第二章 应用层

1. **对于想快速处理从远程客户到服务器的事务，应该使用UDP还是TCP？为什么？**

**解答：**

应该使用UDP。

使用UDP，事务可以在一次往返时间(RTT)内完成——客户端将事务请求发送到UDP套接字，服务器将响应发送回客户端UDP套接字。对于TCP，至少需要两个RTT—一个用于设置TCP连接，另一个用于客户机发送请求，服务器发送回响应。

1. **假定在浏览器上点击一个URL，但这个URL的IP地址以前并没有缓存在本地主机上。因此需要用DNS自动查找和解析。假定要解析到所要找的URL的IP地址共经过n个DNS服务器，所经过的时间分别是RTT1, RTT2,……RTTn。假定从要找的网页上只需要读取一个很小的图片（即忽略这个小图片的传输时间）。从本地主机到这个网页的往返时间是RTTw。试问从点击这个URL开始，一直到本地主机的屏幕上出现所读取的小图片，一共需要经过多少时间？**

**解答：**

解析IP地址需要时间是：RTT1+RTT2+…+RTTn。

建立TCP连接和请求并接收网页需要2RTTw。

1. **上题中，假定同一台服务器的这个HTML文件中又链接了三个非常小的对象。若忽略这些对象的发送时间，试计算客户点击读取这些对象所需的时间。**

（1）没有并行TCP连接的非持续HTTP；

（2）使用并行TCP连接的非持续HTTP；

（3）流水线方式的持续HTTP。

**解答：**

总共是四个对象，每个对象建立TCP连接和请求并接收需要2RTTw。

（1）所需时间 = RTT1+RTT2+…+RTTn+8RTTw。（4 \* 2RTTw）

（2）所需时间 = RTT1+RTT2+…+RTTn+4RTTw。

（发起TCP连接需要RTTw时间，请求并收到HTML文件需要RTTw时间，因为采用非持续的连接，服务器在发送该HTML文件后关闭连接；客户端分析该HTML文件，得到3个引用对象，再次发起TCP连接，需要RTTw时间，然后并行请求3个对象直到收到三个对象，需要RTTw时间）

（3）所需时间 = RTT1+RTT2+…+RTTn+3RTTw。

（发起TCP连接，RTTw

请求并收到HTML文件，RTTw

再次请求3个对象并收到三个对象，RTTw）

1. **考虑当浏览器发送一个HTTP GET报文时，通过Wireshark俘获到下列ASCII字符串（一个HTTP GET报文的实际内容）。字符<cr><lf>是回车和换行符。**

*GET /cs453/index.html HTTP/1.1<cr><lf>Host: gaia.cs.umass.edu<cr><lf>User-Agent: Mozilla/5.0 (Windows;U; Windows NT 5.1; en-US; rv:1.7.2) Gecko/20040804 Netscape/7.2 (ax) <cr><lf>Accept:ext/xml, application/xml, application/xhtml+xml, text/html;q=0.9, text/plain;q=0.8, image/png,\*/\*;q=0.5<cr><lf>Accept-Language: en-us, en;q=0.5<cr><lf>Accept-Encoding: zip, deflate<cr><lf>Accept-Charset: ISO-8859-1,utf-8;q=0.7,\*;q=0.7<cr><lf>Keep-Alive: 300 <cr><lf> Connection:keep-alive <cr><lf><cr><lf>*

回答下列问题：

1. 由浏览器请求的文档的URL是什么？
2. 该浏览器运行的是HTTP的何种版本？
3. 该浏览器请求的是一条非持续连接还是持续连接？
4. 该浏览器所运行的主机的IP地址是什么？
5. 发起报文的浏览器的类型是什么？在一个HTTP请求报文中，为什么需要浏览器类型？

**解答：**

1. 由浏览器请求的文档的URL是http://gaia.cs.umass.edu/cs453/index.html. Host:字段表示服务器的名称，/cs453/index.html表示文件名。
2. 运行的是HTTP version 1.1. 如第一对<cr><lf>之前所示。
3. 浏览器正在请求一条持续连接，如Connection:keep-alive.
4. 这是个刁钻的问题。HTTP消息中不包含此信息。因此，仅从HTTP消息的交换就无法看出这一点。要回答这个问题，需要来自IP数据报(包含HTTP GET请求的TCP段)的信息。
5. Mozilla/5.0. 服务器需要浏览器类型信息来将相同对象的不同版本发送到不同类型的浏览器。
6. **下面显示的是来自服务器的回答，以响应上述问题中HTTP GET请求报文。**

