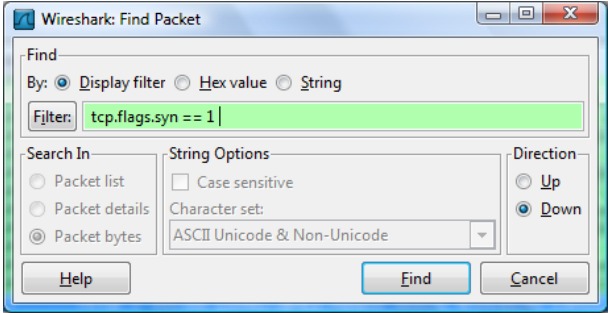
**一、应用Wireshark观察基本网络协议**

**TCP:**

TCP/IP通过三次握手建立一个连接。这一过程中的三种报文是：SYN，SYN/ACK，ACK。

第一步是找到PC发送到网络服务器的第一个SYN报文，这标识了TCP三次握手的开始。

如果你找不到第一个SYN报文，选择**Edit -> Find Packet**菜单选项。选择Display Filter，输入过滤条件：tcp.flags，这时会看到一个flag列表用于选择。选择合适的flag，tcp.flags.syn并且加上==1。点击Find，之后trace中的第一个SYN报文就会高亮出来了。

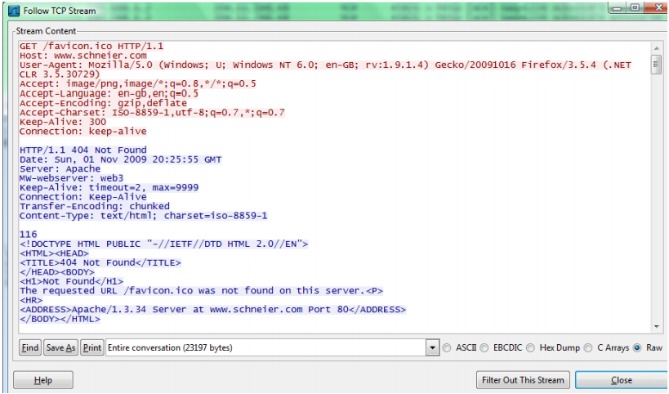


注意：Find Packet也可以用于搜索十六进制字符，比如恶意软件信号，或搜索字符串，比如抓包文件中的协议命令。

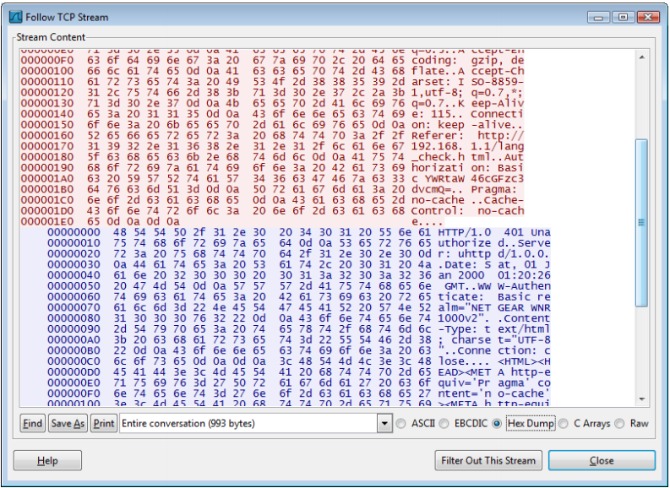
一个快速过滤TCP报文流的方式是在**Packet List Panel**中右键报文，并且选择**Follow TCP Stream**。这就创建了一个只显示TCP会话报文的自动过滤条件。

这一步骤会弹出一个会话显示窗口，默认情况下包含TCP会话的ASCII代码，客户端报文用红色表示服务器报文则为蓝色。

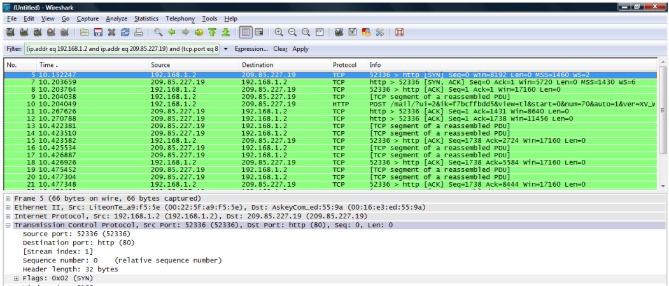
窗口类似下图所示，对于读取协议有效载荷非常有帮助，比如HTTP，SMTP，FTP。



更改为十六进制Dump模式查看载荷的十六进制代码，如下图所示：



关闭弹出窗口，Wireshark就只显示所选TCP报文流。现在可以轻松分辨出3次握手信号。



注意：这里Wireshark自动为此TCP会话创建了一个显示过滤。本例中：(ip.addr eq 192.168.1.2 and ip.addr eq 209.85.227.19) and (tcp.port eq 80 and tcp.port eq 52336)

**SYN**报文：

图中显示的5号报文是从客户端发送至服务器端的SYN报文，此报文用于与服务器建立同步，确保客户端和服务器端的通信按次序传输。SYN报文的头部有一个32 bit序列号。底端对话框显示了报文一些有用信息如报文类型，序列号。

