Linux 2.4 NAT HOWTO

Rusty Russell, mailing list `netfilter@lists.samba.org` v1.0.1 Lunedì 1 Maggio 18:38:22 CST 2000

Questo documento descrive come effettuare il masquerading, il proxy trasparente, il port forwarding, e altre forme di traduzione degli indirizzi di rete (NAT) con i kernel Linux 2.4. Traduzione a cura di Masetti Marco *marcomas@libero.it* `mailto:marcomas@libero.it`

Indice

1 Introduzione

Benvenuto, caro lettore.

Stai per addentrarti nell'affascinante mondo del NAT: Network Address Translation, e questo HOWTO sta per diventare la tua guida accurata al kernel 2.4 di Linux ed altro.

In Linux 2.4 è stata introdotta un'infrastruttura per il mangling (manipolamento) dei pacchetti, chiamata 'netfilter'. Un livello superiore a questo provvede al NAT, completamente reimplementato rispetto ai kernel precedenti.

2 Dove si trovano la pagina Web ufficiale e la lista ?

Ci sono tre siti ufficiali:

- Grazie a *Penguin Computing* `http://netfilter.filewatcher.org/` .
- Grazie a *The Samba Team e SGI* `http://netfilter.samba.org/` .
- Grazie a *Harald Welte* `http://netfilter.gnumonks.org/` .

Per la mailing list ufficiale di netfilter visita

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

2.1 Cos'è il Network Address Translation?

Normalmente i pacchetti nella rete viaggiano dalla sorgente (come il tuo computer di casa) verso la loro destinazione (come ad esempio www.gnumonks.org) passando attraverso diversi collegamenti, nel mio caso circa 19 da dove mi trovo in Australia. Nessuno di questi collegamenti altera in realtà i tuoi pacchetti, non fanno altro che farli proseguire.

Se uno di essi però effettua il NAT, allora sarà in grado di alterare la sorgente o la destinazione del pacchetto così come solo di lasciarlo passare. Come puoi ben immaginare, il sistema non è stato creato per funzionare in questo modo, e quindi NAT è sempre un qualcosa di particolare. In genere i collegamenti che effettuano il NAT ricordano come hanno manipolato il pacchetto, e quindi quando arriva un pacchetto di risposta dall'altra parte, effettuano su di esso la manipolazione inversa, in questo modo tutto funziona.

2.2 Perché dovrei aver bisogno del NAT?

In un mondo perfetto non ne avresti bisogno. Le principali ragioni sono:

Connessioni via modem a internet

La maggior parte degli ISP quando ti colleghi, ti fornisce un singolo indirizzo. Puoi naturalmente inviare pacchetti con qualsiasi indirizzo sorgente, ma i pacchetti di risposta che riceverai saranno solo quelli relativi ai pacchetti inviati con l'indirizzo fornito dal tuo ISP. Se desideri usare più macchine (come ad esempio la tua rete locale di casa) per connetterti a Internet attraverso questo unico collegamento, allora avrai bisogno del NAT.

Questo è oggi l'utilizzo più comune del NAT, noto come 'masquerading' (mascheramento) nel mondo Linux. Io lo chiamo SNAT, in quanto ciò che modifichi è l'indirizzo **sorgente** del primo pacchetto.

Server multipli

Talvolta si vuole cambiare dove si dirigono nella propria rete i pacchetti. Questo perché spesso (come sopra) si ha solo un indirizzo IP, ma si desidera che le persone siano in grado di accedere alle macchine che stanno dietro a quella che possiede l'indirizzo IP 'reale'. Ciò si può fare se si riscrive la destinazione dei pacchetti che arrivano.

Una variazione comune di ciò è il load-sharing, dove un gran numero di pacchetti sono distribuiti su una serie di macchine. Questo tipo di NAT era chiamato nelle versioni precedenti di Linux port-forwarding.

