1. NAT 개념

NAT(Network Address Translation)란 내부 네트워크에서 외부로 나가는 패킷들의 주소를 외부 네트워크 주소로 변환하고 또한, 그 패킷에 대한 응답 패킷의 목적지 주소를 다시 패킷이 발송된 내부 네트워크 주소로 변환하여 주는 기능입니다.

일반적으로 NAT 기능을 사용하게 되면 네트워크 외부로 나가거나 들어오는 각 요구들이 모두 주소 변환과정을 거치기 때문에 보안을 확실하게 하는데 도움이 됩니다. 또한 요청들을 제한하거나 인증하는 등의 추가적인 보안을 제공할 수 있습니다.

내부 네트워크에서 필요한 만큼의 공인 IP 주소가 없는 경우에도 모든 PC등에서 외부 네트워크(인터넷)에 접속이 가능하도록 하여 줍니다.

2. 각 NAT 별 개념 정리

< Static NAT >

(외부 IP Address) : (내부 IP Address)
1 : 1

(외부 <-> 내부)
내부 IP 주소 하나에 외부 IP 주소 하나를 할당하는 1:1 방식의 주소 변환입니다.

< Dynamic NAT >

(외부 IP Address) : (내부 IP Address)
m : n (단, m > n 인 경우가 일반적)

(내부 -> 외부)
Dynamic NAT는 주로 외부 IP 보다 내부 네트워크의 실제 Host 수가 적을 경우, 또는 모든 내부 IP 주소를 일일이 외부용 IP 주소로 고정적으로 할당할 필요가 없을 경우에 사용합니다.

입력된 외부 네트워크 사용을 위한 IP 주소들은 Pool 방식으로 설정됩니다. 또한, 하나의 실재하는 내부 IP 주소에, Pool에 있는 하나의 외부 네트워크 IP 주소가 한번 할당

되면 해당 내부 IP 주소로부터 발생하는 모든 Session들은 이미 할당된 외부용 IP 주소를 통해 통신합니다.

만약, 이러한 방식으로 Pool에 할당된 외부 IP 주소가 모두 사용되고 있을 때, 다른 내부 IP 주소가 추가 할당을 요청하지만, 이미 외부용 IP Pool에는 더 이상 할당해줄 수 있는 외부용 IP가 없기 때문에 외부로의 접근이 불가능합니다. 이러한 경우에도 외부로의 접근이 가능하도록 하려면 PAT 기능을 설정해야 합니다. PAT 기능이 활성화 되어 있다면 자동으로 PAT방식으로 외부에 접속할 수 있도록 하여 줍니다.

< PAT >

(외부 IP Address) : (내부 IP Address)

h : n

(일반적으로 h < n, 내부의 IP 주소가 같은 외부 주소 IP로 변환, Port를 바꾸어가면서 변환.) (내부 -> 외부)

PAT는 하나 또는 하나 이상의 외부용 IP 주소를 여러 개의 내부 IP 주소들이 공유할 경우에 사용되며, 이 같은 경우 같은 외부용 IP 주소로 변환이 되지만 Port를 바꾸어가면서 변환하여 주는 방식입니다.

Static NAT나 Dynamic NAT로 변환하도록 설정되지 않은 내부 IP 주소들, 또는 Dynamic NAT에 설정은 되어 있으나 외부용 IP 주소 Pool이 고갈되어 더 이상 할당받을 수 없는 주소들은 PAT 변환 적용을 받습니다.

NXG 제품군에서 제공되는 PAT는 PAT Pool을 이용한 PAT 적용 방식과 Masking 값을 이용한 Source별 적용 방식이 있습니다.

- PAT Pool을 이용한 PAT 적용 방식 : 외부용 IP 주소를 한 개 이상 등록한 후 내부 IP가 Static NAT나 Dynamic NAT의 적용을 받지 못했을 때 등록해 놓은 외부용 IP 주소의 임의 Port 번호를 사용하여서 주소 변환을 수행합니다. 이때 처음 선택된 외부용 IP주소의 Port 번호가 모두 사용중인 경우(NXG 제품군은 빈 Port를 찾기 위해 20번 Retry를 수행합니다.) 등록된 다른 외부용 IP 주소를 상대로 동일한 동작을 다시 수행합니다.

