[R E P O R T]

정보통신공학전공
200301582
김성태

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 캡춰하고자하는 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 protocal), 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 : 캡춰한 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 않는 다면 모든 패킷들이 그 대상이 될 것이다. 일단 주어지면, 아무리 패킷들이 많아도 조 건식에 부합하는 패킷만을 출력한다.

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

type : 주어진 값의 종류가 무엇인지를 나타낸다. 가능한 type들은 'host', 'net', 'port'가 있다. type이 없는 값들은 type을 host라 가정한다.

dir : id로 부터의 어떤 특정한 전송 방향을 나타낸다. 가능한 방향은 'src', 'dst', 'src or dst', 'src and dst'이다. 만약 방향이 정해지지 않았다면, src or dst 라 가정한다. "For `null' link layers (i.e. point to point protocols such as slip) the inb ound 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 프로토콜 들이 그 대상이 된다.

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

사용 가능한 ***Primitive***들

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 sour ce나 destination 항목은 host이지만, IP source와 destination은 host가 아닐 때를 말한다.
dst net NET
: 패킷의 IP destination 주소가 NET의 network number를 가지고 있을 때 참이 다.
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 가 지고 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.
ehter broadcast
: 패킷이 ethernet broadcast 패킷이라면 참이다. ehter는 생략 가능하다.
ip broadcast
: 패킷이 IP broadcast 패킷이라면 참이다.
ether multicast
: 패킷이 IP multicast 패킷이라면 참이다.
ether proto protocol
: 패킷 이 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.

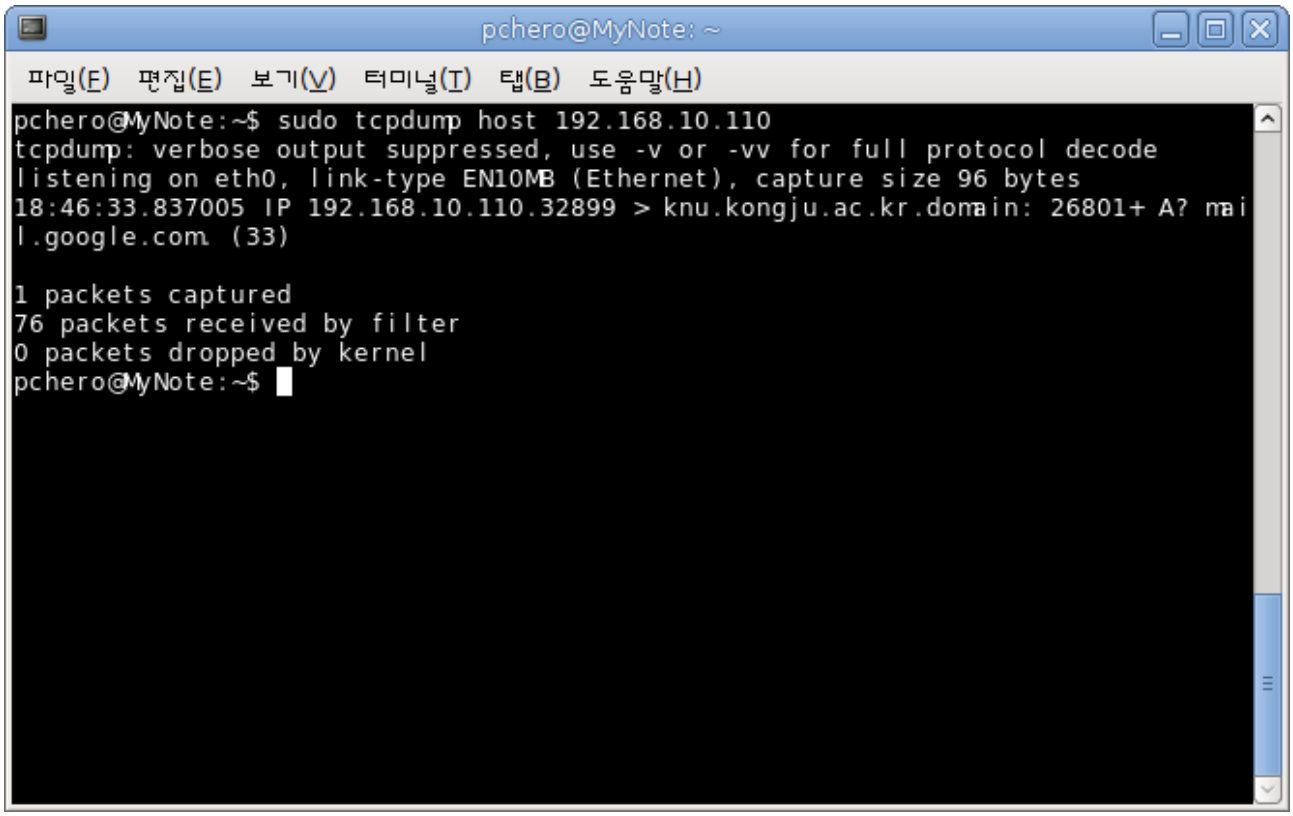
proto : ether, fddi, ip, arp, rarp, tcp, udp, icmp
expr : indicate Byte offset of packet of proto
size : optional. indicate the size of bytes in field of interest
default is one, and can be two or four

