UNIVERSITA' DEGLI STUDI DI PAVIA
Sede distaccata di Mantova
Laboratorio di RETI DI CALCOLATORI
(Prof. Giuseppe Federico Rossi)

Manuale di configurazione del protocollo IPv6 nel sistema operativo FreeBSD

ad uso degli Studenti del Corso di RETI TELEMATICHE
C.d.L. II livello in Ingegneria Informatica

A cura del Dott. Luca Anselmi ed Emanuele Goldoni

Versione 0.1

Mantova, li 29/11/2005

1 - PREMESSA

Il presente manuale è stato pensato quale ausilio per gli Studenti del CdL in Ingegneria Informatica, impegnati nello svolgimento di esercitazioni pratiche presso il Laboratorio Reti attivo presso la sede distaccata di Mantova dell'Università degli Studi di Pavia.
Più precisamente verrà illustrata la configurazione di rete nel sistema operativo FreeBSD, con particolare riferimento allo stack TCP/IPv6 e ai protocolli PPP (Point-to-Point Protocol) ed EthernetV2/IEEE 802.3, formalizzati dagli standard definiti a livello internazionale da IETF e disponibili per consultazione all'URL *http://ww.ietf.org*. Per ogni riferimento dettagliato al funzionamento dei protocolli utilizzati si rimanda pertanto a tali documenti.

La scelta della piattaforma software utilizzata non è peraltro affatto casuale. Per molti anni Unix è stato il sistema operativo più importante e diffuso ed anche oggi, specialmente in ambito universitario o di ricerca, i sistemi Unix-like (vedi Linux) sono molto utilizzati. Originariamente progettato negli anni 70 nei laboratori della compagnia telefonica americana AT&T, UNIX si è ben presto diviso in due differenti rami di sviluppo: System 5, licenziata dalla stessa AT&T a differenti compagnie, e BSD (Berkeley software distribution). Quest'ultima versione in particolare è stata realizzata dal Computer Systems Research Group (CSRG) californiano dell'Università di Berkeley e distribuita gratuitamente ad altre università e centri di ricerca. Un grande impulso alla diffusione di BSD è stato senza dubbio dato dal Dipartimento della Difesa degli Stati Uniti, che alla fine degli anni 80 scelse proprio questo sistema operativo come piattaforma di base per l'implementazione di TCP/IP. e, quindi, di quello che sarebbe poi divenuto Internet.
L'ultima versione ufficiale rilasciata dal CSGR, 4.4BSD-Lite Release 2, risale al 1995: dal quel momento diversi gruppi hanno proseguito lo sviluppo perseguendo obiettivi differenti e realizzando così più versioni. Quelle più note sono FreeBSD, OpenBSD, NetBSD e Darwin e tutte sono oggi attivamente sviluppate liberamente disponibili, compreso il codice sorgente. L'attenzione degli sviluppatori di FreeBSD si è inizialmente concentrata sulla piattaforma x86, cercando di ottenere il massimo delle prestazioni e il corredo software il più completo possibile. NetBSD invece è una distribuzione pensata per essere il più portabile possibile (virtualmente è in grado di funzionare su ogni piattaforma esistente), con un occhio di riguardo per i sistemi embedded. OpenBSD, nato più di recente da un fork di NetBSD, pone l'accento sulla sicurezza attraverso meccanismi di audit del codice e l'integrazione di crittografia, SSH in primis. Infine il nucleo del sistema operativo Apple OS X, Darwin, è sostanzialmente un sistema BSD che si appoggia ad un microkernel Mach. Una recentissima distribuizione derivata da FreeBSD 5 è poi DragonFly BSD; questa in particolare introduce un supporto più efficiente al multi-threading in sistemi mono o multi-processore, comportando sotto questo punto di vista notevoli migliorie anche all'interno dello stack di rete.

Per quel che riguarda in particolare IPv6, FreeBSD integra nativamente da alcuni anni il codice del progetto Kame, un'iniziativa giapponese nato con lo scopo di realizzare un valido stack di rete v6 per i sistemi operativi Unix BSD, con un occhio di riguardo per gli aspetti legati alla sicurezza (IPSec). La qualità e la stabilità del prodotto realizzato sono indubbiamente elevate e, insieme al software incluso nel nucleo del sistema, sono stati sviluppati anche numerosi strumenti per il nuovo protocollo (ping6, traceroute6 solo per citarne alcuni, così come tunneling, faithd e altri meccanismi di transizione 4-6).

Praticamente quasi tutto ciò che troverete in questo manuale presuppone l'utilizzo di un terminale testuale, la shell di Unix, ma è allo stesso modo realizzabile con la rispettiva interfaccia grafica (XWindows, GNOME, KDE, ecc..). Questo viene fatto non per rendere le cose più difficili, bensì ma per dare una visione il più standardizzata possibile della configurazione di rete nei sistemi operativi Unix-like. Tra questi, un altro sistema operativo simile a FreeBSD e con un discreto supporto per IPv6 è sicuramente Linux (in particolare si consiglia di utilizzare le più recenti

versioni del kernel 2.6.x); per ottenere informazioni più specifiche circa la configurazione dell'Internet Protocol di nuova generazione sotto questo sistema operativo si rimanda al *Manuale di configurazione del protocollo IPv6 nel sistema operativo Linux*, disponibile sempre sul sito del Laboratorio Reti.

sicurezza

Sono state istituite alcune varianti per mantenere la sicurezza nella gestione dei sistemi. In pratica, per l'esecuzione dei comandi specificati, è spesso necessario utilizzare l'utenza di amministratore (`root`); se ciò fosse concesso, un utente incauto o malintenzionato potrebbe danneggiare il sistema.
Per questo non è consentito utilizzare l'utenza root, ma sarà consentito l'utilizzo dell'identità `labreti#` (esempio `labreti1`, `labreti2`, ecc..) con password uguale al nome utente.

Per gestire i comandi di rete, sono ammessi solamente i comandi descritti con esecuzione tramite l'utility `sudo` secondo la sintassi:
`$ sudo <nome comando> <opzioni comando>`
In pratica ogni comando che richiede l'utenza `root` per poter funzionare deve essere preceduto da `sudo`.

