# [使用wireshark分析TCP/IP协议中TCP包头的格式](http://blog.csdn.net/ningxuezhu/article/details/39892091)

1. [摘要](http://blog.csdn.net/ningxuezhu/article/details/39892091/#t0)
2. [一概述](http://blog.csdn.net/ningxuezhu/article/details/39892091/#t1)
3. [二TCP报文格式](http://blog.csdn.net/ningxuezhu/article/details/39892091/#t2)
4. [三实例解析](http://blog.csdn.net/ningxuezhu/article/details/39892091/#t3)

**摘要：**

本文简单介绍了TCP面向连接理论知识，详细讲述了TCP报文各个字段含义，并从Wireshark俘获分组中选取TCP连接建立相关报文段进行分析。

**一、概述**

TCP是面向连接的可靠传输协议，两个进程互发数据之前需要建立连接，这里的连接只不过是端系统中分配的一些缓存和状态变量，中间的分组交换机不维护任何连接状态信息。连接建立整个过程如下(即三次握手协议)：

首先，客户机发送一个特殊的TCP报文段；

其次，服务器用另一个特殊的TCP报文段来响应；

最后，客户机再用第三个特殊报文段作为响应。

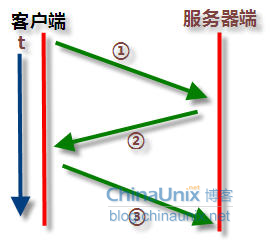


图1 三次握手协议示意图[1]

**二、TCP报文格式**

**2.1 概述**

为了提供可靠的数据传输，TCP报文首部字段有较多的字段，TCP报文格式如下图：

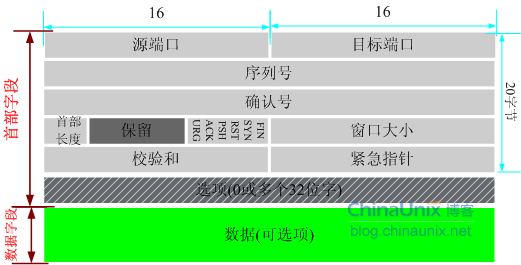


图2 TCP报文格式

**源和目标端口**

用于多路复用/多路分解来自或送至上层应用的数据，可以这样理解，端口用来标识同一台计算机的不同进程。

**序列号和确认号**

这两个字段是TCP可靠传输服务的关键部分，序列号是该报文段首字节的字节流编号(**TCP把数据看成是有序的字节流**，TCP隐式地对数据流的每个字节进行编号)。这样理解可能更直观，当报文被分解成多个报文段时，序列号就是报文段首字节在整个报文的偏移量。**确定号指定下一个期待的字节**。TCP是全双工的，假设从主机A接收到主机B的数据，则主机A填充进报文段的确认号是主机A期望从主机B收到的下一个字节序号。还没理清这两者的关系？见下图(三次握手)：

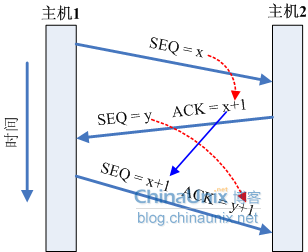


图3 正常情况下TCP连接建立过程

**首部长度(4位)**

因为选项是不定长的，这就需要标识整个首部字段的长度(单位是32位字)，即5+选项个数。4位，单位是32位字(双字节)，所以首部最长是15\*4=60字节，即选项最长是40字节(10个选项)。

**标志**

**URG**

指示报文段里存在着被发送方的上层实体标记为”紧急”数据，当URG=1时，其后的紧急指针指示紧急数据在当前数据段中的位置(相对于当前序列号的字节偏移量)，TCP接收方必须通知上层实体。

**ACK**

当ACK=0时，表示该数据段不包含确认信息，当ACK=1时，表示该报文段包括一个对已被成功接收报文段的确认。

**PSH**

当PSH=1时，接收方在收到数据后立即将数据交给上层，而不是直到整个缓冲区满。

**RST**

用于重置一个已经混乱的连接(如主崩溃)，也可用于拒绝一个无效的数据段或者拒绝一个连接请求。一般而言，如果你得到的数据段被设置了RST位，那说明你这一端有问题了。

**SYN**

用于建立连接过程，在连接请求中，**SYN=1和ACK=0**表示该数据段没有使用**捎带**的确认域，而连接应答捎带一个确认，即**SYN=1和ACK=1**。

注：捎带是指对客户机到服务器数据的**确认**被装载在一个承载服务器到客户机的数据报文段中。

**FIN**

用于释放一个连接，表示发送方已经没有数据要传输了。此时，接收方可能继续接收数据，好在SYN和FIN数据段都有序列号，从而保证了这两种数据段以正确顺序被处理。

**窗口大小**

用于流控制(确保连接的任何一方都不会过快地发送过量的分组而淹没另一方)，窗口大小指定了从**被确认的字节**算起可以发送多少个字节。

**校验和**

提供了额外可靠性，在计算检验和的时候，TCP的Checksum域设为0，如果数据域的字节数为奇数，则数据域填补一个额外的0字节。校验和**[算法](http://lib.csdn.net/base/datastructure" \t "_blank" \o "算法与数据结构知识库)**：将所有的16位字按1的补码形式累加起来，取累加结果的补码。因此，当接收方执行同样计算时(包括Checksum域)，结果应该是0。

**紧急指针**

参考标志字段的URG位。

**选项**

选项部分是为了适合复杂网络环境和更好地服务于应用层设计的。TCP选项最长是40字节。详情见2.2。

**数据**

无任何数据的TCP段也是合法的，通常用于确认和控制信息。

**2.2 选项字段[2]**

TCP选项部分很好出现在已经建立连接的会话中，只要出现在TCP连接建立阶段，即三次握手。TCP选项部分实际运用有以下几种：

**(1)最大报文传输段(MMS, Maximum Segment Size)**

用于发送发与接收方协商**最大报文段长度**(仅仅是净荷数据，不包括TCP首部字段)。TCP在三次握手中，每一方都会通告期望收到的MSS(MSS只出现在SYN数据包中)，如果一方不接受另一方的MSS值，则使用默认的**536**字节净荷数据，即主机能够接受20+536字节的TCP报文段。

**(2)窗口扩大选项(Window scaling)**

TCP报文的窗口大小字段占16位，即最大值是65535，**但随着时延和带宽比较大的通信产生(如卫星通信)，需要更大的窗口满足性能和吞吐率**，这就是窗口扩大选项存在的意义。例子见参考资料[2]。

Windows scaling占3个字节，最后一个字节是**移位值(Shift count)**，即首部的窗口位数16向左移动，如移位值为14，则新的窗口最大值增大到65535\*(2^14)。

窗口扩大选项是在TCP建立之初进行协商，如果已实现了窗口扩大，当不再需要扩大窗口时，发送**移位值=0**就可以恢复到原窗口大小，即65535。

**(3)选择确认选项(SACK, Selective Acknowledgements)**

考虑这样情况，主机A发送报文段12345，主机B收到135且报文无差错，**SACK用来确保只重传缺少的报文段**，而不是重传所有报文段。

SACK选项需要2个功能字节，一个用来指明使用SACK选项(SACK Permission)，另一指明这个选项占多少字节。

那怎么形容丢失的报文段2，说明2的左右边界分别是1、3。TCP的数据报文是有字块边界的，而这种边界是由序列号表示的。

最多能指明多少个字节块的边界信息呢？答案是4个。这是因为选项字段最大是40字节，去除2个功能字节，序列号是32位即4字节，并且需要左右边界，所以(40-2)/8 = 4。

**(4)时间戳选项(timestamps)**

时间戳选项用来计算往返时间RTT，发送方在发送报文段时把当前时钟的时间值放入时间戳字段，接收方将该时间戳字段的值复制到确认报文中，当接收方收到确认报文，对比确认报文的时间戳(等于发送方发送报文段的时间戳)和现在的时钟，即可算出RTT。

时间戳选项还可用于防止回绕序号PAWS。序列号只有32位，每2^32个序列号就会回绕(想想环形队列)，采用时间戳选项很容易区分相同序列号的报文段。

**(5)NOP(NO-Operation)**

TCP的头部必须是4字节的倍数，而大多数选项不是4字节倍数，不足的用NOP填充。除此之外，NOP也用于分割不同的选项数据，如窗口扩大选项和SACK之间使用NOP隔离(下面的实例将看到这一点)。

