Design and implementation of HTTP Proxy Server

Course Title: *Internet Application*

Name: *Liu Xiao (ee07b425 072910)*

Song Yutian (ee07b415 072900)

Date: 2010/1/1

1. Overview

1.1 Target

a) Forward HTTP data from browser to web server.
b) Record and display related information.
c) Forbid the access to sites in black list.

1.2 Target

a) Understand basic principles and operational mechanism of HTTP.
b) Be familiar with socket and web programming.
c) Using wireshark software to analyze network packets.

2. Requirements Analysis

2.1 Development Environment

a) Server Operation System: Red Flag Linux 5.0
Ubuntu 9.10
b) Client Operation System: Red Flag Linux 5.0
Ubuntu 9.10
Windows XP
Windows Vista
c) Client web browser: Windows Internet Explorer 8.0
Firefox 3.0 (Mozilla 5.0) for Windows
Firefox 3.0 (Mozilla 5.0) for Linux
d) Programming language: C language

2.2 Functional requirements in details

The functions of the HTTP proxy server are separated into several parts:
a) It can support multiple browsers concurrently
b) URL can contain domain name and port number. If the user does not insert a port number, it will automatically be set to 8080.
c) If browsers access sites in your “black list” proxy server will redirect them to a page written in HTML with cautions and also with audios and pictures.
d) Display HTTP version, browser type, client OS, language, site name, web server name and web server OS as the output of proxy server:
e) Proxy server can authenticate user by name and password.

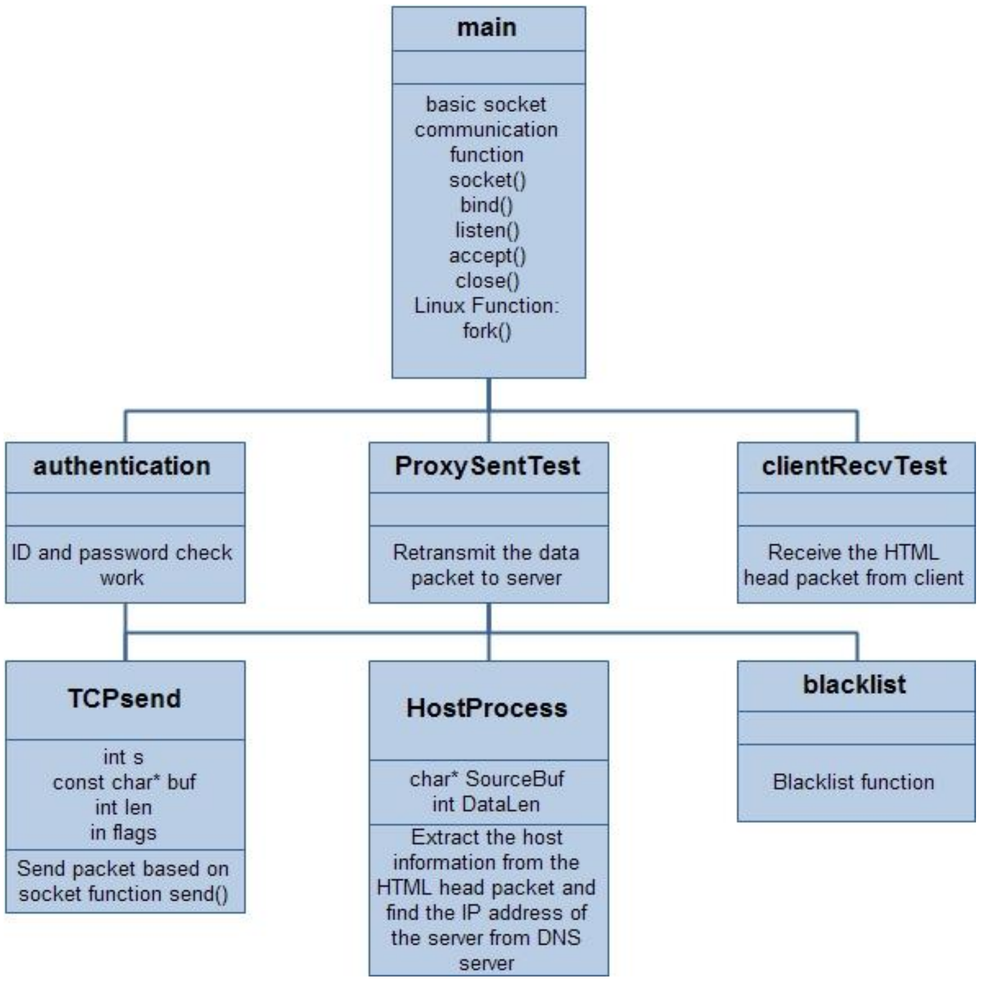
3. Preliminary Design

3.1 Decomposition of functional modules

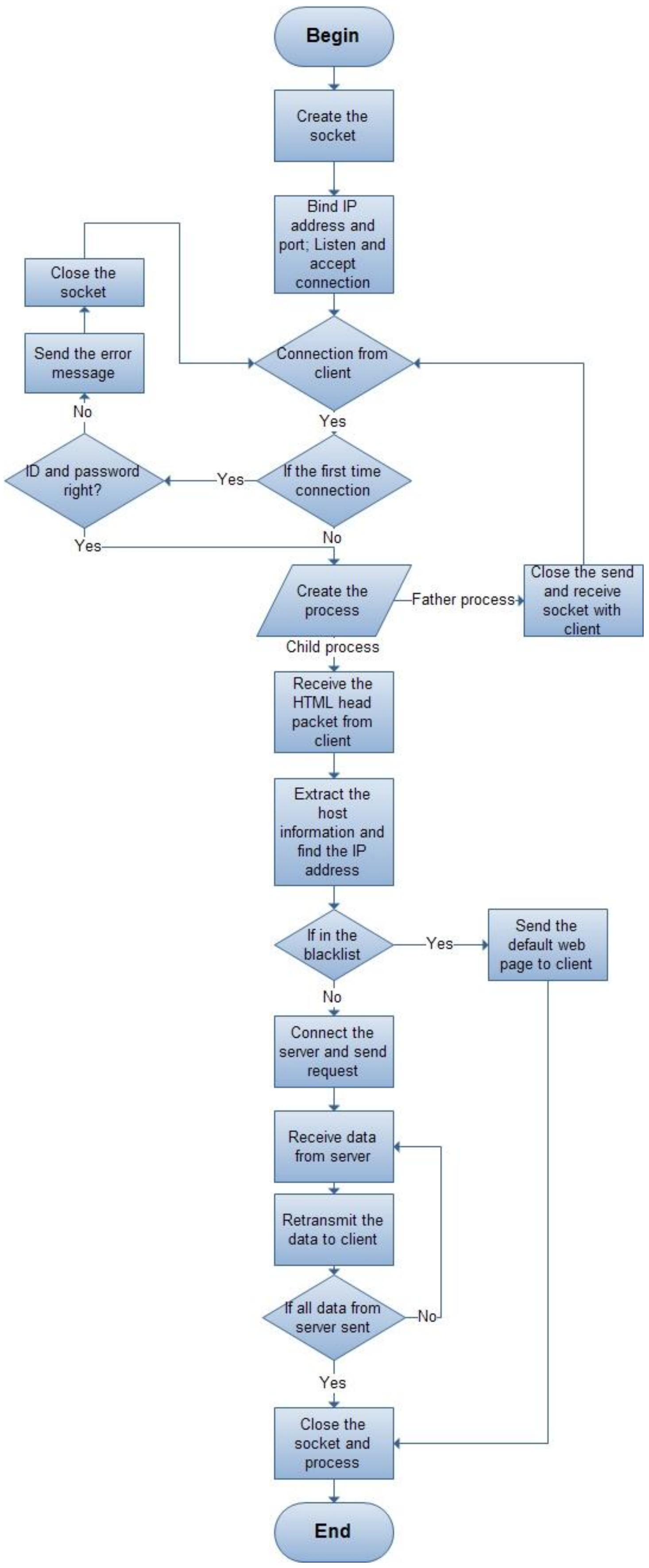
a) Main function: Achieve the basic function of socket communication. Including socket creating process, port and address binding process, listen and accept process, process creating process, socket close process.
b) Authentication function: To do the user ID authentication work when user first open the proxy and visit the website. If user does not insert the correct ID and password, the function will stop user from using the proxy.
c) User packet head receive function: This function is simple. It just receives the head of the HTML packet from client.
d) User packet head process function: The function is to extract the host name and port number of user's forwarding website from HTML packet head. The proxy will use these data to find the host of the website and make the connection.
e) Blacklist function: This function is to check the website which the user wants to visit. If the website is in the preset blacklist, the connection will be forbidden, and a caution page will be sent to client.
f) Data send function: This function is based on the socket function send(). However, it adds the send guaranty mechanism. If some packets are not successfully sent by the TCP/IP protocol, it will automatically resent the fail sending part.

3.2 Relationship and interface between the modules

The relationship and interface between different modules is shown below:


main
basic socket communication function
socket()
bind()
listen()
accept()
close()
Linux Function:
fork()
authentication
ID and password check work
ProxySentTest
Retransmit the data packet to server
clientRecvTest
Receive the HTML head packet from client
TCPsend
int s
const char* buf
int len
in flags
Send packet based on socket function send()
HostProcess
char* SourceBuf
int DataLen
Extract the host information from the HTML head packet and find the IP address of the server from DNS server
blacklist
Blacklist function

3.3 Overall flow chart

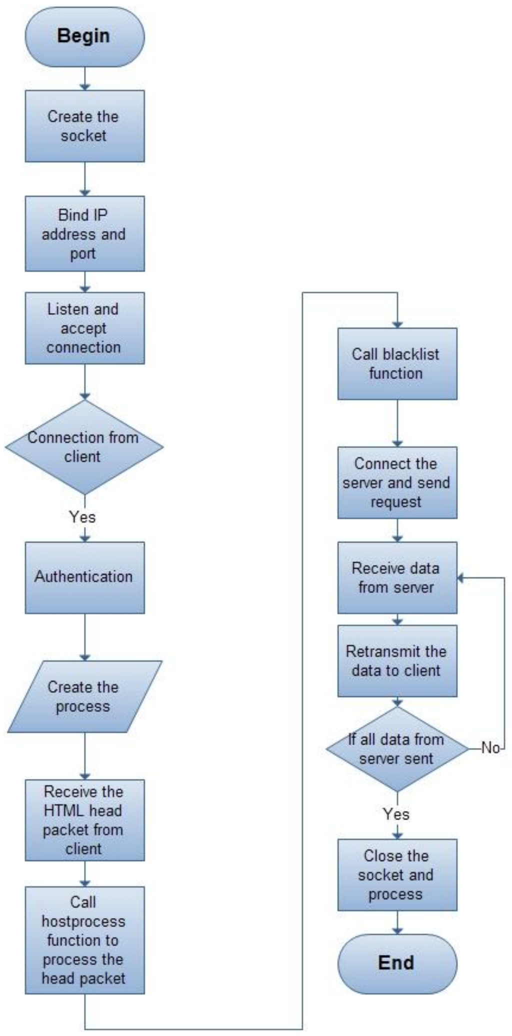

Begin
Create the socket
Bind IP address and port; Listen and accept connection
Close the socket
Send the error message
Connection from client
No
Yes
ID and password right?
Yes
If the first time connection
Yes
No
Create the process
Father process
Close the send and receive socket with client
Child process
Receive the HTML head packet from client
Extract the host information and find the IP address
If in the blacklist
Yes
Send the default web page to client
No
Connect the server and send request
Receive data from server
Retransmit the data to client
If all data from server sent
No
Yes
Close the socket and send process
End

