✅

[WHS][PCAP Programing] 33반 김신아(9502)

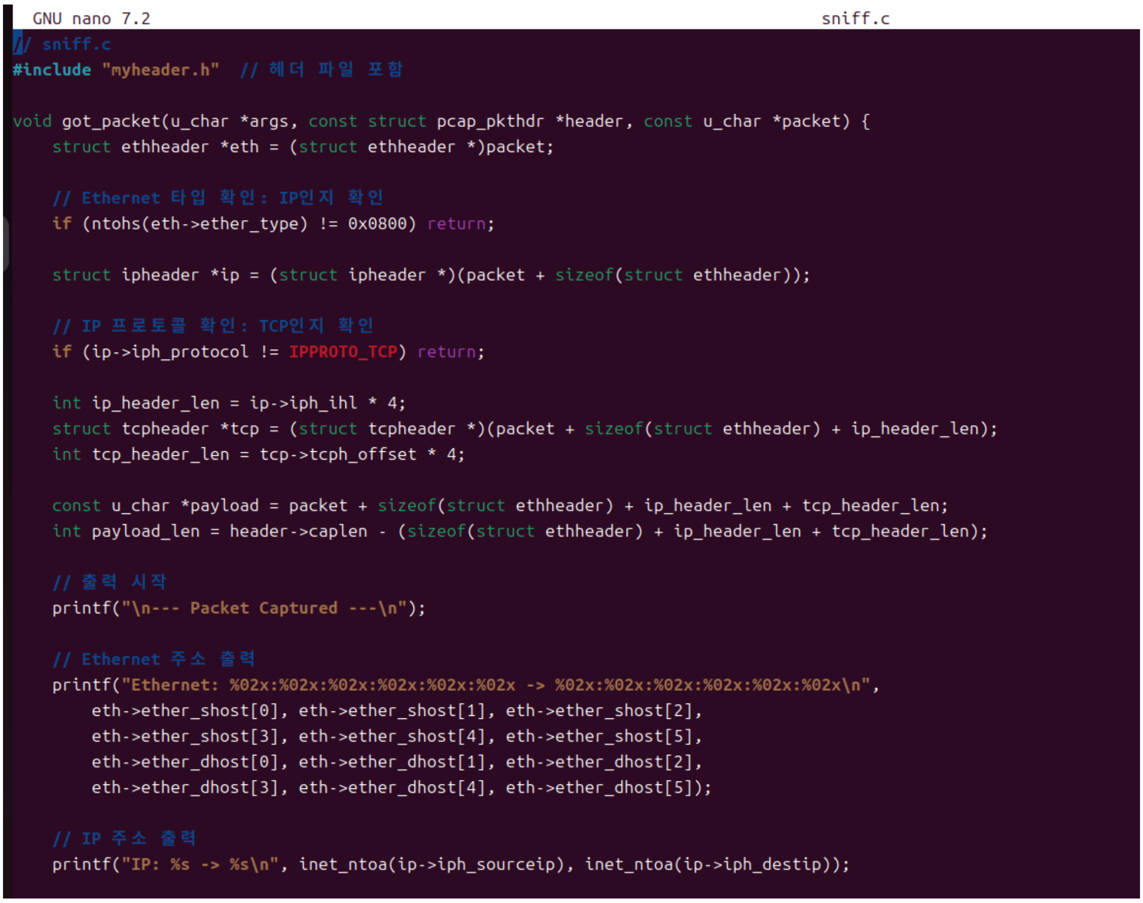
PCAP API를 활용하여 PACKET의 정보를 출력하는 프로그램으로

sniff.c 와 myheader.h 코드 두개를 작성해 주었습니다.

