# [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。

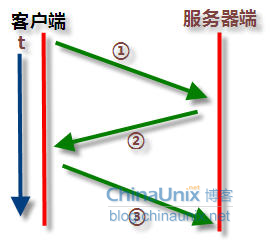


图1 三次握手协议示意图[1]

**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：

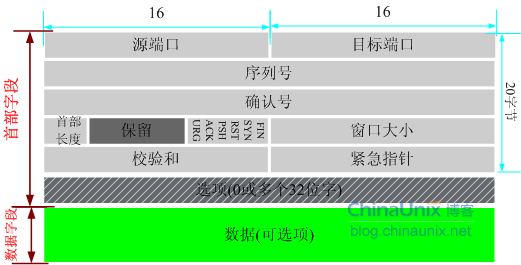


图2 TCP报文格式

**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)：

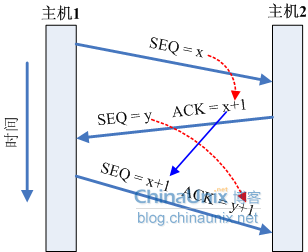


图3 正常情况下TCP连接建立过程

**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)：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：

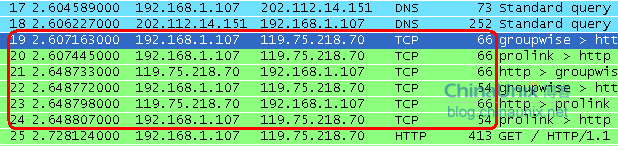


图4 Wireshark俘获建立TCP连接分组

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：

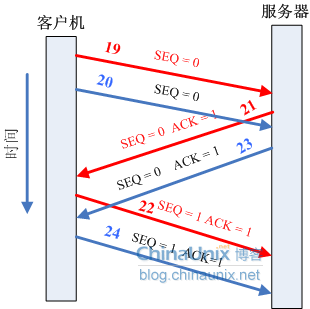


图5 TCP连接建立实例

