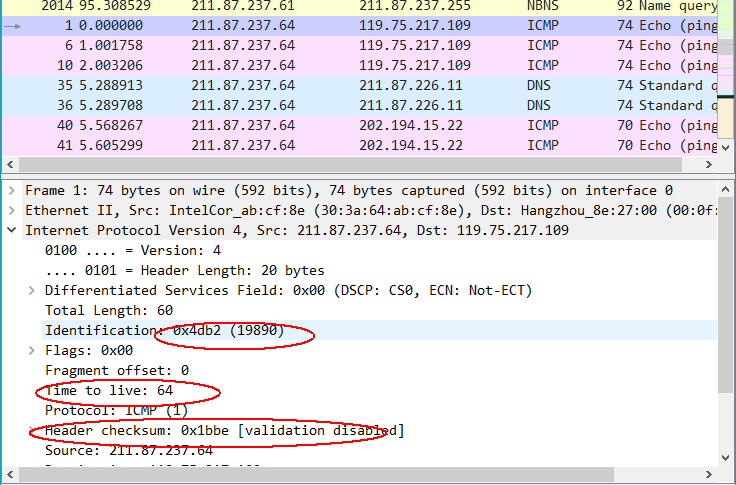
（ 2）单击Source列按钮，这样将对捕获的数据包按源IP地址排序。选择第一个主机发出的ICMP Echo Request消息，在packet details窗口展开数据包的Internet Protocol部分。在“listing of captured packets”窗口，会看到许多后续的ICMP消息（或许还有主机上运行的其他协议的数据包）。结果如图：



##### 思考分析

⮚主机发出的一系列ICMP消息中IP数据报中哪些字段总是发生改变？

IP数据包中的Identification,ttl,header checksum 字段总是发生改变。



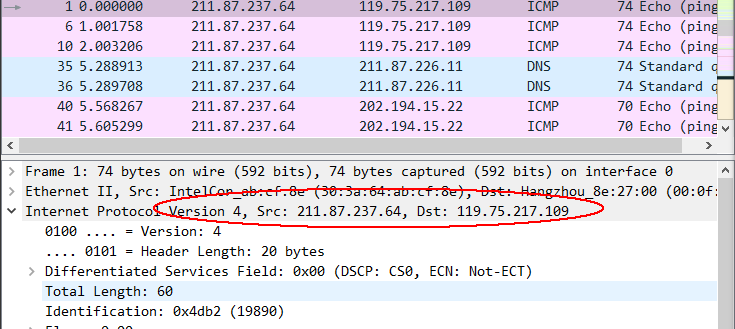
⮚哪些字段必须保持常量？哪些字段必须改变？为什么？

保持常量的字段：Version, Source IP, Destination IP。

版本号规定了数据包的IP协议版本，源IP和目的IP都不能发生变化。

必须改变的字段：identification，首部检验和，TTL。

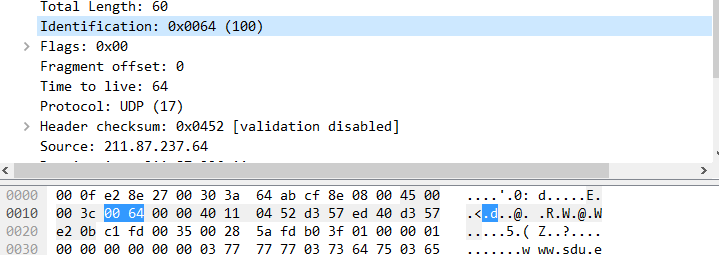
标识字段唯一地标识主机发送的每一份数据报，而校验和是根据前面的数据进行计算，所以会改变。一个IP数据包经过一台路由器转发到另一个网络,该IP数据包的头部字段中一定会发生变化的是TTL。



⮚ IP数据包Identification字段值的形式。

占16位，每产生一个数据报，标识字段加 1，当数据报由于长度超过网络的 MTU 而必须分片时，这个标识字段的值就被复制到所有的数据报的标识字段中。相同的标识字段的值使分片后的各数据报片最后能正确地重装成为原来的数据报。