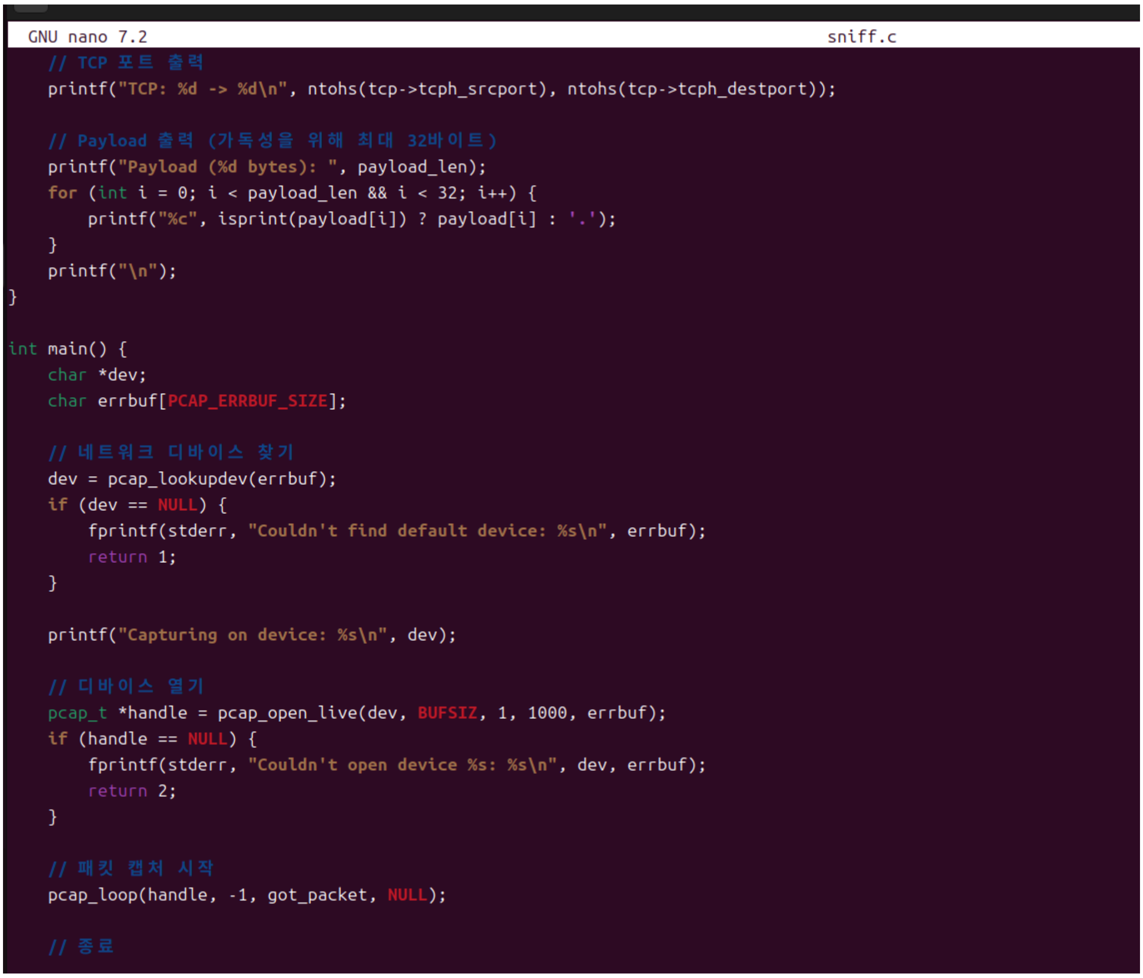
우선

sudo apt-get update
sudo apt-get install libpcap-dev

을 통해 필요한 패키지를 설치해 주었습니다.