Le esercitazioni sono impostate in modo tale che le configurazioni modificate non vengano memorizzate dal sistema; quindi, se il sistema viene riavviato, le impostazioni saranno perse.

2 - CONNESSIONI HARDWARE

I 5 PC che vi troverete di fronte sono dotati di 3 schede Ethernet 10/100 Mbps ognuno e di 2 interfacce seriali.

glossario

È necessario avere molta familiarità con i termini **host**, **interfaccia e scheda** in quanto verranno frequentemente utilizzati in tutto il manuale.

Host: rappresenta il singolo PC o router interconnesso nella rete. Talvolta si fa confusione perchè si chiama l'host indicando il suo indirizzo IP. Occorre fare molta attenzione perchè un host può avere molti indirizzi IP (ed è questo il nostro caso). Nella rete Internet, di solito un host ha un solo indirizzo IP pubblico, quindi dall'esterno può essere indicato con l'indirizzo IP (o anche con il nome completo di dominio) ma i router, ad esempio, hanno molti indirizzi IP pubblici.

Interfaccia: rappresenta ciò che il sistema identifica come dispositivo di connessione di rete (scheda di rete, porta seriale e persino tunnel "virtuali"). Un host può avere più interfacce (i nostri hanno 3 schede di rete ciascuno più le due seriali) ed ogni interfaccia può avere più indirizzi IP associati.

Scheda: è il componente "hardware" che interconnette fisicamente alla rete, detto anche NIC (Network Interface Card). Per maggiori informazioni si consiglia di far riferimento alla *Guida*

Connessioni hardware per le schede Ethernet

Le connessioni possono essere effettuate con i cavi dotati di PLUG RJ45 (simili a quelli telefonici) a 8 fili che possono essere "dritti" o "incrociati" (cross-cable). Ogni scheda di rete ha solitamente una presa conforme allo standard RJ45; se si usano i cavi dritti per interconnettere le schede è necessario utilizzare anche un HUB (concentratore) mentre, se si usano cavi incrociati, è possibile interconnettere con un cavo direttamente due schede di rete tra loro.
Nelle esercitazioni si useranno spesso gli HUB anche se, non dovendo effettuare sniffing del traffico di rete, andrebbero benissimo anche gli SWITCH.
Un HUB (o una cascata di HUB) può interconnettere anche decine di host e gli host connessi su un HUB si intendono come appartenenti ad una stessa sottorete. In teoria è possibile avere più sottoreti su uno stesso HUB, ma questa possibilità non sarà presa in considerazione poiché si tratta di una pratica poco diffusa.

Connessioni hardware per le porte seriali

Un cavo Ethernet incrociato non è l'unico modo possibile per realizzare una connessione punto-a-punto tra due PC; nel nostro caso infatti utilizzeremo un cavo seriale con connettore a 9 poli (DB9) in configurazione *null-modem* (si faccia riferimento anche in questo caso alla *Guida all'Hardware di Rete*).

3 - CREAZIONE DEI DISPOSITIVI DI RETE

Creazione delle interfacce ethernet

Nei sistemi Unix ogni dispositivo (device) è visto come un file e i files che rappresentano tutti i dispositivi si trovano nella directory `/dev`.
Per assegnare un nome alle interfacce di rete, FreeBSD utilizza il nome del driver seguito da un numero, determinato in base all'ordine in cui i dispositivi sono stati rilevati durante il boot del kernel. Per esempio `sis2` dovrebbe essere la terza scheda Ethernet sul sistema che utilizza il driver sis (Silicon Integrated Systems 900), mentre `tl0` la prima scheda Texas Instruments ThunderLAN.
I driver per tutte le schede più diffuse sono generalmente già inclusi nel kernel e quindi non è generalmente necessario alcun tipo di operazione per la creazione dei dispositivi di rete Ethernet.
Tra le schede di rete più diffuse molti ricorderanno quelle basate su chipset NE2000-compatibili/RealTek8029 (`ed`), le schede 3Com Etherlink 3C509x (`xl`), Etherlink III (`ep`) o i tantissimi dispositivi Realtek RTL8129/39 (`rl`); la scheda di rete attualmente più diffusa è basata sul chip Realtek 8139 ed è la stessa utilizzata nei nostri PC. Nel caso in cui il riconoscimento automatico di questo dispositivo fallisca può essere necessario ricorrere al caricamento manuale dell'apposito modulo del kernel; per fare questo è necessario caricare il modulo del kernel specifico attraverso il comando
`kldload <nome_modulo>`
o aggiungendo una riga opportuna al file `/boot/loader.conf` del tipo:
`<nome_modulo>_load="YES"`
In generale, se non si riesce a far funzionare una scheda di rete o altro dispositivo installato, possono essere di aiuto durante la fase di boot i comandi `lsdev` e `lsmod`, che mostrano rispettivamente tutti i dispositivi per i quali potrebbe essere possibile caricare un modulo e i moduli attualmente caricati.

Creazione delle interfacce ppp su porte seriali

Assodato che i dati passeranno su una "rete" costituita da cavi seriali, si tratta ora di trovare un DLC in grado di supportare IPv6. Purtroppo il vecchio protocollo di comunicazione dati per le interfacce seriali SLIP non supporta ne mai supporterà IPv6 e quindi l'unica possibilità è rappresentata dal protocollo PPP (RFC2472 - IP Version 6 over PPP), lo stesso usato ad esempio per una normale connessione ad Internet tramite modem. Fortunatamente il programma `ppp` permette di stabilire una connessione con questo protocollo semplicemente utilizzando il comando:

`$ ppp -dedicated <profilo>`

dove `–dedicated` indica al sistema che la connessione instaurata con l'altro computer sarà diretta e non condivisa da altri. Tutte le altre opzioni possono essere specificate più agevolmente all'interno del file di configurazione principale `/etc/ppp/pppd.conf`. In particolare è possibile impostare più profili con opzioni differenti, indicando al servizio quale di questi utilizzare di volta in volta; sicuramente è necessario per ogni profilo indicare il dispositivo predefinito (`/dev/cuaa0` per la COM1 e `/dev/cuaa1` per la seconda porta seriale).

Questo comando creerà un nuovo dispositivo di rete `tun#` attivabile e configurabile direttamente con il comando `ifconfig`, esattamente come accade per una comune scheda di rete.