Proxy trasparente

Talvolta si vuole che ogni pacchetto che attraversa la Linux box sia destinato ad un programma della Linux box stessa. Questo è usato per realizzare il proxy trasparente: un proxy è un programma collocato tra la propria rete e il mondo esterno che regola la comunicazione. Si dice trasparente perché la propria rete non saprà mai di dialogare con un proxy, a meno che naturalmente il proxy non funzioni.

Squid può essere configurato per operare in questa maniera, detta redirezione o proxy trasparente nelle versioni precedenti di Linux.

3 I due tipi di NAT

Ho diviso il NAT in due parti: **Source NAT** (SNAT) e **Destination NAT** (DNAT).

Il Source NAT si ha quando si altera l'indirizzo sorgente del primo pacchetto ossia si cambia da dove la connessione proviene. Source NAT è sempre effettuata in fase di post-routing, appena prima che il pacchetto sia immesso nella rete. Il mascheramento (masquerading) è una forma specializzata di SNAT.

Il Destination NAT invece si ha quando si altera l'indirizzo destinazione del primo pacchetto ossia si cambia dove la connessione è diretta. Destination NAT è sempre effettuata in fase di pre-routing, appena il pacchetto arriva dalla rete. Port forwarding, load sharing e il proxy trasparente sono tutte forme di DNAT.

4 Guida rapida alla transizione dai kernel 2.0 e 2.2

Mi dispiace per tutti coloro che sono rimasti traumatizzati dal passaggio dalla 2.0 (ipfwadm) alla 2.2 (ipchains). Ci sono novità buone e cattive.

Prima di tutto, puoi utilizzare ipchains e ipfwadm come prima. E' necessario però usare insmod per caricare i moduli del kernel 'ipchains.o' o 'ipfwadm.o' presenti nell'ultima distribuzione di netfilter. Queste sono mutuamente esclusive, o usi uno o l'altro (sei avvertito), e non dovrebbero essere combinati con nessun modulo di netfilter.

Quando hai installato uno o l'altro di questi moduli, puoi usare ipchains e ipfwadm come al solito, con però le seguenti differenze:

- Impostare i timeout del masquerading con le opzioni -M -S di ipchains, o le opzioni -M -s di ipfwadm non serve più a nulla. Dato che i timeout sono più lunghi nella nuova infrastruttura NAT, non dovrebbero più interessare.
- I campi init_seq, delta e previous_delta nell'elenco verbose del masquerade risulteranno sempre posti a 0.
- Azzerare ed elencare i contatori nello stesso momento con le opzioni '-Z -L' non funzionerà più: i contatori non saranno azzerati.

Gli hacker inoltre noteranno che:

- adesso è possibile collegarsi alle porte 61000-65095 anche se si sta usando il masquerading. Il codice del masquerading assumeva qualsiasi porta di questo intervallo come un facile bersaglio, perciò i programmi non potevano usarle.
- Il non documentato 'getsockname', che i programmi di proxy trasparente potevano usare per ricavare la reale destinazione delle connessioni, ora non funziona più.
- Il non documentato bind-to-foreign-address non è implementato; era usato per completare l'illusione del proxy trasparente.

4.1 Voglio giusto il masquerading! Aiuto!

Questo è ciò che la maggior parte delle persone vuole. Se hai un indirizzo IP dinamico PPP (dialup) (se non lo sai, allora ce l'hai), quello che desideri è dire alla tua box che tutti i pacchetti provenienti dalla tua rete locale devono essere visti come provenienti dalla box collegata in dial-up.

```
# Carica il modulo NAT (questo aggiunge tutti gli altri).
modprobe iptable_nat

# Nella tabella NAT (-t nat), appendi una regola (-A) che dopo il routing
# (POSTROUTING) per tutti i pacchetti uscenti dalla ppp0 (-o ppp0)
# mascheri la connessione (-j MASQUERADE).
iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE

# Attiva l'IP forwarding
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Nota che qui non si sta effettuando alcun filtraggio dei pacchetti, per questo tipo di operazioni leggi il Packet Filtering HOWTO: 'Mescolare NAT e filtraggio dei pacchetti'.

4.2 Che cosa a riguardo di ipmasqadm?

Questo è un bacino di utenza ancora più ristretto, perciò non mi sono preoccupato molto di garantire la compatibilità. Per effettuare il port forwarding si può semplicemente usare 'iptables -t nat'. Per esempio, in Linux 2.2 probabilmente si è fatto uso di:

```
# Linux 2.2
# Fai proseguire i pacchetti TCP diretti alla porta 8080 indirizzo 1.2.3.4
# verso l'indirizzo 192.168.1.1 porta 80
ipmasqadm portfw -a -P tcp -L 1.2.3.4 8080 -R 192.168.1.1 80
```

Ora invece si dovrebbe usare:

```
# Linux 2.4
# Appendi una regola di pre-routing (-A PREROUTING) alla tabella NAT (-t nat)
# in modo che i pacchetti TCP (-p tcp) destinati all'indirizzo 1.2.3.4 (-d
# 1.2.3.4) porta 8080 (--dport 8080) siano diretti (-j DNAT) verso l'indirizzo
# 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 desideri che questa regola alteri anche le connessioni locali (es. anche sulla NAT box stessa, prova a fare un telnet all'indirizzo 1.2.3.4 porta 8080 dovresti ottenere in realtà un telnet all'indirizzo 192.168.1.1 porta 80) puoi inserire la stessa regola nella catena OUTPUT (usata per i pacchetti locali in uscita).

```
# 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 Come impostare il NAT

Devi creare delle regole di NAT che segnalino al kernel quali connessioni cambiare e come cambiarle. Per fare ciò, useremo il programma `iptables` il quale è veramente versatile, e che permette di alterare la tabella NAT specificando l'opzione '-t nat'.

La tabella delle regole NAT contiene tre liste chiamate comunemente 'catene' (chains): ogni regola è esaminata per ordine finché una non è soddisfatta. Le tre catene sono chiamate PREROUTING (per il Destination NAT sui pacchetti in arrivo), POSTROUTING (per il Source NAT sui pacchetti in uscita), e OUTPUT (per il Destination NAT sui pacchetti generati localmente).

Il seguente diagramma, se ho un talento artistico, dovrebbe illustrare tutto ciò molto bene.

```
      _____                                     _____
     /     \                                   /     \
   PREROUTING --->[Decisioni] --------------->POSTROUTING----->
     \D-NAT/     [di routing ]                 \S-NAT/
                [instradamento]                   ^
                       |                         __|__
                       |                         /     \
```