**SYN/AC**K报文：

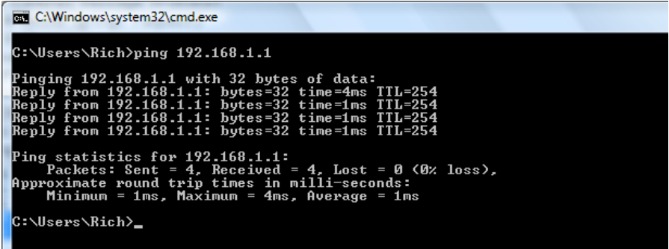
7号报文是服务器的响应。一旦服务器接收到客户端的SYN报文，就读取报文的序列号并且使用此编号作为响应，也就是说它告知客户机，服务器接收到了SYN报文，通过对原SYN报文序列号加一并且作为响应编号来实现，之后客户端就知道服务器能够接收通信。

**ACK**报文：

8号报文是客户端对服务器发送的确认报文，告诉服务器客户端接收到了SYN/ACK报文，并且与前一步一样客户端也将序列号加一，此包发送完毕，客户端和服务器进入ESTABLISHED状态，完成三次握手。

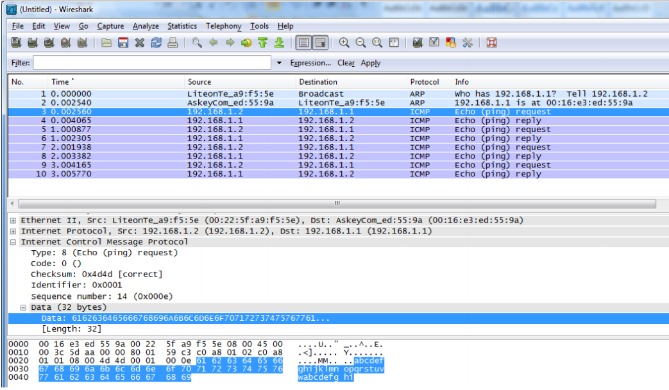
**ARP & ICMP：**

开启Wireshark抓包。打开Windows控制台窗口，使用ping命令行工具查看与相邻机器的连接状况。



停止抓包之后，Wireshark如下图所示。

ARP和ICMP报文相对较难辨认，创建只显示ARP或ICMP的过滤条件。



**ARP**报文：

地址解析协议，即ARP（Address Resolution Protocol），是根据IP地址获取物理地址的一个TCP/IP协议。其功能是：主机将ARP请求广播到网络上的所有主机，并接收返回消息，确定目标IP地址的物理地址，同时将IP地址和硬件地址存入本机ARP缓存中，下次请求时直接查询ARP缓存。

最初从PC发出的ARP请求确定IP地址192.168.1.1的MAC地址，并从相邻系统收到ARP回复。ARP请求之后，会看到ICMP报文。

**ICMP**报文：

网络控制消息协定（Internet Control Message Protocol，ICMP）用于TCP/IP网络中发送控制消息，提供可能发生在通信环境中的各种问题反馈，通过这些信息，令管理者可以对所发生的问题作出诊断，然后采取适当的措施解决。

PC发送echo请求，收到echo回复如上图所示。ping报文被mark成Type 8，回复报文mark成Type 0。

如果多次ping同一系统，在PC上删除ARP cache，使用如下ARP命令之后，会产生一个新的ARP请求。

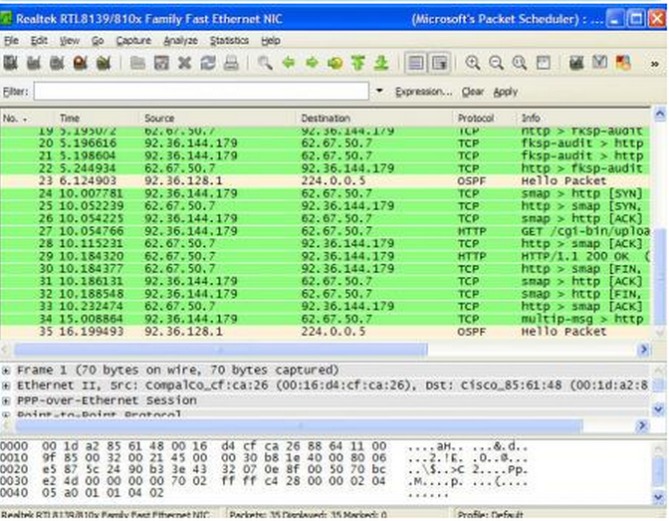
C:\> ping 192.168.1.1

… ping output …

C:\> arp –d \*

**HTTP：**

HTTP协议是目前使用最广泛的一种基础协议，这得益于目前很多应用都基于WEB方式，实现容易，软件开发部署也简单，无需额外的客户端，使用浏览器即可使用。这一过程开始于请求服务器传送网络文件。



从上图可见报文中包括一个GET命令，当HTTP发送初始GET命令之后，TCP继续数据传输过程，接下来的链接过程中HTTP会从服务器请求数据并使用TCP将数据传回客户端。传送数据之前，服务器通过发送HTTP OK消息告知客户端请求有效。如果服务器没有将目标发送给客户端的许可，将会返回403 Forbidden。如果服务器找不到客户端所请求的目标，会返回404。

