Linux 2.4 NAT HOWTO

Rusty Russell, mailing list `netfilter@lists.samba.org`
Tradução para Português do Brasil: Guilherme Ude Braz (guilherme@linuxplace.com.br) Revision: 1.14
Date: 2001/05/26 20:26:48

Este documento descreve como fazer masquerading, proxy transparente, redirecionamento de portas, e outras formas de Troca de Endreço de Rede (Network Address Translations) com os kernels Linux 2.4.

Conteúdo

1 Introdução

Bemvindo, gentil leitor.

Você está prestes a entrar no fascinante (e algumas vezes horrendo) mundo do NAT: Network Address Translation (Troca de Endereço de Rede), e esse HOWTO será seu guia para kernels Linux 2.4 e futuros.

No Linux 2.4, uma estrutura para lidar com pacotes foi introduzida, seu nome é 'netfilter'. Uma camada externa provê NAT, completamente reimplementada em relação a kernels anteriores.

2 Onde encontrar o site oficial e a mailing list?

Estes são os sites oficiais:

- Agradecimentos para *Filewatcher* `http://netfilter.filewatcher.org/` .
- Agradecimentos para *The Samba Team and SGI* `http://netfilter.samba.org/` .
- Agradecimentos para *Harald Welte* `http://netfilter.gnumonks.org/` .

Para a mailing list oficial do Netfilter vá em

Netfilter List `http://www.netfilter.org/contact.html#list` .

2.1 O que é Network Address Translation (Troca de Endereço de Rede)?

Normalmente, pacotes em uma rede viajam de sua origem (por exemplo, o computador de usa casa) para seu destino (por exemplo, www.gnumonks.org) através de vários links. Nenhum destes links realmente alteram seu pacote: eles apenas reenviam o mesmo.

Se algum destes links fizesse NAT, eles iriam alterar a origem ou o destino do pacote. Como você pode imaginar, esta não é a forma pela qual seu sistema foi feito para trabalhar, logo NAT é sempre algo complexo.

Geralmente a máquina que faz NAT vai relembrar-se como o pacote foi alterado, e quando um pacote de resposta segue o caminho inverso, a máquina irá fazer a alteração reversa naquele pacote resposta, é assim que as coisas funcionam.

2.2 Porque utilizar NAT?

Em um mundo perfeito, você não usaria. Por enquanto, as razões são:

Conexões com a Internet via modem

A maioria dos provedores de acesso te dá um único endereço IP quando você conecta. Você pode mandar pacotes com qualquer endereço de origem que você quiser, mas apenas respostas para pacotes com esse endereço de origem retornarão para você. Se você quiser utilizar mais de uma máquina (uma rede local na sua casa) para conectar na Internet através deste único link, você precisará de NAT.

Esta é a maneira de NAT mais utilizada atualmente, e é conhecida como Masquerading no mundo Linux. Eu chamo isso de SNAT, porque você altera o endereço de **origem** do primeiro pacote.

Múltiplos Servidores

Às vezes, você quer mudar o destino dos pacotes que chegam na sua rede. Frequentemente isto ocorre porque (como dito acima) você tem apenas um endereço IP, mas quer que as pessoas alcancem servidores atrás do que tem o endereço IP real. Reescrevendo o destino dos pacotes que chegam, é possível implementar isso.

Uma variação comum disto é load-sharing, na qual a carga de pacotes é dividida entre máquinas. este tipo de NAT era chamado de port-forwarding em versões anteriores do Linux.

Proxy Transparente

Às vezes você pretende que cada pacote que atravesse sua máquina Linux é destinado a um programa específico. Isto é utilizado para fazer proxies transparentes: um proxy é um programa que fica entre sua rede e o mundo extertno, controlando a comunicação entre os mesmos. O termo transparente refere-se ao fato de que sua rede não saberá que está lidando com um proxy, a não ser, é claro, que seu proxy não funcione :).

O Squid pode ser configurado de forma a trabalhar desta maneira, e isto é chamado de redirection ou transparent proxying em versões Linux anteriores.

3 Os dois tipos de NAT

Eu divido NAT em duas categorias distintas: **NAT de Origem (Source NAT)** (SNAT) e **NAT de Destino (Destination NAT)** (DNAT).

