哈爾濱工業大學
Harbin Institute of Technology

计算机网络
课程实验报告

实验名称	HTTP 代理服务器的设计与实现				
姓名	方烨		院系	计算学部人工智能	
班级	1903601		学号	1190202418	
任课教师	李全龙		指导教师	李全龙	
实验地点	格物 207		实验时间	2021.10.30	
实验课表现	出勤、表现得分(10)		实验报告得分(40)		实验总分	
	操作结果得分(50)				
教师评语					

计算机科学与技术学院 SINCE 1956...
School of Computer Science and Technology

实验目的：

熟悉并掌握 Socket 网络编程的过程与技术；
深入理解 HTTP 协议，掌握 HTTP 代理服务器的基本工作原理；
掌握 HTTP 代理服务器设计与编程实现的基本技能。

实验内容：

(1) 设计并实现一个基本 HTTP 代理服务器。要求在指定端口（例如8080）接收来自客户的 HTTP 请求并且根据其中的 URL 地址访问该地址所指向的 HTTP 服务器（原服务器），接收 HTTP 服务器的响应报文，并将响应报文转发给对应的客户进行浏览。
(2) 设计并实现一个支持 Cache 功能的 HTTP 代理服务器。要求能缓存原服务器响应的对象，并能够通过修改请求报文（添加 if-modified-since头行），向原服务器确认缓存对象是否是最新版本。（选作内容，加分项目，可以当堂完成或课下完成）
(3) 扩展 HTTP 代理服务器，支持如下功能：
（选作内容，加分项目，可以当堂完成或课下完成）
a) 网站过滤：允许/不允许访问某些网站；
b) 用户过滤：支持/不支持某些用户访问外部网站；
c) 网站引导：将用户对某个网站的访问引导至一个模拟网站（钓鱼）。

实验过程：

(1) 程序整体设计逻辑：

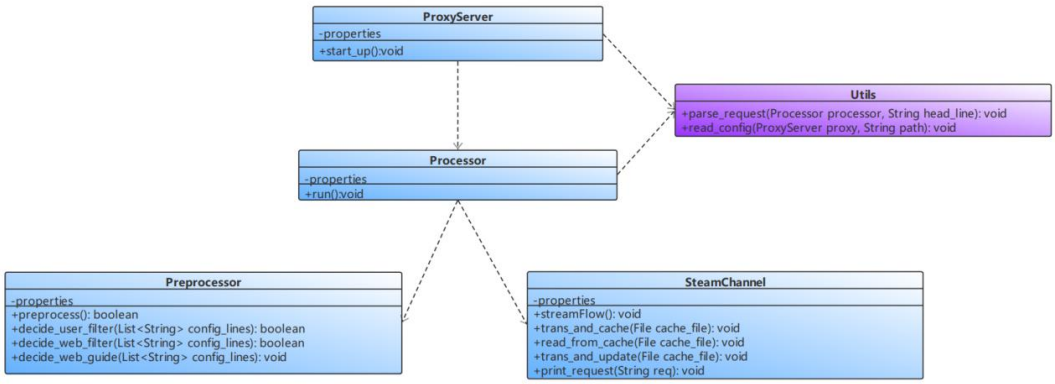


图1-1 程序整体设计类图

类 名	说 明
ProxyServer	代理服务器，监听客户端请求并创建子线程
Processor	处理子线程，处理请求，与服务器建立连接
Preprocessor	对请求报文的预处理，用户/网页过滤、网站引导
StreamChannel	输入输出流的交换与处理（客户端，服务器，缓存）
Utils	工具类，负责读取配置文件，解析函数头部

(2) HTTP 代理服务器的实现原理

◆ **Socket套接字编程通信原理**

客户端去访问服务器时，服务器的进程应该确保已经运行起来，服务器进程会启动一个ServerSocket套接字（欢迎套接字），该套接字是所有客户端与服务器接触的起点，随后该

套接字会生成一个新的套接字，称为连接套接字，连接套接字负责和客户端的套接字进行报文数据的交换,使用流的方式来传输。

具体建立连接及通信的过程如下图：

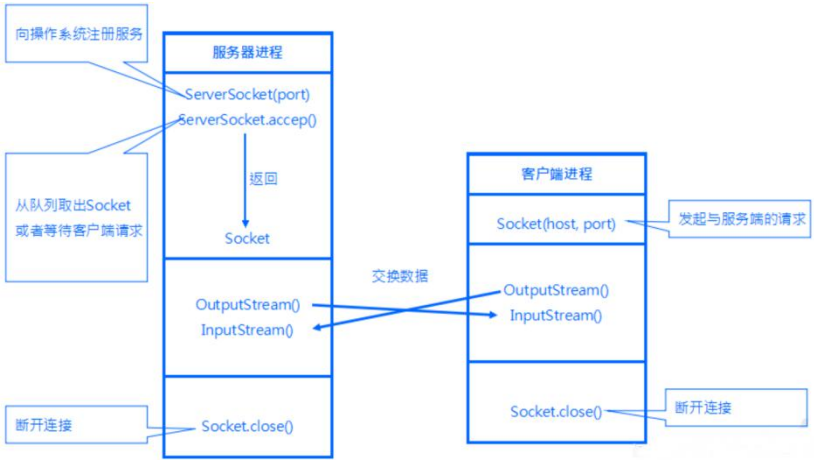


图2-1 Socket编程：客户端服务器建立连接及通信过程

1、在服务端建立一个ServerSocket，绑定相应的端口，并且在指定的端口进行侦听，等待客户端的连接。

2、当客户端创建连接Socket并且向服务端发送请求。

3、服务器收到请求，并且接受客户端的请求信息。一旦接收到客户端的连接请求后，会创建一个连接Socket，用来与客户端Socket进行通信。通过相应的输入/输出流进行数据的交换，数据的发送接收以及数据的响应等等。