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：

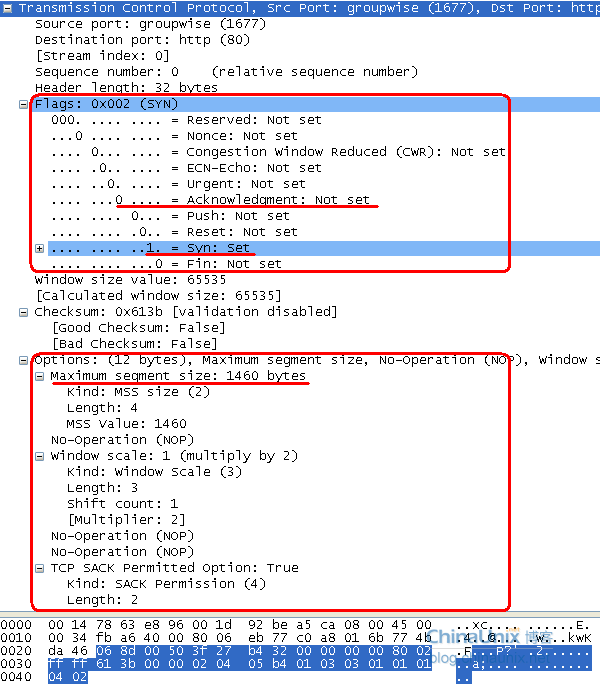


图6 TCP连接第一次握手实例

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]：

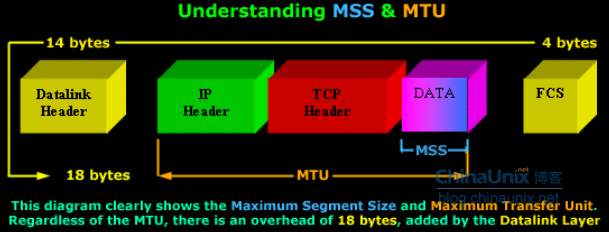


图7 MSS与MTU关系

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：

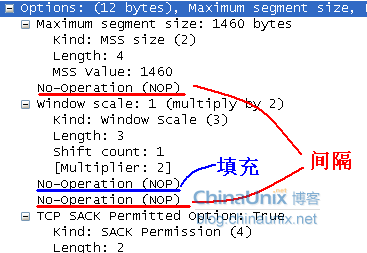


图8 TCP报文NOP字段

**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：

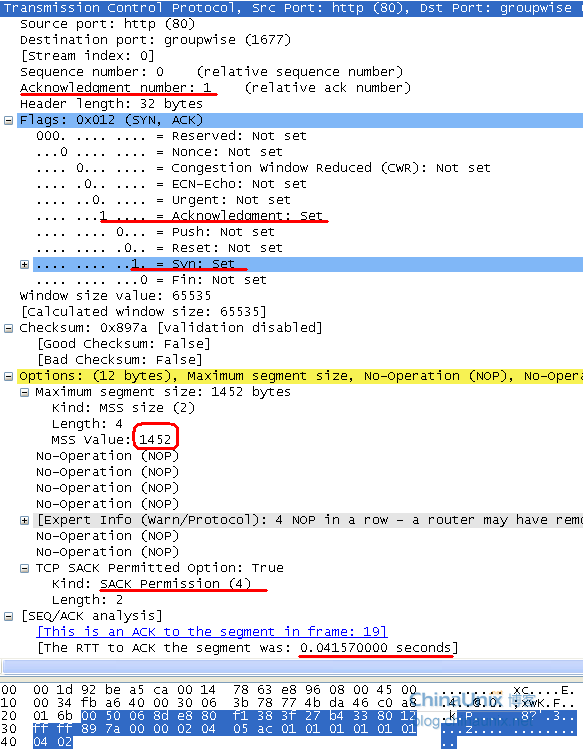


图9 TCP连接第二次握手实例

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：

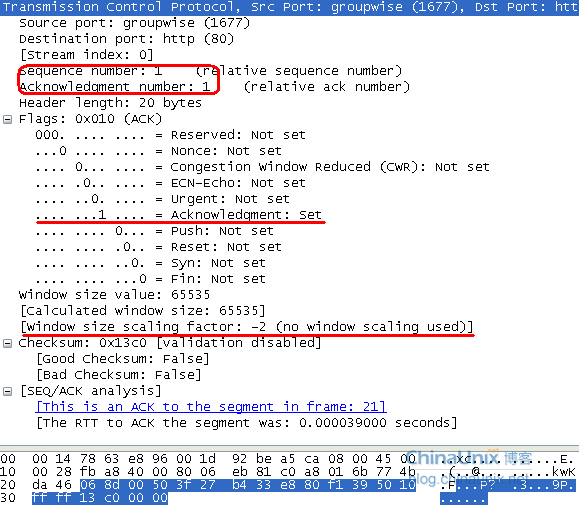


图10 TCP连接第三次握手实例

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:-)