Per semplificare le operazioni di configurazione in laboratorio sono stati preparati due profili `serial0` e `serial1`, utilizzabili per attivare direttamente la connessione sulla porta desiderata.

4 - IMPOSTAZIONE DEI PARAMETRI DI RETE

Ogni interfaccia di rete ha alcuni parametri che è necessario configurare per far sì che possa funzionare, alcuni dei quali alcuni prevedono una configurazione automatica:

- Attivazione dello stack IPv6
- Configurazione delle interfacce Ethernet;
- Configurazione degli indirizzi IPv6;
- Attivazione delle interfacce;
- MTU (Maximum Tranfer Unit);
- Attivazione e disattivazione dell'interfaccia.

Attivazione dello stack IPv6

Lo stack IPv6 all'interno di FreeBSD viene normalmente fornito, insieme a moltissimi altri componenti del kernel, come funzionalità già integrata nativamente.
Per abilitare il supporto IPv6 per tutte le interfaccie presenti nel sistema è sufficiente inserire nel file `/etc/rc.conf` , se non già presente, la riga di configurazione:

```
ipv6_enable="YES"
```

Configurazione interfacce Ethernet

Per le interfacce Ethernet (`eth#`) è necessario impostare indirizzo IPv6 di tipo global ed un prefisso e, in casi particolari, anche MTU o eventuali indirizzi di alias.
Ogni interfaccia Ethernet deve essere configurata con due tipi di indirizzi IPv6: uno di tipo link local, locale al link, ed un secondo di tipo global che deve essere univoco a livello mondiale.
Gli indirizzi link-local nel caso delle interfacce Ethernet sono assegnati dinamicamente per via algoritmica partendo dal MAC Address della scheda, mentre gli indirizzi di tipo Global devono essere assegnati in maniera esplicita.

Per configurare parametri di rete si usa il comando `ifconfig`.

Se si esegue il comando senza parametri, esso riporta l'elenco delle interfacce attive in questo momento. Di seguito un possibile esempio di output del comando ifconfig:

```
eth0      Link encap:Ethernet  HWaddr 00:50:FC:CD:20:2A
          inet6 addr: 3ffe:1001:1c0:2::/64 Scope:Global
          inet6 addr: fe80::250:fcff:fecd:202a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:7 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:566 (566.0 b)
          Interrupt:9 Base address:0xc000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:1011 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1011 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1485416 (1.4 MiB)  TX bytes:1485416 (1.4 MiB)
```

Da questo esempio è possibile verificare che sono attive due interfacce: la `eth0` e quella di *loopback* (indicata con `lo`).
Per ogni interfaccia inoltre vengono riportate informazioni significative quali:

- Il MAC-Address della scheda
- Gli eventuali indirizzi IPv4, e IPv6 (global e link-local)
- Il valore di MTU
- Informazioni varie riguardanti i pacchetti ricevuti o trasmessi.

Attivazione e disattivazione dell'interfaccia

Ogni interfaccia è attivabile in maniera esplicita tramite il comando:

```
$ ifconfig <nome interfaccia> up
```

Si consiglia di attivare un'interfaccia prima di assegnarle eventuali indirizzi IPv6 e si ricorda che i computer utilizzati per le esercitazioni all'avvio vedono tutte le interfacce disattivate.

Allo stesso modo è possibile disattivare in maniera esplicita un'interfaccia specifica digitando:

```
$ ifconfig <nome interfaccia> down
```

Se fosse necessario visualizzare tutte le interfacce disponibili, indipendentemente dal fatto che siano attive o meno, lo si può fare con:

```
$ ifconfig -a
```

Assegnazione di un indirizzo IPv6

Per impostare l'indirizzo IPv6 di una scheda di rete, basta scrivere:

```
$ ifconfig rl# inet6 <indirizzo IPv6>/<prefisso>
```

oppure

```
$ ifconfig rl# inet6 <indirizzo IPv6> prefixlen <lungh_prefisso>
```

dove **#** rappresenta il numero della scheda di rete.
Nel caso fosse necessario assegnare più indirizzi alla stessa interfaccia, basterà ripetere lo stesso comando ovviamente con un indirizzo diverso.

Allo stesso modo, se si volesse togliere un indirizzo IPv6 precedentemente assegnato:

```
$ ifconfig rl# inet6 delete <indirizzo IPv6>/<prefisso>
```

laddove l'indirizzo e il relativo prefisso devo essere stati precedentemente assegnati alla scheda.

Si ricorda che l'indirizzo link-local, nel caso delle interfacce Ethernet, viene assegnato in maniera automatica e non ha quindi bisogno di essere esplicitato.

N.B. Le impostazioni di `ifconfig` verranno perse in caso di riavvio del sistema. Per rendere permanenti le configurazioni effettuate occorre memorizzarle nel file `/etc/rc.conf`; si veda a riguardo il capitolo 6 per una trattazione più approfondita.

Impostazione della MTU (*Maximum Transmission Unit*)

MTU è la lunghezza massima in byte che può avere un pacchetto IP trasmesso sulla Ethernet. Per lo standard ethernet v2 (IEEE 802.3) si ha di norma `MTU = 1500`, che è il massimo valore impostabile. Il minimo MTU è impostabile a 64 byte (per le Ethernet), ma il protocollo IPv6 non accetta MTU al di sotto di 1280 byte. In condizioni normali questa impostazione viene sempre lasciata a 1500 bytes.
Per impostarlo:

```
$ ifconfig rl# mtu <mtu_val>
```

loopback

Va detto che in ogni host è presente una interfaccia di *loopback* che rappresenta l'host stesso; solitamente viene chiamata `lo0` e ha indirizzo IPv4 `127.0.0.1/8` e indirizzo IPv6 `::1/128`.
Questa interfaccia ha lo scopo di poter utilizzare lo stack di protocolli TCP/IP anche se il computer non è dotato di schede di rete. Rappresenta infatti una interfaccia di rete "virtuale", nel senso che i dati non vengono inviati a nessun dispositivo ma vengono ritornati all'host stesso. In questo modo è anche possibile verificare se il sistema operativo è in grado di far funzionare correttamente il modulo TCP/IP (chiamato *socket*) .

