리눅스 2.4 NAT 하우투

Rusty Russell, mailing list netfilter@lists.samba.org v1.0 Tue Mar 21 17:06:19 EST 2000

이 문서는 2.4 리눅스 커널에서 어떻게 마스쿼레이딩, 투명한 프록시, 포트 포워드 그리고 다른 넷워크 주소바꾸기를 어떻게 하는지를 설명한다.

목 차

제 1 절 소개 **2**

제 2 절 어디에 공공 웹 사이트와 리스트가 있을까요 ? **2**

2.1 NAT 가 뭘까요 ? . . . 2

2.2 왜 NAT를 사용할려고 할까요 ? . . . 2

제 3 절 두가지 형태의 NAT **3**

제 4 절 2.0, 2.2 커널로부터의 빠른 전환 **3**

4.1 나는 단지 마스쿼레이딩 하기만 원한다... 도와죠.. . . . 4

4.2 ipmasqadm은 어떻하죠 ? . . . 4

제 5 절 NAT 조절하기 **5**

5.1 iptables 사용할때의 단순한 선택 . . . 5

5.2 어떠한 패킷을 조작할 것인지, 섬세한 선택 . . . 6

제 6 절 패킷을 어떻게 조작할지를 알려주기 **6**

6.1 시작점 NAT . . . 6

6.1.1 마스쿼레이딩 . . . 6

6.2 목적지 NAT . . . 7

6.2.1 방향 재설정 . . . 7

6.3 Mappings In Depth . . . 7

6.3.1 범위로 다수의 주소를 선택하기 . . . 7

6.3.2 Creating Null NAT Mappings . . . 7

6.3.3 기본적인 NAT 행동양식 . . . 7

6.3.4 Implicit Source Port Mapping . . . 8

6.3.5 NAT 가 실패했을 경우 어떤일이 생길까요 ? . . . 8

6.3.6 복수의 설정, 중복, 충돌 . . . 8

6.3.7 지역적으로 생성된 접속의 목적지를 변경하기 . . . 9

제 7 절 특수한 프로토콜 **9**

제 8 절 NAT 에 대한 경고 **9**

제 9 절 감사의 말씀 **9**

제 1 절 소개

얌전한 독자 여러분, 환영합니다.

여러분은 환상적인 (그리고 가끔 무시무시한) NAT의 세상으로 빠져들게 되 었습니다. 네트워크 주소 번역(Network Address Translation, NAT), 그리고 이 하우투는 2.4 리눅스 커널의 세법 정확한 가이드가 되어 줄 것입니다.

리눅스 2.4에서 패킷을 조작하는 기반구조를 '넷필터(netfilter)라고 합 니다. 이것의 최상위에 NAT가 있고 이것은 이전 커널로 부터 완전히 재이행 되었습니다.

제 2 절 어디에 공공 웹 사이트와 리스트가 있을까요 ?

세개의 공공 웹 사이트가 있습니다.

- Thanks to *Penguin Computing* <http://antarctica.penguincomputing.com/~netfilter/>.
- Thanks to *The Samba Team and SGI* <http://www.samba.org/netfilter>.
- Thanks to *Jim Pick* <http://netfilter.kernelnotes.org>.

공공의 넷필터 메일링 리스트는 *Samba's Listserver* <http://lists.samba.org>.

2.1 NAT 가 뭘까요 ?

정상적으로 패킷은 그것의 시작점에서 부터 목적지까지 많은 다른 링크를 경유하게 됩니다. 내가있는 오스트렐리아에서는 약 19개를 거치는 군요. 이 들 링크중 어떤것도 여러분의 패킷을 실제로 변경하지는 않습니다. 단지 이 것을 제 방향으로 보내주기만 합니다.

이러한 링크중 NAT를 사용하는 곳을 만나면 이 패킷은 그 시작점이나 목적 지가 지나는 동안 변화되게 된다. 짐작하겠지만, 이것은 시스템이 작동하 는 방식이 아니다. 그래서 NAT는 언제나 crock 같은 것이다. 일반적으로 NAT를 실행하는 링크는 패킷을 어떻게 조작했는지를 기억하며 응답 패킷 이 지나갈때 그 응답 패킷을 원래대로 되돌려 모든 작동이 가능 하도록 한 다.

2.2 왜 NAT를 사용할려고 할까요 ?