- Masking 값을 이용한 Source별 PAT 적용 방식 : PAT Pool을 이용한 PAT

적용을 Mask값을 달리하여 복수 개 등록함으로써 내부의 서로 다른 여러 개의 네트워크마다 서로 다른 PAT Pool을 이용한 주소 변환을 수행할 수 있습니다.

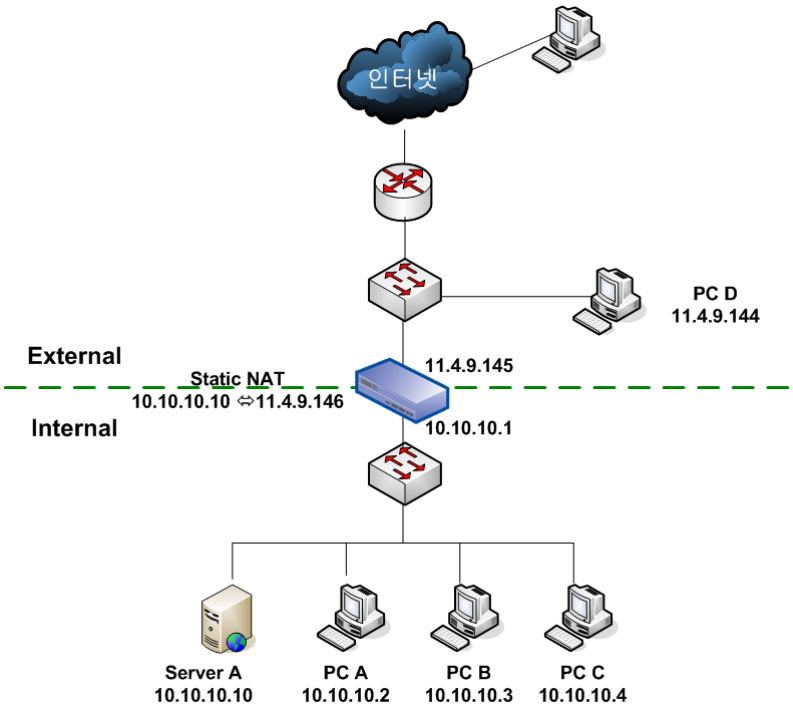
< LSNAT >

(외부 IP Address) : (내부 IP Address)
1 : n

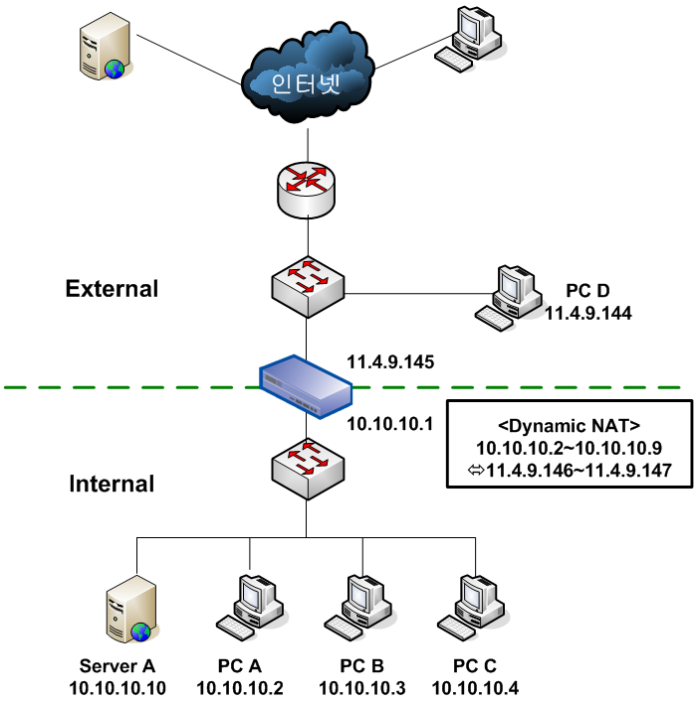
(내부 <- 외부)
LSNAT(Load Sharing NAT)는 Distributed NAT라고도 하는데, 하나의 외부 IP주소를 복수 개의 내부 IP 주소에 대응시켜, 외부 망에서 해당 IP 주소를 목적지로 하는 모든 세션을 할당된 각 내부의 IP 주소로 분산 변환합니다. 이 방식은 하나의 IP 주소에 많은 요청이 들어올 겨우 그 부하를 분산하여 수행속도가 저하되지 않게 하는 것이 주요 목적입니다.