4、当客户端和服务端通信完毕后，需要分别关闭Socket，结束通信。

◆ **HTTP代理服务器简要介绍**

代理服务器（Proxy Server），俗称“翻墙软件”，允许一个网络终端（一般为客户端）通过这个服务与另一个网络终端（一般为服务器）进行非直接的连接。

代理服务器的功能是代理网络用户去取得网络信息，它是网络信息的中转站，是个人网络和Internet服务商之间的中间代理机构，负责转发网络信息。HTTP代理服务器能够代理客户机的HTTP访问，主要是代理浏览器访问网页。

◆ **HTTP代理服务器实现逻辑**

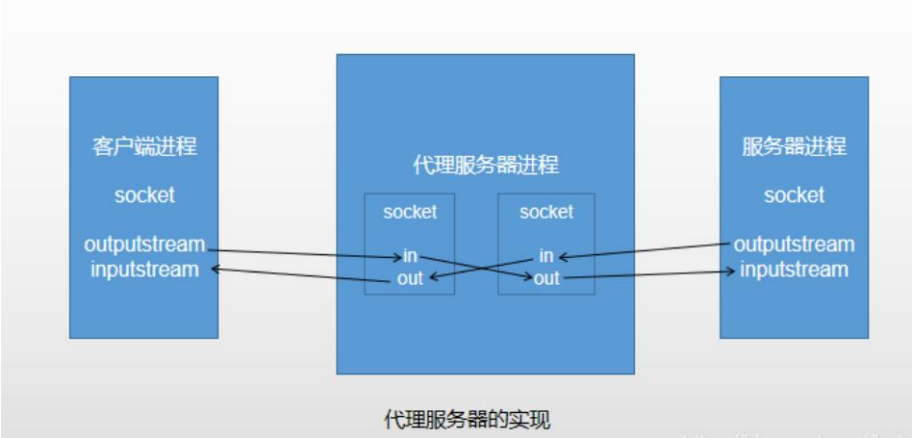


图2-2 代理服务器读写流的过程

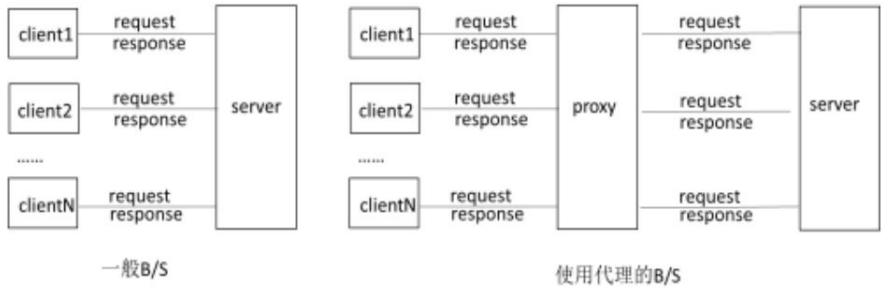


图2-3 Web 应用通信方式对比

通过上图可以发现，代理服务器其实相当于一个“中介”，从功能上来说就是一个和客户端建立连接的服务器以及和客户请求的服务器建立连接的客户端。

对于客户端来说，代理服务器的功能是接收来自客户的HTTP 请求，并对该报文进行相应的处理（解析头部，头部信息修改），并将来自客户的消息流转发给相应的服务器端；

对于服务器端来说，代理服务器接收来自服务器端的响应报文，并对其进行相应的处理（解析头部，缓存响应），并将消息流转发给客户端。

◆ **浏览器使用代理流程**

打开 Windows 10 设置中心、打开网络和 Internet、转到代理、使用代理服务器、输入地址为 127.0.0.1，端口号为 10240。（IE浏览器可以按照实验指导书设置代理）

说明：

①设置代理服务器的地址为 127.0.0.1：127.X.X.X 是本地环回地址，用于本地软件环回测试，因此设置 127.0.0.1 即可表示本机将所有的请求发往127.0.0.1代理服务器即本机。

