Diald Howto

Andrés Seco *AndresSH@ctv.es*
version française par Xavier Serpaggi
v1.03, 17 avril 2000

Ce document présente quelques scénarios typiques permettant de commencer sans douleur à utiliser *Diald*. Ces scénarios comprennent une connexion à un FAI depuis un ordinateur isolé en employant PPP au travers d'un modem sans utiliser `pon/poff` ou `ppp-on/ppp-off`, vers un serveur de proxy/firewall, avec différentes connexions à l'Internet utilisant différents FAIs.

Contents

1 Introduction

1.1 Objectifs

Ce document présente quelques scénarios typiques permettant de commencer sans douleur à utiliser *Diald*. Ces scénarios comprennent une connexion à un FAI depuis un ordinateur isolé en employant PPP au travers d'un modem sans utiliser `pon/poff` ou `ppp-on/ppp-off`, vers un serveur de proxy/firewall, avec différentes connexions à l'Internet utilisant différents FAIs.

Les scénarios suivants seront traités dans ce document :

- connecter un ordinateur isolé à un FAI en utilisant un modem et PPP,
- connecter un ordinateur à divers FAI en utilisant un modem et PPP,
- connecter un proxy/firewall à un FAI en utilisant un modem et PPP.

Dans les versions ultérieures de ce document, d'autres scénarios seront ajoutés, comme la possibilité d'avoir plusieurs instances de *Diald*, l'utilisation des lignes ISDN (RNIS en France) ainsi que des lignes utilisées pour émettre et recevoir des appels.

Avant ce HOWTO, il existait un Diald-mini-Howto, écrit par Harish Pillay *h.pillay@ieee.org* et qui présentait un exemple de connexion à un FAI en utilisant un schéma d'authentification basé sur chat (login et et mot de passe avant le lancement de pppd, sans avoir recours à PAP ou à CHAP).

Des exemples de fichiers de configuration seront inclus dans ce document pour être utilisés comme point de départ à la mise en route de *Diald*. Pour obtenir des performances optimales et connaître tous les attributs des programmes il est nécessaire que vous lisiez toute la documentation de ceux-ci et que vous reconfiguriez les fichiers d'exemples donnés ici.

Enfin, selon la distribution de Linux que vous utilisez, les fichiers de configuration peuvent se trouver dans différents répertoires. Si vous trouvez un des fichiers présentés ici dans un autre répertoire que celui spécifié, je vous demande de me le faire savoir.

1.2 Nouvelles versions

La version la plus récente de ce document peut être trouvée sur ma page web *http://www.ctv.es/USERS/andressh/linux* aux formats SGML et HTML. D'autres versions et d'autres formats sont également disponibles en espagnol sur le site internet d'Insflug *http://www.insflug.org/documentos/Diald-Como/* . Enfin, ce document peut également être trouvé dans d'autres langues au LDP – Linux Documentation Project, *http://www.linuxdoc.org* .

1.3 Remerciements

Je veux remercier les personnes qui m'ont aidé à mettre en place et à faire tourner mon premier *Diald* grâce à leurs fichiers d'exemple (une personne dont j'ai oublié le nom, Mr Cornish Rex, Hoo Kok Mun et John Dalbec), les personnes qui m'ont envoyé des corrections et des suggestions relatives à ce document (Tim Coleman, Jacob Joseph, Paul Schmidt et Jordi Mallach), les futurs traducteurs et bien sûr, toutes les personnes qui ont développé et développent *Diald* pour nous.

2 Conditions de distribution et mise en garde

Nous avons mis dans les informations et autres contenus de ce document le meilleur de notre savoir. Cependant nous pouvons avoir fait des erreurs. C'est donc à vous de déterminer si vous voulez suivre les instructions données dans ce document.

Personne ne pourra être tenu pour responsable des éventuels dommages survenus à votre ordinateur ou d'autres pertes, suites à l'utilisation des informations contenues dans ce document.

L'AUTEUR ET LES PERSONNES QUI TIENNENT CE DOCUMENT À JOUR NE SONT RESPONSABLES D'AUCUN DOMMAGE SURVENU SUITE AUX ACTIONS EFFECTUÉES EN SE BASANT SUR LES INFORMATIONS CONTENUES DANS CE DOCUMENT.

Bien sûr, je suis ouvert à tout type de suggestions et à toutes corrections concernant le contenu de ce document.

3 Description rapide du fonctionnement de Diald

En quelques mots, *Diald* crée une nouvelle interface réseau et la définit comme étant la passerelle par défaut. Cette interface n'est pas réelle (dans la documentation originale elle est appelée `proxy interface`). *Diald* surveille cette interface et quand un paquet arrive, crée une connexion `ppp`, attend qu'elle soit établie et définit cette nouvelle interface `ppp` (en général `ppp0`) comme étant la passerelle par défault.