4. Detailed Design

a) main function (main and ProxySentTest)

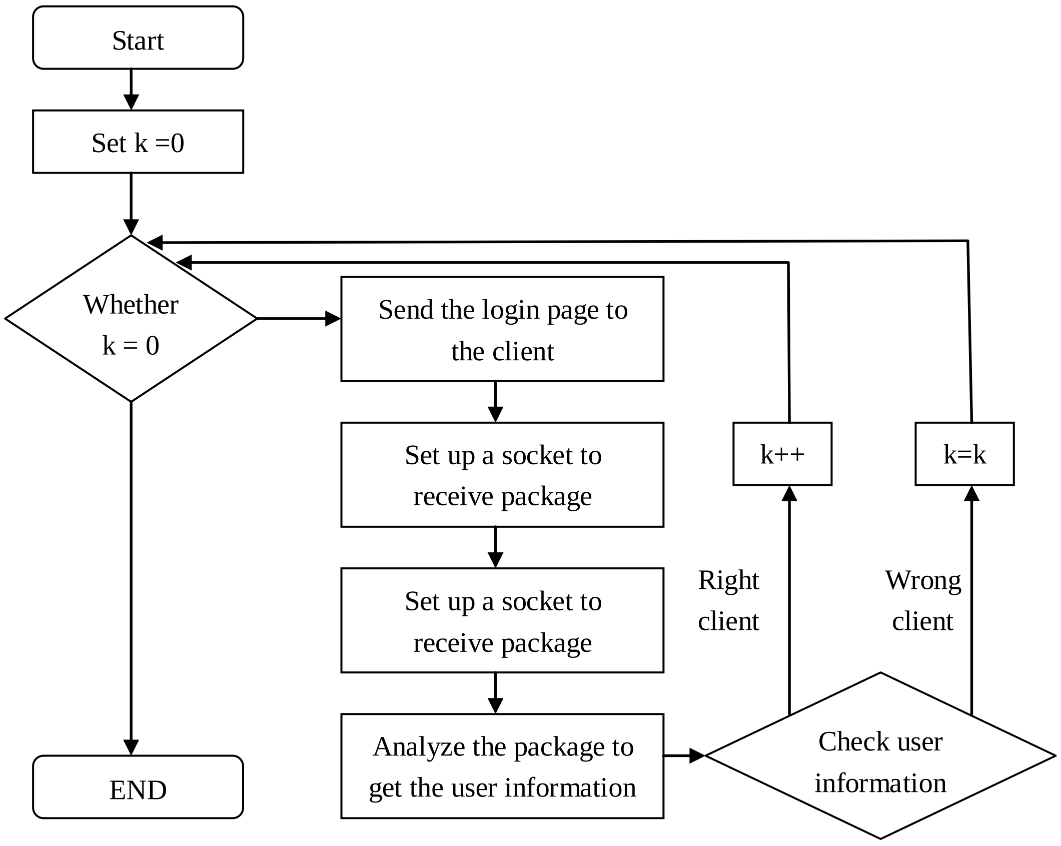
The main function is the main part of the program. It first creates the socket and binds it with the specific IP address and port. The default configuration of IP address is the address of the host who is running the proxy program and the default port number is 8080. It then starts to listen to this IP address and port number and does the authentication work. If the user successfully passes the identification authentication, it will create the process and receive the HTML head packet from client. It then calls the hostprocess function to process the head packet and connects the server. When the connection is successfully created, the main function will receive data from server and retransmit it to the client. After all data are sent, the main function will close the socket and finish the whole flow.

Flow chart:


Begin
Create the socket
Bind IP address and port
Listen and accept connection
Connection from client
Yes
Authentication
Create the process
Receive the HTML head packet from client
Call hostprocess function to process the head packet
Call blacklist function
Connect the server and send request
Receive data from server
Retransmit the data to client
If all data from server sent
No
Yes
Close the socket and process
End

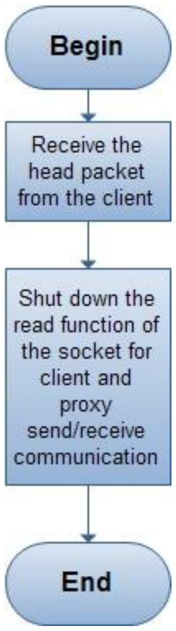
b) Authentication:

This part was built in the main function before we start the fork(). After we set up the connection with the client, we will send a buffer with the html code which can generate a webpage with the two labels let the client to type in ID and password. After the client click the button "login" on the page, we will receive a package with the ID and the password. From the result of the wieshark we san see the format of the package ---"IDXXX&PASSWORDXXX&LOGINLOGIN&". Then by using strtok() and strstr() we can get the ID and the password. Then we compare the ID and password with what we defined. If both of them are the same we will leave this process and go into the main function. Then a notice message will tell the client the success of the login. If any of them not correct we will stop the socket and go back to the start of the authentication.


Start
Set k =0
Whether k = 0
Send the login page to the client
Set up a socket to receive package
Set up a socket to receive package
Analyze the package to get the user information
Check user information
Right client
Wrong client
k++
k=k
END

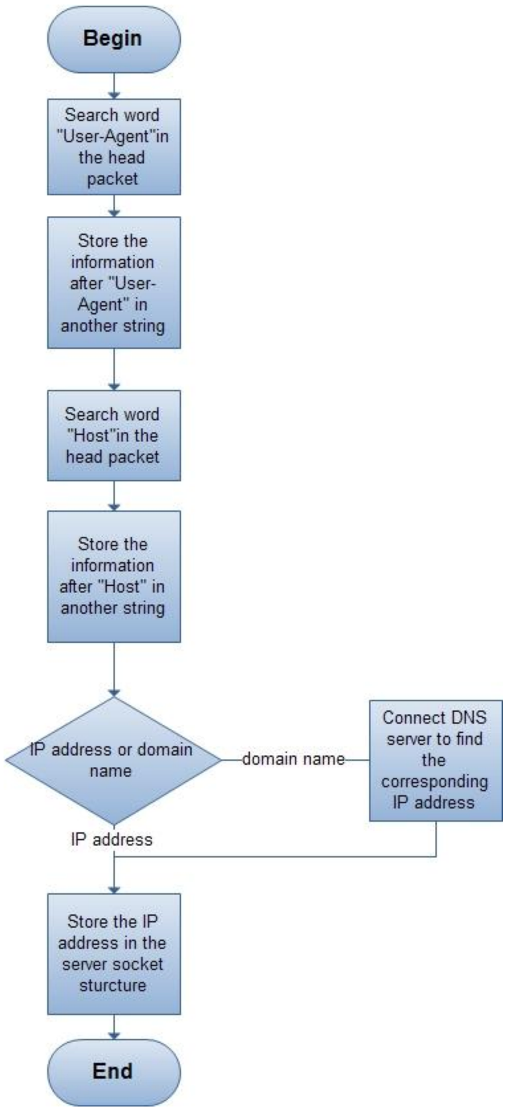
c) <u>User packet head receive function (clientRecvTest):</u> This function is simple. It just receives the head of the HTML packet from client.

Flow chart:


Begin
Receive the head packet from the client
Shut down the read function of the socket for client and proxy send/receive communication
End

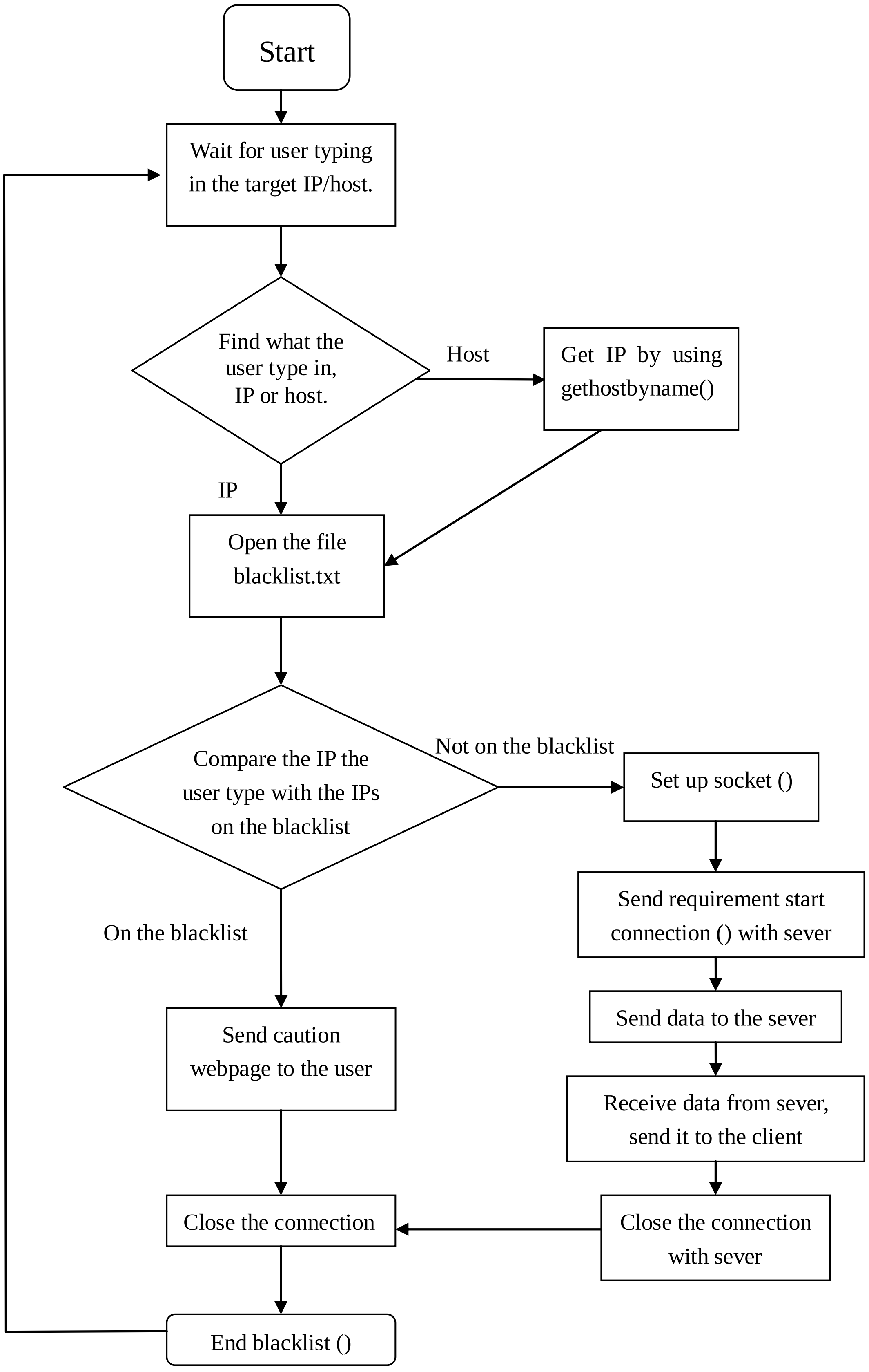
<u>d) User packet head process function (HostProcess):</u> The function first receives the HTML packet from the main function, which is stored in a string. The function then searches the string to find the “Host” word. When found, it will cut the “Host” and store the IP address or domain name after it in another string. Then the function finds the “User-Agent” in the same way. The function then to make sure host name which user types in is an IP address or domain name. If it is a domain name, the function will connect the DNS server and obtain the corresponding IP address. At last it saves the IP address into the server socket structure.