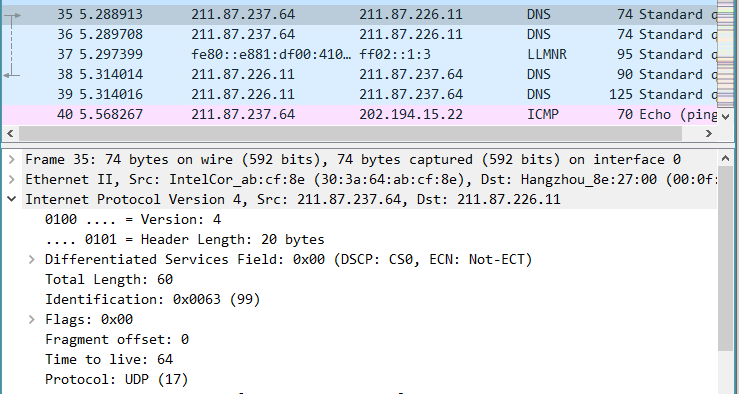




（ 3）找到由最近的路由器（第一跳）返回给你主机的 ICMP Time-to-live exceeded消息。 思考下列问题：啊TTL啊

⮚Identification字段和TTL字段的值是什么？

分别是：0x0063,60

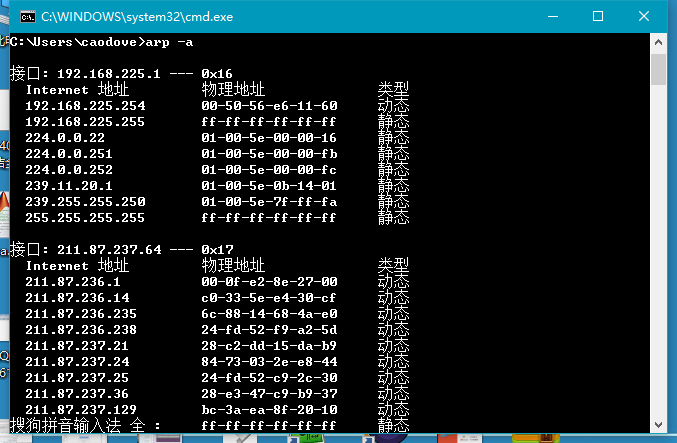


⮚最近的路由器（第一跳）返回给你主机的 ICMP Time-to-live exceeded消息中这些值是否保持不变？为什么？

保持不变，均为64。因为线路和路由器都是固定的，所以数据包的TTL也是固定的。

### 抓取 ARP 数据包

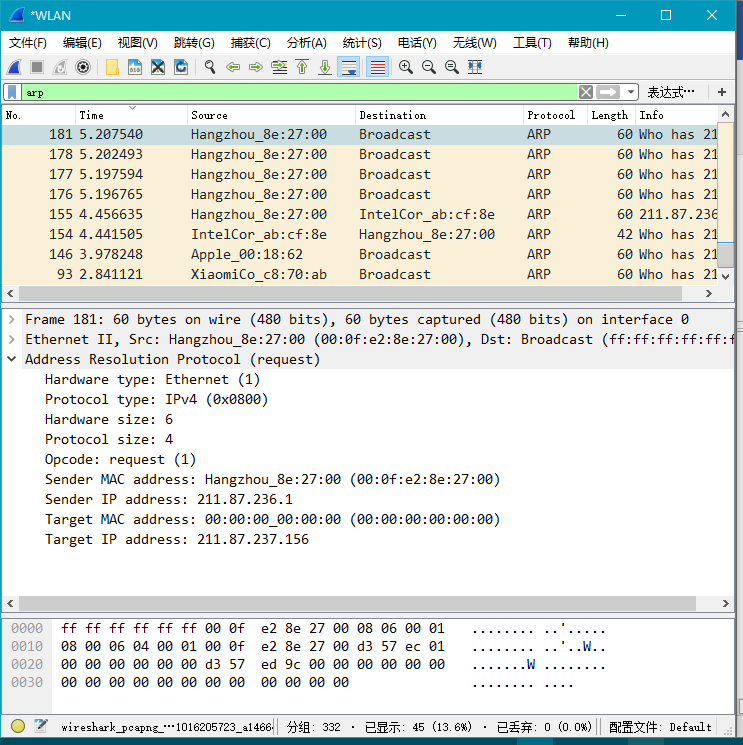
（ 1）利用 MS-DOS 命令： arp 或 c:\windows\system32\arp 查看主机 上 ARP 缓存的内容。



（ 2）在命令行模式下输入： ping 192.168.254.1（实验一中用到过的地址）



（ 3）启动 Wireshark，开始分组俘获。抓取的数据包如下：



##### 思考分析

（ 1）利用 MS-DOS 命令： arp 或 c:\windows\system32\arp 查看主机 上 ARP 缓存的内容。说明 ARP 缓存中每一列的含义是什么?