3. 테스트 환경 구성도

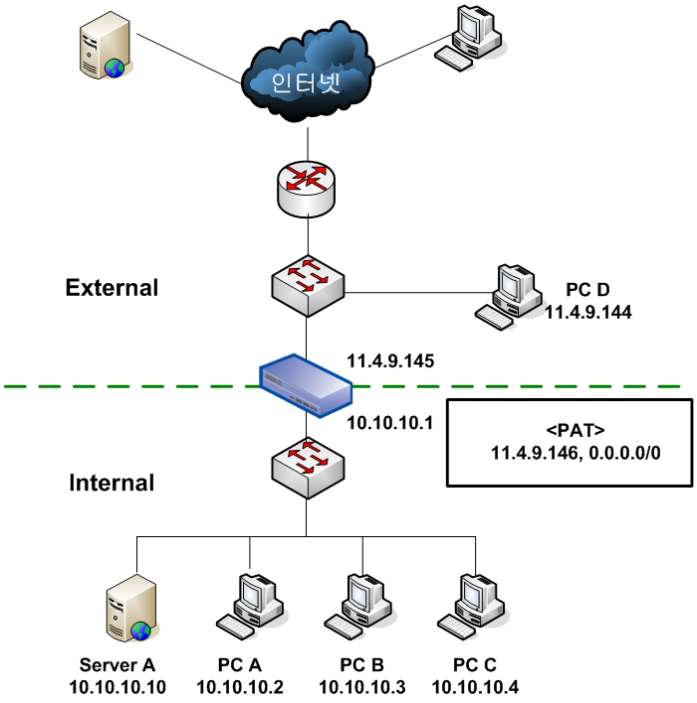
<Static NAT 테스트 환경>



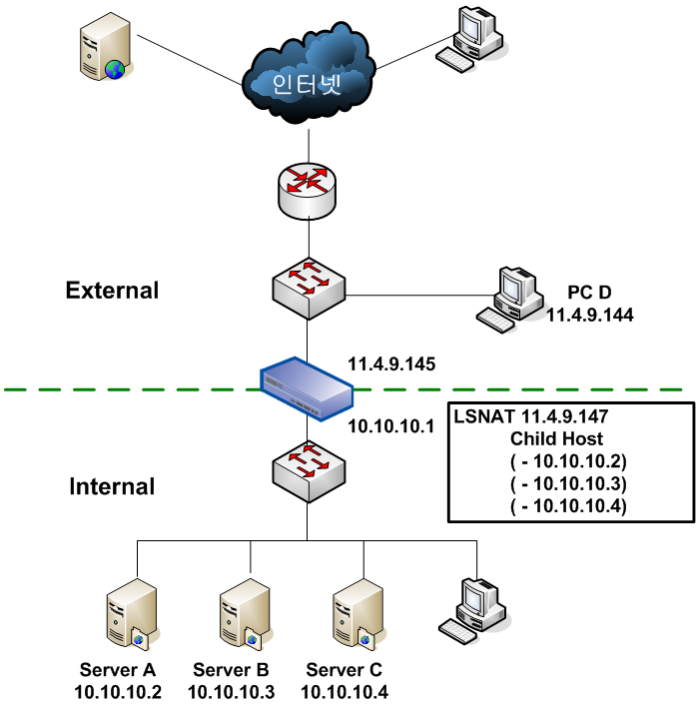
<Dynamic NAT 테스트 환경>



<PAT 테스트 환경>



<LSNAT 테스트 환경>



4. 테스트 환경 구성

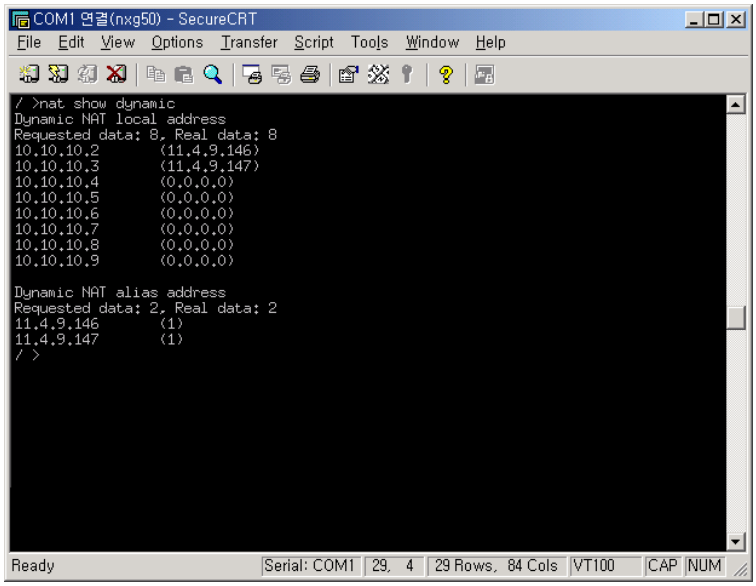
기본적인 테스트 환경을 구성할 때에는 "3. 테스트 환경 구성도"를 참고하시면 됩니다.