Diald surveille l'interface pour savoir quels paquets ont-été reçus par l'interface ainsi que leur type. Cela lui permettra de décider s'ils seront destinés à établir la connexion `ppp`, à maintenir le lien, à le fermer ou à ne rien faire, et cela lui permettra également de savoir combien de temps le lien devra être maintenu après la transmission du paquet.

En définitive, s'il n'y a plus de trafic et que la durée de vie du dernier paquet est écoulée, *Diald* fermera le lien.

Vous pouvez contrôler les jours et les heures auxquelles le lien peut ou ne peut pas être établi, et de ce fait vous pouvez n'établir le lien qu'aux moments où les prix sont bas ou le trafic faible.

Ce qui précède n'est valide que pour les versions de *Diald* supérieures à la 0.16.5 (la dernière version, au moment où ce document est écrit, est la 0.99.3), mais les dernières versions incluent des attributs supplémentaires comme une liste d'utilisateurs autorisés, un décompte du temps avancé, un meilleur support des lignes ISDN, de meilleures performances grâce à l'utilisation (à la place de `slip`) du device `ethertap` comme proxy (cela se comporte comme une interface réseau qui lit/écrit au travers d'une *socket* en lieu et place d'un vrai périphérique réseau), la sauvegarde des connexions et d'autres fonctions.

4 Note sur l'authentification

Quand vous vous connectez à un Fournisseur d'Accès Internet, vous devez en général donner votre nom d'utilisateur et votre mot de passe. Cela peut être fait en utilisant plusieurs méthodes ; la méthode que vous utilisez en pratique est déterminée par votre fournisseur.

En plus des trois possibilités exposées, vous pouvez utiliser un lien sans authentification (en général quand vous contrôlez également l'autre bout de la ligne).

4.1 Nom d'utilisateur et mot de passe – invites Login et password

En vérité il n'y a pas de méthode canonique pour l'authentification auprès d'un FAI lors de l'accès à l'Internet.

L'identification est faite avant le lancement de `pppd`, et c'est le logiciel qui numérote, en général `chat`, qui envoie le nom d'utilisateur et le mot de passe. Ces données sont envoyées sous forme de texte brut, donc cette méthode ne doit pas être considérée comme sûre.

Un script d'exemple pour `chat`, dans lequel vous pouvez voir comment faire passer un nom d'utilisateur et un mot de passe avant de lancer `pppd`, ressemblera à quelque chose comme ceci :

```
ABORT BUSY
ABORT "NO CARRIER"
ABORT VOICE
ABORT "NO DIALTONE"
ABORT "NO ANSWER"
"" ATZ
OK ATDT_Numéro_De_Téléphone_
CONNECT \d\c
ogin _Nom_d_Utilisateur_
assword _Mot_De_Passe_
```

Les deux dernières lignes définissent le nom d'utilisateur et le mot de passe et quand les envoyer (après avoir reçu respectivement « ogin » et « assword »). Le script chat n'a besoin de voir qu'une partie des mots « login » et « password » ce qui permet de ne pas tester la première lettre de chaque mot, et donc de s'affranchir des problèmes de majuscule/minuscule.

Supposons que ce script se nomme `provider`, et qu'il soit enregistré dans le répertoire `/etc/chatscript/`. Alors, vous pouvez le lancer avec :

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

4.2 PAP – Password Authentification Protocol : Protocole d'Authentification du mot de Passe

Si le fournisseur dont vous utilisez les services demande le protocole d'authentification PAP, pendant la négociation LCP de PPP, ce dernier devra l'utiliser. Quand, après l'utilisation de `chat`, la connexion est établie, `pppd` est lancé. Dans ce scénario, pppd va envoyer le nom d'utilisateur et le mot de passe qu'il cherchera dans le fichier `/etc/ppp/pap-secrets`. Ce fichier doit avoir les droits de lecture et d'écriture pour `root` uniquement. De ce fait, personne ne peut lire les mots de passe qu'il contient.

PAP n'est pas très sûr vu que le mot de passe est envoyé sous forme de simple texte et donc peut être lu par quelqu'un qui surveille votre ligne de transmission.

Voici un exemple simple de fichier `/etc/ppp/pap-secrets` :

```
_Nom_d_Utilisateur * _Mot_De_Passe_
```

4.3 CHAP – Challenge Authentication Protocol : Protocole d'Authentification Codé

Si le fournisseur dont vous utilisez les services demande le protocole d'authentification CHAP, pendant la négociation LCP de PPP, ce dernier devra l'utiliser. Quand, après l'utilisation de `chat`, la connexion est établie, `pppd` est lancé. Dans ce scénario, pppd va envoyer le nom d'utilisateur et le mot de passe qu'il cherchera dans le fichier `/etc/ppp/pap-secrets`. Ce fichier doit avoir les droits de lecture et d'écriture pour `root` uniquement. De ce fait, personne ne peut lire les mots de passe qu'il contient.

