Diald Howto

Andrés Seco *AndresSH@ctv.es*

v1.03, April 17, 2000

Questo documento mostra alcuni possibili scenari per cominciare ad usare facilmente *Diald*; questi includono una connessione da un computer isolato ad un ISP utilizzando PPP con un modem senza utilizzare `pon/poff` o `ppp-on/ppp-off`, ad un server proxy/firewall con connessioni internet differenti attraverso vari ISP.

Contents

1 Introduzione

1.1 Obbiettivi

Questo documento mostra alcuni possibili scenari per cominciare ad usare facilmente *Diald*; questi includono una connessione da un computer isolato ad un ISP utilizzando PPP con un modem senza utilizzare `pon/poff` o `ppp-on/ppp-off`, verso un server proxy/firewall con connessioni internet differenti attraverso vari ISP.

Nel presente documento verranno esaminati i seguenti scenari:

- Connessione di un computer isolato verso un ISP utilizzando un modem e PPP.
- Connessione di un computer verso un gruppo di ISP differenti con un modem e PPP.
- Connessione di un server proxy/firewall verso un ISP utilizzando un modem e PPP

Nelle versioni successive di questo documento verranno aggiunti altri scenari come l'utilizzo di sessioni multiple di *Diald*, linee ISDN e linee utilizzate per ricevere e comporre chiamate.

Prima di questo documento esiste un Diald-mini-howto, scritto da Harish Pillay *h.pillay@ieee.org* , che presentava una esempio per una connessione ad un ISP utilizzando un sistema di autenticazione basato si chat (login e password prima della partenza di pppd senza utilizzare PAP o CHAP).

In questo documento sono inclusi alcuni esempi di file di configurazione che possono essere utili come punto di partenza per far funzionare *Diald*. Per ottenere le massime prestazioni e tutte le potenzialità dei programmi è necessario leggere la documentazione allegata ai programmi stessi e riconfigurare i file di esempio inclusi nel presente documento. Inoltre i file di configurazione menzionati possono trovarsi in directory differenti a seconda di quale distribuzione GNU/Linux venga utilizzata. Se i file si trovassero in directory diverse da quelle menzionate qui, siete pregati di scrivermi.

1.2 Nuove versioni

L'ultima versione di questo documento la potete trovare nella mia web page *http://www.ctv.es/USERS/andressh/linux* , sia in formato SMGL che HTML. Altre versioni e formati sono disponibili in spagnolo al sito *http://www.insflug.org/documentos/Diald-Como/* , e in altri linguaggi sul sito LDP - Linux Documentation Project, *http://www.linuxdoc.org* .

1.3 Ringraziamenti

Vorrei ringraziare le persone che mi hanno aiutato a far funzionare il mio primo *Diald* con i loro file di esempio (qualcuno di cui ho dimenticato il nome, Mr Cornish Rex, Hoo Kok Mun e John Dalbec), le persone che mi hanno scritto per segnalare correzioni e suggerimenti nel presente documento (Tim Coleman, Jacob Joseph, Paul Schmidt e Jordi Mallach), i futuri traduttori dello stesso, e ovviamente tutte le persone che hanno sviluppato e sviluppano *Diald* per noi.

Questo documento è stato scritto originariamente in spagnolo. L'autore stesso lo ha tradotto e altri hanno apportato qualche correzione.

2 Copyright and discharge of responsibility

(Per motivi legati alla validità della licenza le note di copyright devono essere mantenute in lingua originale)

This document is `Copyright © 2000 Andres Seco`, and it's free. You can distribute it under the terms of the **GNU General Public License**, which you can get at *http://www.gnu.org/copyleft/gpl.html* . You can get unofficial translated issues somewhere in the Internet.

Information and other contents in this document are the best of our knowledge. However, we may have made errors. So you should determine if you want to follow the instructions given in this document.

Nobody is responsible for any damage to your computer and any other loss derived from the use of the information contained herein.

THE AUTHOR AND MAINTAINERS ARE NOT RESPONSIBLE FOR ANY DAMAGE INCURRED DUE TO ACTIONS TAKEN BASED ON INFORMATION CONTAINED IN THIS DOCUMENT.

ovvero:

Questo documento è `Copyright © 2000 Andres Seco` ed è gratuito. Può essere distribuito rispettando le condizioni della **Licenza pubblica GNU**, che può essere scaricata all'indirizzo *http://www.gnu.org/copyleft/gpl.html* . È possibile ottenere copie tradotte non ufficiali in internet.

Le informazioni contenute in questo documento sono il meglio della nostra conoscenza. Comunque è possibile che ci siano errori. Si deve essere sicuri di voler seguire le istruzioni contenute nel documento. Nessuno può essere considerato responsabile per eventuali danneggiamenti o perdite conseguenti all'utilizzo delle informazioni contenute in questo documento.

L'AUTORE E I CURATORI NON SONO RESPONSABILI PER QUALUNQUE DANNO APPORTATO DA AZIONI INTRAPRESE SULLA BASE DELLE INFORMAZIONI CONTENUTE IN QUESTO DOCUMENTO.

Ovviamente sono disponibile a tutti i suggerimenti e correzioni riguardanti il contenuto del documento.

3 Breve descrizione del funzionamento di Diald

In poche parole, *Diald* crea una nuova interfaccia e la imposta come gateway predefinito. Questa interfaccia non è reale (nella documentazione originale è detta `proxy interface`). *Diald* controlla tale interfaccia e, quando un pacchetto arriva, crea una connessione `ppp`, aspetta perché venga stabilita e cambia il gateway predefinito a questa nuova interfaccia `ppp` (normalmente `ppp0`).

Diald controlla l'interfaccia in modo da determinare quali e che tipo di pacchetti siano stati ricevuti in modo da decidere se devono essere considerati per attivare l'interfaccia `ppp`, mantenere il collegamento, chiuderlo, non fare nulla o per quanto tempo il collegamento debba essere mantenuto dopo la trasmissione del pacchetto.