**三、实例解析**

**3.1 概述**

还是以访问百度首页为例，首先用DNS协议将URL解析成IP地址，接着在客户机和服务器间建立TCP连接，用Wireshark俘获的分组如下图：

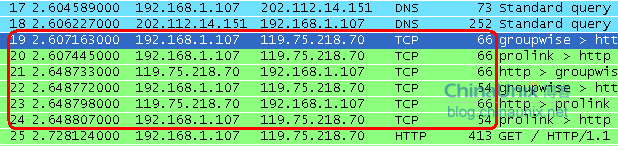


图4 Wireshark俘获建立TCP连接分组

你一看会觉得有些奇怪，理论上应该是3个分组的，怎么有6个分组？先不急，先把这6个报文收发示意图作出来(结合时间和报文含义)，如下：

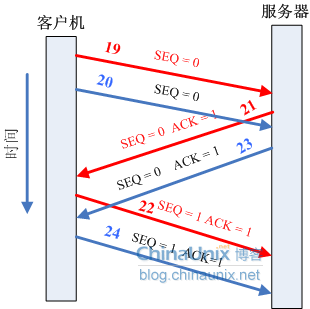


图5 TCP连接建立实例

从图可知，连接建立伊始，客户机发了两个报文段，这也许是为了更快建立连接(假设有个请求报文段丢失，也不至于要等一段时间，重发报文)。接下来，以19、21、22(上图红色线条所示)分析TCP连接建立过程。

**3.1 第一次握手19**

Wireshark俘获TCP连接第一次握手的报文段如下：

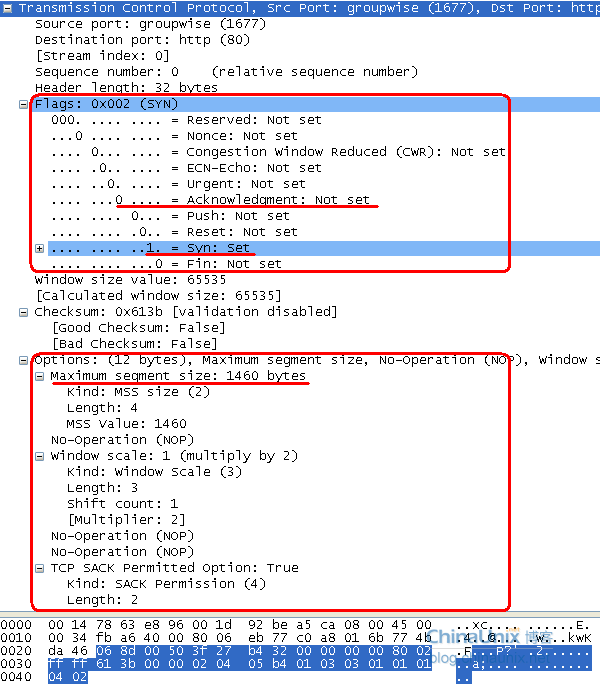


图6 TCP连接第一次握手实例

这里主要挑几个字段分析：

标志字段，SYN=1、ACK=0表示该数据段没有使用**捎带**的确认域。

最大报文段长度(MMS)1460是怎么来的，链路层的以太网物理特性决定数据帧长度为1500(即MTU，最大传输单元)，1460=1500-20(IP首部长度)-20(TCP首部长度)。不要被该报文首部长度32字节所迷惑，这只是建立连接过程。MSS与MTU关系见下图[2]：

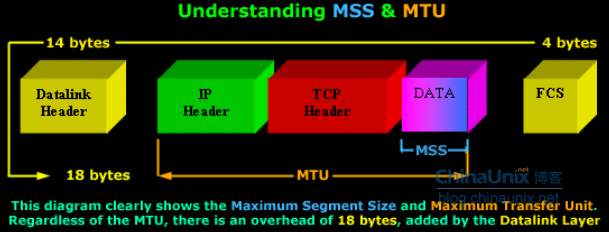


图7 MSS与MTU关系

NOP字段，可以作为不足4倍数字节填充，也可作为选项间分隔，该报文段出现了3个NOP，具体功能见下图：

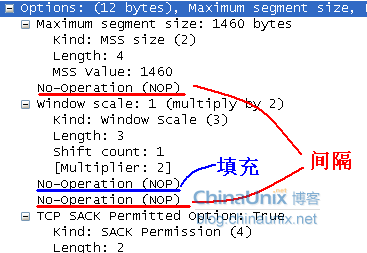


图8 TCP报文NOP字段

**3.3 第二次握手21**

服务器响应客户端TCP报文段，此时确认号为1了，SYN=1、ACK=1表明连接应答捎带一个确认，Wireshark俘获分组如下：

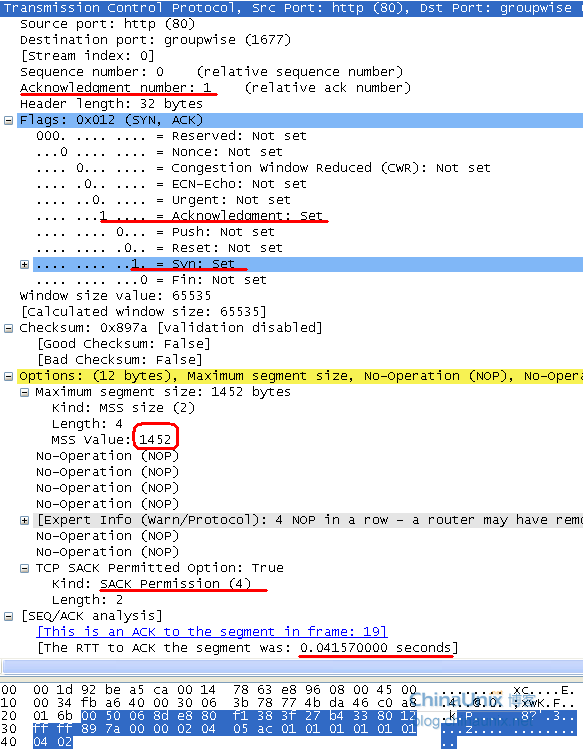


图9 TCP连接第二次握手实例

为什么MSS是1452而不是1460?这是因为使用PPPoE(Point-to-Point over Ethernet，可以使以太网的主机通过一个简单的桥接设备连到一个无端的接入集中器上[3])拨号上网，PPoP首部是8个字节，所以PPPoE的MTU是1492，MSS也就为1492-40=1452。

那么，TCP连接建立后数据传输的MSS是多少呢，1460 or 1452 or 536 ？我的理解是默认值536，这样理解对吗？求指点！

**3.4 第三次握手22**

客户机再次服务器的报文段，此时序列号和确认号都为1，没有选项字段，Wireshark俘获的分组信息如下：

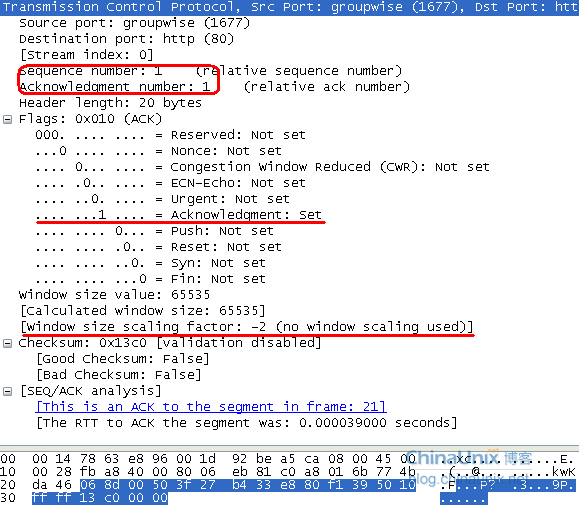


图10 TCP连接第三次握手实例

值得注意的，因为窗口扩展大小协商未果，所以就不扩大窗口了，即窗口大小最大为65535。

如此，TCP连接建立:-)

