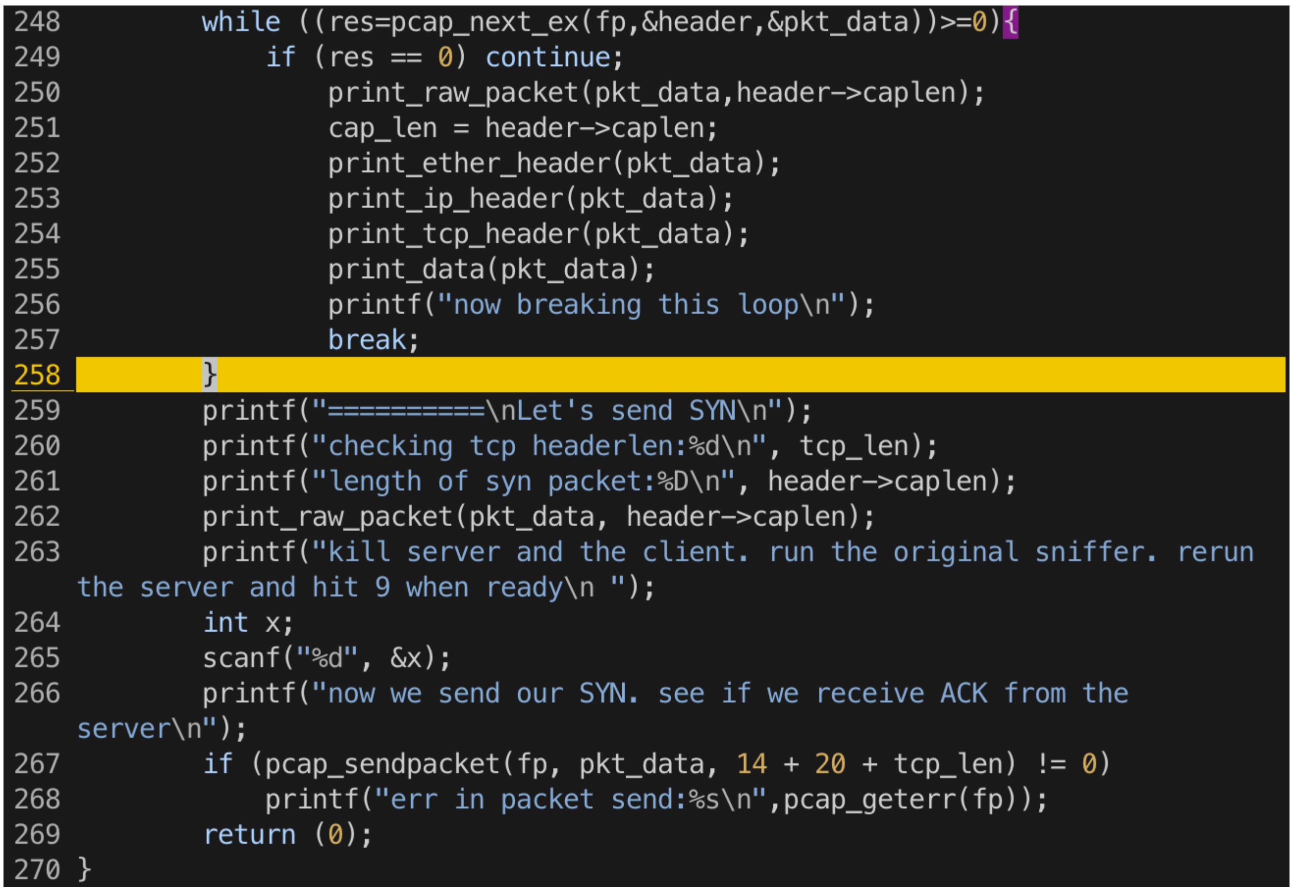
정보보호론 lect7_first

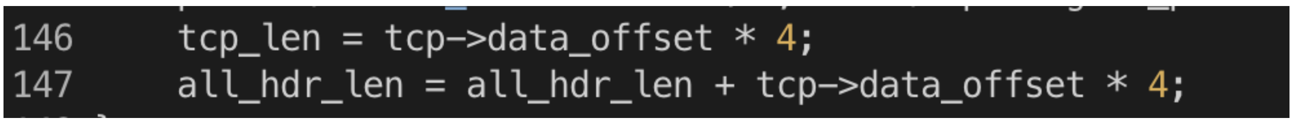
12141163 이욱진

Homework
1) Send a SYN packet to the server.
1-1)
  - Modify your sniffer such that it can send a SYN packet instead of sniffing.
    -- break out of the while loop after capturing the first SYN packet.
    -- display the packet in raw bytes.
    -- kill the server and the client (manually)
    -- run the original sniffer
    -- rerun the server
    -- send the captured SYN packet to the server
    -- check if you can see this packet in the original sniffer
    -- check if you can see the ACK packet from the server.
*If the captured SYN packet shows checksum=0, do not send as it is: you have to compute the checksum by yourself (checksum offload case).



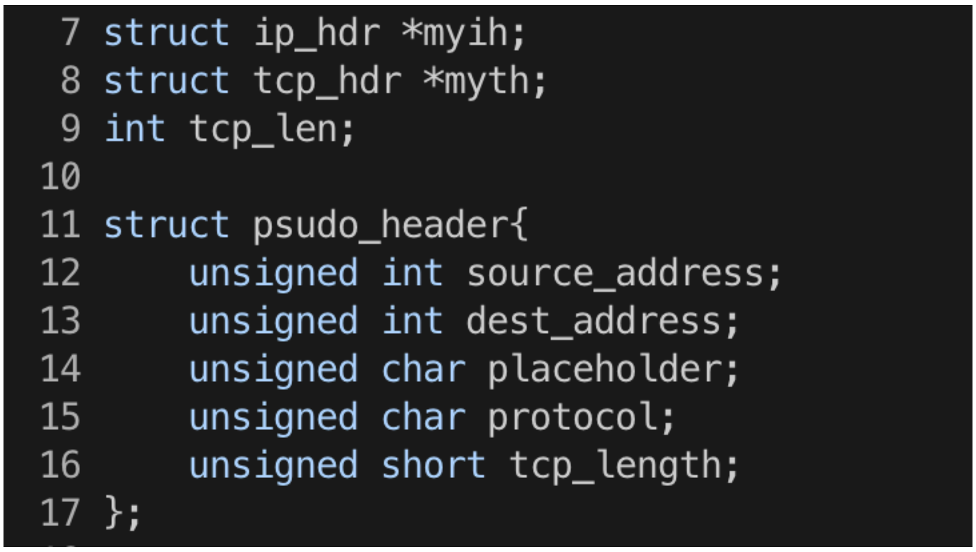
```c
        while ((res=pcap_next_ex(fp,&header,&pkt_data))>=0){
            if (res == 0) continue;
                print_raw_packet(pkt_data,header->caplen);
                cap_len = header->caplen;
                print_ether_header(pkt_data);
                print_ip_header(pkt_data);