Alla fine, se non c'è più traffico e il tempo stabilito è passato dalla ricezione dell'ultimo pacchetto *Diald* chiude il collegamento.

È possibile controllare il giorno e l'ora in cui il collegamento debba o non debba essere stabilito, così è possibile utilizzare le ore o i giorni a basso costo o a basso traffico.

La descrizione precedente è valida per versioni di *Diald* dalla 0.16.5 alla più recente (0.99.3 quando questo documento è stato finito), ma le ultime versioni includono anche caratteristiche addizionali come le "user enable list", accounting avanzato, un supporto migliore per le linee ISDN, prestazioni migliori quando si utilizza un dispositivo `ethertap` come proxy (questa è un'interfaccia di rete che legge/scrive su un socket invece di una vera scheda di rete) al posto di `slip`, connessioni di backup e altre funzioni.

4 Note circa l'autenticazione

Quando ci si connette ad un Internet Service Provider, è generalmente necessario mandare uno username e una password. Questo si può fare utilizzando molti modi diversi; il metodo da utilizzare viene stabilito dal provider.

In aggiunta alle tre opzioni mostrate, si può utilizzare una connessione senza autenticazione (per esempio quando l'estremità remota è sempre vostra).

4.1 Username e password - prompt per Login e password

In realtà questo non è un metodo comune per accedere ad internet attaverso un ISP.

L'identificazione viene fatta prima che `pppd` venga attivato e, di solito, è il chiamante che spedisce login e password. I dati vengono spediti in chiaro e quindi questo metodo non è da considerarsi sicuro.

Un esempio di script per `chat` dove si può vedere come specificare username e password per spedirli prima di caricare `pppd` sarebbe simile a qualcosa del genere:

```
ABORT BUSY
ABORT "NO CARRIER"
ABORT VOICE
ABORT "NO DIALTONE"
ABORT "NO ANSWER"
"" ATZ
OK ATDT_TelephoneNumber_
CONNECT \d\c
ogin _Username_
assword _Password_
```

Le ultime due righe definiscono lo username e la password e quando spedirli (dopo aver ricevuto «ogin» e «assword» rispettivamente). Lo script chat ha solo bisogno di vedere parte delle parole «login» e «password» e quindi non abbiamo bisogno di controllare la prima lettera di ciascuna. In questo modo non abbiamo bisogno di preoccuparci delle maiuscole o minuscole.

Supponiamo che questo script sia chiamato `provider` e che sia salvato nella directory `/etc/chatscripts`. È quindi possibile lanciarlo con:

```
/usr/sbin/chat -v -f /etc/chatscripts/provider
```

4.2 PAP - Password Authentication Protocol

Se il provider che si sta utilizzando richiede PAP come protocollo di autenticazione, allora durante la negoziazione LCP del PPP, a questo sarà richiesto di utilizzare tale protocollo. Quando si è connessi, dopo aver usato `chat`, viene lanciato `pppd`. In questo caso, pppd si occuperà di mandare lo username e la password che troverà nel file `/etc/ppp/pap-secrets`. Questo file deve avere i permessi in scrittura e lettura solo per `root` in modo che nessun altro possa leggere la password scritta all'interno.

PAP non è molto sicuro e la password viene trasmessa in chiaro così che potrebbe essere letta da chiunque stia monitorando la linea di trasmissione.

Questo è un semplice esempio di `/etc/ppp/pap-secrets`:

```
_Username_ * _Password_
```

4.3 CHAP - Challenge Authentication Protocol

Se il provider che si sta utilizzando richiede CHAP come protocollo di autenticazione, allora durante la negozioazione LCP nel PPP, questo verrà richiesto di utilizzare tale protocollo. Quando si è connessi, dopo aver usato `chat`, viene lanciato `pppd`. In questo caso, pppd si occuperà di mandare lo username e la password che troverà nel file `/etc/ppp/chap-secrets`. Questo file deve avere i permessi in scrittura e lettura solo per `root` in modo che nessun altro possa leggere la password scritta all'interno.

CHAP è più sicuro di PAP in quanto la password non viene spedita in chiaro sulla linea di trasmissione. Il server per l'autenticazione spedisce un identificatore casuale (il challenge), che il client deve criptare con la sua password e quindi spedire tutto indietro al server. Questo è un semplice esempio di `/etc/ppp/chap-secrets`:

```
_Username_ * _Password_
```

Quelche volta un provider utilizza PAP, altre volte CHAP, quindi è normale definire la propria username e password in entrambi i file.

5 Note circa la risoluzione dei nomi tramite DNS

Ogni volta che ci si connette ad un ISP, è necessario aver configurato la risoluzione dei nomi tramite DNS in modo che il proprio computer possa trovare gli indirizzi IP corrispondenti ai nomi dei computer.

Gli indirizzi IP dei server DNS vengono scritti nel file `/etc/resolv.conf`. In un computer isolato che si connette ad Internet questo file di solito contiene gli indirizzi IP dei DNS del proprio ISP:

```
#file /etc/resolv.conf per nomeISP
nameserver 111.222.333.444
nameserver 222.333.444.555
```

In un computer con proxy/firewall, questo file, in genere, contiene il proprio indirizzo IP (o l'indirizzo di loopback 127.0.0.1) e sul computer è attivo un server DNS che traduce i nomi in indirizzi IP interrogando un DNS esterno.

```
#file /etc/resolv.conf per risoluzione DNS locale
nameserver 127.0.0.1
```

L'installazione di un DNS locale è al di là degli scopi di questo documento. Esiste molta documentazione circa questo argomento, ma un approccio veloce e buono può essere trovato nel DNS-Howto(*http://www.linuxdoc.org/HOWTO/DNS-HOWTO.html*).