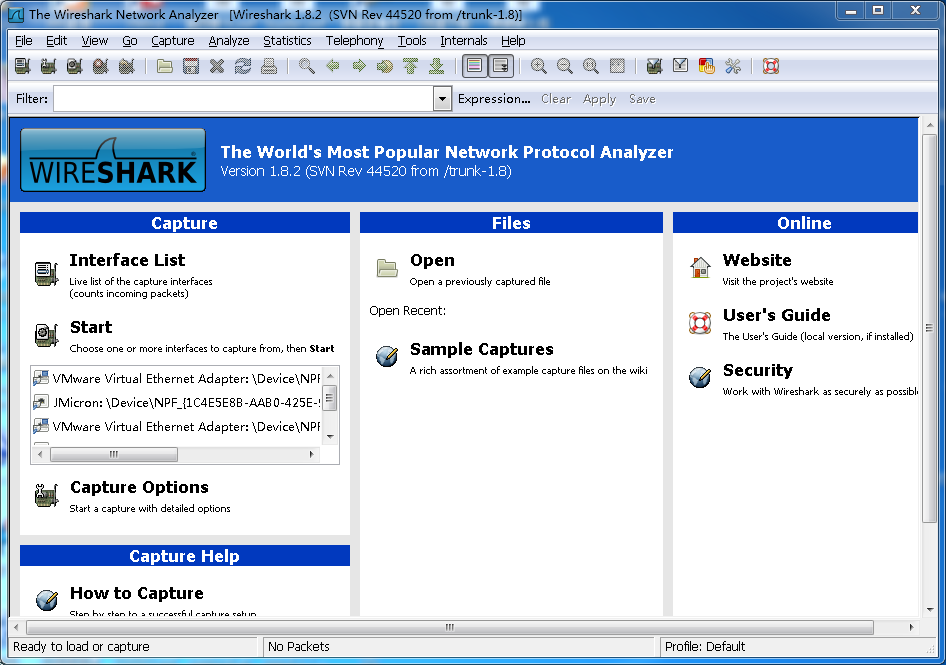
# wireshark抓包详细图文教程

wireshark是非常流行的网络封包分析软件，功能十分强大。可以截取各种网络封包，显示网络封包的详细信息。使用wireshark的人必须了解网络协议，否则就看不懂wireshark了。  
为了安全考虑，wireshark只能查看封包，而不能修改封包的内容，或者发送封包。

wireshark能获取HTTP，也能获取HTTPS，但是不能解密HTTPS，所以wireshark看不懂HTTPS中的内容，总结，如果是处理HTTP,HTTPS 还是用Fiddler,**其他协议比如TCP,UDP 就用wireshark.**

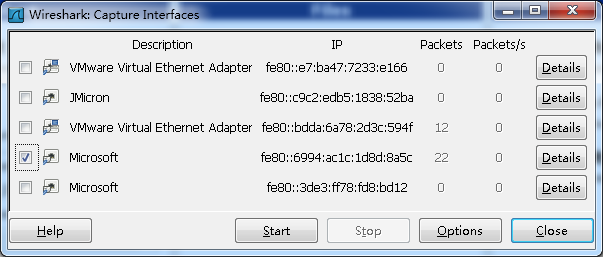
**wireshark 开始抓包**

开始界面

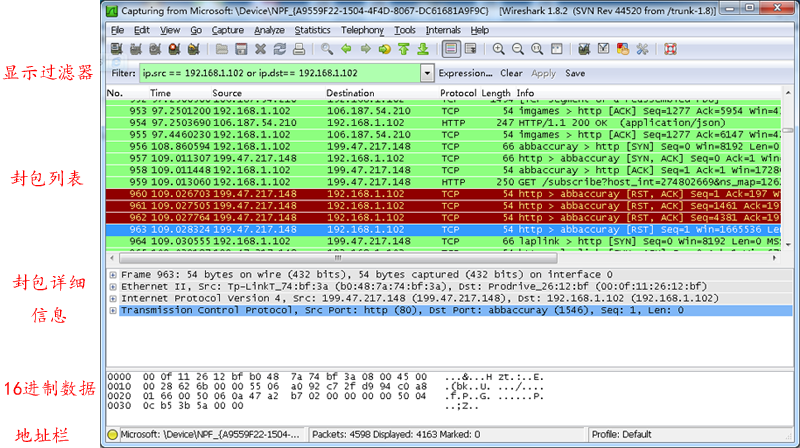


**wireshark是捕获机器上的某一块网卡的网络包**，当你的机器上有多块网卡的时候，你需要选择一个网卡。

点击Caputre->Interfaces.. 出现下面对话框，选择正确的网卡。然后点击"Start"按钮, 开始抓包



Wireshark 窗口介绍



**WireShark 主要分为这几个界面**

1. Display Filter(显示过滤器)，  用于过滤

2. Packet List Pane(封包列表)， 显示捕获到的封包， 有源地址和目标地址，端口号。 颜色不同，代表

3. Packet Details Pane(封包详细信息), 显示封包中的字段

4. Dissector Pane(16进制数据)

5. Miscellanous(地址栏，杂项)

**第 2 页 Wireshark 显示过滤**

**http://www.9upk.com/article/UploadPic/2013-5/20135220328859.png**

使用过滤是非常重要的， 初学者使用wireshark时，将会得到大量的冗余信息，在几千甚至几万条记录中，以至于很难找到自己需要的部分。搞得晕头转向。

过滤器会帮助我们在大量的数据中迅速找到我们需要的信息。

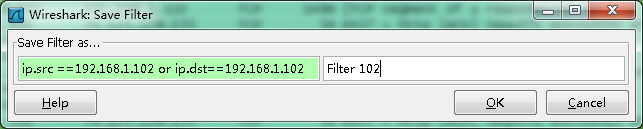
过滤器有两种，

一种是显示过滤器，就是主界面上那个，用来在捕获的记录中找到所需要的记录

一种是捕获过滤器，用来过滤捕获的封包，以免捕获太多的记录。 在Capture -> Capture Filters 中设置

保存过滤

在Filter栏上，填好Filter的表达式后，点击Save按钮， 取个名字。比如"Filter 102",



Filter栏上就多了个"Filter 102" 的按钮。

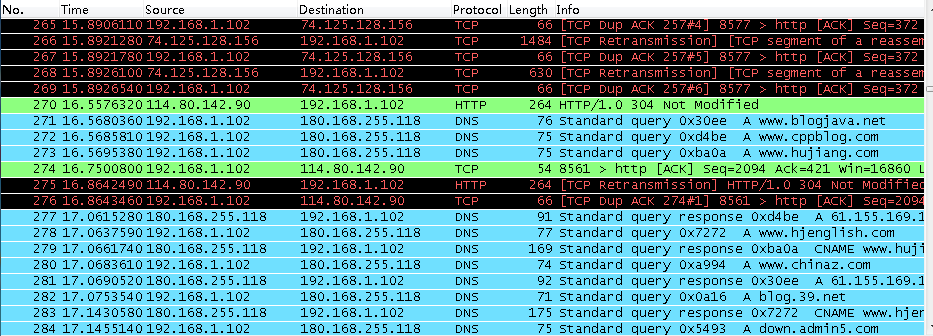
http://www.9upk.com/article/UploadPic/2013-5/20135220328630.png

|  |  |
| --- | --- |
| 过滤表达式 | 用途 |
| http | 只查看HTTP协议的记录 |
| ip.src ==192.168.1.102 or ip.dst==192.168.1.102 | 源地址或者目标地址是192.168.1.102 |
|  |  |
|  |  |

封包列表(Packet List Pane)

封包列表的面板中显示，编号，时间戳，源地址，目标地址，协议，长度，以及封包信息。 你可以看到不同的协议用了不同的颜色显示。

你也可以修改这些显示颜色的规则，  View ->Coloring Rules.



封包详细信息 (Packet Details Pane)