5 - CONFIGURAZIONE DEI PARAMETRI DI ROUTING

Il routing è un argomento estremamente vasto (protocolli di routing, *load balancing*, routing dinamico ecc.) che non può certamente essere esaurito in poche righe; per ovvi motivi noi ci limiteremo alle nozioni base necessarie per portare a termine l'esercitazione e per risolvere comunque le più diffuse problematiche.
Il comando principale che permette di gestire il routing tra interfacce è `route`. `route` permette di "manipolare" le tabelle di routing del sistema, inserendo o cancellando regole di routing statiche, che cioè che non cambiano mai. Durante l'esercitazione basterà quindi implementare le tabelle di routing già predisposte durante la soluzione dell'esercizio proposto utilizzando le opzioni del comando `route` presentati di seguito.

Per aggiungere una regola che riguarda una rete o una sottorete:
`$ route add -inet6 <dest_ip>/<netmask> <gateway>`

Per aggiungere una regola riguardante un singolo host:
`$ route add –inet6 -host <dest_ip> <gateway>`

Per aggiungere la regola di *default*:
`$ route add -inet6 default <gateway>`
oppure:
`$ route add -inet6 ::/0 <gateway>`

Per eliminare una regola è possibile utilizzare la stessa sintassi vista sopra sostituendo `add` con `delete`.

Per visualizzare rapidamente le regole inserite si deve utilizzare poi il comando `netstat`:
`$ netstat  -r –f inet6`

Per visualizzare la tabella in formato esteso, potendo così di leggere in maniera completa le regole immesse è tuttavia necessario aggiungere l'opzione '`W`':
`$ netstat  -r –W –f inet6`

E' da tenere presente che nella tabella di routing per ogni interfaccia vengono già inserite di default una o più regole, che noi però trascureremo.

Un'altra opzione che può avere qualche applicazione è `-interface <interf>`, che permette di specificare su quale interfaccia deve essere inviato il pacchetto destinato ad una determinata rete.
Nella maggior parte dei casi non è necessario specificare questo parametro, in quanto l'interfaccia da utilizzare può essere identificata immediatamente in base al proprio indirizzo IP; tuttavia torna utile nel caso in cui ad esempio si voglia testare un host connesso alla stessa rete mediante due diverse interfacce. Un altro utilizzo si può trovare nei cosiddetti tunnel, ovvero quando un pacchetto non deve essere inviato direttamente ad una scheda ma deve essere prima "incapsulato" in un altro pacchetto, per poi essere immesso in un tunnel.

Ulteriori opzioni di `route` sono usate per scopi raffinati o diagnostici, nonché dai programmi di instradamento automatico dei pacchetti quali il demone `routed`, che implementa il protocollo RIPv2.

6 - ABILITAZIONE DEL FORWARDING

In FreeBSD è possibile che un host funzioni da router, commutando i pacchetti da una interfaccia all'altra (*forwarding*). Per default, e per ragioni di sicurezza, il sistema operativo disabilita la funzione di forwarding; se si intende rendere attivo il forwarding bisogna pertanto abilitarlo esplicitamente andando a modificare le impostazioni del kernel stesso.

sysctl

Molti dei parametri di funzionamento del kernel FreeBSD possono essere ottimizzati "al volo" senza dover ricorrere ad un riavvio della macchina ad ogni modifica. Tali modifiche possono essere apportate utilizzando il comando `sysctl` e salvate all'occorrenza nel file di configurazione `/etc/sysctl.conf`, per poterle così caricare ad ogni boot del sistema.
Ad esempio il numero massimo di processi consentiti per utente è impostabile a 100 digitando:
`sysctl kern.maxprocperuid=100`

Nel nostro caso il forwarding deve essere abilitato sui PC che fungono da Router.
Per abilitare (ON = `1`) il forwarding su una specifica interfaccia scrivere:

`$ sysctl net.inet6.ip6.forwarding=1`

Per disabilitarlo procedere in maniera analoga, impostando il valore del parametro a `0`.

Per controllare che il forwarding sia veramente abilitato, si può scrivere:

`$ sysctl -a | grep forwading`

e verificare che alla riga:

`net.inet6.ip6.forwarding`

corrisponda il valore desiderato.

rc.conf

Il file di configurazione principale contenente le impostazioni dell'hostname e di ogni interfaccia di rete, nonché dei servizi da avviare durante il boot è `/etc/rc.conf`. Tutte le opzioni salvate in questo file vengono automaticamente caricate ad ogni boot del sistema e la sintassi è del tipo:
`<parametro>="<valore>"`
Ad esempio per avviare al boot il servizio di autenticazione Kerberos5 basta impostare:
`kerberos5_server_enable="YES"`
Allo stesso modo, per abilitare automaticamente all'avvio il protocollo IPv6 ed il forwarding dei pacchetti IPv6 è sufficiente aggiungere nel file le righe:
`ipv6_enable="YES"`
`ipv6_gateway_enable="YES"`
Per assegnare invece staticamente un indirizzo IPv6 ad esempio all'interfaccia `fxp0` si usa:
`ipv6_ifconfig_fxp0="<indirizzo_ip6>"`
Mentre il default router è impostabile con il parametro:
`ipv6_defaultrouter="<indirizzo_ip6_default_gw>"`

7 - VERIFICHE DEL FUNZIONAMENTO DELLA RETE

Siamo giunti alla prova finale, ovvero la verifica delle connessioni di rete. Si tratta di verificare se ogni PC è in grado di "comunicare" in maniera esatta con gli altri PC.

Il comando utilizzato per verificare le connessioni IPv6 è `ping6`.

`ping6` ha la stessa funzione del noto `ping` per la versione IPv4. La risposta a questa domanda può assumere tre forme diverse:

1) Il `ping` va a buon fine e l'host interpellato risponde correttamente;
2) L'host contattato non risponde alla richiesta;
3) Il nostro host non riesce ad inviare il pacchetto di ping.

