

# [ R E P O R T ]

정보통신공학전공

200301582

김성태



**국립공주대학교**

KONGJU NATIONAL UNIVERSITY

# 1. 과제의 목표

\* 리눅스 상에서 패킷을 잡는 기법에 대하여 공부한다. 구체적으로 pcap library 와 관련 테크닉을 공부하고 예제 프로그램을 해석한다. 참고로 본 예제 프로그램은 앞으로 개발한 SimTL 의 기초가 되므로 완벽한 이해가 필수적이다.

# 2. 과제 수행 내용 및 수행 결과

## \* tcpdump

TCPDUMP 란 주어진 조건식을 만족하는 네트워크 인터페이스를 거치는 패킷들의 헤더들을 출력해 주는 프로그램이다.

## \* tcpdump 의 옵션들.

- a : Network & Broadcast 주소들을 이름들로 바꾼다.
- c Number : 제시된 수의 패킷을 받은 후 종료한다.
- d : comile된 packet-matching code를 사람이 읽을 수 있도록 바꾸어 표준 출력으로 출력하고, 종료한다.
- dd : packet-matching code를 C program의 일부로 출력한다.
- ddd : packet-matching code를 숫자로 출력한다.
- e : 출력되는 각각의 행에 대해서 link-level 헤더를 출력한다.
- f : 외부의 internet address를 가급적 심볼로 출력한다(Sun의 yp server와의 사용은 가급적 피하자).
- F file : filter 표현의 입력으로 파일을 받아들인다. 커맨드라인에 주어진 추가의 표현들은 모두 무시된다.
- i device : 어느 인터페이스를 경유하는 패킷들을 잡을지 지정한다. 지저되지 않으면 시스템의 인터페이스 리스트를 뒤져서 가장 낮은 번호를 가진 인터페이스를 선택한다(이 때 loopback은 제외된다).
- l : 표준 출력으로 나가는 데이터들을 line buffering한다. 다른 프로그램에서 tcpdump 로부터 데이터를 받고자 할 때, 유용하다.
- n : 모든 주소들을 번역하지 않는다(port,host address 등등)
- N : 호스트 이름을 출력할 때, 도메인을 찍지 않는다.
- O : packet-matching code optimizer를 실행하지 않는다. 이 옵션은 optimizer에 있는 버그를 찾을 때나 쓰인다.
- p : 인터페이스를 promiscuous mode로 두지 않는다.
- q : 프로토콜에 대한 정보를 덜 출력한다. 따라서 출력되는 라인이 좀 더 짧아진다.
- r file : 패킷들을 '-w'옵션으로 만들어진 파일로 부터 읽어 들인다. 파일에 "-" 가 사용되면 표준 입력을 통해서 받아들인다.
- s length: 패킷들로부터 추출하는 샘플을 default값인 68Byte외의 값으로 설정할 때 사용한다(SunOS의 NIT에서는 최소가 96Byte이다). 68Byte는 IP,ICMP, TCP, UDP등에 적절한 값이지만 Name Server나 NFS 패킷들의 경우에는 프로토콜의 정보들을 Truncation할 우려가 있다. 이 옵션을 수정할 때는 신중해야만 한다. 이유는 샘플 사이즈를 크게 잡으면 곧 패킷 하나 하나를 처리하는데 시간이 더 걸릴 뿐만아니라 패킷 버퍼의 사이즈도 자연

히 작아지게 되어 손실되는 패킷들이 발생할 수 있기 때문이다. 또, 작게 잡으면 그만큼의 정보를 잃게되는 것이다. 따라서 가급적 캡취하고자하는 프로토콜의 헤더 사이즈에 가깝게 잡아주어야 한다.