ARP缓存中每一列的含义是：

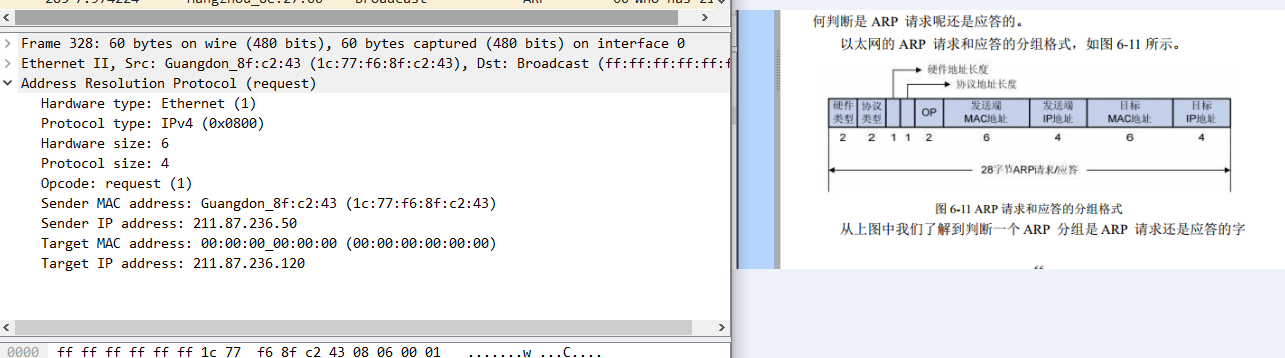
第一列为IP地址，第二列为物理地址，第三列为静态地址

（ 2）清除主机上 ARP 缓存的内容,抓取 ping 命令时的数据包。分析 数据包,回答下面的问题：  
⮚ARP数据包的格式是怎样的？由几部分构成，各个部分所占的字

节数是多少？

ARP数据包的格式是由以太网首部，ARP帧，填充位组成，各个部分所占的字节数分别是以太网首部（14字节），ARP帧（28字节），填充位。

如图所示：



⮚如何判断一个ARP数据是请求包还是应答包？

请求包与应答包看协议的Opcode(request/reply).