완벽한 세상에서는 여러분은 사용하려 않을 것입니다. 그러나 주 이유는 다음과 같습니다.

인터넷에 모뎀으로 연결

여러분이 모뎀접속하였을때 대부분의 ISP들은 하나의 IP 주소를 부여합니다. 여러분은 어떠한 시작점의 주소에서도 패킷을 보낼 수 있습니다만 이 시작점의 IP 주소에서 보내진 패킷에게만 응답이 있을 것입니다. 여러분이 여러개의 다른 기계(홈 네트워그에서처럼)에서 이 하나의 링크를 통하여 인터넷에 접속하기를 원한다면 NAT가 필요할 것입니다.

이것이 요즘 NAT를 사용하는 가장 흔한 이유이고 리눅스 세상에서는 '마스쿼레이딩'으로 알려져 있죠. 나는 이것을 SNAT라고 부르는데, 왜냐하면 첫번째 패킷의 **시작점** 주소를 바꾸는 것이기 때문입니다.

복수의 서버들

가끔 여러분의 네트워크로 가는 패킷의 방향을 바꾸고자 할 것입니다. 종종 이것은 (위에서 기술했듯이) 여러분은 하나의 IP 주소를 가지고 있지만 여러분의 하나의 실제 IP 주소 뒤에 여러사람이 참여하고자 할 때입니다. 여러분이 들어오는 패킷의 목적지를 다시 고쳐줌으로 이러한것을 할수 있습니다.

이것의 일반적인 변형은 부하분산으로, 여러대의 기계가 물려있을때 패킷을 이들에게로 나누어 주는 것입니다. 이러한 형태의 NAT를 이전 버전의 리눅스에서는 포트 포워딩이라고 하였습니다.

투명한 프록시 (Transparent Proxying)

가끔 여러분의 리눅스 박스를 지나가는 패킷이 리눅스 박스 자체의 프로그램으로 향하도록 하는 것을 좋아할 수가 있습니다. 이것을 사용하는 것은 투명한 프록시를 만드는 것으로 이 프록시는 바깥세상과 여러분의 네트워크사이에 존재하는 프로그램으로 이 둘사이의 통신을 속이는 것입니다. 이것이 투명하다는 것은 여러분의 네트워크는 이 프록시가 작동하지 않기 전까지는 프록시를 통해서 통신이 이루어진다는 것을 알수 없기 때문입니다.

Squid는 이런식으로 장동하도록 설정될수있고 이것을 이전의 리눅스 버전에서는 투명한 프록시 또는 방향재설정(redirection)이라고 불렀습니다.

제 3 절 두가지 형태의 NAT

나는 NAT를 두가지의 다른 형태로 구분하였습니다.: **시작점 NAT(Source NAT)** (SNAT) and **목적지 NAT(Destination NAT)** (DNAT).

Source NAT 은 첫 패킷의 시작점주소를 변경할때이다. 즉, 들어오는 접속을 변경하는 것이다. Source NAT 는 언제나 라우팅후에 이루어지고 패킷이 바깥으로 나가기 직전에 이루어 진다. 마스쿼레이딩은 SNAT 의 특병한 형태이다.

Destination NAT 은 첫 패킷의 목적지 주소를 변경할때이다. 즉, 접속이 어디로 향하는지를 변경할때 이다. Destination NAT은 언제나 라우팅이전에 이루어지고 패킷이 처음으로 들어왔을때 이루어 진다. 포트 포워딩, 부하분산, 그리고 투명한 프록시가 모두가 DNAT의 형태이다.

제 4 절 2.0, 2.2 커널로부터의 빠른 전환

아직 2.0 (ipfwadm)에서 2.2 (ipchains)로의 전환의 쇼크에 빠져 있는 사람에게는 미안하다. 좋은 뉴스와 나쁜 뉴스가 있다.

이전대로 ipchains와 ipfwadm을 사용할 수 있다. 이렇게 하기위해서는 단순히 넷필터의 배포판에 포함되있는 'ipchins.o' 나 'ipfwadm.o' 커널 모듈을 삽입하기만 하면 된다. 이들은 서로 호환되지 않는다. 그리고 어떤 다른 넷필터 모듈과도 함께 사용될 수 없다.

이들 모듈이 한번 설치되고 나서는 다음과 같은 차이점 외에는 정상적으로 ipchains와 ipfwadm을 사용할 수 있다.