如果没有更多数据，连接可被终止，类似于TCP三次握手信号的SYN和ACK报文，这里发送的是FIN和ACK报文。当服务器结束传送数据，就发送FIN/ACK给客户端，此报文表示结束连接。接下来客户端返回ACK报文并且对FIN/ACK中的序列号加1。这就从服务器端终止了通信。要结束这一过程客户端必须重新对服务器端发起这一过程。必须在客户端和服务器端都发起并确认FIN/ACK过程。

**二、Wireshark抓包实例诊断TCP连接问题**

前文论述了TCP基础知识，从本节开始，通过TCP抓包实例来诊断TCP常见问题。

TCP进程通讯时，双方打开连接，发送数据，最后关闭连接。当TCP打开连接时，从源端口到目的端口发送一个请求。在应用建立或关闭时可能发生一些问题。本文讨论用Wireshark网络抓包的方法来定位及解决这一问题。

**问题的表现形式:**

问题可能有多种表现类型：

·         尝试运行应用程序但发现应用程序无法工作。尝试浏览网络但无法获得响应。

·         尝试发送邮件但无法连接到邮件服务器。

·         问题可能由简单原因引起，如服务器宕机，服务器上没有运行应用程序，或在客户端到服务器的某一处网络断开。

·         问题也可能由复杂原因引起，如DNS问题，服务器内存不足无法连接（例如某一应用占用高内存空间），重复IP，以及其他原因。

**处理方法:**

下文会介绍解决问题的线索以及如何通过抓包来诊断TCP连接问题。通常，这些问题会导致运行应用程序时无法得到任何结果。

当你在运行一个应用程序时，例如**数据库**客户端，邮件客户端，观看视频等等，而又无法获得输出，按照以下步骤诊断：

1. 确认服务器和应用程序正在运行。
2. 确认客户端正在运行，IP地址已配置（手动或通过DHCP），并连接至网络。
3. Ping服务器并确认连接正常。
4. 在某些情况下，ping不通服务器但连接正常。这是由于防火墙拦截了ICMP信息，所以如果无法ping通并不一定表示连接有问题。防火墙可能是网络中的专用设备或Windows/Linux/UNIX终端设备上安装的防火墙。

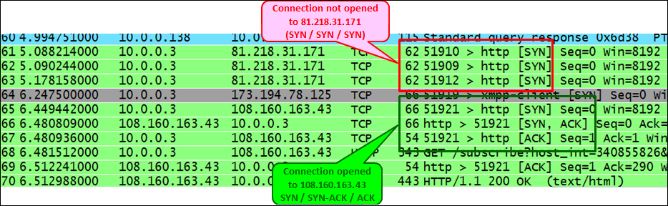
5.     抓包文件中，查找以下模式：

·         三重SYN信息而没有响应（见以下截屏）

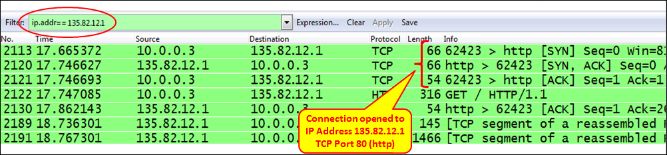
·         SYN信息带一个reset(RST)响应

**这两种情况下都有可能是防火墙拦截了特定应用程序或应用程序没有在运行。**

以下截屏是一个简单的case：客户端无法连接到web服务器81.218.31.171（报文61,62和63）。可能是由于不被防火墙允许，或服务器发生故障。可以看到另一个站点108.160.163.43（报文65,66和67）的连接正常，因此连接问题仅限于81.218.31.171。



下例是一个这种情况相对复杂的case。该case中，客户想要登录到camera服务器来访问远程站点的camera。camera服务器的IP地址为135.82.12.1，问题在于客户能够看到服务器主页上的登录窗口，但无法登进系统。在下面的截图中可以看到，打开了一个到IP地址135.82.12.1的连接。到HTTP服务器的TCP连接是打开的，一开始看上去没有连接问题：



当我们过滤出目的IP地址为135.82.12.1的数据流，也就是camera服务器。这里可以看到，当尝试连接TCP端口6036时，得到了一个RST/ACK响应，有以下可能性：

·         防火墙拦截了端口6036

·         如果配置了端口地址转换（PAT），那么仅转换端口80而非6036

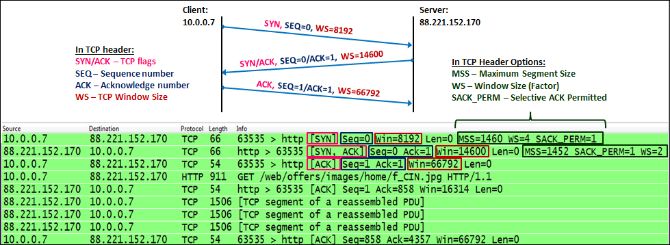
·         用户名和密码验证是在TCP端口6036上完成的，防火墙仅允许端口80，验证被拦截，应用无法工作



**总之，当无法正常连接服务器时，检查服务器和客户端是否所有TCP/UDP端口都能通过网络转发，以及是否有未知的端口。**

**工作过程：**

TCP连接开始时，发生了以下三步：



1.     客户端TCP进程发送了一个SYN报文。该报文中SYN标志位设置为1。这一报文中客户端：

·         指定自己的初始序列号。这是客户端发送给服务器的第一个字节。

·         指明自己的窗口大小。这是客户端分配给进程的缓存大小（位于客户端的RAM）。

·         设置自己将要使用的选项：MSS，Selective ACK，等等。

2.     当服务器收到建立连接请求，服务器：

·         发送SYN/ACK给客户端，确认接收到SYN请求。

·         指明服务器端的初始序列号。这是服务器发送给客户端的第一个字节。

·         指明服务器的窗口大小。这是服务器分配给进程的缓存大小（位于服务器RAM）。

·         回复请求选项并设置服务器端选项。

3.     当接收到服务器的SYN/ACK，客户端：

·         发送ACK报文给服务器，确认从服务器接收到SYN/ACK.