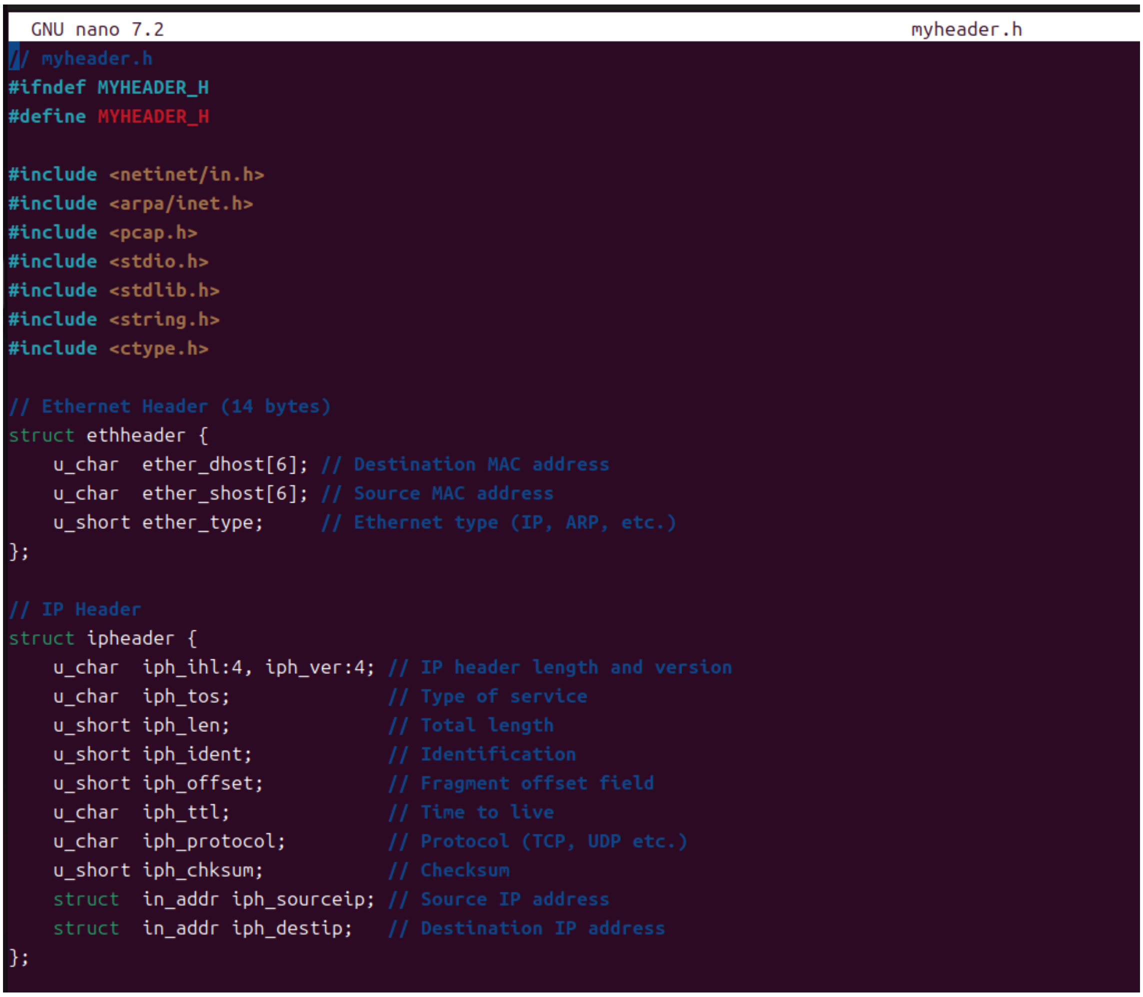
```
// sniff.c
#include "myheader.h"  // 헤더 파일 포함

void got_packet(u_char *args, const struct pcap_pkthdr *header, const u_char *packet) {
    struct ethheader *eth = (struct ethheader *)packet;

    // Ethernet 타입 확인: IP인지 확인
    if (ntohs(eth->ether_type) != 0x0800) return;

    struct ipheader *ip = (struct ipheader *)(packet + sizeof(struct ethheader));

    // IP 프로토콜 확인: TCP인지 확인
    if (ip->iph_protocol != IPPROTO_TCP) return;

    int ip_header_len = ip->iph_ihl * 4;
    struct tcpheader *tcp = (struct tcpheader *)(packet + sizeof(struct ethheader) + ip_header_len);
    int tcp_header_len = tcp->tcph_offset * 4;

    const u_char *payload = packet + sizeof(struct ethheader) + ip_header_len + tcp_header_len;
    int payload_len = header->caplen - (sizeof(struct ethheader) + ip_header_len + tcp_header_len);

    // 출력 시작
    printf("\n--- Packet Captured ---\n");

    // Ethernet 주소 출력
    printf("Ethernet: %02x:%02x:%02x:%02x:%02x:%02x -> %02x:%02x:%02x:%02x:%02x:%02x\n",
        eth->ether_shost[0], eth->ether_shost[1], eth->ether_shost[2],
        eth->ether_shost[3], eth->ether_shost[4], eth->ether_shost[5],
        eth->ether_dhost[0], eth->ether_dhost[1], eth->ether_dhost[2],
        eth->ether_dhost[3], eth->ether_dhost[4], eth->ether_dhost[5]);

    // IP 주소 출력
    printf("IP: %s -> %s\n", inet_ntoa(ip->iph_sourceip), inet_ntoa(ip->iph_destip));
```