Source NAT ocorre quando o endreço de origem do primeiro pacote é alterado: por exemplo, quando se muda de onde vem uma conexão. Source NAT sempre ocorre no momento de post-routing, logo antes que o pacote deixe o firewall. Masquerading é uma forma especializada de SNAT.

Destination NAT ocorre quando o endereço de destino do primeiro pacote é alterado: por exemplo, quando se altera o destino final de uma conexão. Destination NAT sempre ocorre no momento de pre-routing, quando o primeiro pacote está prestes a entrar no firewall. Port forwarding, load sharing, e proxy transparente são forma de DNAT.

4 Pequenas notas dos Kernels 2.0 e 2.2

Peço desculpas àqueles que ainda esperam para fazer a transição das versões 2.0 (ipfwadm) e 2.2 (ipchains). Há notícias boas e ruins.

Primeiramente, pode-se usar ipchains ou ipfwadm como antes. Para fazer isto, carregue os módulos do kernel 'ipchains.o' ou 'ipfwadm.o' que estão na sua distribuição do netfilter. Tais módulos são incompatíveis (você está avisado!), e não podem ser utilizados em conjunto com quaisquer outros módulos do netfilter.

Tendo estes módulos instalados, ipchains e ipfwadm podem ser utilizados normalmente, com as seguintes diferenças:

- Configurar masquerading timeouts com ipchains -M -S, ou ipfwadm -M -s não adianta. Uma vez que os timeouts são maiores na nova infraestrutura NAT, isso não faz muita diferença.
- Os campos init_seq, delta e previous_delta no verbose masquerade listing são sempre zero.
- Zerar e escutar os contadores ao mesmo tempo '-Z -L' não funciona mais: os contadores não serão zerados.

Hackers também notarão:

- É possível bloquear as portas 61000-65095 até fazendo masquerading. O código de masquerading code assumia que não havia ataques neste port range, abrindo uma brecha de segurança.
- O hack (não documentado)'getsockname', que permitia que programas transparent proxy achassem o destino real dos pacotes, não funciona mais.
- O hack (não documentado) 'bind-to-foreign-address' também não foi implementado; ele era utilizado para completar o serviço de proxy transparente.

4.1 Socorro! Eu só quero masquerading!

Masquerading é o que a maioria das pessoas quer. Se você tem um IP dinâmico via PPP (se você não sabe, você tem um), e quer apenas passar para sua máquina que todos os pacotes vindos da sua rede interna deve parecer que têm como origem sua máquina com uma conexão dialup PPP.

```
# Carregar o módulo NAT (isso carrega todos os outros.
modprobe iptable_nat

# Na tabela NAT (-t nat), adicionar uma regra (-A) após o routing
# (POSTROUTING) para todos os pacotes saindo por ppp0 (-o ppp0) dizendo para
# MASCARAR a conexão (-j MASQUERADE).
iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE

# Habilitar IP forwarding
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Note que você não está filtrando nenhum pacote aqui: para isso, veja o Packet Filtering HOWTO (Como fazer filtragem de pacotes): 'Misturando NAT e Filtragem de Pacotes'.

4.2 E o ipmasqadm?

O ipmasqadm era utilizado por um menor número de usuários, então não me preocupei muito com a compatibilidade de versões anteriores. Você pode utilizar 'iptables -t nat' para fazer port forwarding. No Linux 2.2 você faria:

```
# Linux 2.2
# Transformar pacotes TCP indo para a porta 8080 de 1.2.3.4 para a porta 80 de 192.168.1.1's
ipmasqadm portfw -a -P tcp -L 1.2.3.4 8080 -R 192.168.1.1 80
```

Agora, você faria:

