Ethernet Bridge + netfilter Howto

Nils Radtke

v0.8, luglio 2005

Installare un ethernet bridge offre la possibilità di integrare in una rete già esistente uno strumento di controllo e di regolazione del traffico in maniera del tutto trasparente. Questa installazione non richiede cambiamenti alla topologia logica della rete e viene realizzata collegando il bridge ethernet nella topologia fisica, tra la rete stessa e l'istanza di instradamento (quella parte dell'hardware connessa a Internet). Traduzione a cura di Ginox (ginox@toglimi.autistici.org) e Michele Ferritto (m.ferritto@toglimi.virgilio.it), revisione a cura di Daniele A.

Contents

1 Introduzione 2

2 Software necessario 2

- 2.1 Caratteristiche del kernel Linux . . . 3
- 2.2 Strumento spazio utente: `brctl` . . . 5
- 2.3 Note sul Kernel . . . 6

3 Configurare Linux 6

- 3.1 Configurare il bridge . . . 6
- 3.2 Configurare l'instradamento . . . 7
- 3.3 Facciamolo succedere di nuovo! . . . 8

4 Provare il nuovo ambiente bridge! 9

- 4.1 Terreno di prova . . . 9
- 4.2 Pingalo Jim! . . . 10
- 4.3 Configurazione Attuale . . . 12
  - 4.3.1 Configurazione dell'interfaccia . . . 12
  - 4.3.2 Configurazione del Routing . . . 13
  - 4.3.3 Configurazione di Iptables . . . 13
- 4.4 Nota finale (Importante!) . . . 13
- 4.5 Bug-Notes . . . 14

5 Esperienze degli utenti 14

- 5.1 Fedora Core 3 . . . 14

6 Links 15

- 6.1 Ethernet-Bridge . . . 15
- 6.2 Argomenti correlati: . . . 15

Questo Howto è disponibile in diversi formati . Preferibilmente scaricabile come: tarball . Si può reperire l'Howto presso il Linux Documentation Project .

Si sta cercando una traduzione in un'altra lingua? Si veda la versione tedesca . Da Marzo 2003 è c'è una versione francese di questo HOWTO, disponibile grazie a Francois Romieu e Guillaume Lelarge: versione francese . Ultimamente, a luglio 2005, sono inciampato su di una versione italiana di questo HOWTO!

2004-06-13:

Aggiornamento sulle versioni del kernel utilizzate, avvertimento sulla 3.1 (amministrazione remota), codice per 2.1 (netfilter debugging). Nuova traduzione disponibile: Francese (si veda il link sopra). Abbandonata la versione tedesca.

2002-09-19:

Aggiornati i link relativi ad ebtables nella sezione "Argomenti Correlati". Aggiunta una nota sul 4.5 ("falso positivo" debugging output di br-nf).

2002-10-08:

Aggiunta la sezione 4.3 (Configurazione attuale) e suggerimenti sull'instradamento in 3.2 (Configurare l'instradamento), 4.2 (Pingalo Jim!)

1 Introduzione

I bridge ethernet uniscono due o più differenti segmenti ethernet in maniera trasparente.
Un bridge ethernet smista i pacchetti in ingresso sulle porte associate con l'interfaccia del bridge. Ciò viene realizzato in maniera ponderata: quando il bridge sa a quale porta corrisponde il MAC address del destinatario del pacchetto, questo viene instradato soltanto su quell'interfaccia invece di inquinare tutte le porte.

Le interfacce ethernet possono essere aggiunte ad un'interfaccia bridge già esistente e divenire quindi porte (logiche) di quest'ultima.

Aggiungere una struttura netfilter su di un'interfaccia bridge fornisce al sistema un meccanismo di filtraggio creato in modo trasparente che, inoltre, non necessita di un indirizzo IP per funzionare. In ogni caso è possibile assegnare un indirizzo IP all'interfaccia bridge per scopi amministrativi (chiaramente soltanto attraverso ssh ;-).