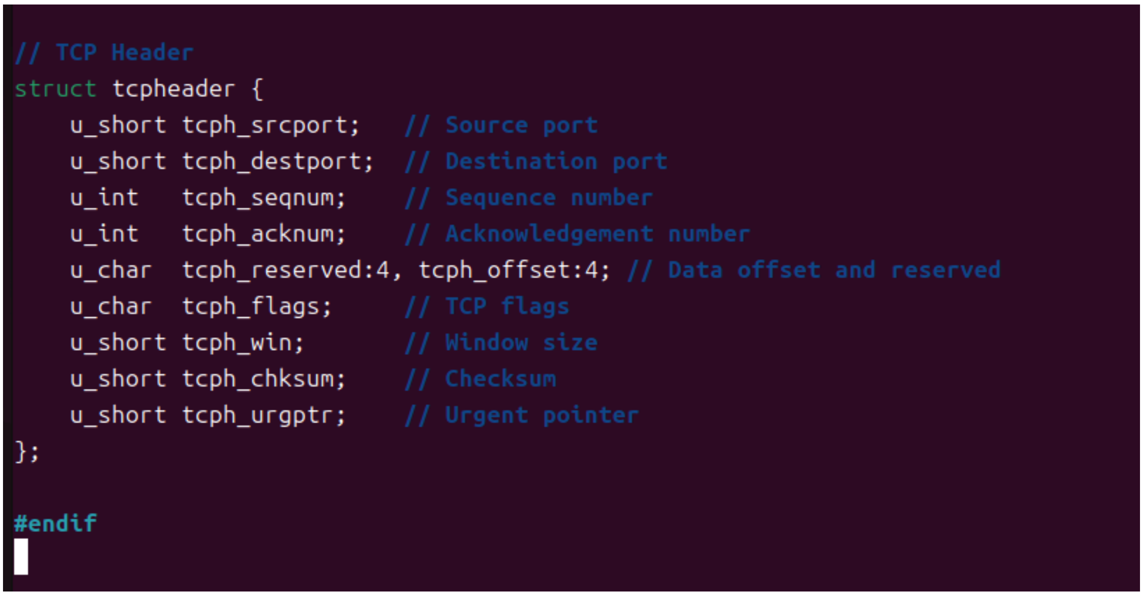
```c
    // TCP 포트 출력
    printf("TCP: %d -> %d\n", ntohs(tcp->tcph_srcport), ntohs(tcp->tcph_destport));

    // Payload 출력 (가독성을 위해 최대 32바이트)
    printf("Payload (%d bytes): ", payload_len);
    for (int i = 0; i < payload_len && i < 32; i++) {
        printf("%c", isprint(payload[i]) ? payload[i] : '.');
    }
    printf("\n");
}

int main() {
    char *dev;
    char errbuf[PCAP_ERRBUF_SIZE];

    // 네트워크 디바이스 찾기
    dev = pcap_lookupdev(errbuf);
    if (dev == NULL) {
        fprintf(stderr, "Couldn't find default device: %s\n", errbuf);
        return 1;
    }

    printf("Capturing on device: %s\n", dev);

    // 디바이스 열기
    pcap_t *handle = pcap_open_live(dev, BUFSIZ, 1, 1000, errbuf);
    if (handle == NULL) {
        fprintf(stderr, "Couldn't open device %s: %s\n", dev, errbuf);
        return 2;
    }

    // 패킷 캡처 시작
    pcap_loop(handle, -1, got_packet, NULL);

    // 종료
```

sniff.c코드 작성

sniff.c



```
  GNU nano 7.2                                                           myheader.h
// myheader.h
#ifndef MYHEADER_H
#define MYHEADER_H

#include <netinet/in.h>
#include <arpa/inet.h>
#include <pcap.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

// Ethernet Header (14 bytes)
struct ethheader {
    u_char  ether_dhost[6]; // Destination MAC address
    u_char  ether_shost[6]; // Source MAC address
    u_short ether_type;     // Ethernet type (IP, ARP, etc.)
};

// IP Header
struct ipheader {
    u_char  iph_ihl:4, iph_ver:4; // IP header length and version
    u_char  iph_tos;              // Type of service
    u_short iph_len;              // Total length
    u_short iph_ident;            // Identification
    u_short iph_offset;           // Fragment offset field
    u_char  iph_ttl;              // Time to live
    u_char  iph_protocol;         // Protocol (TCP, UDP etc.)
    u_short iph_chksum;           // Checksum
    struct  in_addr iph_sourceip; // Source IP address
    struct  in_addr iph_destip;   // Destination IP address
};
```