*HTTP/1.1 200 OK<cr><lf>Date: Tue, 07 Mar 2008 12:39:45GMT<cr><lf>Server: Apache/2.0.52 (Fedora)<cr><lf>Last-Modified: Sat, 10 Dec2005 18:27:46 GMT<cr><lf>ETag: ”526c3-f22-a88a4c80”<cr><lf>Accept-Ranges: bytes<cr><lf>Content-Length: 3874<cr><lf>Keep-Alive: timeout=max=100<cr><lf>Connection:Keep-Alive<cr><lf>Content-Type: text/html; charset=ISO-8859-1<cr><lf><cr><lf><!doctype html public ”-//w3c//dtd html 4.0 transitional//en”><lf><html><lf><head><lf> <meta http-equiv=”Content-Type” content=”text/html; charset=iso-8859-1”><lf> <meta name=”GENERATOR” content=”Mozilla/4.79 [en] (Windows NT 5.0; U) Netscape]”><lf> <title>CMPSCI 453 / 591 /NTU-ST550ASpring 2005 homepage</title><lf></head><lf><much more document text following here (not shown)>*

回答下列问题：

1. 服务器能否成功地找到那个文档？该文档提供回答是什么时间？
2. 该文档最后修改是什么时间？
3. 文档中被返回的字节有多少？
4. 文档被返回的前5个字节是什么？该服务器同意一条持续连接吗？

**解答：**

1. 状态码为200，短语OK表示服务器能够成功定位文档。文档提供回答是Tuesday, 07 Mar 2008 12:39:45 Greenwich Mean Time.
2. 该文档最后修改时间是Saturday 10 Dec 2005 18:27:46 GMT.
3. 文档中被返回的字节有3874 bytes.
4. 文档被返回的前5个字节是: <!doc. 该服务器同意一条持续连接，如 Connection: Keep-Alive所示。
5. **假设域名为m.a.com的主机，由于重启动的原因两次向本地DNS服务器dns.a.com查询域名为www.abc.net的IP地址。请说明每一次域名转换的过程。**

**解答：**

对主机m.a.com的第一次请求，域名转换过程如下：

（1）主机m.a.com作为客户机向本地域名服务器发出DNS查询报文，查询域名为www.abc.net的主机的IP地址。

（2）由于本地域名服务器所管理的缓存中没有该域名，本地域名服务器以客户身份继续向顶级域名服务器dns.net查询。

（3）顶级域名服务器根据被查询域名中的abc，将下属的权威域名服务器dns.abc.net的IP地址返回给本地域名服务器dns.a.com。然后，本地域名服务器直接向权威域名服务器dns.abc.net进行查询。