- ipchains 의 -M -S 나 ipfwadm 의 -M -s 와 같은 마스쿼레이딩 타임아웃 셋팅은 작동하지 않습니다. 왜냐하면 이 타임아웃은 NAT 의 내부구조의 시간보다 길기 때문입니다. 그렇더라도 상관없죠.
- verbose 마스쿼레이드 리스팅의 init_seq, delta, previous_delta 는 언제나 '0' 입니다.

- '-Z -L' 을 이용하여 카운터를 리스트하고 '0'으로 만드는 것은 더이상 작동하지 않습니다. 카운터가 '0'으로 되지 않을 것입니다.

해커들은 다음에도 주의하여야 합니다.

- 마스쿼레이딩 중에도 61000-65095 포트를 뭉칠수 있습니다. 마스쿼레이딩 코드는 이 범위를 게임으로 여깁니다. 그러므로 프로그램은 이것을 사용할 수 없습니다.
- (이전에 기록되었던) 투명한 프록시에서 실제 연결 목적지를 알아내는 'getsockname' 해킹은 더이상 작동하지 않습니다.
- (이전에 기록되어있지 않던) 투명한 프록시의 환상을 종식시키는 bind-to-foreign-address 해킹은 더이상 적용되지 않습니다.

4.1 나는 단지 마스쿼레이딩 하기만 원한다... 도와죠..

이것은 대부분의 사람들이 원하는 것이죠. 여러분이 자동으로 할당되는 IP PPP dialup을 사용한다면(어떤 것인지 모른다면, 여러분이 사용하는 게 이거다.), 여러분의 모든 내부 네트워크로부터 오는 패킷은 PPP dialup 박스로부터 오는 것으로 보여지도록 하라고 여러분의 컴퓨터에게 이야기하기를 원할 것이다.

```
# 할당되는 IP 주소를 받은후 (POSTROUTING) NAT 테이블에 (-t nat) 밖으로 향하는
# ppp0 (-o ppp0) 의 모든 패킷들이 마스쿼레이드 되도록 접속하도록 (-j
# MASQUERADE) 새로운 규칙을 추가하면 (-A) 된다.
iptables -A POSTROUTING -t nat -o ppp0 -j MASQUERADE

# IP forwarding 을 켠다.
echo 1 > /proc/sys/net/ipv4/ip_forward
```

여기서 어떠한 패킷 필터링도 하지않은것에 주의하세요. 패킷 필터링을 위해서는 패킷 필터링 하우투를 보세요.

4.2 ipmasqadm은 어떻하죠 ?

사용자 입장에서 이것이 훨씬 적절하므로 나는 이전것과의 호환성에 대하여 많이 걱정하지 않습니다. 포트 포워딩하기위해서 간단히 'iptables -t nat'를 사용하면 됩니다. 예로 리눅스 2.2 에서는 다음과 같이 하였을 것입니다.

```
# 리눅스 2.2
# 1.2.3.4의 8080포트로 가는 TCP 패킷을 192.168.1.1의 80 포트로 가도록 포워딩
ipmasqadm portfw -a -P tcp -L 1.2.3.4 8080 -R 192.168.1.1 80
```

이제는 다음과 같이 하여야 합니다.

```
# 리눅스 2.4
# 라우팅 전에 규칙을 추가하도록 (-A PREROUTING) NAT 테이블에게 (-t nat) 이야기
# 하는데 대상 패킷은 TCP 패킷으로 (-p tcp) 원래 목적주소는 1.2.3.4 이고 (-d
# 1.2.3.4) 8080 포트이던것을 (--dport 8080) 새로운 목적 주소를 갖도록합니다.
# (-j DNAT) 그 새로운 목적 주소는 192.168.1.1 이고 80 포트를 이용합니다 (--to
# 192.168.1.1:80)
iptables -A PREROUTING -t nat -p tcp -d 1.2.3.4 --dport 8080 \
        -j DNAT --to 192.168.1.1:80
```

지역 접속에서도 위의 규칙이 적용되도록 할려면(즉, NAT 박스 자신이 1.2.3.4의 8080포트로 접속할려할때 192.168.1.1의 80포트로 연결되도록 할려면), 똑같은 규칙을 출력 체인에도 추가시켜주면 됩니다.