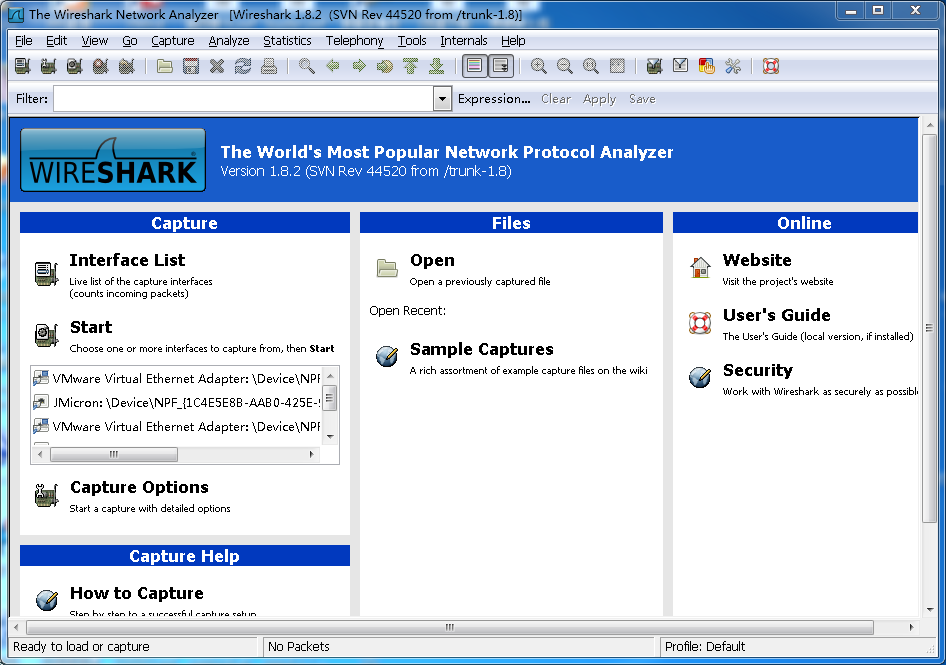
# wireshark抓包详细图文教程

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.**

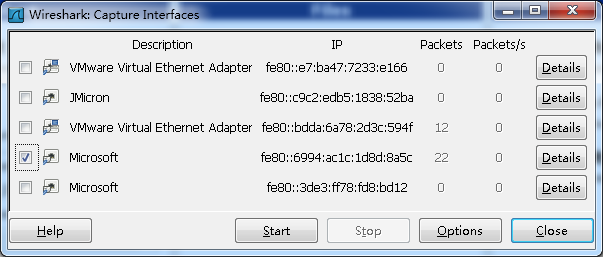
**wireshark 开始抓包**

开始界面

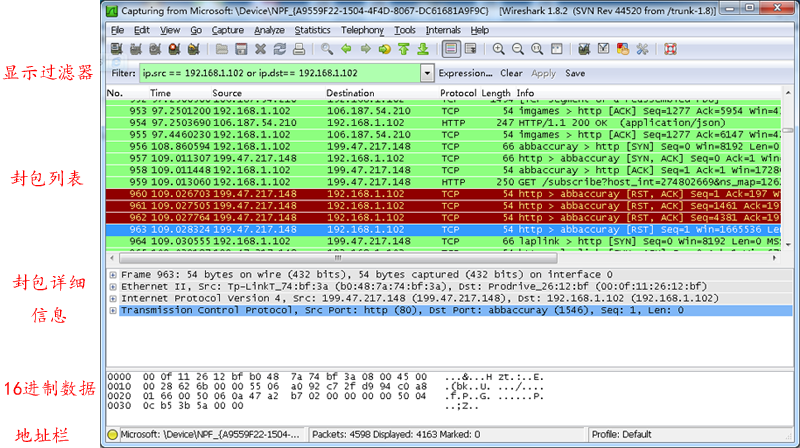


**wireshark是捕获机器上的某一块网卡的网络包**，当你的机器上有多块网卡的时候，你需要选择一个网卡。

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。

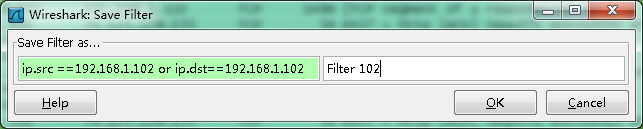
过滤器有两种，

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。