6 Connessione di un computer isolato ad un ISP utilizzando un modem e PPP

Quando si configura *diald* per connettersi con il proprio computer ad un ISP, è necessario operare come segue:

- Avere installato il pacchetto *Diald*. Il metodo più veloce è quello di installare il pacchetto contenuto nella propria distribuzione GNU/linux.
- Configurare il file per la risoluzione dei nomi (`/etc/resolv.conf`).
- Controllare di potersi connettere all'ISP. Se la propria distribuzione consente l'utilizzo di una utility per configurare la connessione, il metodo più veloce consiste nell'utilizzarla (pppconfig per Debian, kppp se si usa kde ecc.). Se si incontrano problemi per stabilire la connessione, può essere di aiuto il PPP-Howto (*http://www.linuxdoc.org/HOWTO/PPP-HOWTO.html*), Modem-Howto (*http://www.linuxdoc.org/HOWTO/Modem-HOWTO.html*) and Serial-Howto (*http://www.linuxdoc.org/HOWTO/Serial-HOWTO.html*).
- Configurare lo username e la password nei file `/etc/ppp/pap-secrets` e `/etc/ppp/chap-secrets` menzionati nella sezione precedente.

E infine configurare *Diald*:

- Preparare il file di configurazione di *Diald* (`/etc/diald/diald.options` per la versione 0.16.5 and `/etc/diald/diald.conf` per le successive).
- Preparare il file filtro `/etc/diald/standard.filter`, o meglio, lasciare quel file così com'è e modificare una copia di esso che può essere chiamato `/etc/diald/personal.filter`.
- Preparare lo script per connettersi (`/etc/diald/diald.connect` con i permessi di esecuzione per root) e il file con le istruzioni per `chat` (`/etc/chatscripts/provider`), che verrà utilizzato dallo script precedente.
- Preparare gli script che verranno eseguiti quando la connessione viene stabilita o interrotta (`/etc/diald/ip-up` e `/etc/diald/ip-down`) nel caso si voglia usarli (entrambi dovranno avere i permessi per l'esecuzione per root)
- Preparare gli script per impostare o cancellare le tabelle per l'instradamento (`/etc/diald/addroute` e `/etc/diald/delroute`) se lo si vuole (entrambi dovranno avere i permessi per l'esecuzione per root). Quasto passo non è necessario se si usa una singola istanza di *Diald*.
- Infine far partire il demone *diald* («`/etc/init.d/diald start`» con Debian, «`/etc/rc.d/init.d/diald start`» con RedHat). Generalmente, il processo di installazione del pacchetto *Diald* prepara lo script per attivare *Diald* durante il boot del computer nella directory /etc/rcX.d.

Se vengono fatti dei cambiamenti al file di configurazione quando *diald* è attivo, è necessario farlo ripartire («`/etc/init.d/diald restart`» con Debian, «`/etc/rc.d/init.d/diald restart`» con RedHat).

6.1 Il file /etc/diald/diald.options o diald.conf

In questo file di esempio bisogna controllare:

- La porta com a cui è connesso il modem. Opzione `device`.
- La velocità della porta com con cui dialogare con il modem. Opzione `speed`.
- Lo username da usare con ppp. Opzione `pppd-options`.
- I contatori per i tentativi e i timer.
- Le ore abilitate alla connessione. Opzione `restrict`.

- L'indicazione per utilizzare gli script `ip-up` e `ip-down`. Opzione `ip-up` e `ip-down`.
- L'indicazione per utilizzare gli script `addroute` e `delroute`. Opzione `addroute` e `delroute`. Generalmente non è necessario modificare tali opzioni, ma se si usano più istanze di *Diald* oppure si ha una configurazione complessa, probabilmente è necessario.
- L'indicazione per l'uso del file dei filtri standard o quello personale. Opzione `include`.

```
##########################
# /etc/diald/diald.options

# Dispositivo a cui è connesso il modem
device /dev/ttyS0

# File di log
accounting-log /var/log/diald.log

# Monitoring queue
#fifo /var/run/diald/diald.fifo

# Attivazione debug
# Attivare il debug riduce le prestazioni
#debug 31

# Usiamo PPP come incapsulatore
mode ppp

# IP locale (quando si è connessi questo indirizzo viene
# automaticamente modificato dall'ISP se si utilizza l'opzione per l'IP
# dinamico)
local 127.0.0.5

# IP remoto (quando si è connessi questo indirizzo viene
# automaticamente cambiato nell'IP del server remoto che riceve la
# chiamata)
remote 127.0.0.4

# Maschera di sottorete per la connessione wan
netmask 255.255.255.0

# L'indirizzo IP viene assegnato quando si stabilisce la connessione
dynamic

# Se la connessione viene chiusa dalla parte remota, ristabilisci la
# connessione solo se ci sono pacchetti uscenti
two-way

# Quando la connessione è attiva, indirizza direttamente
# all'interfaccia ppp reale e non all'interfaccia proxy. Non fare questo
# riduce le prestazioni del 20%. Ci sono vecchi kernel che non
# supportano il reroute (reindirizzamento). Consultare il manuale di diald
# per ulteriori informazioni.
reroute

# Diald assegna l'instradamento di default all'interfaccia SLIP usata
```

```
# come proxy
defaultroute

# Script per personalizzare l'instradamento
#addroute "/etc/diald/addroute"
#delroute "/etc/diald/delroute"

# Script da eseguire quando la connessione è su e pronta oppure quando
# è giù e chiusa. Nella versione 0.9x c'è anche un'altra opzione
# ip-goingdown che può essere utilizzata per utilizzare comandi
# quando la connessione sta per essere chiusa ma è ancora su.
ip-up /etc/diald/ip-up
#ip-down /etc/diald/ip-down

# Script usati per connettere o disconnettere l'interfaccia
connect "/etc/diald/diald.connect"
#disconnect "/etc/diald/diald.disconnect"

# Usere il lock UUCP per segnalare che il dispositivo è utilizzato
#lock

# Ci si connette con un modem. Attenzione: Non specificare queste
# opzioni nel file options di ppp perché creeranno conflitto con quelle
# usate da diald. Per sapere quali opzioni di ppp non possono essere
# usate nel file options leggere la man page di diald e cercare pppd-options.
modem
crtscts
speed 115200

# Alcuni timer e opzioni per la ripetizione dei tentativi
# riferirsi alla man page di diald per maggiori informazioni
connect-timeout 120
redial-timeout 60
start-pppd-timeout 120
died-retry-count 0
redial-backoff-start 4
redial-backoff-limit 300
dial-fail-limit 10

# Opzioni da passare a pppd
# Queste opzioni possono essere incluse nel file /etc/ppp/options che
# contiene le opzioni di default per pppd, ma se si ha bisogno di
# utilizzare configurazioni differenti di diald per più di una istanza,
# è necessario metterle qui.
# noauth - non richiedere al remoto di autenticarsi
# user   - lo username. chiedere all'isp la sintassi, alcuni non
# richiedono la terminazione @isp.
pppd-options noauth user user@isp

# Le restrizioni
# Questa sezione deve stare prima dei filtri
# Il comando restrict è sperimentale e può variare in altre versioni
# di diald. Controllare la man page. (questo esempio è stato controllato
# con la versione 0.16, ma penso che funzioni anche con versioni
# successive).
```

```
# Per esempio: usare solo di notte dal lunedì al venerdì e tutto il
# giorno di sabato e domenica.
restrict 8:00:00 18:00:00 1-5 * *
down
restrict * * * * *

# Non ci sono considerazioni speciali sulle tariffe.
# (i primi secondi sono inclusi nel costo iniziale, l'unità della
# tariffa è il secondo, tempo in secondi per controllare se è il caso di
# chiudere)
#impulse 0,0,0
# Bononet Noche (Spain-Telefónica) viene pagata in secondi dopo i
# primi 160 secondi
impulse 160,0,0
# se fosse pagata in minuti venisse sempre pagata dopo i primi 10 minuti
#impulse 600,60,10

# Filtri standard
#include /etc/diald/standard.filter
# o filtri personali
include /etc/diald/personal.filter
```