⮚为什么ARP查询要在广播帧中传送，而ARP响应要在一个有着明确目的局域网地址的帧中传送？

ARP查询要在广播帧中传送是因为ARP请求MAC地址，而ARP响应要在一个有着明确目的局域网地址的帧中传送是因为不在同一个网段中，无法收到ARP相应。

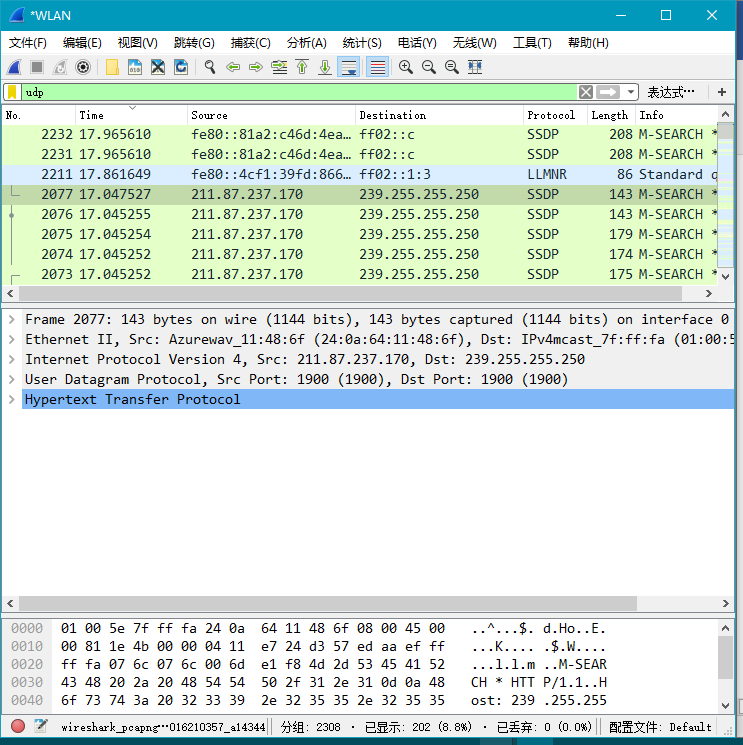
### 抓取 UDP 数据包

（ 1）启动 Wireshark，开始分组捕获；

（ 2）发送 QQ 消息给你的好友；

（ 3）停止 Wireshark 组捕获；

（ 4）在显示筛选规则中输入“udp”并展开数据包的细节，如图：



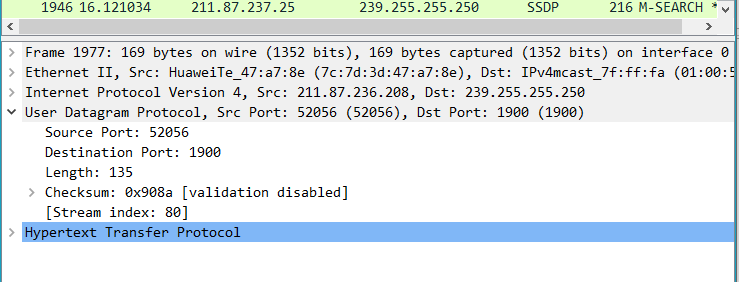
分析 QQ 通讯中捕获到的 UDP 数据包。

思考问题：  
⮚ 消息是基于UDP的还是TCP的？

UDP  
⮚ 你的主机ip地址是什么？目的主机ip地址是什么？

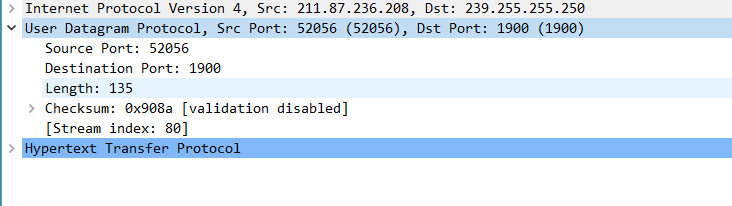
主机：211.87.237.170，目的：239.255.255.250  
⮚你的主机发送QQ消息的端口号和QQ服务器的端口号分别是多少？

发送QQ消息的端口号：52056，QQ服务器的端口号：1900

  
⮚数据报的格式是什么样的？都包含哪些字段，分别占多少字节？

UDP数据报格式如图所示：



  
⮚为什么发送一个ICQ数据包后，服务器又返回给你的主机一个 ICQ数据包？这UDP的不可靠数据传输有什么联系？对比前面的TCP协议分析，你能看出UDP是无连接的吗？

发送一个ICQ数据包后，服务器又返回给你的主机一个ICQ数据包，是为了确认状态。正因为UDP的数据传输为不可靠的，所以服务器发送一个数据包来确认刚刚发送的数据状态。

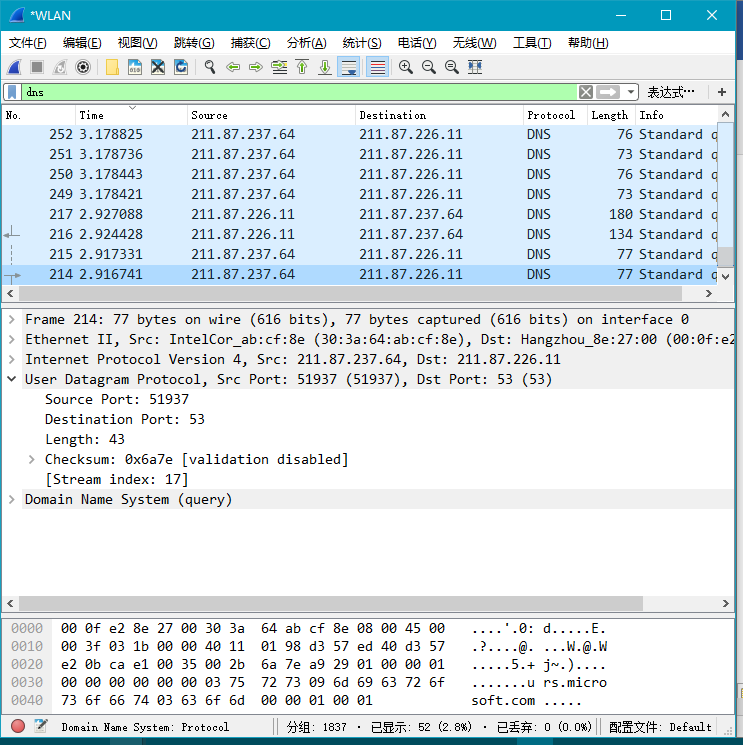
能看出来：和TCP不同，UDP是一个无连接的传输协议。TCP用SYN分组和SYN ACK分组来显示地打开一个连接，而UDP却直接开始发送包含数据的分组。同样，TCP用FIN分组和FIN ACK分组来显示地关闭一个连接，而UDP却只简单地停止包含数据的分组的传输。

### 利用 WireShark 进行 DNS 协议分析

（ 1）打开浏览器键入:www.baidu.com

（ 2）打开 Wireshark,启动抓包.

（ 3）在控制台回车执行完毕后停止抓包.Wireshark 捕获的 DNS 报文如图：



##### 分析

定位到DNS查询报文和查询响应报文，这两种报文的发送是基于TCP的。

检查DNS查询报文，DNS查询为A类查询，该查询报文中不包含“answers”

TYPE A

Answers RRs: 0.