CHAP est plus sûr que PAP puisque le mot de passe n'est jamais envoyé en clair sur la ligne. Le serveur d'authentification envoie un identifiant au hasard (le *challenge*) que le client va crypter avec son mot de passe avant de le renvoyer à ce premier.

Voici un exemple simple de fichier `/etc/ppp/chap-secrets` :

```
_Nom_d_Utilisateur * _Mot_De_Passe_
```

Parfois un même FAI varie l'utilisation de ces protocoles, il n'est donc pas rare de définir le nom d'utilisateur et le mot de passe dans les deux fichiers.

5 Notes concernant la résolution de noms

Chaque fois que vous vous connectez à un FAI il est nécessaire qu'un DNS (service de noms) soit configuré, comme cela votre ordinateur peut trouver les adresses IP associées à un nom de machine.

Les adresses IP de votre serveur DNS sont placées dans le fichier `/etc/resolv.conf`.

Dans le cas d'un ordinateur isolé se connectant à l'internet, ce fichier contient généralement les adresses IP du serveur DNS de votre FAI.

```
#fichier /etc/resolv.conf pour les noms du FAI
nameserver 111.222.333.444
nameserver 222.333.444.555
```

Dans le cas d'un ordinateur qui est un proxy/firewall, ce fichier contient généralement la propre adresse IP de cette machine (ou l'adresse loopback – *de bouclage* : 127.0.0.1). Cet ordinateur doit faire tourner un serveur DNS qui transforme les noms de domaine en adresses IP en questionnant un DNS externe.

```
#fichier /etc/resolv.conf pour une résolution locale des noms
nameserver 127.0.0.1
```

L'installation d'un serveur DNS n'est pas le sujet de ce document, mais une approche rapide et pertinente peut être trouvée dans le DNS-Howto :*http://www.freenix.org/unix/linux/HOWTO/DNS-HOWTO.html* .

6 Connecter un ordinateur isolé à un FAI en utilisant un modem et PPP

Au moment de configurer *Diald* pour connecter votre ordinateur à votre FAI, il est nécessaire de faire ce qui suit :

- Installer le paquetage *Diald*. Le plus rapide étant d'installer le paquetage qui est contenu dans votre distribution de Linux.
- Configurer le DNS (à l'aide du fichier `/etc/resolv.conf`).
- Vérifier que vous pouvez appeler un FAI. Si votre distribution Linux a déjà un utilitaire pour configurer une connexion, le plus simple est de l'utiliser (pppconfig est l'utilitaire de Debian, kppp celui de KDE, etc.). Si vous avez des problèmes pour vous connecter à un FAI, les « Howto » suivants peuvent vous aider :
  - le PPP-Howto : *http://www.freenix.org/unix/linux/HOWTO/PPP-HOWTO.html* ,
  - le Modems-Howto : *http://www.freenix.org/unix/linux/HOWTO/Modems-HOWTO.html* et
  - le Serial-Howto : *http://www.freenix.org/unix/linux/HOWTO/Serial-HOWTO.html* .
- Configurer les fichiers `/etc/ppp/pap-secrets` et `/etc/ppp/chap-secrets` en y mettant votre nom d'utilisateur et votre mot de passe comme décrit plus haut.

Et enfin, pour *Diald* il faut :

- Préparer le fichier de configuration de *Diald* (`/etc/diald/diald.options` pour la version 0.16.5, et `/etc/diald/diald.conf` pour les versions plus récentes).
- Préparer le fichier de filtres `/etc/diald/standard.filter`, ou mieux, laisser ce fichier tel quel, et en modifier une copie que vous pouvez appeler `/etc/diald/personal.filter`.
- Préparer le script qui va se charger de l'appel (`/etc/diald/diald.connect` qui doit avoir les droits d'exécution de root) et le fichier d'instruction de `chat` (`/etc/chatscripts/provider`) qui sera utilisé par le précédent.
- Préparer les scripts qui vont être lancés aux moments où le lien sera établi et supprimé (`/etc/diald/ip-up` et `/etc/diald/ip-down`) si vous voulez utiliser cette possibilité. Les deux scripts doivent avoir les droits d'exécution de root.
- Préparer, si vous le désirez, les scripts pour établir et supprimer les routes (`/etc/diald/addroute` et `/etc/diald/delroute`). Tous les deux doivent avoir les droits d'exécution de root. Cette étape n'est pas nécessaire si vous n'utilisez qu'une seule instance de *Diald*.
- Finalement, lancer le démon `diald` (« `/etc/init.d/diald start` » pour une Debian et « `/etc/rc.d/init.d/diald start` » pour une RedHat). Normalement, les scripts pour lancer *Diald* au moment du démarrage de l'ordinateur sont mis en place au moment de l'installation du paquetage dans les répertoires `/etc/rcX.d`.