这个面板是我们最重要的，用来查看协议中的每一个字段。

各行信息分别为

Frame:   物理层的数据帧概况

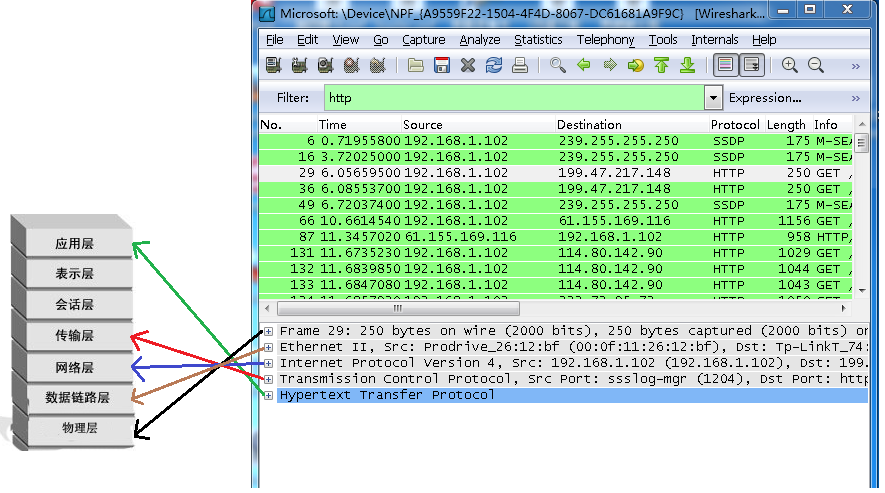
Ethernet II: 数据链路层以太网帧头部信息

Internet Protocol Version 4: 互联网层IP包头部信息

Transmission Control Protocol:  传输层T的数据段头部信息，此处是TCP

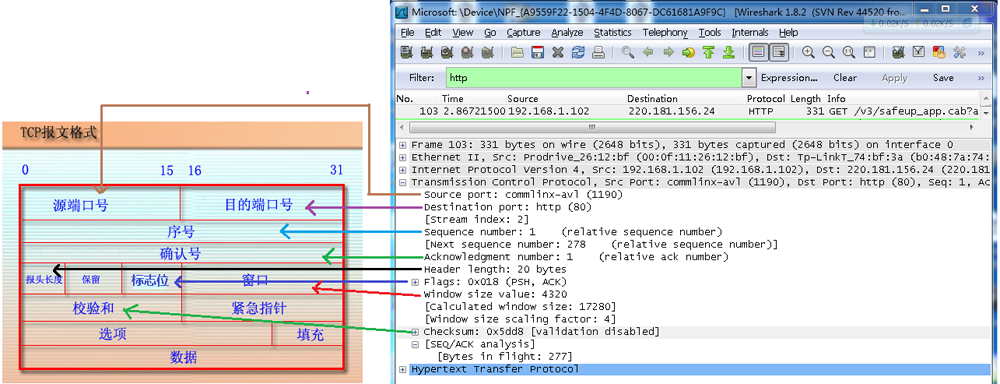
Hypertext Transfer Protocol:  应用层的信息，此处是HTTP协议

**第 3 页 wireshark与对应的OSI七层模型**

****

**TCP包的具体内容**

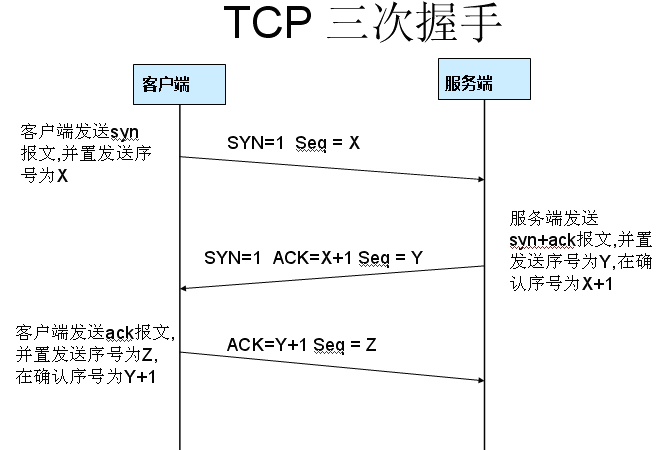
 从下图可以看到wireshark捕获到的TCP包中的每个字段。



**第 4 页 实例分析TCP三次握手过程**

看到这， 基本上对wireshak有了初步了解， 现在我们看一个TCP三次握手的实例

 三次握手过程为

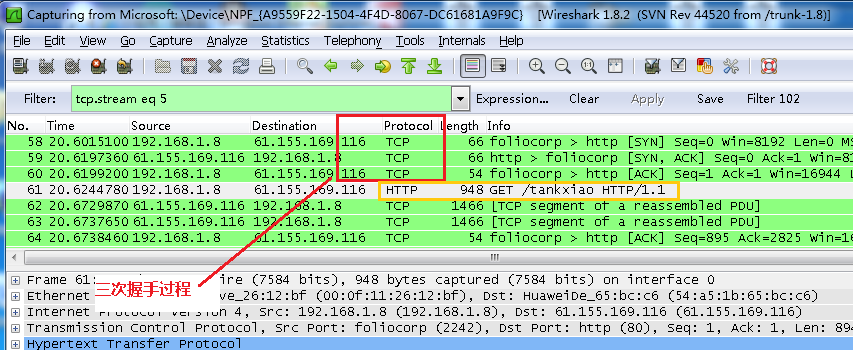


这图我都看过很多遍了， 这次我们用wireshark实际分析下三次握手的过程。

打开wireshark, 打开浏览器输入 [http://www.9upk.com](http://www.9upk.com/)