## 第二大部分：ntop分析数据流量

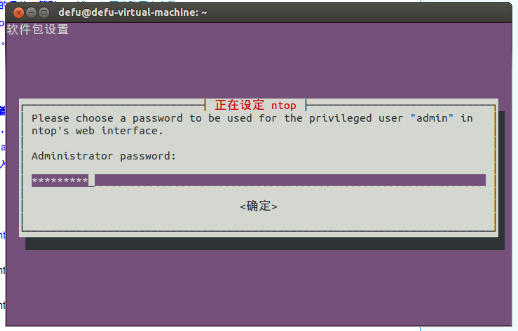
### 在Ubuntu 12.04 系统下配置ntop，相关命令如下：

安装：

sudo apt-get install ntop

设置管理员密码：

sudo ntop --set-admin-password



启动和停止ntop服务：  
    启动：

    sudo /etc/init.d/ntop start    
    停止：

    sudo /etc/init.d/ntop stop   
    重启：

    sudo /etc/init.d/ntop restart

（在这里试过好多教程，这个版本是最快捷有用的，而且启动服务也不会报类似的数据库错误之类的）

相关补充：

Ntop工作时需要使用zlib、gd、libpcap及libpng的函数，安装前须检查服务器中是否已经含有下列的软件：zlib（zlib-1.1.3-xx以上）、gd（gd-1.3.xx以上）、libpng。可以使用RPM来确认：