                print_tcp_header(pkt_data);
                print_data(pkt_data);
                printf("now breaking this loop\n");
                break;
        }
        printf("==========\nLet's send SYN\n");
        printf("checking tcp headerlen:%d\n", tcp_len);
        printf("length of syn packet:%D\n", header->caplen);
        print_raw_packet(pkt_data, header->caplen);
        printf("kill server and the client. run the original sniffer. rerun the server and hit 9 when ready\n ");
        int x;
        scanf("%d", &x);
        printf("now we send our SYN. see if we receive ACK from the server\n");
        if (pcap_sendpacket(fp, pkt_data, 14 + 20 + tcp_len) != 0)
            printf("err in packet send:%s\n",pcap_geterr(fp));
        return (0);
}
```

hw 1-1을해결하기위하여 캡쳐하는방식을 이용하여 패킷을 보내려고 하였다 우선 위의 tcp_len 전역변수는 int 형이며 tcp header를 출력하는구문에서 길이를계산하여 값을할당하게하였다.



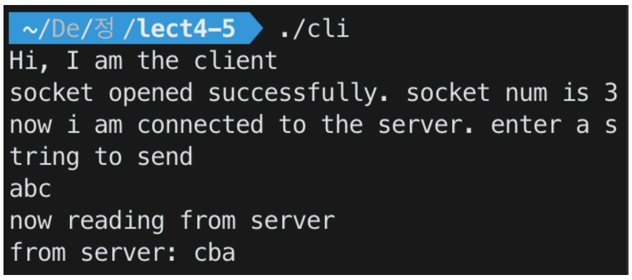
```c
    tcp_len = tcp->data_offset * 4;
    all_hdr_len = all_hdr_len + tcp->data_offset * 4;
```

그리고 수도 헤더와 더불어 myih , myth ip, tcp header를 선언해주었다.