在wireshark中输入http过滤， 然后选中GET /tankxiao HTTP/1.1的那条记录，右键然后点击"Follow TCP Stream",

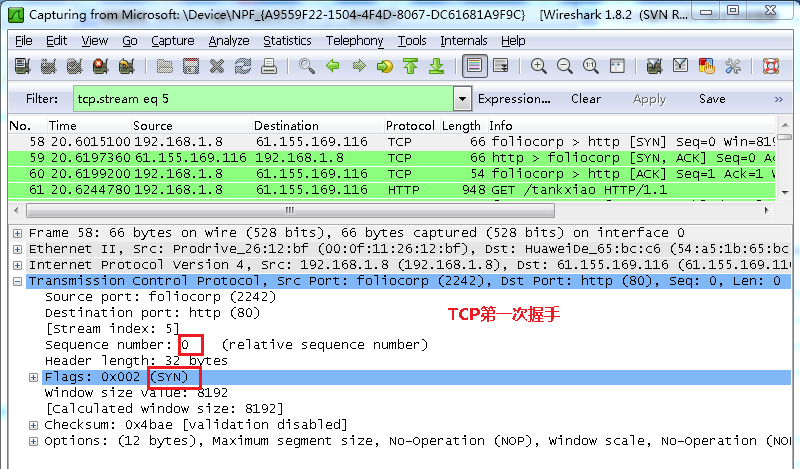
这样做的目的是为了得到与浏览器打开网站相关的数据包，将得到如下图



图中可以看到wireshark截获到了三次握手的三个数据包。第四个包才是HTTP的， 这说明HTTP的确是使用TCP建立连接的。

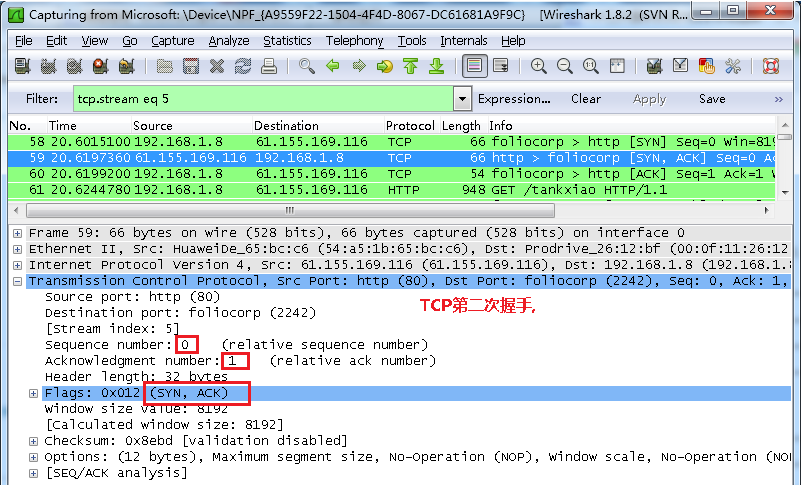
**第一次握手数据包**

客户端发送一个TCP，标志位为SYN，序列号为0， 代表客户端请求建立连接。 如下图



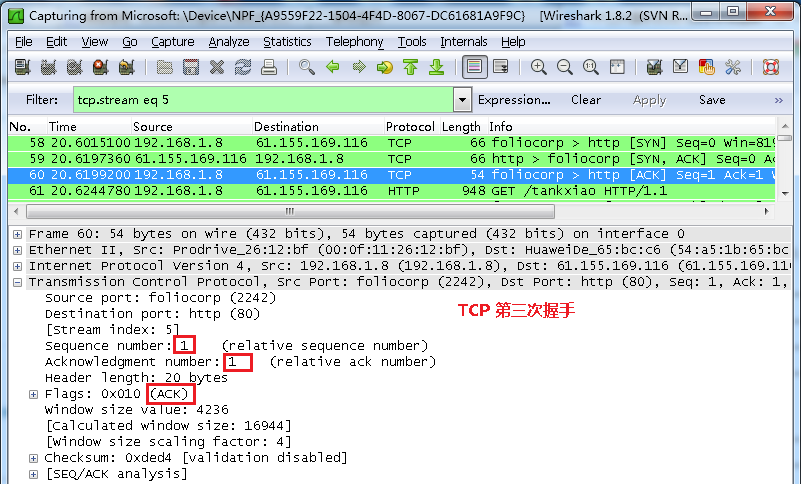
**第二次握手的数据包**

服务器发回确认包, 标志位为 SYN,ACK. 将确认序号(Acknowledgement Number)设置为客户的I S N加1以.即0+1=1, 如下图



**第三次握手的数据包**

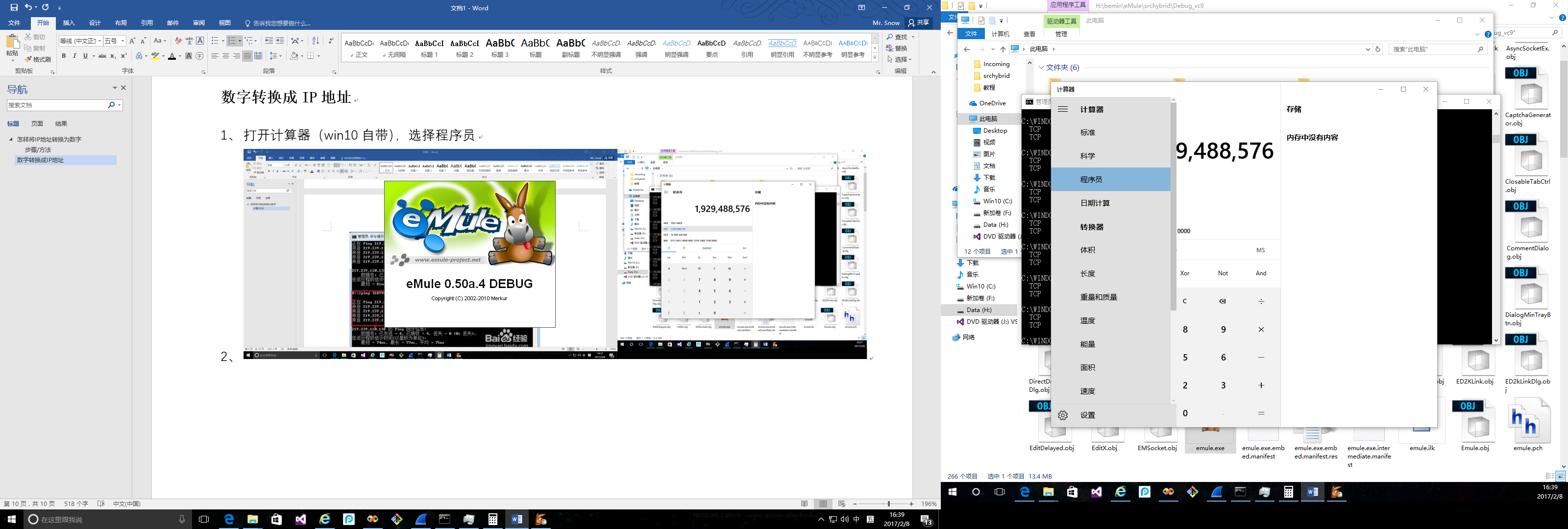
客户端再次发送确认包(ACK) SYN标志位为0,ACK标志位为1.并且把服务器发来ACK的序号字段+1,放在确定字段中发送给对方.并且在数据段放写ISN的+1, 如下图:



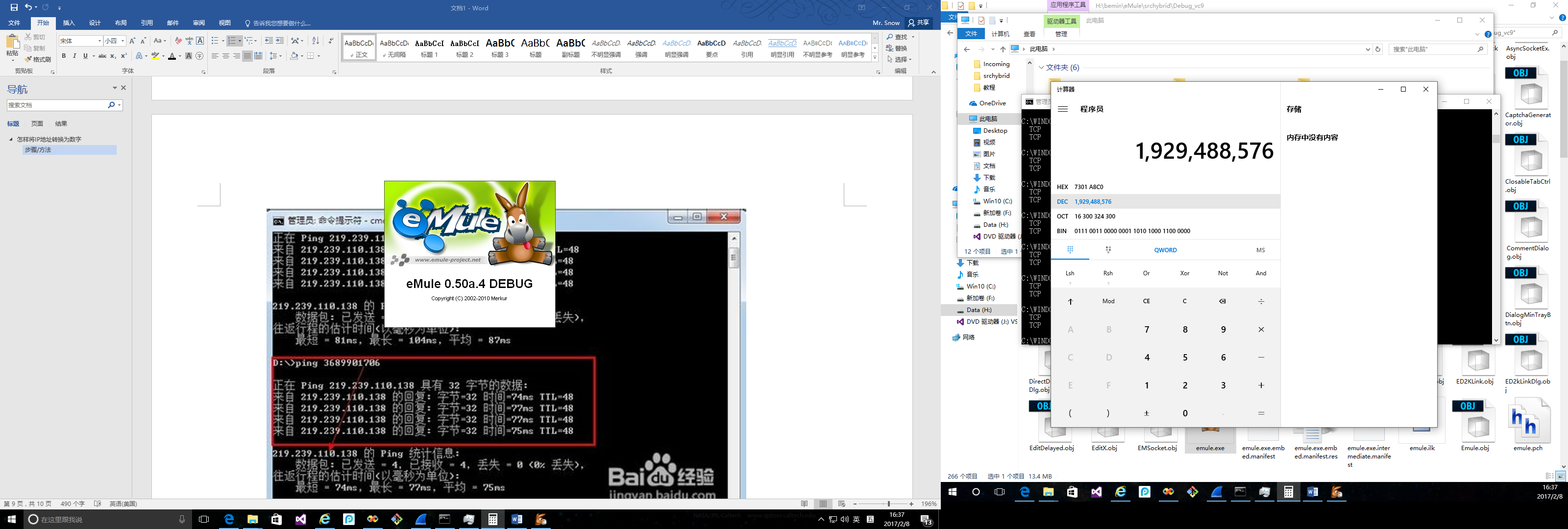
 就这样通过了TCP三次握手，建立了连接

# 数字转换成IP地址

1. 打开计算器（win10自带），选择程序员



2、点击DEC,输入整数1929488576



3、得到对应的十六进制数7301A8C0

4、区分是顺字节序还是逆字节序，本例中的数字是逆字节序，所以应从右往左输入数字

5、点击HEX，输入C0，得到DEC 192，依次输入A8，01，73，分别得到十进制数168，1，115。

6、IP地址就是192.168.1.115

7、如果是顺字节序，则得到的整数应该是**3232235891，HEX值就是C0A80173**

# ICMP

ICMP是（Internet Control Message Protocol）Internet控制[报文](http://baike.baidu.com/view/175122.htm)协议。它是[TCP/IP协议族](http://baike.baidu.com/view/2221037.htm)的一个子协议，用于在IP[主机](http://baike.baidu.com/view/23880.htm)、[路由](http://baike.baidu.com/view/18655.htm)器之间传递控制消息。控制消息是指[网络通](http://baike.baidu.com/view/8079702.htm)不通、[主机](http://baike.baidu.com/view/23880.htm)是否可达、[路由](http://baike.baidu.com/view/18655.htm)是否可用等网络本身的消息。这些控制消息虽然并不传输用户数据，但是对于用户数据的传递起着重要的作用。

中文名

ICMP

外文名

Internet Control Message Protocol

类    型

控制[报文](http://baike.baidu.com/view/175122.htm)协议

协议族

[TCP/IP协议族](http://baike.baidu.com/view/2221037.htm)