```
# Linux 2.4
# Adicionar uma regra em pre-routing (-A PREROUTING) para a tabela NAT (-t nat) na qual
# pacotes TCP (-p tcp) indo para 1.2.3.4 (-d 1.2.3.4) na porta 8080 (--dport 8080)
# têm seu destino trocado (-j DNAT) para 192.168.1.1, porta 80
# (--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
```

Se você quer que esta regra altere também as conexões locais (a NAT box tentando conectar-se via telnet com o endereço 1.2.3.4 porta 8080, ela chegará no endereço 192.168.1.1 porta 80), você pode inserir a mesma regra na chain OUTPUT (que serve para pacotes de saídagerados localmente):

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

5 Controlando o que será submetido a NAT

Você precisa criar regras NAT as quais dirão ao kernel quais conexões serão alteradas, e como alterá-las. Para fazer isso, utiliza-se o versátil `iptables` e dizemos a ele para alterar a tabela NAT ao especificar a opção '-t nat'.

As regras da tabela NAT contêm três listas chamadas 'chains': cada regra é examinada em ordem, até que alguma delas corresponda-se com o pacote analizado. As três 'chains' são chamadas PREROUTING (para Destination NAT, quando os pacotes estão prestes a entrar), POSTROUTING (para Source NAT, assim que os pacotes saem), e OUTPUT (para Destination NAT de pacotes gerados localmente.

O diagrama abaixo ilustraria isso perfeitamente se eu tivesse talento artístico: :)

```
      -----                                       -----
     /     \                                     /     \
   PREROUTING -->[Decisão ]----------------->POSTROUTING----->
     \D-NAT/     [de Roteamento]                 \S-NAT/
                     |                              ^
                     |                            __|__
                     |                           /     \
                     |                          | OUTPUT|
```

```
|                          \D-NAT/
|                             ^
|                             |
----> Processamento Local ----
```

Em cada um dos pontos acima, quando um pacote passa, é verificado a conexão com a qual ele está associado. Se é uma nova conexão, é verificada a chain correspondente na tabela NAT para ver o que fazer com a mesma. A resposta dada será idêntica para todos os outros pacotes relacionados com tal conexão.

5.1 Seleção simples utilizando iptables

`iptables` tem um número de opções padrão conforme a lista abaixo. Todas as opções com dois hífens (–) podem ser abreviadas, desde que o `iptables` ainda possa diferenciá-las das demais opções disponíveis. Se seu kernel tem suporte a iptables via módulos, você precisará carregar o módulo ip_tables.o antes com o seguinte comando: 'insmod ip_tables'.

A opção mais importante é a seleção da tabela com '-t'. Para todas as operações NAT, será necessária a opção '-t nat' para a tabela NAT. A segunda mais importante é a opção '-A', que adiciona uma nova regra no fim da chain ('-A POSTROUTING'), ou '-I' para adiconá-la no início (eg. '-I PREROUTING').

Você pode especificar a origem ('-s' or '–source') e o destino ('-d' or '–destination') dos pacotes que sofrerão NAT. Tais opções podem ser seguidas de um simples endereço IP (192.168.1.1), um nome (www.gnumonks.org), ou um endereço de rede (192.168.1.0/24 ou 192.168.1.0/255.255.255.0).

Também podem ser especificadas as interfaces de entrada ('-i' ou '–in-interface') ou de saída ('-o' ou '–out-interface'). Qual das interfaces que poderá ser especificada dependerá da chain que receberá a regra: em PREROUTING você pode selecionar apenas a interface de entrada, e em POSTROUTING (e OUTPUT) apenas a interface de saída é selecionada. Se você fizer a opção errada, o `iptables` restornará um erro.

5.2 Pontos mais específicos sobre o tratamento dos pacotes

Foi dito acima que podem ser especificados endreços de origem e destino. Se o endereço de origem foi omitido, a regra servirá para qualquer origem. Se o endereço de destino for omitido, a regra serviá para qualquer destino.

Um protocolo específico também pode ser indicado ('-p' or '–protocol'), como o TCP ou o UDP. Ao selecionar um protocolo, apenas pacotes do mesmo se encaixarão na regra. A principal razão para selecionar um protocolo, é que isso permite algumas opções extras. Especificamente as opções '–source-port' e '–destination-port' (abreviadas como '–sport' and '–dport').

Essas opções permitem especificar que apenas pacotes com certa porta de origem e destino se encaixarão na regra. Isso é útil para redirecionamento de requisições web (porta TCP 80 ou 8080).

Todas essas opções devem seguir a '-p' (a qual carrega uma biblioteca compartilhada para aquele protocolo). Podem ser usados os números das portas ou um nome do arquivo /etc/services.

Todas as outras formas de se selecionar um pacote estão descritas em detalhes na página de manual do `iptables` (`man iptables`).

6 E agora, como tratar os pacotes?

Agora já sabemos como escolher os pacotes a serem tratados. Para completar, precisamos dizer ao kernel o que fazer com estes pacotes.

6.1 Source NAT

Você quer fazer Source NAT; mudar o endereço de origem das conexões para algo diferente. Isso é feito na chain POSTROUTING, logo antes de que o pacote saia da máquina. Isso é um detalhe muito importante, pois significa que qualquer tarefa da sua máquina Linux (routing, packet filtering) verá o pacote inalterado. Isso também significa que a opção '-o'(interface de saída) pode ser utilizada.

Source NAT é especificado com '-j SNAT', e a opção '–to-source' demonstra um endereço IP, um range de endereços IP, e uma porta opcional ou um range de portas (apenas para os protocolos TCP e UDP).

