

实验一 网络常用命令的使用及 DNS 层次查询、SMTP 协议分析； 利用分组嗅探器（Wireshark）进行应用层协议分析

实验内容一：网络常用命令的使用

■ windows 命令

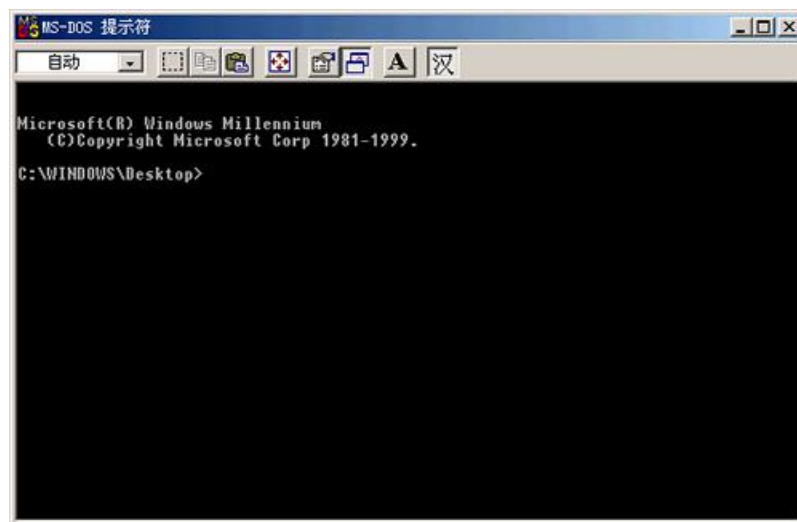
不同的操作系统要用不同的命令进入命令行界面。

在 Win9x/Me 的开始菜单中的运行程序中键入"command"命令，可进入命令行界面。在 Win2000/NT 的开始菜单中的运行程序中键入"cmd"命令，可进入命令行界面。

开始——> 运行——> 键入 cmd 命令或 command 命令——> 回车



进入了命令行操作界面（DOS 窗口），在 DOS 窗口中只能用键盘来操作。如下所示：



■ 网络常用命令的作用与格式

了解和掌握网络常用命令将会有助于更快地检测到网络故障所在，从而节省时间，提高效率。网络命令数量比较多，在本次实验中我们学习的网络命令是为数不多的一些常用网络命令。

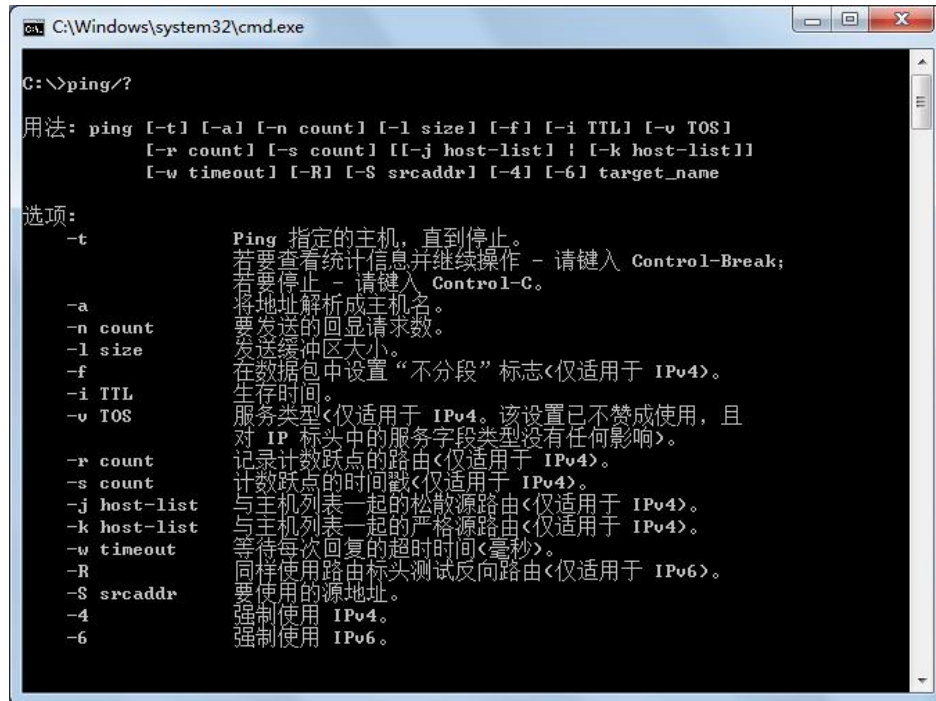
由于每个网络命令都有不同的作用，为了更好地掌握这些网络常用命令应该了解这些命

令的基本格式，基本格式如下：

网络命令 参数 1 参数 2 参数 3 参数...

查看这些参数的方法是在网络命令后加“/?”，如要查看 ping 命令的参数可以输入 ping/?

显示如下：



```
C:\Windows\system32\cmd.exe

C:\>ping/?

用法: ping [-t] [-a] [-n count] [-l size] [-f] [-i TTL] [-v TOS]
          [-r count] [-s count] [[-j host-list] ! [-k host-list]]
          [-w timeout] [-R] [-S srcaddr] [-4] [-6] target_name

选项:
  -t           Ping 指定的主机，直到停止。
               若要查看统计信息并继续操作 - 请键入 Control-Break;
               若要停止 - 请键入 Control-C。
  -a           将地址解析成主机名。
  -n count     要发送的回显请求数。
  -l size      发送缓冲区大小。
  -f           在数据包中设置“不分段”标志<仅适用于 IPv4>。
  -i TTL       生存时间。
  -v TOS       服务类型<仅适用于 IPv4。该设置已不赞成使用，且
               对 IP 标头中的服务字段类型没有任何影响>。
  -r count     记录计数跃点的路由<仅适用于 IPv4>。
  -s count     计数跃点的时间戳<仅适用于 IPv4>。
  -j host-list 与主机列表一起的松散源路由<仅适用于 IPv4>。
  -k host-list 与主机列表一起的严格源路由<仅适用于 IPv4>。
  -w timeout   等待每次回复的超时时间<毫秒>。
  -R           同样使用路由标头测试反向路由<仅适用于 IPv6>。
  -S srcaddr   要使用的源地址。
  -4           强制使用 IPv4。
  -6           强制使用 IPv6。
```

