实验练习-协议层

**目标：**

了解如何在数据包中表示协议和分层。它们是构建文本§1.3和§1.4所述网络的关键概念。在做实验之前复习这些部分。

**要求：**

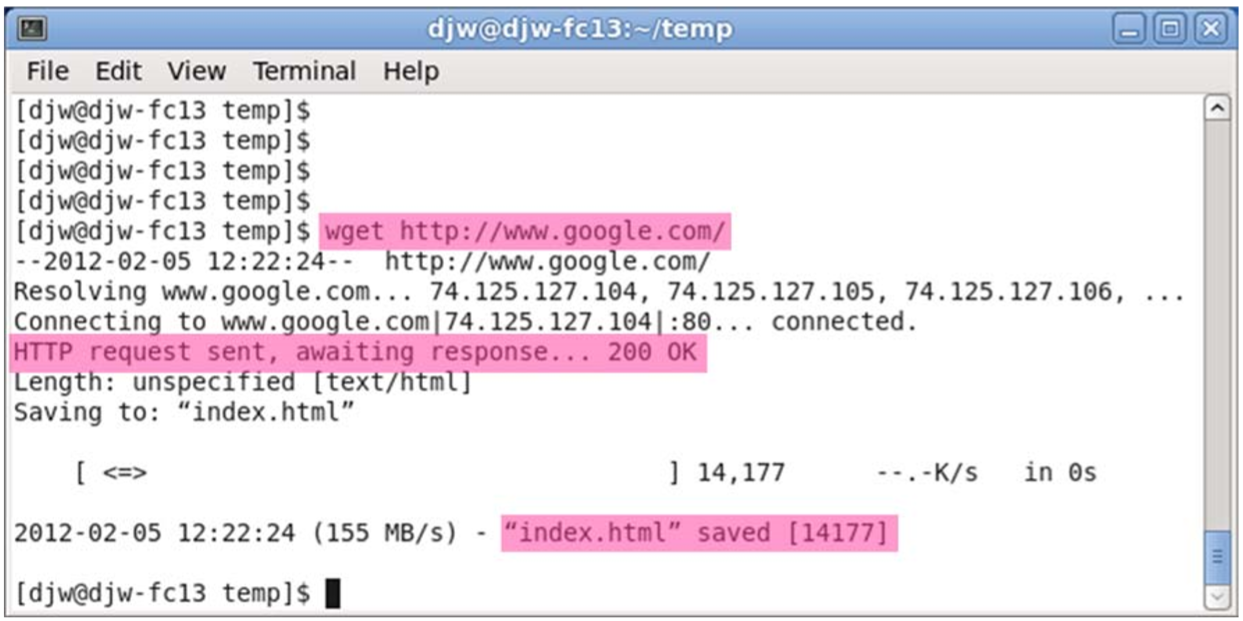
Wireshark：该实验使用Wireshark软件工具捕获和检查数据包跟踪。数据包跟踪是网络上某个位置的通信量记录，就好像对通过特定线路的所有比特进行了快照一样。数据包跟踪记录每个数据包的时间戳，以及组成数据包的位，从低层头到高层内容。Wireshark在大多数操作系统上运行，包括Windows、Mac和Linux。它提供了一个图形用户界面，显示了数据包的序列，以及当解释为协议头和数据时位的含义。它根据数据包的类型对数据包进行颜色编码，并有各种方法来过滤和分析数据包，以便您调查网络协议的行为。Wireshark广泛用于网络故障排除。你可以从www.wireshark.org，如果您的计算机上尚未安装。我们强烈建议您观看网站上5分钟的短片“Wireshark简介”。

wget/curl：这个实验使用wget（Linux和Windows）和curl（Mac）获取web资源。wget和curl是允许获取URL的命令行程序。与获取并执行整个页面的web浏览器不同，wget和curl让您可以精确控制获取哪些URL以及何时获取它们。在Linux下，可以通过软件包管理器安装wget。在Windows下，wget是二进制的；请在上查找下载信息http://www.gnu.org/software/wget/.在Mac下，curl随操作系统一起安装。两者都有很多选项（请尝试“wget--help”或“curl--help”查看），但只需使用“wget-URL”或“curl-URL”即可获取URL。