I vantaggi di questo sistema sono evidenti. La trasparenza evita all'amministratore la pena di ristrutturare la topologia della rete. Gli utenti non noteranno l'esistenza del bridge anche se le loro connessioni verranno bloccate; inoltre non saranno disturbati mentre si effettua l'installazione (si consideri un'azienda dove la perdita della connettività si traduce in un alto costo).

Un altro caso comune è quello del cliente connesso alla rete globale tramite un router concesso in uso. Poiché i provider raramente concedono privilegi di amministrazione sul loro hardware, il cliente non può cambiare la configurazione di connessione. Disponendo di una rete funzionante e nell'intento di spendere meno possibile, non è sua intenzione riconfigurarla interamente. Utilizzando un dispositivo di bridging non è costretto a farlo.

2 Software necessario

Questa è la configurazione software necessaria sul computer che farà da bridge, secondo quanto descritto nella sezione 4 (Terreno di prova).

2.1 Caratteristiche del kernel Linux

L'uso del kernel 2.6 non è ancora una buona idea. Sì, è strano. Il perchè e dove il codice di bridging non funzioni non è ancora saltato fuori. Non è raccomandato utilizzare questa serie. Si ha la soluzione? Si sia certi di poterlo dimostrare e la si invii per e-mail a me (l'indirizzo è nella pagina introduttiva). Si vedano anche le 2.3 (Note sul Kernel) per le informazioni aggiuntive. Intanto si usino kernel della serie 2.4.

A partire dalla versione *2.4.18* del kernel, il supporto per l'Ethernet Bridge è già presente. Non è necessaria alcuna patch. A proposito delle versioni più recenti del kernel, bisogna precisare che la versione *2.4.23* è la meno raccomandabile, in particolar modo se usata insieme a ebtables e netfilter-bridging. Sono raccomandate versioni più recenti.
Il paragrafo che segue è sorpassato (12-07-2005) poichè tutto quello di cui si necessita è disponibile nel kernel. Lo si può tranquillamente saltare, viene mantenuto solo per ragioni storiche:
Se si ha intenzione di usare le risorse netfilter, perché si vuole eseguire iptables sul nuovo router/firewall Linux, si rende necessario applicare una patch. Le patch possono essere scaricate dalla 6.1 (homepage del progetto Ethernet Bridge su sourceforge).

```
root@bridge:~> cd /usr/src/
root@bridge:~> wget -c http://bridge.sourceforge.net/devel/bridge-nf/bridge-nf-0.0.7-against-2.4.18.diff
root@bridge:~> cd /usr/src/linux/
root@bridge:~> patch -p1 -i ../bridge-nf/bridge-nf-0.0.7-against-2.4.18.diff
```

Supponendo di voler utilizzare netfilter sull'interfaccia bridge e di aver già applicato la patch al kernel, si dovranno ora attivare alcune opzioni di configurazione di quest'ultimo. Per i dettagli su come realizzare la propria immagine del kernel si veda CD-Net-Install-HOWTO, Toolbox . Ah si, è ancora solo in tedesco. Ehm, dovrei metterlo a posto un giorno di questi, ma mi manca il tempo... Qualche volontario? (questo mortale silenzio è assordante.. ;)

Per ora, in ogni caso: In

```
Code maturity level options
```

si attivi

```
[*] Prompt for development and/or incomplete code/drivers
```

e in

```
Loadable module support
```

```
[*] Enable loadable module support
[*]   Set version information on all module symbols
[*]   Kernel module loader
```

Ok, per ora tutto bene. Quindi in

```
Networking options
```

si abiliterà

```
[*] Network packet filtering (replaces ipchains)
[ ]   Network packet filtering debugging
```

Nota:

In precedenza, la suindicata opzione di debugging era selezionata. Per ora, a meno che non si voglia che la propria partizione `/var/log/` venga riempita in breve tempo, la si disattivi.
Se viene attivata, nel dmesg appariranno centinaia di messaggi in `/var/log/`{`kern.log,debug,syslog,messages`}simili al seguente:

```
skb: pf=2 (unowned) dev=br0 len=52
PROTO=6 156.136.32.121:3709 192.168.101.2:112 L=52 S=0x00 I=35470 F=0x4000 T=51
nf_hook: hook 1 already set.
skb: pf=2 (unowned) dev=br0 len=52
PROTO=6 156.136.32.121:3709 192.168.101.2:112 L=52 S=0x00 I=35470 F=0x4000 T=51
nf_hook: hook 0 already set.
skb: pf=2 (unowned) dev=br0 len=52
PROTO=6 192.168.101.11:2828 192.168.101.2:202 L=52 S=0x10 I=63 F=0x4000 T=64
nf_hook: hook 1 already set.
skb: pf=2 (unowned) dev=br0 len=52
PROTO=6 192.168.101.11:2828 192.168.101.2:202 L=52 S=0x10 I=63 F=0x4000 T=64
nf_hook: hook 3 already set.
skb: pf=7 (owned) dev=eth1 len=1500
```

Poi in

```
  IP: Netfilter Configuration  --->
```

si selezionerà tutto quello di cui si necessita come modulo. Ed ora il campo tanto atteso: attivare

```
<M> 802.1d Ethernet Bridging
```

insieme a

```
[*]   netfilter (firewalling) support
```

Nota:

La precedente opzione sarà presente solo se è stata applicata con successo la patch al kernel!

Infine si ha semplicemente bisogno di eseguire:

```
root@bridge:~> make dep clean bzImage modules modules_install
```

se il comando và a buon fine, tutto è a posto. Non ci si dimentichi di cambiare `/etc/lilo.conf` e di rilanciare

```
root@bridge:~> lilo -t
root@bridge:~> lilo
root@bridge:~> reboot
```

Suggerimento:

Si vuole evidenziare il nuovo kernel come bridge kernel? Si apra in un editor il Makefile che si trova nella directory dei sorgenti del kernel e si modifichi la linea di intestazione chiamata EXTRAVERSION =. Si potrebbe modificarla in, magari, bridge? ;-) Dopo il modules_install i nuovi moduli saranno in /lib/modules/2.4.18bridge
Per gli utenti debian (si usi eventualmente prima `export PATCH_THE_KERNEL=YES` e –added_patches your_patches con make-kpkg):

```
root@bridge:~> make-kpkg --revision=tf.1.0 kernel_image
```

2.2 Strumento spazio utente: `brctl`

Ora che il kernel è configurato con il supporto per l'Ethernet Bridge e netfilter, si prepara lo strumento spazio utente `brctl`. Questo è lo strumento da utilizzare per 3 (configurare) quanto necesario.

Si scarichi 6.1 (il tarball del sorgente), si decomprima e ci si posizioni nella directory appropriata.

```
root@bridge:~> wget -c http://bridge.sourceforge.net/bridge-utils/bridge-utils-0.9.5.tar.gz
root@bridge:~> tar xvzf bridge-utils-0.9.5.tar.gz
root@bridge:~> cd bridge-utils-0.9.5
```

Si consultino il file `README` e quanto presente nella sottodirectory `doc/`. Quindi si compili con un semplice make e si copi il risultante eseguibile `brctl/brctl` in `/sbin/`.

```
root@bridge:~> make
root@bridge:~> cp -vi brctl/brctl /sbin/
```

Questo è tutto. Si può procedere con la 3 (Configurazione.)