②设置监听端口号为10240：可以设置为公认端口号之外的任何未被使用的端口号，用于不停监听来自本机的网络请求。

图2-4 设置代理服务器界面

(3) HTTP 代理服务器的关键技术及解决方案

◆ **基本功能及整体流程**

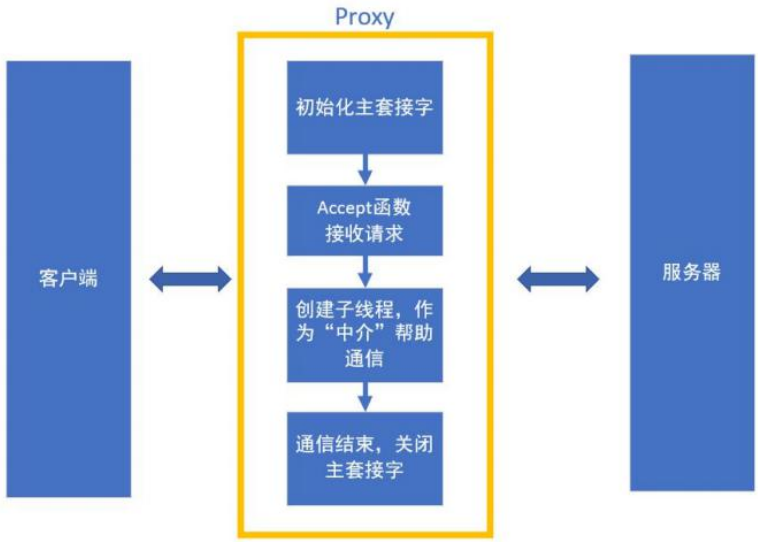


图3-1 代理服务器整体流程图

1.代理服务器监听来自客户端的请求

```
socket_s = new ServerSocket(port); //建立用于监听客户端请求的套接字
while(true) {  //不断监听来自客户端的请求
    new Processor(this, socket_s.accept()).start(); //新建子线程处理连接请求
}
```

2.新建子线程处理连接请求：

I.与客户端建立连接

```
public Processor(ProxyServer proxy, Socket socket_c) {
    this.proxy = proxy;
    this.socket_c = socket_c;   //与客户端建立连接
    this.user_host = socket_c.getInetAddress().getHostAddress();//获得用户主机地址,用于过滤受限用户  }
```

II.读取请求报文内容，解析头部行，保存在程序变量中

a.解析请求报文的首行

```
//获取 url, 头部行中的 host, 以及(如果有)端口号
Utils utils = new Utils();
utils.parse_request(this, line);
```

b.读取并保存请求消息内容

```
while(line!=null) {//读取请求消息内容, 保存在 request_mes 中
    try {
        request_mes += line+"\r\n";
        socket_c.setSoTimeout(this.proxy.timeout); //设置超时时间用于跳出阻塞状态
        line = reader.readLine();
        socket_c.setSoTimeout(0);
    }catch(SocketTimeoutException e) {
        break;
}}
```

III.根据请求报文进行预处理

```
1.  //进行预处理:判断过滤用户/过滤网站，以及对钓鱼网站进行判断和处理
2.  Preprocessor pre = new Preprocessor(this);
3.  boolean flag = pre.preprocess(); //标志预处理判断后是否能进行下一步处理
4.  if(!flag) return;           //如果预处理函数返回 false,则表示为过滤用户/网页，不能进行下一步处理
```

IV.进行转发消息/读写流的操作（非屏蔽用户/网站）

```
1.  //非过滤网站:进行下一步处理请求，客户端与服务器/缓存之间进行流的交换
2.  socket_s = new Socket(this.des_host, this.des_port);
3.  StreamChannel channel = new StreamChannel(this);
4.  channel.streamFlow();
```

V.关闭套接字，断开连接

```
1.  //关闭与服务器端、客户端通信的套接字，断开连接
2.  socket_s.close();
3.  socket_c.close();
```

◆ **缓存功能原理及实现**

代理服务器要实现缓存功能，就需要将服务器的响应报文缓存在本地（代理服务器）文件中，并根据客户端的需要进行查询历史文件的缓存信息，并判断是否直接返回缓存文件中的信息作为响应报文。

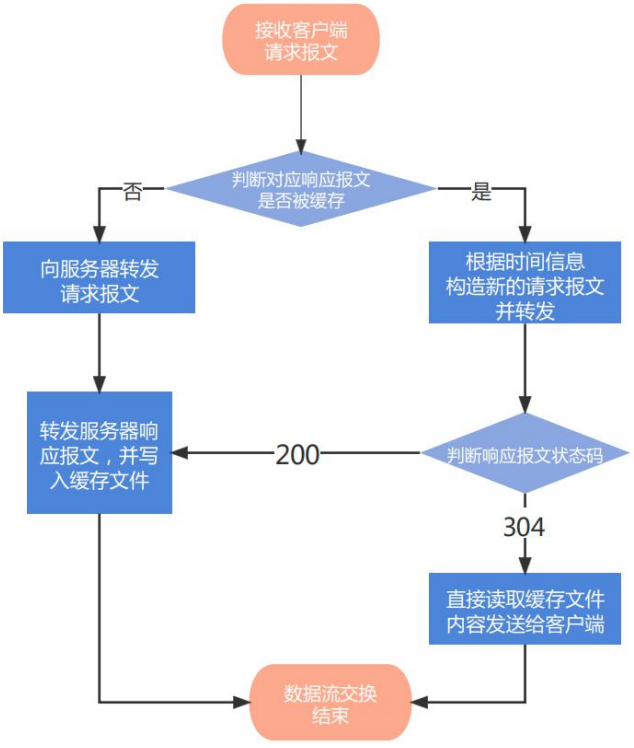


图3-2 缓存操作的基本判断逻辑

实现思路：

• 写缓存基本思路：

在向浏览器输出来自远程服务器的应答的同时，还要向一个文件里面输出相同的内容。

• 基本判断逻辑：

1. 代理服务器收到客户端的请求报文

2. 代理服务器判断对应响应报文是否被缓存：根据客户端请求报文的头部信息，在代理服务器的缓存文件中进行检索；

a. 若不可检索到对应文件，说明未被缓存，则直接向服务器递交请求；并转发服务器响应报文，将其写入本地缓存文件中。

b. 若可检索到对应文件，代理服务器程序在客户的请求报文首部插入<If-Modified-Since: 对象文件的最新被修改时间>，并向原 Web 服务器转发修改后的请求报文。修改过的请求报文被服务器接收后，服务器会发送响应报文。