**第一步：捕捉跟踪**

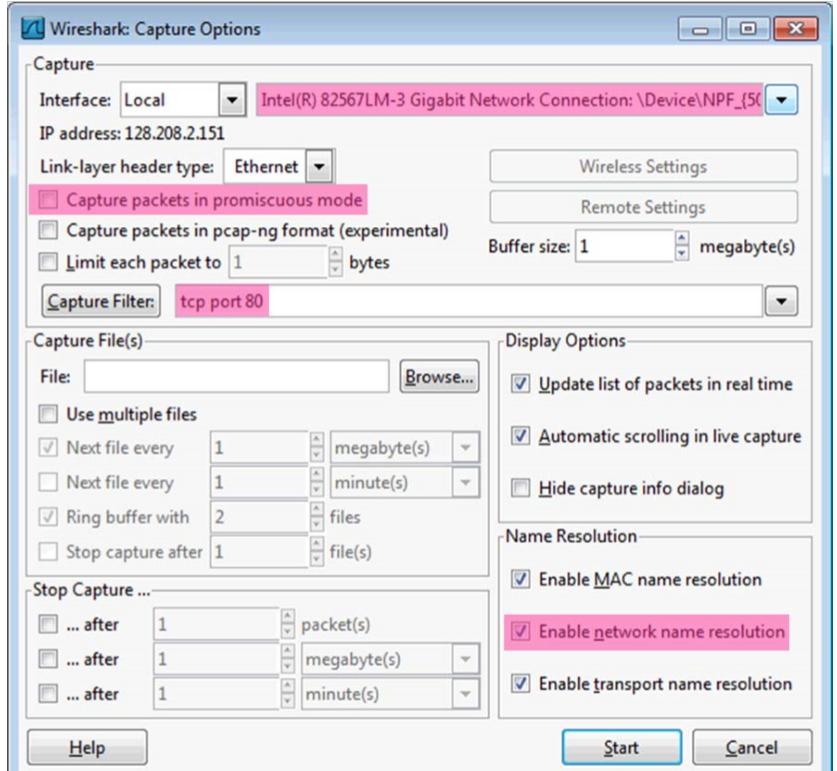
按照以下步骤捕获网络流量的跟踪；或者，您可以使用提供的跟踪。我们希望这个跟踪可以查看数据包的协议结构。从您选择的服务器到您的计算机（即客户机）的简单Web获取URL将用作流量。

1. 选择一个URL并用wget或curl获取它。例如，“wget”http://www.google.com或“curl http://www.google.com”. 这将获取资源并将其写入文件（wget）或屏幕（curl）。您正在检查fetch是否工作并检索一些内容。下面显示了wget的一个成功示例（添加了突出显示）。您需要一个状态代码为“200 OK”的单一响应。如果抓取不起作用，那么尝试其他URL；如果没有URL，那么调试wget/curl的使用或Internet连接。



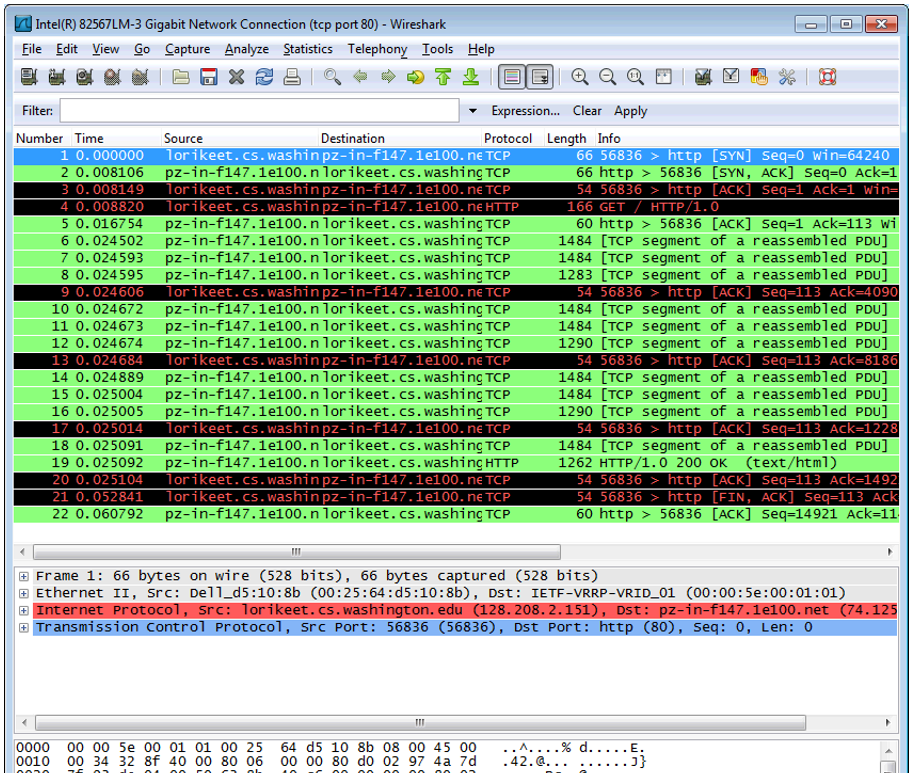
2.关闭不必要的浏览器选项卡和窗口。通过最小化浏览器活动，您将阻止计算机获取不必要的web内容，并避免跟踪中的意外流量。