Flow chart:


Begin
Search word "User-Agent"in the head packet
Store the information after "User-Agent" in another string
Search word "Host"in the head packet
Store the information after "Host" in another string
IP address or domain name
domain name
Connect DNS server to find the corresponding IP address
IP address
Store the IP address in the server socket sturcture
End

e) Blacklist:

Function: Void blacklist ()

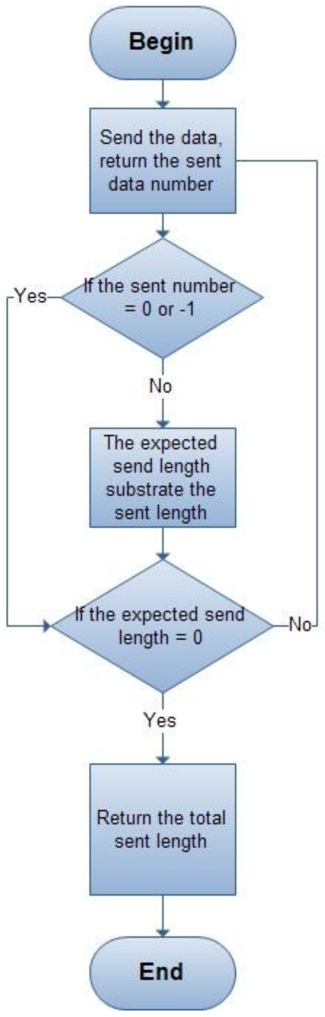


Start
Wait for user typing in the target IP/host.
Find what the user type in, IP or host.
Host
Get IP by using gethostbyname()
IP
Open the file blacklist.txt
Compare the IP the user type with the IPs on the blacklist
Not on the blacklist
Set up socket ()
Send requirement start connection () with sever
Send data to the sever
Receive data from sever, send it to the client
On the blacklist
Send caution webpage to the user
Close the connection
Close the connection with sever
End blacklist ()

In this module we use a .txt file to store the blacklist and limit user to access the website with the

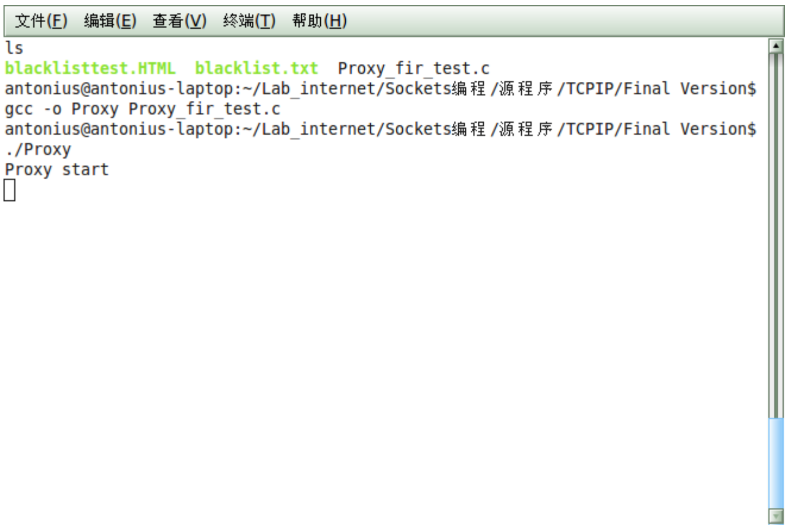
same IP on the blacklist. Here we read the file by using “fgets”. The data will be read maximum 20 bits per line then store the data in the buffer. The buffer has a max-length of 20 bits for the IP will not be longer than 20 bits. And this also makes the blacklist a fixed form that each line of the file has only one “black IP”.
This function will be used every time user wants to visit a website. When the user type in the IP or host name, we will store it in the buffer. As we have analysis the package before. We will get the target IP by gethostbyname() or directly from what the user typed in. Then we will compare the IP with the blacklist. If it on the blacklist, we will send a caution webpage by reading an .html file and close the connection. If the IP not on the blacklist we will set up the connection with the sever and start to send and receive data.

f) Data send function (TCPsend): The function is based on the socket function send(). However, when we use the send() function, one problem will occur, that send() cannot guarantee that all data will be successfully sent to the destination. So we rewrite the function. In this function, we first use the send() function to send the data. The send() function will return a value of the actual send byte number. The function will compare the return value with the expected value for sending. If they are not the same, the function will store the difference between them and send the last data depending on this difference. Only when all data are sent does the function exits.
Flow chart:

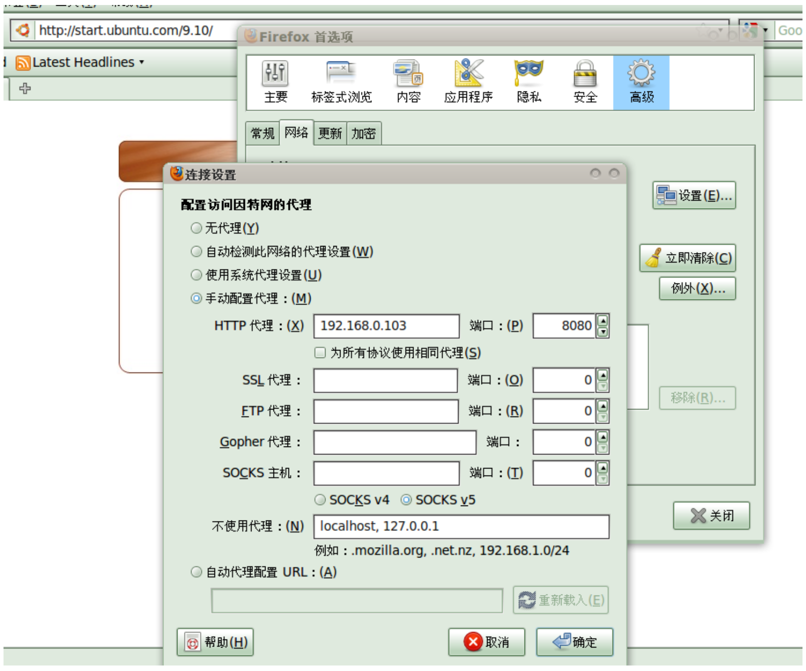

Begin
Send the data, return the sent data number
If the sent number = 0 or -1
Yes
No
The expected send length substrate the sent length
If the expected send length = 0
No
Yes
Return the total sent length
End

5. Results

a) Proxy start and the user should configure the browser



```
ls
blacklisttest.HTML  blacklist.txt  Proxy_fir_test.c
antonius@antonius-laptop:~/Lab_internet/Sockets编程/源程序/TCPIP/Final Version$
gcc -o Proxy Proxy_fir_test.c
antonius@antonius-laptop:~/Lab_internet/Sockets编程/源程序/TCPIP/Final Version$
./Proxy
Proxy start
```



配置访问因特网的代理

- 无代理(Y)
- 自动检测此网络的代理设置(W)
- 使用系统代理设置(U)
- [✓ 手动配置代理：(M)]

HTTP 代理：(X) 192.168.0.103　端口：(P) 8080

☐ 为所有协议使用相同代理(S)

SSL 代理：　端口：(O) 0

FTP 代理：　端口：(R) 0

Gopher 代理：　端口： 0

SOCKS 主机：　端口：(T) 0

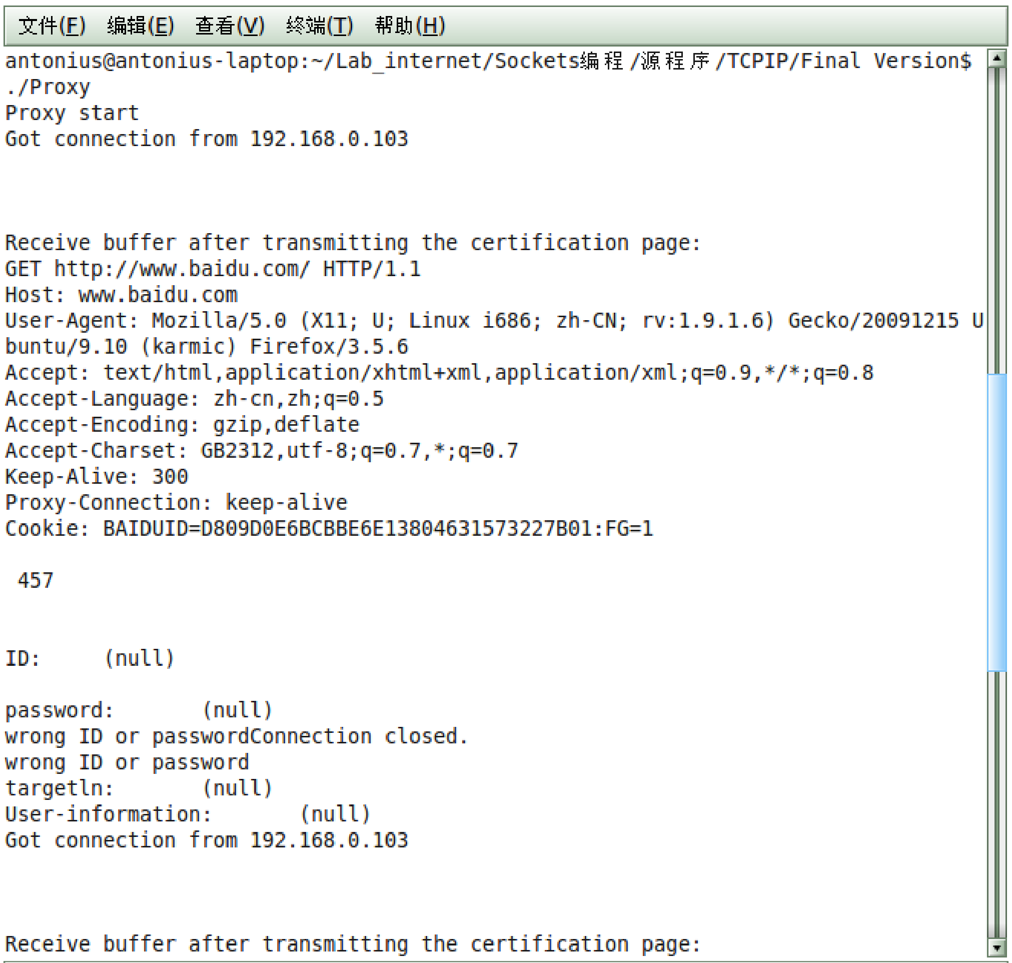
SOCKS v4 [✓ SOCKS v5]