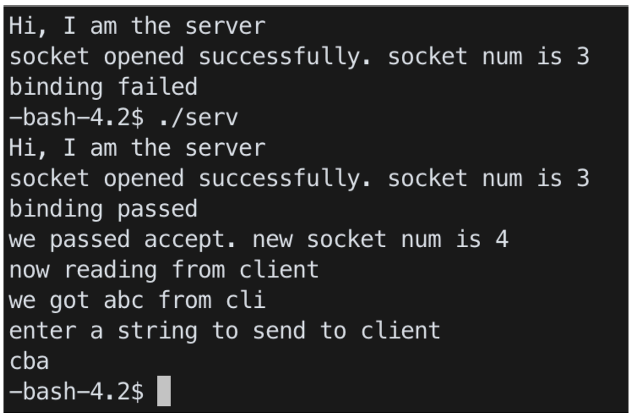


```c
struct ip_hdr *myih;
struct tcp_hdr *myth;
int tcp_len;

struct psudo_header{
    unsigned int source_address;
    unsigned int dest_address;
    unsigned char placeholder;
    unsigned char protocol;
    unsigned short tcp_length;
};
```

클라이언트와 서버가 문자를주고받게하였다.

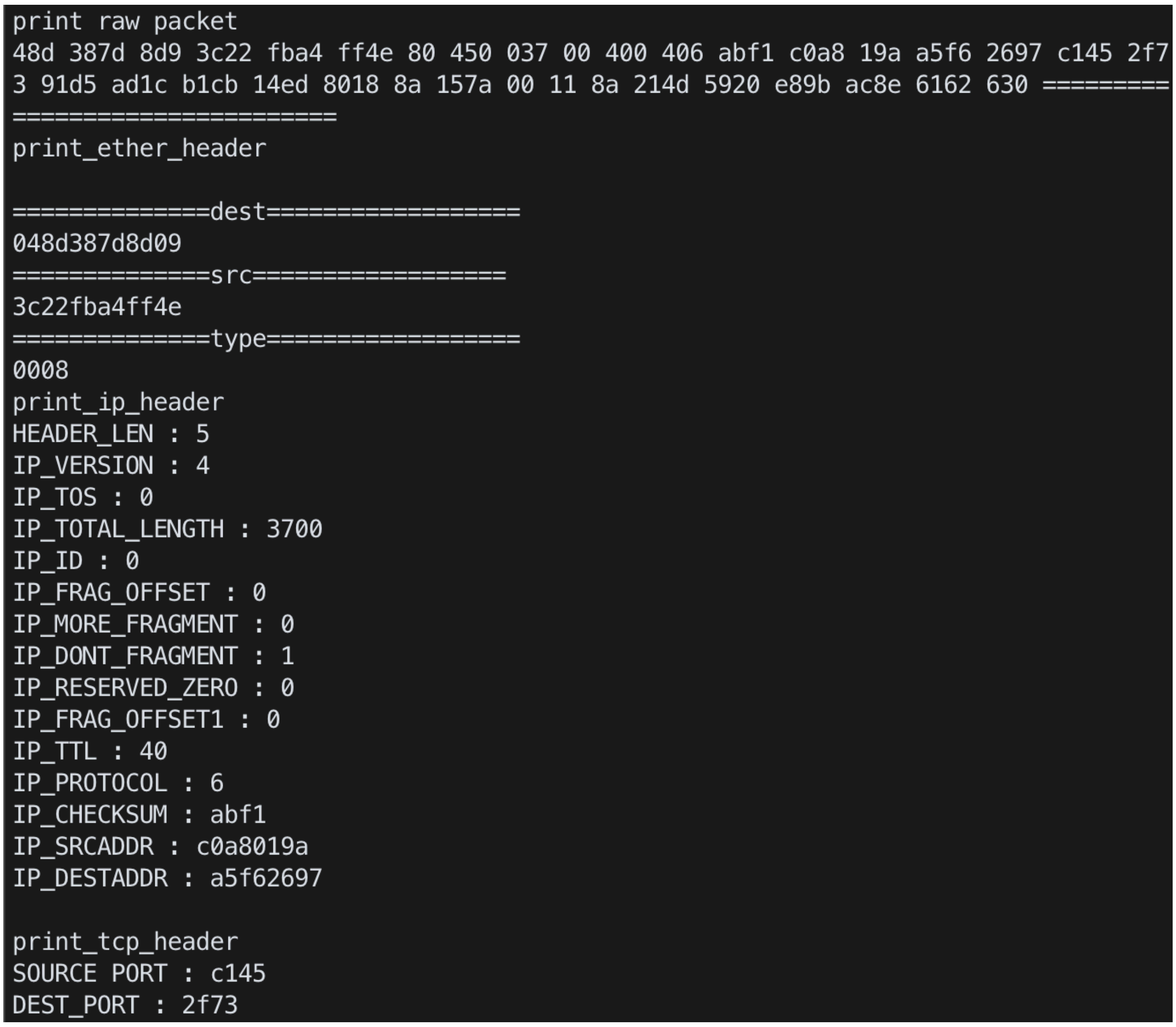


```
~/De/정 /lect4-5 ./cli
Hi, I am the client
socket opened successfully. socket num is 3
now i am connected to the server. enter a s
tring to send
abc
now reading from server
from server: cba
```