3.启动Wireshark，使用“tcp port 80”过滤器启动捕获，并选中“启用网络名称解析”。这个过滤器只会记录标准的网络流量，而不会记录计算机可能发送的其他类型的数据包。该检查将把发送和接收数据包的计算机的地址转换为名称，这将帮助您识别数据包是发送到您的计算机还是来自您的计算机。除了突出显示之外，您的捕获窗口应该与下图类似。选择要从中捕获的接口作为计算机用于连接到Internet的主要有线或无线接口。如果不确定，请猜测，如果捕获不成功，请稍后重新执行此步骤。取消选中“在混乱模式下捕获数据包”。此模式有助于偷听广播网络上其他计算机发送的数据包。我们只想记录发送到/从您的计算机发送的数据包。将其他选项保留为默认值。捕获筛选器（如果存在）用于防止捕获计算机可能发送或接收的其他流量。在Wireshark 1.8上，捕获过滤器框直接出现在选项屏幕上，但在Wireshark 1.9上，可以通过双击界面来设置捕获过滤器。



4.当捕获开始时，使用上面的wget/curl重复web获取。这一次，Wireshark将在内容传输时记录数据包。

5.提取成功后，返回Wireshark并使用菜单或按钮停止跟踪。如果成功，上方的Wireshark窗口将显示多个数据包，很可能数据包已满。捕获多少数据包取决于网页的大小，但跟踪中至少应有8个数据包，通常为20-100个，其中许多数据包将被涂成绿色。下面是一个例子。恭喜你，你抓到了一条线索！

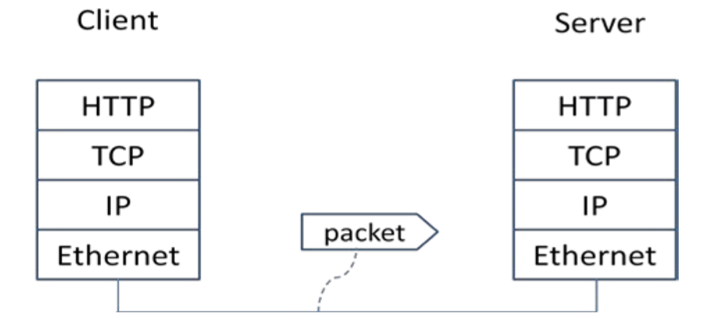


第2步：检查跟踪

Wireshark将允许我们选择一个数据包（从顶部面板）并查看其协议层，包括头部字段（在中间面板）和组成数据包的字节（在底部面板）。在上图中，选择了第一个数据包（以蓝色显示）。请注意，我们在这里使用“数据包”作为通用术语。严格来说，链接层的一个信息单元称为帧。在网络层称为数据包，在传输层称为数据段，在应用层称为消息。Wireshark正在收集帧，并向我们展示它可以识别的帧中承载的更高层数据包、段和消息结构。为了方便起见，我们通常会使用“数据包”，因为每个帧包含一个数据包，而感兴趣的往往是数据包或更高层的细节。

选择一个协议列为“HTTP”且信息列显示为GET的数据包。它是将从计算机发送到服务器的web（HTTP）请求传送到服务器的数据包。（您可以单击列标题按该值排序，但通过检查应该不难找到HTTP数据包。）让我们仔细看看数据包结构如何反映正在使用的协议。

由于我们正在获取一个网页，我们知道所使用的协议层如下所示。也就是说，HTTP是用于获取URL的应用层web协议。与许多互联网应用程序一样，它运行在TCP/IP传输和网络层协议之上。链路和物理层协议取决于您的网络，但如果您的计算机是有线的，则通常以以太网（如图所示）的形式组合，如果您的计算机是无线的，则以802.11（如图所示）的形式组合。