·         指明客户端窗口大小。尽管这一参数在第一个报文中定义过了，服务器还是会参考这个值，因为这是最新的窗口大小。

       在TCP头部的选项字段中，有以下几个主要选项：

·         Maximum Segment Size（MSS）：TCP数据报的最大字节数，即从TCP头部开始直到报文末尾的字节数。

·         Windows Scale Option (WSopt)：这一因子与TCP头部的Window Size字段相乘，通知接收方扩大缓存。由于头部最大窗口大小是64KB，乘以因子4也就是256KB窗口大小。

·         SACK：Selective ACK，该选项使连接双方能够仅确认指定报文，当单个报文丢失，只有这个报文会被重传。连接建立时，双方都需要同意SACK。

·         Timestamps Option（TSopt）：该参数指客户端和服务器之间的延时。

       在这一阶段，双方：

·         同意建立连接

·         知道对方的初始序列号

·         知道对方的窗口大小

**在建立连接时，除了三路握手信号之外，其他都表示有问题。**包括SYN没有响应，SYN之后SYN/ACK最后没有ACK，SYN响应为RST，等等。

**总结：**

·         **如果SYN报文收到回复RST，则检查拦截了port号的防火墙。**

·         **三次SYN而没有任何回复，或者是由于应用程序没有响应，或者是由于防火墙拦截了特定端口上的请求。**

·         **永远记住确认一下是否有NAT**，**端口转发，以及涉及TCP和UDP端口的机制。这些机制可能会中断TCP正常操作。**

**三、Wireshark抓包实例分析TCP重传**

TCP发送一个或一组报文，会等待收到报文的确认信息。重传，即发生在报文没有到达或确认信息没有及时返回的情况下。当发现网速变慢时，原因之一可能就是重传。发生重传的原因有多种，在客户机或服务器两边端口应用Wireshark有助于诊断问题。本文通过抓包实例阐述各种可能性。

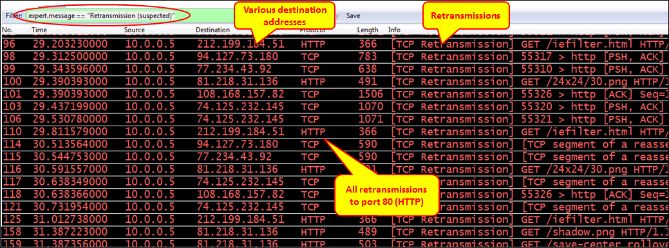
**诊断过程:**

1. 在相应端口开始抓数据。
2. 找到**Analyze** | **Expert Info**菜单。
3. 在**Notes**之下，查找**Retransmission**。
4. 点击（+）符号即可打开重传列表。鼠标点击各行可在抓包面板看到重传报文。
5. 现在问题来了，怎样定位问题呢？
6. 通过以下方式查看重传来自哪里：

* 在Expert Info窗口一个一个查看报文，在抓包面板查看哪些是重传报文（适合于有经验的用户）
* 在报文面板，配置显示过滤器expert.message == “Retransmission (suspected)”，即可看到抓包文件中所有重传报文
* 应用过滤器，在**Statistics & Conversations**窗口查看**Limit to display filter**部分。

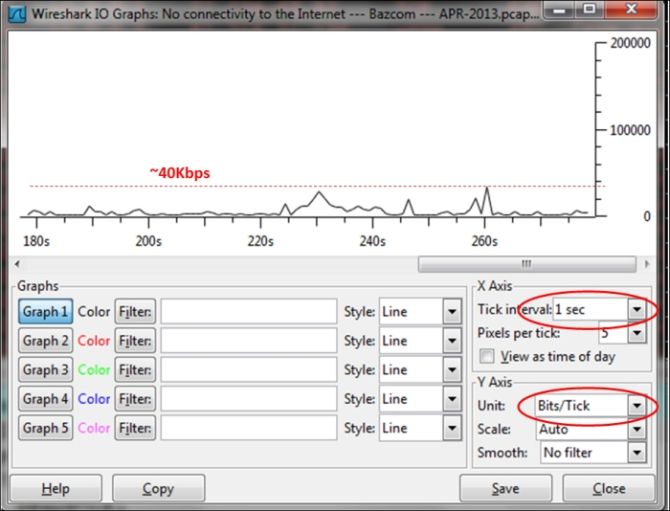
Case 1：重传至多个目的地址

以下截屏中，可看到有多次重传，分布于多台服务器，目的端口号为80（HTTP）。也可以发现重传由端口10.0.0.5发送，因此报文是丢失在发往Internet的途中，或确认信息没有及时从web服务器发回。