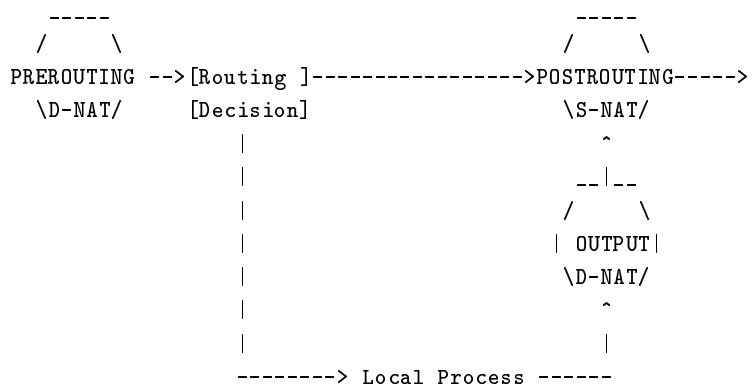
```
# 리눅스 2.4
iptables -A OUTPUT -t nat -p tcp -d 1.2.3.4 --dport 8080 \
        -j DNAT --to 192.168.1.1:80
```

제 5 절 NAT 조절하기

어떤 접속을 바꿀것인가 그리고 어떻게 바꿀것인가를 커널에게 알려주기 위하여 NAT 규칙들을 만들 필요가 있을 것입니다. 이렇게 하기 위해서, 우리는 아주 다재다능한 iptables를 사용합니다. 이것은 '-t nat' 이라는 특별한 옵션을 사용하여 NAT 테이블을 변경해 줌으로 가능합니다.

NAT 테이블에는 세개의 '체인'이라 불리는 규칙들이 있습니다. 각 규칙은 일치하는 것이 나올때까지 순서대로 시험됩니다. 세개의 체인은 PREROUTING(목적지 NAT의 겨우 패킷이 처음으로 들어올때), POSTROUTING (시작점 NAT의 경우 패킷이 떠나갈때), 그리고 출력 (지역 네트워크에서 생성된 목적지 NAT 패킷) 입니다.

아래 그림은 이러한 관계를 잘 나타내고 있습니다.



패킷이 지나갈때 위의 각 단계에서 어떠한 접속과 연관되어 있는지를 살펴봅니다. 새로운 접속이라면 이것이 적용되는 체인이 무었인지를 알아보고 이것을 어떻게 할 것인지를 알아보기 위하여 NAT 테이블을 살펴봅니다. NAT 테이블의 내용은 이후 이 접속으로부터 오는 모든 패킷에 적용됩니다.

5.1 iptables 사용할때의 단순한 선택

iptables 은 아래에 나열된 몇개의 기본 옵션이 있습니다. 모든 두줄의 옵션은 iptables가 다른 가능한 옵션과 구분할 수 있다면 줄여 쓸수 있습니다.

가장 중요한 옵션은 모든 NAT 작동에 사용되는 NAT 테이블을 지칭하는 '-t nat' 옵션입니다. 두번째로 중요한 옵션은 새로운 규칙을 그 체인의 끝에 추가하는 '-A' 옵션입니다.(예. '-A POSTROUTING'). '-I'옵션은 체인의 처음에 추가합니다 (예. '-I PREROUTING').

NAT를 이용할 패킷의 시작점이나 ('-s', '–source') 목적지 ('-d', '–destination')을 지정할 수 있습니다. 이 옵션뒤에는 하나의 IP 주소(예. 192.168.1.1)나 이름 (예. www.kernelnotes.org)이나 네트워크 주소 (예. 192.168.1.0/24 또는 192.168..0/255.255.255.0) 가 따라옵니다.

일치하는 인터페이서를 지정할 수도 있는데 들어오는 것('-i', '–in-interface')과 나가는 것('-o', '–out-interface')를 따로 지정합니다. 그러나 어떤 체인에 이 규칙을 적용할 것인지를 결정해 주어야 합니다. PREROUTING 에는 들어오는 인터페이서만 선택할 수 있고 POSTROUTING (그리고 출력) 에는 나가는 인터페이서만 선택할 수 있습니다. 이것을 잘못 선택하면 iptables는 에러를 낼 것입니다.

5.2 어떠한 패킷을 조작할 것인지, 섬세한 선택

위에서 시작점과 목적지의 주소를 지정할 수 있다고 이야기 했습니다. 시작점 주소의 옵션을 사용했다면 모든 시작점 주소가 조작 될것이고 목적지 주소의 옵션을 사용했다면 모든 목적지 주소가 조작될 것입니다.