- T type** : 조건식에 의해 선택된 패킷들을 명시된 형식으로 표시한다. type에는 다음과 같은 것들이 올 수 있다. rpc(Remote Procedure Call), rtp(Real-Time Applications protocol), rtcp(Real-Time Application control protocol), vat(Visual Audio Tool), wb(distributed White Board)
- S** : TCP sequence번호를 상대적인 번호가 아닌 절대적인 번호로 출력한다.
- t** : 출력되는 각각의 라인에 시간을 출력하지 않는다.
- tt** : 출력되는 각각의 라인에 형식이 없는 시간들을 출력한다.
- v** : 좀 더 많은 정보들을 출력한다.
- vv** : '-v'보다 좀 더 많은 정보들을 출력한다.
- w** : 캡취한 패킷들을 분석해서 출력하는 대신에 그대로 파일에 저장한다.
- x** : 각각의 패킷을 헥사코드로 출력한다.

## \* tcpdump 의 조건식(expression)

옵션의 제일 마지막인 조건식은 어떤 패킷들을 출력할지를 선택하는데 쓰인다. 조건식이 주어지지 않는다면 모든 패킷들이 그 대상이 될 것이다. 일단 주어지면, 아무리 패킷들이 많아도 조건식에 부합하는 패킷만을 출력한다.

조건식들은 하나 또는 몇 개의 primitive들로 구성되어 있다. primitive들은 보통 하나 혹은 몇개의 qualifier들 다음에 오는 하나의 값으로 이루어진다. Qualifier들은 모두 3종류이며 다음과 같다.

- type** : 주어진 값의 종류가 무엇인지를 나타낸다. 가능한 type들은 'host', 'net', 'port'가 있다. type이 없는 값들은 type을 host라 가정한다.
- dir** : id로부터의 어떤 특정한 전송 방향을 나타낸다. 가능한 방향은 'src', 'dst', 'src or dst', 'src and dst'이다. 만약 방향이 정해지지 않았다면, src or dst such as slip) the in bound and out bound qualifiers can be used to specify a desired direction."
- proto** : 매칭을 특정 프로토콜에 한해서 수행한다. 가능한 프로토콜들은 ether, fddi, ip, arp, rarp, decnet, lat, sca, moprc, mopdl, tcp, udp이다. 만약 프로토콜이 명시되지 않았다면, 해당하는 값의 type에 관련된 모든 프로토콜들이 그 대상이 된다.

이 밖에도 위의 패턴을 따르지 않는 Primitive들이 존재한다(gateway, broadcast, less, greater, 산술식). 좀 더 정교한 조건식들을 사용하려면, 'and(&&)', 'or(||)', 'not(!)'들을 사용하여 여러 primitive들을 연결하면 된다. 같은 표현들은 생략될 수 있다.

### 사용 가능한 Primitive들

#### **dst host HOST**

: packet의 IP destination 항목이 HOST일때 참이 된다.

#### **src host HOST**

: packet의 IP source 항목이 HOST일때 참이 된다.

#### **host HOST**

: IP source, IP destination 항목 중 어느 하나라도 HOST이면 참이다.

**ether dst ehost**

: ethernet destination 주소가 ehost일 때 참이다.

**ether src ehost**

: ethernet source 주소가 ehost일 때 참이다.

**ether host ehost**

: ethernet source, destination 항목들 중 어느 하나라도 ehost이면 참이다.

**gateway host**

: 패킷이 host를 게이트웨이로 사용하면 참이다. 이 말의 의미는 ethernet source나 destination 항목은 host이지만, IP source와 destination은 host가 아닐 때를 말한다.

**dst net NET**

: 패킷의 IP destination 주소가 NET의 network number를 가지고 있을 때 참이다.

**src net NET**

: 패킷의 IP source 주소가 NET의 network number를 가지고 있을 때 참이다.

**net NET**

: 패킷의 IP source 주소 혹은 destination 주소가 NET의 network number를 가지고 있을 때 참이다.

**net netmask mask**

: IP 어드레스가 지정된 netmask를 통해서 net과 매칭되면 참이다.

**net net/len**

: IP 어드레스가 netmask와 len 비트만큼 매치되면 참이다.

**dst port PORT**

: 패킷이 ip/tcp, ip/udp 프로토콜의 패킷이고 destination port의 값이 PORT일 때 참이다. port는 /etc/services에 명시된 이름일 수도 있고 그냥 숫자일 수도 있다. 만약 이름이 사용됐다면 port 번호와 프로토콜이 같이 체크될 것이다. 만약 숫자나 불 확실한 이름이 사용됐을 경우에는 port 번호만이 체크될 것이다.

