厦門大學

信息学院软件工程系

《计算机网络》实验报告

题　　目　实验七　代理服务器软件

班　　级　软件工程 2020 级数媒班

姓　　名　陈鑫蕾

学　　号　22920202202877

实验时间　2022 年 12 月 21 日

2022 年 12 月 21 日

填写说明

1、本文件为 Word 模板文件，建议使用 Microsoft Word 2019 打开，在可填写的区域中如实填写；

2、填表时勿破坏排版，勿修改字体字号，打印成 PDF 文件提交；

3、文件总大小尽量控制在 1MB 以下，最大勿超过 5MB；

4、应将材料清单上传在代码托管平台上；

5、在实验课结束 14 天内，按原文件发送至课程 FTP 指定位置。

1 实验目的

通过完成实验，掌握基于 RFC 应用层协议规约文档传输的原理，实现符合接口且能和已有知名软件协同运作的软件。

2 实验环境

操作系统：linux

虚拟机：VMware

3 实验结果

本次实验在学习之后对附录二程序做注解，分析其程序结构，做适当的笔记。（详见代码）

部分代码截图：

```
754  }
755
756  int main(int argc, char* argv[])
757  {
758      int ret;
759      log_file = stdout; //将日志信息打印到控制台
760
761      //初始化身份验证模式以及用户名和密码
762      auth_type = NOAUTH;
763      arg_username = "user";
764      arg_password = "pass";
765
766      //初始化多线程锁
767      pthread_mutex_init(&lock, NULL);
768
769      //设置中断信号signal, 当客户端关闭时,客户端会向服务端发送SIGPIPE信号表明连接断开
770      signal(SIGPIPE, SIG_IGN);
771
772      //解析程序命令行参数,一共有-n,-u,-p,-l,-a,-h和-d等7个选项, 其中只有-h和-d命令选项不用带参数
773      while ((ret = getopt(argc, argv, "n:u:p:l:a:hd")) != -1)
774      {
775          switch (ret)
776          {
777              //设置"守护进程"的模式
778          case 'd':
779          {
780
781              daemon_mode = 1;
782              daemonize();
783              break;
784          }
785          //把-n命令选项后面带的参数赋给port,表示用户设置的监听端口
```

该软件的主要的函数是 app_loop 函数，此函数实现了程序的循环。

首先通过创建 socket，设置为流模式，

```
int app_loop()
{
    int sock_fd, net_fd;
    int optval = 1;
    struct sockaddr_in local, remote;
    socklen_t remotelen;

    //创建socket,设置为流模式
    if ((sock_fd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
    {
        log_message("socket()");
        exit(1);
    }

    //设置socket状态,SO_REUSEADDR表示允许在bind ()过程中本地地
    if (setsockopt(sock_fd, SOL_SOCKET, SO_REUSEADDR, (char*
        sizeof(optval)) < 0)
    {
        log_message("setsockopt()");
        exit(1);
    }
```

接着通过 bind 函数，绑定 socket

```
    //使用bind()来绑定socket
    if (bind(sock_fd, (struct sockaddr*)&local, sizeof(local
    {
        log_message("bind()");
        exit(1);
    }
```

通过 listen 函数进行监听，并设置监听队列长为 25

```
//启动监听函数,获取客户连接信息,并设置监听队列长度为25
if (listen(sock_fd, 25) < 0)
{
    log_message("listen()");
    exit(1);
}

remotelen = sizeof(remote);
memset(&remote, 0, sizeof(remote));
```

通过 while(1)的死循环，保证程序一直运行。

使用 accept 函数接受请求，并在处理请求时，调用其他封装的函数，读取 1 个字节，判断 SOCKS 版本号是 4 还是 5，交由相应函数处理。

```
while (1)
{
    //使用accept()函数来处理客户端tcp连接请求
    if ((net_fd =
        accept(sock_fd, (struct sockaddr*)&remote,
            &remotelen)) < 0)
    {
        log_message("accept()");
        exit(1);
    }

    int one = 1;
    setsockopt(sock_fd, SOL_TCP, TCP_NODELAY, &one, size

    //创建子线程来处理该客端户的代理服务
    if (pthread_create(&worker, NULL, &app_thread_proces
        (void*)&net_fd) == 0)
    {
        pthread_detach(worker);
    }
    else
    {
        log_message("pthread_create()");
    }
}
```