Un esempio di `ping6`:

```
$ ping6 3ffe:1001:1c0:ffff::
```

la cui risposta può essere:

```
PING 3ffe:1001:1c0:ffff::(3ffe:1001:1c0:ffff::) 56 data bytes
64 bytes from 3ffe:1001:1c0:ffff::: icmp_seq=0 ttl=64 time=0.187 ms
64 bytes from 3ffe:1001:1c0:ffff::: icmp_seq=1 ttl=64 time=0.181 ms
64 bytes from 3ffe:1001:1c0:ffff::: icmp_seq=2 ttl=64 time=0.179 ms
64 bytes from 3ffe:1001:1c0:ffff::: icmp_seq=3 ttl=64 time=0.179 ms
```

Questo indica non solo che l'interfaccia interpellata sta funzionando e risponde ai "ping", ma che abbiamo anche ricevuto 64 bytes in un tempo di andata e ritorno di circa 0.0180 ms.

NB: il comando ping senza opzioni esegue un test continuativo; per interromperlo è necessario premere i tasti **`CTRL + C`**.

Ping verso le interfacce locali

Una delle prime cose da verificare in seguito alla configurazione di un host, è la corretta configurazione delle interfacce locali. Per verificare la corretta assegnazione dell'indirizzo IP all'interfaccia è possibile eseguire verso di essa un `ping6`.

Esempio:
Supponiamo che un host abbia i seguenti indirizzi IP:
`rl0: 3ffe:1001:1c0:1::1`
`rl1: 3ffe:1001:1c0:3::2`

Lanciando i comandi

```
$ ping6 ::1     (interfaccia di loopback)
$ ping6 3ffe:1001:1c0:1::1
$ ping6 3ffe:1001:1c0:3::2
```

se tutte le interfacce sono funzionanti, come risposta ad ogni ping specifico avremo:

```
64 bytes from xxx: icmp_seq=1 ttl=64 time=0.056 ms
64 bytes from xxx: icmp_seq=2 ttl=64 time=0.056 ms
```

Se tra questi comandi ce n'è uno (o più) che non dà la risposta corretta, si è sbagliato qualcosa durante l'impostazione dei parametri di rete oppure, più semplicemente, non si è digitato il comando
$ `ifconfig rl# up`.

Test di raggiungibilità degli altri host

Realizzata la rete e configurati i computer, occorre verificare se il nostro host è in grado di comunicare correttamente con gli altri host. Questo lo si fa "pingando" tutti gli altri indirizzi IP della rete.
Le possibili risposte del sistema sono:

a) *`64 bytes from xxx: icmp_seq=1 ttl=64 time=0.056 ms`*

Se tutto funziona correttamente, appaiono sempre righe simili a quelle sopra citate, dove xxx è l'indirizzo IPv6 puntato:

b) `connect: Network is unreachable`

Se uno dei nodi intermedi non riesce ad instradare il pacchetto di "ping" (non esiste una regola di routing per instradare quel pacchetto):

c) `From 192.168.0.1 icmp_seq=1 Destination Host Unreachable`

Se la rete è raggiungibile, ma l'interfaccia non risponde:

Possibili cause di errore possono essere:

1. Si è sbagliato l'indirizzo IP, o comunque l'indirizzo non può essere raggiunto, pur essendo logicamente adiacente all'host in questione.
2. Alcune tabelle di routing sono sbagliate o incomplete sul nodo stesso o su host intermedi.
3. Non si è configurato correttamente il percorso di andata e ritorno del pacchetto tramite il comando route sui vari host.
4. Se tra il nostro host e il destinatario c'è di mezzo un firewall (anche se non è il nostro caso), talvolta i ping possono non andare a buon fine; in tal caso non significa che l'host non è raggiungibile, ma semplicemente che il firewall ha impedito il `ping` disabilitando il protocollo ICMP.

Verifica del percorso seguito da un pacchetto

Talvolta è necessario verificare quale percorso abbia seguito un pacchetto; il comando `traceroute6` permette di registrare il percorso seguito sia all'andata che al ritorno visualizzando gli indirizzi IP coinvolti

`$ traceroute6 <indirizzo ip>`

Verifica della comunicazione

A rete configurata, una verifica molto importante consiste nell'osservare i pacchetti che transitano sul canale di comunicazione: tale procedura è in gergo chiamata "sniffing" della rete.
In parole povere, essa consiste nel vedere "cosa passa" nei cavi.

Tcpdump

Un programma utile per eseguire lo sniffing è `tcpdump`. Questo software permette di mettersi in "ascolto" su una certa interfaccia e vedere l'intestazione IP e TCP dei pacchetti che passano.
Se si lancia:
`$ tcpdump -i <nome interfaccia>`
ad esempio
`$ sudo tcpdump -i rl0`
Si ottengono delle righe simili:

```
...
17:21:17.662664 3ffe:1001:1c0::1 > 3ffe:1001:1c0::2: icmp6 echo request 0
17:21:17.662777 3ffe:1001:1c0::2 > 3ffe:1001:1c0::1: icmp6 echo reply 0
```

In pratica significa che alle `17:21:17` il sistema `3ffe:1001:1c0::1` ha lanciato una `icmp6 echo request` (`ping6`) verso `3ffe:1001:1c0::1`, il quale ha poi risposto (`echo reply`).

Dall'esempio proposto si vede come `tcpdump` sia in grado di fornire informazioni molto importanti sul transito dei pacchetti di rete.

Ethereal

Un altro tool, molto più potente di `tcpdump`, è `ethereal`. Per farlo funzionare è necessario accedere all'interfaccia grafica e scrivere a terminale grafico `sudo ethereal` (anche questo comando è normalmente disabilitato per impedire che qualsiasi utente senza particolari priveligi possa "sniffare" il traffico di rete della macchina su cui sta lavorando). A questo punto scegliendo `start` dal menù `capture` è possibile indicare il dispositivo di rete da "sniffare"; premendo poi `OK` il programma inizierà a raccogliere informazioni sui pacchetti in transito, classificandoli per protocollo.
Una volta concluse le operazioni di cattura, sarà possibile andare a vedere nel dettaglio tutti i pacchetti transitati ed il loro contenuto (e non solo le intestazioni, come accade invece con `tcpdump`).

Verifica delle comunicazioni TCP/UDP

Il comando `netstat` visualizza le connessioni TCP, le porte d'ascolto TCP e UDP, nonché lo stato degli stream Unix.
Se si digita:

```
# netstat -an |more
```

o meglio:

```
# netstat -an --inet6 | more
```

Si ottiene, nella parte iniziale della risposta, l'elenco completo delle porte TCP e UDP che attualmente il sistema sta "ascoltando".
Inoltre l'elenco contiene anche le connessioni TCP che sono instaurate (*established*), nonché l'elenco delle trasmissioni di pacchetti UDP che si sono appena verificate.