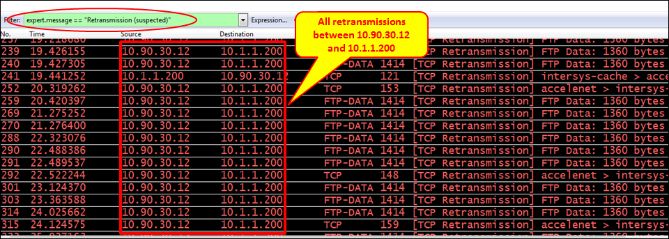
问题发生在发往Internet的线路上，怎样知道是什么问题呢？

1. 从**Statistics**菜单，打开**IO Graph**。
2. 本例中，可看到链路负载非常低。可能是有故障，或有另一条高负载链路。
3. 可以通过登录到通信设备或SNMP浏览器查看引起重传的原因：**报文丢失及错误**。参考以下截屏：



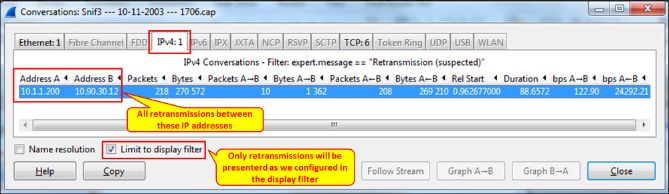
Case 2：重传至单一连接

如果所有重传发生于同一IP，同一TCP端口号，则可能是**慢速应用引起**。看以下截屏：

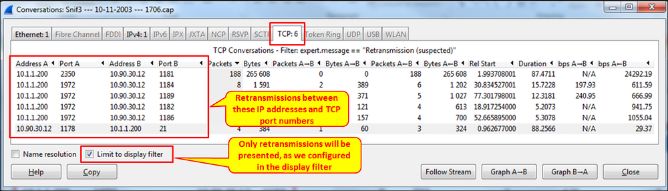


对于单一连接的重传，进行以下操作：

1. 从Statistics菜单打开Conversations，选择Limit to display filter，可以看到所有发生重传的会话，本情况下，是一个单一会话。
2. 如下图所示，通过选择**IPv4**标签可看到从哪个IP地址重传：



   3.  如下图所示，通过选择TCP标签看到重传来自哪一端口：

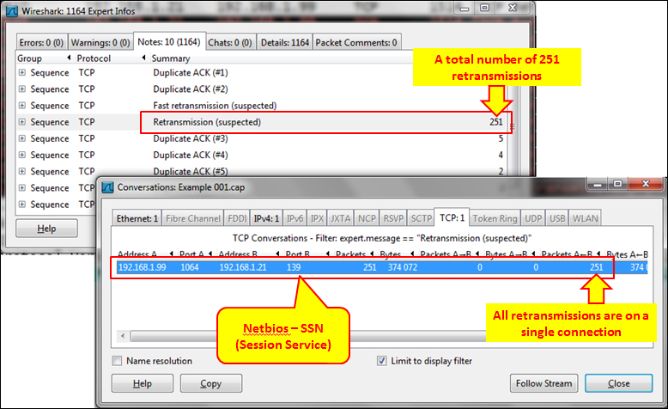


要定位问题，进行以下步骤：

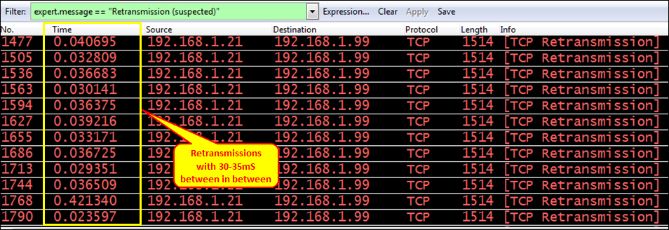
1. 查看IO graph，确保链路不忙。（链路忙的表征例如流量接近带宽。例如，带宽为10Mbps，在IO graph中看见流量接近10Mbps，这就表明链路负载较高。不忙的链路IO会有很多高低起落，峰值以及空闲间隙）。
2. 如果链路不忙，则可能是服务器对于IP地址10.1.1.200有问题（10.90.30.12发送了绝大多数重传，所以可能是10.1.1.200响应较慢）
3. 从报文面板可以看出应用是FTP数据。有可能FTP服务器工作于active模式。因此在端口2350打开连接，服务器将端口更改为1972，所以可能是慢速FTP软件响应问题引起的重传。

Case 3：重传模式

观察TCP重传的一个重要考量是是否能看出一些重传模式。在以下截屏中，可以看见所有重传来自单一连接，位于客户端与服务器的NetBIOS会话服务（TCP端口139）。



看起来像一个简单的服务器/客户端问题，但查看抓包面板，如下图所示：



可以看见重传总是周期性的每30ms发生一次。问题是由于客户端在软件中执行了财务进程，导致软件每30-36ms就减速一次。

Case 4：应用无响应导致重传

另一个可能导致重传的原因是客户端或服务器没有响应请求。这种情况下，会看到五次重传，时间也会逐渐延长。五次连续重传后，发送方认为连接断开（某些情况下，会发送reset来关闭连接，取决于软件实施）。断开连接之后，可能会发生两件事情：

* 发送SYN请求至客户端，以打开一个新的连接。这种情况下用户会看到应用冻结，过了15-20秒之后重新开始工作。
* 不发送SYN，用户需要重新运行应用程序（或应用程序的一部分）