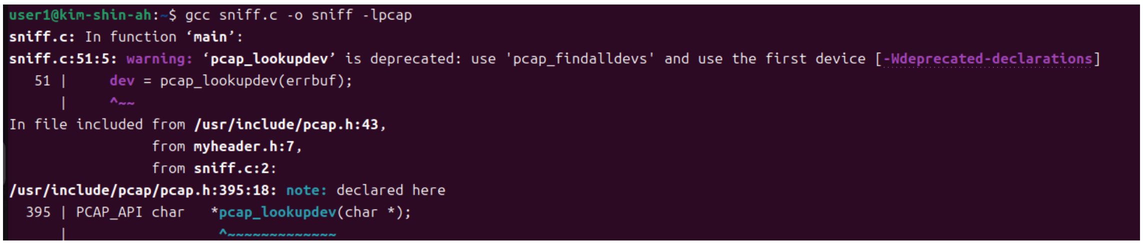


```
// TCP Header
struct tcpheader {
    u_short tcph_srcport;   // Source port
    u_short tcph_destport;  // Destination port
    u_int   tcph_seqnum;    // Sequence number
    u_int   tcph_acknum;    // Acknowledgement number
    u_char  tcph_reserved:4, tcph_offset:4; // Data offset and reserved
    u_char  tcph_flags;     // TCP flags
    u_short tcph_win;       // Window size
    u_short tcph_chksum;    // Checksum
    u_short tcph_urgptr;    // Urgent pointer
};

#endif
```

myheader.h 코드 작성

myheader.h

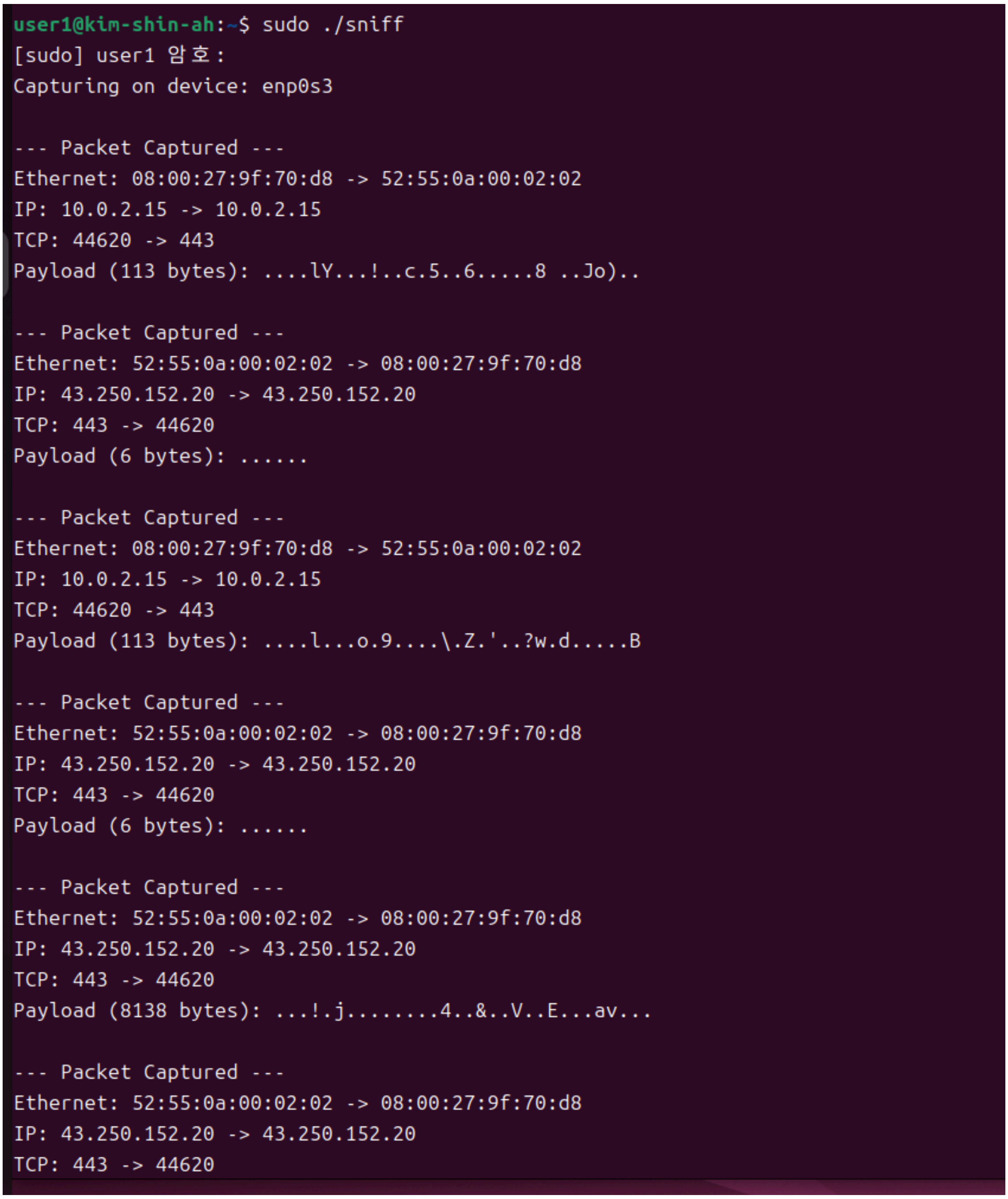
sniff.c 와 myheader.h 를 작성할 때 두 파일을 같은 디렉토리 안에 있도록 작성해주었습니다.



```
user1@kim-shin-ah:~$ gcc sniff.c -o sniff -lpcap
sniff.c: In function ‘main’:
sniff.c:51:5: warning: ‘pcap_lookupdev’ is deprecated: use 'pcap_findalldevs' and use the first device [-Wdeprecated-declarations]
   51 |     dev = pcap_lookupdev(errbuf);
      |     ^~~
In file included from /usr/include/pcap.h:43,
                 from myheader.h:7,
                 from sniff.c:2:
/usr/include/pcap/pcap.h:395:18: note: declared here
  395 | PCAP_API char   *pcap_lookupdev(char *);
      |                  ^~~~~~~~~~~~~~
```