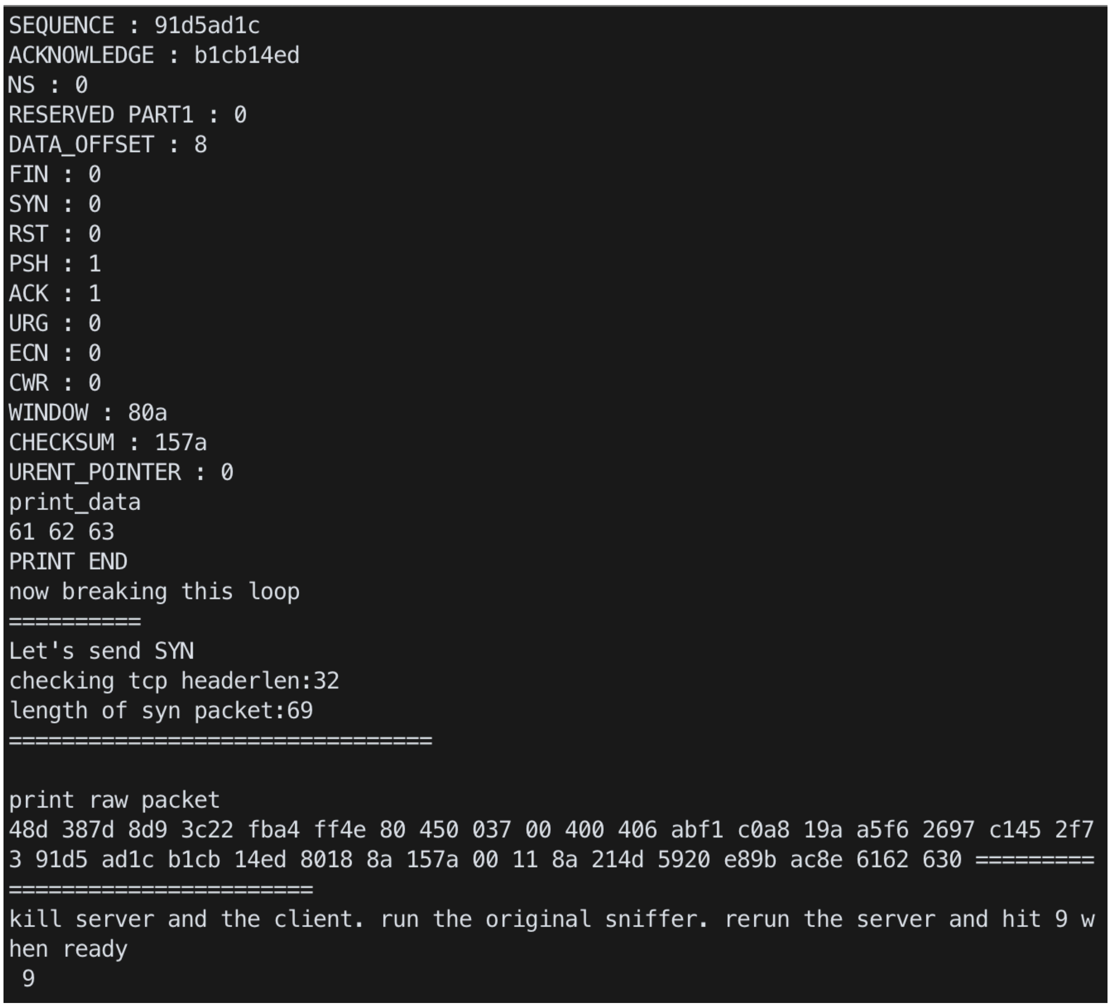


```
Hi, I am the server
socket opened successfully. socket num is 3
binding failed
-bash-4.2$ ./serv
Hi, I am the server
socket opened successfully. socket num is 3
binding passed
we passed accept. new socket num is 4
now reading from client
we got abc from cli
enter a string to send to client
cba
-bash-4.2$
```