**src port PORT**

: 패킷의 source port의 값으로 PORT를 가지면 참이다.

**port PORT**

: 패킷의 source, destination port 중에 하나라도 PORT이면 참이다.

**less length**

: 패킷이 length보다 짧거나 같으면 참이다.(len <= length)

**greater length**

: 패킷이 length보다 짧거나 같으면 참이다.(len >= length)

**ip proto protocol**

: 패킷이 지정된 종류의 프로토콜의 ip패킷이면 참이다. Protocol은 icmp, igrp, udp, nd, tcp 중의 하나 혹은 몇 개가 될 수 있다. 주의할 점은 tcp, udp, icmp들은 '\'로 escape되어야 한다.

**ether broadcast**

: 패킷이 ethernet broadcast 패킷이라면 참이다. ether는 생략 가능하다.

**ip broadcast**

: 패킷이 IP broadcast 패킷이라면 참이다.

**ether multicast**

: 패킷이 IP multicast 패킷이라면 참이다.

**ether proto protocol**

: 패킷이 ether type의 protocol이라면 참이다. protocol은 ip, arp, rarp 중에 하나 혹은 몇개가 될 수 있다. ip proto protocol에서와 마찬가지로 ip, arp, rarp는 escape 되어야 한다.

**decnet src host**

: 만약 DECNET의 source address가 host이면 참이다. 이 어드레스는 '10.123'이 나 DECNET의 host name일 수 있다. DECNET host name은 DECNET에서 돌아가도록 설정된 Ultrix 시스템에서만 사용 가능하다.

**decnet dst host**

: DECNET destination address가 host이면 참이다.

**decnet host HOST**

: DECNET source, destination address중의 하나라도 HOST이면 참이다.

**ip, arp, rarp, decnet**

: ether proto [ip|arp|rarp|decnet]의 약어

**lat, moprc, mopdl**

: ether proto [lat|moprc|mopdl]의 약어

**tcp, udp, icmp**

: ip proto [tcp|udp|icmp]의 약어

**expr relop expr**

: proto [expr:size]의 형식을 띤다. proto, expr, size에 올 수 있는 것들은 다음과 같다.