Si vous faites quelque changement que ce soit aux fichiers de configuration de *Diald* pendant que ce dernier est en train de tourner, vous devez le redémarrer (« `/etc/init.d/diald restart` » pour une Debian et « `/etc/rc.d/init.d/diald restart` » pour une RedHat).

6.1 Fichiers /etc/diald/diald.options ou diald.conf

Dans ce fichier d'exemple vous devez vérifier :

- le port de communication sur lequel votre modem est connecté. Option `device` ;
- la vitesse à laquelle votre port va communiquer avec votre modem. Option `speed` ;
- le nom d'utilisateur pour ppp. Option `pppd-options` ;
- le nombre d'essais de reconnexion après échec et le temps entre ces essais ;
- les heures de connexion autorisées : option `restrict` ;
- si vous désirez utiliser les scripts `ip-up` et `ip-down`. Options `ip-up` et `ip-down` ;
- si vous désirez utiliser les scripts `addroute` et `delroute` : options `addroute` et `delroute`. En général il n'est pas nécessaire de modifier ces scripts mais si vous utilisez plus d'une instance de *Diald* ou si vous avez une configuration complexe, vous devrez le faire ;
- si vous utilisez le fichier de filtre standard ou personnalisé : option `include`.

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

# Device auquel votre modem est connecté
device /dev/ttyS0

# Fichier de log
accounting-log /var/log/diald.log

# Surveillance de la queue
#fifo /var/run/diald/diald.fifo

# Activation du deboguage
# Activer le deboguage réduit les performances
#debug 31

# Nous utilisons PPP comme encapsulateur
mode ppp

# L'adresse IP locale (cette adresse sera automatiquement remplacée par celle
# que vous donne votre FAI au moment où vous vous connectez si vous utilisez
# l'option dynamic).
local 127.0.0.5

# L'adresse IP distante (au moment de la connexion cette adresse est remplacée
# automatiquement par l'adresse IP du serveur distant qui reçoit notre appel).
remote 127.0.0.4

# Masque de sous-réseau pour le lien vers le WAN
# (Wide Area Network -- réseau non local).
netmask 255.255.255.0

# L'adresse IP sera allouée dynamiquement au début de la connexion.
dynamic
```

```
# Si le lien est brisé par une action distante, ne le relancer que s'il reste
# des paquets à envoyer.
two-way

# Si le lien existe, router directement vers la vraie interface ppp et non pas
# vers le proxy. Si vous ne faites pas cela, vous aurez une perte de performances
# d'à peu prés 20%. Certains noyaux anciens ne suportent pas reroute.
# Reportez-vous au manuel de diald pour plus d'informations.
reroute

# Diald s'occupera d'établir la route par défaut.
defaultroute

# Scripts pour établir les routes personnalisées.
#addroute "/etc/diald/addroute"
#delroute "/etc/diald/delroute"

# Scripts à exécuter une fois que le lien est établi et prêt, ou une fois que le
# lien est détruit et fermé. Dans les versions 0.9x de Diald, il existe l'option
# ip-goingdown qui peut être utilisée au moment où le lien va être détruit mais
# qu'il est encore actif.
ip-up /etc/diald/ip-up
#ip-down /etc/diald/ip-down

# Scripts utilisés pour connecter ou déconnecter l'interface.
connect "/etc/diald/diald.connect"
#disconnect "/etc/diald/diald.disconnect"

# Utiliser un verrou UUCP pour signaler que le périphérique est en cours
# d'utilisation.
#lock

# Nous nous connectons à l'aide d'un modem. ATTENTION : ne mettez pas cette
# option dans le fichier d'options de ppp sinon il y aura conflit avec les
# options de diald. Pour connaître les options ppp que vous ne pouvez pas
# utiliser dans le fichier pppd-options, reportez-vous à la page de manuel de
# diald et recherchez pppd-options.
modem
crtscts
speed 115200

# Quelques options sur les compteurs et le nombre d'essais.
# Reportez-vous à la page de manuel de Diald pour plus d'informations.
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

# Options qui seront passées à pppd.
# Ces options peuvent être incluses dans le fichier /etc/ppp/options. Elles
# seront considérées comme options par défaut de pppd, mais si vous avez
# d'autres besoins de configurations pour Diald, dans le cas où vous avez
```

```
# plusieurs instances, vous devez péciser ici
# noauth - ne pas demander d'authentification au serveur distant.
#          "Infovía Plus" (en Espagne) n'identifie pas votre machine.
# user   - votre nom d'utilisateur et de fai. Demandez à votre FAI la syntaxe
#          exacte vu que certains n'ont pas besoin du @fai.
pppd-options noauth user utilisateur@fai

# Restrictions horaires.
# Cette section doit apparaître avant les filtres.
# La commande restrict est expérimentale et peut changer dans les versions
# futures de diald. Repotez-vous à la page de manuel. (Cet exemple a été testé
# pour la version 0.16 mais je pense qu'il fonctionne sur les versions plus
# récentes.)
# Exemple : utilisation uniquement la nuit entre lundi et vendredi et toute la
# journée les samedi et dimanche.
restrict 8:00:00 18:00:00 1-5 * *
down
restrict * * * * *