【实验目的】

- 1、掌握网络常用命令的使用；
- 2、利用网络常用命令对网络中常见现象进行分析判断。

【实验内容】

1、掌握 PING 命令的基本使用方法（包括参数的使用），对网络常见故障利用命令进行分析判断：

Ping 是测试网络联接状况以及信息包发送和接收状况非常有用的工具，是网络测试最常用的命令。Ping 向目标主机(地址)发送一个回送请求数据包，要求目标主机收到请求后给予答复，从而判断网络的响应时间和本机是否与目标主机(地址)联通。

如果执行 Ping 不成功，则可以预测故障出现在以下几个方面：网线故障，网络适配器配置不正确，IP 地址不正确。如果执行 Ping 成功而网络仍无法使用，那么问题很可能出在网络系统的软件配置方面，Ping 成功只能保证本机与目标主机间存在一条连通的物理路径。

命令格式：

ping IP 地址或主机名 [-t] [-a] [-n count] [-l size]

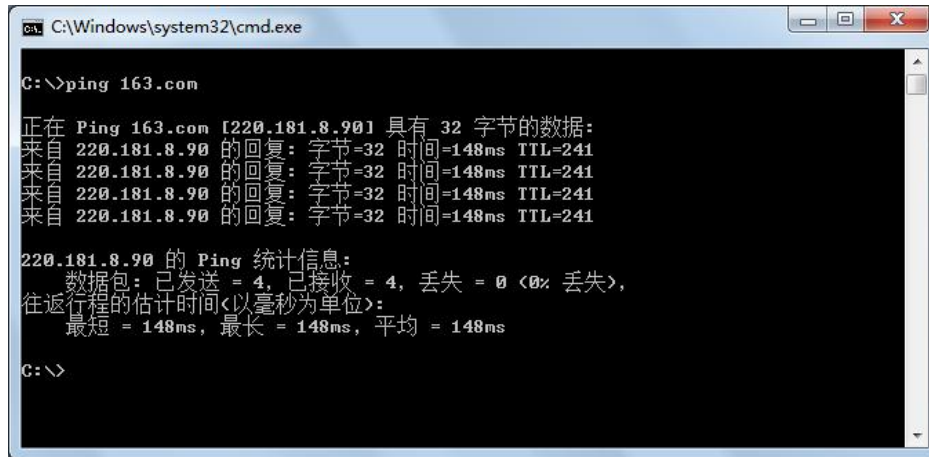
常用参数含义：

-t 不停地向目标主机发送数据；

-a 以 IP 地址格式来显示目标主机的网络地址 ；

-n count 指定要 Ping 多少次，具体次数由 count 来指定 ；

-l size 指定发送到目标主机的数据包的大小。



```
C:\Windows\system32\cmd.exe
C:\>ping 163.com

正在 Ping 163.com [220.181.8.90] 具有 32 字节的数据:
来自 220.181.8.90 的回复: 字节=32 时间=148ms TTL=241
来自 220.181.8.90 的回复: 字节=32 时间=148ms TTL=241
来自 220.181.8.90 的回复: 字节=32 时间=148ms TTL=241
来自 220.181.8.90 的回复: 字节=32 时间=148ms TTL=241

220.181.8.90 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
    往返行程的估计时间(以毫秒为单位):
        最短 = 148ms, 最长 = 148ms, 平均 = 148ms

C:\>
```

2、用 Tracert 命令用来显示数据包到达目标主机所经过的路径，并显示到达每个节点的时间，分析网络延时产生的原因。

Tracert 命令用来显示数据包到达目标主机所经过的路径，并显示到达每个节点的时间。命令功能同 Ping 类似，但它所获得的信息要比 Ping 命令详细得多，它把数据包所走的全部路径、节点的 IP 以及花费的时间都显示出来。该命令比较适用于大型网络。

命令格式：

tracert IP 地址或主机名 [-d][-h maximumhops][-j host_list] [-w timeout]

参数含义：

-d 不解析目标主机的名字；

-h maximum_hops 指定搜索到目标地址的最大跳跃数；

-j host_list 按照主机列表中的地址释放源路由；

-w timeout 指定超时时间间隔，程序默认的时间单位是毫秒。

例如大家想要了解自己的计算机与目标主机 www.163.com 之间详细的传输路径信息，可以在 MS-DOS 方式输入 www.163.com。

```
C:\Windows\system32\cmd.exe

C:\>tracert www.163.com

通过最多 30 个跃点跟踪
到 163.xdws.cache.g1b0.lxdns.com [60.210.18.169] 的路由:

 1  <1 毫秒    <1 毫秒    <1 毫秒    172.27.38.1
 2  <1 毫秒    <1 毫秒    <1 毫秒    172.25.2.29
 3  <1 毫秒    <1 毫秒    <1 毫秒    172.25.0.2
 4  <1 毫秒    <1 毫秒    <1 毫秒    221.2.164.1
 5  1 ms       <1 毫秒    <1 毫秒    221.2.131.25
 6  21 ms      9 ms       9 ms       60.215.136.21
 7  11 ms      11 ms      12 ms      60.215.131.122
 8  11 ms      11 ms      11 ms      221.1.4.90
 9  10 ms      10 ms      10 ms      61.156.17.38
10  11 ms      11 ms      11 ms      60.210.18.169

跟踪完成。

C:\>
```

如果我们在 Tracert 命令后面加上一些参数，还可以检测到其他更详细的信息，例如使用参数-d，可以指定程序在跟踪主机的路径信息时，同时也解析目标主机的域名。

3、利用 Netstat 命令了解网络的整体使用情况。显示当前正在活动的网络连接的详细信息，例如显示网络连接、路由表和网络接口信息，统计目前总共有哪些网络连接正在运行。

Netstat 命令可以帮助网络管理员了解网络的整体使用情况。它可以显示当前正在活动的网络连接的详细信息，例如显示网络连接、路由表和网络接口信息，可以统计目前总共有哪些网络连接正在运行。

利用命令参数，命令可以显示所有协议的使用状态，这些协议包括 TCP 协议、UDP 协议以及 IP 协议等，另外还可以选择特定的协议并查看其具体信息，还能显示所有主机的端口号以及当前主机的详细路由信息。

命令格式：

netstat [-r] [-s] [-n] [-a]