响应报文也分两种情况：

①若响应报文头部行中含有 304 Not Modified，则说明我们的缓存信息是可用的，未被更新的，所以我们直接将缓存文件中的响应报文发送给客户端；

②若响应报文头部行中含有200 OK，则说明请求信息已被更新，需要转发服务器响应报文，并更新本地缓存文件。

◆ **用户过滤 & 网站过滤 & 网站引导：**

我在Preprocessor类中的preprocess函数中进行这三个操作的预处理。

首先，读取config.txt中的配置文件得到过滤用户、过滤网站、钓鱼网站。

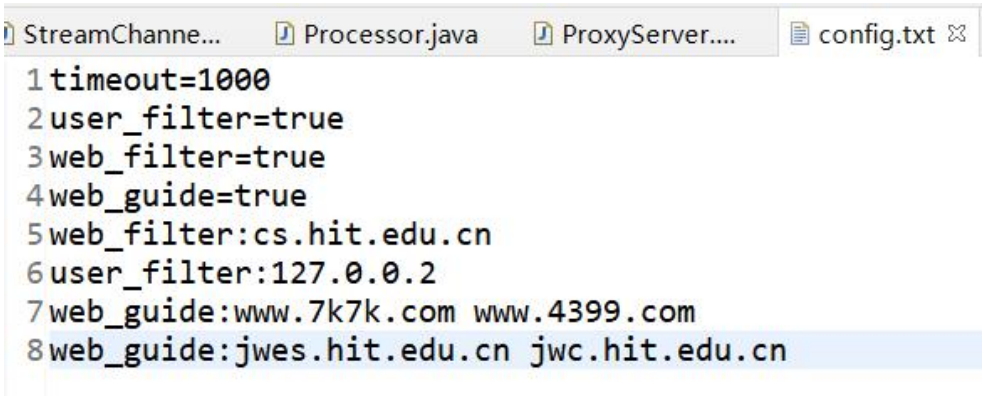


图3-3 配置文件中的过滤用户、过滤网站、引导网站信息

预处理操作具体实现逻辑：

A. 用户过滤：

代理服务器根据与客户端通信的套接字获取客户端的主机，若是过滤文件中指明的要被过滤掉的用户，则直接丢弃客户的请求报文，不再向服务器发送请求。

```
public boolean decide_web_filter(List<String> config_lines) {//判断网页是否为受限网页
    for (String line:config_lines) {
        if(line.contains(processor.des_host) && line.contains("web_filter")) {
            System.out.println("网页受限：\t"+processor.des_host);
            return true;
        }
    }
    return false;
}
```

B. 网站过滤：

代理服务器提取请求报文中的头部行获取目的主机，若是过滤文件中指明的要被过滤掉的主机名，则直接丢弃客户的请求报文，不再向服务器发送请求。

```
public boolean decide_web_filter(List<String> config_lines) {//判断网页是否为受限网页
    for (String line:config_lines) {
        if(line.contains(processor.des_host) && line.contains("web_filter")) {
            System.out.println("网页受限：\t"+processor.des_host);
            return true;
        }
    }
    return false;
}
```

C. 网站引导：

代理服务器提取客户端请求报文中的目的主机，若是过滤文件中指明的要被引导的网站，则将该请求报文中的头部行中的 URL 进行替换，替换为要引导向的 URL 地址，以此达到网页引导的目的。

```
public void decide_web_guide(List<String> config_lines) {//判断是否为钓鱼网站，如果是则将原来的url和请求报文首部行替换
    for (String line:config_lines) {
        if(line.contains(processor.des_host+" ") && line.contains("web_guide")) {
            String old_host = processor.des_host;
            processor.des_host = line.split(" ")[1];
            //替换url中的目的主机，替换请求报文中的头部行
            processor.url = processor.url.replace(old_host, processor.des_host);
            processor.des_port = 80;
            processor.request_mes = processor.request_mes.replace(old_host, processor.des_host);
            System.out.println("网站引导：\t"+old_host+"-->"+processor.des_host);
        }
    }
}
```