# Pas de considérations spéciales sur la tarification (le nombre de secondes
# au départ comprises dans le coût, l'unité de tarification en secondes, temps
# en secondes pour tester s'il faut détruire le lien)
impulse 160,0,0
# Si c'est tarifé à la minute et que les 10 premières seront toujours comptées :
#impulse 600,60,10

# Filtres standards
#include /etc/diald/standard.filter
# Ou filtres personnels
include /etc/diald/personal.filter
```

6.2 Fichier de filtres personnels

Vous devez modifier ce fichier avec beaucoup de précautions. Il est utilisé pour décider quand et pourquoi il faut établir la connexion, la maintenir, la détruire ou ignorer les paquets en fonction du type de trafic.

En général le fichier de filtres standard de *Diald* convient pour la plupart des cas mais il peut s'avérer trop ou au contraire pas assez restrictif dans certaines situations. Le fichier `personal.filter` que nous présentons a subi quelques corrections par rapport à l'original de la version 0.16.

Dans les versions futures de ce document, d'autres exemples commentés plus restrictifs seront inclus.

```
# /etc/diald/personal.filter
# Les régles de filtrage présentées ici sont les mêmes que dans le fichier
# standard.filter avec les changements suivants :
#
# Changement de 10 à 4 minutes dans le cas de "n'importe quelle autre connexion
# tcp".
# Ajout de "ignore tcp tcp.fin" pour ignorer les paquets FIN ACK.
# Ignorer les paquets icmp (ping et traceroute n'activent pas l'interface).
#


# Voici un ensemble de règles de filtrages assez compliquées
# (Ce sont celles que j'utilise.)
```

```
#
# J'ai décomposé les règles en quatre sections :
# Les trames TCP, les trames UDP, les trames ICMP et enfin une règle
# générale pour tout le reste.

ignore icmp any

#----------------------------------------------------------------------------
# Règles pour les trames TCP.
#----------------------------------------------------------------------------
# Commentaires généraux sur cet ensemble de règles :
#
# En général nous souhaitons ne traiter que les données d'une liaison TCP
# ayant un sens pour le temps de déconnexion. Cependant, nous essayons
# d'ignorer les trames sans données.
# Puisque la taille minimale d'un en-tête dans une trame TCP est de 40 octets,
# toutes les trames d'une longueur de 40 ne devraient pas contenir de données.
# De cette manière il est possible de manquer des trames vides (des informations
# optionnelles de routage et d'autres choses supplémentaires peuvent être
# présentes dans un en-tête IP), mais nous devrions en capter la majorité.
# Remarquez que nous ne voulons pas rejeter les trames avec un champ tcp.live
# vide puisque nous les utiliserons plus tard pour accélérer la déconnexion de
# certaines liaisons TCP.
#
# Nous voulons également nous assurer que les trames WWW restent en vie même
# si la socket TCP est arrêtée. Nous faisons cela parce que WWW ne garde pas la
# connexion une fois que les données ont été transférées, et il serait
# gênant d'avoir une liaison qui se crée et se coupe à chaque document.
#
# En dehors de WWW, l'utilisation la plus courante de TCP concerne les
# connexions de longue durée dont la coupure signifie que vous n'allez plus
# avoir besoin du réseau.
# Nous ne voulons pas nécessairement avoir à attendre 10 minutes que la
# connexion se termine alors que nous n'avons ni telnet ni rlogin en cours, donc
# nous voulons accélérer le délai de déconnexion sur les liaisons TCP qui sont
# terminées. Cela est réalisé en mettant dans un cache les trames qui n'ont pas l'indicateur
# live positionné.

# --- début proprement dit de l'ensemble de règles ---

# Quand on débute une connexion on ne donne tout d'abord au lien que 15
# secondes. L'idée ici est de pouvoir éventuellement se rendre compte que le
# réseau de l'autre côté de la connexion n'est pas accessible. Dans ce cas
# il n'est pas nécessaire de donner un temps de vie de 10 minutes au lien.
# Avec la règle ci-dessous nous ne lui donnons initialement que 15 secondes.
# Si le réseau est accessible, nous devrions normalement recevoir une
# réponse contenant des données dans les 15 secondes. Si cela pose un problème
# parce que vous avez des réponses lentes de certains sites que vous
# visitez régulièrement, vous pouvez augmenter le temps avant déconnexion ou
# bien supprimer cette règle.
accept tcp 15 tcp.syn

# Empêcher named de garder la connexion active.
ignore tcp tcp.dest=tcp.domain
ignore tcp tcp.source=tcp.domain
```

```
# (Argh ! Le telnet de SCO commence par envoyer des SNY vides et n'initie la
# connexion que s'il obtient une réponse. Pfuuutt...
accept tcp 5 ip.tot_len=40,tcp.syn