参数含义：

- a 显示所有连接和侦听端口。
- b 显示在创建每个连接或侦听端口时涉及的可执行程序。

在某些情况下，已知可执行程序承载多个独立的组件，这些情况下，显示创建连接或侦听端口时涉及的组件序列。此情况下，可执行程序的名称位于底部[]中，它调用的组件位于顶部，直至达到 TCP/IP。注意，此选项可能很耗时，并且在您没有足够权限时可能失败。

- e 显示以太网统计。此选项可以与 -s 选项结合使用。
- f 显示外部地址的完全限定域名(FQDN)。

-n 以数字形式显示地址和端口号。

-o 显示拥有的与每个连接关联的进程 ID。

-p proto 显示 proto 指定的协议的连接；proto 可以是下列任何一个: TCP、UDP、TCPv6 或 UDPv6。如果与 -s 选项一起用来显示每个协议的统计，proto 可以是下列任何一个: IP、IPv6、ICMP、ICMPv6、TCP、TCPv6、UDP 或 UDPv6。

-r 显示路由表。

-s 显示每个协议的统计。默认情况下，显示 IP、IPv6、ICMP、ICMPv6、TCP、TCPv6、UDP 和 UDPv6 的统计；-p 选项可用于指定默认的子网。

-t 显示当前连接卸载状态。

interval 重新显示选定的统计，各个显示间暂停的间隔秒数。

按 CTRL+C 停止重新显示统计。如果省略，则 netstat 将打印当前的配置信息一次。

4、利用 IPCONFIG 命令显示所有当前的 TCP/IP 网络配置值、刷新动态主机配置协议 (DHCP) 和域名系统 (DNS) 设置。使用不带参数的 IPCONFIG 显示所有适配器的 IP 地址、子网掩码、默认网关。

命令格式：

Ipconfig[/all][[/batch file]][/renew all][[/release all]][/renew n][[/release n]]

参数含义：

/? 显示帮助信息

/all 显示现时所有网络连接的设置

/release 释放某一个网络上的 IP 位置

/renew 更新某一个网络上的 IP 位置

/flushdns 把 DNS 解析器的暂存内容全数删除

5、利用 ARP 确定对应 IP 地址的网卡物理地址。查看本地计算机或另一台计算机的 ARP 高速缓存中的当前内容。

在以太网协议中规定，同一局域网中的一台主机要和另一台主机进行直接通信，必须要知道目标主机的 MAC 地址。而在 TCP/IP 协议栈中，网络层和传输层只关心目标主机的 IP 地址。这就导致在以太网中使用 IP 协议时，数据链路层的以太网协议接到上层 IP 协议提供的的数据中，只包含目的主机的 IP 地址。于是需要一种方法，根据目的主机的 IP 地址，获得

其 MAC 地址。这就是 ARP 协议要做的事情。所谓地址解析（address resolution）就是主机在发送帧前将目标 IP 地址转换成目标 MAC 地址的过程。

另外，当发送主机和目的主机不在同一个局域网中时，即便知道目的主机的 MAC 地址，两者也不能直接通信，必须经过路由转发才可以。所以此时，发送主机通过 ARP 协议获得的将不是目的主机的真实 MAC 地址，而是一台可以通往局域网外的路由器的 MAC 地址。于是此后发送主机发往目的主机的所有帧，都将发往该路由器，通过它向外发送。这种情况称为 ARP 代理（ARP Proxy）

命令格式：

```
arp[-a [InetAddr] [-N IfaceAddr]] [-g [InetAddr] [-N IfaceAddr]] [-d InetAddr [IfaceAddr]]  
[-s InetAddr EtherAddr [IfaceAddr]]
```

参数含义：

-a [InetAddr] [-N IfaceAddr] 显示所有接口的当前 ARP 缓存表。要显示特定 IP 地址的 ARP 缓存项，请使用带有 InetAddr 参数的 **arp -a**，此处的 InetAddr 代表 IP 地址。如果未指定 InetAddr，则使用第一个适用的接口。要显示特定接口的 ARP 缓存表，请将 **-N IfaceAddr** 参数与 **-a** 参数一起使用，此处的 IfaceAddr 代表指派给该接口的 IP 地址。**-N** 参数区分大小写。

-g [InetAddr] [-N IfaceAddr] 与 **-a** 相同。

-d InetAddr [IfaceAddr] 删除指定的 IP 地址项，此处的 InetAddr 代表 IP 地址。对于指定的接口，要删除表中的某项，请使用 IfaceAddr 参数，此处的 IfaceAddr 代表指派给该接口的 IP 地址。要删除所有项，请使用星号 (*) 通配符代替 InetAddr。

-s InetAddr EtherAddr [IfaceAddr] 向 ARP 缓存添加可将 IP 地址 InetAddr 解析成物理地址 EtherAddr 的静态项。要向指定接口的表添加静态 ARP 缓存项，请使用 IfaceAddr 参数，此处的 IfaceAddr 代表指派给该接口的 IP 地址。

/? 在命令提示符下显示帮助。

6、课上补充讲解其他网络命令的使用。

【实验方式】现场参观并由实验指导教师讲解、演示；分组讨论与实践。

【实验地点】学院实验室。

【实验报告】在实验报告中写出网络常用命令的操作过程及效果，分析并总结实验中遇到的问题，写出实验体会。

实验内容二：DNS 层次查询、SMTP 协议分析

【实验前需要学习掌握的知识】

- 1、掌握 DNS 基本构成原理及三层结构。
- 2、电子邮件系统的构成，包含在发送方、接收方进行邮件传递涉及的各种协议及协议构成，区分 SMTP 协议与邮件消息格式的异同点。
- 3、了解常用捕包软件。捕包软件不但可以分析数据包的流向，也可以对数据包的内容进行监听，可以观察 TCP/IP 协议族中应用层、传输层、网络层、数据链路层和有关网络安全的各种协议的活动。

【实验目的】

- 1、了解和掌握 DNS 层次结构，利用 NSLOOKUP 命令对 DNS 层次结构进行访问；
- 2、了解电子邮件系统发送及接受处理过程，对 SMTP 协议进行分析；
- 3、掌握捕包软件 Wireshark 的使用，了解网络协议实体间进行交互以及报文交换的情况；

【实验内容】

1、熟练掌握 nslookup 命令，并对 nslookup 命令的参数进行熟练掌握。

具体参数： nslookup -qt=类型 目标域名

注意 qt 必须小写。类型可以是一下字符，不区分大小写：

A 地址记录(Ipv4)