不使用代理：(N) localhost, 127.0.0.1

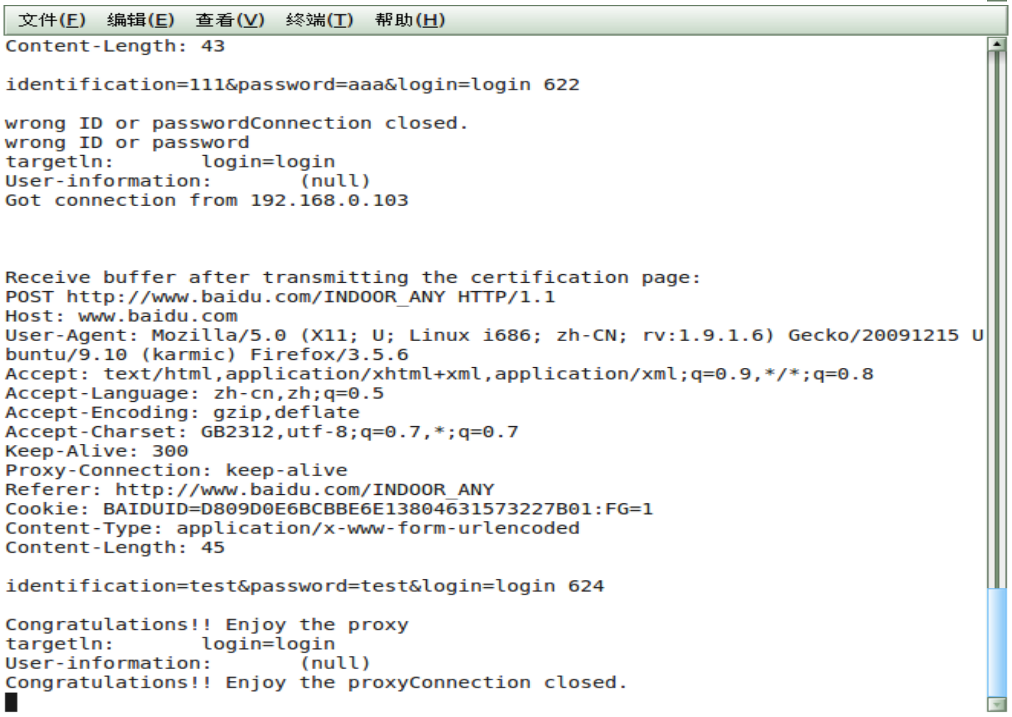
例如：.mozilla.org, .net.nz, 192.168.1.0/24

- 自动代理配置 URL：(A)

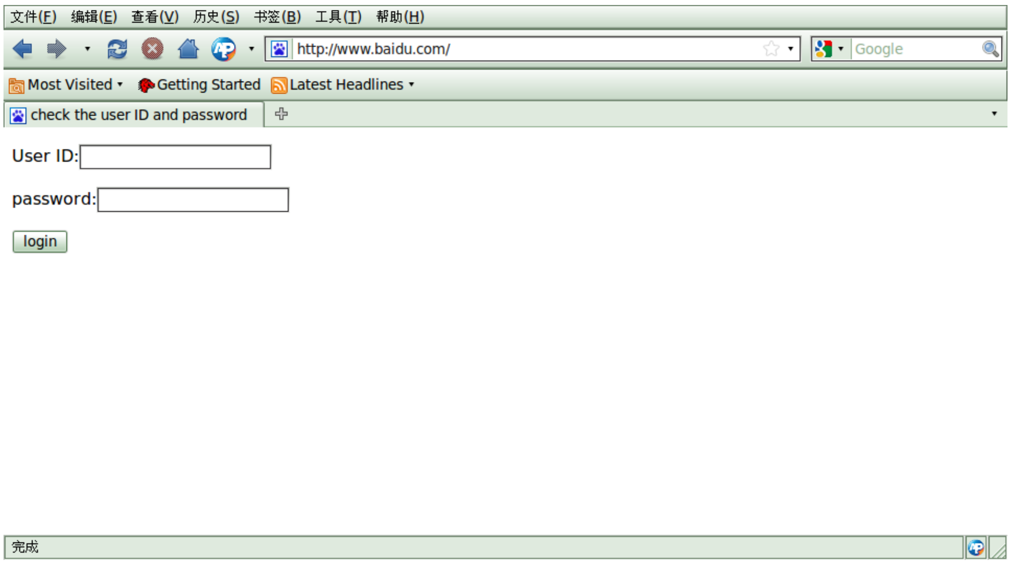
b) When user connects the proxy, proxy will receive the data packet and sends back an authentication page



```
antonius@antonius-laptop:~/Lab_internet/Sockets编程/源程序/TCPIP/Final Version$
./Proxy
Proxy start
Got connection from 192.168.0.103



Receive buffer after transmitting the certification page:
GET http://www.baidu.com/ HTTP/1.1
Host: www.baidu.com
User-Agent: Mozilla/5.0 (X11; U; Linux i686; zh-CN; rv:1.9.1.6) Gecko/20091215 U
buntu/9.10 (karmic) Firefox/3.5.6
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-cn,zh;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: GB2312,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Proxy-Connection: keep-alive
Cookie: BAIDUID=D809D0E6BCBBE6E13804631573227B01:FG=1

 457


ID:     (null)

password:       (null)
wrong ID or passwordConnection closed.
wrong ID or password
targetln:       (null)
User-information:       (null)
Got connection from 192.168.0.103



Receive buffer after transmitting the certification page:
```



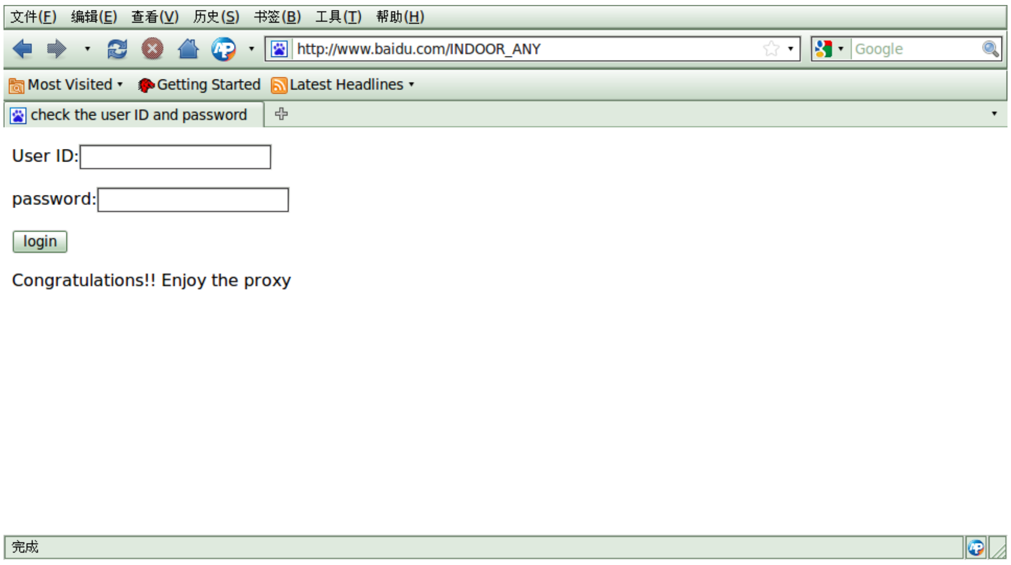
```
Content-Length: 43

identification=111&password=aaa&login=login 622

wrong ID or passwordConnection closed.
wrong ID or password
targetln:       login=login
User-information:       (null)
Got connection from 192.168.0.103



Receive buffer after transmitting the certification page:
POST http://www.baidu.com/INDOOR_ANY HTTP/1.1
Host: www.baidu.com
User-Agent: Mozilla/5.0 (X11; U; Linux i686; zh-CN; rv:1.9.1.6) Gecko/20091215 U
buntu/9.10 (karmic) Firefox/3.5.6
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-cn,zh;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: GB2312,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Proxy-Connection: keep-alive
Referer: http://www.baidu.com/INDOOR_ANY
Cookie: BAIDUID=D809D0E6BCBBE6E13804631573227B01:FG=1
Content-Type: application/x-www-form-urlencoded
Content-Length: 45

identification=test&password=test&login=login 624

Congratulations!! Enjoy the proxy
targetln:       login=login
User-information:       (null)
Congratulations!! Enjoy the proxyConnection closed.
```


文件(F) 编辑(E) 查看(V) 历史(S) 书签(B) 工具(T) 帮助(H)
http://www.baidu.com/
Google
Most Visited
Getting Started
Latest Headlines
check the user ID and password

User ID:

password:

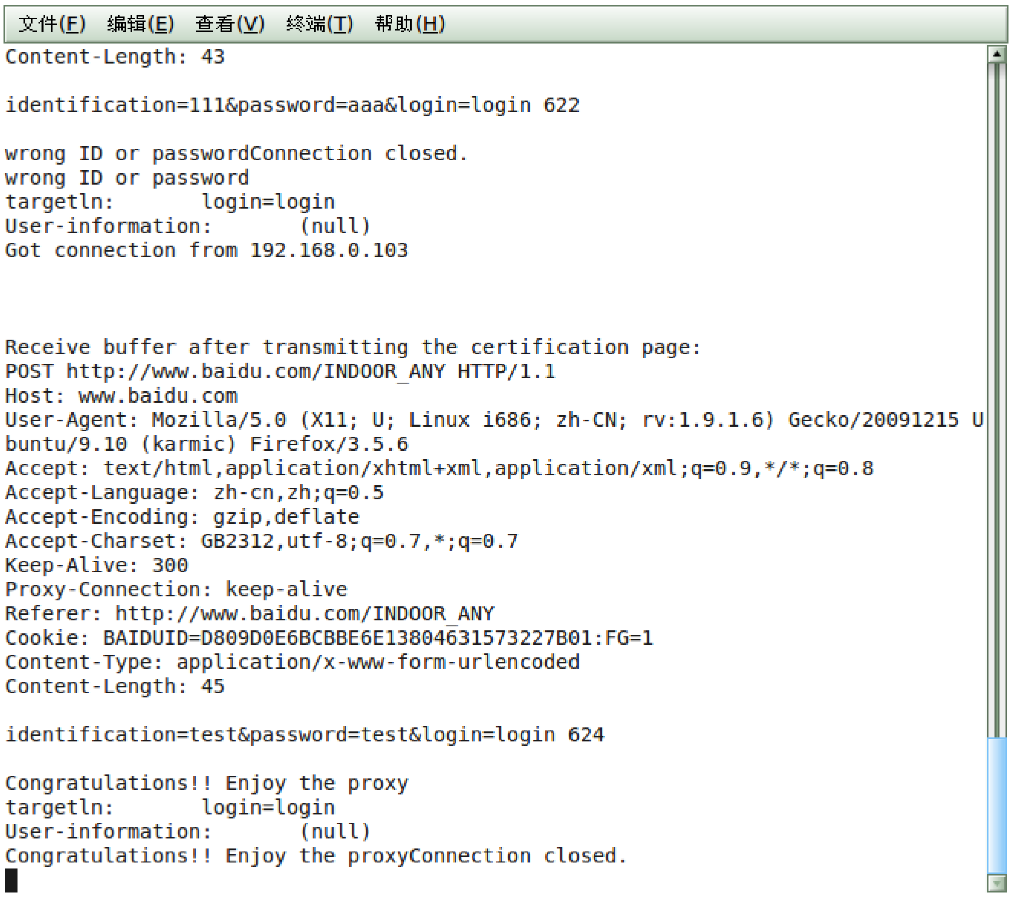
c) If the user inserts the right ID and password, the web page will show the corresponding words


文件(F) 编辑(E) 查看(V) 历史(S) 书签(B) 工具(T) 帮助(H)
http://www.baidu.com/INDOOR_ANY
Google
Most Visited
Getting Started
Latest Headlines
check the user ID and password