预处理过程整体流程图：

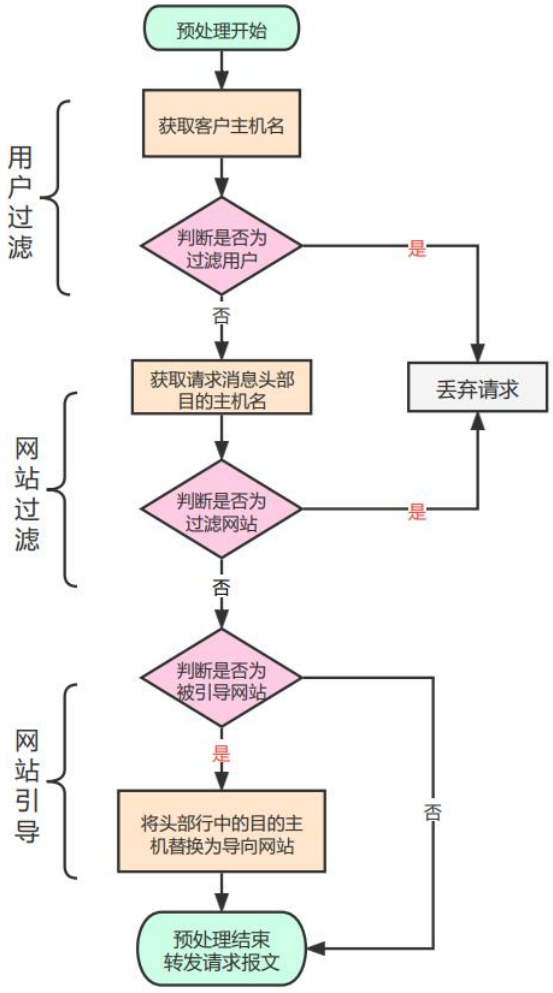


图3-4 预处理过程整体流程图

实验结果：

1. 设计并实现一个基本 HTTP 代理服务器

首先开启代理服务器设置，并运行程序：

(1) 访问教务系统网站：www.jwts.hit.edu.cn

a. 访问页面显示：

b. 对应客户端请求报文：

```
================Request message================
GET http://jwts.hit.edu.cn/ HTTP/1.1
Host: jwts.hit.edu.cn
Proxy-Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) C
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Cookie: _ga=GA1.3.263431105.1597411802
If-Modified-Since: Thu, 28 Oct 2021 14:54:21 GMT
```

(2) 访问4399游戏网站：www.4399.com

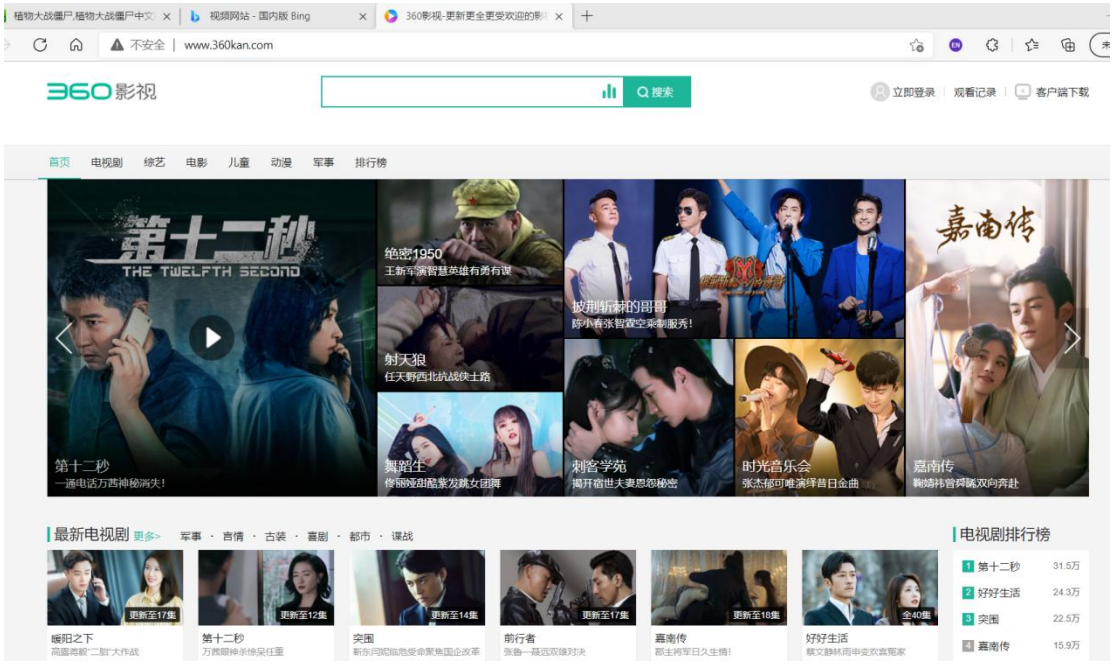
a. 访问页面显示：

b. 对应客户端请求报文：

```
================Request message================
GET http://www.4399.com/js/4399stat.js HTTP/1.1
Host: www.4399.com
Proxy-Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0
Accept: */*
Referer: http://www.4399.com/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
If-Modified-Since: Thu, 28 Oct 2021 09:56:38 GMT


响应报文来源:缓存文件    更新缓存:否  文件名:1297086003.txt

================Request message================
GET http://imga5.5054399.com/upload_pic/2021/6/24/4399_15184854142.jpg HTTP/1.1
Host: imga5.5054399.com
Proxy-Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0.4
Accept: image/avif,image/webp,image/apng,image/*,*/*;q=0.8
Referer: http://www.4399.com/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
If-Modified-Since: Thu, 28 Oct 2021 09:56:37 GMT
```

(3) 访问360视频网站：www.360kan.com

a. 访问页面显示：



b. 对应客户端请求报文：

```
================Request message================
CONNECT s.360kan.com:443 HTTP/1.1
Host: s.360kan.com:443
Proxy-Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chro


Connection reset
缓存文件不存在，需转发请求，文件名:474248816

================Request message================
CONNECT p.s.360kan.com:443 HTTP/1.1
Host: p.s.360kan.com:443
```