만약 NAT에 대해서 잘 모르신다면 운영자 매뉴얼 등의 문서를 참고하십시오.

5. NAT 테스트 시에 알아야 할 점.

다음의 명령어들은 NXG에 콘솔 접속 후 상태확인을 할 때 사용합니다. 이 외에도 다양한 명령어 형식이 있습니다. 이를 확인하려면 "nat"만 입력하시면 사용하실 수 있는 명령어 형식이 나타납니다.

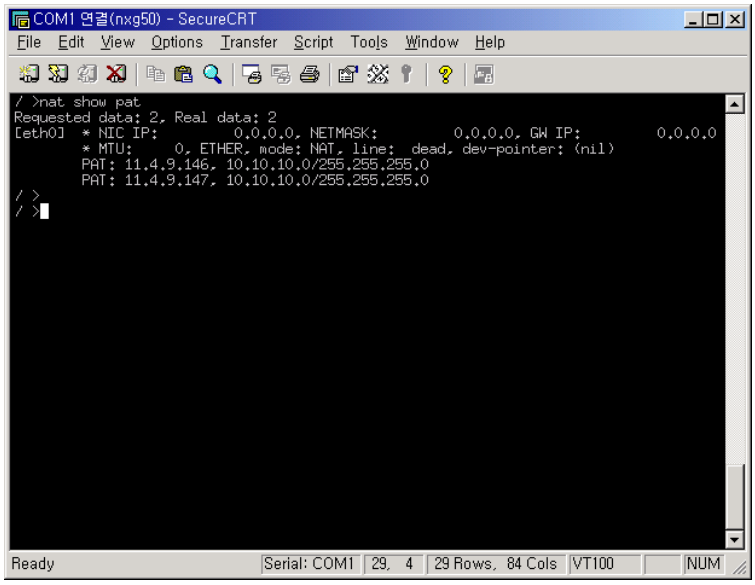
- Dynamic NAT의 환경설정을 확인하는 명령어
  i. "nat show dynamic"