归    属

网络层协议

作    用

在主机与路由器之间传递控制信息

## 目录

1. 1 [定义](http://baike.baidu.com/link?url=IwJMNAKRYLDhGzLochvs-AMwtiXoFXqp9E4OoVxVrdqntoVZJOiVvHg_eQ4mGGgLkzqvz2z2hGhWKwsxNo6SC8yU20Fc34XaD4PGP1d0LZXBE6ZTnjURNHiaeGBSQyhf7BQe35TmdMSfxtR-7lLauK#1)
2. 2 [协议内容](http://baike.baidu.com/link?url=IwJMNAKRYLDhGzLochvs-AMwtiXoFXqp9E4OoVxVrdqntoVZJOiVvHg_eQ4mGGgLkzqvz2z2hGhWKwsxNo6SC8yU20Fc34XaD4PGP1d0LZXBE6ZTnjURNHiaeGBSQyhf7BQe35TmdMSfxtR-7lLauK#2)
3. 3 [重要性](http://baike.baidu.com/link?url=IwJMNAKRYLDhGzLochvs-AMwtiXoFXqp9E4OoVxVrdqntoVZJOiVvHg_eQ4mGGgLkzqvz2z2hGhWKwsxNo6SC8yU20Fc34XaD4PGP1d0LZXBE6ZTnjURNHiaeGBSQyhf7BQe35TmdMSfxtR-7lLauK#3)
4. 4 [校验算法](http://baike.baidu.com/link?url=IwJMNAKRYLDhGzLochvs-AMwtiXoFXqp9E4OoVxVrdqntoVZJOiVvHg_eQ4mGGgLkzqvz2z2hGhWKwsxNo6SC8yU20Fc34XaD4PGP1d0LZXBE6ZTnjURNHiaeGBSQyhf7BQe35TmdMSfxtR-7lLauK#4)
5. 5 [抵御攻击](http://baike.baidu.com/link?url=IwJMNAKRYLDhGzLochvs-AMwtiXoFXqp9E4OoVxVrdqntoVZJOiVvHg_eQ4mGGgLkzqvz2z2hGhWKwsxNo6SC8yU20Fc34XaD4PGP1d0LZXBE6ZTnjURNHiaeGBSQyhf7BQe35TmdMSfxtR-7lLauK#5)
6. 6 [防御方法](http://baike.baidu.com/link?url=IwJMNAKRYLDhGzLochvs-AMwtiXoFXqp9E4OoVxVrdqntoVZJOiVvHg_eQ4mGGgLkzqvz2z2hGhWKwsxNo6SC8yU20Fc34XaD4PGP1d0LZXBE6ZTnjURNHiaeGBSQyhf7BQe35TmdMSfxtR-7lLauK#6)
7. 7 [ICMP类型](http://baike.baidu.com/link?url=IwJMNAKRYLDhGzLochvs-AMwtiXoFXqp9E4OoVxVrdqntoVZJOiVvHg_eQ4mGGgLkzqvz2z2hGhWKwsxNo6SC8yU20Fc34XaD4PGP1d0LZXBE6ZTnjURNHiaeGBSQyhf7BQe35TmdMSfxtR-7lLauK#7)

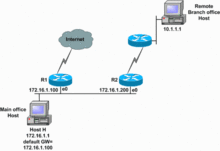
## ICMP定义

ICMP协议是一种面向无连接的协议，用于传输出错报告控制信息。它是一个非常重要的协议，它对于[网络安全](http://baike.baidu.com/view/17495.htm)具有极其重要的意义。[1]

它是[TCP/IP协议](http://baike.baidu.com/view/7649.htm)族的一个子协议，属于网络层协议，主要用于在主机与路由器之间传递控制信息，包括报告错误、交换受限控制和状态信息等。当遇到IP数据无法访问目标、IP[路由器](http://baike.baidu.com/view/1360.htm)无法按当前的传输速率转发[数据包](http://baike.baidu.com/view/25880.htm)等情况时，会自动发送ICMP消息。ICMP报文在IP帧结构的首部协议类型字段（Protocol 8bit)的值=1.

如下图所示，ICMP包有一个8字节长的包头，其中前4个字节是固定的格式，包含8位类型字段，8位代码字段和16位的校验和；后4个字节根据ICMP包的类型而取不同的值。

[](http://baike.baidu.com/pic/ICMP/572452/0/ae51f3deb48f8c54ca50c28039292df5e1fe7f99?fr=lemma&ct=single)ICMP报文格式

[](http://baike.baidu.com/pic/ICMP/572452/0/dbf554ed9529cbccb31cb1d0?fr=lemma&ct=single)ICMP原理

ICMP提供一致易懂的出错报告信息。发送的出错[报文](http://baike.baidu.com/view/175122.htm)返回到发送原数据的设备，因为只有发送设备才是出错报文的逻辑接受者。发送设备随后可根据ICMP报文确定发生错误的类型，并确定如何才能更好地重发失败的数据包。但是ICMP唯一的功能是报告问题而不是纠正错误，纠正错误的任务由发送方完成。

我们在网络中经常会使用到ICMP协议，比如我们经常使用的用于检查网络通不通的[Ping](http://baike.baidu.com/view/709.htm)命令（Linux和Windows中均有），这个“Ping”的过程实际上就是ICMP协议工作的过程。还有其他的[网络命令](http://baike.baidu.com/view/41338.htm)如[跟踪路由](http://baike.baidu.com/view/493712.htm)的Tracert命令也是基于ICMP协议的。

## ICMP协议内容

ICMP的全称是 Internet Control Message Protocol 。从技术角度来

[](http://baike.baidu.com/pic/ICMP/572452/0/f392492cc28fe9f48a1399ce?fr=lemma&ct=single)ICMP常用类型

说，ICMP就是一个“错误侦测与回报机制”，其目的就是让我们能够检测网路的连线状况﹐也能确保连线的准确性﹐其功能主要有：

· 侦测远端主机是否存在。

· 建立及维护路由资料。

· 重导资料传送路径（[ICMP重定向](http://baike.baidu.com/view/5622994.htm)）。

· 资料[流量控制](http://baike.baidu.com/view/190232.htm)。ICMP在沟通之中，主要是透过不同的[类别](http://baike.baidu.com/view/738139.htm)(Type)与[代码](http://baike.baidu.com/view/41.htm" \t "_blank)(Code) 让机器来识别不同的连线状况。常用的[类别](http://baike.baidu.com/view/738139.htm)如下表所列﹕

ICMP 是个非常有用的协议﹐尤其是当我们要对网路连接状况进行判断的时候。

## ICMP重要性

ICMP协议对于[网络安全](http://baike.baidu.com/view/17495.htm)具有极其重要的意义。ICMP协议本身的特

[](http://baike.baidu.com/pic/ICMP/572452/0/aa251d4fd9a2b72eafc3abc8?fr=lemma&ct=single)ICMP常用类型

点决定了它非常容易被用于攻击网络上的[路由器](http://baike.baidu.com/view/1360.htm)和[主机](http://baike.baidu.com/view/23880.htm)。例如，在1999年8月[海信集团](http://baike.baidu.com/view/61422.htm" \t "_blank)“悬赏”50万元人民币测试防火墙的过程中，其防火墙遭受到的ICMP攻击达334050次之多，占整个攻击总数的90%以上！可见，ICMP的重要性绝不可以忽视！

比如，可以利用[操作系统](http://baike.baidu.com/view/880.htm)规定的ICMP数据包最大尺寸不超过64[KB](http://baike.baidu.com/view/170925.htm)这一规定，向[主机](http://baike.baidu.com/view/23880.htm)发起“Ping of Death”（[死亡之Ping](http://baike.baidu.com/view/113328.htm)）攻击。“Ping of Death” 攻击的原理是：如果ICMP数据包的尺寸超过64KB上限时，[主机](http://baike.baidu.com/view/23880.htm)就会出现[内存分配](http://baike.baidu.com/view/549628.htm)错误，导致TCP/IP[堆栈](http://baike.baidu.com/view/93201.htm)崩溃，致使主机死机。（[操作系统](http://baike.baidu.com/view/880.htm)已经取消了发送ICMP数据包的大小的限制，解决了这个漏洞）

此外，向目标[主机](http://baike.baidu.com/view/23880.htm)长时间、连续、大量地发送ICMP数据包，也会最终使系统瘫痪。大量的ICMP数据包会形成“ICMP风暴”，使得目标[主机](http://baike.baidu.com/view/23880.htm)耗费大量的CPU资源处理，疲于奔命。

## ICMP校验算法

以下[代码](http://baike.baidu.com/view/41.htm)在Visual Studio 2008 + Windows 7下调试通过。

lpsz指定要计算的数据包首地址，\_dwSize指定该数据包的长度。

int CalcCheckSum(char\* lpsz,DWORD \_dwSize)

{

int dwSize;

\_\_asm // 嵌入汇编

{

mov ecx,\_dwSize

shr ecx,1

xor ebx,ebx

mov esi,lpsz

read: //所有word相加，保存至EBX寄存器

lodsw

movzx eax,ax

add ebx,eax

loop read

test \_dwSize,1 //校验数据是否是奇数位的

jz calc

lodsb

movzx eax,al

add ebx,eax

calc:

mov eax,ebx //高低位相加

and eax,0ffffh

shr ebx,16

add eax,ebx

not ax

mov dwSize,eax

}

return dwSize;

}

## ICMP抵御攻击

虽然ICMP协议给[黑客](http://baike.baidu.com/view/1960.htm)以可乘之机，但是ICMP攻击也并非无药可医。只要在日常[网络管理](http://baike.baidu.com/view/325702.htm)中未雨绸缪，提前做好准备，就可以有效地避免ICMP攻击造成的损失。