```
## Mudando o endereço de origem para 1.2.3.4.
# iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 1.2.3.4

## Mudando o endereço de origem para 1.2.3.4, 1.2.3.5 ou 1.2.3.6
# iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 1.2.3.4-1.2.3.6

## Mudando o endereço de origem para 1.2.3.4, portas 1-1023
# iptables -t nat -A POSTROUTING -p tcp -o eth0 -j SNAT --to 1.2.3.4:1-1023
```

6.1.1 Masquerading

Há um caso especial de Source NAT chamado masquerading, que só deve ser utilizado para endereços IP dinâmicos, como as conexões dialup (para endereços IP estáticos, utilize SNAT descrito acima).

No masquerading, não é necessário explicitar o endereço de origem: será utilizado o endereço de origem da interface de saída do pacote. Mas o mais importante é que se o link cair, o endereço de origem que estava sendo utilizado é descartado, dando lugar ao novo endereço de origem da interface quando o link for restabelecido.

```
## Mascarando tudo que sai pela interface ppp0.
# iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
```

6.2 Destination NAT

DNAT é feito na chain PREROUTING, assim que o pacote chega; isso significa que todas as outras tarefas da sua máquina Linux (routing, packet filtering) verão o pacote já indo para seu destino verdadeiro. A opção '-i' (interface de entrada) pode ser utilizada.

Para alterar o destino de pacotes gerados localmente, a chain OUTPUT deve ser utilizada, mas isso não é muito utilizado (além disso, mudar o destino para outro local que não a própria máquina Linux ainda não funciona – Abril 2001).

Destination NAT é especificada utilizando '-j DNAT', e a opção '–to-destination' especifica um endereço IP, um range de IPs, e uma porta opcional ou um range de portas (apenas para protocolos TCP e UDP).

```
## Mudando destino para 5.6.7.8
# iptables -t nat -A PREROUTING -i eth0 -j DNAT --to 5.6.7.8

## Mudando destino para 5.6.7.8, 5.6.7.9 ou 5.6.7.10.
# iptables -t nat -A PREROUTING -i eth0 -j DNAT --to 5.6.7.8-5.6.7.10

## Mudando destino do tráfego web para 5.6.7.8, porta 8080.
# iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth0 \
        -j DNAT --to 5.6.7.8:8080

## Redirecionar pacotes locais com destino a 1.2.3.4 para loopback.
# iptables -t nat -A OUTPUT -d 1.2.3.4 -j DNAT --to 127.0.0.1
```

6.2.1 Redirecionamento

Há um caso especializado de Destination NAT chamado redirecionamento: é uma simples conveniência, a qual equivale a fazer DNAT para o endereço da própria interface de entrada.