# Empêcher les trames vides (autres que les trames
# vides SNY) de maintenir le lien actif.
ignore tcp ip.tot_len=40,tcp.live

# Modifications d'Andres Seco pour ignorer les paquets FIN ACK.
ignore tcp tcp.fin

# On s'assure que le transfert http maintient la ligne active pendant 2
# minutes, même après que c'est terminé.
# REMARQUE : votre fichier /etc/services ne devrait pas définir le service tcp
# www, auquel cas vous devez commenter les deux lignes suivantes et vous
# procurer un fichier /etc/services plus récent. Lisez la FAQ pour savoir
# comment obtenir un nouveau fichier /etc/services.
accept tcp 120 tcp.dest=tcp.www
accept tcp 120 tcp.source=tcp.www
# Idem pour https
accept tcp 120 tcp.dest=tcp.443
accept tcp 120 tcp.source=tcp.443

# Une fois que le lien n'est plus actif, nous tentons de stopper la connexion
# rapidement. Remarquez que si le lien est déjà arrêté, un changement d'état
# ne le ramènera pas à l'état actif.
keepup tcp 5 !tcp.live
ignore tcp !tcp.live

# une donnée ftp ou une connexion ftp peut être attendue pour rendre compte
# du trafic relativement fréquent.
accept tcp 120 tcp.dest=tcp.ftp
accept tcp 120 tcp.source=tcp.ftp

# REMARQUE : les données ftp ne sont pas définies dans le fichier /etc/services
# distribué dans les dernières versions de NETKIT, donc j'ai commenté ce
# passage.
# Si vous désirez le définir, ajoutez la ligne suivante à votre fichier
# /etc/services :
# ftp-data        20/tcp
# et décommentez les deux règles suivantes :
#accept tcp 120 tcp.dest=tcp.ftp-data
#accept tcp 120 tcp.source=tcp.ftp-data

# Si nous n'avons pas réussi à l'avoir avec les règles ci-dessus, donnons au
# lien 10 minutes de plus.
#accept tcp 600 any
# Modification d'Andres Seco. Ne lui donnons que 4 minutes de plus.
accept tcp 240 any

# Règles pour les trames UDP.
#
# Nous donnons dès à présent un temps limite aux requêtes de domaine puisque
# nous voulons seulement qu'elles établissent le lien, pas qu'elles le maintiennent
```

```
# pour un long moment.
# Cela parce que le réseau sera généralement établi par un appel de la
# bibliothèque de résolution de nom (à moins que vous n'ayez mis toutes les
# adresses que vous utilisez fréquemment dans /etc/hosts, auquel cas vous
# découvrirez d'autres problèmes.)
# Remarquez que vous ne devez pas donner une valeur de temps limite de
# déconnexion plus courte que le temps supposé que va mettre votre DNS pour
# répondre. Sinon, quand le lien original s'établit, il risque d'y avoir
# une attente supérieure à celle qu'il y a entre les séries de trames initiales
# avant qu'une trame destinée à maintenir le lien passe par ce dernier.

# Ne pas activer le lien pour rwho.
ignore udp udp.dest=udp.who
ignore udp udp.source=udp.who
# Ne pas activer le lien pour RIP.
ignore udp udp.dest=udp.route
ignore udp udp.source=udp.route
# Ne pas activer le lien pour NTP ou pour 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
# Ne pas activer les requêtes de nom de domaine entre deux exécutions de named.
ignore udp udp.dest=udp.domain,udp.source=udp.domain
# Activer le réseau pour les requêtes de domaine qui ne viennent pas de
# named.
accept udp 30 udp.dest=udp.domain
accept udp 30 udp.source=udp.domain
# Faire la même chose pour les diffusions de netbios-ns.
# REMARQUE : votre fichier /etc/services peut ne pas définir le service
# netbios-ns, auquel cas vous devez commenter les trois lignes suivantes.
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
# empécher les transferts de routed et gated de maintenir le lien actif
ignore udp tcp.dest=udp.route
ignore udp tcp.source=udp.route
# Le reste a droit à 2 minutes.
accept udp 120 any

# Récupérer toutes les trames que nous n'avons pas traitées auparavant et donner 30
# secondes de durée de vie à la connexion.
accept any 30 any
```

6.3 Téléphoner

Le fichier `/etc/diald/diald.connect` (il doit avoir les droits d'exécution) :

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

Le fichier `/etc/chatscripts/provider`. Dans ce fichier d'exemple vous devez modifier le numéro de téléphone à appeler (le 0123456789) :

```
ABORT BUSY
```

```
ABORT "NO CARRIER"
ABORT VOICE
ABORT "NO DIALTONE"
ABORT "NO ANSWER"
"" ATZ
OK ATDT0123456789
CONNECT \d\c
```

6.4 Script de lancement de la connexion

Il doit avoir les droits d'exécution.

Ce script peut être utilisé pour de nombreuses tâches (synchronisation, envoi des messages en attente, récupération des messages chez le FAI, etc.).

