器发送响应。如果浏览器请求一个在该服务器中不存在的文件，服务器应当返回一个“404 Not Found”差错报文。

在配套网站中，我们提供了用于该服务器的框架代码。你的任务是完善该代码，运行服务器，通过在不同主机上运行的浏览器发送请求来测试该服务器。如果运行你服务器的主机上已经有一个 Web 服务器在运行，你应当为该 Web 服务器使用一个不同于 80 端口的其他端口。

作业 2：UDP ping 程序

在这个编程作业中，你将用 Python 编写一个客户 ping 程序。该客户将发送一个简单的 ping 报文，接收一个从服务器返回的对应 pong 报文，并确定从该客户发送 ping 报文到接收到 pong 报文为止的时延。该时延称为往返时延（RTT）。由该客户和服务器提供的功能类似于在现代操作系统中可用的标准 ping 程序。然而，标准的 ping 使用互联网控制报文协议（ICMP）（我们将在第 4 章中学习 ICMP）。此时我们将创建一个非标准（但简单）的基于 UDP 的 ping 程序。

你的 ping 程序经 UDP 向目标服务器发送 10 个 ping 报文。对于每个报文，当对应的 pong 报文返回时，你的客户要确定和打印 RTT。因为 UDP 是一个不可靠的协议，由客户发送的分组可能会丢失。为此，客户不能无限期地等待对 ping 报文的回答。客户等待服务器回答的时间至多为 1 秒；如果没有收到回答，客户假定该分组丢失并相应地打印一条报文。

在此作业中，将给出服务器的完整代码（在配套网站中可找到）。你的任务是编写客户代码，该代码与服务器代码非常类似。建议你先仔细学习服务器的代码，然后编写你的客户代码，可以不受限制地从服务器代码中剪贴代码行。

作业 3：邮件客户

这个编程作业的目的是创建一个向任何接收方发送电子邮件的简单邮件客户。你的客户将必须与邮件服务器（如谷歌的电子邮件服务器）创建一个 TCP 连接，使用 SMTP 协议与邮件服务器进行交谈，经该邮件服务器向某接收方（如你的朋友）发送一个电子邮件报文，最后关闭与该邮件服务器的 TCP 连接。

对本作业，配套 Web 站点为你的客户提供了框架代码。你的任务是完善该代码并通过向不同的用户账户发送电子邮件来测试你的客户。你也可以尝试通过不同的服务器（例如谷歌的邮件服务器和你所在大学的邮件服务器）进行发送。

作业 4：多线程 Web 代理服务器

在这个编程作业中，你将研发一个简单的 Web 代理服务器。当你的代理服务器从一个浏览器接收到对某对象的 HTTP 请求，它生成对相同对象的一个新 HTTP 请求并向初始服务器发送。当该代理从初始服务器接收到具有该对象的 HTTP 响应时，它生成一个包括该对象的新 HTTP 响应，并发送给该客户。这个代理将是多线程的，使其在相同时间能够处理多个请求。

对本作业而言，配套 Web 网站对该代理服务器提供了框架代码。你的任务是完善该代码，然后测试你的代理，方法是让不同的浏览器经过你的代理来请求 Web 对象。

Wireshark 实验：HTTP

在实验 1 中，我们已经初步使用了 Wireshark 分组嗅探器，我们现在准备使用 Wireshark 来研究运行中的协议。在本实验中，我们将研究 HTTP 协议的几个方面：基本的 GET/回答交互，HTTP 报文格式，检索大 HTML 文件，检索具有内嵌 URL 的 HTML 文件，持续和非持续连接，HTTP 鉴别和安全性。

如同所有的 Wireshark 实验一样，对该实验的全面描述可查阅本书的 Web 站点 http://www.awl.com/kurose-ross。

Wireshark 实验：DNS

在本实验中，我们仔细观察 DNS 的客户端（DNS 是用于将因特网主机名转换为 IP 地址的协议）。