2.3 Note sul Kernel

Sintomo: Durante la configurazione funziona tutto, ma i pacchetti non attraversano più le interfacce del bridge come succedeva con il kernel 2.4.
ipuk s (qasuari_ @ _yahoo.com) scrive (più o meno a giugno 2005):

```
[...]
Compilo il mio kernel dal 2.4.18-14 al 2.6.0 e attivo
bridge-netfilter&ebtables.
Dopo la compilazione, non riesco a pingare da un host verso l'interfaccia della linux box.
La linux box ha semplicemente un interfaccia.Cosa ho sbagliato nella compilazione???
[...]
```

3 Configurare Linux

3.1 Configurare il bridge

È necessario che il sistema riconosca il bridge. A questo proposito, prima di tutto deve essere indicato che si vuole un'interfaccia bridge Si veda 4 (Terreno di prova)).

```
root@bridge:~> brctl addbr br0
```

Secondo, non si ha bisogno di STP (Spanning Tree Protocol). Nell'esempio proposto c'è un unico router, quindi un loop è molto improbabile. Si può quindi disabilitare questa funzionalità. (La rete risulterà anche meno congestionata):

```
root@bridge:~> brctl stp br0 off
```

Dopo questi preparativi, si possono finalmente eseguire i comandi effettivi. Si aggiungeranno le due (o più) interfacce fisiche, ciò significa che verranno associate all'interfaccia logica (virtuale) del bridge br0 appena creata.

```
root@bridge:~> brctl addif br0 eth0
root@bridge:~> brctl addif br0 eth1
```

Nota Importante:

> Molte persone mi hanno inviato email dicendo che li avrebbe aiutati parecchio se fossi stato più esaustivo e chiaro sul rischio di essere tagliati fuori. Per cui a questo punto si faccia attenzione ai miei ammonimenti:
> Se si sta leggendo questo, si è ad un (piccolo) passo dall'essere _isolati__ dalla macchina che si sta

> trasformando in un bridging device.
> Se vi piace vivere al limite del dissanguamento è il momento giusto per tirare fuori la cassetta del pronto soccorso. Ne avrete veramente bisogno.
> Se non avete l'accesso fisico e nessuno alla vostra portata lo ha:
> NON PROCEDERE A MENO CHE LE VOSTRE DITA NON ABBIANO LASCIATO LA TASTIERA CHE AVETE DI FRONTE E I VOSTRI OCCHI NON SIANO CONCENTRATI SU ALTRO CHE NON SIA LA CONSOLE SU CUI STATE LAVORANDO.
> Ora siete stati avvisati. Non mi assumo nessuna responsabilità.

Ora, le due interfacce fisiche sono diventate un'unica porta logica del bridge. Ehm, ok, avevamo ed avremo sempre due interfacce fisiche. Sono sempre lì, controllate ;-) Ma ora sono parte di un dispositivo bridge logico e quindi non c'è bisogno di alcuna assegnazione di IP. Dunque si possono rilasciare gli indirizzi:

```
root@bridge:~> ifconfig eth0 down
root@bridge:~> ifconfig eth1 down
root@bridge:~> ifconfig eth0 0.0.0.0 up
root@bridge:~> ifconfig eth1 0.0.0.0 up
```

Grande! Ora abbiamo una linux box senza IP. Dunque se hai intenzione di configurare il tuo futuro firewall/router via TP, usa la tua console locale ;-)) Ne possiedi una seriale ? Buon per te :-)

Optional:

Si può associare alla nuova interfaccia logica un IP in questo modo:

```
root@bridge:~> ifconfig br0 10.0.3.129 up
```

Si confronti la sezione 4.2 (Note Importanti)!

3.2 Configurare l'instradamento

Nel caso in cui si intenda configurare un gateway si dovrà abilitare l'inoltro (forwarding) nel kernel.

```
root@bridge:~> echo "1" > /proc/sys/net/ipv4/ip_forward
```

La macchina possiede già un IP, ma non una route di default. Si imposta con

```
root@bridge:~> route add default gw 10.0.3.129
```

Alla fine si dovrebbe avere una rete funzionante da, verso e attraverso il gateway.