TCP 나 UDP 처럼 프로토콜을 지정할 수 있습니다 ('-p', '–protocol'). 지정한 프로토콜의 패킷만 규칙과 일치하는지 검사할 것입니다. 이렇게 하는 주 이유는 tcp나 udp를 지정하면 추가의 옵션이 가능하기 때문입니다. 그 추가의 옵션은 '–source-port'와 '–destination-port' 입니다(줄임말. '–sport', '–dport').

이러한 옵션은 어떤 시작점이나 목적지 포트의 것만을 지정할 수 있게 합니다. 이것은 웹 요구들(TCP 80이나 8080 포트)를 다른곳으로 방향을 바꾸고 다른패킷은 그냥 두려고 할때 유용합니다.

이러한 옵션은 '-p' 옵션뒤에 따라와야 합니다. (이것은 그 프로토콜에 대한 확장 라이브러리를 로딩하는 부작용도 있습니다.) /etc/services 에서 얻을 수 있는 이름이나 포트 번호를 사용할 수도 있습니다.

패킷을 선택하는 여러 등급의 방법들은 고통스러운 세부사항들에 대한 설명이 나와있는 메뉴얼 페이지를 참고하세요. (`man iptables`).

제 6 절 패킷을 어떻게 조작할지를 알려주기

이제 조작할려는 패킷을 어떻게 선택하는지를 알아보았습니다. 규칙을 완성할려면 패킷을 어떻게 할 것인지를 커널에게 알려주어야 합니다.

6.1 시작점 NAT

시작점 NAT를 사고싶다면 ; 접속의 시작점 주소를 다른 것으로 바꿉니다. 이것은 POSTROUTING 체인에서 바깥으로 내보내 지기 바로 직전에 이루어 집니다. 이것은 아주 중요한 세부사항으로 이것은 리눅스 박스 자체에서는 바뀌기 전의 패킷을 인식한다는 것을 의미합니다. 이것은 또한 '-o' (밖으로 나가는 인터페이서) 옵션을 사용할 수 있다는 것을 의미합니다.

시작점 NAT 는 '-j SNAT'를 사용함으로 지정되고 '–to-source' 옵션으로 IP 주소, IP 주소의 범위, 포트, 포트의 범위 (UDP 와 TCP 프로토콜에서만) 를 지정할 수 있습니다.

```
## 시작점 주소를 1.2.3.4로 바꾸기
# iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 1.2.3.4

## 시작점 주소를 1.2.3.4 나 1.2.3.5 또는 1.2.3.6으로 바꾸기
# iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 1.2.3.4-1.2.3.6

## 시작점 주소를 1.2.3.4 포트 1-1023 으로 바꾸기
# iptables -t nat -A POSTROUTING -p tcp -o eth0 -j SNAT --to 1.2.3.4:1-1023
```

6.1.1 마스쿼레이딩

시작점 NAT의 특별한 경우로 마스쿼레이딩이라는게 있습니다. 이것은 표준 전화접속처럼 동적으로 할당되는 IP 주로를 사용할때만 사용해야 합니다. (정적인 IP 주소를 사용할때에는 위의 SNAT를 사용하세요.)

마스쿼레이딩할때 시작점 주소를 정확히 적어줄 필요는 없습니다. 이것은 밖으로 나가는 패킷이 이용하는 인터페이서를 이용할 것입니다. 그러나 더욱 중요한 것은, 그 연결이 끊어지면 그 접속은 잃어버리게 됩니다. 즉, 새로운 IP 주소를 받게되면 드문 돌연한 이상이 생길 수 있음을 의미합니다.

```
## ppp0 로 나가는 모든 패킷을 마스쿼레이드 하기
# iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
```

6.2 목적지 NAT

이것은 PREROUTING 체인에서 패킷이 들어오기 바로 직전에 이루어 집니다. 이것은 리눅스 박스 자체에서는 들어오는 패킷의 '실제' 목적지를 알게 된다는 것을 의미합니다. 이것은 또한 '-I' (들어오는 인터페이서) 옵션을 사용할 수 있음을 의미합니다.

지역 네트워크에서 생성된 패킷의 목적지를 조작할려면 출력 체인을 이용하면 가능한데, 이것은 거의 사용되지 않습니다.

