实验 A.2：搭建 HTTP 代理

一：实验介绍

在该实验中，需要实现一个 Web 代理，该代理同时在多个 Web 客户端和 Web 服务器之间传递请求和数据。该实验的目的是熟悉 Internet 上最流行的应用程序协议之一，超文本传输协议（HTTP），并介绍 Berkeley 套接字 API。完成实验后，学生应当能够配置 Web 浏览器以将个人代理服务器用作 Web 代理。

二：实验环境

一台 Linux 系统的机器，需要包含 gcc、Makefile 工具

三：相关背景介绍

1. HTTP 传输协议

超文本传输协议（HTTP）是用于 Web 上进行通信的协议：它定义 Web 浏览器如何从 Web 服务器请求资源以及服务器如何响应。为简单起见，在该实验中将处理 HTTP 协议的 1.0 版。HTTP 通信以事务形式进行，其中事务由客户端向服务器发送请求，然后读取响应组成。 请求和响应消息共享一个通用的基本格式：

- 初始行（请求或响应行）
- 零个或多个头部行
- 空行（CRLF）

- 可选消息正文。

对于大多数常见的 HTTP 事务，协议归结为一系列相对简单的步骤：

首先，客户端创建到服务器的连接；然后客户端通过向服务器发送一行文本来发出请求。这请求行包 HTTP 方法(比如 GET，POST、PUT 等)，请求 URI(类似于 URL)，以及客户机希望使用的协议版本(比如 HTTP/1.0)；接着，服务器发送响应消息，其初始行由状态线（指示请求是否成功)，响应状态码(指示请求是否成功完成的数值)，以及推理短语(一种提供状态代码描述的英文消息组成)；最后一旦服务器将响应返回给客户端，它就会关闭连接。

2. HTTP 代理

通常，HTTP 是客户端-服务器协议。 客户端（通常是 Web 浏览器）直接与服务器（Web 服务器软件）进行通信。 在某些情况下，引入称为代理的中间实体可能会很有用。从概念上讲，代理位于客户端和服务器之间。在最简单的情况下，客户端不是将请求直接发送到服务器，而是将其所有请求发送到代理。 然后，代理打开与服务器的连接，并传递客户端的请求。 代理从服务器接收答复，然后将该答复发送回客户端。 从这个角度来看，代理实际上同时扮演 HTTP 客户端（到远程服务器）和 HTTP 服务器（到初始客户端）两个角色。

使用代理的几种可能的原因：

- **性能**：通过保存所获取页面的副本，代理可以减少创建与远程服务器的连接的需求。这可以减少检索页面所涉及的总体延迟，尤其是在服务器位于远程或负载较重的情况下。
- **内容过滤和转换**：虽然在最简单的情况下，代理仅获取资源而不检查资源，但没有任何内容说明代理仅限于盲目的获取和提供文件。代理可以检查请求的 URL 并有选择地阻止对某些域的访问，重新格式化网页（例如，通过剥离图像以使页面更易于在手持式或其他资源有限的客户端上显示），或执行其他转换和过滤。
- **隐私**：通常，Web 服务器记录所有传入的资源请求。此信息通常至少包括客户端的 IP 地址，他们正在使用的浏览器或其他客户端程序（称为用户代理），日期和时间以及所请求的文件。如果客户端不希望记录此个人身份信息，则通过代理路由 HTTP 请求

是一种解决方案。来自使用同一代理的客户端的所有请求似乎都来自代理本身的 IP 地址和 User-Agent，而不是单个客户端。如果许多客户端使用相同的代理（例如，整个企业或大学），则将特定的 HTTP 事务链接到单台计算机或个人将变得更加困难。

三：实验功能要求

1. 基本功能

本实验的基本任务是构建一个 Web 代理，该代理能够接受 HTTP 请求，将请求转发到远程（原始）服务器，并将响应数据返回给客户端。 代理必须通过使用 fork()系统调用为每个新的客户端请求派生一个进程来处理并发请求。 为了简单起见，本实验仅需学生实现 GET 方法。代理收到的所有其他请求方法应引发“未实现”（501）错误。