3.3 Facciamolo succedere di nuovo!

Meglio detto come: dobbiamo rendere le modifche permanenti al riavvio.
Per ottenere questo, è necessario qualche tipo di script di shell messo nell'appropriata directory di boot-up: `/etc/init.d/`
In seconda battuta, bisogna creare un link nella propria directory di runlevel. La directory giusta dipende dal proprio gusto e ovviamente dalla distribuzione Linux in uso. I valori di runlevel più comuni sulle workstation, sono: `2`, `3` e `5`. Degli esempi sono: `/etc/rc?.d/` (sostuire il ? con il giusto runlevel)
Inoltre, bisogna farsi un idea di quando le proprie interfacce di rete devono essere attivate. Per ora si assume che le interfacce sono attivate alla priorità di sistema `S` per cui non dobbiamo tenerne conto. Se si ha la necessità di conoscere con esattezza il momento, si guardi in `/etc/rcS.d/`. Si vuole semplicemente che il bridge sia attivo il prima possibile per cui si imposta la priorità a `10`. (Ci si assicuri che nessun servizio che richiede i dispositivi di bridging parta primai, che abbia cioè valore di priorità inferiore a `10`)
Per ora si assume che il proprio runlevel è `5`:

```
root@bridge:~> mv -i bridge.sh /etc/init.d/
root@bridge:~> cd /etc/rc5.d/
root@bridge:~> ln -s ../init.d/bridge.sh S10bridge.sh
```

Di solito, tutte le distribuzioni forniscono qualche tipo di runlevel-checker o strumento equivalente che assiste nel noioso lavoro di amministrare i link di runlevel. Si consulti la documentazione della propria distribuzione su questo.
Suggerimento: debian ha update-rc.d, redhat e successori hanno chkconfig. Infine, anche SuSE ha il suo (il nome del quale non riesco a ricordare facilmente..).
Curiosità sul contenuto di bridge.sh? ;-)

```
#!/bin/bash
PATH="/sbin:/usr/sbin:/usr/local/sbin";
slaveIfs="1 2 3 4 6 7 8 9 10";
cmd="$1";
[ -z "$cmd" ] && cmd="start";
case "$cmd" in
  start)
    brctl addbr br0;
    brctl stp br0 on;
    brctl addif br0 eth0;
    brctl addif br0 eth1;
    (ifdown eth0 1>/dev/null 2>&1;);
    (ifdown eth1 1>/dev/null 2>&1;);
    ifconfig eth0 0.0.0.0 up;
    ifconfig eth1 0.0.0.0 up;
    ifconfig br0 10.0.3.129 broadcast 10.0.3.255 netmask 255.255.255.0 up ### Adapt to your needs.
    route add default gw 10.0.3.129; ### Adapt to your needs.
    for file in br0 eth0 eth1;
    do
      echo "1" > /proc/sys/net/ipv4/conf/${file}/proxy_arp;
```

```
      echo "1" > /proc/sys/net/ipv4/conf/${file}/forwarding;
    done;
    echo "1" > /proc/sys/net/ipv4/ip_forward;
    ;;
  stop)
    brctl delif br0 eth0;
    brctl delif br0 eth1;
    ifconfig br0 down;
    brctl delbr br0;
    #ifup eth0; ### Adapt to your needs.
    #ifup eth1; ### Adapt to your needs.
    ;;
  restart,reload)
    $0 stop;
    sleep 3;
    $0 start;
    ;;
esac;
```

E si, lo si renda eseguibile..

```
root@bridge:~> chmod 700 /etc/init.d/bridge.sh
```

Dopo di questo, ci si assicuri che il bridge sopravviva a riavvii inattesi. È come per i backup: bisogna provarli prima che diventino necessari.