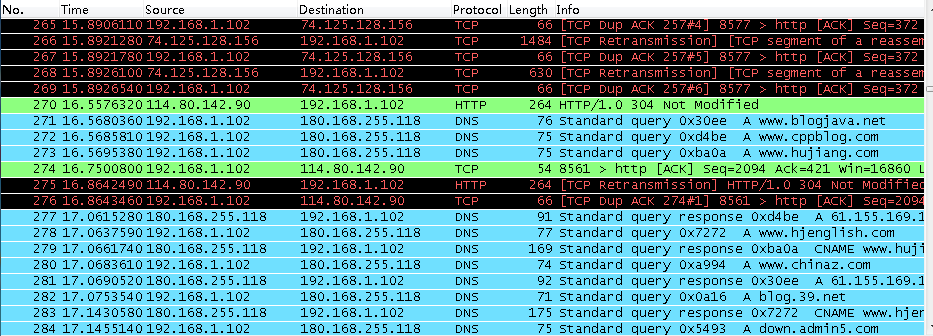
http://www.9upk.com/article/UploadPic/2013-5/20135220328630.png

|  |  |
| --- | --- |
| 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

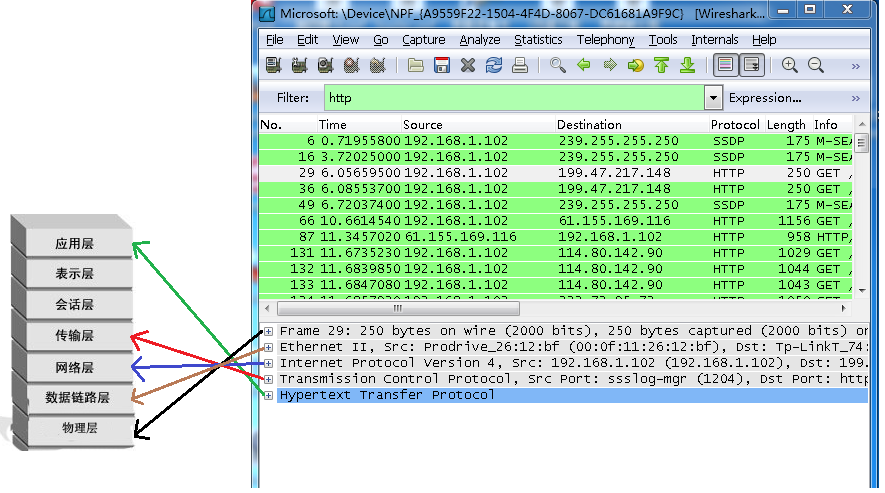
Ethernet II: 数据链路层以太网帧头部信息

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

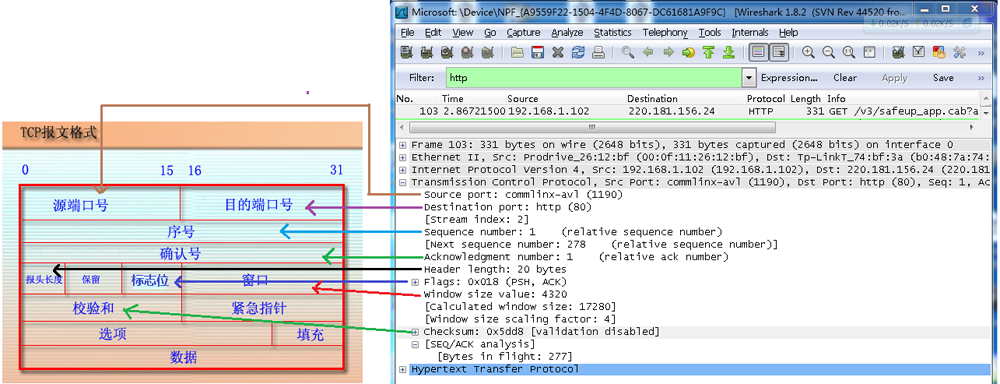
Hypertext Transfer Protocol:  应用层的信息，此处是HTTP协议