**proto** : ether, fddi, ip, arp, rarp, tcp, udp, icmp

**expr** : indicate Byte offset of packet of proto

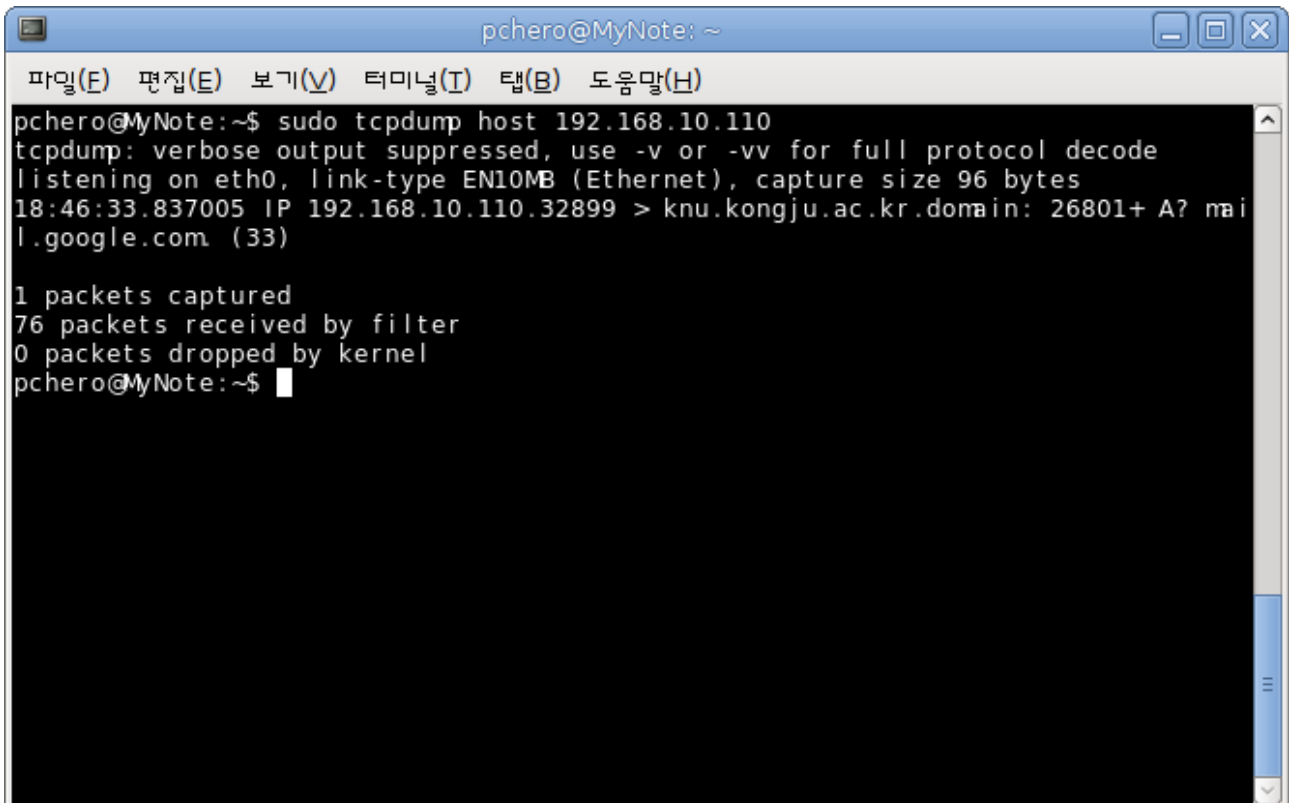
**size** : optional. indicate the size of bytes in field of interest  
default is one, and can be two or four

: RELOP

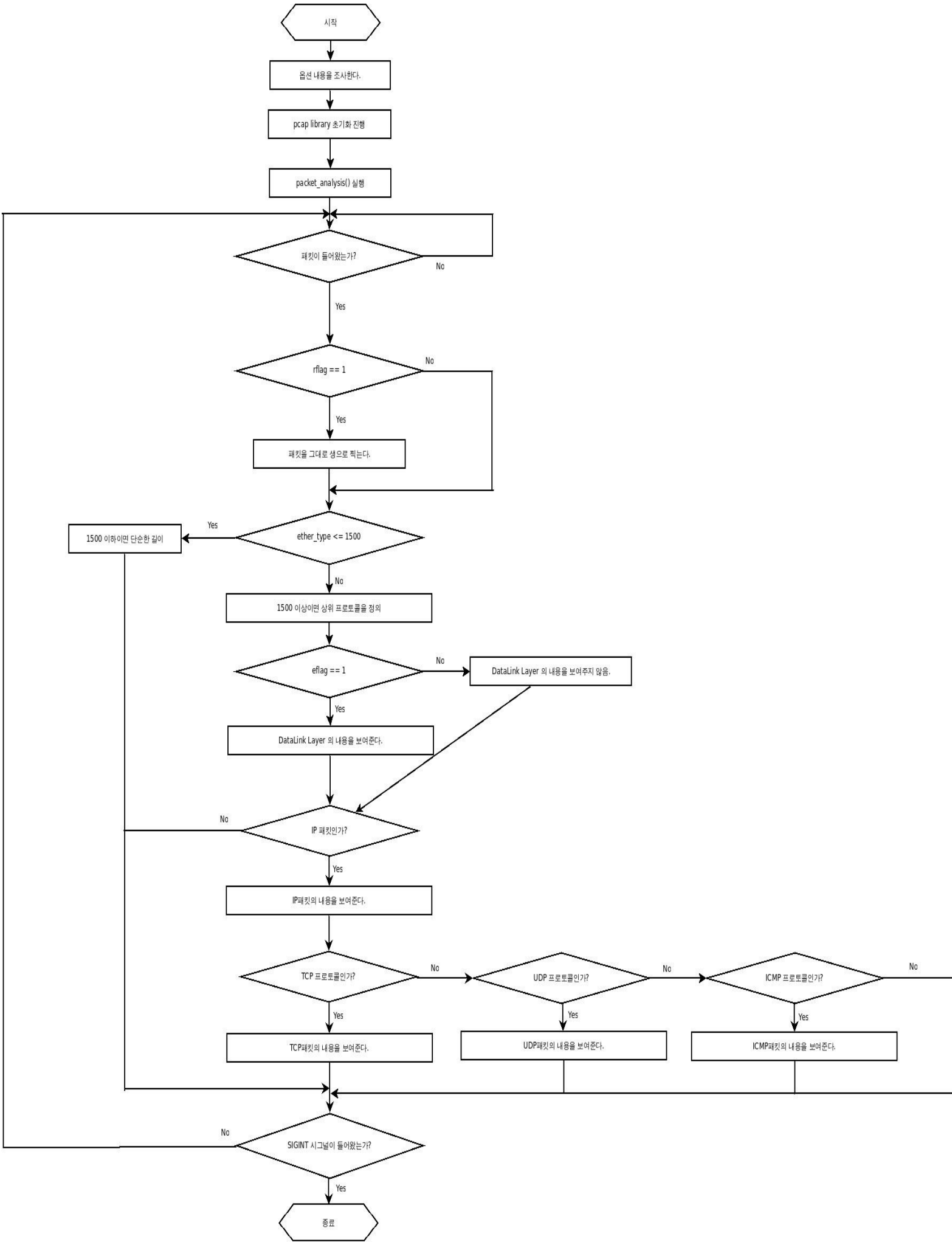
!=, =, <=, >=, etc.