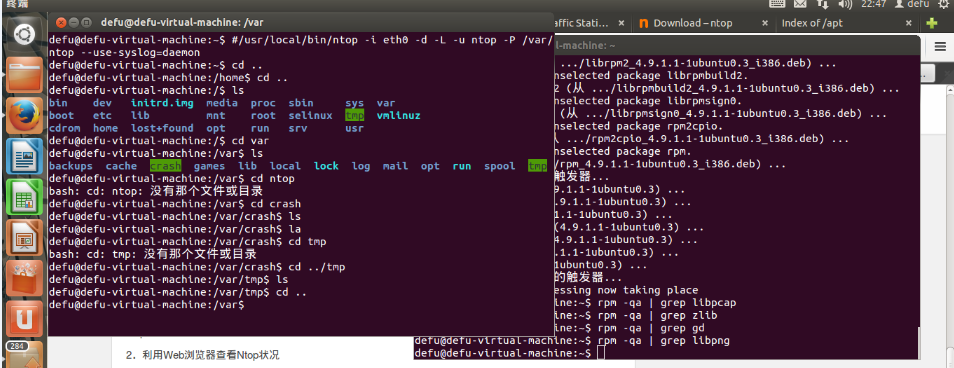
rpm -qa | grep libpcap

rpm -qa | grep zlib

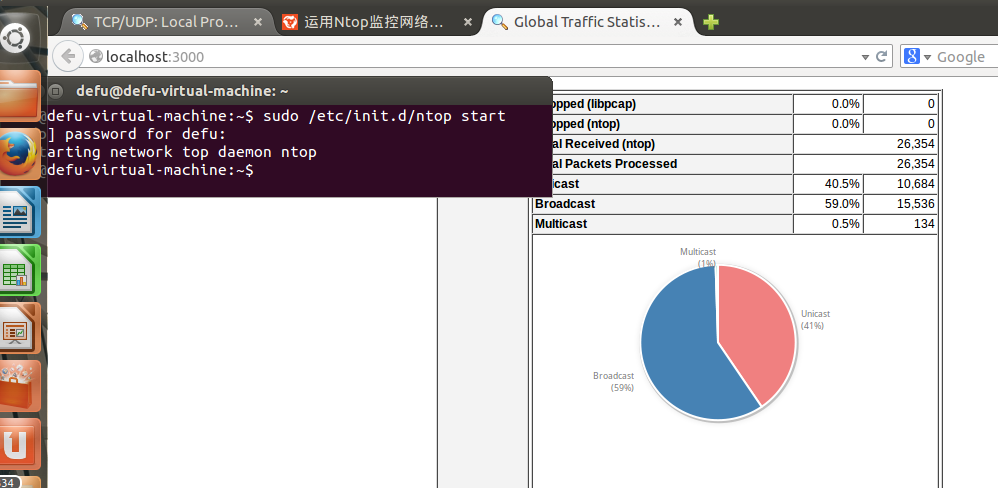
rpm -qa | grep gd

rpm -qa | grep libpng

配置成功结果如图：



### 正常启动并分析数据



### 功能分析：

（一）Traffic

（1）Traffic:显示全局流量统计，包括活动接口流量统计，全局协议分布，TCP/UDP协议分布及端口流量分布统计

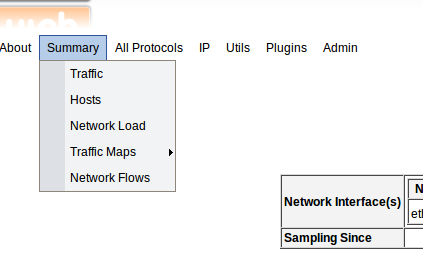
Host：显示左右可见主机信息

Network load：网络覆在统计，显示10min,1 h,1d的流量统计

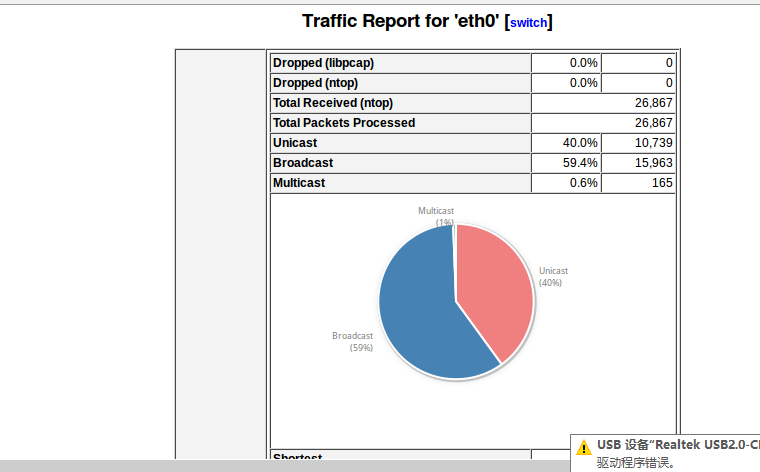
Hosts World map：显示世界各地主机分布图

Network flows：列出用户定义的流的规制信息

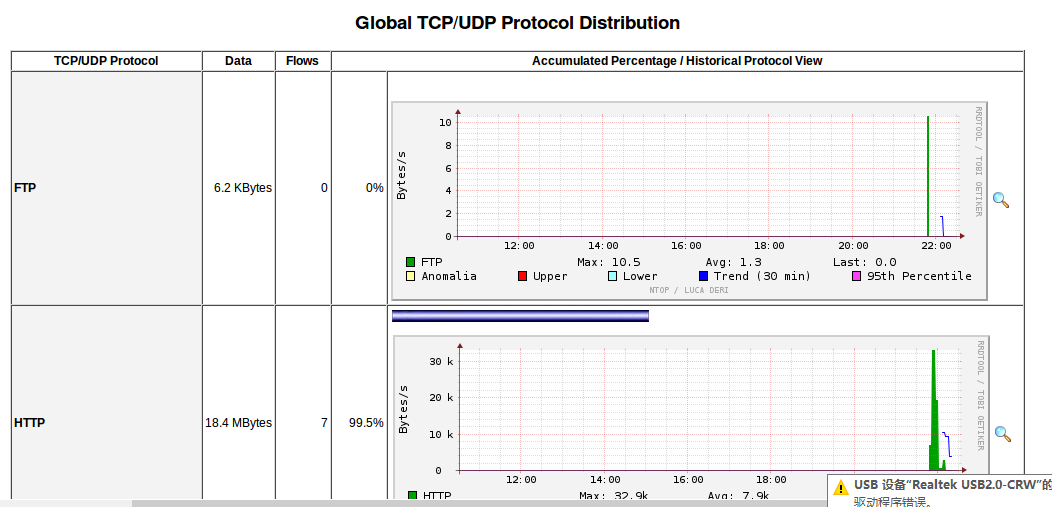
Summary->traffic:对监听的网卡接口所捕获的packets进行分析：