6.2 Il file dei filtri personali

Le modifiche a questo file devono essere fatte con molta attenzione. Questo file viene utilizzato per decidere quando e come stabilire la connessione, mantenerla, chiuderla o ignorare un pacchetto a seconda del tipo di traffico.

Generalmente il file dei filtri standard di *Diald* è sufficiente per la maggior parte dei casi, ma forse potrebbe essere troppo restrittivo o non abbastanza in certe situazioni. Il file `personal.filter` che viene riportato mostra alcune modifiche rispetto all'originale della versione 0.16.

Nelle prossime versioni di questo documento verranno inclusi esempi commentati maggiormente restrittivi.

```
# /etc/diald/personal.filter
# Le regole per i filtri mostrate qui sono le stesse del file
# standard.filter con i  seguenti cambiamenti:

# Modificato da 10 a 4 minuti in "any other tcp conectio".
# Aggiunto "ignore tcp tcp.fin" per ignorare il FIN ACK packet.
# Ignorare i pacchetti icmp (ping e traceroute non fanno partire
# l'interfaccia).


# Questo è un insieme di regole decisamente complicato.
# (questo set è quello che uso io)
# Ho diviso le regole in 4 sezioni.
# Pacchetti TCP, UDP, ICMP e una serie di regole generali alla fine.

ignore icmp any

#-------------------------------------------------------------------------------
# Regole per pacchetti TCP.
#-------------------------------------------------------------------------------
# Commenti generali sull'insieme di regole:
```

```
#
# In generale, si vorrebbero trattare solamente i dati sul collegamento TCP come
# significativi per i timeout. Quindi si cerca di ignorare i pacchetti
# che non contengono dati. Dal momento che l'insieme di header più breve
# in un pacchetto TCP/IP è di 40byte, ogni pacchetto di lunghezza 40 non
# può contenere dati.
# Potremmo perdere alcuni pacchetti vuoti in questa maniera (informazioni
# di instradamento o altri extra potrebbero essere contenuti nell'header
# IP), ma dovremmo comunque intercettarne la maggior parte. Da notare che non
# vogliamo filtrare i pacchetti con tcp.live non settato, dal momento
# che possiamo utilizzarli per accelerare la disconnesione su certi link TCP.
#
# Vogliamo inoltre essere sicuri che i pacchetti WWW sopravvivano anche
# se il socket TCP è chiuso. Facciamo così perchè WWW non mantiene aperta
# la connessione ona volta che i dati sono stati trasferiti e sarebbe
# molto scomodo avere una connessione che continua ad andare su e giù
# ogni volta che si scarica un documento.
#
# al di là del WWW l'uso più comune del TCP rimane quello per le
# connessioni su tempi lunghi, le chiusura delle quali significa che non
# si ha più bisogno della connessione in rete. Non vogliamo aspettare 10
# minuti perché la connesione venga chiusa quando non abbiamo telnet o
# rlogin che stanno girando, quindi vogliamo accelerare il timeout sulle
# connessioni TCP che sono state chiuse. riusciamo ad ottenerlo
# catturando i pacchetti che non hanno il flag live settato.

# --- inizio dell'insieme di regole ---

# Quando iniziamo una connessione le concediamo, inizialmente, solo 15
# secondi. L'idea è quella di avere a che fare con la possibilità che la
# rete all'etremità opposta della connessione non sia raggiungibile. In
# questo caso non è il caso di dare un tempo di vita di 10 minuti alla
# connessione. Quindi, con la regola più sotto diamo solo 15 secondi
# alla connessione iniziale. Se la rete è raggiungibile normalmente
# verrà ricevuta una risposta prima di 15 secondi. Se questo causa
# qualche problema nel caso di risposte perticolarmente lente da qualche
# sito a cui si accede normalmente, si può rimuovere il timeout oppure
# la regola stessa.
accept tcp 15 tcp.syn

# Evitiamo che gli xfer di named mantengano il link attivo
ignore tcp tcp.dest=tcp.domain
ignore tcp tcp.source=tcp.domain

# (Ack! Il telnet di SCO comincia mandando dei SYN vuoti e stabilisce
# la connessione solamente se riceve una risposta.)
accept tcp 5 ip.tot_len=40,tcp.syn

# Evitiamo che pacchetti vuoti mantengano su la connessione (pacchetti
# diversi da SYN vuoti)
ignore tcp ip.tot_len=40,tcp.live

# Una modifica di Andres Seco per ignorare i pacchetti FIN ACK.
ignore tcp tcp.fin
```

```