选择HTTP GET数据包后，仔细查看它与我们的协议栈之间的异同，如下所述。协议块列出在中间面板中。可以展开每个块（通过单击“+”扩展器或图标）以查看其详细信息。

•第一个Wireshark块是“帧”。这不是一个协议，而是一个描述数据包整体信息的记录，包括数据包被捕获的时间和长度。

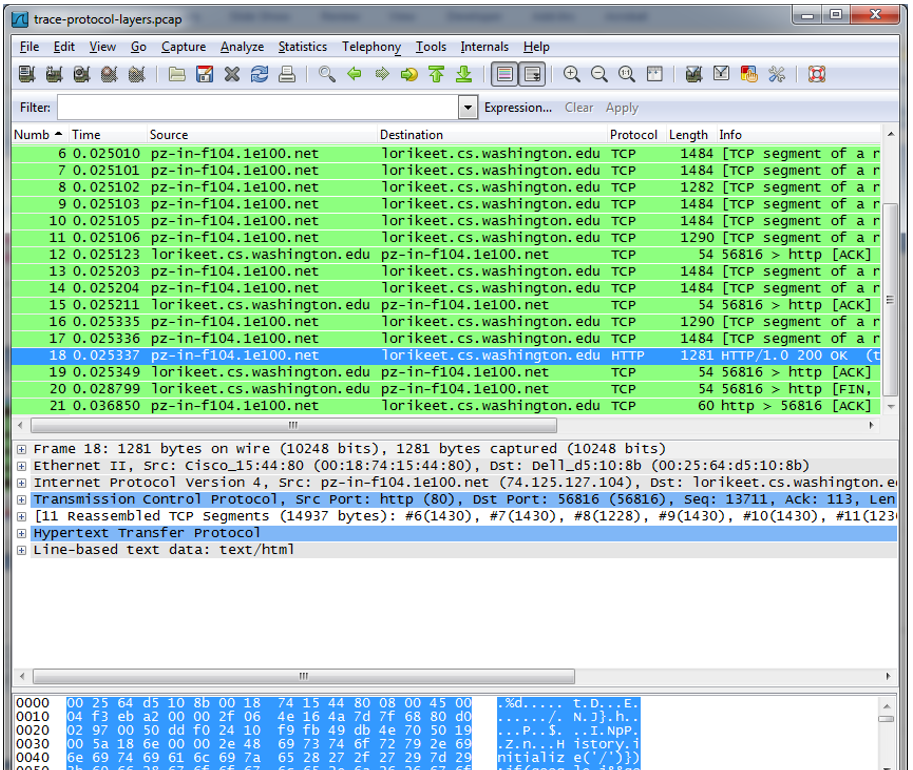
•第二块是“以太网”。这符合我们的图表！请注意，您可能已经在使用802.11的计算机上进行了跟踪，但仍然可以看到以太网块而不是802.11块。为什么？之所以会出现这种情况，是因为我们要求Wireshark在捕获选项中以以太网格式捕获流量，所以它将真实的802.11报头转换为伪以太网报头。

•然后是IP、TCP和HTTP，它们正是我们想要的。请注意，顺序是从协议栈的底部向上。这是因为当数据包沿着堆栈向下传递时，较低层协议的报头信息被添加到较高层协议信息的前面，如文本的图1-15所示。也就是说，较低层的协议首先出现在“在线”数据包中。

现在找到另一个HTTP数据包，即从服务器到您的计算机的响应，并查看此数据包的结构与HTTP GET数据包的差异。此数据包的信息字段中应该有“200 OK”，表示成功获取。在我们的跟踪中，细节面板中有两个额外的块，如下图所示。

•第一个额外块显示“[11 reassembled TCP segments …]”。捕获过程中的细节会有所不同，但此块描述的不仅仅是数据包本身。最有可能的是，网络响应通过网络作为一系列数据包发送，这些数据包在到达计算机后放在一起。标记为HTTP的数据包是web响应中的最后一个数据包，块列出了连接在一起以获得完整web响应的数据包。这些数据包中的每一个都显示为具有TCP协议，即使这些数据包包含HTTP响应的一部分。当可以理解完整的HTTP消息时，只有最后一个数据包显示为具有HTTP协议，并且它列出了连接在一起以做出HTTP响应的数据包。