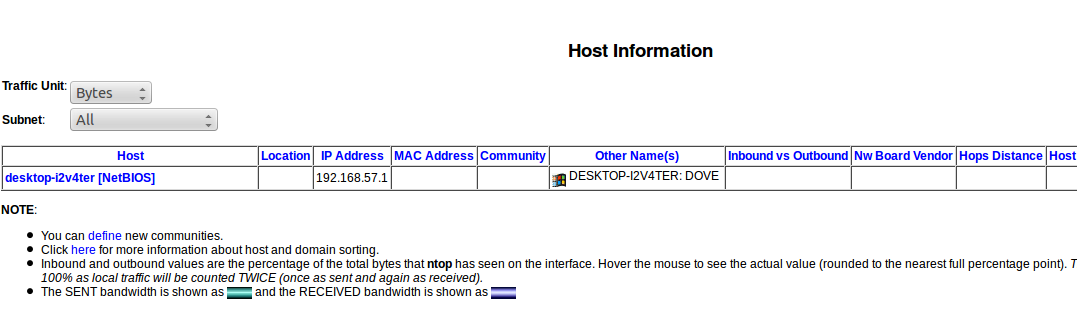
（2）



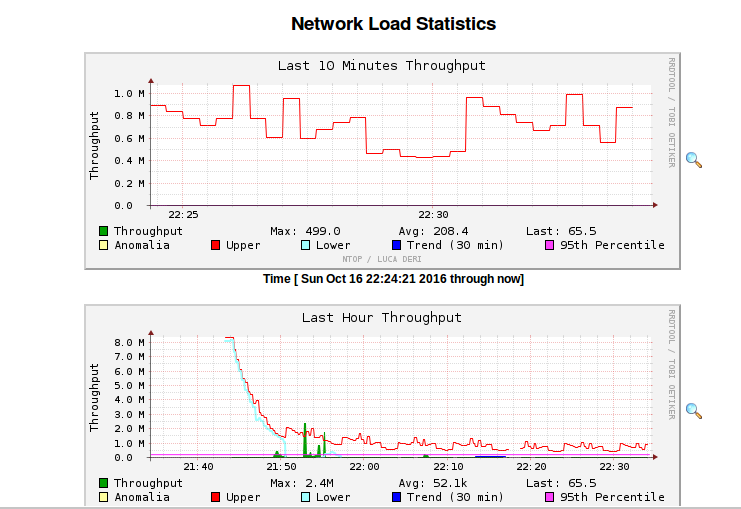
（3）summary->traffic：所有的TCP/UDP协议分布如图所示，该图对网络的一些协议类型的流量进行了统计分析。



（4）Summary –host 就是每台主机的流量情况，点击相应的host就可以了解更详细的信息



（5）summary->network load statistics:该项可以查看10分钟，一个小时的网络流量统计：



（二）：All protocols 选项

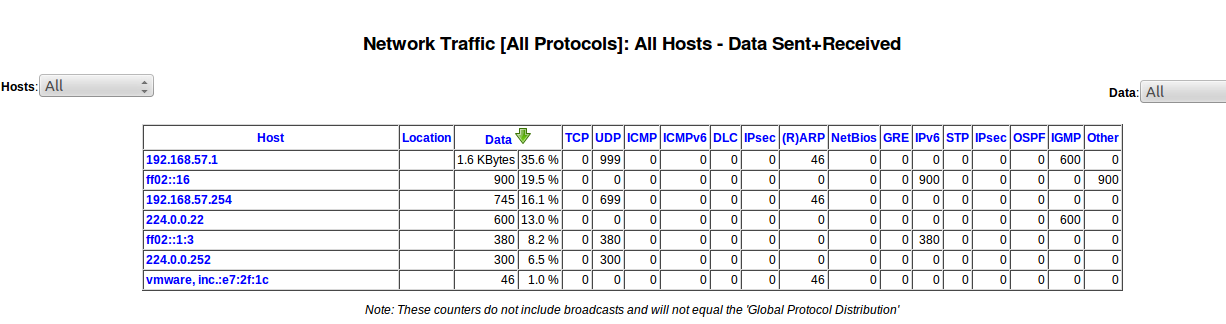
Traffic：列出每个可见主机的流量信息

Throughput：显示网络吞吐量

Activity：显示可见主机每小时的流量

All protocols->traffic:该页面统计了所有课件主机的网络流量信息，可以查看分布位置，是DNS,HTTP,DHCP等，包含了TCP,UDP和一些其他协议，要查看具体某个主机的流量信息，只需点击对应的Host即可查看。