**第 3 页 wireshark与对应的OSI七层模型**

****

**TCP包的具体内容**

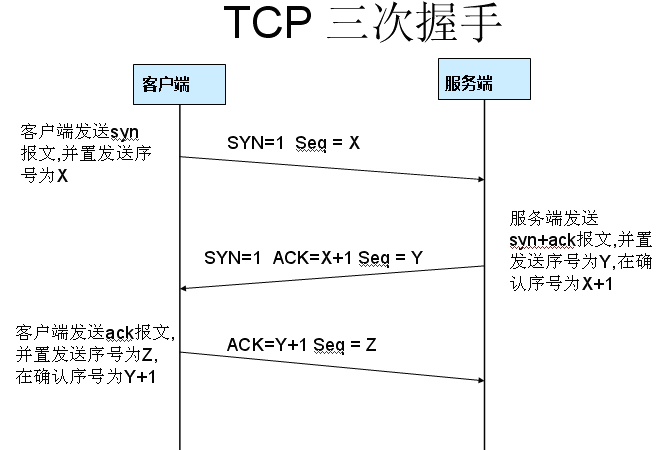
 从下图可以看到wireshark捕获到的TCP包中的每个字段。



**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

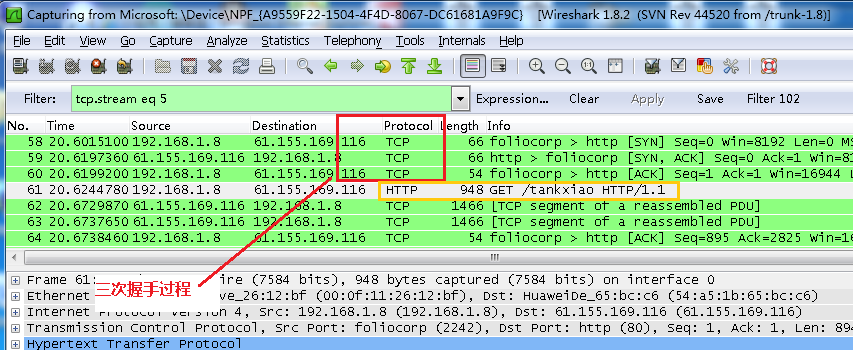


这图我都看过很多遍了， 这次我们用wireshark实际分析下三次握手的过程。

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",

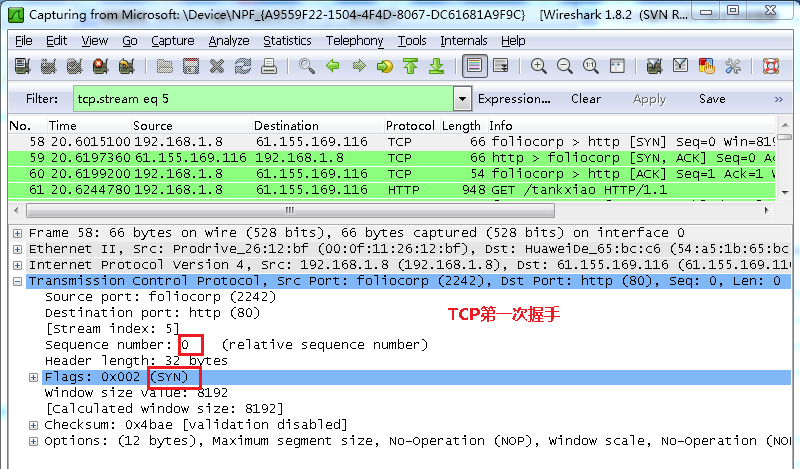
这样做的目的是为了得到与浏览器打开网站相关的数据包，将得到如下图