2. 实现代理服务器缓存功能

第一次访问今日哈工大网站：today.hit.edu.cn

缓存不命中：

```
缓存文件不存在，需转发请求，文件名:179077266

================Request message================
GET http://today.hit.edu.cn/ HTTP/1.1
Host: today.hit.edu.cn
Proxy-Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like G
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Cookie: _ga=GA1.3.263431105.1597411802; Drupal.visitor.DRUPAL_UID=40028
```

代理服务器转发请求，并将响应消息流写入客户端以及存到本地缓存中。

第二次访问今日哈工大网站：today.hit.edu.cn

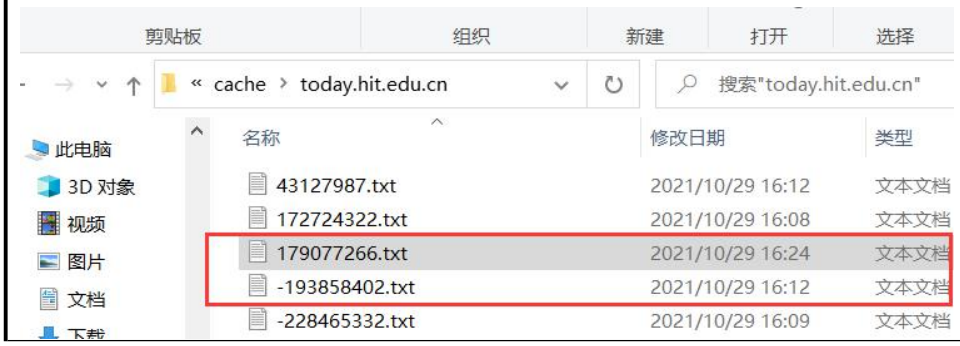
请求下图中资源中显示缓存命中，直接从缓存中读取流写入客户端：

```
================Request message================
GET http://today.hit.edu.cn/themes/custom/hit_front/images/list-hover-lgx.png HTTP/1.1
Host: today.hit.edu.cn
Proxy-Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) C
Accept: image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://today.hit.edu.cn/themes/custom/hit_front/css/style.css?r1q5z4
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Cookie: _ga=GA1.3.263431105.1597411802; Drupal.visitor.DRUPAL_UID=40028


缓存命中数:1        命中:http://today.hit.edu.cn/      文件名:179077266.txt
```

请求下图中资源时，发现需要更新缓存，转发请求，并将服务器响应消息写入客户端，并更新缓存文件：

```
================Request message================
GET http://today.hit.edu.cn/sites/today1.prod1.dpweb1.hit.edu.cn/files/styles/355x22
Host: today.hit.edu.cn
Proxy-Connection: keep-alive
If-Modified-Since: Fri, 29 Oct 2021 07:30:24 GMT
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, lik
If-None-Match: "22838-5cf78ccecb0fc"
Accept: image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://today.hit.edu.cn/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Cookie: _ga=GA1.3.263431105.1597411802; Drupal.visitor.DRUPAL_UID=40028


响应报文来源:服务器端   新建缓存:是  文件名:-193858402.txt
响应报文来源:缓存文件   更新缓存:否  文件名:179077266.txt
```

对应本地缓存文件



3. 实现用户过滤、网站过滤、网站引导

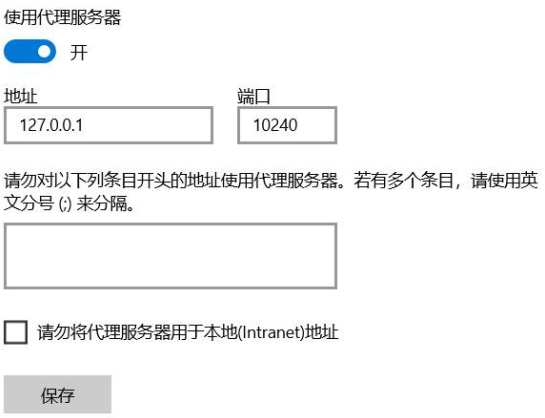
◆ 用户过滤

我们将配置文件config.txt中的受限用户地址设为127.0.0.1

ConcreteVert... UseAnimals.java TestUseAnim... test.java

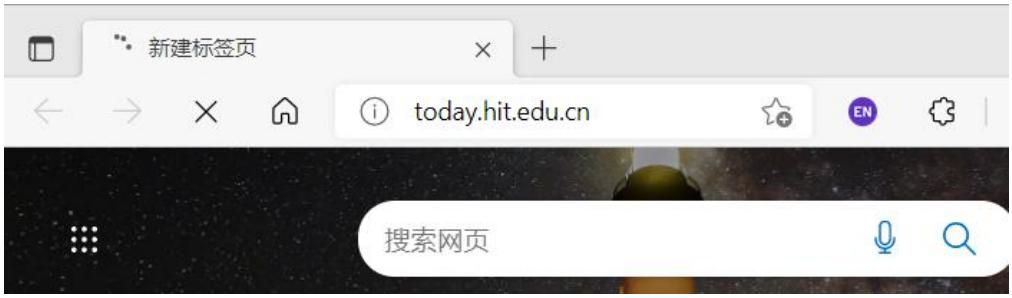
```
1 timeout=1000
2 user_filter=true
3 web_filter=true
4 web_guide=true
5 web_filter:cs.hit.edu.cn
6 user_filter:127.0.0.1
7 web_guide:www.7k7k.com www.4399.com
8 web_guide:jwes.hit.edu.cn jwc.hit.edu.cn
```