看分布位置，是DNS,HTTP,DHCP等，包含了TCP,UDP和一些其他协议，要查看具体某个主机的流量信息，只需点击对应的Host即可查看

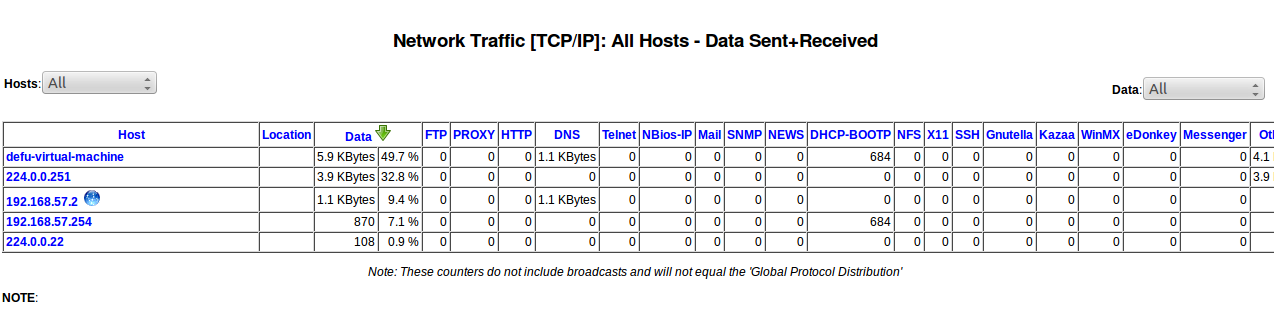


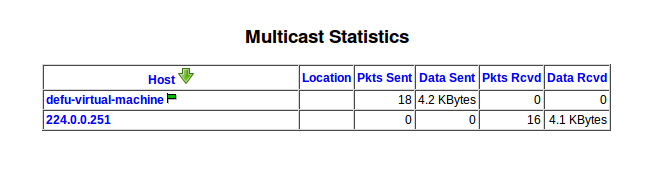
（三）IP选项

IP选项提供的功能：



相应操作如下：

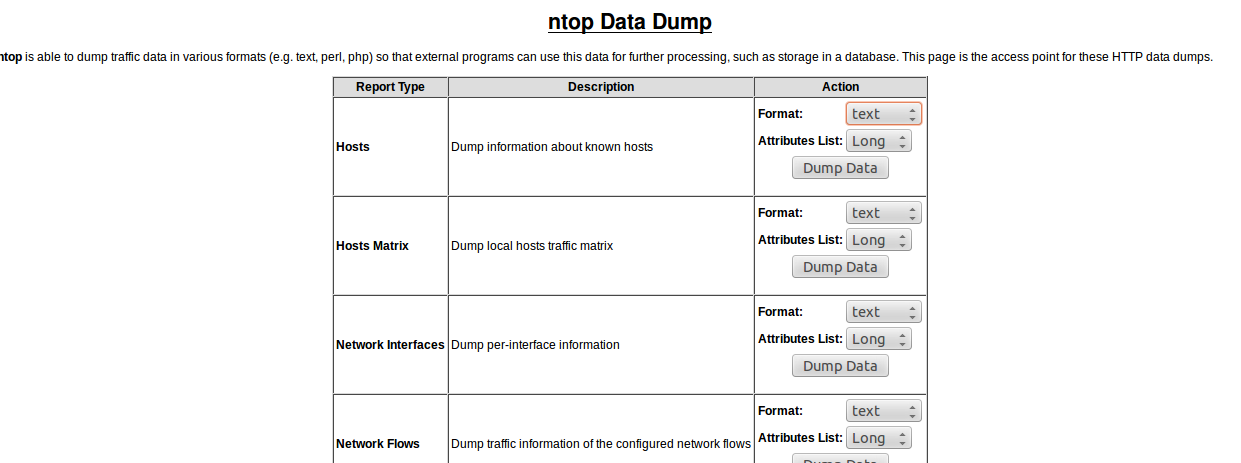




（四）：其他相应操作及结束运行

转储功能：

Ntop还支持把流量转储成其他格式（如文本文件、Perl、[PHP](http://www.2cto.com/kf/web/php/)、[Python](http://www.2cto.com/kf/web/Python/)），以便其他外部程序可以对数据进行深加工。可以选择Utils→Data Dump命令，相关操作如下：



停止运行：

后来在切断网络连接的时候，发现网址已经连不上。



## 实验心得

这次实验相对花的时间还是很长的，然而，收获也是巨大的。学会并且恰当使用了wireshark以及ntop这两个流量分析的工具。这是两款十分切实可用的软件，分析的过程以及抓包的过程都相对简单，重点在于我们对已经获得的数据的分析。但是如果使用不当，可能会对个人的一些隐私造成泄漏，给有恶意的人带来可趁之机，为互联网带来安全隐患。

这次实验不但加深了我对每个报文的每个段的理解，更是扫除了之前课上的好多知识盲点，搞明白了许多计算机网络之间数据传输及服务器与客户端响应的问题，让我更加清晰的了解到了流量的变动以及具体的分析。

这次实验遇到了一个影响深刻的问题是：实验过程中由于第一次使用Wireshark，很多功能都不太了解，抓到的数据包太多无法准确确定。我搞了好久，才从网上学习到了两个方法，除了通过包的类型的筛选以外，关闭其他的一些后台程序减少网络数据包的交互，也能够更好容易的定位包。这样就解决了我为什么没打开浏览器还是能抓到包的问题