4 Provare il nuovo ambiente bridge!

4.1 Terreno di prova

Si immagini uno scenario simile a questo:

```
                                                          /\
          Ethernet           Ethernet           ATM    /-/  \
---------          ---------          ---------     /-/      |
|  Box  |----------|Bridge |----------|Router |-----| Inter-  \
---------          ---------          ---------     \  net  ---|
         ^        ^         ^        ^               \     /
         |        |         |        |                \---/
        eth0     eth0      eth1     if0                 ^
         |        |         |        |                  |
      10.0.3.2   none/10.0.3.1      195.137.15.7    anything else
                  \         /
                   \       /
```

```
   ^               \-br0-/
   |                                    ^              ^
   |                  ^                 |              |
   |                  |                 |              |
  own                own             foreign        hostile
```

Il proprio potere amministrativo si limita alle macchine contrassegnate con `own`, il Router è completamente off-limits e così pure Internet, naturalmente.
Se si vuole controllare il traffico da e per la rete si può decidere di integrare un firewall nel bridge.
Il modo usuale di risolvere il problema consisterebbe nel cambiare il default gateway su tutti gli host della propria rete. Ma questo è un sistema piuttosto oneroso, nessuno vorrebbe cambiare più di 5 default route su 5 diversi host per più di una volta. Si tenga anche presente il dispendio di tempo. E non si dimentichi che questa soluzione potrebbe porre problematiche per quanto riguarda la sicurezza.
L'altro sistema è più semplice, richiede meno tempo, è più sicuro e meno soggetto a errori. Più sicuro poiché non vi è necessità di assegnare alcun indirizzo IP. Niente IP, niente pericoli. Almeno in teoria, si spera, gli stack sono salvi. (In ogni caso sarebbe meglio non farci affidamento...). Il vantaggio quindi sta nell'avere un'installazione completamente trasparente, nessun IP e nessun MAC da cambiare.
Ognuno sceglie il metodo che preferisce, qui verrà trattato solo il più elegante ;-)

4.2 Pingalo Jim!

Si configurerà l'interfaccia eth0 della propria macchina come d'abitudine. Le interfacce del bridge saranno configurate come descritto nella sezione 3 (Configurazione).
Se si desidera utilizzare l'inoltro (forwarding) si dovrà eseguire:

```
root@bridge:~> echo "1" > /proc/sys/net/ipv4/ip_forward
```

Opzionalmente si potrà configurare una default route:

```
root@bridge:~> route add default gw 10.0.3.129
```

Quindi si imposteranno alcune regole di iptables sull' host `bridge`:

```
root@bridge:~> iptables -P FORWARD DROP
root@bridge:~> iptables -F FORWARD
root@bridge:~> iptables -I FORWARD -j ACCEPT
root@bridge:~> iptables -I FORWARD -j LOG
root@bridge:~> iptables -I FORWARD -j DROP
root@bridge:~> iptables -A FORWARD -j DROP
root@bridge:~> iptables -x -v --line-numbers -L FORWARD
```

L'ultima linea genera il seguente output:

```
Chain FORWARD (policy DROP 0 packets, 0 bytes)
num      pkts      bytes target   prot opt in     out     source   destination
1           0         0 DROP       all  --  any    any     anywhere anywhere
2           0         0 LOG        all  --  any    any     anywhere anywhere      LOG level warning
3           0         0 ACCEPT     all  --  any    any     anywhere anywhere
4           0         0 DROP       all  --  any    any     anywhere anywhere
```

Il `LOG` target inserisce fa il log di tutti i pacchetti attraverso `syslogd`. Attenzione però, questo è da intendersi a solo scopo di testing, in un ambiente di produzione questa regola deve essere rimossa. Altrimenti si rischia di riempire i file di log e l'hard disk, oppure chiunque potrebbe usare questo sistema per un causare un Denial of Service.
Per provare le regole si esegua un ping verso il router (195.137.15.7) sull'host `box`:

```
root@box:~> ping -c 3 195.137.15.7
PING router.provider.net (195.137.15.7) from 10.0.3.2 : 56(84) bytes of data.
--- router.provider.net ping statistics ---
3 packets transmitted, 0 received, 100% loss, time 2020ms
^C
root@box:~>
```