```
|                        | OUTPUT|
|                         \D-NAT/
|                            ^
|                            |
--------> Processi locali ----
```

Per ogni punto, quando un pacchetto passa viene osservata quale connessione vi è associata. Se è una nuova connessione, per sapere cosa fare, osserveremo la catena corrispondente nella tabella NAT. La risposta che si ottiene sarà poi applicata anche a tutti i pacchetti successivi appartenenti a questa connessione.

5.1 Semplici selezioni usando iptables

`iptables` vanta una serie di opzioni standard elencate sotto. Tutte le opzioni con doppio trattino possono essere abbreviate, sempre che `iptables` possa distinguerle dalle altre opzioni disponibili. Se il tuo kernel supporta iptables usando un modulo, dovrai prima di tutto caricare il modulo ip_tables.o utilizzando 'insmod ip_tables'.

L'opzione più importante è l'opzione di selezione della tabella '-t'. Per tutte le operazioni di NAT, dovrai sempre utilizzare '-t nat'. La seconda opzione più importante è '-A' usata per appendere una nuova regola alla fine della catena (es. -A 'POSTROUTING') o '-I' per inserirne una all'inizio (es. '-I PREROUTING').

Puoi specificare la sorgente ('-s' o '–source') e la destinazione ('-d' o '–destination') dei pacchetti su cui vuoi effettuare il NAT. Queste opzioni possono essere seguite da un indirizzo IP singolo (es. 192.168.1.1), da un nome (es. www.gnumonks.org) oppure da un indirizzo di rete (es. 192.168.1.0/24 oppure 192.168.1.0/255.255.255.0).

Puoi inoltre specificare l'interfaccia di ingresso ('-i' o '–in-interface') o l'interfaccia di uscita ('-o' o '–out-interface'), ma quale usare dipende dalla catena in cui vuoi aggiungere la regola, nella catena PREROUTING puoi specificare solo l'interfaccia di ingresso, nella catena POSTROUTING (e OUTPUT) solo quella di uscita. Se usi quella sbagliata `iptables` comunque ti restituirà un errore.

5.2 Note utili sulla selezione dei pacchetti da manipolare

Come detto precedentemente puoi specificare un indirizzo sorgente o di destinazione. Se ometti l'opzione di indirizzo sorgente, allora si intenderà qualsiasi indirizzo sorgente. Se ometti l'opzione di indirizzo destinazione, allora si intenderà qualsiasi indirizzo destinazione.

Puoi anche indicare uno specifico protocollo ('-p' o '–protocol'), come ad esempio TCP o UDP, in questo modo solo i pacchetti con quel protocollo saranno sottoposti alla regola. La ragione principale per cui è utile specificare il protocollo è che tcp e udp consentono di utilizzare delle opzioni extra, precisamente le opzioni '–source-port' e '–destination-port' (abbreviate '–sport' e '–dport').

Queste opzioni ti permettono di specificare che solo i pacchetti con quella certa porta sorgente e destinazione devono essere sottoposti alla regola. Questo è utile per le richieste di redirezione del web (TCP porta 80 o 8080) e per lasciar stare gli altri pacchetti.

Queste opzioni devono essere inserite dopo l'opzione '-p' (che ha come effetto quello di caricare la libreria estensione del protocollo). Per le porte puoi utilizzare i numeri corrispondenti, o i nomi presenti nel file /etc/services.

Tutte le differenti caratteristiche utilizzabili per selezionare i pacchetti sono dettagliate in modo esauriente nelle pagine del manuale (`man iptables`).

6 Spiega come manipolare i pacchetti

Ora conosci come selezionare i pacchetti che vuoi manipolare. Per completare la tua regola, dobbiamo dire al kernel con precisione che cosa vogliamo fare a questi pacchetti.

6.1 Source NAT

Se vuoi effettuare il Source NAT allora devi cambiare l'indirizzo sorgente della connessione con qualcosa di differente. Questo però deve essere fatto nella catena POSTROUTING, appena prima che il pacchetto sia inviato. Questo è un dettaglio importante, perché solo così qualsiasi altra cosa nella Linux box (instradamento, filtraggio dei pacchetti) vedrà il pacchetto come invariato. Ciò significa inoltre che si potrà utilizzare l'opzione '-o' (interfaccia uscente).

Il Source NAT si specifica usando '-j SNAT', l'opzione '–to-source' permette di indicare un indirizzo o un intervallo di indirizzi IP e opzionalmente una o un intervallo di porte (però solo con i protocolli UDP e TCP).

```
## Cambia l'indirizzo sorgente in 1.2.3.4.
# iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 1.2.3.4

## Cambia l'indirizzo sorgente in 1.2.3.4, 1.2.3.5 oppure 1.2.3.6
# iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 1.2.3.4-1.2.3.6

## Cambia l'indirizzo sorgente in 1.2.3.4, porte 1-1023
# iptables -t nat -A POSTROUTING -p tcp -o eth0 -j SNAT --to 1.2.3.4:1-1023
```

6.1.1 Masquerading

Esiste un caso specializzato di Source NAT denominato mascheramento (masquerading): dovrebbe essere utilizzato solo se gli indirizzi IP sono assegnati dinamicamente, come ad esempio nel caso di connessione via modem (dial up), nel caso di indirizzi IP statici invece si usi il già citato SNAT.

Non è necessario con il mascheramento (masquerading) indicare esplicitamente l'indirizzo sorgente in quanto sarà utilizzato l'indirizzo dell'interfaccia da cui il pacchetto uscirà. Ancora più importante è il fatto che se il collegamento dovesse interrompersi, la connessione sarà dimenticata (sarebbe comunque persa), in questo modo non ci saranno grossi problemi quando la connessione sarà ristabilita, naturalmente con un nuovo indirizzo IP.