对于“Ping of Death”攻击，可以采取两种方法进行防范：第一种方法是在[路由器](http://baike.baidu.com/view/1360.htm)上对ICMP[数据包](http://baike.baidu.com/view/25880.htm)进行[带宽](http://baike.baidu.com/view/10821.htm)限制，将ICMP占用的带宽控制在一定的范围内，这样即使有ICMP攻击，它所占用的带宽也是非常有限的，对整个网络的影响非常少；第二种方法就是在[主机](http://baike.baidu.com/view/23880.htm)上设置ICMP数据包的处理规则，最好是设定拒绝所有的ICMP数据包。

设置ICMP数据包处理规则的方法也有两种，一种是在[操作系统](http://baike.baidu.com/view/880.htm)上设置[包过滤](http://baike.baidu.com/view/150853.htm)，另一种是在[主机](http://baike.baidu.com/view/23880.htm)上安装[防火墙](http://baike.baidu.com/view/3067.htm)。具体设置如下：

**1．在Windows 2000 Server中设置ICMP过滤**

[Windows 2000 Server](http://baike.baidu.com/view/765690.htm)提供了“[路由](http://baike.baidu.com/view/18655.htm)与[远程访问](http://baike.baidu.com/view/183974.htm)”服务，但是默认情况下是没有启动的，因此首先要启动它：点击“[管理工具](http://baike.baidu.com/view/1251417.htm)”中的“路由与远程访问”，启动设置向导。在其中选择“手动配置服务器”项，点击[下一步]按钮。稍等片刻后，系统会提示“路由和[远程访问服务](http://baike.baidu.com/view/253162.htm)现在已被安装。要开始服务吗？”，点击[是]按钮启动服务。

[](http://baike.baidu.com/pic/ICMP/572452/0/a28d62d9a83a41f139012fe7?fr=lemma&ct=single)图1

服务启动后，在[计算机](http://baike.baidu.com/view/3314.htm)名称的分支下会出现一个“IP[路由选择](http://baike.baidu.com/view/632880.htm)”，点击它展开分支，再点击“常规”，会在右边出现服务器中的网络连接（即[网卡](http://baike.baidu.com/view/4230.htm)）。用[鼠标](http://baike.baidu.com/view/2199.htm)右键点击你要配置的网络连接，在弹出的菜单中点击“属性”，会弹出一个网络连接属性的窗口，如图1所示。

图1中有两个按钮，一个是“输入筛选器”（指对此服务器接受的[数据包](http://baike.baidu.com/view/25880.htm)进行筛选），另一个是“输出筛选器”（指对此服务器发送的数据包进行筛选），这里应该点击[输入筛选器] 按钮，会弹出一个“添加筛选器”窗口，再点击[添加]按钮，表示要增加一个筛选条件。

在“协议”右边的下拉列表中选择“ICMP”，在随后出现的“ICMP类型”和“ICMP编码”中均输入“255”，代表所有

[](http://baike.baidu.com/pic/ICMP/572452/0/35e940dfc4818b0b495403ed?fr=lemma&ct=single)图2

的ICMP类型及其编码。ICMP有许多不同的类型（Ping就是一种类型），每种类型也有许多不同的状态，用不同的“编码”来表示。因为其类型和编码很复杂，这里不再叙述。

点击[确定]按钮返回“输入筛选器”窗口，此时会发现“筛选器”列表中多了一项内容（如图2所示）。点击[确定]按钮返回“[本地连接](http://baike.baidu.com/view/179248.htm)”窗口，再点击[确定]按钮，此时筛选器就生效了，从其他[计算机](http://baike.baidu.com/view/3314.htm)上Ping这台[主机](http://baike.baidu.com/view/23880.htm)就不会成功了。

[](http://baike.baidu.com/pic/ICMP/572452/0/b94f65ec7fd176c92f2e21eb?fr=lemma&ct=single)图3

**2． 用防火墙设置ICMP过滤**

许多[防火墙](http://baike.baidu.com/view/3067.htm)在默认情况下都启用了ICMP过滤的功能。如果没有启用，只要选中“防御ICMP攻击”、“防止别人用[ping命令](http://baike.baidu.com/view/36175.htm)[探测](http://baike.baidu.com/view/716064.htm)”就可以了，如图3所示。

## ICMP防御方法

**选择合适的防火墙**

有效防止ICMP攻击，[防火墙](http://baike.baidu.com/view/3067.htm)应该具有状态检测、细致的[数据包](http://baike.baidu.com/view/25880.htm)完整性检查和很好的过滤规则控制功能。

[状态检测防火墙](http://baike.baidu.com/view/1638561.htm)通过跟踪它的连接状态，动态允许外出数据包的响应信息进入防火墙所保护的网络。例如，状态检测防火墙可以记录一个出去的 PING（ICMP Echo Request），在接下来的一个确定的时间段内，允许目标主机响应的ICMP Echo Reply直接发送给前面发出了PING命令的IP，除此之外的其他ICMP Echo Reply消息都会被防火墙阻止。与此形成对比的是，包过滤类型的防火墙允许所有的ICMP Echo Reply消息进入防火墙所保护的网络了。许多路由器和基于[Linux内核](http://baike.baidu.com/view/573460.htm)2.2或以前版本的防火墙系统，都属于包过滤型，用户应该避免选择这些系统。

新的攻击不断出现，防火墙仅仅能够防止已知攻击是远远不够的。通过对所有数据包进行细致分析，删除非法的数据包，防火墙可以防止已知和未知的 DoS攻击。这就要求防火墙能够进行数据包一致性检查。[安全策略](http://baike.baidu.com/view/160028.htm)需要针对ICMP进行细致的控制。因此防火墙应该允许对ICMP类型、代码和包大小进行过滤，并且能够控制连接时间和ICMP包的生成速率。

**配置防火墙以预防攻击**

一旦选择了合适的防火墙，用户应该配置一个合理的安全策略。以下是被普遍认可的防火墙安全配置惯例，可供管理员在系统安全性和易用性之间作出权衡。

防火墙应该强制执行一个缺省的拒绝策略。除了出站的ICMP Echo Request、出站的ICMP Source Quench、进站的TTL Exceeded和进站的ICMP Destination Unreachable之外，所有的ICMP消息类型都应该被阻止。

## ICMPICMP类型

这是一个完整的ICMP类型的列表：**Table C-1. ICMP类型**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TYPE** | **CODE** | **Description** | **Query** | **Error** |
| 0 | 0 | Echo Reply——回显应答（Ping应答） | x |  |
| 3 | 0 | Network Unreachable——网络不可达 |  | x |
| 3 | 1 | Host Unreachable——主机不可达 |  | x |
| 3 | 2 | Protocol Unreachable——协议不可达 |  | x |
| 3 | 3 | Port Unreachable——端口不可达 |  | x |
| 3 | 4 | Fragmentation needed but no frag. bit set——需要进行分片但设置不分片比特 |  | x |
| 3 | 5 | Source routing failed——源站选路失败 |  | x |
| 3 | 6 | Destination network unknown——目的网络未知 |  | x |
| 3 | 7 | Destination host unknown——目的主机未知 |  | x |
| 3 | 8 | Source host isolated (obsolete)——源主机被隔离（作废不用） |  | x |
| 3 | 9 | Destination network administratively prohibited——目的网络被强制禁止 |  | x |
| 3 | 10 | Destination host administratively prohibited——目的主机被强制禁止 |  | x |
| 3 | 11 | Network unreachable for TOS——由于服务类型TOS，网络不可达 |  | x |
| 3 | 12 | Host unreachable for TOS——由于服务类型TOS，主机不可达 |  | x |
| 3 | 13 | Communication administratively prohibited by filtering——由于过滤，通信被强制禁止 |  | x |
| 3 | 14 | Host precedence violation——主机越权 |  | x |
| 3 | 15 | Precedence cutoff in effect——优先中止生效 |  | x |
| 4 | 0 | Source quench——源端被关闭（基本流控制） |  |  |
| 5 | 0 | Redirect for network——对网络重定向 |  |  |
| 5 | 1 | Redirect for host——对主机重定向 |  |  |
| 5 | 2 | Redirect for TOS and network——对服务类型和网络重定向 |  |  |
| 5 | 3 | Redirect for TOS and host——对服务类型和主机重定向 |  |  |
| 8 | 0 | Echo request——回显请求（Ping请求） | x |  |
| 9 | 0 | Router advertisement——路由器通告 |  |  |
| 10 | 0 | Route solicitation——路由器请求 |  |  |
| 11 | 0 | TTL equals 0 during transit——传输期间生存时间为0 |  | x |
| 11 | 1 | TTL equals 0 during reassembly——在数据报组装期间生存时间为0 |  | x |
| 12 | 0 | IP header bad (catchall error)——坏的IP首部（包括各种差错） |  | x |
| 12 | 1 | Required options missing——缺少必需的选项 |  | x |
| 13 | 0 | Timestamp request (obsolete)——时间戳请求（作废不用） | x |  |
| 14 |  | Timestamp reply (obsolete)——时间戳应答（作废不用） | x |  |
| 15 | 0 | Information request (obsolete)——信息请求（作废不用） | x |  |
| 16 | 0 | Information reply (obsolete)——信息应答（作废不用） | x |  |
| 17 | 0 | Address mask request——地址掩码请求 | x |  |
| 18 | 0 | Address mask reply——地址掩码应答 |  |  |