CNAME 别名记录

MX 邮件服务器记录

NS 名字服务器记录

PTR 反向记录（从 IP 地址解释域名）

RP 负责人记录

RT 路由穿透记录

SRV TCP 服务器信息记录

TXT 域名对应的文本信息

X25 域名对应的 X.25 地址记录

实验具体内容要求：到网上查找 13 个根名称的 IP；任选一个根名称服务器，利用 NSLOOKUP，在根名称服务器、顶级域名称服务器、权威名称服务器上，手动逐级进行 NDS

解析，并进行记录和分析；在本地名称服务器，利用 NSLOOKUP，手动逐级进行 NDS 解析，并进行记录和分析。

2、利用 TELNET 进行 SMTP 的邮件发送。

具体要求：连接 smtp 服务器->发命令"HELO <host_name>" ->发命令"AUTH LOGIN", 然后服务器会以 base64 编码后的形式提示输入用户名->以 base64 编码后的形式输入用户名, 如果用户名合法，服务器提示输入口令形式为“334 *****”->以 base64 编码后的形式输入口令。如果检验正确，服务器会返回"235 Authentication successful"; 编辑电子邮件（注意区分 SMTP 协议格式与邮件格式），利用 SMTP 协议进行收发电子邮件；观察并分析收发过程及协议信息。编码软件见 Centri64.zip。利用 TELNET 进行 POP3 邮件接收。

简单示例：

TELENT 邮件服务器 IP（或域名） 25

HELO LIU

AUTH LOGIN

*****（输入以 base64 编码后用户名）

*****（输入以 base64 编码后密码）

MAIL FROM:<发送方的邮件地址>

RCPT TO:<接收方的邮件地址>

DATA

FROM:发送方邮箱

TO:接收方邮箱

SUBJECT:LIU

liu like Philatelic!

.(这个点与回车，必须的)

QUIT

实验内容三：熟练掌握抓包软件 Wireshark

要深入理解网络协议，需要观察它们的工作并使用它们，即观察两个协议实体之间交换的报文序列，探究协议操作的细节，使协议实体执行某些动作，观察这些动作及其影响。这些任务可以在仿真环境下或在如因特网这样的真实网络环境中完成。

观察在正在运行协议实体间交换报文的基本工具被称为分组嗅探器（packet sniffer）。一个分组嗅探器俘获（嗅探）由你的计算机发送和接收的报文。一般情况下，分组嗅探器将存储和显示出被俘获报文的各协议字段的内容。图 1 显示了一个分组嗅探器的结构。

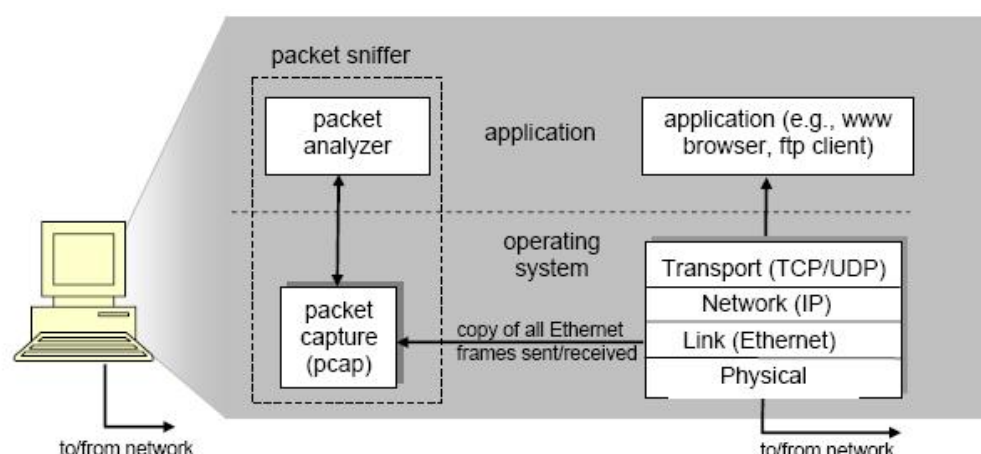


Figure 1: Packet sniffer structure

图 1 右边是计算机上正常运行的协议（在这里是因特网协议）和应用程序（如：web 浏览器和 ftp 客户端）。分组嗅探器（虚线框中的部分）是附加计算机普通软件上的，主要有两部分组成。分组俘获库（packet capture library）接收计算机发送和接收的每一个链路层帧的拷贝。高层协议（如：HTTP、FTP、TCP、UDP、DNS、IP 等）交换的报文都被封装在链路层帧中，并沿着物理媒体（如以太网的电缆）传输。图 1 假设所使用的物理媒体是以太网，并且上层协议的报文最终封装在以太网帧中。

分组嗅探器的第二个组成部分是分组分析器。分组分析器用来显示协议报文所有字段的内容。为此，分组分析器必须能够理解协议所交换的所有报文的结构。例如：我们要显示图 1 中 HTTP 协议所交换的报文的各个字段。分组分析器理解以太网帧格式，能够识别包含在帧中的 IP 数据报。分组分析器也要理解 IP 数据报的格式，并能从 IP 数据报中提取出 TCP 报文段。然后，它需要理解 TCP 报文段，并能够从中提取出 HTTP 消息。最后，它需要理解 HTTP 消息。

Wireshark 是一种可以运行在 Windows, UNIX, Linux 等操作系统上的分组分析器。Wireshark 是免费的，可以从 [Http://www.Wireshark.com](http://www.Wireshark.com) 得到。