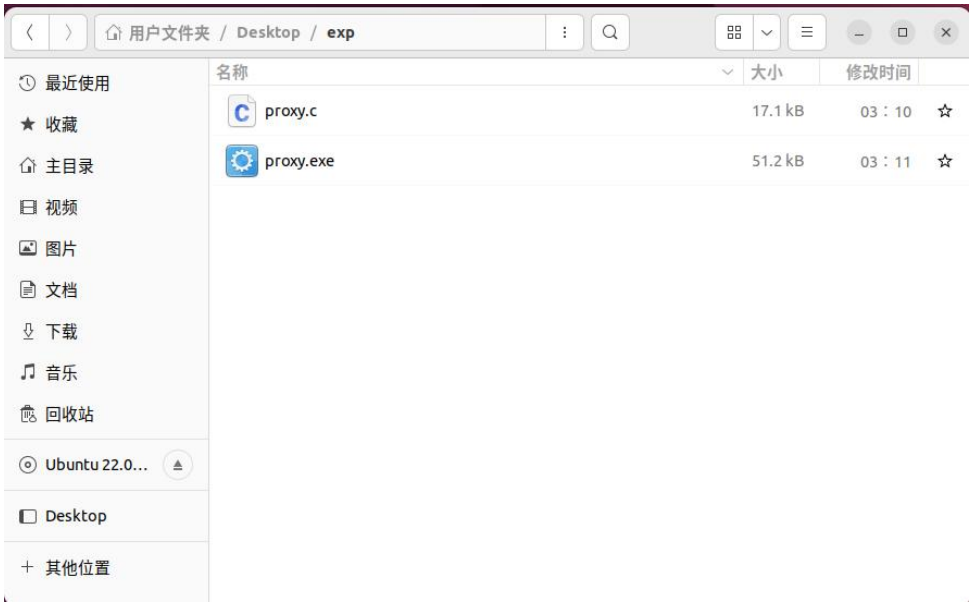
在 app_thread_process 函数中，针对不同 SOCKS 版本进行不同的操作来访问外网

```
//程序子进程处理函数
void* app_thread_process(void* fd)
{
    int net_fd = *(int*)fd;
    int version = 0;
    int inet_fd = -1;
    //为子进程创建socks4或5的socks代理服务
    char methods = socks_invitation(net_fd, &version);

    //根据socks4或socks5,进行不同的操作来访问外网
    switch (version)
    {
    case VERSION5:
    {
        //进行身份验证
        socks5_auth(net_fd, methods);
        int command = socks5_command(net_fd);

        //进行连接对应的IP地址
        if (command == IP)
        {
            char* ip = socks_ip_read(net_fd);
            unsigned short int p = socks_read_port(net_fd);

            inet_fd = app_connect(IP, (void*)ip, ntohs(p));
            if (inet_fd == -1)
            {
                app_thread_exit(1, net_fd);
            }
            socks5_ip_send_response(net_fd, ip, p);
```

在 linux 系统下，使用命令 gcc proxy.c -pthread -Wno -unused-result -g -std=gnu99-Wall -o proxy.exe 对代码进行编译。

```
cutie@cutie-virtual-machine:~/Desktop/exp$ gcc proxy.c -pthread -Wno-unuse-resul
t -g -std=gnu99 -Wall -o proxy.exe
```

编译结果如下：



运行 proxy.exe——打开代理服务器，监听 1080 端口：

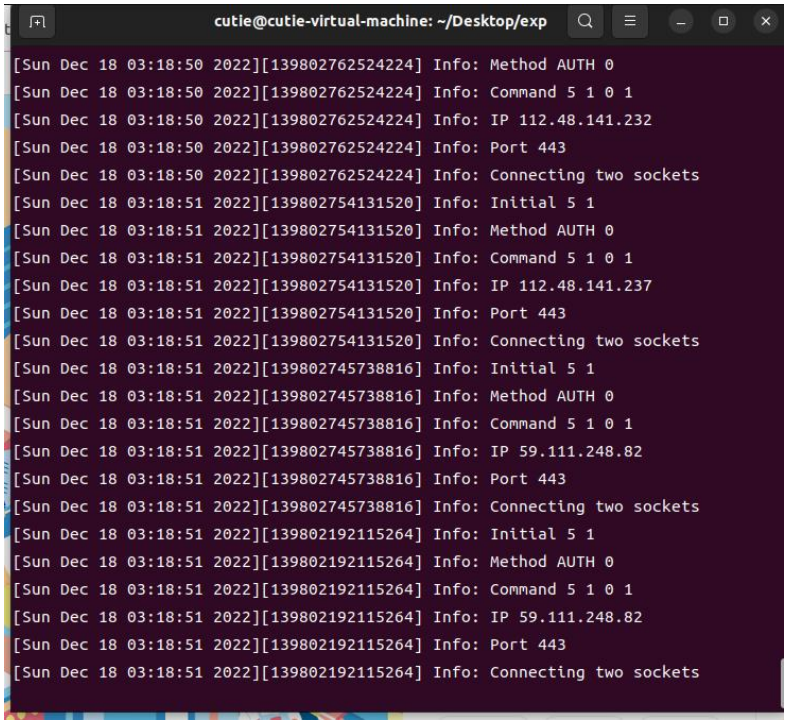
```
cutie@cutie-virtual-machine:~/Desktop/exp$ ./proxy.exe
[Sun Dec 18 03:13:13 2022][140430935553856] Info: Starting with authtype 0
[Sun Dec 18 03:13:13 2022][140430935553856] Info: Listening port 1080...
```