이 조건식을 사용하기 위해서는 먼저 해당하는 Protocol(proto)의 헤더에 관련된 것들을 자세히 알아야만 한다. proto에는 대상이 될 프로토콜을 지정한다. expr에는 프로토콜 헤더의 처음부터의 Byte Offset을 지정하는 식이 들어가게 된다. Size는 Option이며 지정이 안 되어 있을 경우에는 자동으로 1byte를 지칭한다. 따라서 이 조건식을 사용하게 되면 헤더에 포함된 정보를 Bitmask를 사용하여 직접 원하는 패킷인지를 가려낼 수 있기 때문에, 보다 정밀한 사용이 가능하게 된다.

**\* tcpdump 사용 모습**



# \* pcap library 를 이용한 패킷 분석기 Flowchart



# \* pcap 프로그램 사용모습

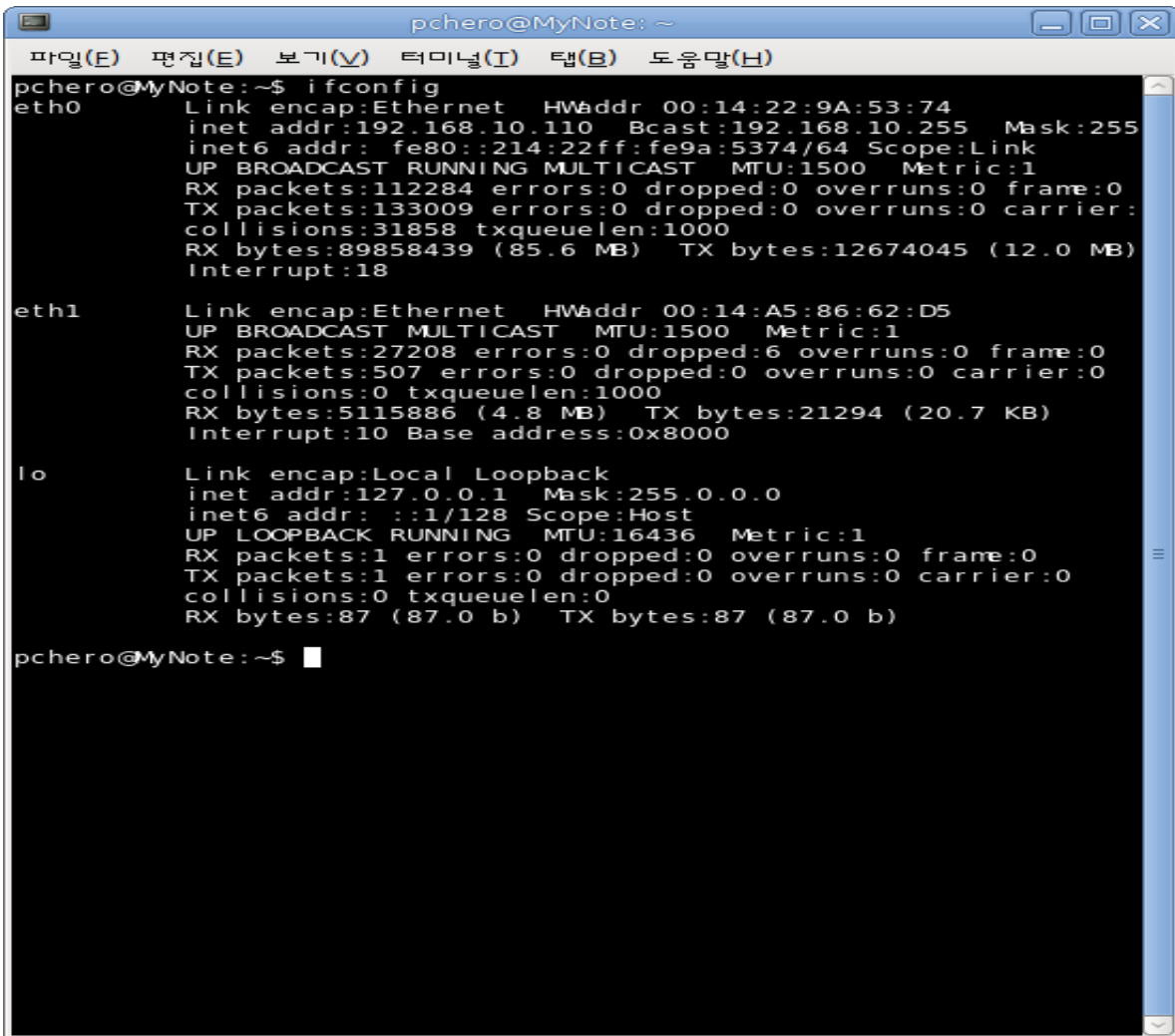
```
pchero@MyNote: ~/Desktop/Study/REPORT/profe.kimt/Internet Engineering/02/test_pcap
파일(F) 편집(E) 보기(V) 터미널(T) 탭(M) 도움말(H)

===== TCP DATA (HEXA) =====
47 45 54 20 2f 6a 73 2f 74 77 2d 73 61 63 6b 2e
6a 73 20 48 54 54 50 2f 31 2e 31 0d 0a 48 6f 73
74 3a 20 6f 2d 67 61 6d 65 2e 63 6f 2e 6b 72 0d
0a 55 73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a
69 6c 6c 61 2f 35 2e 30 20 28 58 31 31 3b 20 55
3b 20 4c 69 6e 75 78 20 69 36 38 36 3b 20 6b 6f
2d 4b 52 3b 20 72 76 3a 31 2e 38 2e 31 2e 31 33
29 20 47 65 63 6b 6f 2f 32 30 30 38 30 33 32 35
20 55 62 75 6e 74 75 2f 37 2e 31 30 20 28 67 75
74 73 79 29 20 46 69 72 65 66 6f 78 2f 32 2e 30
2e 30 2e 31 33 0d 0a 41 63 63 65 70 74 3a 20 2a
2f 2a 0d 0a 41 63 63 65 70 74 2d 4c 61 6e 67 75
61 67 65 3a 20 6b 6f 2d 6b 72 2c 6b 6f 3b 71 3d
30 2e 38 2c 65 6e 2d 75 73 3b 71 3d 30 2e 35 2c
65 6e 3b 71 3d 30 2e 33 0d 0a 41 63 63 65 70 74
2d 45 6e 63 6f 64 69 6e 67 3a 20 67 7a 69 70 2c
64 65 66 6c 61 74 65 0d 0a 41 63 63 65 70 74 2d
