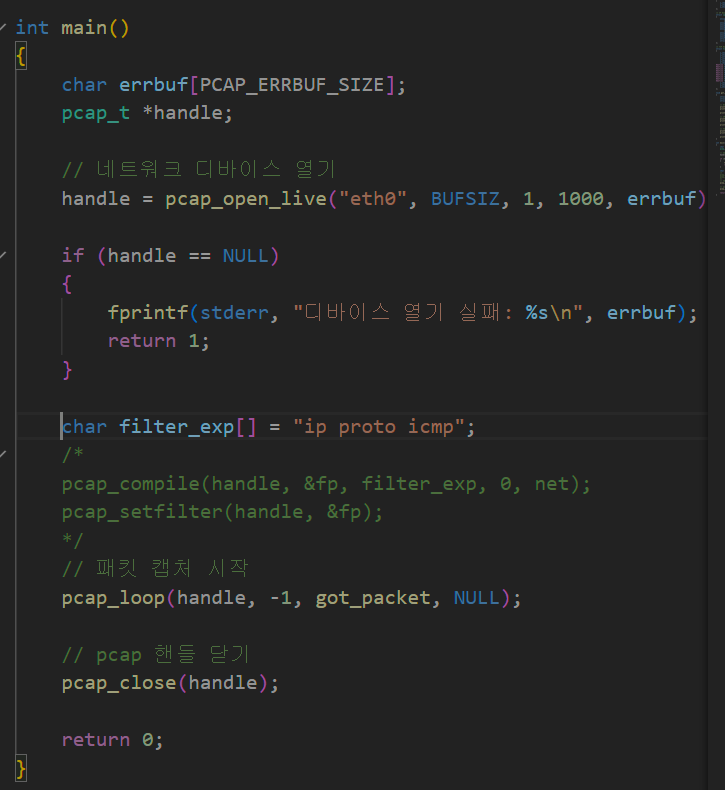
**[WHS][PCAP Programming] 18반 김소정(5649)**

1. 헤더 처리

myheader.h에서 UDP 대상 구조체를 제외하고 사용하였다.

ipheader, etherheader, tcpheader를 사용하였다.

1. main함수



우선, 네트워크 디바이스를 열어야한다. 이를 위해서 pcap\_open\_live 함수를 사용한다.

* pcap\_t \*pcap\_open\_live(char \*device, int snaplen, int promisc, int to\_ms, char \*ebuf)

|  |  |
| --- | --- |
| **Device** | 패킷을 오픈할 인터페이스 |
| **snaplen** | 받아들일수 있는 패킷의 최대 크기(byte) |
| **Promisc** | 네트워크 디바이스를 “promiscuous mode”로 할것인지를 결정하기 위해서 사용한다.  1일 때는 로컬 네트웍의 모든 패킷을 캡쳐하게 된다. 0일 때는 에는 자기에게만 향하는 패킷을 캡쳐한다. |
| **To\_ms** | Time out 지정 |
| **ebuf** | 에러 메시지 발생 시 저장 |
| **Return값** | 성공 – 패킷 캡쳐 descriptor 반환  실패 – NULL |

그 후 에러가 존재하면 errbuf에 저장된 에러 내용을 출력하고, 아니면 패킷 캡처를 시작한다.