User ID:

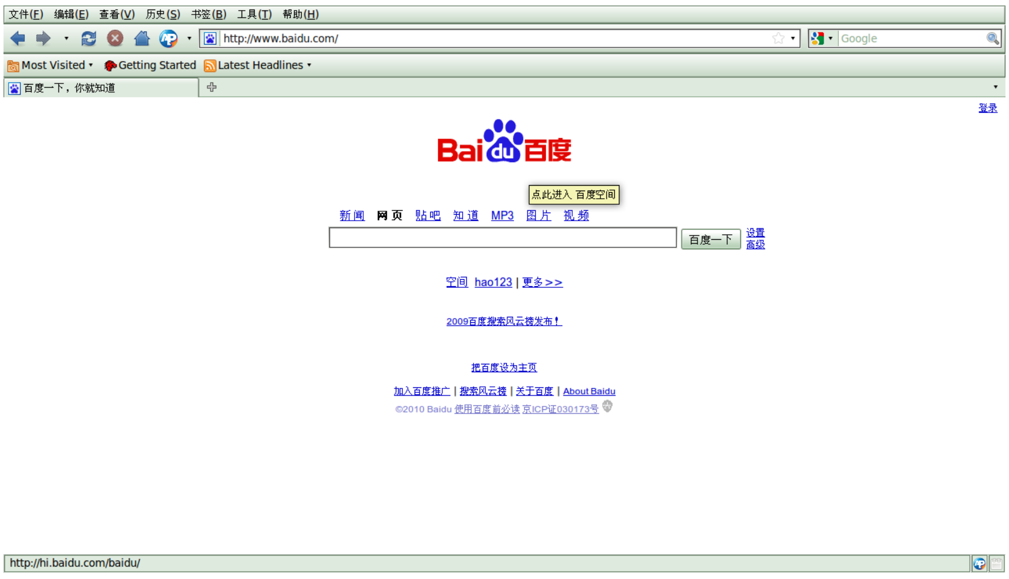
password:

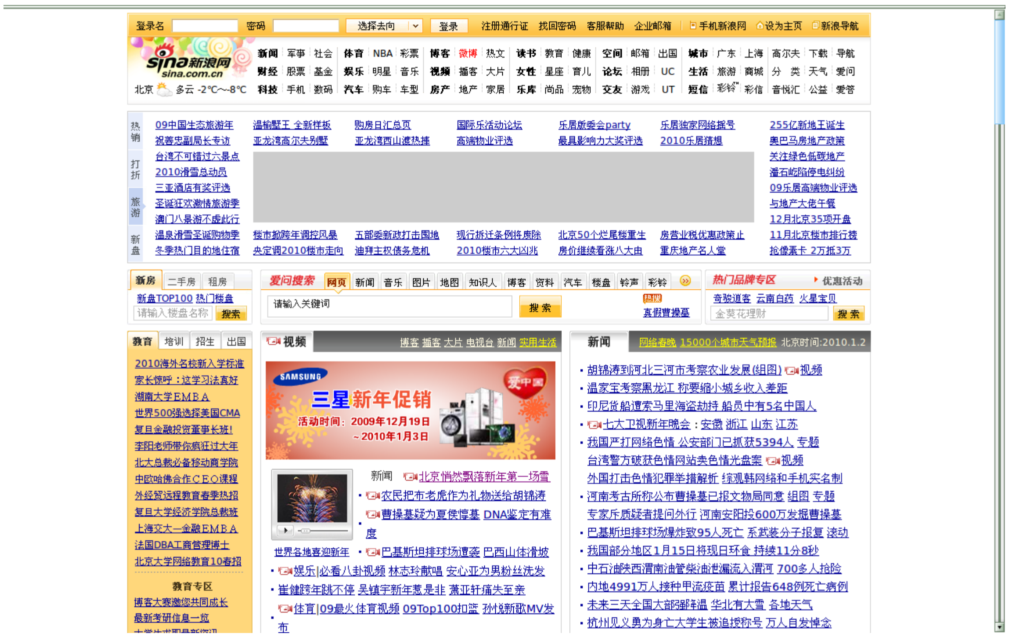
Congratulations!! Enjoy the proxy



```
Content-Length: 43

identification=111&password=aaa&login=login 622

wrong ID or passwordConnection closed.
wrong ID or password
targetln:       login=login
User-information:       (null)
Got connection from 192.168.0.103



Receive buffer after transmitting the certification page:
POST http://www.baidu.com/INDOOR_ANY HTTP/1.1
Host: www.baidu.com
User-Agent: Mozilla/5.0 (X11; U; Linux i686; zh-CN; rv:1.9.1.6) Gecko/20091215 U
buntu/9.10 (karmic) Firefox/3.5.6
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-cn,zh;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: GB2312,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Proxy-Connection: keep-alive
Referer: http://www.baidu.com/INDOOR_ANY
Cookie: BAIDUID=D809D0E6BCBBE6E13804631573227B01:FG=1
Content-Type: application/x-www-form-urlencoded
Content-Length: 45

identification=test&password=test&login=login 624

Congratulations!! Enjoy the proxy
targetln:       login=login
User-information:       (null)
Congratulations!! Enjoy the proxyConnection closed.
```

d) Then the user can insert the web site


文件(F) 编辑(E) 查看(V) 历史(S) 书签(B) 工具(T) 帮助(H)
http://www.baidu.com/
Google
Most Visited
Getting Started
Latest Headlines
百度一下，你就知道
登录
点此进入 百度空间
新闻 网页 贴吧 知道 MP3 图片 视频
百度一下
设置
高级
空间 hao123 | 更多>>
2009百度搜索风云榜发布！
把百度设为主页
加入百度推广 | 搜索风云榜 | 关于百度 | About Baidu
©2010 Baidu 使用百度前必读 京ICP证030173号
http://hi.baidu.com/baidu/


登录名 密码 选择去向 登录 注册通行证 找回密码 客服帮助 企业邮箱 手机新浪网 设为主页 新浪导航

sina新浪网 sina.com.cn 北京 多云 -2℃~-8℃

新闻 军事 社会 体育 NBA 彩票 博客 微博 热文 读书 教育 健康 空间 邮箱 出国 城市 广东 上海 高尔夫 下载 导航
财经 股票 基金 娱乐 明星 音乐 视频 播客 大片 女性 星座 育儿 论坛 相册 UC 生活 旅游 商城 分 类 天气 爱问
科技 手机 数码 汽车 购车 车型 房产 地产 家居 乐库 尚品 宠物 交友 游戏 UT 短信 彩铃 彩信 音悦汇 公益 爱答

热销：09中国生态旅游年 祝善忠副局长专访
打折：台湾不可错过六景点 2010滑雪总动员 三亚酒店有奖评选
旅游：圣诞狂欢激情旅游季 澳门八景游不虚此行
新盘：温泉滑雪圣诞购物季 冬季热门目的地住宿

温榆墅王 全新样板 亚龙湾高尔夫别墅 购房日汇总页 亚龙湾西山渡热捧 国际乐活动论坛 高端物业评选 乐居版委会party 最具影响力大奖评选 乐居独家网络摇号 2010乐居猜想

楼市掀跨年调控风暴 央定调2010楼市走向 五部委新政打击囤地 迪拜主权债务危机 现行拆迁条例将废除 2010楼市六大凶兆 北京50个烂尾楼重生 房价继续看涨八大由 房营业税优惠政策止 重庆地产名人堂

255亿新地王诞生 奥巴马房地产政策 关注绿色低碳地产 潘石屹陷停电纠纷 09乐居高端物业评选 与地产大佬午餐 12月北京35项开盘 11月北京楼市排行榜 抢傢素卡 2万抵3万

新房 二手房 租房 新盘TOP100 热门楼盘 请输入楼盘名称 搜索

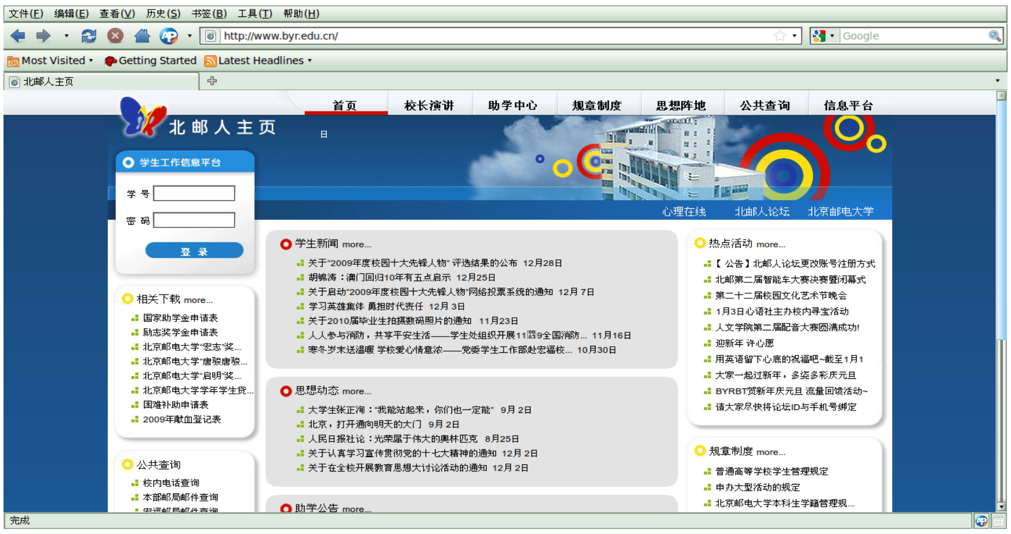
爱问搜索 网页 新闻 音乐 图片 地图 知识人 博客 资料 汽车 楼盘 铃声 彩铃 请输入关键词 搜 索 热搜 真假曹操墓

热门品牌专区 优惠活动 奇骏逍客 云南白药 火星宝贝 金葵花理财 搜 索

教育 培训 招生 出国
2010海外名校新入学标准
家长惊呼：这学习法真好
湖南大学EMBA
世界500强选择美国CMA
复旦金融投资董事长班!
李阳老师带你疯狂过大年
北大总裁必备移动商学院
中欧哈佛合作CEO课程
外经贸远程教育春季热招
复旦大学经济学院总裁班
上海交大—金融EMBA
法国DBA工商管理博士
北京大学网络教育10春招
教育专区
博客大赛邀您共同成长
最新考研信息一览

视频 博客 播客 大片 电视台 新闻 实用生活
三星新年促销 活动时间：2009年12月19日～2010年1月3日
新闻 北京悄然飘落新年第一场雪
· 农民把布老虎作为礼物送给胡锦涛
· 曹操墓疑为夏侯惇墓 DNA鉴定有难度
世界各地喜迎新年 · 巴基斯坦排球场遭袭 巴西山体滑坡
· 娱乐|必看八卦视频 林志玲献唱 安心亚为男粉丝洗发
· 崔健跨年跳不停 吴镇宇新年惹是非 萧亚轩痛失至亲
· 体育|09最火体育视频 09Top100扣篮 孙悦新歌MV发布