```
## Maschera qualsiasi cosa esca da ppp0.
# iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
```

6.2 Destination NAT

Questo avviene nella catena PREROUTING, appena il pacchetto arriva; ciò significa che qualsiasi cosa nella Linux box (instradamento, filtraggio dei pacchetti) vedrà il pacchetto arrivato con il suo indirizzo di destinazione 'reale'. Ciò permette di utilizzare l'opzione '-i' (interfaccia ingresso).

Per alterare la destinazione dei pacchetti generati localmente si può usare la catena OUTPUT, ma questo è inusuale.

Destination NAT si specifica usando '-j DNAT', l'opzione '–to-destination' permette di specificare un indirizzo o un intervallo di indirizzi IP e opzionalmente una o un intervallo di porte (solo per i protocolli UDP e TCP).

```
## Cambia l'indirizzo di destinazione in 5.6.7.8
# iptables -t nat -A PREROUTING -i eth1 -j DNAT --to 5.6.7.8

## Cambia l'indirizzo di destinazione in 5.6.7.8, 5.6.7.9 oppure 5.6.7.10.
# iptables -t nat -A PREROUTING -i eth1 -j DNAT --to 5.6.7.8-5.6.7.10

## Cambia l'indirizzo di destinazione del traffico web in 5.6.7.8, porta 8080.
# iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth1 \
        -j DNAT --to 5.6.7.8:8080

## Redireziona i pacchetti locali diretti all'indirizzo 1.2.3.4 verso loopback.
# iptables -t nat -A OUTPUT -d 1.2.3.4 -j DNAT --to 127.0.0.1
```

6.2.1 Redirezione

Esiste un caso specializzato di Destination NAT chiamato redirection (redirezione), ed è semplicemente una comodità in più in quanto esattamente uguale al DNAT effettuato però esclusivamente sull'indirizzo associato all'interfaccia di ingresso.

```
## Invia i pacchetti diretti alla porta 80 verso il tuo proxy (trasparente)
## squid
# iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 80 \
        -j REDIRECT --to-port 3128
```

6.3 Mapping in dettaglio

Ci sono alcune sottigliezze nel NAT con cui la maggior parte delle persone non avrà mai a che fare. Queste sono documentate qui di seguito per i curiosi.

6.3.1 Selezione di indirizzi multipli in un intervallo

Se è dato un intervallo di indirizzi IP, l'indirizzo IP da usare è scelto in base all'IP correntemente meno utilizzato per le connessioni conosciute dalla macchina. Questo garantisce il load-balancing primitivo.

6.3.2 Creare mapping NAT nullo

Si può usare l'obiettivo '-j ACCEPT' per lasciare che la connessione prosegua senza che avvenga alcuna operazione di NAT.

6.3.3 Comportamento Standard del NAT

Il comportamento standard è di alterare la connessione il meno possibile entro i vincoli delle regole date dall'utente. Questo significa che le porte non saranno rimappate a meno che non ce ne sia bisogno.

6.3.4 Mapping implicito delle porte sorgenti

Anche quando il NAT non è richiesto per una connessione, la traduzione della porta sorgente può avvenire implicitamente se un'altra connessione è stata mappata sopra la nuova. Considera il caso di mascheramento, che è comune:

1. Una connessione è stabilita da una box 192.1.1.1 dalla porta 1024 verso www.netscape.com porta 80
2. Questa è mascherata dalla masquerading box per usare il suo indirizzo IP sorgente (1.2.3.4).
3. Il masquerading box prova a effettuare una connessione web con www.netscape.com sulla porta 80 dall'indirizzo 1.2.3.4 (indirizzo dell'interfaccia esterna) porta 1024.
4. Il codice NAT altera l'indirizzo sorgente della seconda connessione alla porta 1025, così le due non si intralciano.

Quando si ha questo mapping implicito della sorgente, le porte sono divise in tre classi:

- Porte sotto la 512
- Porte tra la 512 e la 1023
- Porte 1024 e successive.

Una porta non sarà mai mappata implicitamente in una classe differente.

6.3.5 Cosa succede quando il NAT fallisce

Se non esiste un modo per mappare unicamente una connessione come richiede l'utente, allora sarà rifiutata. Questo accade anche ai pacchetti che non possono essere classificati come appartenenti ad una qualsiasi connessione, in quanto malformati o perché la box ha esaurito la memoria, ecc.

6.3.6 Mapping multipli, sovrapposizioni, conflitti

Puoi avere regole NAT che mappano i pacchetti sullo stesso intervallo; il codice del NAT è sufficientemente abile ad evitare conflitti. Perciò pur avendo due regole che mappano gli indirizzi sorgente 192.168.1.1 e 192.168.1.2 su 1.2.3.4 tutto funziona correttamente.

In aggiunta, puoi mappare gli indirizzi IP reali usati, sempre che questi indirizzi attraversino la box che effettua il mapping. Quindi se hai una rete assegnata (1.2.3.0/24), e una rete interna che usa questi indirizzi e un'altra ancora che usa indirizzi internet privati del tipo 192.168.1.0/24, puoi effettuare tranquillamente il NAT dei pacchetti con indirizzo sorgente 192.168.1.0/24 sulla rete 1.2.3.0, senza aver paura di eventuali sovrapposizioni.

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

La stessa logica si può applicare agli indirizzi usati dalla NAT box stessa: è così che funziona il masquerading (condividendo l'indirizzo dell'interfaccia tra pacchetti mascherati e pacchetti reali provenienti dalla box stessa).

Inoltre, puoi mappare gli stessi pacchetti su differenti obiettivi, ed essi saranno suddivisi. Per esempio, se non vuoi mappare nulla sopra 1.2.3.5, allora puoi usare:

```
# 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 Alterare la destinazione delle connessioni generate localmente

Se la destinazione di un pacchetto generato localmente viene cambiata (es. dalla catena OUTPUT), e questo causa il passaggio del pacchetto attraverso una diversa interfaccia, allora anche l'indirizzo sorgente sarà modificato con quello dell'interfaccia. Per esempio, cambiare la destinazione di un pacchetto di loopback che passa per eth0 risulterà nella sostituzione dell'indirizzo sorgente 127.0.0.1 con l'indirizzo dell'interfaccia eth0; diversamente da altri mapping sulla sorgente questo avviene immediatamente. Naturalmente entrambi questi mapping sono invertiti nei pacchetti di risposta che arrivano.

7 Protocolli speciali

Alcuni protocolli non amano subire il NAT. Per ciascuno di questi protocolli bisognerebbe scrivere due estensioni, una per il connection tracking e una per il NAT.

Nella distribuzione di netfilter ci sono al momento dei moduli per ftp: ip_conntrack_ftp.o e ip_nat_ftp.o. Se carichi questi moduli (o se sono compilati nel kernel), ogni tipo di NAT effettuato sulle connessioni ftp dovrebbe funzionare. Altrimenti, potrai utilizzare solo ftp passivo, e perfino questo potrebbe non funzionare correttamente se si sta facendo qualcosa di più del semplice Source NAT.

8 Avvertimenti riguardo il NAT

Se stai effettuando il NAT su una connessione, tutti i pacchetti che transitano in **entrambe** le direzioni (dentro e fuori la rete) devono passare attraverso la NAT box, altrimenti il NAT non funzionerà correttamente. In particolare il codice di tracciamento della connessione si occupa di riassemblare i frammenti, quindi non solo il tracciamento potrebbe non essere affidabile ma anche i pacchetti potrebbero non passare del tutto, i frammenti infatti potrebbero essere rifiutati.

9 Ringraziamenti

Grazie prima di tutto alla WatchGuard, e a David Bonn, che hanno creduto nell'idea di netfilter così tanto da darmi il loro supporto mentre ci lavoravo.

E a tutti coloro che hanno sopportato i miei sproloqui, affinché potessi apprendere le bruttezze del NAT, e specialmente infine quelli che hanno letto il mio diario.

Rusty.