下图显示了打开新连接的情况：



Case 5：由于延时变化导致重传

TCP能够充分容忍延时，前提是延时大小不发生变化。当延时改变时，就会发生重传。诊断是否由该原因引起的方法：

1. 第一件事是ping目的地址，并且得到检查通信链路延时的第一条信息。
2. 检查延时变量，可能由以下原因引起：

* 由于**不稳定或繁忙通信链路**引起。这种情况下，可以看到ping命令的延时变化，通常由于带宽较窄。
* 由于**应用过载或资源不足**，这种情况下，只有该应用发生很多重传。
* **通信设备过载（CPU,缓存）**引起延时。检查方式直接连接通信设备。

   3.  使用Wireshark工具诊断延时问题。

如果重传达到0.5个百分比，性能就会下降，断开连接将会达到5个百分比。这取决于应用及其对于重传的敏感性。

定位重传问题

当你看到通信链路上发生重传，进行以下步骤：

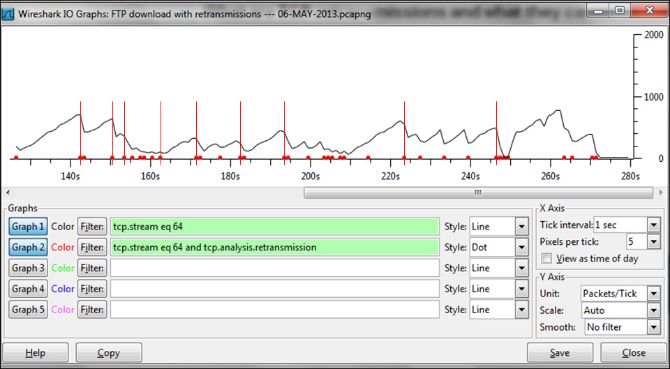
1. 定位问题——是一个特定IP地址，特定连接，特定应用，还是其他问题。
2. 查看问题是否由于通信链路，丢包，慢速服务器还是PC。查看应用是否慢速。
3. 如果不是由于上述原因，检查延时变化。

**工作原理:**

那么重传是由什么原因引起呢？当报文确认信息丢失，或ACK没有及时到达，发送方会进行以下两步操作：

1. 再次发送报文
2. 减少吞吐量。

以下图中展示了重传减少发送方吞吐量（红色细线）：



**四、Wireshark抓包实例分析TCP重复ACK与乱序**

TCP的一大常见问题在于重复ACK与快速重传。这一现象的发生也是由于性能问题，本章讨论如何发现这一问题以及他们意味着什么。

另一个常见问题是前一片段丢失以及乱序片段。某些情况下，这一现象喻示着故障发生，可能是由于网络问题或是抓包中断。

**重复ACK与快速重传:**

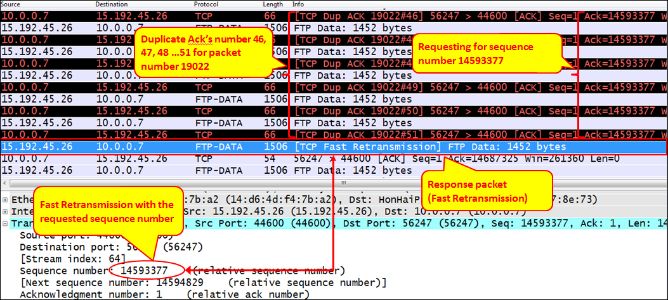
当网速变慢时，重复ACK是可能的原因之一。大多数情况下，重复ACK的发生是由于高延时，延迟的变化，或无法响应ACK请求的慢速终端。

1. 当重复ACK的数量保持在合理范围时，即1或2个百分比，则可能不是本机问题。
2. 当有大量的重复ACK时（假设有10个），则可能：

* 通信链路繁忙引起延迟改变
* 服务器或客户端无响应

   3.  快速重传是对重复ACK的响应报文。

   4. 下图是该问题的示例。本例中51个重复ACK之后发生了快速重传：



    5. 以下是如何解决该问题：

* 如果重复ACK和重传数量较少（少于1个百分比），是可以接受的。
* 如果重复ACK发生在无线网络环境，或是Internet之上的连接，延时或是延时的改变对于这类网络来说很常见，所以也没有什么可做的。
* 如果发生在组织内的网络，则可能有问题。如果发生在LAN之上，检查严重的问题，例如缓存和CPU负载，慢速服务器，等等。如果发生在WAN之上，查看延时，负载以及线路不稳定。

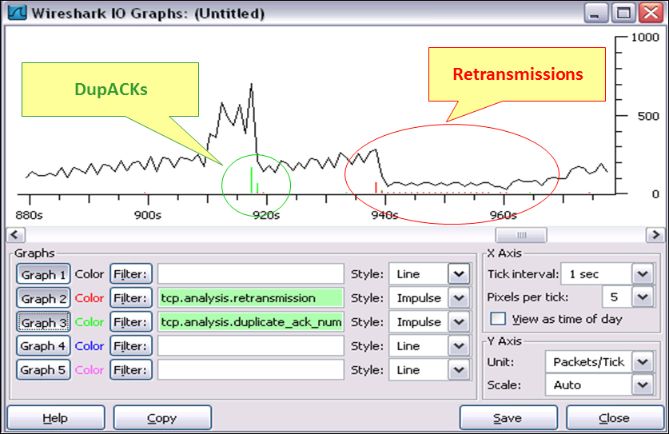
工作原理

当发现有丢失报文时（期望的序列号没有收到），或者收到了预期之外的序列号。这种情况下，接收端生成一个ACK，声明自己希望收到的下一个序列号。接收方持续生成丢失片段的ACK请求，直到实际收到。