运行 Wireshark 程序时，其图形用户界面如图 2 所示。最初，各窗口中并无数据显示。

Wireshark 的界面主要有五个组成部分：

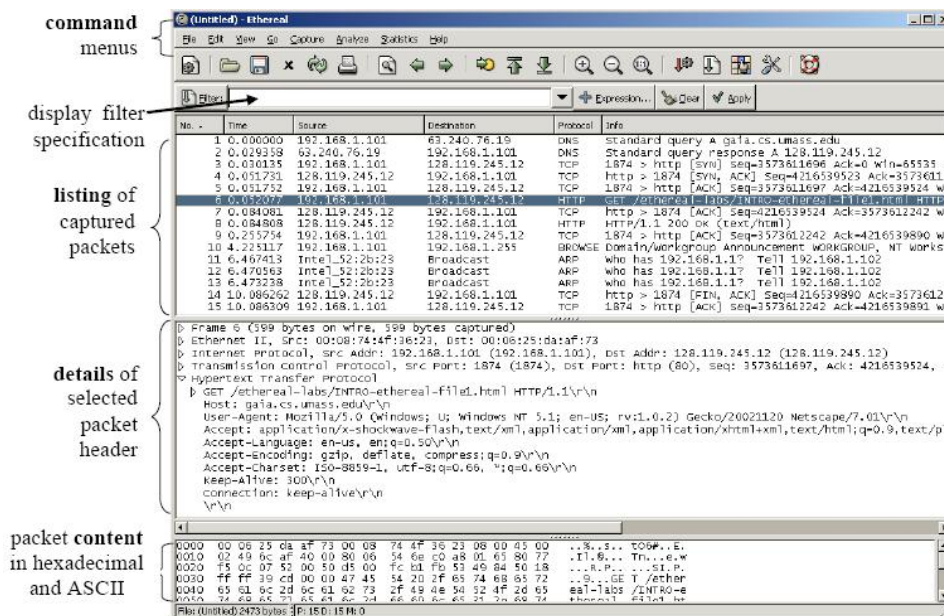


Figure 2: Ethereal Graphical User Interface

命令菜单（command menus）：命令菜单位于窗口的最顶部，是标准的下拉式菜单。最常用菜单命令有两个：File、Capture。File 菜单允许你保存俘获的分组数据或打开一个已被保存的俘获分组数据文件或退出 Wireshark 程序。Capture 菜单允许你开始俘获分组。

俘获分组列表（listing of captured packets）：按行显示已被俘获的分组内容，其中包括：Wireshark 赋予的分组序号、俘获时间、分组的源地址和目的地址、协议类型、分组中所包含的协议说明信息。单击某一列的列名，可以使分组列表按指定列进行排序。在该列表中，所显示的协议类型是发送或接收分组的最高层协议的类型。

分组首部明细（details of selected packet header）：显示俘获分组列表窗口中被选中分组的头部详细信息。包括：与以太网帧有关的信息，与包含在该分组中的 IP 数据报有关的信息。单击以太网帧或 IP 数据报所在行左边的向右或向下的箭头可以展开或最小化相关信息。另外，如果利用 TCP 或 UDP 承载分组，Wireshark 也会显示 TCP 或 UDP 协议头部信息。最后，分组最高层协议的头部字段也会显示在此窗口中。

分组内容窗口（packet content）：以 ASCII 码和十六进制两种格式显示被俘获帧的完整内容。

显示筛选规则（display filter specification）：在该字段中，可以填写协议的名称或其他信息，根据此内容可以对分组列表窗口中的分组进行过滤。

实验步骤

- (1)启动主机上的 web 浏览器。
- (2)启动 Wireshark。会看到如图 2 所示的窗口，只是窗口中没有任何分组列表。
- (3)开始分组俘获：选择“capture”下拉菜单中的“Start”命令，会出现如图 3 所示的“Wireshark: Capture Options”窗口，可以设置分组俘获的选项。

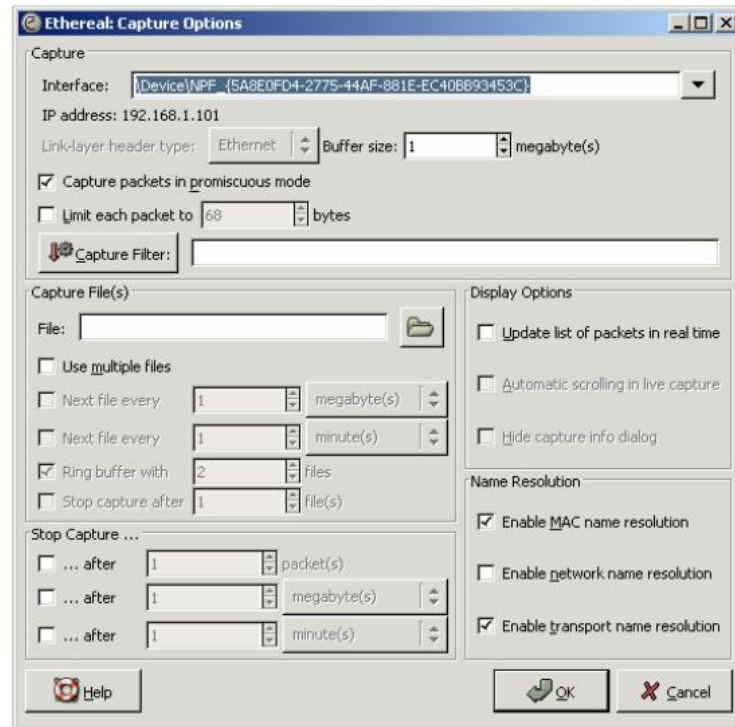


Figure 3: Ethereal Capture Options Window

(4)在实验中，可以使用窗口中显示的默认值。在“Wireshark: Capture Options”窗口的最上面有一个“interface”下拉菜单，其中显示计算机所具有的网络接口（即网卡）。当计算机具有多个活动网卡时，需要选择其中一个用来发送或接收分组的网络接口（如某个有线接口）。随后，单击“ok”开始进行分组俘获，所有由选定网卡发送和接收的分组都将被俘获。

(5)开始分组俘获后，会出现如图 4 所示的分组俘获统计窗口。该窗口统计显示各类已俘获分组的数量。在该窗口中有一个“stop”按钮，可以停止分组的俘获。但此时最好不要停止俘获分组。