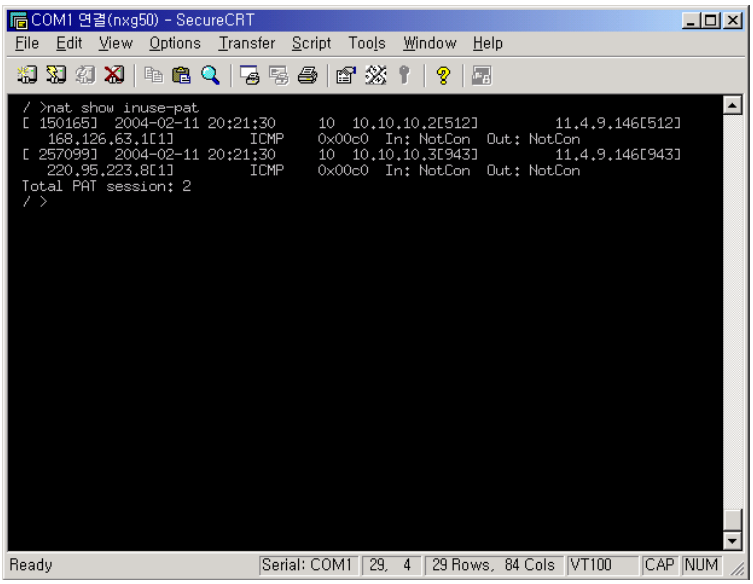
<nat show dynamic의 실행 예>

- PAT에 관련하여서 환경 설정을 확인하는 명령어

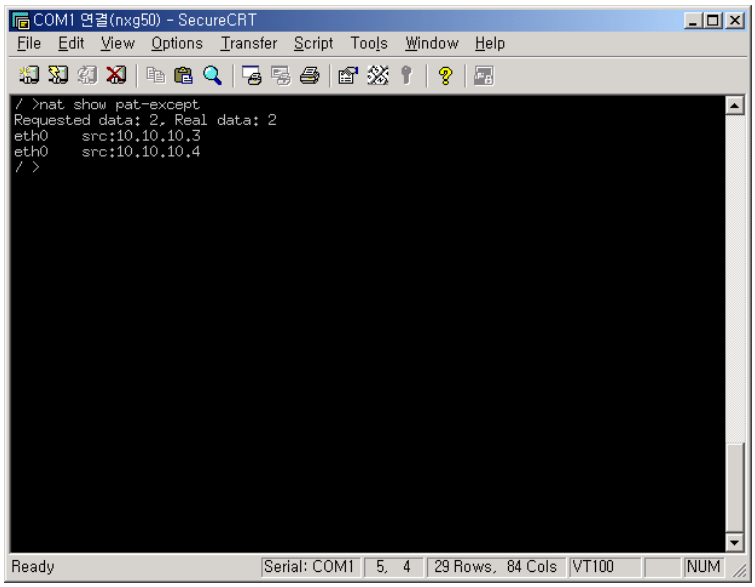
i. nat show pat (현재 PAT 설정 확인)



ii. nat show inuse-pat (현재 사용하고 있는 PAT의 포트 번호 및 정보확인 명령)



iii. nat show pat-except (현재 설정된 PAT 제외 리스트를 확인하는 명령)



-
-
-

tcpdump 명령어 정리

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 tcpdump로부터 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 하나하나를 처리하는데 시간이 더 걸릴 뿐만아니라 패킷 버퍼의 사이즈도 자연히 작아지게 되어 손실되는 패킷들이 발생할 수 있기 때문이다. 또, 작게 잡으면 그만큼의 정보를 잃게되는 것이다. 따라서 가급적 캡춰하고자 하는 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.

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 primitive 들로 구성되어 있다. primitive 들은 보통 하나 혹은 몇개의 qualifier 들 다음에 오는 하나의 값으로 이루어진다. Qualifier 들은 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 dst라 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 프로토콜들이 그 대상이 된다.

이 밖에도 위의 패턴을 따르지 않는 Primitive 들이 존재한다(gateway, broadcst, less, greater, 산술식).

좀 더 정교한 조건식들을 사용하려면, 'and(&&)', 'or(||)', 'not(!)'들을 사용하여 여러 primitive 들을 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 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 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*
  패킷이 ether type의 protocol이라면 참이다. protocol은 ip, arp, rarp 중에 하나 혹은 몇개가 될 수 있다. ip proto protocol에서와 마찬가지로 ip, arp, rarp는 escape되어야 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*
  - 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

o 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.

또한 tcpdump를 사용할 때 알아야 할 점으로는 tcpdump 결과를 보면 항상 NAT 처리 이후의 주소가 보이는데 그 이유는 tcpdump process가 NAT process 이후에서 동작하기 때문입니다. 즉 Packet이 나갈 때는 변환된 NAT 주소로 보이지만 들어올 때는 원래 IP로 보입니다.

PING 명령어를 사용하여서 테스트 시에 외부 인터넷 망으로 Ping 명령을 보내는 것과 N 테스트 하는 NXG의 외부망에 연결된 PC(단, 외부 인터넷 망의 서버가 아닌)로 보내는 것을 병행해야 한다. 그 이유는 우리 회사의 네트워크 망 앞쪽에도 방화벽이 설치되어 있기 때문에 그 영향을 받을 수가 있다는 것을 고려 해야 하기 때문이다.

6.