gcc sniff.c -o sniff -lpcap

을 통해 컴파일을 해주었습니다.



```
user1@kim-shin-ah:~$ sudo ./sniff
[sudo] user1 암호:
Capturing on device: enp0s3

--- Packet Captured ---
Ethernet: 08:00:27:9f:70:d8 -> 52:55:0a:00:02:02
IP: 10.0.2.15 -> 10.0.2.15
TCP: 44620 -> 443
Payload (113 bytes): ....lY...!..c.5..6.....8 ..Jo)..

--- Packet Captured ---
Ethernet: 52:55:0a:00:02:02 -> 08:00:27:9f:70:d8
IP: 43.250.152.20 -> 43.250.152.20
TCP: 443 -> 44620
Payload (6 bytes): ......

--- Packet Captured ---
Ethernet: 08:00:27:9f:70:d8 -> 52:55:0a:00:02:02
IP: 10.0.2.15 -> 10.0.2.15
TCP: 44620 -> 443
Payload (113 bytes): ....l...o.9....\.Z.'..?w.d.....B

--- Packet Captured ---
Ethernet: 52:55:0a:00:02:02 -> 08:00:27:9f:70:d8
IP: 43.250.152.20 -> 43.250.152.20
TCP: 443 -> 44620
Payload (6 bytes): ......

--- Packet Captured ---
Ethernet: 52:55:0a:00:02:02 -> 08:00:27:9f:70:d8
IP: 43.250.152.20 -> 43.250.152.20
TCP: 443 -> 44620
Payload (8138 bytes): ...!.j........4..&..V..E...av...

--- Packet Captured ---
Ethernet: 52:55:0a:00:02:02 -> 08:00:27:9f:70:d8
IP: 43.250.152.20 -> 43.250.152.20
TCP: 443 -> 44620
```

sudo ./sniff

최종적으로 실행을 해주어 packet이 출력되는 것을 확인해주었습니다.

Ethernet, IP, TCP 출력