```
#!/bin/sh

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

# mise à jour de l'heure et de la date
# netdate ntp.server.somecountry

# s'occupe des mails dans la file d'attente
# runq

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

7 Connexion d'un ordinateur à différents FAI à l'aide d'un modem et de PPP

Il n'est pas rare qu'un ordinateur isolé ne se connecte pas qu'à un unique réseau. Il est courant de se connecter à différents réseaux ou à l'Internet par le biais de différents FAI. Dans ce cas il peut être agaçant de changer vos fichiers de configuration à chaque fois que vous voulez vous connecter à un aute site.

La solution que je propose ici consiste à utiliser un jeu de plusieurs fichiers de configuration ; un pour chaque connexion. Vous pouvez trouver ici quelques scripts qui permettent d'automatiser le changement de l'un à l'autre.

7.1 Remarque sur l'envoi de courrier à l'aide d'un hôte relais

Si le progamme avec lequel vous lisez votre courrier utilise un Agent de Transfert de Messages (MTA en anglais) avec un hôte relais `smtp` à qui envoyer tous les messages, ou s'il envoie les messages directement au serveur `smtp` de votre FAI, changer de connexion signifie avoir à reconfigurer cette option pour le serveur relais `smtp`. Cela à cause du fait que les FAI vérifient en général si la boîte de réception est locale ou bien sur tout domaine qui leur est directement rattaché ou si l'adresse IP de l'envoyeur appartient bien à l'intervalle d'adresses IP assigné par ce FAI et ce, dans le but d'éviter d'avoir un serveur relais ouvert qui puisse être utilisé pour envoyer des *spams*, des messages anonymes, etc.

Dans les exemples suivant vous apprendrez comment changer ce paramètre dans les fichiers de configuration de `Smail` en un fichier simple grâce auquel tous les messages externes seront envoyés à un serveur `smtp` relais. Si vous utilisez un autre MTA sur votre système, vous pouvez m'envoyer les changements nécessaires pour que je les inclue ici. Si vous utilisez un programme pour lire votre courrier qui s'adresse directement au serveur `smtp` externe (Kmail, Netscape, etc.), envoyez-moi également vos modifications.

7.2 Scripts pour automatiser les connexions multiples et les changements de l'une à l'autre

7.2.1 Démarrage

En tout premier lieu vous devez créer un sous-répertoire à `/etc/diald` appelé `providers` (*traduction anglaise de fournisseurs !*) où vous stockerez 1) vos scripts pour passer automatiquement d'un FAI à l'autre, et 2) les sous-répertoires contenant le jeu de fichiers nécessaire à la configuration de la connexion chez chaque FAI.

Le script suivant crée ce répertoire et y met les fichiers de configuration actuels de *Diald*, *chat*, *pppd* et *Smail*. Ces derniers seront utilisés comme modèles pour les futures configurations.

```
#!/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 nouveau FAI

Le script suivant vous aidera à copier la configuration qui sert de modèle vers un nouveau répertoire, pour la modifier dans l'optique d'une nouvelle connexion à un FAI ou à un réseau. Ce script (`/etc/diald/providers/newdialdprovider/`) prendra comme paramètre le nom du FAI ou du réseau.

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

À présent vous pouvez modifier, en fonction de vos besoins, les fichiers nouvellement créés dans `/etc/diald/providers/providername`, tout en gardant à l'esprit que `providername` est le paramètre passé au script `newdialdprovider`.

7.2.3 Passer de l'un à l'autre

Pour terminer, ce script vous permettra de changer les fichiers de configuration de *Diald* pour pouvoir vous connecter à un autre FAI ou à un autre réseau. J'utilise des liens symboliques pour éviter d'avoir à dupliquer les fichiers. L'utilisation de tels liens permet que toute modification d'un fichier original comme par exemple `/etc/resolv.conf` soit reportée dans le fichier lié, ici `/etc/diald/providers/providername/resolv.conf`.

```
#!/bin/sh
#Fichier /etc/diald/providers/setdialdprovider
/etc/init.d/diald stop
#On attend que Diald s'arrête
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 Connexion d'un proxy/firewall à un FAI à l'aide d'un modem et de PPP

Connecter un réseau privé à l'Internet avec un serveur dédié qui gère le routage des paquets à partir du réseau local vers l'Internet tout en proposant des services de proxy/cache et la sécurité d'un firewall est un thème complexe qui va au-delà du but de ce document. Il existe d'autres « Howto » qui expliquent ces sujets de manière beaucoup plus compréhensible. Vous trouverez, 10 (à la fin de ce document), une liste de liens et de références vers ces documents.