在发送方，当它收到三个相同的ACK（初始ACK和两个重复ACK），就会假设有报文丢失并重传该报文，无论重传计时器是否过期。再次发送的报文称为快速重传。

重复ACK也减少了发往网络的吞吐量。减少了多少吞吐量取决于TCP版本。比较早期的TCP版本中出现了重复ACK，发送方将吞吐量减少为之前的一半。在多个DupACK的情况下，吞吐量减到最小。

下图显示了重复ACK和重传的典型例子，本图中第一次重复ACK将吞吐量降低至大约40%，之后重传将吞吐量减至最小。



**乱序报文:**

在两端抓包，乱序情况下需要关注三种现象：

* **先前片段丢失**：当前收到报文的序列号高于该连接的下一个期望序列号时，表明之前的一个或多个报文未能到达
* **乱序报文**：当前报文的序列号低于该连接先前收到的报文
* **先前片段未能捕捉**：（Wireshark 1.8.x及以上版本）：同先前报文丢失。

何时发生？

用户可能在以下情况看到乱序报文：

* **连接开始时抓包**：当建立连接时抓包，这时，看到连接上的报文没有SYN/SYN-ACK/ACK，因此，Wireshark认为连接有问题。

* **确实有报文丢失**：这时会看到丢失报文重传和/或重复ACK告知发送方重传丢失报文。



上图是报文丢失的典型示例。从图中可见，10.0.0.6尝试浏览站点62.90.90.210。这一过程中， TCP片段每个1420字节发送到web服务器，334到336之间3个报文丢失，338到340之间2个报文丢失。两者Wireshark都有提示：TCP’s previous segment is not captured.

* **延时变化**：这可能是由于报文从源地址到目的地址经由不同的路由。检查这一点可以使用Tracert，在源和目的地址之间查找路由改变。如果在公司内部网络上是可以做到的，例如，在路由器上配置trap。

* **数据捕捉问题**：可能报文正常收发，但Wireshark没有捕捉到。可能有以下几种原因：
  + 数据量比较大时，Wireshark在高比特率的情况下可能会丢失报文（高于150-180 Mbps）。要避免这一问题，使用其他工具（大多数需要付费）。
  + 台式机不够强大，内存或CPU无法让Wireshark工作的足够快。这一点很好发现。
  + 当LAN交换机的端口缓存太小，报文可能被丢弃。连接到交换机（用控制台或telnet连接）使用交换机命令行来检查该问题。
  + 无线网络抓包，由于某种原因没有看到所有发送报文。

总结

乱序报文的原理很简单。TCP发送以其字节数为编号的报文到接收方。当一个报文没有按照顺序到达时，Wireshark就会注意到。原因有两点：

* **确实有问题**：这时会看到重传和重复ACK，这是TCP对于收到乱序报文的响应。
* **抓包问题**：这时仅看到乱序报文，但没有看到对可能丢失及乱序报文的响应，可能实际上并没有问题。

**五、Wireshark抓包实例分析TCP窗口及reset**

 TCP最重要的机制之一是滑动窗口机制，以及用以控制TCP终端节点愿意接收的数据总量的流控机制。

TCP reset可以在几种情况下被发送。有一些是协议的正常工作过程，有一些则表明可能有问题。本节中，我们查找问题以及分析解决问题的方法。

本章讨论以上两个问题。

**TCP窗口问题:**

TCP零窗口，零窗口探测，零窗口违例

TCP零窗口发生于接收方在TCP头部的window字段广播接收窗口零字节的时候。这一事件告知发送方停止发送数据，因为接收方缓存已满。这也表明接收方可能发生以下问题：

* 服务器无法为进程分配组后的内存
* 应用碰到没有足够内存的问题，因此TCP需告知发送方停止发送数据
* 应用消耗太多内存因此操作系统要限制应用资源

TCP零窗口探测由发送方发出，以查看接收方的零窗口是否依然存在。这一消息通过发送下一字节数据给接收方，如果接收方回复窗口大小仍然为零，则发送方的探测计时器加倍。

TCP窗口违例：发送方忽略接收方的零窗口大小并发送额外字节数据。TCP零窗口违例表明协议栈中有TCP错误。为了检查是何问题，检查是否有以下事件：

* 某一终端设备（服务器或客户机）报出终端设备故障
* 某一应用报出常规应用错误
* 应用中执行某一操作时报错，例如，打开一个表格，发送一份文件至打印机，创建一个报告，或其他操作。这种情况下，是应用问题。

TCP窗口更新

TCP将窗口更新发送至连接的对端，以表明缓存大小更改，并且准备好接受更高或更低的数据速率（缓存大小决定了发送方被允许的发送速率）。这一情况发生于：

* TCP接收方从零窗口中恢复，告知发送方重新发送速率。这一情况下，无需进行处理，只需检查第一次导致零窗口的问题。
* TCP接收方频繁更改窗口大小。该情况下检查接收方被干扰的原因。可能是应用问题，内存问题，或终端设备上的其他问题。

看到这类现象无需担心，这就是TCP的工作机制。

TCP窗口已满

这一信息表明已发送的报文会完全填满接收方的接收缓存。发生于接收方没有对先前接收到的数据发出任何ACK确认信息，因此，这将会成为发送方从接收方收到ACK之前的最后一个报文数据。