43 68 61 72 73 65 74 3a 20 45 55 43 2d 4b 52 2c
75 74 66 2d 38 3b 71 3d 30 2e 37 2c 2a 3b 71 3d
30 2e 37 0d 0a 4b 65 65 70 2d 41 6c 69 76 65 3a
20 33 30 30 0d 0a 43 6f 6e 6e 65 63 74 69 6f 6e
3a 20 6b 65 65 70 2d 61 6c 69 76 65 0d 0a 52 65
66 65 72 65 72 3a 20 68 74 74 70 3a 2f 2f 6f 2d
67 61 6d 65 2e 63 6f 2e 6b 72 2f 68 6f 6d 65 2e
70 68 70 0d 0a 49 66 2d 4d 6f 64 69 66 69 65 64
2d 53 69 6e 63 65 3a 20 4d 6f 6e 2c 20 30 37 20
41 70 72 20 32 30 30 38 20 30 39 3a 34 35 3a 32
32 20 47 4d 54 0d 0a 49 66 2d 4e 6f 6e 65 2d 4d
61 74 63 68 3a 20 22 31 36 37 00 00 31 00 00 00
00 00 00 00 68 6f 73 74 20 31 39 32 2e 31 36
===== TCP DATA (CHAR) =====
GET /js/tw-sack.js HTTP/1.1
Host: o-game.co.kr
User-Agent: Mozilla/5.0 (X11; U; Linux i686; ko-KR; rv:1.8.1.13) Gecko/20080325 Ubuntu/7
.10 (gutsy) Firefox/2.0.0.13
Accept: */*
Accept-Language: ko-kr,ko;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Encoding: gzip,deflate
Accept-Charset: EUC-KR,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Referer: http://o-game.co.kr/home.php
If-Modified-Since: Mon, 07 Apr 2008 09:45:22 GMT
If-None-Match: "1671host 192.16
<<<<< End of Data >>>>>

===== Datalink layer =====
0: 50: 18: 19: 66: C -----> 0:14:22:9A:53:74
ether_type -> 800

===== IP HEADER =====
192.168.10.110 -----> Version : 4
Header Length : 5
Service : 0
Identification : 40
Fragment Offset : 16384
Time to Live : 64
Checksum : 38167
```