8 - IMPOSTAZIONE DEI NOMI DEGLI HOST

Ogni host connesso in rete ha un suo nome e a tale nome è associato un indirizzo IP; i nomi tuttavia sono molto più pratici e semplici da ricordare rispetto ad una sequenza di numeri.
In teoria ogni host dovrebbe conoscere sempre l'indirizzo IP associato a ciascuno degli altri host connessi per poter comunicare con esso nel caso un cui venga specificato solamente il nome; questo significa che ogni host deve essere in grado di "risolvere" un nome, cioè trovare l'indirizzo IP associato ad un *hostname*.
Nelle reti piccole è possibile impostare staticamente in un tabella presente su ogni host i nomi degli altri host e l'indirizzo associato; nelle grandi reti (Internet) questa però non è più ovviamente una soluzione praticabile, poiché in reti di questo tipo

1) i nomi, gli indirizzi e le informazioni sono in continua evoluzione;
2) gli host connessi possono essere tantissimi (si parla di quasi 200 milioni di host connessi a Internet);
3) l'aggiornamento e la ricerca delle voci deve avvenire in tempi accettabili

Quello che viene utilizzato in Internet è invece un servizio che si occupa di risolvere i nomi in maniera automatica per l'intera rete Internet; il servizio DNS rappresentano database distribuiti su migliaia di calcolatori connessi a Internet in grado di mantenere tutte le informazioni per i domini registrati. Per fare delle prove, con un PC correttamente connesso a Internet si possono usare i comandi `nslookup`, `dig` e `host` per trovare l'indirizzo IP di un host specificando il suo nome. L'output del comando `dig` per un host IPv6-enabled è:

```
# dig www6.ipv6.polito.it

; <<>> DiG 9.2.4 <<>> www6.ipv6.polito.it
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 37534
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 1, ADDITIONAL: 1

;; QUESTION SECTION:
;www6.ipv6.polito.it.         IN          A

;; ANSWER SECTION:
www6.ipv6.polito.it.          349059      IN    CNAME freebsd.ipv6.polito.it.
freebsd.ipv6.polito.it.       349059      IN    A     130.192.16.134

;; AUTHORITY SECTION:
ipv6.polito.it.               349059      IN    NS    freebsd.ipv6.polito.it.

;; ADDITIONAL SECTION:
freebsd.ipv6.polito.it. 349059      IN    AAAA  2001:760:400:1:201:2ff:fe07:a2d7

;; Query time: 432 msec
;; SERVER: 2001:760:2::a#53(2001:760:2:0::a)
;; WHEN: Mon Jan 17 16:40:24 2005
;; MSG SIZE  rcvd: 117
```

La risposta ci dice che il server `www6.ipv6.polito.it` (il nome della macchina che risponde è `freebsd.ipv6.polito.it`) ha per indirizzo IPv4 `130.192.16.134` (si noti come sia una risposta di tipo A ad una query DNS) e `2001:760:400:1:201:2ff:fe07:a2d7` come indirizzo IPv6 (risposta di tipo AAAA). Come facilmente si può intuire, in una piccola rete come quella realizzata da noi in laboratorio non è necessario impostare i servizi DNS; per i computer locali è sufficiente editare il file `/etc/hosts` e specificare:

```
IP   nome_completo_host   [nome_locale_host]
```

Ad esempio:

```
217.169.102.120     www.unimn.it    www
```

Una volta effettuata questa operazione su tutti i PC interessati, è possibile usare i nomi locali o i nomi completi per effettuare tutte le operazioni in sostituzione degli indirizzi IP (decisamente meno facili da ricordare!)

Per far sì che un sistema Unix connesso a Internet utilizzi il servizio DNS, basta inserire l'indirizzo IP del server DNS preferito nel file `/etc/resolv.conf`, nella seguente modalità:

```
nameserver <DNS_IP_Addr>
```

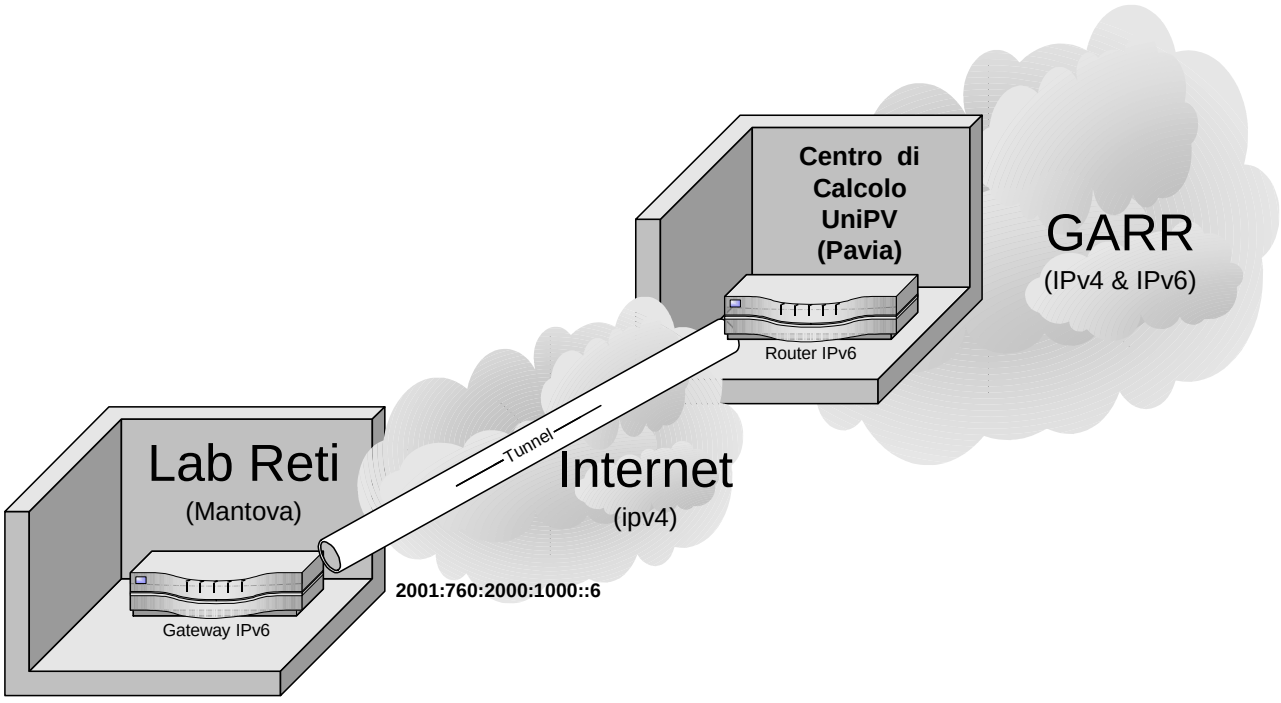
Ad esempio:

```
nameserver 2001:760:2:0::a
```