# Facciamo in modo che i trasferimenti http mantengano la connessione
# per 2 minuti anche dopo che sono finiti.
# NOTA: Il file /etc/services potrebbe non definire il servizio www tra
# quelli tcp, nel qual caso si devono commentare le seguenti righe
# oppure ottenere un file /etc/services aggiornato. Leggere le FAQ per
# informazioni su come ottenere un nuovo /etc/services.
accept tcp 120 tcp.dest=tcp.www
accept tcp 120 tcp.source=tcp.www
# Stessa cosa per http
accept tcp 120 tcp.dest=tcp.443
accept tcp 120 tcp.source=tcp.443

# Una volta che la connessione non è più attiva cerchiamo di chiudere
# la connessione velocemente. Notare che se la connessione è giù, un
# cambiamento di stato non la riattiverà.
keepup tcp 5 !tcp.live
ignore tcp !tcp.live

# Un ftp-data o una connessione ftp potrebbe essere considerato un
# traffico ragionevolmente frequente.
accept tcp 120 tcp.dest=tcp.ftp
accept tcp 120 tcp.source=tcp.ftp

# NOTA: L'ftp-data non è definito nel file /etc/services che viene
# installato con le ultime versioni di netkit, quindi è meglio
# commentare le regole seguenti. E' possibile definire il servizio
# aggiungendo le seguenti righe nel file /etc/services:
# ftp-data        20/tcp
# e levando il commento alle seguenti righe
#accept tcp 120 tcp.dest=tcp.ftp-data
#accept tcp 120 tcp.source=tcp.ftp-data

# Se non rientra nei casi precedenti, diamo alla connesione 10 minuti di
# tempo.
#accept tcp 600 any
# Modificacion de Andres Seco. Solo dejar 4 minutos mas.
accept tcp 240 any

#-----------------------------------------------------------------------------
# Regole per pacchetti UDP
#-----------------------------------------------------------------------------

# Mandiamo direttamente in timeout le richieste al nameserver,
# vogliamo che facciano attivare la connessione non mantenerla su a lungo.
# Questo serve perché, di solito, la connessione verrà attivata da una
# chiamata alla libreria del resolver (almeno che non si abbiano gli
# indirizzi frequentemente utilizzati in /etc/hosts nel qual caso
# sorgerebbero altri problemi).

Da notare che non si dovrebbe rendere il timeout più breve del tempo
che ci si apetta che il server DNS impieghi a rispondere. Altrimenti
quando la connessione iniziale verrebbe stabilita potrebbe esserci un
ritardo più grande del timeout tra la serie di pacchetti iniziale e prima
che passi per essa un pacchetto che la mantenga su.
```

```
# Non attivare la connessione per rwho
ignore udp udp.dest=udp.who
ignore udp udp.source=udp.who
#Non attivare la connessione per RIP
ignore udp udp.dest=udp.route
ignore udp udp.source=udp.route
# Non attivare la connessione per NTP o timed
ignore udp udp.dest=udp.ntp
ignore udp udp.source=udp.ntp
ignore udp udp.dest=udp.timed
ignore udp udp.source=udp.timed
# Non attivare la connessione su richieste al name server tra due
# named attivi
ignore udp udp.dest=udp.domain,udp.source=udp.domain
# Tiriamo su la rete ogni volta che tentiamo una richiesta di dominio da qualche
# altro posto rispetto a named.
accept udp 30 udp.dest=udp.domain
accept udp 30 udp.source=udp.domain
# Facciamo lo stesso per i broadcast del netbios-ns
# NOTA: il file /etc/services potrebbe non definire il servizio
# netbios-ns, nel qual caso si dovrebbero commentare le seguenti 3 righe
ignore udp udp.source=udp.netbios-ns,udp.dest=udp.netbios-ns
accept udp 30 udp.dest=udp.netbios-ns
accept udp 30 udp.source=udp.netbios-ns
# impediamo ai trasferimenti di routed e gated di tenere su il link
ignore udp tcp.dest=udp.route
ignore udp tcp.source=udp.route
# Ogni altra cosa prende 2 minuti
accept udp 120 any

# Catturiamo ogni pacchetto che non abbiamo già esaminato e diamo alla
# connessione 30 secondi di vita.
accept any 30 any
```

6.3 Fare la chiamata

Il file `/etc/diald/diald.connect` (deve avere i permessi in esecuzione):

```
/usr/sbin/chat -f /etc/chatscripts/provider
```

Il file `/etc/chatscripts/provider`. In questo esempio bisogna mettere il numero di telefono da chiamare.

```
ABORT BUSY
ABORT "NO CARRIER"
ABORT VOICE
ABORT "NO DIALTONE"
ABORT "NO ANSWER"
"" ATZ
OK ATDT123456789
CONNECT \d\c
```

6.4 Lo script per la connessione

Deve avere i permessi di esecuzione.

Questo script può essere utilizzato per compiti diversi (tempo di sincronizzazione, spedire la posta in coda, scaricarla ecc.).