新闻 网络春晚 15000个城市天气预报 北京时间:2010.1.2
· 胡锦涛到河北三河市考察农业发展(组图) 视频
· 温家宝考察黑龙江 称要缩小城乡收入差距
· 印尼货船遭索马里海盗劫持 船员中有5名中国人
· 七大卫视新年晚会：安徽 浙江 山东 江苏
· 我国严打网络色情 公安部门已抓获5394人 专题
台湾警方破获色情网站卖色情光盘案 视频
外国打击色情犯罪举措解析 综观韩网络和手机实名制
· 河南考古所称公布曹操墓已报文物局同意 组图 专题
专家斥质疑者提问外行 河南安阳投600万发掘曹操墓
· 巴基斯坦排球场爆炸致95人死亡 系武装分子报复 滚动
· 我国部分地区1月15日将现日环食 持续11分8秒
· 中石油陕西渭南油管柴油泄漏流入渭河 700多人抢险
· 内地4991万人接种甲流疫苗 累计报告648例死亡病例
· 未来三天全国大部降温 华北有大雪 各地天气
· 杭州见义勇为身亡大学生被追授称号 万人自发悼念


文件(F) 编辑(E) 查看(V) 历史(S) 书签(B) 工具(T) 帮助(H)
http://www.byr.edu.cn/
Most Visited Getting Started Latest Headlines
北邮人主页

首页 校长演讲 助学中心 规章制度 思想阵地 公共查询 信息平台
北邮人主页
心理在线 北邮人论坛 北京邮电大学

学生工作信息平台
学 号
密 码
登 录

相关下载 more...
国家助学金申请表
励志奖学金申请表
北京邮电大学"宏志"奖...
北京邮电大学"唐骏唐骏...
北京邮电大学"启明"奖...
北京邮电大学学年学生贷...
困难补助申请表
2009年献血登记表

公共查询
校内电话查询
本部邮局邮件查询

学生新闻 more...
关于"2009年度校园十大先锋人物"评选结果的公布 12月28日
胡锦涛：澳门回归10年有五点启示 12月25日
关于启动"2009年度校园十大先锋人物"网络投票系统的通知 12月 7日
学习英雄集体 勇担时代责任 12月 3日
关于2010届毕业生拍摄数码照片的通知 11月23日
人人参与消防，共享平安生活——学生处组织开展11?9全国消防... 11月16日
寒冬岁末送温暖 学校爱心情意浓——党委学生工作部赴宏福校... 10月30日

思想动态 more...
大学生张正洵："我能站起来，你们也一定能" 9月 2日
北京，打开通向明天的大门 9月 2日
人民日报社论：光荣属于伟大的奥林匹克 8月25日
关于认真学习宣传贯彻党的十七大精神的通知 12月 2日
关于在全校开展教育思想大讨论活动的通知 12月 2日

助学公告 more...

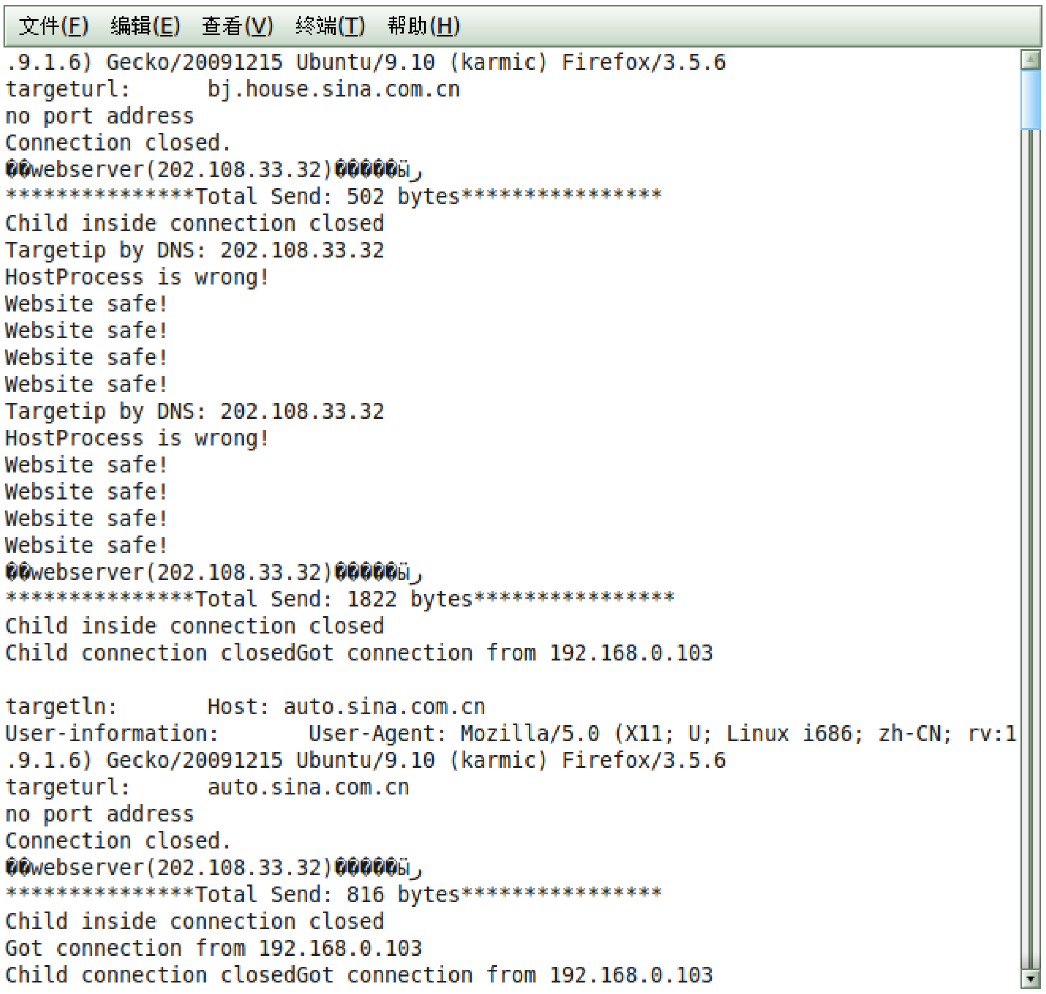
热点活动 more...
【公告】北邮人论坛更改账号注册方式
北邮第二届智能车大赛决赛暨闭幕式
第二十二届校园文化艺术节晚会
1月3日心语社主办校内寻宝活动
人文学院第二届配音大赛圆满成功!
迎新年 许心愿
用英语留下心底的祝福吧~截至1月1
大家一起过新年，多姿多彩庆元旦
BYRBT贺新年庆元旦 流量回馈活动~
请大家尽快将论坛ID与手机号绑定

规章制度 more...
普通高等学校学生管理规定
申办大型活动的规定
北京邮电大学本科生学籍管理规...

完成

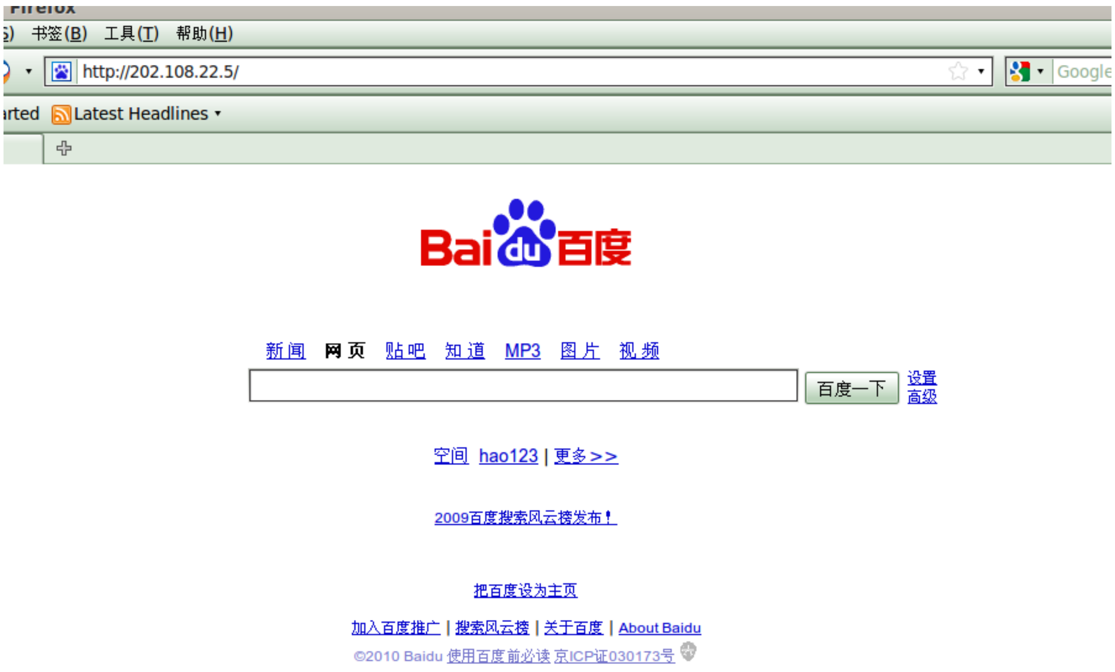
文件(F) 编辑(E) 查看(V) 历史(S) 书签(B) 工具(T) 帮助(H)
http://www.tudou.com/programs/view/CKbGDy6lfCA/
Most Visited Getting Started Latest Headlines
土豆网_每个人都是生活的导演...
新《三国》CG视频曝光 火烧赤...
土豆网 Tudou.com
视频
搜 索
注册 登录 便利豆单(0) 我看过的 土豆软件 我的 上传
首页 视频 豆单 专区 NEW 豆丸 黑豆 社区 标签 小组 相册
影视 播客 娱乐 动画 音乐 体育 汽车 财富 女性 游戏 乐活 科技 教育
CG
土豆网
00:06
00:57
相关视频
不知道看什么？豆丸一下
正在从 218.60.33.14 传送数据...

e) The corresponding terminal shot is shown below

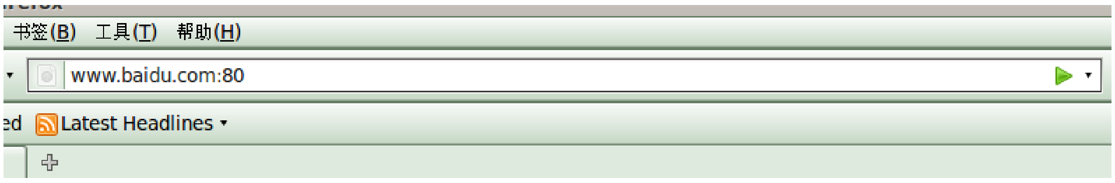


```
.9.1.6) Gecko/20091215 Ubuntu/9.10 (karmic) Firefox/3.5.6
targeturl:      bj.house.sina.com.cn
no port address
Connection closed.
��webserver(202.108.33.32)������ʲ¸
***************Total Send: 502 bytes****************
Child inside connection closed
Targetip by DNS: 202.108.33.32
HostProcess is wrong!
Website safe!
Website safe!
Website safe!
Website safe!
Targetip by DNS: 202.108.33.32
HostProcess is wrong!
Website safe!
Website safe!
Website safe!
Website safe!
��webserver(202.108.33.32)������ʲ¸
***************Total Send: 1822 bytes****************
Child inside connection closed
Child connection closedGot connection from 192.168.0.103

targetln:       Host: auto.sina.com.cn
User-information:       User-Agent: Mozilla/5.0 (X11; U; Linux i686; zh-CN; rv:1
.9.1.6) Gecko/20091215 Ubuntu/9.10 (karmic) Firefox/3.5.6
targeturl:      auto.sina.com.cn
no port address
Connection closed.
��webserver(202.108.33.32)������ʲ¸
***************Total Send: 816 bytes****************
Child inside connection closed
Got connection from 192.168.0.103
Child connection closedGot connection from 192.168.0.103
```