# [ping 原理与ICMP协议](http://www.cnblogs.com/Akagi201/archive/2012/03/26/2418475.html)

## **ping** 的**原理**

**ping** 程序是用来探测**主机到主机之间是否可通信**，如果不能**ping**到某台主机，表明不能和这台主机建立连接。**ping** 使用的是**ICMP协议**，**它发送icmp回送请求消息给目的主机**。**ICMP协议规定：目的主机必须返回ICMP回送应答消息给源主机。**如果源主机在一定时间内收到应答，则认为主机可达。

**ICMP协议通过IP协议发送的，IP协议是一种无连接的，不可靠的数据包协议。**在Unix/Linux，序列号从0开始计数，依次递增。而Windows　**ping**程序的ICMP序列号是没有规律。

**ICMP协议在实际传输中数据包：20字节IP首部 + 8字节ICMP首部+ 1472字节<数据大小>38字节**

**ICMP报文格式:IP首部(20字节)+8位类型+8位代码+16位校验和+(不同的类型和代码，格式也有所不同)**

**Ping工作过程——**  
    假定主机A的IP地址是192.168.1.1，主机B的IP地址是192.168.1.2，都在同一子网内，则当你在主机A上运行“**Ping** 192.168.1.2”后，都发生了些什么呢?  
首先，**Ping**命令会构建一个固定格式的ICMP请求数据包，然后由ICMP协议将这个数据包连同地址“192.168.1.2”一起交给IP层协议（和ICMP一样，实际上是一组后台运行的进程），IP层协议将以地址“192.168.1.2”作为目的地址，本机IP地址作为源地址，加上一些其他的控制信息，构建一个IP数据包，并在一个映射表中查找出IP地址192.168.1.2所对应的物理地址（也叫MAC地址，熟悉网卡配置的朋友不会陌生，这是数据链路层协议构建数据链路层的传输单元——帧所必需的），一并交给数据链路层。后者构建一个数据帧，目的地址是IP层传过来的物理地址，源地址则是本机的物理地址，还要附加上一些控制信息，依据以太网的介质访问规则，将它们传送出去。

    其中映射表由ARP实现。ARP(Address Resolution Protocol)是地址解析协议,是一种将IP地址转化成物理地址的协议。ARP具体说来就是将网络层（IP层，也就是相当于OSI的第三层）地址解析为数据连接层（MAC层，也就是相当于OSI的第二层）的MAC地址。  
  
    主机B收到这个数据帧后，先检查它的目的地址，并和本机的物理地址对比，如符合，则接收；否则丢弃。接收后检查该数据帧，将IP数据包从帧中提取出来，交给本机的IP层协议。同样，IP层检查后，将有用的信息提取后交给ICMP协议，后者处理后，马上构建一个ICMP应答包，发送给主机A，其过程和主机A发送ICMP请求包到主机B一模一样。

即先由IP地址，在网络层传输，然后再根据mac地址由数据链路层传送到目的主机

## ICMP——

### 1.IMCP协议介绍

前面讲到了，IP协议并不是一个可靠的协议，它不保证数据被送达，那么，自然的，保证数据送达的工作应该由其他的模块来完成。其中一个重要的模块就是ICMP(网络控制报文)协议。

当传送IP数据包发生错误－－比如主机不可达，路由不可达等等，ICMP协议将会把错误信息封包，然后传送回给主机。给主机一个处理错误的机会，这 也就是为什么说建立在IP层以上的协议是可能做到安全的原因。ICMP数据包由8bit的错误类型和8bit的代码和16bit的校验和组成。而前 16bit就组成了ICMP所要传递的信息。

尽管在大多数情况下，错误的包传送应该给出ICMP报文，但是在特殊情况下，是不产生ICMP错误报文的。如下

1. ICMP差错报文不会产生ICMP差错报文（出IMCP查询报文）（防止IMCP的无限产生和传送）
2. 目的地址是广播地址或多播地址的IP数据报。
3. 作为链路层广播的数据报。
4. 不是IP分片的第一片。
5. 源地址不是单个主机的数据报。这就是说，源地址不能为零地址、环回地址、广播地 址或多播地址。

虽然里面的一些规定现在还不是很明白，但是所有的这一切规定，都是为了防止产生ICMP报文的无限传播而定义的。

ICMP协议大致分为两类，一种是查询报文，一种是差错报文。其中查询报文有以下几种用途:

1. **ping**查询
2. 子网掩码查询（用于无盘工作站在初始化自身的时候初始化子网掩码）
3. 时间戳查询（可以用来同步时间）

而差错报文则产生在数据传送发生错误的时候。就不赘述了。

### 2.ICMP的应用--**ping**

**ping**可以说是ICMP的最著名的应用，当我们某一个网站上不去的时候。通常会**ping**一下这个网站。**ping**会回显出一些有用的信息。一般的信息如下:

Reply from 10.4.24.1: bytes=32 time<1ms TTL=255  
Reply from 10.4.24.1: bytes=32 time<1ms TTL=255  
Reply from 10.4.24.1: bytes=32 time<1ms TTL=255  
Reply from 10.4.24.1: bytes=32 time<1ms TTL=255  
  
**Ping** statistics for 10.4.24.1:  
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),  
Approximate round trip times in milli-seconds:  
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

**ping**这个单词源自声纳定位，而这个程序的作用也确实如此，它利用ICMP协议包来侦测另一个主机是否可达。**原理**是用类型码为0的ICMP发请 求，受到请求的主机则用类型码为8的ICMP回应。**ping**程序来计算间隔时间，并计算有多少个包被送达。用户就可以判断网络大致的情况。我们可以看到， **ping**给出来了传送的时间和TTL的数据。我给的例子不太好，因为走的路由少，有兴趣地可以**ping**一下国外的网站比如sf.net，就可以观察到一些 丢包的现象，而程序运行的时间也会更加的长。  
**ping**还给我们一个看主机到目的主机的路由的机会。这是因为，ICMP的**ping**请求数据报在每经过一个路由器的时候，路由器都会把自己的ip放到该数 据报中。而目的主机则会把这个ip列表复制到回应icmp数据包中发回给主机。但是，无论如何，ip头所能纪录的路由列表是非常的有限。如果要观察路由， 我们还是需要使用更好的工具，就是要讲到的Traceroute(windows下面的名字叫做tracert)。

### 3.ICMP的应用--Traceroute

Traceroute是用来侦测主机到目的主机之间所经路由情况的重要工具，也是最便利的工具。前面说到，尽管**ping**工具也可以进行侦测，但是，因为ip头的限制，**ping**不能完全的记录下所经过的路由器。所以Traceroute正好就填补了这个缺憾。

Traceroute的**原理**是非常非常的有意思，它受到目的主机的IP后，首先给目的主机发送一个TTL=1（还记得TTL是什么吗？）的UDP(后面就 知道UDP是什么了)数据包，而经过的第一个路由器收到这个数据包以后，就自动把TTL减1，而TTL变为0以后，路由器就把这个包给抛弃了，并同时产生 一个主机不可达的ICMP数据报给主机。主机收到这个数据报以后再发一个TTL=2的UDP数据报给目的主机，然后刺激第二个路由器给主机发ICMP数据 报。如此往复直到到达目的主机。这样，traceroute就拿到了所有的路由器ip。从而避开了ip头只能记录有限路由IP的问题。

有人要问，我怎么知道UDP到没到达目的主机呢？这就涉及一个技巧的问题，TCP和UDP协议有一个端口号定义，而普通的网络程序只监控少数的几个号码较 小的端口，比如说80,比如说23,等等。而traceroute发送的是端口号>30000(真变态)的UDP报，所以到达目的主机的时候，目的 主机只能发送一个端口不可达的ICMP数据报给主机。主机接到这个报告以后就知道，主机到了。