```
## Mandando tráfego web da porta-80 para nosso squid proxy (transparente)
# iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 80 \
        -j REDIRECT --to-port 3128
```

Lembre-se que o squid precisa ser configurado para saber que está sendo um proxy transparente!

6.3 Algumas curiosidades menos utilizadas...

Há alguns refinamentos do NAT os quais a maioria das pessoas nunca utilizará. Eles estão aqui documentados apenas para os curiosos.

6.3.1 Seleçao de múltiplos endereços em um range

Se um range de endereços IP é dado, o IP escolhido para uso é o que está sendo menos utilizado pelas conexões conhecidas. Isso porvê um balanceamento de carga (load-balancing) bem primitivo.

6.3.2 Criando mapeamentos NAT nulos

Você pode utilizar '-j ACCEPT' para aceitar uma conexão sem que haja nenhuma espécie de NAT.

6.3.3 Comportamento padrão do NAT

O comportamento padrão é alterar a conexão o mínimo possível, modificando apenas o definido pelo usuário nas suas regras. Isso significa que as portas não serão remapeadas a não ser que as regras mandem que elas sejam.

6.3.4 Mapeamento de portas de origem implícito

Mesmo quando nenhum tipo de NAT é requisitado para uma conexão, a alteração da porta de origem pode ocorrer de forma implícita, quando uma conexão requisita uma porta que está em uso. Considere um caso de masquerading, que gera esse tipo de situação:

1. Uma conexão web estabelecida pela máquina 192.1.1.1 vinda da porta 1024 para www.netscape.com porta 80.
2. Tal conexão é mascarada pela máquina de masquerading a fim de usar o endreço de IP de origem 1.2.3.4, que é o endereço de origem da máquina que realiza o masquerading.
3. A máquina que faz o masquerading tenta realizar a conexão web com www.netscape.com porta 80 vinda de 1.2.3.4 (endereço de origem da interface de saída) porta 1024.
4. O código de NAT alterará a porta de origem da segunda conexão para a porta 1025, assim as duas conexão não conflitarão.

Quando esse mapeamento de origem implícito ocorre, as portas são divididas em três classes:

- Portas menores que 512
- Ports entre 512 e 1023
- Portas acima de 1024.

Uma porta NUNCA será mapeada implicitamente para uma classe diferente.

6.3.5 O que ocorre quando NAT falha

Se não há formas de mapear uma conexão conforme requisitado pelo usuário, a mesma será descartada (DROP). Isso também se aplica a pacotes que não foram classificados como parte de qualquer conexão, devido a malformação, ou falta de memória da máquina, etc.

6.3.6 Mapeamentos múltiplos, overlap e conflitos

Você pode ter regras NAT que mapeiam pacotes para o mesmo range; o código NAT suficientemente inteligente para evitar conflitos. Não há nenhum problema em mapear os endereços de origem 192.168.1.1 e 192.168.1.2 para 1.2.3.4.

Além disso, você pode mapear para endereços IP reais e utilizados, desde que tais endereços passem pela máquina responsável pelo mapeamento. Então, se você tem uma rede válida (1.2.3.0/24), mas possui uma rede interna que utilize um IP privado (192.168.1.0/24), você pode realizar SNAT do endereço 192.168.1.0/24 para a rede 1.2.3.0, sem medo de quaisquer conflitos:

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

A mesma lógica se aplica aos endereços utilizados pela própria máquina que realiza NAT: essa é a forma que o masquerading funciona (compartilhando o endereço da interface de saída entre os pacotes mascarados e os gerados localmente).

Você também pode mapear os mesmos pacotes para alvos distintos, e estes serao compartilhados. Por exemplo, se você não quer mapear NADA para o endereço 1.2.3.5, você faria:

```
# 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 Alterando destino de pacotes gerados localmente

Se o destino de um pacote gerado localmente é alterado (pela chain OUTPUT), e isso leva o pacote a passar por uma interface diferente, e então o endereço de origem também é alterado para o endereço da interface de saída. Por exemplo, mudando o destino de um pacote loopback para sair por eth0 irá resultar na mudança do endereço de origem de 127.0.0.1 para o endereço de eth0; ao contrário dos outros mapeamentos de origem (que são feitos depois de tudo, na chain POSTROUNTING) esse é realizado imediatamente. Naturalmente, todos esses mapeamentos são revertidos no momento que os pacotes de resposta chegam.

7 Protocolos especiais

Alguns protocolos não se dão muito bem com NAT. Para cada um destes protocolos, duas extensões tiveram de ser escritas; uma para o acompanhamento do protocolo, e a outra para o atual NAT.

Dentro da distribuição do netfilter, há atualmente módulos para ftp: ip_conntrack_ftp.o e ip_nat_ftp.o. Se você carregar esses módulos para dentro de seu kernel (ou compilá-los permanentemente), qualquer tipo de NAT em conexões FTP deve funcionar. Se não funcionar, você só pode utilizar ftp passivo, e talvez nem isso funcione de forma confiável se você está fazendo mais que um simples Source NAT.

8 Algumas prevenções sobre NAT

Se você está fazendo NAT em uma conexão, todos os pacotes passando nas **duas** direções (entrando e saindo da rede) DEVEM passar pela máquina responsável pelo NAT, ou nada funcionará confiavelmente. Particularmente, o código de acompanhamento de conexões remonta fragmentos, o que significa que além do acompanhamento de conexões, toda a rede pode ter problemas, uma vez que os fragmentos serão detidos.