f) The user can also visit the website with IP address

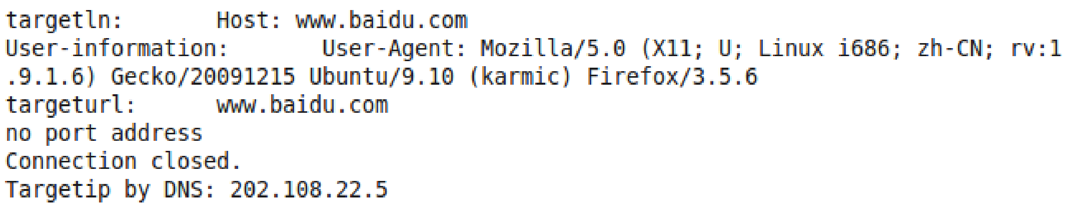

Firefox
) 书签(B) 工具(T) 帮助(H)
http://202.108.22.5/
Google
rted Latest Headlines
Bai du 百度
新闻 网页 贴吧 知道 MP3 图片 视频
百度一下
设置
高级
空间 hao123 | 更多>>
2009百度搜索风云榜发布！
把百度设为主页
加入百度推广 | 搜索风云榜 | 关于百度 | About Baidu
©2010 Baidu 使用百度前必读 京ICP证030173号

g) Website with port number is also allowed



```
www.baidu.com:80
```

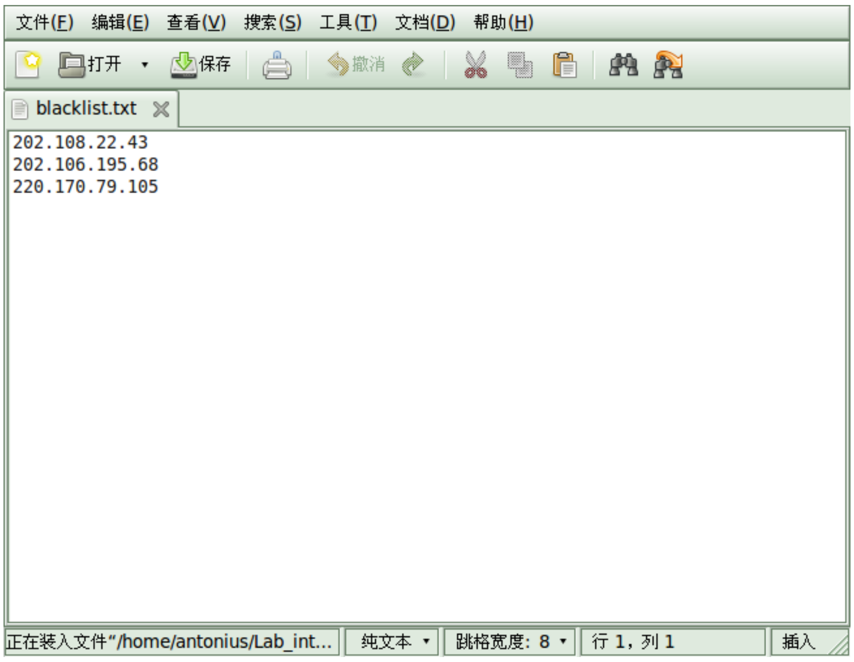
h) Host information can be viewed



```
targetln:       Host: www.baidu.com
User-information:       User-Agent: Mozilla/5.0 (X11; U; Linux i686; zh-CN; rv:1
.9.1.6) Gecko/20091215 Ubuntu/9.10 (karmic) Firefox/3.5.6
targeturl:      www.baidu.com
no port address
Connection closed.
Targetip by DNS: 202.108.22.5
```

i) Blacklist can be set; here we use www.byr.edu.cn (123.127.134.58)

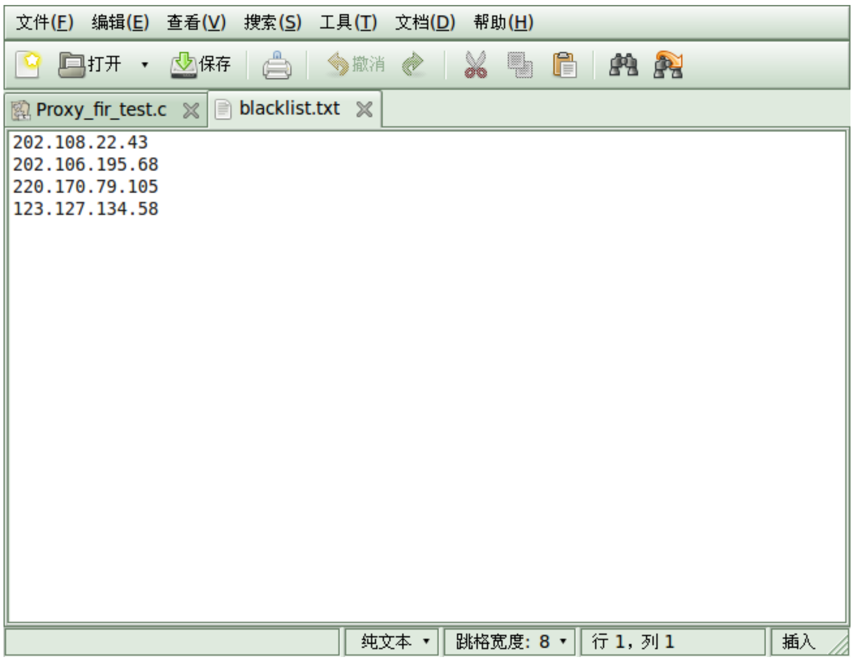
First the file does not contain the IP address of byr.edu.cn, and then it can be accessed.



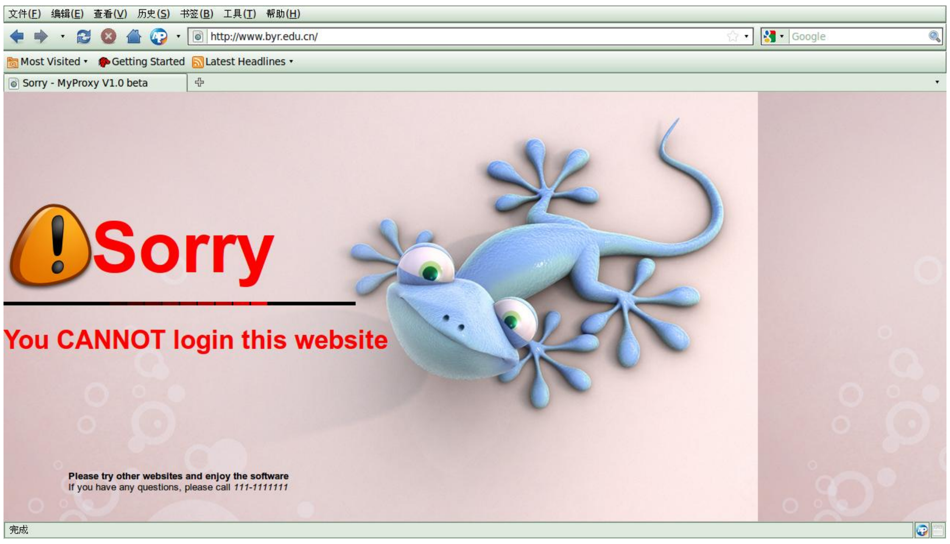
```
202.108.22.43
202.106.195.68
220.170.79.105
```

```
正在装入文件"/home/antonius/Lab_int...   纯文本   跳格宽度: 8   行 1, 列 1   插入
```

Then we add the IP address of byr.edu.cn to the blacklist. And the website will not be allowed to visit now.



```
202.108.22.43
202.106.195.68
220.170.79.105
123.127.134.58
```


文件(F) 编辑(E) 查看(V) 历史(S) 书签(B) 工具(T) 帮助(H)
http://www.byr.edu.cn/
Google
Most Visited
Getting Started
Latest Headlines
Sorry - MyProxy V1.0 beta
完成

Sorry

You CANNOT login this website

Please try other websites and enjoy the software
If you have any questions, please call 111-1111111

6. Summary and Conclusion

We divide the job into several parts as we mentioned in 3.1. The main body which can realize the proxy function include modules a), c), d), f) were finished by Liu Xiao. As the modules a), c), d), f) have a close relationship, we think these parts should be finished together. To finish these functions together will make it easy to run the function and check the error. The additional functions b), e) were written by Song Yutian. These two parts was finished separately. We add the additional function to the main body by using function or a while() loop which will not disrupt the main function’s performance. Because there have some values are used in different parts, we use overall values to make them available for different parts.

The basic function performs very well. We allow the client visit most website outside the blacklist with a high speed. Send data is written in a function TCPSEND. In this function, we use a while loop to keep sure the data will be the client completely. Because we don’t have any function to prevent spend a long time to receive a single package. If the client visits a busy website the webpage will take a very long time to display on the screen. So add a time control function will make the proxy process more fluent.

The additional parts can realize the requirement of the experiment. The blacklist and authentication performs well and robust. But there still have some problem like after the client login the webpage will not go to the target website automatically. This can be improved by set up a socket to realize the work. But as the time limit and afraid to make the hole program broke down I give up improving the code. Also a client register can make it better for multi-client work.

Appendix: Source Codes

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <string.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <sys/socket.h>
#include <sys/wait.h>
#include <arpa/inet.h>
#include <netdb.h>
#include<ctype.h>

#define IE_REQUEST_LEN 2048
#define MAX_CHARBUFF 20480
#define BACKLOG 10

int sock_client_found, sock_client_SR, sock_server_SR;        /* listen on sock_client_found,
send and receive on sock_client_SR */
struct sockaddr_in proxy_addr;
struct sockaddr_in clnt_addr;
struct sockaddr_in server_addr;
struct in_addr **addr_test;
pid_t pid;
int sin_size;
int k=1;
char recvbuffer[MAX_CHARBUFF];
char sendbuffer[MAX_CHARBUFF];
char checkbuffer[MAX_CHARBUFF];
char *targetip;
```