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。

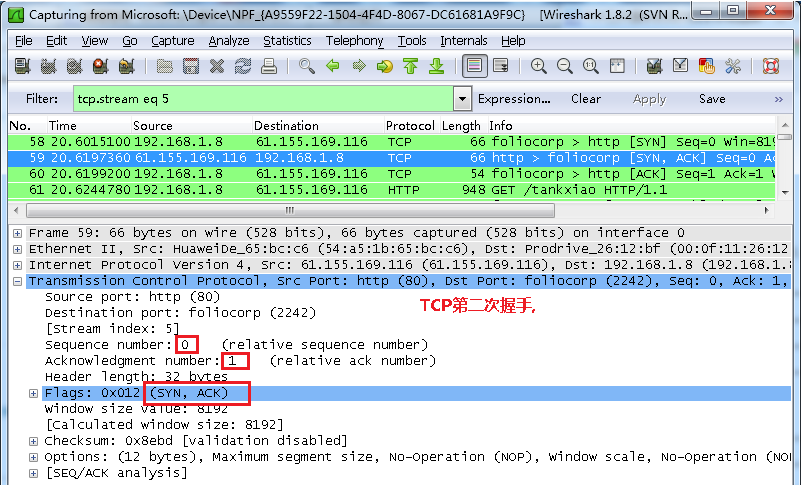
**第一次握手数据包**

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



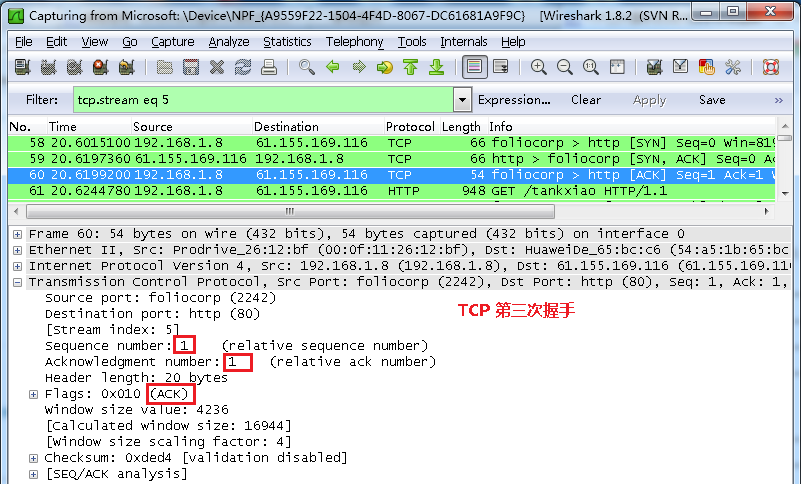
**第二次握手的数据包**

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**

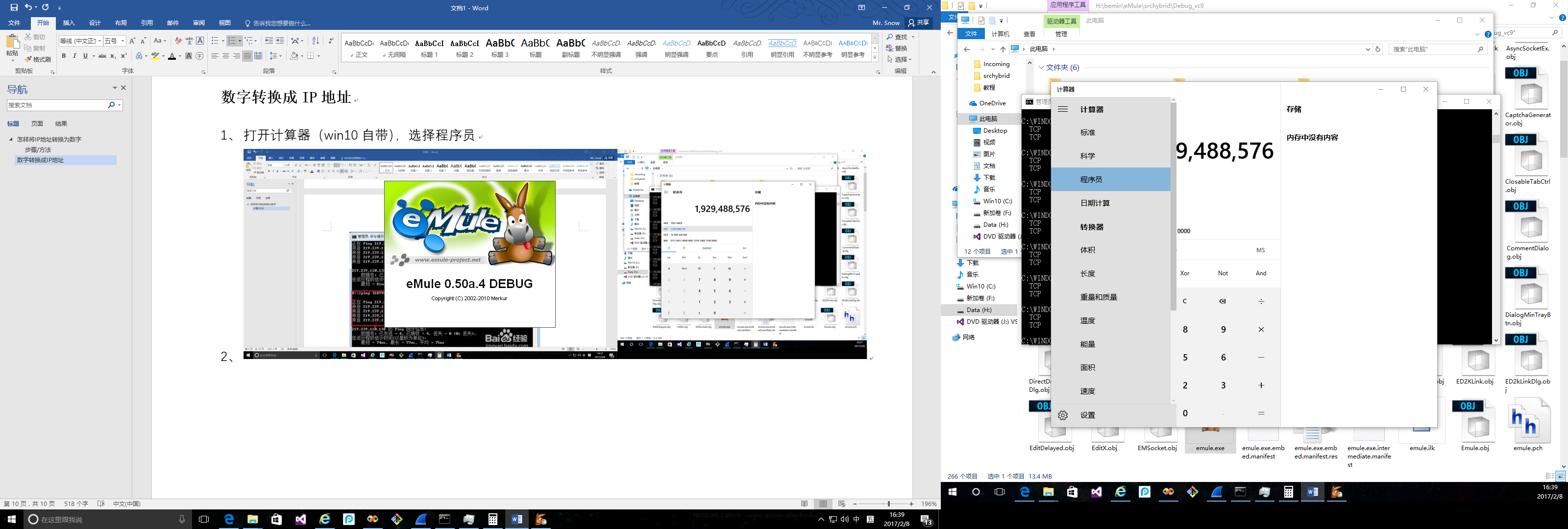
客户端再次发送确认包(ACK) SYN标志位为0,ACK标志位为1.并且把服务器发来ACK的序号字段+1,放在确定字段中发送给对方.并且在数据段放写ISN的+1, 如下图:



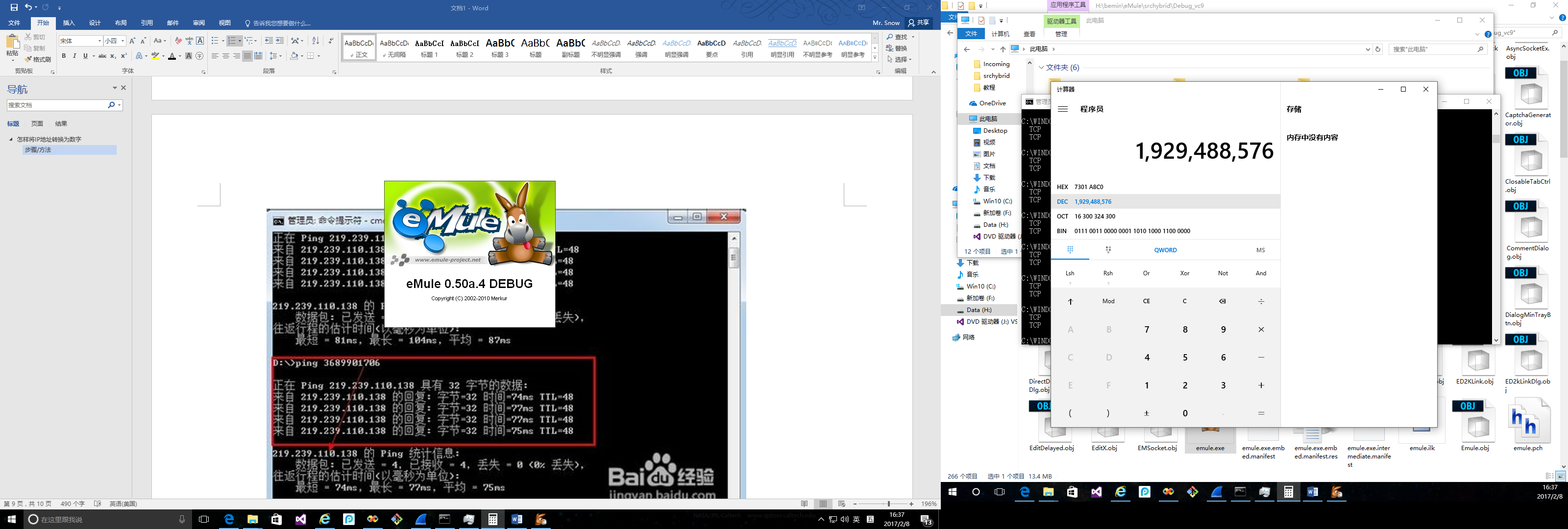
 就这样通过了TCP三次握手，建立了连接

# 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**