Figure 4: Ethereal Packet Capture Window

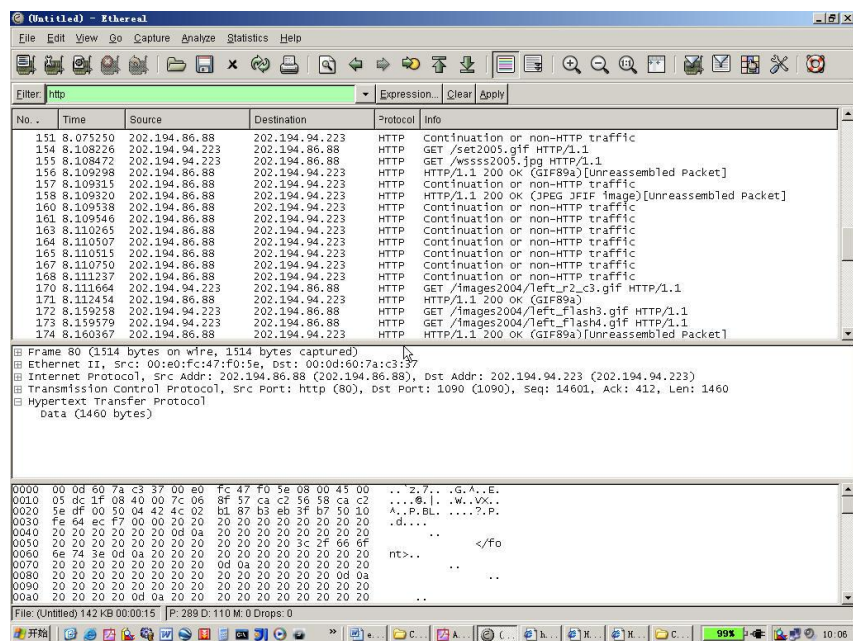
(6)在运行分组俘获的同时，在浏览器地址栏中输入某网页的 URL，如：

<http://www.hitwh.edu.cn> 为显示该网页，浏览器需要连接 www.hitwh.edu.cn 的服务器，并与之交换 HTTP 报文，以下载该网页。包含这些 HTTP 报文的以太网帧将被 Wireshark 俘获。

(7)当完整的页面下载完成后，单击 Wireshark 俘获窗口中的 stop 按钮，停止分组俘获。此时，分组俘获窗口关闭。Wireshark 主窗口显示已俘获的你的计算机与其他网络实体交换的所有协议报文，其中一部分就是与 www.hitwh.edu.cn 服务器交换的 HTTP 报文。此时主窗口与图 2 相似。

(8)在显示筛选规则中输入“http”，单击“apply”，分组列表窗口将只显示 HTTP 协议报文。

(9)选择分组列表窗口中的第一条 http 报文。它应该是你的计算机发向 www.hitwh.edu.cn 服务器的 HTTP GET 报文。当你选择该报文后，以太网帧、IP 数据报、TCP 报文段、以及 HTTP 报文首部信息都将显示在分组首部子窗口中。单击分组首部详细信息子窗口中向右和向下箭头，可以最小化帧、以太网、IP、TCP 信息显示量，可以最大化 HTTP 协议相关信息的显示量。其结果与图 5 相似。



(10)退出 Wireshark。

【实验方式】由实验指导教师讲解、演示；分组讨论与实践。

【实验地点】学院实验室。

【实验报告】在实验报告中写出 DNS 层次查询的实现过程、利用 SMTP 协议实现接发邮件的过程、回答捕包实验中所提问题。分析并总结实验中遇到的问题，写出实验体会。

(1)列出在第 7 步中分组列表窗口所显示的所有协议类型。

(2)从发出 HTTP GET 报文到接收到 HTTP OK 响应报文共需要多长时间？（在默认的情况下，分组列表窗口中 Time 列的值是从 Wireshark 开始追踪到分组被俘获的总的时间数，以秒为单位。若要按 time-of-day 格式显示 Time 列的值，需选择 View 下拉菜单，再选择 Time Display Format，然后选择 Time-of-day。）

(3)你主机的 IP 地址是什么？你所访问的主页所在服务器的 IP 地址是什么？

(4)写出两个第 9 步所显示的 HTTP 报文头部行信息。

实验内容四：利用 Wireshark 分别对 TCP 套接字的实现及 UDP 套接字的实现捕包分析

安装 java 虚拟机。熟悉捕包软件的使用，进行相应的捕包。具体程序上课时拷贝。

TCP 客户端套接字程序

```
import java.io.*;

import java.net.*;

class TCPClient{

    public static void main(String argv[]) throws Exception

    {

        String sentence;

        String modifiedSentence;

        BufferedReader inFromUser =

            new BufferedReader(

                new InputStreamReader(System.in));

        Socket ClientSocket = new Socket("222.194.1.36",6789);

        DataOutputStream outToServer =

            new DataOutputStream(

                ClientSocket.getOutputStream());

        BufferedReader inFromServer =

            new BufferedReader(new InputStreamReader(

                ClientSocket.getInputStream()));

        sentence =inFromUser.readLine();

        outToServer.writeBytes(sentence + '\n');

        modifiedSentence = inFromServer.readLine();

        System.out.println("FROM SERVER:"+

                               modifiedSentence);

        ClientSocket.close();

    }

}
```

```
}
```

TCP 服务器端套接字程序

```
import java.io.*;

import java.net.*;

class TCPServer{

    public static void main(String argv[]) throws Exception

    {

        String ClientSentence;

        String capitalizedSentence;

        ServerSocket welcomeSocket = new ServerSocket(6789);

        while(true){

            Socket connectionSocket = welcomeSocket.accept();

            BufferedReader inFromClient =

                new BufferedReader(new InputStreamReader(

                    connectionSocket.getInputStream()));

            DataOutputStream outToClient =

                new DataOutputStream(

                    connectionSocket.getOutputStream());

            ClientSentence = inFromClient.readLine();

            capitalizedSentence =

                ClientSentence.toUpperCase() + '\n';

            outToClient.writeBytes(capitalizedSentence);

        }

    }

}
```

UDP 客户端套接字程序

```
import java.io.*;

import java.net.*;

class UDPClient {
```



```

public static void main(String args[]) throws Exception
{
    BufferedReader inFromUser =
        new BufferedReader(new InputStreamReader(System.in));

    DatagramSocket clientSocket = new DatagramSocket();

    InetAddress IPAddress = InetAddress.getByName("222.194.1.36");

    byte[] sendData = new byte[1024];
byte[] receiveData = new byte[1024];

    String sentence = inFromUser.readLine();

    sendData = sentence.getBytes();

    DatagramPacket sendPacket =
        new DatagramPacket(sendData, sendData.length,
        IPAddress,9876);

    clientSocket.send(sendPacket);

    DatagramPacket receivePacket =
        new DatagramPacket(receiveData, receiveData.length);

    clientSocket.receive(receivePacket);

    String modifiedSentence=
        new String(receivePacket.getData());

    System.out.println("FROM SERVER:" + modifiedSentence);

    clientSocket.close();
}
}