## \* 컴퓨터 인증화면



```
pchero@MyNote: ~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:14:22:9A:53:74
          inet addr:192.168.10.110  Bcast:192.168.10.255  Mask:255.255.255.0
          inet6 addr: fe80::214:22ff:fe9a:5374/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:112284 errors:0 dropped:0 overruns:0 frame:0
          TX packets:133009 errors:0 dropped:0 overruns:0 carrier:0
          collisions:31858 txqueuelen:1000
          RX bytes:89858439 (85.6 MB)  TX bytes:12674045 (12.0 MB)
          Interrupt:18

eth1      Link encap:Ethernet  HWaddr 00:14:A5:86:62:D5
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:27208 errors:0 dropped:6 overruns:0 frame:0
          TX packets:507 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:5115886 (4.8 MB)  TX bytes:21294 (20.7 KB)
          Interrupt:10 Base address:0x8000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:1 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:87 (87.0 b)  TX bytes:87 (87.0 b)

pchero@MyNote: ~$
```

## 3. 참조

1. <http://kldp.org> : .한국 리눅스 문서 프로젝트
2. <http://www.tcpdump.org> : tcpdump 홈페이지
3. <http://phpshool.com> : php 개발자 커뮤니티
4. <http://blog.naver.com/linuxint?Redirect=Log&logNo=100007507589> : 곱단이네 블로그
5. Unix Network Programming – 스티븐슨

## 4. 요구등급

- 모든 사항은 독자수행하였음.