同时代理服务器地址设为127.0.0.1



访问今日哈工大网站：today.hit.edu.cn

```
**********************************************
Proxy Server:    Run
Listening Port: 10240
User Filtering: Open
Web Filtering:  Open
Web Guide:      Open
Cache:          Enable
**********************************************
用户受限： 127.0.0.1
用户受限： 127.0.0.1
```



控制台显示用户受限，且无法成功加载网页，说明**用户过滤功能**已经实现。

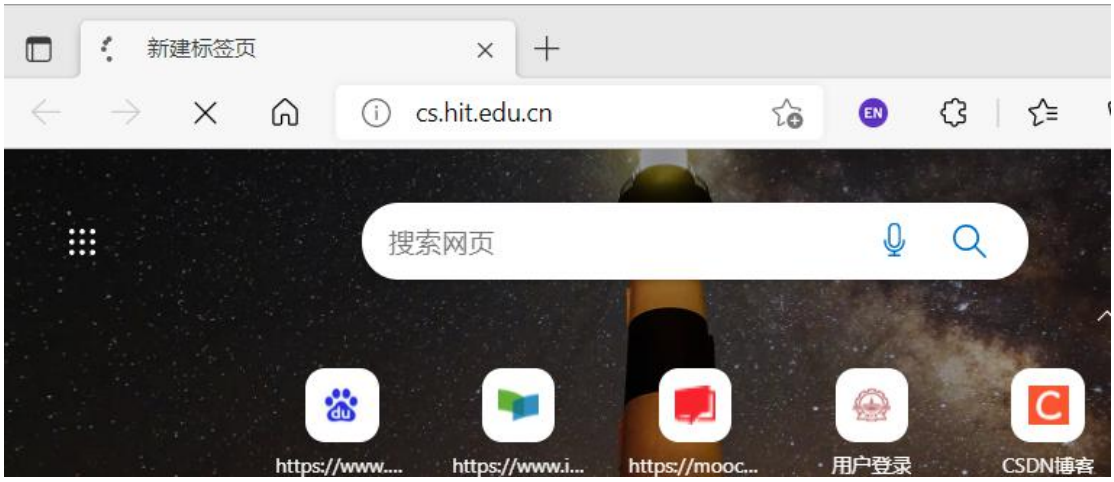
◆ **网站过滤：**

我们设置被过滤网站为：cs.hit.edu.cn

ConcreteVert... UseAnimals.java TestUseAnim... t

```
1 timeout=1000
2 user_filter=true
3 web_filter=true
4 web_guide=true
5 web_filter:cs.hit.edu.cn
6 user_filter:127.0.0.2
7 web_guide:www.7k7k.com www.4399.com
8 web_guide:jwes.hit.edu.cn jwc.hit.edu.cn
```

在浏览器中输入cs.hit.edu.cn




```
================Request message================
CONNECT assets.msn.cn:443 HTTP/1.1
Host: assets.msn.cn:443
Proxy-Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
If-Modified-Since: Fri, 29 Oct 2021 08:24:02 GMT


Connection reset
网页受限： cs.hit.edu.cn
```

网页无法成功加载且控制台输出网页受限，说明**网站屏蔽功能**已经实现。

◆ **网站引导：**

我们设置钓鱼网站：

www.7k7k.com → www.4399.com

jwes.hit.edu.cn → jwc.hit.edu.cn

```
1 timeout=1000
2 user_filter=true
3 web_filter=true
4 web_guide=true
5 web_filter:cs.hit.edu.cn
6 user_filter:127.0.0.2
7 web_guide:www.7k7k.com www.4399.com
8 web_guide:jwes.hit.edu.cn jwc.hit.edu.cn
```

（1）输入网址：www.7k7k.com

```
================Request message================
GET http://ptlogin.3304399.net/resource/ucenter.js?202110292 HTTP/1.1
Host: ptlogin.3304399.net
Proxy-Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Intervention: <https://permanently-removed.invalid/feature/5718547946799104>; level="warning"
Accept: */*
Referer: http://www.7k7k.com/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6

网站引导： www.7k7k.com-->www.4399.com
缓存文件不存在，需转发请求，文件名:1384703666
```

网站被引导至www.4399.com，说明**网站引导**成功。

（2）输入网址：jwes.hit.edu.cn

```
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Cookie: _ga=GA1.3.263431105.1597411802; JSESSIONID=F3AABC13B197E555BCA9

响应报文来源:服务器端  新建缓存:是 文件名:-2037626555.txt
网站引导： jwes.hit.edu.cn-->jwc.hit.edu.cn
缓存文件不存在，需转发请求，文件名:-792862074

================Request message================
GET http://jwc.hit.edu.cn/_wp3services/generalQuery?queryObj=trailer&si
Host: jwc.hit.edu.cn
Proxy-Connection: keep-alive
```

网站被引导至jwc.hit.edu.cn，说明**网站引导**成功。

问题讨论：

- 从套接字读写流的时候，需要设置超时时间：

在从 client_socket 与 server_socket 的流中读取数据时，需要设置超时时间，因为如果不设置超时时间，流的末尾只有到套接字关闭的时候才会出现，因此程序会陷入阻塞，无法继续执行。

```
while(line!=null) {//读取请求消息内容，保存在request_mes中
    try {
        request_mes += line+"\r\n";
        socket_c.setSoTimeout(this.proxy.timeout); //设置超时时间用于跳出阻塞状态
        line = reader.readLine();
        socket_c.setSoTimeout(0);
    }catch(SocketTimeoutException e) {
        break;
    }
}
```