（4）权威域名服务器dns.abc.net将所查到的IP地址返回给本地域名服务器dns.a.com。

（5）本地域名服务器dns.a.com将查到的IP地址告知主机m.a.com，同时在本地域名服务器中缓存下来。

对主机m.a.com的第二次请求，域名转换过程如下：

（1）主机m.a.com向本地域名服务器dns.a.com提出对www.abc.net的域名解析请求。

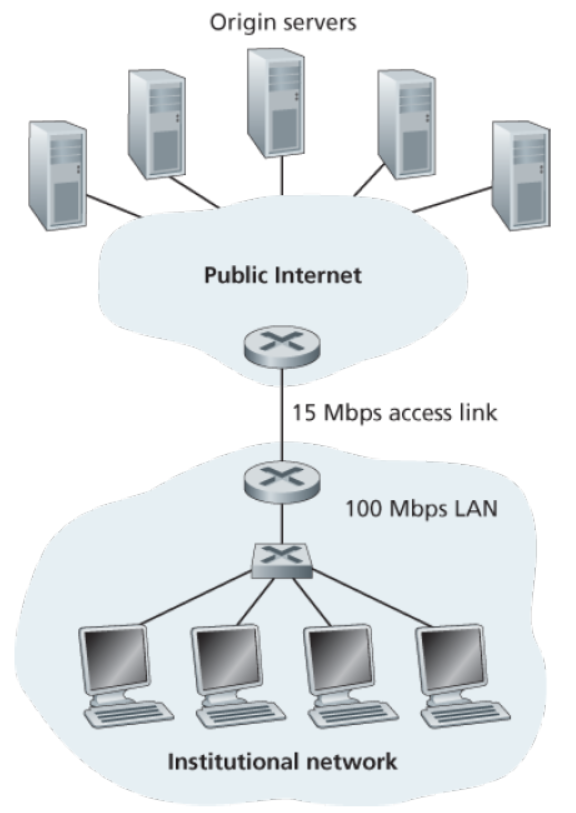
（2）本地域名服务器从缓存中发现有该域名的记录。

（3）本地域名服务器取出该记录，并将对应的IP地址告知主机m.a.com。

1. **如下图，其中只有一个机构的网络和因特网相连。假设对象的平均长度为 850000 比特，从这个机构网的浏览器到初始服务器的平均请求率是每秒 16 个请求。还假定从接入链路的因特网一侧的路由器转发一个 HTTP 请求开始，到接收到其响应的平均时间是 3 秒。将总的平均响应时间建模为平均接入时延（即从因特网路由器到机构路由器的时延）和平均因特网时延之和。对于平均接入时延，使用 Δ/(1 - Δβ)，式中 Δ 是跨越接入链路发送一个对象的平均时间，β 是对象对该接入链路的平均到达率。**

**（a）求出总的平均响应时间。**

**（b）现在假定在这个机构 LAN 中安装了一个缓存器。假定命中率为 0.4，求出总的响应时间。**



**解答：**

1. 若通过链路的速率R，传输大小为L的对象的时间为L/R。平均时间是对象的平均大小除以R：

Δ = (850,000 bits)/(15,000,000 bits/sec) = 0.0567 sec

该链接的流量强度由 β 给出，Δ = (16 requests/sec)(0.0567 sec/request) = 0.907。因此，平均访问延迟为(0.0567 sec)/(1 - 0.907) ≈ 0.6 seconds。因此，总平均响应时间为0.6秒+ 3秒= 3.6秒。

1. 由于60%的请求在机构网络内得到满足，因此访问链路上的流量强度降低了60%。因此，平均访问延迟为(0.0567秒)/[1 -(0.4)(0.907)]= 0.089秒。如果缓存满足请求(概率为0.6)，响应时间近似为零; 缓存失败的平均响应时间是0.089秒+ 3秒= 3.089秒(这种情况发生的概率是40%)。所以平均响应时间是(0.6)(0秒)+(0.4)(3.089秒)= 1.24秒。因此，平均响应时间从3.6秒减少到1.24秒。
2. **在一台主机上安装编译TCPClient和UDPClient Python程序，在另一台主机上安装编译TCPServer和TCPClient程序。**
3. 假设在运行TCPServer之前运行了TCPClient，将发生什么现象？为什么？
4. 假设在运行UDPServer之前运行了UDPClient，将发生什么现象？为什么？
5. 如果对客户端和服务器端使用了不同的端口，将发生什么现象？

**解答：**

1. 如果先运行TCPClient，那么客户机将尝试与不存在的服务器进程建立TCP连接。故不会建立正确的TCP连接。
2. UDPClient没有与服务器建立TCP连接。因此，应该首先运行UDPClient，然后运行UDPServer，然后在键盘上输入一些内容。
3. 如果使用不同的端口号，那么客户端将尝试使用错误的进程或不存在的进程建立TCP连接。会发生错误。
4. **假定在UDPClient.py中创建套接字后增加了下面一行：**

clientSocket.bind(‘’, 5432)

有必要修改UDPServer.py吗？UDPClient和UDPServer中的套接字端口号是多少？在变化之前它们是多少？

**解答：**

在原程序中，UDPClient在创建套接字时不指定端口号。在本例中，代码允许底层操作系统选择端口号。增加附加行，当UDPClient执行时，将创建一个UDP套接字，端口号为5432。

UDPServer需要知道客户机端口号，以便能够将数据包发送回正确的客户机套接字。注意看UDPServer，我们看到客户机端口号没有“hard-wired”到服务器代码中;相反，UDPServer通过解析从客户机接收到的数据报来确定客户机端口号。因此UDP服务器将工作与任何客户端端口号，包括5432。因此不需要修改UDPServer。

在变化之前，Client socket = x (chosen by OS)，Server socket = 9876

在变化之后，Client socket = 5432。