Questo indica che il sistema utilizzerà server DNS indicato per trovare gli indirizzi IP degli host che non sono presenti nel file `/etc/hosts`. Nel file `/etc/resolv.conf` si possono specificare più server DNS, utilizzando per ogni riga la forma sopra indicata (se si specificano più server DNS, nel caso in cui il primo non risponda, si utilizzerà il secondo e così via…).

9 – CONNESSIONE ALLA RETE MONDIALE IPv6

6Net è invece un progetto Europeo, nato nel 2002, per la sperimentazione di IPv6 nel vecchio continente. In Italia il progetto *6Net* prevede la realizzazione di una rete nativa IPv6, parallela all'attuale rete di produzione di GARR. La Rete GARR (il cui acronimo significa "Gestione Amplimento Rete Ricerca") è composta da tutte le Entità che rappresentano la Comunità Accademica e della Ricerca Scientifica in Italia. GARR segue direttamente lo sviluppo del protocollo IPv6 e della tecnologia ad esso collegata, partecipando ai gruppi di lavoro di RIPE e di IETF e coordinando in Italia le sperimentazioni del progetto europeo *6Net*. Nel progetto sono coinvolti attivamente diversi enti, tra cui il CASPUR, il CNR e le Università di Milano, Torino, Bologna, Napoli e Roma Tre. Nel Febbraio 2005 anche l'Università degli Studi di Pavia è stata inserita nell'elenco degli enti coinvolti nel progetti di sperimentazione 6NET-GARR. Lo spazio di indirizzamento IPv6 assegnato a Pavia per l'intera rete di Ateneo è 2001:760:2000::/48. Questo blocco di indirizzi è stato a sua volta suddiviso dal Centro di Calcolo dell'Università per realizzare più sottoreti; al Laboratorio Reti, attivamente impegnato all'interno dell'Ateneo nella sperimentazione del nuovo protocollo, è stato assegnato lo spazio di indirizzamento **2001:760:2000:1000::/56.**
Tuttavia non è disponibile a Mantova alcuna connessione diretta verso la dorsale GARR ma quest'ultima viene raggiunta tramite un punto di accesso offerto dal CdC di Pavia. La connessione IPv6 tra il Centro di Calcolo e la sede di Mantova è stata realizzata mediante Tunneling IPv6 incapsulato in IPv4.



Realizzazione pratica del Tunnel

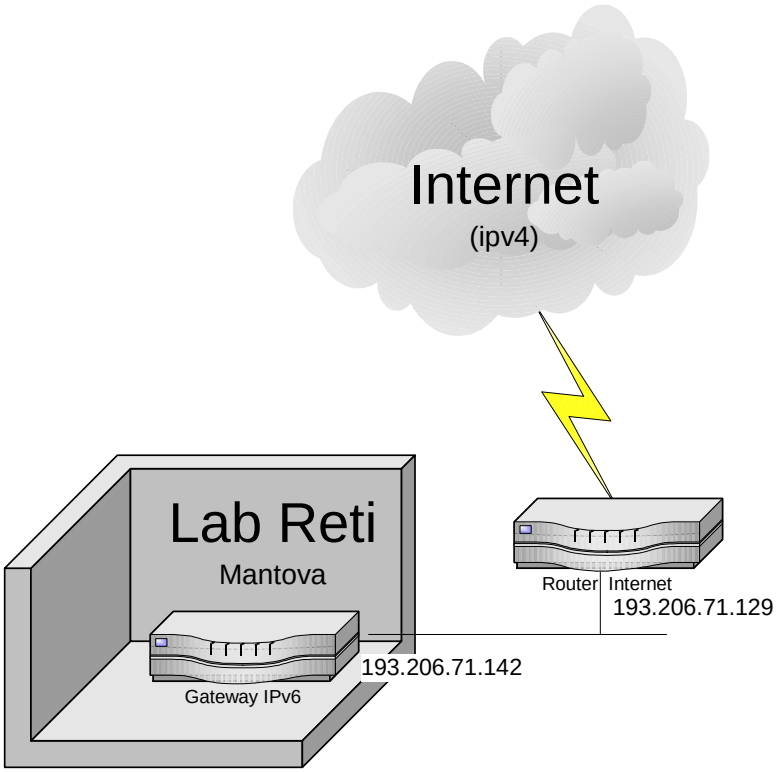
Come spiegato nella sezione precedente, la connettività verso la rete 6Net è realizzata da un tunnel che incapsula i pacchetti IPv6 all'interno di comuni pacchetti IPv4. All'interno del laboratorio esiste un computer destinato esclusivamente a svolgere tale funzione.

In particolare, il router utilizzato come end-point del tunnel consiste in un computer Dual-Stack con installato il sistema operativo FreeBSD 5.3. Questa macchina funge, all'interno della rete locale del Laboratorio, da gateway predefinito per il traffico sia IPv4 che IPv6 operando in maniera del tutto trasparente. Per rendere effettivo il tunnel sarà quindi necessario svolgere le seguenti operazioni:

a) Configurare un'interfaccia con numerazione pubblica verso la rete mondiale IPv4

b) Configurare il tunnel

c) Definire la una tabella di Routing capace di instradare verso il Tunnel tutte le comunicazioni che sono rivolte all'esterno.



a) Configurare un'interfaccia con numerazione pubblica verso la rete mondiale IPv4

Il Gateway IPv6 deve essere configurato in modo da avere un'interfaccia che comunica con la rete Internet IPv4 in modo da poter raggiungere la controparte del tunnel posta presso il TILab.