```

UDP 服务器端套接字程序

```

import java.io.*;

import java.net.*;

class UDPServer {

    public static void main(String args[]) throws Exception
    {

        DatagramSocket serverSocket = new DatagramSocket(9876);

```



```

byte[] receiveData = new byte[1024];

byte[] sendData = new byte[1024];

while(true)
{
    DatagramPacket receivePacket =
        new DatagramPacket(receiveData, receiveData.length);
    serverSocket.receive(receivePacket);

    String sentence = new String(receivePacket.getData());

    InetAddress IPAddress = receivePacket.getAddress();

    int port = receivePacket.getPort();

    String capitalizedSentence = sentence.toUpperCase();

    sendData = capitalizedSentence.getBytes();

    DatagramPacket sendPacket =
        new DatagramPacket(sendData, sendData.length,
        IPAddress,port);

    serverSocket.send(sendPacket);
}
}
}

```

实验内容五：利用 Wireshark 分析协议 HTTP、FTP 和 DNS

一、实验目的

- 1、分析 HTTP 协议
- 2、分析 DNS 协议

二、实验环境

与因特网连接的计算机网络系统；主机操作系统为 windows；Wireshark、IE 等软件。

三、实验步骤

1、HTTP GET/response 交互

首先通过下载一个非常简单的 HTML 文件（该文件非常短，并且不嵌入任何对象）。

- (1) 启动 Web browser。
- (2) 启动 Wireshark 分组嗅探器。在窗口的显示过滤说明处输入“http”，分组列表子窗口中将只显示所俘获到的 HTTP 报文。
- (3) 一分钟以后，开始 Wireshark 分组俘获。
- (4) 在打开的 Web browser 窗口中输入一下地址（浏览器中将显示一个只有一行文字的非常简单的 HTML 文件）：

<http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html>

- (5) 停止分组俘获。

窗口如图 1 所示。根据俘获窗口内容，回答“四、实验报告内容”中的 1-6 题。

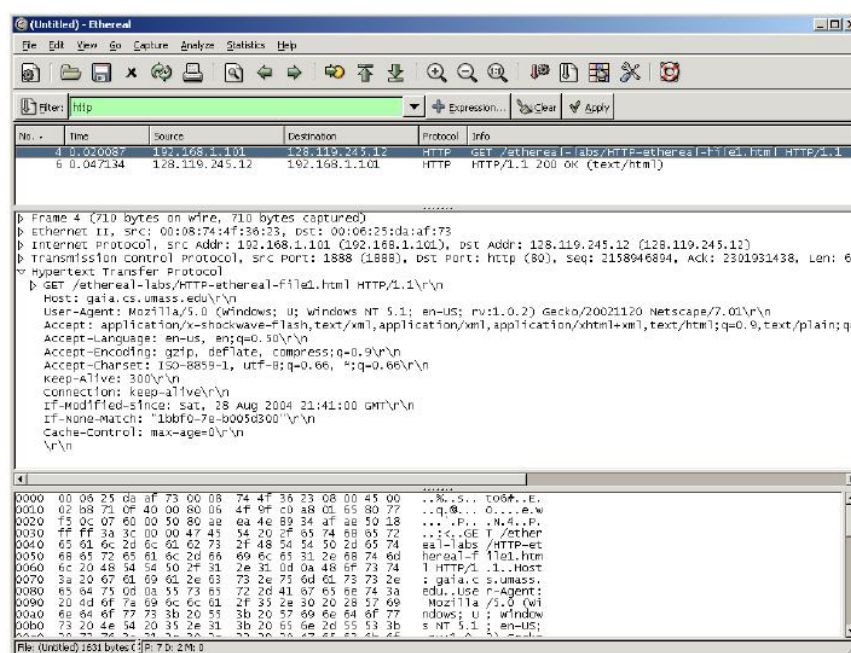


图 1 分组俘获窗口

2、HTTP 条件 GET/response 交互

- (1) 启动浏览器，清空浏览器的缓存（在浏览器中，选择“工具”菜单中的“Internet 选项”命令，在出现的对话框中，选择“删除文件”）。
- (2) 启动 Wireshark 分组俘获器。开始 Wireshark 分组俘获。
- (3) 在浏览器的地址栏中输入以下 URL：
<http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file2.html>，你的浏览器中将显示一个具有五行的非常简单的 HTML 文件。
- (4) 在你的浏览器中重新输入相同的 URL 或单击浏览器中的“刷新”按钮。
- (5) 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。

根据操作回答“四、实验报告内容”中的 7-10 题。

3、获取长文件

- (1) 启动浏览器，将浏览器的缓存清空。
- (2) 启动 Wireshark 分组俘获器。开始 Wireshark 分组俘获。
- (3) 在浏览器的地址栏中输入以下 URL：
<http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file3.html>，浏览器将显示一个相当大的美国权力法案。
- (4) 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。

根据操作回答“四、实验报告内容”中的 11-13 题。

4、嵌有对象的 HTML 文档

- (1) 启动浏览器，将浏览器的缓存清空。
- (2) 启动 Wireshark 分组俘获器。开始 Wireshark 分组俘获。
- (3) 在浏览器的地址栏中输入以下 URL：
<http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file4.html>，浏览器将显示一个具有两个图片的短 HTTP 文件
- (4) 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。

根据操作回答“四、实验报告内容”中的 14-15 题。

5、HTTP 认证

- (1) 启动浏览器，将浏览器的缓存清空。
- (2) 启动 Wireshark 分组俘获器。开始 Wireshark 分组俘获。
- (3) 在浏览器的地址栏中输入以下 URL：
http://gaia.cs.umass.edu/wireshark-labs/protected_pages/HTTP-wireshark-file5.html
，浏览器将显示一个 HTTP 文件，输入所需要的用户名和密码(用户名: wireshark-students, 密码: network)。
- (4) 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。

根据操作回答“四、实验报告内容”中的 16-17 题。

6、跟踪 DNS

nslookup 工具允许运行该工具的主机向指定的 DNS 服务器查询某个 DNS 记录。如果没有指明 DNS 服务器，nslookup 将把查询请求发向默认的 DNS 服务器。其命令的一般格式是：

```
nslookup -option1 -option2 host-to-find dns-server
```

ipconfig 命令用来显示你当前的 TCP/IP 信息，包括：你的地址、DNS 服务器的地址、适配器的类型等信息。如果，要显示与主机相关的信息用命令：

```
ipconfig/all
```