Nell'esempio viene spedito un messaggio a `root` con i dati passati allo script (interfaccia, maschera di sottorete, indirizzo ip locale, indirizzo ip remoto e il costo di indirizzamento):

```
#!/bin/sh

iface=$1
netmask=$2
localip=$3
remoteip=$4
metric=$5

# Spedisci ora e data
# netdate ntp.server.somecountry

# Spedisci la posta in coda
# runq

echo `date` $1 $2 $3 $4 $5 | mail -s "diald - conecting" root@localhost
```

7 Connessione di un computer ad un gruppo di ISP differenti con un modem e il PPP.

Molte volte un computer isolato non viene connesso ad una sola rete. È invece più comune che venga connesso a reti differenti o ad Internet utilizzando diversi service provider. In questo caso cambiare i file di configurazione ogni volta che ci si vuole connettere ad un sito differente può essere noioso.

La soluzione che propongo qui consiste nell'utilizzare differenti insiemi di file di configurazione per ogni connessione. Qui si possono anche trovare degli script per cambiare automaticamente dall'una all'altra.

7.1 Nota a proposito della spedizione di mail utilizzando un relay host.

Se il client email utilizzato utilizza un MTA (message tranfer agent) con un relay host `smtp` per spedire tutti i messaggi, o si utilizza un client email che spedisce i messaggi direttamente al server `smtp` del provider, cambiare connessione significa cambiare anche le opzioni per il relay del server `smtp`. Questo succede perché i provider, in genere, controllano se la mailbox in ricezione è locale, appartiene ad un dominio mantenuto direttamente dal provider stesso oppure se l'indirizzo originale è nell' intervallo di indirizzi che il provider assegna, per evitare di avere un relay server aperto che potrebbe essere utilizzato per spedire spam, messaggi anonimi e così via.

Negli esempi seguenti, è possibile trovare il modo in cui cambiare questo parametro nei file di configurazione di *Smail* con una configurazione molto semplice in cui tutti i messaggi sono spediti ad un relay server `smtp` esterno. Se si utilizza un altro Message Transfer Agent (MTA) nel sistema, è possibile spedirmi i cambiamenti necessari perché vengano inclusi qui. Se si utilizza un client email che spedisce direttamente al server `smtp` (Kmail, Netscape, etc.), speditemi i cambiamenti anche in questo caso.

7.2 Alcuni script per automatizzare connessioni multiple e cambiare da una all'altra.

7.2.1 Cominciare

Prima di tutto, bisogna creare una sottodirectory di `/etc/diald` chiamata `providers` dove è possibile tenere i vari script per cambiare da un provider all'altro e le varie sottodirectory con gli insiemi di file per la configurazione di ogni connessione al provider.

Con il prossimo script è possibile creare questa directory e riempirla con i file di configurazione da *Diald*, *chat*, *pppd* e *Smail*, che verrà utilizzato per le successive configurazioni.

```
#!/bin/sh
#File /etc/diald/providers/setupdialdmultiprovider
mkdir /etc/diald/providers
mkdir /etc/diald/providers/setup
cp /etc/ppp/pap-secrets /etc/diald/providers/setup
cp /etc/ppp/chap-secrets /etc/diald/providers/setup
cp /etc/resolv.conf /etc/diald/providers/setup
cp /etc/diald/diald.options /etc/diald/providers/setup
cp /etc/diald/standard.filter /etc/diald/providers/setup
cp /etc/diald/personal.filter /etc/diald/providers/setup
cp /etc/diald/diald.connect /etc/diald/providers/setup
cp /etc/chatscripts/provider /etc/diald/providers/setup
cp /etc/diald/ip-up /etc/diald/providers/setup
cp /etc/diald/ip-down /etc/diald/providers/setup
cp /etc/smail/routers /etc/diald/providers/setup
```

7.2.2 Un nuovo provider

Con il prossimo script la configurazione di esempio viene copiata in una nuova directory per prepararla per un nuovo provider o una nuova connessione in rete. Questo script (`/etc/diald/providers/newdialdprovider`) avrà bisogno di un parametro con il nome del provider o della connessione di rete.

```
#!/bin/sh
#File /etc/diald/providers/newdialdprovider
mkdir /etc/diald/providers/$1
cp /etc/diald/providers/setup/* /etc/diald/providers/$1
```

Ora bisogna modificare i file in `/etc/diald/providers/provdidername`, dove `providername` è il parametro passato allo script `newdialdprovider`.

7.2.3 Cambiare dall'uno all'altro

Infine con questo script sarà possibile cambiare tutti i file di configurazione relativi a *Diald* per connettersi ad un nuovo provider o ad una nuova rete. Utilizzando i link simbolici, quando si cambiano i file di configurazione nella loro locazione originale come `/etc/resolv.conf`, i cambiamenti vengono fatti anche nei file di `/etc/diald/providers/providername/resolv.conf`.

```
#!/bin/sh
#File /etc/diald/providers/setdialdprovider
/etc/init.d/diald stop
```

```
#wait for Diald to stop.
sleep 4
ln -sf /etc/diald/providers/$1/pap-secrets /etc/ppp
ln -sf /etc/diald/providers/$1/chap-secrets /etc/ppp
ln -sf /etc/diald/providers/$1/resolv.conf /etc
ln -sf /etc/diald/providers/$1/diald.options /etc/diald
ln -sf /etc/diald/providers/$1/standard.filter /etc/diald
ln -sf /etc/diald/providers/$1/personal.filter /etc/diald
ln -sf /etc/diald/providers/$1/diald.connect /etc/diald
ln -sf /etc/diald/providers/$1/provider /etc/chatscripts
ln -sf /etc/diald/providers/$1/ip-up /etc/diald
ln -sf /etc/diald/providers/$1/ip-down /etc/diald
ln -sf /etc/diald/providers/$1/routers /etc/smail
/etc/init.d/diald start
```

8 Connettere un proxy/firewall ad un ISP utilizzando un modem e PPP.

Connettere una rete privata ad Internet con un server dedicato che maneggia i pacchetti indirizzandoli dalla rete locale verso internet e facendo funzioni di proxy/caching è un tema complesso che esula dagli scopi di questo documento. Esistono altri «Howto» che trattano la materia in maniera più dettagliata. Alla fine di questo documento è possibile trovare una lista di link e di riferimenti a tali documenti.