: RELOP

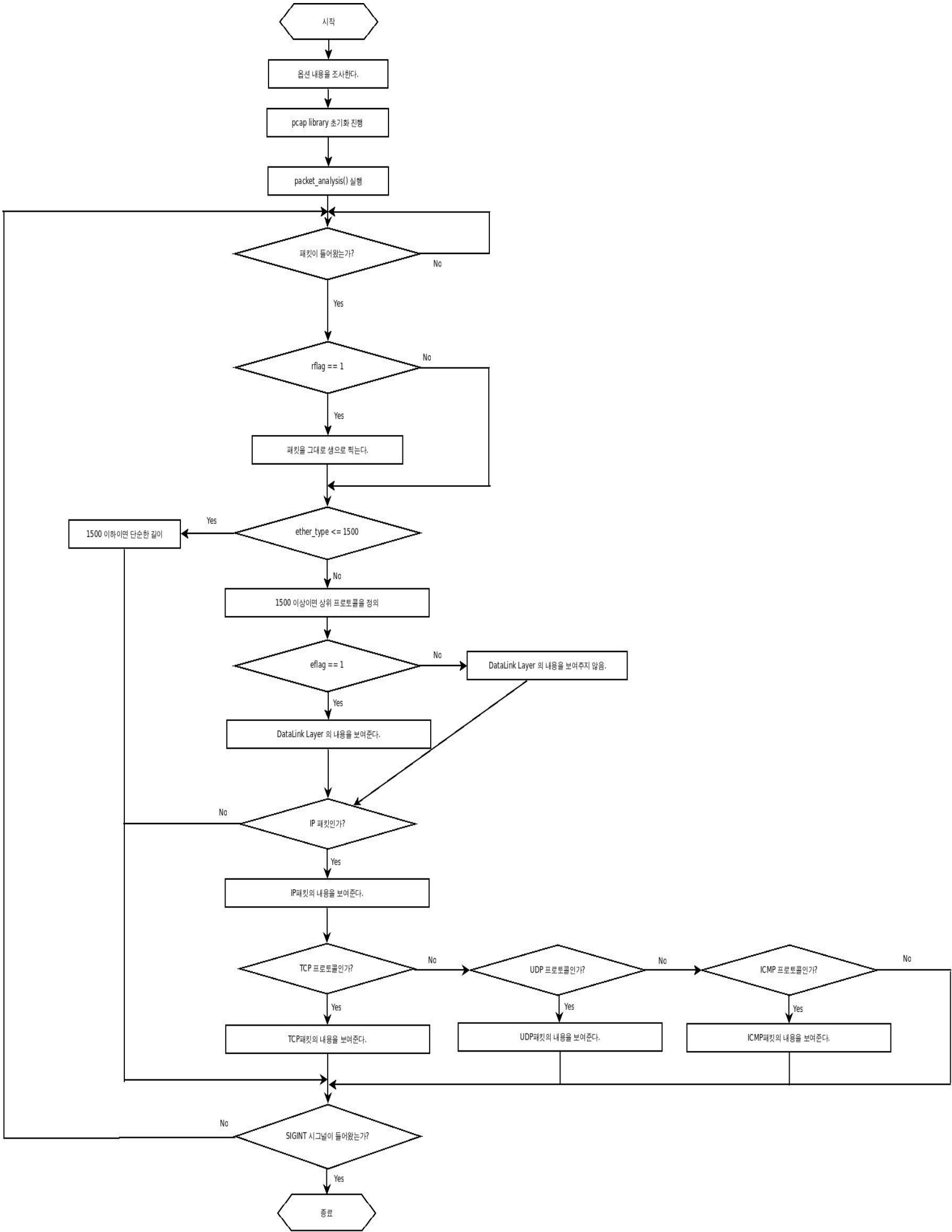
!=, =, <=, >=, etc.