```
void clientRecvTest();
void ProxySentTest();
int HostProcess(char * SourceBuf,int DataLen);
int TCPsend(int s,const char *buf,int len,int flags);
int TCPrecv(int s,char *buf,int len,int flags);
void blacklist();
int n=1;
int tcpsendnum=0;


int main()
{

        int k=0;
        char *check="identification=";
        char *check1="password=";
        char *buffer,*buffer1;
        char *ID,*password;

        fputs("Proxy start\n",stdout);
    if((sock_client_found = socket(AF_INET, SOCK_STREAM, 0))==-1)
        {
            perror("Socket: ");
            exit(1);
      }
        memset(&proxy_addr,0,sizeof(proxy_addr));
    proxy_addr.sin_family = AF_INET;
    proxy_addr.sin_addr.s_addr = INADDR_ANY;
        proxy_addr.sin_port = htons(8080);
    if(bind(sock_client_found,(struct sockaddr*)&proxy_addr,sizeof(struct sockaddr)) == -1)
        {
            perror("Bind: ");
            exit(1);
        }
        if(listen(sock_client_found,BACKLOG)== -1)
        {
            perror("Listen: ");
            exit(1);
        }




    while(1)
        {
            setsockopt(sock_client_found,SOL_SOCKET,SO_REUSEADDR,&k,sizeof(int));
            sin_size = sizeof(struct sockaddr_in);

            if ((sock_client_SR = accept(sock_client_found, (struct sockaddr
*)&clnt_addr,&sin_size)) == -1)
                {
                    perror("Accept: ");
                    continue;
                }
            printf("Got connection from %s\n", inet_ntoa(clnt_addr.sin_addr));

            if(k==0)
            {
             memset(checkbuffer,0,MAX_CHARBUFF);
             char reply[]="<html><head><title>check the user ID and password</title><meta
http-equiv=Content-Type content=\"text/html;charset=gb2312\"></head><body><form id=
\"loginForm\"name=\"login\" action=\"INDOOR_ANY\"method=\"post\"><p><label>User
ID:<input type=\"text\"name=\"identification\"/></label></p><p><label>password:<input
type=\"password\"name=\"password\"/></label></p><input name= \"login\" type=\"submit\"
value=\"login\"class=\"submit\"></form></body></html>";
             tcpsendnum = TCPsend(sock_client_SR,reply,strlen(reply),0);
             int Checkrecv;
             Checkrecv= recv(sock_client_SR,checkbuffer,MAX_CHARBUFF,0);
               if(Checkrecv>0)
               {
                printf("\n\n\nReceive buffer after transmitting the certification page:
\n%s %d\n\n",checkbuffer,strlen(checkbuffer));
                //perror("recv");
               }

             buffer=(char*)malloc(sizeof(char)*(strlen(checkbuffer)+1));
             strcpy(buffer,checkbuffer);

             buffer1=(char*)malloc(sizeof(char)*(strlen(checkbuffer)+1));
             strcpy(buffer1,checkbuffer);

             ID=strtok(strstr(buffer,check),"&");
             password=strtok(strstr(buffer1,check1),"&");

             int isid = 1, ispassword= 1;


             if(ID!= NULL)
             {
```

```
        isid = strcmp("identification=test",ID);
      }
      else
      {
printf("\nID:\t%s\n",ID);
      }


      if(password!= NULL)
      {
        ispassword = strcmp("password=test",password);
      }
      else
      {
printf("\npassword:\t%s\n",password);
      }


      if(isid!=0 || ispassword !=0)
        {
          printf("wrong ID or password");
        }
      else
        {
          printf("Congratulations!! Enjoy the proxy");
          char* tempsend = "Congratulations!! Enjoy the proxy";
          tcpsendnum = TCPsend(sock_client_SR,tempsend, strlen(tempsend),0);
          k=k+1;
        }
      close(sock_client_SR);
      }


  pid = fork();
    if (pid == 0)
        {
                clientRecvTest();
                ProxySentTest();
                fputs("Child connection closed",stdout);
        }
    else if (pid <0)
          {
                printf("Fork error!\n");
                exit(-1);
```

```
                                }
                    else if (pid >0)
                            {
                                    close(sock_client_SR);
                                    printf("Connection closed.\n");
                            }
            }
      return 0;

}



void clientRecvTest()
{
    char clntName[INET_ADDRSTRLEN];
    inet_ntop(AF_INET,&clnt_addr.sin_addr.s_addr,clntName,sizeof(clntName));
    int numBytesRcvd;
    int numBytesSent;
    numBytesRcvd = recv(sock_client_SR,recvbuffer,MAX_CHARBUFF,0);

    shutdown(sock_client_SR,SHUT_RD);

    // fputs(recvbuffer,stdout);
}

void ProxySentTest()
{
      int flag=0;
      int numBytesRecvS;
      server_addr.sin_family = AF_INET;
      if((sock_server_SR = socket(AF_INET, SOCK_STREAM, 0))==-1)
              {
                      perror("Socket: ");
                      exit(1);
              }

          if(HostProcess(recvbuffer, strlen(recvbuffer))== 0);
        {
                  printf("HostProcess is wrong!\n");

        }
```

```
  blacklist();

if (connect(sock_server_SR, (struct sockaddr *)&server_addr, sizeof(server_addr))== -1)
      {
             perror("connect ");
             exit(1);
      }




numBytesRecvS = TCPsend(sock_server_SR,recvbuffer, strlen(recvbuffer),0);

char      dest[1000];
numBytesRecvS=1;
int senlen=0;
memset(dest,0,1000);
numBytesRecvS=recv(sock_server_SR,dest,sizeof(dest)-1,0);
if(numBytesRecvS==0)
{
      printf("connection with webserver(%s)close\n",inet_ntoa(clnt_addr.sin_addr));
      shutdown(sock_server_SR,SHUT_RD);
      return;
  }
  else if(numBytesRecvS==-1)
{
  shutdown(sock_server_SR,SHUT_RD);
  return;
}


char szRespondLine[1000]={0};
int count = 0;

char *pch;
char *pdest;
pdest = strchr(dest,'\n');    //'\n' first position
count = pdest - dest+1;
pdest++; // go over '\n'
pch = strchr(pdest,'\n');    // '\n' second position
count +=(pch -pdest+1);
strncpy(szRespondLine,dest,count);
szRespondLine[count]='\0';
```

```
    senlen = TCPsend(sock_client_SR,dest,numBytesRecvS,0);
    if(senlen==0)
    {
        printf("conection with (%s) close...\n",inet_ntoa(clnt_addr.sin_addr));
        return;
    }
      else if(senlen==-1)
    {
        return;
    }



    while(numBytesRecvS>0)
    {
          memset(dest,0,1000);
          numBytesRecvS=recv(sock_server_SR,dest,sizeof(dest),0);
          if(numBytesRecvS==0)
  {
    printf("connection with webserver(%s)close\n",inet_ntoa(server_addr.sin_addr));
    shutdown(sock_server_SR,SHUT_RD); //close receive
    break;
  }
          else if(numBytesRecvS==-1)
  {
        break;
  }

  senlen = TCPsend(sock_client_SR,dest,numBytesRecvS,0);    //send the data to the client
  if(senlen==0)
  {
          printf("connection with (%s) closed...\n",inet_ntoa(clnt_addr.sin_addr));
          break;
  }
  else if(senlen==-1)
  {
        break;
  }
}
tcpsendnum += senlen;

      close(sock_client_SR);
      close(sock_server_SR);
      printf("***************Total Send: %d bytes****************\n", tcpsendnum);
```

```
        fputs("Child inside connection closed\n",stdout);
}


int HostProcess(char * SourceBuf,int DataLen)
{


    char *search="Host";
    char *search1="User-Agent";

    char *targetln,*targetla,*targetan,*targetport,*targeturl;
    char *buffer,*buffer1;
    struct hostent *targetIpByDN;
    char * SourceBufe;

    SourceBufe = SourceBuf;

    server_addr.sin_family = AF_INET;

    buffer=(char*)malloc(sizeof(char)*(strlen(SourceBufe)+1));
    strcpy(buffer,SourceBufe);

    buffer1=(char*)malloc(sizeof(char)*(strlen(SourceBufe)+1));
    strcpy(buffer1,SourceBufe);

    targetln=strtok(strstr(buffer,search),"\r\n");
    printf("\ntargetln:\t%s\n",targetln);

    targetan=strtok(strstr(buffer1,search1),"\r\n");
    printf("User-information:\t%s\n",targetan);

    targeturl=strtok((char*)strcasestr(targetln,":")+2,"\r\n");
    targetip=(char*)malloc(sizeof(char)*(strlen(targeturl)+1));


    printf("targeturl:\t%s\n",targeturl);

    strcpy(targetip,targeturl);

    targetport=strstr(targeturl,":");

    if(targetport!= NULL)
    {
            targetport+=1;
            printf("targetport:\t%s\n",targetport);
            server_addr.sin_port = htons((u_short)atoi(targetport));
    }
    else
    {
            server_addr.sin_port = htons(80);
            printf("no port address\n");
    }

    targetip=strtok(targetip,":");
    if(*targetip >= '0' && *targetip <= '9')
    {
            printf("targetip:\t%s\n",targetip);
            server_addr.sin_addr.s_addr = inet_addr(targetip);
    }
    else
    {
                targetIpByDN =(struct hostent *)gethostbyname(targetip);
                if(!targetIpByDN)
                {
                        printf("get sever formation is wrong\n");
                        printf("%s\n",targetip);
                        return -1;
                }
            addr_test = ((struct in_addr **)targetIpByDN->h_addr_list);
            printf("Targetip by DNS: %s \n", inet_ntoa(*addr_test[0]));
            server_addr.sin_addr.s_addr = inet_addr(inet_ntoa(*addr_test[0]));
    }

    return 1;
}



int TCPrecv(int s,char *buf,int len,int flags)
{

  int nRev=0,recvCount=0;
  int length =len;

  if(buf==NULL)
    return 0;
    printf("\nbull error3\n");

  // receive data
  while(length>0)
  {
    nRev =recv(s,buf+recvCount,length,flags);

    printf("receive number: %d", nRev);

    if(nRev==0)
    {
      printf("\nerror for nRev = 0\n");
      break;
    }
    if(nRev==-1)//error
    {
      printf("\nNW error\n");
      break;
    }
    printf("\nNo error\n");
    length-=nRev;
    recvCount+=nRev;
    printf("\n %d %d\n",length,recvCount);
  }
  printf("\nNo error2\n");
  return recvCount; //return the number of data received
}


int TCPsend(int s,const char *buf,int len,int flags)
{
  int n=0,sendCount=0;
  int length =len;
  if(buf==NULL)
    return 0;
  while(length>0)
  {
    n=send(s,buf+sendCount,length,flags); //send data
    if(n==0)
    {
      break;
    }
    if(n==-1)//error
    {
      break;

    }
    length-=n;
    sendCount+=n;
  }

  return sendCount; // return the number send to the client
}

//blacklist
void blacklist()
{
    char blip[20];
    FILE *inFile;

    inFile = fopen("blacklist.txt","r");

    if(inFile == NULL)
    {
        printf("\n Failed to open blacklist.\n");
        exit(1);
    }
    while (fgets(blip,20,inFile)!=NULL)
    {
        strtok(blip,"\n");

        int isblack;

        if(*targetip >= '0' && *targetip <= '9')
        {
          isblack = strcmp(targetip,blip);
        }
        else
        {
          isblack = strcmp(inet_ntoa(*addr_test[0]),blip);//IP get by gethostbyname
        }

        if(isblack == 0)
        {
                char send[250];
                FILE *caution;
                caution = fopen("blacklisttest.HTML","r");
                char caubuff[20480];
                char picbuff[20480];

                if(caution == NULL)
                {
                    printf("\n Failed to open blacklist.\n");
                    exit(1);
                }

                while(fgets(send,250,caution)!=NULL)
                {
                  strcat(caubuff,send);
                }
                fclose(caution);



                //char reply[]="<html>You are logining a wesite on the blacklist</html>";
                int numBytesSent = TCPsend(sock_client_SR,caubuff,strlen(caubuff),0);
                int numBytesRecv=    recv(sock_client_SR,picbuff,MAX_CHARBUFF,0);
                printf("Website in BLACK LIST!\n");
                //printf("\n\n\nReceive buffer after black list: \n%s %d\n\n",picbuff,strlen(picbuff));


                close(sock_server_SR);
        }
        else
        {
                printf("Website safe!\n");//connection,resv,send
        }
      }
        fclose(inFile);
}
```