L'indirizzo pubblico IPv4 assegnato al Gateway IPv6 è `193.206.71.142`

Il comando per assegnarlo all'interfaccia di rete è:

```
$ ifconfig rl0 193.206.71.142 netmask 255.255.255.224
```

Come è possibile vedere dallo schema sopra riportato il laboratorio fa parte di una rete locale, il cui punto di accesso verso Internet è un router il cui indirizzo è `193.206.71.129`

L'interfaccia del router IPv4 deve essere impostato come *default gateway* nella tabella di routing con il comando:

```
$ route add default 193.206.71.129
```

Una volta eseguiti questi due comandi il computer fa parte della rete Internet, per cui è bene testare la connessione alla rete provando un comando di `ping` verso un indirizzo pubblico conosciuto:

```
$ ping  www.unipv.it
```

b) Configurare il tunnel

FreeBSD dispone di un interfaccia virtuale `gif` da utilizzare per la creazione di tunnel,

incapsulando dati IPv4 in IPv4 (ad esempio per la creazione di VPN) piuttosto che dati IPv6 in IPv6 o IPv4. Per la configurazione del nostro tunell verso il CdC di Pavia dovremo prima di tutto creare l'interfaccia virtuale del tunnel con il comando:

```
$ ifconfig gif0 create
```

E' possibile quindi verificare che sia stata creata digitando:

```
$ # ifconfig gif0
gif0: flags=8010<POINTOPOINT,MULTICAST> mtu 1280
```

Occorre a questo punto configurare il tunnel, che ricordiamo incapsulerà IPv6 in IPv4, impostando gli indirizzi IPv4 dei due endpoint:

```
$ ifconfig gif0 tunnel <ipv4_locale> <ipv4_remoto>
```

Una volta stabilito e attivato un canale capace di incapsulare il protocollo IPv6, sarà possibile assegnare al capo del tunnel che ci compete l'indirizzo IPv6 previsto (al quale punteranno tutte le regole in Internet per l'inoltro del traffico verso la nostra rete)

```
$ ifconfig gif0 inet6 alias <indirizzo_IPv6_endpoint_tunnel>
```

c) Definire la una tabella di Routing

Sul Router deve essere infine definita una tabella di routing capace di instradare verso il tunnel tutte le comunicazioni che sono rivolte all'esterno con il comando:

```
$ route add –inet6 default –interface gif0
```

In questo modo si crea una regola di default verso il tunnel stesso (questi, una volta instaurato, viene infatti visto come fosse un'interfaccia) per tutto il traffico IPv6.

E' fondamentale ricordare come questa macchina, fungendo da router, debba necessariamente avere il forwading abilitato.

10 - INFORMAZIONI AGGIUNTIVE SU IFCONFIG

Un riferimento molto striminzito alle opzioni di `ifconfig` si ottiene digitando
`$ ifconfig –help`
La pagina di manuale è consultabile in qualsiasi momento digitando:
`$ man ifconfig`
anche se molto spesso risultano più aggiornate le pagine di documentazione GNU/Info, navigabili con il comando:
`$ info ifconfig`

FAMIGLIE DI INDIRIZZI

Durante l'assegnazione di un'indirizzo ad una interfaccia è necessario anche specificare la tipologia dell'indirizzo stesso; l'opzione di default è `inet` (IPv4) ma è possibile specificare indirizzi `inet6` (IPV6), `atalk` (AppleTalk, usato su reti Machintosh), `ipx` (IPX/SPX Microsoft), `link` (MAC-address di livello 2), `lladdr`…

In pratica per assegnare un MAC address diverso da quello del dispositivo:
`# ifconfig rl0 link <MAC_Addr>`

ARP

Le opzioni `arp` e -`arp` servono per attivare o disattivare l'utilizzo di protocolli ARP per la risoluzione degli indirizzi MAC. Nelle Ethernet ARP è necessario, altrimenti si dovrebbero specificare manualmente tutte le voci dell'ARP-table con il comando `arp`.

PROMISCUOUS

Normalmente il sistema analizza solo i pacchetti che sono direttamente indirizzati al suo indirizzo IP. Come detto, se utilizziamo un HUB, ogni pacchetto è trasmesso su tutte le interfacce. Nella modalità *promiscuous* è possibile ricevere anche i pacchetti che non sono stati indirizzati a noi.
Si attiva nel seguente modo:
`$ ifconfig rl0 promisc`
e si disattiva nel seguente modo:
`$ ifconfig rl0 -promisc`
In altre implementazioni, `promisc` è simile ad `allmulti`.

DRIVER MODE

Serve per impostare la modalità di funzionamendo del driver; nelle lan wireless, ad esempio, può essere '`802.11a`', '`802.11b`', '`802.11g`'.

STATISTICHE DI FUNZIONAMENTO

`$ ifconfig -m` visualizza i tipi di link e le modalità di funzionamento supportate dalle periferiche.

11 - ALTRI ARGOMENTI NON TRATTATI IN QUESTO MANUALE

Su Internet si trovano la maggior parte dei documenti informativi riguardanti questo argomento.

Per ulteriori informazioni sulle specifiche dei protocolli di rete:

http://www.ietf.org
http://www.ietf.org/rfc
http://www.faqs.org
http://www.ieee.org
http://www.iana.org
http://www.6bone.net
http://www.6net.org
http://www.tuttoreti.com
http://www.mrtg.org
http://dast.nlanr.net

Per documentazione e notizie riguardanti FreeBSD o applicativi funzionanti sotto questo sistema:

http://www.freebsd.org
http://www.gufi.org
http://www.gnu.org
http://www.google.com/bsd
http://bsd.slashdot.org
http://sourceforge.net
http://www.freebsd.org/ports/ipv6.html

Inoltre il manuale ufficiale di riferimento per ogni aspetto inerente l'utilizzo e la configurazione di FreeBSD è il "FreeBSD Handbook" (*http://www.freebsd.org/handbook/*), disponibile gratuitamente in rete.

Siti IPv6-enabled

www.ipv6forum.com
www.kame.net
www.6bone.net
www6.caspur.it
www6.ipv6.polito.it
www.freebsd.org

DNS utilizzabili

IPv4 – `151.1.1.1`
IPv6 – `2001:760:2:0::a`