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 직 접 원하는 패킷인지를 가려낼 수 있기 때문에, 보다 정밀한 사용이 가능하게 된다.

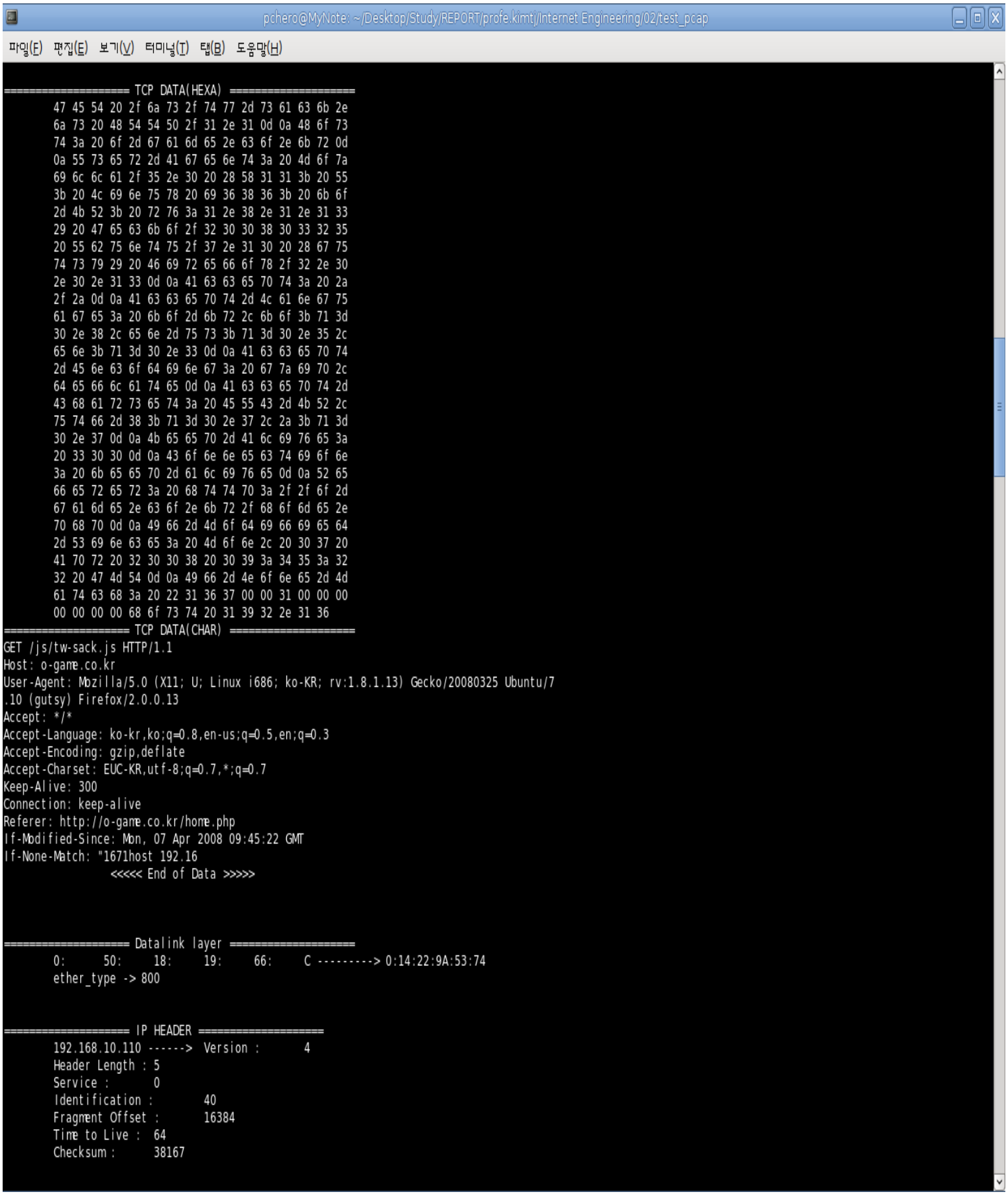
* tcpdump 사용 모습