9 Source NAT e roteamento

Se você está fazendo SNAT, você vai querer ter certeza de que toda máquina destino dos pacotes SNAT'ed vai responder para a máquina que está fazendo NAT. Por exemplo, se você está mapeando alguns pacotes que estão saindo com o endereço de origem 1.2.3.4, o roteador externo precisa saber que as respostas devem ser enviadas para 1.2.3.4. Isso pode ser feito das seguintes formas:

1. Se você está fazendo SNAT para o próprio endereço da máquina que realiza NAT (para a qual o roteamento e todos os outros serviços já funcionam), nada precisa ser feito.

2. Se você está fazendo SNAT para um endereço inutilizado na sua rede local (por exemplo, você está mapeando para 1.2.3.99, um endereço livre na sua rede 1.2.3.0/24 network), sua máquina responsável pelo NAT terá de responder a requisiçoes ARP para aquele endereço (1.2.3.99) assim como responde os destinados a ela própria. A maneira mais fácil de implementar isso é criando um apelido de IP (alias), conforme esse exemplo:

```
# ip address add 1.2.3.99 dev eth0
```

3. Se você está fazendo SNAT para um endereço completamente diferente, você terá de ter certeza que as máquinas atingidas pelos pacotes SNAT'ed rotearão tal endereço de volta para a máquina responsável por NAT. Isso já está implementado se a NAT box é o gateway padrão da LAN, caso contrário você terá de esclarecer uma rota (caso você esteja utilizando algum protocolo de roteamento) ou adicionar rotas manualmente em cada máquina envolvida.

10 Destination NAT para a mesma rede

Se você está fazendo portforwarding para a mesma rede, você precisa ter certeza que pacotes de requisição e de resposta passarão pela máquina responsável por NAT (podendo então serem alterados). O código NAT vai agora (desde 2.4.0-test6), bloquear redirecionamento de saída ICMP que ocorre quando o pacote NAT'ed sai pela mesma interface pela qual entrou, mas o servidor que recebeu ainda tentará responder diretamente para o cliente (que não vai reconhecer a resposta).

O caso clássico ocorre quando as máquinas internas tentam acessar o web server 'público', que na verdade sofre DNAT do endereço externo (1.2.3.4) para uma máquina interna (192.168.1.1), como descrito abaixo:

```
# iptables -t nat -A PREROUTING -d 1.2.3.4 \
        -p tcp --dport 80 -j DNAT --to 192.168.1.1
```

Uma maneira é rodar um servidor DNS interno que sabe os endereços IP reais (internos) do seu web site público, e repassa todas as outras requisições para um servidor DNS externo. Isso significa que o log no seu web server mostrará os endereços IP internos corretamente.

A outra forma é dizer à máquina responsável por NAT que mapeie o endereço IP de origem para o seu próprio nessas conexões. Nesse exemplo, faríamos o seguinte (assumindo que o endereço interno da NAT box seja 192.168.1.250):

```
# iptables -t nat -A POSTROUTING -d 192.168.1.1 -s 192.168.1.0/24 \
        -p tcp --dport 80 -j SNAT --to 192.168.1.250
```

Devido ao fato de que a chain **PREROUTING** é analizada antes, os pacotes já estarão destinados para o web server interno: podemos dizer quais conexões têm origem interna pelo endereço IP de origem.

11 Agradecimentos

Primeiramente, gostaria de agradecer à WatchGuard, e ao David Bonn, aqueles que acreditaram na idéia do netfilter e me apoiaram enquanto eu o desenvolvia.

E a todos aqueles que me levantaram enquanto eu aprendia sobre a feiura do NAT, especialmente àqueles que leram meu diário.

Rusty

E eu, (Guilherme Ude, tradutor para o Português do Brasil) gostaria de agradecer a todo o time da LinuxPlace, ao pessoal do dicionário Merriam-Webster (http://www.m-w.com) que muito me ajudou nesta tradução e a todas as pessoas que sempre acreditaram em mim. Muito obrigado!

Tradutor para o Português do Brasil: Guilherme Ude Braz (guilherme@linuxplace.com.br).