목적지 NAT 는 '-j DNAT' 를 사용함으로 지정할 수 있고 '–to-destination' 옵션으로 IP 주소, IP 주소의 범위, 포트나 포트의 범위(UDP 와 TCP 프로토콜에서만)를 지정할수 있습니다.

```
## 목적지 주소를 5.6.7.8로 바꾸기
# iptables -t nat -A PREROUTING -i eth1 -j DNAT --to 5.6.7.8

## 목적지 주소를 5.6.7.8 이나 5.6.7.9 또는 5.6.7.10으로 바꾸기
# iptables -t nat -A PREROUTING -i eth1 -j DNAT --to 5.6.7.8-5.6.7.10

## 웹 트래픽의 목적지 주소를 5.6.7.8의 8080 포트로 바꾸기
# iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth1 \
        -j DNAT --to 5.6.7.8:8080

## 1.2.3.4로 가는 지역 패킷을 loopback으로 재설정 하기
# iptables -t nat -A OUTPUT -d 1.2.3.4 -j DNAT --to 127.0.0.1
```

6.2.1 방향 재설정

목적지 NAT 중의 특수한 경우로 마스쿼레이딩이 있습니다. 이것은 들어오는 인터페이스의 주소로 DNAT 를 이용하는 것과 정확히 일치하는 것으로 단순한 편의성을 위한 것입니다.

```
## 들어오는 80 포트의 웹 정보를 squid (투명한) 프록시로 보내기
# iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 80 \
        -j REDIRECT --to-port 3128
```

6.3 Mappings In Depth

대부분의 사람이 다루지 않을 내용주 NAT를 사용하는데 미묘한 내용이 좀 있습니다. 이상하게 여기는 사람을 위하여 이곳에 그것들에 대한 내용을 씁니다.

6.3.1 범위로 다수의 주소를 선택하기

IP 주소의 범위가 주어지면 최근에 가장 사용되지 않았던 IP 로 접속을 하도록 합니다. 이렇게 함으로 기본적인 부하균형을 이룹니다.

6.3.2 Creating Null NAT Mappings

'-j ACCEPT' 를 타겟으로 사용함으로 접속이 어떠한 NAT도 커치지 않고 되도록 할 수 있습니다.

6.3.3 기본적인 NAT 행동양식

기본적인 행동양식은 사용자가 설정한 규칙의 한도내에서 가능한 최소한의 접속만 하도록 합니다. 이것은 꼭 필요하지 않는 경우에 포트들을 재 설정할 필요가 없다는 것을 의미합니다.

6.3.4 Implicit Source Port Mapping

접속을 하는데 NAT 를 요구하지 않더라도 이전의 접속에 겹쳐서 접속을 시도하려하면 시작점 포트는 강제로 변경되도록 합니다. 좀 자주 사용되는 경우로 마스쿼레이딩의 경우를 봅시다.

1. 192.1.1.1의 1024 포트로부터 www.netscape.com의 포트 80으로 웹 접속을 하는 경우를 가정해 봅니다.
2. 그리고 이것은 시작점 IP 주소(1.2.3.4)를 가진 것으로 마스퀘레이더 된다고 가정합니다.
3. 마스쿼레이딩 박스는 1.2.3.4(이것의 외부 인터페이서 주소)의 1024 포트로부터 www.netscape.com의 80 포트로 웹 접속을 이룩합니다.
4. NAT 코드는 그 다음의 접속은 1025포트로 시작점 주소를 바꾸어 이 둘이 서로 충돌하지 않도록 합니다.

이러한 강제적인 시작점 설정이 이루어질때 포드는 다음의 세가지 클래서로 나누어 집니다.

- 512 이하의 포트
- 512 와 1023 사이의 포트
- 1024 보다 큰 포트

서로 다른 클래스로 포트가 설정되는 일은 없을 것입니다.

6.3.5 NAT 가 실패했을 경우 어떤일이 생길까요 ?

사용자가 요구한데로 접속을 유일하게 설정 할 방법이 없는 경우 이것은 DROP 됩니다. 이것은 잘못된 패킷의 경우처럼 어떠한 접속의 형태로도 구분되지 않을때, 그리고 메모리 제한을 벗어났을때 등의 경우에도 DROP 됩니다.

6.3.6 복수의 설정, 중복, 충돌

NAT 규칙에서 그 대상 패킷이 같은 범위를 가질 수 있습니다. 이 경우 충돌을 피할 정도로 NAT 코드는 똘똘합니다. 그러므로 두개의 규칙을 1.2.3.4로의 접속에 적용하면서 그 시작점 주소를 같이 192.168.1.1과 192.168.1.2로 정하는 것도 가능합니다.