•第二个额外的块表示“Line-based text data …”。捕获的详细信息会有所不同，但此块描述的是获取的网页的内容。在我们的例子中，它是text/html类型，尽管它可能很容易是text/xml、image/jpeg或许多其他类型。与帧记录一样，这不是真正的协议。相反，它是Wireshark制作的数据包内容描述，帮助我们了解网络流量。



步骤3：数据包结构

为了显示您对数据包结构的理解，请绘制一个HTTP GET数据包的图形，以字节为单位显示TCP、IP和以太网协议头的位置和大小。您的图形可以简单地将整个数据包显示为一个细长的矩形。最左边的元素是第一个通过导线发送的元素。在这张图上，显示以太网报头的范围和IP通过网络传送到以太网以发送的以太网有效负载。要显示协议层的嵌套结构，请注意IP头和IP有效负载的范围。在查看每个协议时，您可能会对其中的字段产生疑问。我们将在未来的实验室中详细探讨这些协议和领域。

为了计算出大小，观察到当你点击中间面板中的一个协议块（块本身，而不是“+”扩展器）时，WiReSARK将突出显示它在下面板中的包对应的字节，并显示窗口底部的长度。例如，在跟踪中点击一个数据包的IP版本4报头，就会显示长度为20字节。（如果是IPv6，您的跟踪将不同，即使是IPv4，也可能不同，具体取决于不同的选项。）您还可以使用“长度”列或“帧详细信息”块中显示的总数据包大小。

上交：把你的包图纸交上来。

第4步：协议开销

估计下载协议开销，或协议开销占用的下载字节的百分比。为此，考虑HTTP数据（报头和消息）是网络要携带的有用数据，而下层报头（TCP、IP和以太网）是开销。我们希望这个开销小一些，以便大多数比特被用来承载应用程序关心的内容。要解决这个问题，首先只查看下载方向上的数据包，以便进行一次web获取。您可以在“目的地”列上进行排序以找到它们。这些数据包应该以一个称为SYN ACK的短TCP数据包开始，这是连接的开始。它们后面的大部分是较长的中间包（大约1到1.5 KB），其中最后一个是HTTP包。这是下载的主要部分。它们很可能以一个短的TCP数据包结束，这是结束连接的一部分。对于每个数据包，您可以检查它以Ethernet/IP/TCP报头的形式有多少开销，以及它在TCP负载中携带了多少有用的HTTP数据。您还可以查看Wireshark中的HTTP数据包，了解所有下载数据包的TCP有效负载中有多少数据。

上交：您对上述下载协议开销的估计。请告诉我们，您是否认为这项开销很重要。

第5步：解复用键

当以太网帧到达计算机时，以太网层必须将其包含的数据包交给下一个更高的层进行处理。找到正确的更高层来处理接收到的数据包的行为被称为解复用。我们知道，在我们的例子中，更高层是IP。但是以太网协议是如何知道这一点的呢？毕竟，更高层可能完全是另一个协议（比如ARP）。我们在IP层也遇到了同样的问题——IP必须能够确定IP消息的内容是TCP数据包，以便它可以将其交给TCP协议进行处理。答案是，协议使用其报头中称为“解复用密钥”的信息来确定更高层。

详细查看下载数据包的以太网和IP报头，回答以下问题：

1.哪个以太网报头字段是解复用密钥，告诉它下一层是IP？该字段中使用什么值来表示“IP”？

2.哪个IP头字段是解复用密钥，告诉它下一个更高层是TCP？该字段中使用什么值来表示“TCP”？

上交：交上以上问题的答案。

独自探索

我们鼓励您在完成本实验后探索协议和分层。一些想法：

•看一个不携带更高层数据的短TCP数据包。这个包裹寄往哪个实体？毕竟，如果它不携带更高层的数据，那么对于HTTP这样的更高层协议来说，它似乎不是很有用！

•在经典的分层模型中，来自更高层的一条消息有一个由较低层附加的头，并成为一条新消息。但情况并非总是如此。在上面，我们看到了一个跟踪，其中web响应（一条由HTTP头和HTTP有效负载组成的HTTP消息）被转换为多个低层消息（即多个TCP数据包）。假设您已经为第一个和最后一个承载web响应的TCP数据包绘制了数据包结构（如步骤2所示）。图纸有什么不同？

•在上面描述的经典分层模型中，较低层将头附加到从较高层传递下来的消息中。如果较低的层添加加密，这种模式会发生什么变化？

•在上面描述的经典分层模型中，较低层将头附加到从较高层传递下来的消息中。如果较低的层增加了压缩，这个模型会发生什么变化？