Di default viene eseguito un `DROP` su ogni pacchetto. Nessuna risposta, nessun pacchetto nei log. Questa configurazione è progettata per scartare ogni pacchetto a meno che non si ometta la prima regola, subito prima di quella con il target `LOG`:

```
root@bridge:~> iptables -D FORWARD 1
root@bridge:~> iptables -x -v --line-numbers -L FORWARD
```

Ora le regole diventano:

```
Chain FORWARD (policy DROP 0 packets, 0 bytes)
num      pkts      bytes target   prot opt in     out     source   destination
2           0         0 LOG        all  --  any    any     anywhere anywhere      LOG level warning
3           0         0 ACCEPT     all  --  any    any     anywhere anywhere
4           0         0 DROP       all  --  any    any     anywhere anywhere
```

E tutti i pacchetti possono attraversare il bridge. Proviamo con un ping dall'host `box`:

```
root@box:~> ping -c 3 195.137.15.7
PING router.provider.net (195.137.15.7) from 10.0.3.2 : 56(84) bytes of data.
64 bytes from router.provider.net (195.137.15.7): icmp_seq=1 ttl=255 time=0.103 ms
64 bytes from router.provider.net (195.137.15.7): icmp_seq=2 ttl=255 time=0.082 ms
64 bytes from router.provider.net (195.137.15.7): icmp_seq=3 ttl=255 time=0.083 ms
```

```
--- router.provider.net ping statistics ---
3 packets transmitted, 3 received, 0% loss, time 2002ms
rtt min/avg/max/mdev = 0.082/0.089/0.103/0.012 ms
root@box:~>
```

Yippeah! il router è vivo e funzionante.

Nota Importante:

Quando viene attivata, l'interfaccia del bridge impiega circa 30 secondi per diventare completamente operativa. Questi 30 secondi sono la fase di apprendimento del bridge. Durante questo tempo il bridge esegue un test per stabilire quali indirizzi MAC esistono su quali porte. L'autore del codice, Lennert, dichiara nel suo TODO che la durata della fase di apprendimento verrà opportunatamente diminuita prima o poi.
Si ricordi che durante la fase di test, nessun pacchetto attraverserà il bridge e nessun ping sarà possibile.

4.3 Configurazione Attuale

Questa sezione vuole offrire alcuni suggerimenti relativi a come dovrebbe presentarsi il sistema dopo aver eseguito con successo quanto descritto in questo howto.

4.3.1 Configurazione dell'interfaccia

L'output di `ifconfig` dovrebbe essere simile al seguente:

```
root@bridge:~> ifconfig
br0       Link encap:Ethernet  HWaddr 00:04:75:81:D2:1D
          inet addr:10.0.3.129  Bcast:195.30.198.255  Mask:255.255.255.128
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:826 errors:0 dropped:0 overruns:0 frame:0
          TX packets:737 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:161180 (157.4 Kb)  TX bytes:66708 (65.1 Kb)

eth0      Link encap:Ethernet  HWaddr 00:04:75:81:ED:B7
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5729 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3115 errors:0 dropped:0 overruns:0 carrier:656
          collisions:0 txqueuelen:100
          RX bytes:1922290 (1.8 Mb)  TX bytes:298837 (291.8 Kb)
          Interrupt:11 Base address:0xe400

eth1      Link encap:Ethernet  HWaddr 00:04:75:81:D2:1D
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:1 frame:0
```

```
          TX packets:243 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:342 (342.0 b)  TX bytes:48379 (47.2 Kb)
          Interrupt:7 Base address:0xe800

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:1034 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1034 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:82068 (80.1 Kb)  TX bytes:82068 (80.1 Kb)
```

4.3.2 Configurazione del Routing

L'output del proprio comando `route` dovrebbe assomigliare a:

```
root@bridge:~> route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.0.3.129      0.0.0.0         255.255.255.128 U     0      0        0 br0
0.0.0.0         10.0.3.129      0.0.0.0         UG    0      0        0 br0
root@bridge:~>
```

4.3.3 Configurazione di Iptables

Si legga la sezione 4.2 (Pingalo Jim!).