本实验须使用 C 语言完成。它应该使用 gcc/g++编译并运行，产生一个称为 proxy 的二进制文件，该文件将其侦听的端口作为第一个参数。服务器可设置为知名的网址，代理使用 DNS 服务来获得服务器 IP，同时不能预设客户端 IP 且客户端不能来自预先确定的 IP。

2. 监听

当代理启动时，它要做的第一件事就是建立一个套接字连接，用来监听传入的连接。代理应监听从命令行指定的端口，并等待传入的客户端连接。 每个新的客户端请求均被接受，并使用 fork()生成一个新进程来处理该请求。代理可以创建的进程数应有合理的限制（例如 100 个）。 客户端连接后，代理应从客户端读取数据，随后检查格式正确的 HTTP 请求（本实验已提供解析 HTTP 请求行和标头的库）。具体来说，您将使用我们的库来确保代理接收包含有效请求行的请求，格式如下:

```
<METHOD> <URL> <HTTP VERSION>
All other headers just need to be properly formatted:
<HEADER NAME>: <HEADER VALUE>
```

3. 解析库

本实验提供了一个解析库来对请求的标头进行字符串解析。该库位于框架代码中的proxy_parse.[c | h]中。该库可以将请求解析为一个名为 ParsedRequest 的结构，该结构具有诸如主机名（域名）和端口之类的字段。 它还将自定义标头解析为一组 ParsedHeader 结构，每个结构都包含与标头相对应的键和值。学生可通过键查找标题并进行修改。给定结构中的信息，该库还可以将标头重新编译为字符串。

4. 解析 URL

代理接收到有效的 HTTP 请求后，将需要解析请求的 URL。代理至少需要三项信息：请求的主机和端口以及请求的路径。在此需要解析在请求行中指定的绝对 URL，对此可使用解析库来做解析。若 URL 中指示的主机名未指定端口，则应使用默认的 HTTP 端口 80。

5. 从远端服务器获取数据

代理解析完 URL 后，随后与请求的主机建立连接（使用指定的远程端口，如果未指定，则使用默认值 80），并发送 HTTP 请求。不管代理如何从客户端收到请求，其终以相对 URL +主机头格式发送请求。

从客户端接收请求：

```
GET http://http://www.csu.edu.cn/ HTTP/1.0
```

发送到远端服务器：

```
GET / HTTP/1.0
Host: http://www.csu.edu.cn
Connection: close
```

6. 返回数据到客户端

接收到来自远程服务器的响应后，代理应通过相应的套接字将响应消息原样发送给客户端。

7. 测试代理程序

使用以下命令运行客户端：./proxy <port>， port 表示代理应该侦听的端口号。作为功能的基本测试，可尝试使用 telnet 请求页面：

```
telnet localhost <port>
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^]'.
GET http://www.baidu.com/ HTTP/1.0
```

如果代理运行正常，则终端屏幕会显示 Baidu 主页的标题和 HTML。请注意，我们请求绝对 URL(http://www.baidu.com/)，而不仅仅是相对 URL(/)。另外，尝试从两个不同的 shell 并发地请求一个使用 telnet 的页面。

8. 配置 Firefox 浏览器以使用代理

在 FireFox 10.X 版中先设置好代理配置（主机名和端口）；修改 Firefox 的代理协议修改为 HTTP/1.0，具体操作如下：

（1）在标题栏中键入‘About：config’。

（2）在搜索/筛选栏中，键入‘network.http.Proxy’

（3）可看到：network.http.proxy.keepalive，network.http.proxy.pipelining，和 network.http.proxy.version. 此时将 keepalive 设置为 false，version 修改为 1.0，并确保 pipelining 设置为假。

9. 套接字编程

使用 Berkeley 套接字库来构建代理（比如解析地址，建立连接，创建服务器套接字，通过连接进行通信等），并用 fork，waitpid 等函数来创建并管理多进程。

五：实验评分

由助教进行检查评分。