《计算机网络协议开发》实验报告

第三次实验： 逆向工程套接字编程实验

姓名： 吕玉龙

学号： 191220076

2019 级计算机科学与技术院/系

邮箱： 1931015836@qq.com

时间：2022.3.25

一、实验目的

理解协议的逆向分析方法并掌握客户端套接字编程。

二、实验内容

设计方式：使用 socket 进行 tcp 连接的建立。

```
int main(int argc, char* argv[]){
    int sockfd;
    struct sockaddr_in servaddr;
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    bzero(&servaddr, sizeof(servaddr));
    servaddr.sin_family = AF_INET;
    servaddr.sin_port = htons(DEST_PORT);
    servaddr.sin_addr.s_addr= inet_addr(DEST_IP);
    int connect_rt = connect(sockfd, (struct sockaddr *) &servaddr, sizeof(servaddr));
    if (connect_rt < 0){
        printf("connect failed\n");
    }
    else{
        Page2toPage1(sockfd);
    }
    return 0;
}
```

整体的逻辑是用Page1toPage2和Page2toPage1这两个函数根据用户的输入进行不断地切换。

首先先完成第一个界面的设计。首先使用 gets 读到输入，然后按空格进行分割，得到输入的字符串，剩余部分都忽略掉。不使用 scanf 的原因是有输入缓冲区的问题。

截取完成之后，对于截到的这个字符串进行比较操作，判断有没有输入 c，#或者其它字符。判断完成后，进行包的构造以及发送。发送的数据包的构造使用了结构体，结构体如下，step 为输入 01 还是 02，对应是第一个界面还是第二个界面发送的包。days 则会在第二个界面中输入 1 2 3 查询时填充的有差别，data 则是填充的数据，比如城市的名称，最后的 day 也是输入 1 2 3 查询时填充的有差别，作用是具体查询哪一天。

```
struct Packet_data{
    char step;
    char days;
    char data[30];
    char day;
};

struct Recv_data
{
    char head[2];
    char data[30];
    char weather[20];
    char tail[75];
};
```

第一个结构体是发送的包的结构，第二个结构是接收的包的结构。从 wireshark 截取的包来看，发送的包一共 33bytes，接收的包为 127bytes，具体结构看 wireshark 截到的包就行。收到的包就用第二个结构体接收。当然 head 是用来判断收到的包应该做哪种处理。例如在第一个界面对于不合法的城市输入，服务

器都会发回开头为 02 的包，这时提示没有该城市的信息。否则的话进入第二个界面。
第二个界面的逻辑与第一个界面的逻辑几乎一样，仍然是截取字符串之后，进行字符串比较，然后对包的内容进行填充。

```
if(strcmp(input,"1") == 0){
    memset(sendbuff,0,sizeof(sendbuff));
    struct Packet_data packet;
    memset(&packet, 0x00 , sizeof(struct Packet_data));
    packet.step = 0x02;
    packet.days = 0x01;
    packet.day = 0x01;
    strcpy(packet.data, name);
    memcpy(sendbuff,&packet,sizeof(struct Packet_data));
    if(send(sockfd,sendbuff,sizeof(sendbuff),0))
    {
        //printf("%s\n",packet.data);
        //printf("send2 succeed\n");
    }
    memset(recvbuff,0,sizeof(recvbuff));
    int recvfd = recv(sockfd, recvbuff, MAX_SIZE,0);
    if (recvfd == 0 || recvfd == -1){
        printf("error receive!\n");
    }
```

具体代码如上。
如果接收的包没有问题的话，使用 getweather(char* p, int kind)函数原型进行打印，kind 参数用来判断是输入 1 2 3 时进行不同的打印。打印的结果根据收到的包的 data 不同，对应天气的判定如下：
00 overcast
01 sunny
02 cloudy
03 rain
04 fog
05 rainstorm
06 thunderstorm
07 breeze
08 sandstorm
查询结束之后，如果用户输入 r，那么调用 Page2toPage1 即可。

三、隐藏的 flag

1.对于接收到的包，最后一些字节为 2e 2e 20 2e 2d 2e 2e 20 2d 2d 2d 2d 2d 20 2e 2e 2e 2d 20 2e 20 2d 2e 20 2e 2d 2d 2e 20 2d 2e 2e 00 00 00 00 00 00 00 00，翻译过后为.. .-.. --- -- ...- . -. .--. -..，这是摩斯电码，破译之后为 IL0VENPD。
2.对于收到的天气信息，如果填充码为 08 的话，天气会显示为 Please guess: ZmxhZ3tzYW5kX3N0MHJtfQ==;这是用 base64 加密过后的编码，解码之后为 flag(sand_storm)，所以第九个天气为 sandstorm。

四、实验附录以及问题

对于所需要交上的 dump 文件，我不是很清楚怎么截取 dump 文件，但是可以复制出 HEX dump，并不是很清楚是不是所要的 dump 文件。所以我这里交的是 pcapng，以及一个 txt，txt 中存放的是一个城市为 nanjing 的发送包和接收包的 HEX dump。

五、附录

时间共 6h，对于 wireshark 截取包以及破译 flag 花了 1h，剩下时间编码以及报告。