Qui, viene spiegato come configurare solo *Diald* supponendo che il computer già utilizza IP-Masquerading, possiede un proxy funzionante come *Squid* o simile, una connessione verso un ISP configurata correttamente e che la sicurezza per l'accesso alle porte TCP/UDP è stata impostata (il file `/etc/inetd.conf`, o altri come `securetty`, `host.allow`, etc).

In linea generale l'unica cosa da fare è riconfigurare le regole per il masquerading/filtering/accessing ogni volta che l'insieme di interfacce cambia, cioè quando l'interfaccia ppp0 viene stabilita e quando viene cancellata. Un buon posto dove farlo sono gli script ip-up e ip-down di *pppd*.

8.1 Un esempio per Debian 2.1

Con Debian è sufficiente installare il pacchetto *ipmasq* rispondendo che si vogliono cambiare le regole in modo sincrono con *pppd* quando lo si attiva. Verranno creati due script detro le directory `/etc/ppp/ip-up.d` e `/etc/ppp/ip-down.d` per chiamare `/sbin/ipmasq` che analizzano l'interfaccia esistente e creano una semplice configurazione che è valida in molti casi e che è personalizzabile utilizzando il file di regole `/etc/ipmasq/rules`.

Le uniche correzioni dopo l'installazione di questo pacchetto sono quelle di cancellare, quando lo script per *ipmasq* viene lanciato, il link simbolico da `/etc/rcS.d` e creandone uno nuovo in `/etc/rc2.d` in modo che venga lanciato dopo `S20diald`. Ora, quando `ipmasq` viene eseguito per analizzare le interfacce, `sl0` già esiste. `S90ipmasq` è un buon nome per il nuovo link simbolico a `/etc/init.d/ipmasq`.

Se si utilizza Debian non c'è da preoccuparsi della versione del kernel visto che lo script `/sbin/ipmasq` utilizza `ipfwadm` o `ipchains` a seconda delle richieste.

8.2 Esempio per Suse 6.1

Questo esempio è di Mr Cornish Rex, *troll@tnet.com.au* .

I seguenti comandi per ip-masq e per il controllo dell'instradamento devono essere utilizzati con le versioni 2.2 del kernel perche', utilizzando ipchains, non sono validi per i kernel 2.0.

Stiamo supponendo che l'interfaccia di rete ha come indirizzo ip 192.168.1.1 con netmask di 16 bit e cioe' 255.255.0.0.

Questo è il file `/etc/ppp/ip-up`:

```
#!/bin/sh
# $1 = Interface
# $2 = Tty device
# $3 = speed
# $4 = local ip
# $5 = remote ip
# $6 = ipparam
/sbin/ipchains -F input
/sbin/ipchains -P input DENY
/sbin/ipchains -A input -j ACCEPT -i eth0 -s 192.168.0.0/16 -d 0.0.0.0/0
/sbin/ipchains -A input -j DENY -p udp -i $1 -s 0.0.0.0/0 -d $4/32 0:52 -l
/sbin/ipchains -A input -j DENY -p udp -i $1 -s 0.0.0.0/0 -d $4/32 54:1023 -l
/sbin/ipchains -A input -j DENY -p tcp -i $1 -s 0.0.0.0/0 -d $4/32 0:112 -l
/sbin/ipchains -A input -j DENY -p tcp -i $1 -s 0.0.0.0/0 -d $4/32 114:1023 -l
/sbin/ipchains -A input -j DENY -p tcp -i $1 -s 0.0.0.0/0 -d $4/32 6000:6010 -l
/sbin/ipchains -A input -j DENY -p icmp --icmp-type echo-request \
-i $1 -s 0.0.0.0/0 -l
/sbin/ipchains -A input -j DENY -p icmp -f -i $1 -s 0.0.0.0/0 -l
/sbin/ipchains -A input -j DENY -p udp -i $1 -s 0.0.0.0/0 -d $4/32 5555 -l
/sbin/ipchains -A input -j DENY -p udp -i $1 -s 0.0.0.0/0 -d $4/32 8000 -l
/sbin/ipchains -A input -j DENY -p tcp -i $1 -s 0.0.0.0/0 -d $4/32 8000 -l
/sbin/ipchains -A input -j DENY -p udp -i $1 -s 0.0.0.0/0 -d $4/32 6667 -l
/sbin/ipchains -A input -j DENY -p tcp -i $1 -s 0.0.0.0/0 -d $4/32 6667 -l
/sbin/ipchains -A input -j DENY -p tcp -i $1 -s 0.0.0.0/0 -d $4/32 4557 -l
/sbin/ipchains -A input -j DENY -p tcp -i $1 -s 0.0.0.0/0 -d $4/32 4559 -l
/sbin/ipchains -A input -j DENY -p tcp -i $1 -s 0.0.0.0/0 -d $4/32 4001 -l
/sbin/ipchains -A input -j DENY -p tcp -i $1 -s 0.0.0.0/0 -d $4/32 2005 -l
/sbin/ipchains -A input -j DENY -p tcp -i $1 -s 0.0.0.0/0 -d $4/32 6711 -l
/sbin/ipchains -A input -j DENY -i $1 -s 192.168.0.0/16 -d 0.0.0.0/0 -l
/sbin/ipchains -A input -j ACCEPT -i $1 -s 0.0.0.0/0 -d $4/32
/sbin/ipchains -A input -j ACCEPT -i lo -s 0.0.0.0/0 -d 0.0.0.0/0
/sbin/ipchains -A input -j DENY -s 0.0.0.0/0 -d 0.0.0.0/0 -l

/sbin/ipchains -F output
/sbin/ipchains -P output DENY
/sbin/ipchains -A output -j ACCEPT -i eth0 -s 0.0.0.0/0 -d 192.168.0.0/16
/sbin/ipchains -A output -j DENY -i $1 -s 192.168.0.0/16 -d 0.0.0.0/0 -l
/sbin/ipchains -A output -j ACCEPT -i $1 -s $4/32 -d 0.0.0.0/0
/sbin/ipchains -A output -j ACCEPT -i lo -s 0.0.0.0/0 -d 0.0.0.0/0
/sbin/ipchains -A output -j DENY -s 0.0.0.0/0 -d 0.0.0.0/0

/sbin/ipchains -F forward
/sbin/ipchains -P forward DENY
/sbin/ipchains -M -S 120 120 120
/sbin/ipchains -A forward -j MASQ -s 192.168.1.0/24
/sbin/ipchains -A forward -j DENY -s 0.0.0.0/0 -d 0.0.0.0/0

exit 0
```