这一事件的触发原因与触发零窗口的原因相同，是服务器或应用无响应的标志。典型实例如下图所示：



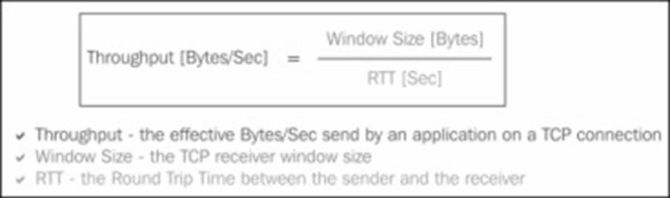
上图中可以看到：

1. 报文183816，192.168.2.138告知192.168.1.58发送窗口已满。
2. 下一个报文，192.168.1.58发送一个信号至192.168.2.138，告知对方停止发送数据。这是一个零窗口信号。
3. 双方继续发送零窗口以及零窗口探测。
4. 连接的最后一个报文是192.168.2.138发送的RST报文，目的是断开连接。
5. 某些情况下零窗口可以通过窗口更改信息来回复。某些情况下可通过reset来关闭（可以是应用零窗口从而没有收到任何数据导致）。

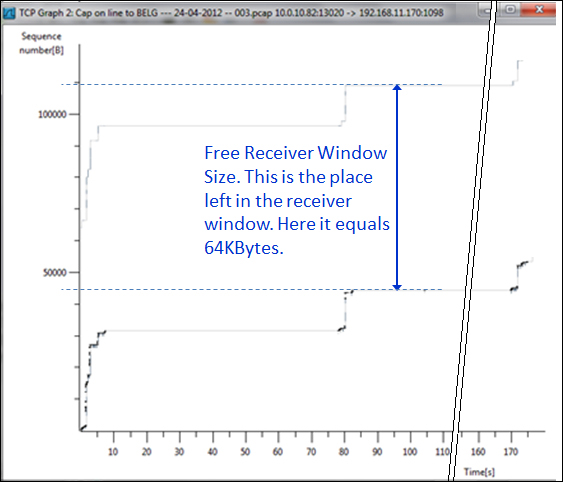
工作原理

TCP滑动窗口机制如下：

1. 连接建立之后，发送方将数据发送至接收方，填入接收窗口。
2. 若干报文之后，接收方发送ACK至发送方，确认接收到其发送的字节数。发送ACK将接收窗口清零。
3. 这一过程持续下去，发送方向窗口中填入数据，接收方清空并发送确认信息。
4. 扩大接收窗口大小告知发送方增加吞吐量，减小窗口告知对方减小吞吐量。这一机制按照WS/RTT规则（随着TCP版本不同而有所改变）：



也可以通过TCP吞吐量图表和IO图来查看问题。在TCP吞吐量图表中，使用TCP trace图表，上面一行显示了窗口大小，与下面一行的距离表明窗口的剩余大小。没有距离表示零窗口。



两行之间的固定距离表明接收方工作良好。当两行渐渐靠近，表明发送方速度高于接收方。只要这两行没有重叠，TCP就会继续发送数据。

**TCP reset及原因:**

在可疑的链路或服务器两端连接Wireshark，开始抓包。观察抓包窗口的每一个窗口信息。TCP reset可以在几种情况下被发送。有一些是协议的正常工作过程，有一些则表明可能有问题。本节中，我们查找问题以及分析解决问题的方法。

reset是用以告知接收方断开连接的TCP信号，通过将RST标志位置1来发送。正常的操作过程中，TCP通过SYN信号打开连接，通过FIN信号关闭连接。TCP的一大特性在于有问题时，或只是为了更好的性能，它能够快速关闭连接。

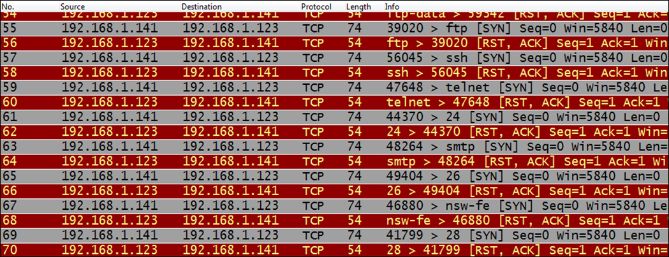
无故障时发送reset

TCP关闭连接的标准方式是通过FIN和FIN-ACK信号。为了关闭连接，用户需要四个报文：来自一方的FIN/ACK和ACK，以及另一方的同样报文。当你打开一个网页，可能同时打开了数十个连接（主页，新闻，广告，定期更新的图片等），要关闭所有这些有时需要数百个FIN和FIN-ACK报文。为了防止其发生，web服务器在很多情况下会在发送请求数据之后用reset断开连接。这是标准的做法，并取决于应用程序。

有故障时发送reset

某些情况下reset表明有故障发生（并不一定是通信故障）：

* **防火墙发送的reset**：当远端服务器尝试打开连接但没有结果时，也许会看到返回RST信号。这是防火墙阻隔连接的情况。下图中，可看到发送的每一个SYN都返回以RST。



* **由于收发一方有问题发送的reset**：可能的原因如：
  + 五个连续没有收到ACK回复的重传。当发送方没有收到任何重传回复，它就会发送一个reset信号到对端，告知其断开连接。
  + 另一个原因是连接之上几分钟都没有任何数据（分钟数取决于系统默认）。打开连接的一方通常会发送reset（但并不总是会这样做，取决于实现方式）。