더군다나, 그 주소가 주소설정을 하고있는 컴퓨터를 지난다면, 사용하고있는 실제 IP 주소도 겹쳐 쓸수가 있습니다. 그래서, 여러분이 설정된 네트워크(1.2.3.0/24)를 가지고 있지만 이러한 주소중 하나를 내부의 네트워크로 사용하고있고 하나는 기초적인 인터넷 주소인 192.168.1.0/24를 사용할 경우 여러분은 단순히 1.2.3.0 네트워그로 192.168.1.0/24를 시작점 주소로하는 NAT를 아무런 충돌에 대한 두려움 없이 사용할 수 있습니다.

```
# iptables -t nat -A POSTROUTING -s 192.168.1.0/24 -o eth1 \
        -j SNAT --to 1.2.3.0/24
```

같은 논리가 NAT 컴퓨터 자신이 사용하는 주소에도 해당됩니다. 이것이 마스쿼레이딩이 작동하는 방식입니다. (마스쿼래이더하는 컴퓨터 자신으로부터 오는 실제의 패킷과 마스쿼레이더 되는 패킷이 인터페이서의 주소를 공유함으로 작동하죠.)

더더욱, 많은 다른 타겟을 같은 패킷에 설정할 수도 있다. 그러면 이들은 공유될 것이다. 예를 들어 여러분이 1.2.3.5로는 아무것도 설정되지 않기를 바란다면 그렇게 할 수 있다.

```
# iptables -t nat -A POSTROUTING -s 192.168.1.0/24 -o eth1 \
        -j SNAT --to 1.2.3.0-1.2.3.4 --to 1.2.3.6-1.2.3.254
```

6.3.7 지역적으로 생성된 접속의 목적지를 변경하기

지역적으로 생성된 패킷의 목적지가 변경된다면 (출력 체인을 통해서) 그래서 그 패킷이 다른 인터페이서를 통해서 나가도록 한다면, 시작점 주소도 그 인터페이서의 주소로 바꿀 수가 있다. 예를 들어, 루프백 패킷의 목적지를 eth0로 향하도록 변경한다면 그 시작점도 127.0.0.1 에서 eth0의 주소로 바뀌게 된다. 다른 시작점 설정과는 달리 이것은 즉각 시행된다. 자연적으로 이 설정은 응답패킷이 왔을때 역전된다.

제 7 절 특수한 프로토콜

어떤 프로토콜을 NAT에 의해 사용되어 지는 것을 싫어한다. 이러한 프로토콜들을 위하여 두개의 확장이 기록되어야 한다. 하나는 그 프로토콜의 접속추적을 위한 것이고 다른 하나는 실제 NAT를 위한 것이다.

넷필터 배포판에는 현제 ftp를 위한 모듈로 ip_conntrack_ftp.o와 ip_nat_ftp.o 이다. 이 둘을 여러분의 커널에 모듈로 추가한다면 (또는 이들을 커널에 영구히 추가되도록 컴파일 하였다면) ftp 접속에 대한 어떠한 NAT도 작동할 것이다. 그러지 않으면 수동적인 ftp만을 사용할 수 있고 이렇게 되면 단순한 시작점 NAT 이상의 것을 신뢰성있게 실행할 수 없다.

제 8 절 NAT 에 대한 경고

접속에서 NAT 를 사용한다면 모든 패킷은 꼭 NAT 를 사용하는 컴을 이용해야 합니다. 그렇지 않으면 이것은 제대로 작동하지 않을 것입니다. 특히, 접속 추적 코드가 분절을 재결합할때는, 즉 접속 추적이 믿을 수 없게 될뿐 아니라 여러분의 패킷들은 제대로 전달되지 않고 분절은 보류되게 될 것입니다.

제 9 절 감사의 말씀

먼저 WatchGuard에 감사하고 내가 작업할때에 충분히 도움이 될 정도로 넷필터에 대한 많은 생각을 밀어준 David Bonn에게 감사합니다.

그리고 NAT의 엉성함을 배워가느라 야단 법석을 떨때 도와수신 모든분들, 특히 나의 기록을 읽어 주신분들께 감사합니다.

Rusty.

이상 수빈아빠의 엉터리 번역이었습니다. ^^;