如果查看 DNS 缓存中的记录用命令：

```
ipconfig/displaydns
```

要清空 DNS 缓存，用命令：

```
ipconfig /flushdns
```

运行以上命令需要进入 MSDOS 环境。

- (1) 利用 ipconfig 命令清空你的主机上的 DNS 缓存。
- (2) 启动浏览器，将浏览器的缓存清空。
- (3) 启动 Wireshark 分组俘获器，在显示过滤筛选说明处输入
“ip.addr==your_IP_address” (如: ip.addr==10.17.7.23)，过滤器将会删除所有
目的地址和源地址都与指定 IP 地址不同的分组。
- (4) 开始 Wireshark 分组俘获。
- (5) 在浏览器的地址栏中输入: <http://www.ietf.org>
- (6) 停止分组俘获。

根据操作回答“四、实验报告内容”中的 18-24 题。

- (7) 开始 Wireshark 分组俘获。
- (8) 在 www.mit.edu 上进行 nslookup (即执行命令: nslookup www.mit.edu)。
- (9) 停止分组俘获。

根据操作回答“四、实验报告内容”中的 25-28 题。

- (10) 重复上面的实验, 只是将命令替换为: nslookup - type=NS [mit.edu](http://www.mit.edu)

根据操作回答“四、实验报告内容”中的 29-31 题。

- (11) 重复上面的实验, 只是将命令替换为: nslookup www.aiit.or.kr bitsy.mit.edu

根据操作回答“四、实验报告内容”中的 32-34 题。

四、 实验报告内容(全部问题必须实际操作并思考, 可以选做 8 道题)

在实验的基础上, 回答以下问题:

- (1) 你的浏览器运行的是 HTTP1.0, 还是 HTTP1.1? 你所访问的服务器所运行的 HTTP 版本号是多少?
- (2) 你的浏览器向服务器指出它能接收何种语言版本的对象?
- (3) 你的计算机的 IP 地址是多少? 服务器 gaia.cs.umass.edu 的 IP 地址是多少?
- (4) 从服务器向你的浏览器返回的状态代码是多少?
- (5) 你从服务器上所获取的 HTML 文件的最后修改时间是多少?
- (6) 返回到你的浏览器的内容一供多少字节?
- (7) 分析你的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, 是否有一行是: IF-MODIFIED-SINCE?
- (8) 分析服务器响应报文的内容, 服务器是否明确返回了文件的内容? 如何获知?
- (9) 分析你的浏览器向服务器发出的第二个“HTTP GET”请求, 在该请求报文中是否有一行是: IF-MODIFIED-SINCE? 如果有, 在该首部行后面跟着的信息是什么?
- (10) 服务器对第二个 HTTP GET 请求的响应中的 HTTP 状态代码是多少? 服务器是否明确返回了文件的内容? 请解释。
- (11) 你的浏览器一共发出了多少个 HTTP GET 请求?
- (12) 承载这一个 HTTP 响应报文一共需要多少个 data-containing TCP 报文段?
- (13) 与这个 HTTP GET 请求相对应的响应报文的状态代码和状态短语是什么?
- (14) 你的浏览器一共发出了多少个 HTTP GET 请求? 这些请求被发送到的目的地的 IP 地址是多少?
- (15) 浏览器在下载这两个图片时, 是串行下载还是并行下载? 请解释。

- (16) 对于浏览器发出的最初的 HTTP GET 请求, 服务器的响应是什么 (状态代码和状态短语)?
- (17) 当浏览器发出第二个 HTTP GET 请求时, 在 HTTP GET 报文中包含了哪些新的字段?
- (18) 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 UDP 还是基于 TCP 的?
- (19) DNS 查询报文的目的端口号是多少? DNS 查询响应报文的源端口号是多少?
- (20) DNS 查询报文发送的目的地的 IP 地址是多少? 利用 `ipconfig` 命令 (`ipconfig/all`) 决定你主机的本地 DNS 服务器的 IP 地址。这两个地址相同吗?
- (21) 检查 DNS 查询报文, 它是哪一类型的 DNS 查询? 该查询报文中包含 “answers” 吗?
- (22) 检查 DNS 查询响应报文, 其中提供了多少个 “answers”? 每个 answers 包含哪些内容?
- (23) 考虑一下你的主机发送的并发 TCP SYN 分组, SYN 分组的目的 IP 地址是否与在 DNS 查询响应报文中提供的某个 IP 地址相对应?
- (24) 打开的 WEB 页中包含图片, 在获取每一个图片之前, 你的主机发出新的 DNS 查询了吗?
- (25) DNS 查询报文的目的端口号是多少? DNS 查询响应报文的源端口号是多少?
- (26) DNS 查询报文发送的目的地的 IP 地址是多少? 这个地址是你的默认本地 DNS 服务器的地址吗?
- (27) 检查 DNS 查询报文, 它是哪一类型的 DNS 查询? 该查询报文中包含 “answers” 吗?
- (28) 检查 DNS 查询响应报文, 其中提供了多少个 “answers”? 每个 answers 包含哪些内容?
- (29) DNS 查询报文发送的目的地的 IP 地址是多少? 这个地址是你的默认本地 DNS 服务器的地址吗?
- (30) 检查 DNS 查询报文, 它是哪一类型的 DNS 查询? 该查询报文中包含 “answers” 吗?
- (32) DNS 查询报文发送的目的地的 IP 地址是多少? 这个地址是你的默认本地 DNS 服务器的地址吗? 如果不是, 这个 IP 地址相当于什么?
- (33) 检查 DNS 查询报文, 它是哪一类型的 DNS 查询? 该查询报文中包含 “answers” 吗?
- (34) 检查 DNS 查询响应报文, 其中提供了多少个 “answers”? 每个 answers 包含哪些内容?

计算机网络与通信实验报告（一）			
学 号	姓 名	班 级	报告日期
实验内容	网络常用命令的使用及 DNS 层次查询、SMTP 协议分析；利用分组嗅探器（Wireshark）进行应用层协议分析		
实验目的			
实验预备知识			
实验过程描述			
实验结果			

实验当中问题 及解决方法	1、 2、			
成绩（教师打分）	优秀	良好	及格	不及格