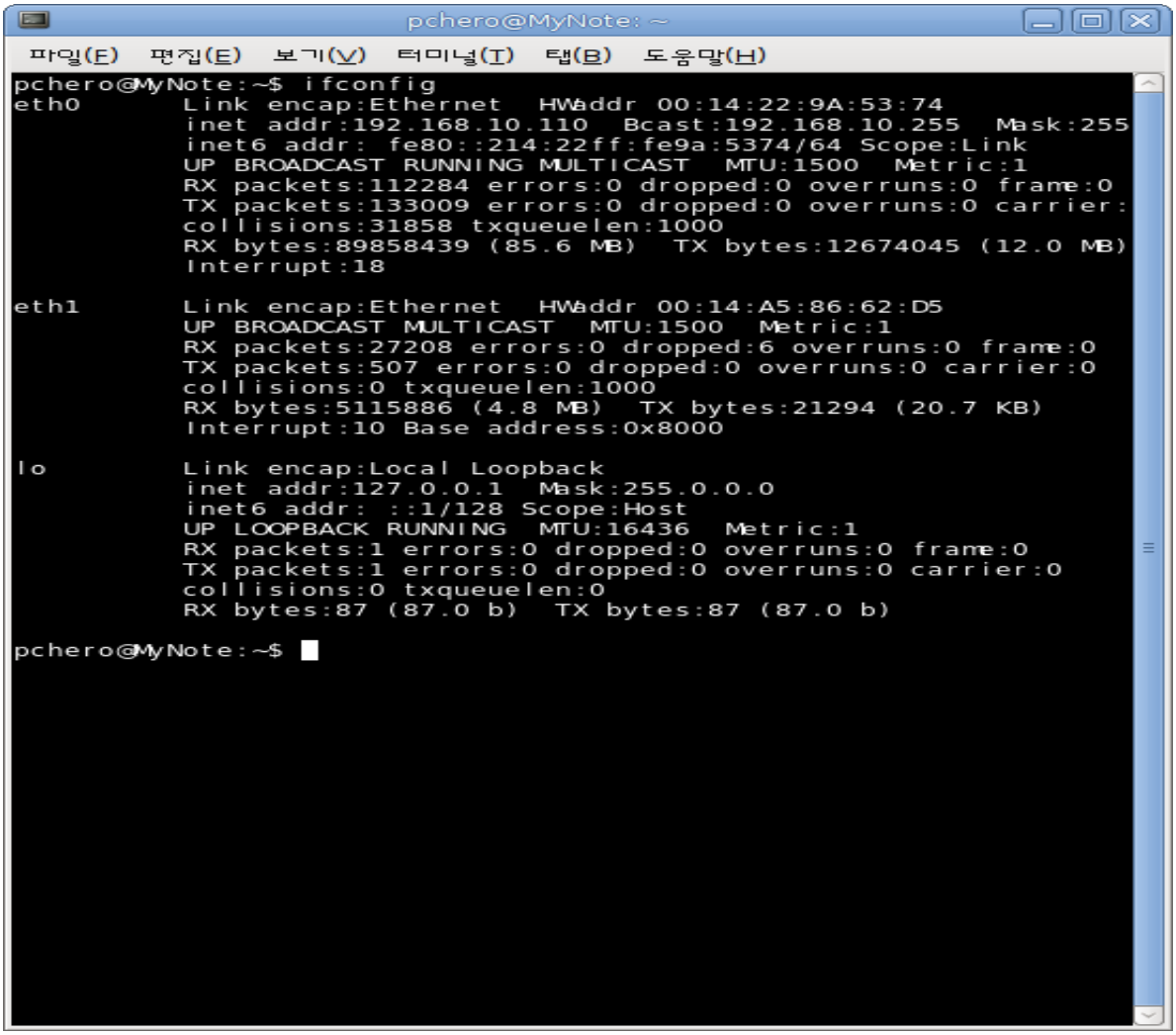
* pcap library 를 이용한 패킷 분석기 Flowchart



* pcap 프로그램 사용모습



* 컴퓨터 인증화면



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.