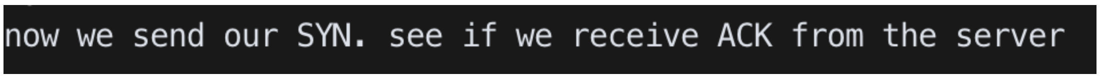
그리고 수정한 스니퍼를 실행시켜 확인해보았다.



```
print raw packet
48d 387d 8d9 3c22 fba4 ff4e 80 450 037 00 400 406 abf1 c0a8 19a a5f6 2697 c145 2f7
3 91d5 ad1c b1cb 14ed 8018 8a 157a 00 11 8a 214d 5920 e89b ac8e 6162 630 =========
=======================
print_ether_header

==============dest==================
048d387d8d09
==============src==================
3c22fba4ff4e
==============type==================
0008
print_ip_header
HEADER_LEN : 5
IP_VERSION : 4
IP_TOS : 0
IP_TOTAL_LENGTH : 3700
IP_ID : 0
IP_FRAG_OFFSET : 0
IP_MORE_FRAGMENT : 0
IP_DONT_FRAGMENT : 1
IP_RESERVED_ZERO : 0
IP_FRAG_OFFSET1 : 0
IP_TTL : 40
IP_PROTOCOL : 6
IP_CHECKSUM : abf1
IP_SRCADDR : c0a8019a
IP_DESTADDR : a5f62697

print_tcp_header
SOURCE PORT : c145
DEST_PORT : 2f73
```



```
SEQUENCE : 91d5ad1c
ACKNOWLEDGE : b1cb14ed
NS : 0
RESERVED PART1 : 0
DATA_OFFSET : 8
FIN : 0
SYN : 0
RST : 0
PSH : 1
ACK : 1
URG : 0
ECN : 0
CWR : 0
WINDOW : 80a
CHECKSUM : 157a
URENT_POINTER : 0
print_data
61 62 63
PRINT END
now breaking this loop
==========
Let's send SYN
checking tcp headerlen:32
length of syn packet:69
================================

print raw packet
48d 387d 8d9 3c22 fba4 ff4e 80 450 037 00 400 406 abf1 c0a8 19a a5f6 2697 c145 2f7
3 91d5 ad1c b1cb 14ed 8018 8a 157a 00 11 8a 214d 5920 e89b ac8e 6162 630 =========
=======================
kill server and the client. run the original sniffer. rerun the server and hit 9 w
hen ready
 9
```



```
now we send our SYN. see if we receive ACK from the server
```

1-2)

- Modify the sniffer further such that it re-computes ip and tcp checksum.
    - -- break out of the while loop after capturing the first SYN packet.
    - -- copy them into another buffer: pkt_data=>packet
    - -- set ip_check_sum and tcp_check_sum to zero
    - -- recompute ip_check_sum
    - -- recompute tcp_check_sum
    - -- display the packet in raw bytes. this should be same as pkt_data
    - -- kill the server and the client (manually)
    - -- run the original sniffer
    - -- rerun the server
    - -- send packet to the server
    - -- check if you can see this packet in the original sniffer
    - -- check if you can see the ACK packet from the server.

2) Implement a stealth scanner.

- Modify the packet sender such that it sends a SYN packet to all possible ports in the server. Detect which ports are live by finding out those who respond with SYN/ACK. You may need seperate sniffer for such detection.

(* If you have "fatal bad memory block" error, change win10pcap to wpcap. Delete win10pcap and reinstall wpcap.)