- 对于本地缓存而言，需要向服务器端确认是否更新：

如果缓存文件命中，需要判断是否需要更新，采用的方式是对原来的客户端请求报文进行重构，将if-modified-since Date 语句加在请求报文的倒数第二句（倒数第一句为\r\n）。

```
//请求报文中加入时间信息，并将消息流发送给服务器端
DateFormat df = new SimpleDateFormat("EEE, d MMM yyyy HH:mm:ss z", Locale.ENGLISH);
df.setTimeZone(TimeZone.getTimeZone("GMT"));
processor.request_mes = processor.request_mes.replace("\r\n\r\n", "\r\nIf-Modified-Since: "+df.format(c
out_to_server.write(processor.request_mes);
out_to_server.flush();
print_request(processor.request_mes);
```

- C++，Java实现代理服务器网页加载效率差异问题：

在编程过程中，我先后使用了两种语言进行编程实现，一开始选择的是c++语言，参考实验指导书做出了HTTP代理服务器的基本功能，不过c++加载网页的效果较差，像教务系统网站、7k7k网站上的部分图片和内容很难加载出来。

而后来转而使用Java进行编程后，网页加载速度明显提升，并且基本http开头的网站都

能加载出来。

一开始推测，是Java的流处理比[实验指导书]上的C++代码更细致：

```
//将客户端发送的HTTP数据报文直接转发给目标服务器
ret = send(((ProxyParam*)lpParameter)->serverSocket, Buffer, strlen(Buffer) + 1, 0);
//等待目标服务器返回数据
recvSize = recv(((ProxyParam*)lpParameter)->serverSocket, Buffer, MAXSIZE, 0);
if (recvSize <= 0) {
    printf("返回目标服务器的数据失败 ");
    goto error;
}
//将目标服务器返回的数据直接转发给客户端
ret = send(((ProxyParam*)lpParameter)->clientSocket, Buffer, sizeof(Buffer), 0);
```

实验指导书上的C++代码转发数据的操作仅调用send, recv函数，设置了转发的上限，但对于怎么读取数据流，读取数据流的时间都没有处理。

而用Java实现时，我的流处理是更加完善的，数据流以什么样的单位进行读取，如何储存，数据流的超时时间，每一次数据流动都进行了处理，所以后来Java代码的加载效果比较好。

```
List<Byte> server_bytes = new ArrayList<>();
while(true) { //从服务器端读响应流，保存在字节列表server_bytes中
    try {
        socket_s.setSoTimeout(processor.proxy.timeout); //设置超时时间用于跳出阻塞状态
        int b = in_from_server.read();
        if(b == -1) break;  //the end of the stream is reached
        else {
            server_bytes.add((byte) (b));
            socket_s.setSoTimeout(0);
        }
    }catch(SocketTimeoutException e) {
        break;
    }
}
```

后来和同学讨论时发现csdn上一篇博客对c++的这一问题提供了解决思路：

解决HTTP/1.1以及图片加载问题

在解决之前，在Github上转了一圈，所看有限几个repo中有的绕过了这个部分，直接像上面一样直接解析发送HTTP/1.0的请求，有的直接无差别用readline导致图片等文件仍然陷入read导致必须等待对方服务器断开连接后才能读到完整数据从read中出来，而导致网页加载速度奇慢。

下面就从HTTP的协议入手，寻找一个妥善的方法解决该问题。

1. 当客户端请求时是Connection: keep-alive的时候，服务器返回的形式Transfer-Encoding: chunked的形式，以确保页面数据是否结束，长连接就是这种方式，用chunked形式就不能用content-length
2. content-length设置响应消息的实体内容的大小，单位为字节。对于HTTP协议来说，这个方法就是设置Content-Length响应头字段的值。
3. 因为当浏览器与WEB服务器之间使用持久(keep-alive)的HTTP连接，如果WEB服务器没有采用chunked传输编码方式，那么它必须在每一个应答中发送一个Content-Length的响应头来表示各个实体内容的长度，以便客户端能够分辨出上一个响应内容的结束位置。
4. 当不是keep-alive，就是常用短连接形式，会直接把连接关掉，不需要长度。
5. 服务器上取得是动态内容，所有没有content-length这项
6. 如果是静态页面，则有content-length

整体思路和之前自己分析的结果差不多，主要由于数据处理不够精细，加上阻塞导致了读出完整数据很慢。博文中提出了不能无脑读响应消息头部，要对头部进行解析，记录消息长度，在读消息体的时候对长度更加精细处理。

问题参考链接：https://blog.csdn.net/weixin_33736048/article/details/94193381

心得体会：

本次实验，让我对socket编程有了初步的了解，进一步理解了基于TCP连接的通信过程，掌握了HTTP代理服务器的基本原理，结合代码实践，回看MOOC中的socket编程这一讲时，感觉有一种真正理解了知识的感觉。同时通过实现附加功能：缓存功能和过滤及引导功能，掌握了一个小型HTTP代理服务器设计与编程实现的基本技能。

通过Java编程，熟悉了Java的流处理机制，加深了对Java代码的熟练程度，同时对使用socket套接字编程有了更深刻的体会与认知。