4.4 Nota finale (Importante!)

Mi piacerebbe avere vostre notizie! :-)
Avete gradito il viaggio?
Vi manca qualcosa?
Avete bisogno di aiuto? (Chiedete al vostro assistente locale ;-) oppure rtfm.
Siete ancora online? Allora mandatemi un messaggio via email. Mi farà contento.
Volete mandarmi un assegno? Per favore non posso accettarli.. (Sto scherzando;)
Date valore al mio tempo, semplicemente mandatemi qualche bella parola, è sufficiente.
Niente motiva di più dei partecipanti felici che ti danno feedback positivi.
Per cui, forza, spendete un minuto e scrivetemi un email!
Grazie!

```
Nils
```

4.5 Bug-Notes

Pare vi sia un bug nel codice del br-nf:

```
From: Bart De Schuymer <bart.de.schuymer_@_pandora.be>
Date: Sun, 1 Sep 2002 21:52:46 +0200
To: Nils Radtke <Nils.Radtke_@_Think-Future.de>
Subject: Re: Ethernet-Brigde-netfilter-HOWTO

Ciao Nils,

[...]
Inoltre, il debugging del filtraggio dei pacchetti con la patch br-nf è
in generale una cattiva idea. Possono essere segnalati molti falsi avvisi
(relativi a presunti bug) nei log.
[...]
```

Personalmente non ho mai riscontrato falsi positivi nei miei log. Forse il bug è stato corretto. A questo proposito Bart ha scritto:

```
From: Bart De Schuymer <bart.de.schuymer_@_pandora.be>
Date: Mon, 2 Sep 2002 18:30:25 +0200
To: Nils Radtke <Nils.Radtke_@_Think-Future.de>
Subject: Re: Ethernet-Brigde-netfilter-HOWTO

On Monday 02 September 2002 00:39, Nils Radtke wrote:
> La revisione del codice nf-debug nel br-nf sarà migliorata?

Devo ammettere di non aver più fatto girare alcun kernel con il
netfilter debugging attivo ultimamente. Di sicuro qualche mese
fa generava dei falsi positivi (la mailing list sul bridge contiene
diversi post su questo problema), mi è mancato il tempo di capire
il motivo e se accade ancora. È nella mia lista delle cose da fare.
[...]
```

Ma (alla data attuale, 19-09-2002) non ho trovato un annuncio ufficiale che segnalasse la chiusura del bug. Si controlli costantemente la 6.1 (ethernet bridge mailinglist) se si è interessati alla risoluzione del problema.

5 Esperienze degli utenti

5.1 Fedora Core 3

James Dinkel (jdinkel_ @ _gmail.com) ha scritto Martedi, 8 Mar 2005 10:59:22 -0600:

```
[...]
Sto usando Fedora Core 3 e tutto quello che ho dovuto fare è stato "yum install bridge-utils"
per utilizzare il comando brctl. Non ho dovuto fare nessuna ricompilazione del kernel o
configurazioni o giochetti con i moduli del kernel.
E'stato molto facile.
[...]
```

6 Links

L'autore dell'howto può essere contattato via e-mail .
Homepage dell'autore .

6.1 Ethernet-Bridge

- `Ethernet Bridge Mailinglist`
- Utilità spazio utente, patch, ecc.: `Home of Linux kernel Ethernet Bridge`
- `Bridge-STP-HOWTO`
- `Firewalling for Free, Shawn Grimes`

6.2 Argomenti correlati:

- Filtraggio dei pacchetti, Ethernet-Bridging-Tables:
  `ebtables, sourceforge`
  `ebtables, caratteristiche supportate`
  `ebtables, esempi:  base , avanzati`
- IP mode, estensione Linux Bridge: `IP mode, LVS`
- Linux in ambienti High-Availability: `High-Availability Linux`
- Linux Virtual Server: `LVS`