questo è il file `/etc/ppp/ip-down`:

```
#!/bin/sh
# $1 = Interface
# $2 = Tty device
# $3 = Speed
# $4 = Local ip
# $5 = Remote ip
/sbin/ipchains -F input
/sbin/ipchains -F output
/sbin/ipchains -F forward
/sbin/ipchains-restore < /etc/ppp/orig.chains
```

L'ultimo file nell'ultimo script, orig.chains, è il file seguente (stato originale di ipchains):

```
# orig.chains
# created with: ipchains-save > orig.chains
:input ACCEPT
:forward ACCEPT
:output ACCEPT
-A input -s 0.0.0.0/0.0.0.0 -d 192.168.1.1/255.255.255.255
-A output -s 192.168.1.1/255.255.255.255 -d 0.0.0.0/0.0.0.0
```

8.3 Esempio per Slackware 3.6

Questo esempio è di Hoo Kok Mun, *hkmun@pacific.net.sg* .

Questo è l'esempio più semplice che ho visto, ma è pienamente funzionale. All'inizio, prim che esista `sl0` questo esempio configura il masquerading che non modifica quando viene creata l'interfaccia `ppp0`. Se servono considerazioni avanzate sulla sicurezza potrebbe risultare un po' limitato.

```
#/etc/rc.d/rc.local
/sbin/ipfwadm -F -p deny
/sbin/ipfwadm  -F -a m -S 192.168.0.0/24 -D 0.0.0.0/0
```

Come si può vedere è valido per le versioni 2.0 del kernel.

9 Programmi e versioni utilizzate

Per scrivere questo documento ho utilizzato le seguenti versioni di diald:

- Diald 0.16.5 - Ultima versione mantenuta dall'autore originale di diald.
- Diald 0.99.3 - Ultima versione fino alla prima edizione di questo documento.

la seguente versione di pppd:

- pppd 2.3.5

Diald versione 0.16.5 è, forse, la più comune e quella che viene inclusa dalle distribuzioni Linux. È sufficiente per molti siti ed è molto affidabile, ma certamente versioni successive hanno alcune capacità interessanti.

10 Ulteriori informazioni

Le informazioni originali da cui questo documento è stato tratto possono essere trovate nelle pagine del manuale di `diald`, `diald-examples`, `diald-control`, `diald-monitor`, `dctrl`, `pppd`, `chat`, e nelle informazioni contenute nelle directory in /usr/doc e nelle pagine web di questi pacchetti:

- Home Page Ufficiale di Diald: *http://diald.sourceforge.net/*
- Download delle nuove versioni: *ftp://diald.sourceforge.net/pub/diald/*
- Home page precedente di Diald: *http://diald.unix.ch*
- Vecchia home page di diald fino alla versione 0.16.5*http://www.loonie.net/~erics/diald.html*
- Sito FTP per pppd: *ftp://cs.anu.edu.au/pub/software/ppp/*
- Un altro sito: *http://www.p2sel.com/diald*
- Ancora uno: *http://rufus.w3.org/linux/RPM/*

C'è una mailing list per la discussione su diald sul server per liste di David S. Miller su vger.rutgers.edu. Per iscriversi basta spedire un messaggio a *Majordomo@vger.rutgers.edu* con il testo «subscribe linux-diald» nel corpo del messaggio.

Un archivio della lista può essere reperito su *http://www.geocrawler.com* .

Esistono anche vari documenti RCF (Request For Comments) che definiscono come l'incapsulamento della linea PPP e i protocolli associati (LCP, IPCP, PAP, CHAP, ...) debbano essere fatti. Tali documenti possono essere reperiti nella directory /usr/doc/doc-rfc e su alcuni siti WWW come *http://metalab.unc.edu* and *http://nic.mil/RFC* . È possibile chiedere informazioni sulle RCF in *RFC-INFO@ISI.EDU* .

I seguenti HowTo possono essere di aiuto:

- DNS-HOWTO - *http://www.linuxdoc.org/HOWTO/DNS-HOWTO.html*
- Firewall-HOWTO - *http://www.linuxdoc.org/HOWTO/Firewall-HOWTO.html*
- IP-Masquerade-HOWTO - *http://www.linuxdoc.org/HOWTO/IP-Masquerade-HOWTO.html*
- IPCHAINS-HOWTO - *http://www.linuxdoc.org/HOWTO/IPCHAINS-HOWTO.html*
- Modem-HOWTO - *http://www.linuxdoc.org/HOWTO/Modem-HOWTO.html*
- NET3-4-HOWTO - *http://www.linuxdoc.org/HOWTO/NET3-4-HOWTO.html*
- PPP-HOWTO - *http://www.linuxdoc.org/HOWTO/PPP-HOWTO.html*
- Serial-HOWTO - *http://www.linuxdoc.org/HOWTO/Serial-HOWTO.html*