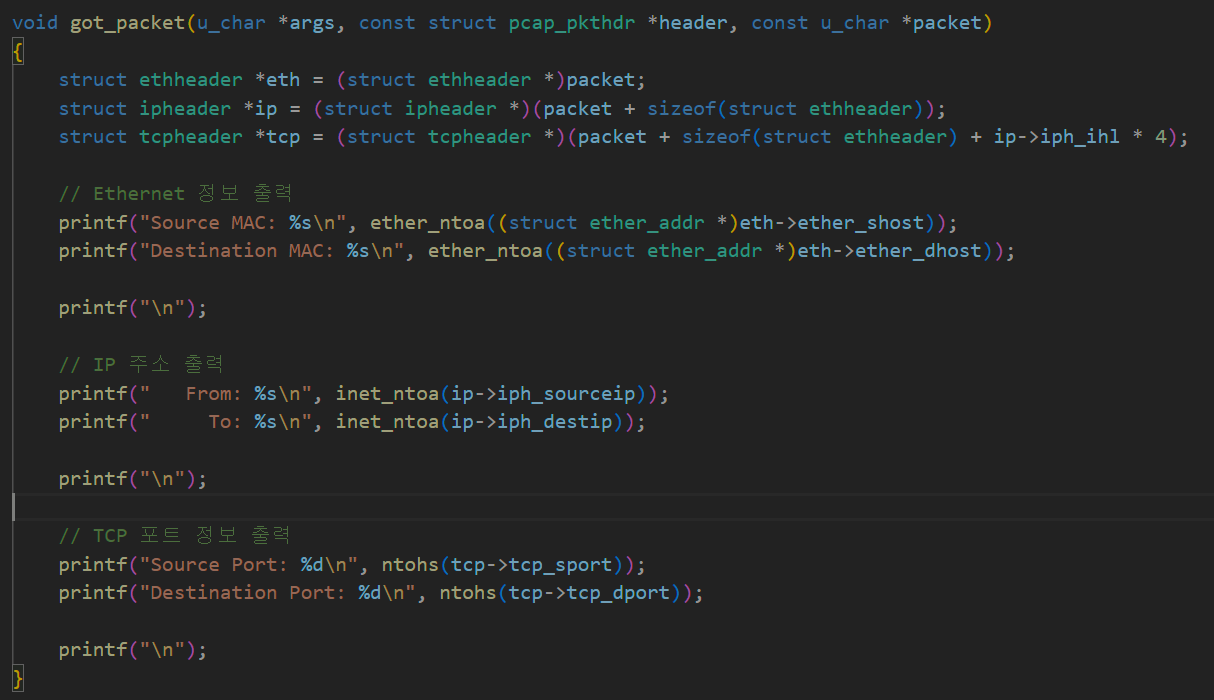
* int pcap\_loop(pcap\_t \*p, int cnt, pcap\_handler callback, u\_char \*user);

|  |  |
| --- | --- |
| **p** | 패킷을 캡처할 pcap 세션을 가리키는 포인터. pcap\_open\_live로 열린 pcap을 넣어준다. |
| **cnt** | 읽을 패킷의 최대 개수. -1일 때는 무한루프 |
| **callback** | 각 패킷에 대해 호출될 콜백 함수의 포인터 |
| **user** | 사용자 정의 데이터를 포함하는 포인터 |
| **int** | 성공: 처리된 패킷 개수  실패: 음수 값 |

패킷 캡처를 마치면 pcap\_close를 이용하여 패킷을 종료한다.

- void pcap\_close (pcap\_t \* p): p와 연관된 파일을 닫고 자원을 할당 해제

1. void got\_packet(u\_char \*args, const struct pcap\_pkthdr \*header, const u\_char \*packet)



구조체 변수 세 개를 정의해주었다.

1. eth: 패킷 버퍼에서 추출한 이더넷 헤더 포인터
2. ip: ip 헤더 포인터이다. 이때 이더넷 헤더를 건너뛴 다음이 ip 헤더의 시작 주소이므로 packet + sizeof(struct ethheader)를 해주었다.
3. tcp: tcp 헤더 포인터이다. tcp는 이더넷 헤더와 ip 헤더를 모두 건너뛰어야 한다. 그러므로 ip 주소에서 ip->iph\_ihl \* 4를 해주었다. 이때 ip->iph\_ihl은은 ip 헤더의 길이인데, 실제 4bytes씩 할당되므로 4를 곱해줘서 더해야 한다.

그 후로 순서대로 Ethernet 정보, IP 주소, TCP 포트 정보를 출력하였다.

1. eth 구조체에 있는 eth->ether\_shost, eth->ether\_dhost를 각각 사용하였다. 이때 ether\_ntoa 함수는 이더넷 주소를 "XX:XX:XX:XX:XX:XX"와 같은 형식의 문자열로 변환하여 반환한다.
2. IP 주소를 반환할 때는 ip 구조체에 있는 iph\_sourceip, iph\_destip를 사용하였다. inet\_ntoa 함수는 IPv4 주소를 받아서 해당 주소를 문자열로 변환한다.
3. tcp 포트 정보를 반환할 때는 tcp 구조체에 있는 tcp\_sport, tcp\_dport를 사용하였다. ntohs 함수는 networt to host short의 약자로, "네트워크 바이트 순서"에서 "호스트 바이트 순서로 16비트 정수를 변환하는 함수이다.