查看本机 ip 地址

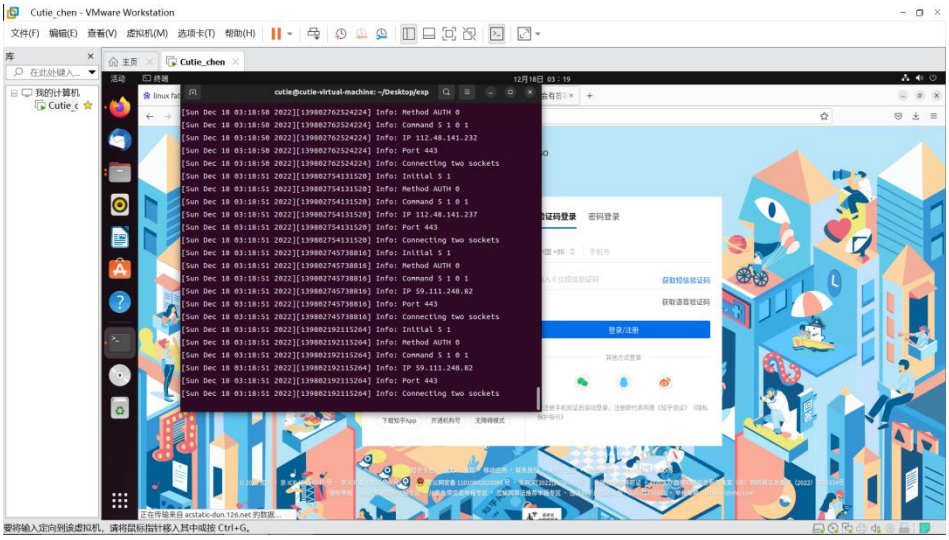
```
cutie@cutie-virtual-machine:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.116.128  netmask 255.255.255.0  broadcast 192.168.116
.255
        inet6 fe80::d3da:dc47:2809:7419  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:42:28:74  txqueuelen 1000  (以太网)
        RX packets 45214  bytes 52654366 (52.6 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12332  bytes 1641533 (1.6 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (本地环回)
        RX packets 1491  bytes 162560 (162.5 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1491  bytes 162560 (162.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

设置浏览器的代理服务器的 ip 地址和端口号。

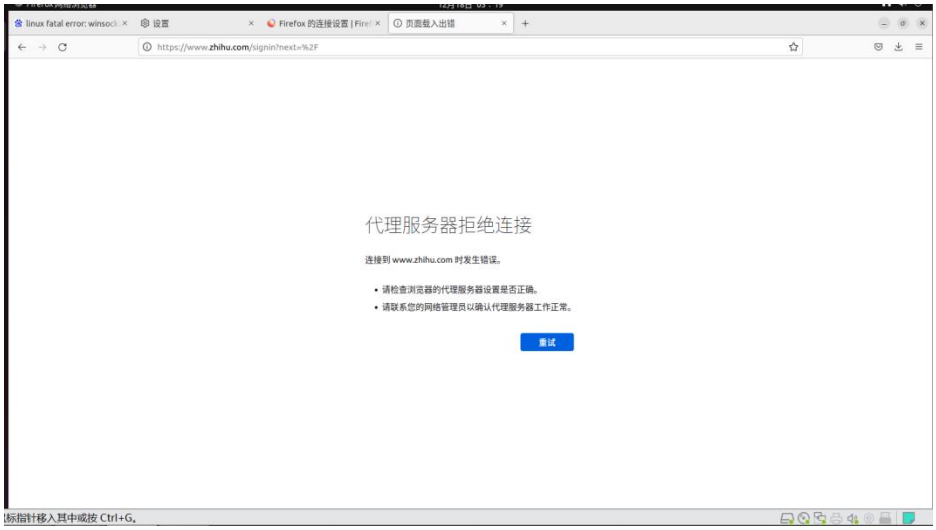
设置完成后程序运行情况：



点击其他网页测试：



关闭代理服务器测试：



网页无法打开，测试成功。

4 实验代码

本次实验的代码已上传于以下代码仓库：https://gitee.com/Cutie_Chen/computer-network

5 实验总结

本次实验主要是对附录二的程序进行了阅读学习，并做了相应的注释。然后在 linux 系统下进行编译实验。加强了我对代理服务器的理解，也更加熟悉了 linux 的操作。