Ici, nous nous contentons de configurer *Diald* en supposant que l'ordinateur fait déja du masquage d'adresses IP (*IP-Masquerading*), dispose d'un proxy tel que *Squid*, d'une connexion à un FAI correctement configurée et que la sécurité des accès aux ports TCP/UDP a été revue (fichiers `/etc/inetd.conf` et autres tels que `securetty`, `host.allow`, etc.).

Pour faire simple, il suffit uniquement de reconfigurer les règles pour le masquage/le filtrage/l'accès, à chaque fois que le jeu d'interfaces change, donc quand l'interface ppp0 est établie et qu'elle est détruite. De bons endroits pour faire cela sont les scripts `ip-up` et `ip-down` de *pppd*.

8.1 Exemple pour Debian 2.1

Avec Debian il vous suffit d'installer le paquetage *ipmasq* et de dire, au moment de la configuration, que vous voulez changer les règles de manière synchrone à *pppd*. Des scripts seront créés dans les répertoires `/etc/ppp/ip-up.d` et `/etc/ppp/ip-down.d` pour prendre en charge le lancement de `/sbin/ipmasq`, un script qui analyse les interfaces existantes et crée une configuration simple qui est valable dans la plupart des cas ; vous pouvez le personnaliser à l'aide des fichiers de règles de `/etc/ipmasq/rules`.

La seule correction à faire après avoir installé ce paquetage est de modifier le moment du lancement du script *ipmasq* en effaçant le lien symbolique contenu dans le répertoire `/etc/rcS.d` et en en créant un nouveau dans `/etc/rc2.d` *après* `S20diald`. À présent, quand `ipmasq` est exécuté pour analyser les interfaces, `sl0` existe déjà. `S90ipmasq` est un bon nom pour ce lien symbolique qui pointe vers `/etc/init.d/ipmasq`.

Le fait d'utiliser Debian vous permet de ne pas avoir à vous soucier de la version de votre noyau puisque le script `/sbin/ipmasq` utilise `ipfwadm` ou `ipchains` selon le cas.

8.2 Exemple pour Suse 6.1

Cet exemple nous vient de M. Cornish Rex, *troll@tnet.com.au* .

Les commandes d'ip-masq et du contrôle de routage mises en oeuvre ici sont à employer avec les noyaux version 2.2 qui utilisent ipchains et ne sont pas valides pour les versions 2.0.

Nous allons supposer que l'interface ethernet a l'adresse ip 192.168.1.1 avec un masque réseau de 16 bits, soit 255.255.0.0.

Voici le fichier `/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
```

Voici le fichier `/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
```

Voici le dernier fichier du dernier script, `origin.chain` (statut original d'ipchains) :

```
# orig.chains
# crée avec : 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 Exemple pour Slackware 3.6

Cet exemple nous a été donné par Hoo Kok Mun, *hkmun@pacific.net.sg* .

C'est l'exemple le plus simple qu'il m'ait été donné de voir mais il est complètement fonctionnel. Cet exemple nous montre comment configurer le masquage avant l'existence de l'interface `sl0` et il ne change pas quand l'interface ppp0 apparaît. Si vos exigences en matière de sécurité sont assez élevées, il peut s'avérer limité.

```
#/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
```

Comme vous pouvez le voir, il est destiné aux noyaux version 2.0.

9 Programmes et versions utilisées

Pour écrire ce document j'ai utilisé les versions suivantes de diald :

- Diald 0.16.5 – dernière version maintenue par l'auteur original de diald.

- Diald 0.99.3 – dernière version avant la première édition de ce document.

Et les versions suivantes de pppd :

- pppd 2.3.5

La version 0.16.5 de diald est peut-être la plus étendue et celle qui est incluse dans le plus de distributions Linux. Elle est suffisante pour la plupart des sites et elle est très stable, mais bien sûr, les versions plus récentes ont de nombreuses fonctions intéressantes.

10 Pour en savoir plus

Les informations originales, sources de ce document, peuvent être trouvées dans les pages de manuel de `diald`, `diald-examples`, `diald-control`, `diald-monitor`, `dctrl`, `pppd`, `chat`, ainsi que dans les informations contenues dans les répertoires de `/usr/doc` et sur les pages web de ces paquetages :

- Nouvelle page officielle de Diald : *http://diald.sourceforge.net/*
- Téléchargement des nouvelles versions : *ftp://diald.sourceforge.net/pub/diald/*
- Ancienne page de Diald : *http://diald.unix.ch*
- Ancienne page de Diald avant la version 0.16.5 : *http://www.loonie.net/ erics/diald.html*
- Site FTP de pppd : *ftp://cs.anu.edu.au/pub/software/ppp/*
- Autres sites : *http://www.p2sel.com/diald*
- Et un de plus : *http://rufus.w3.org/linux/RPM/*

Il existe une liste de diffusion, pour ce qui concerne les discussions qui tournent autour de diald, sur le serveur de liste de David S. Miller à vger.rutgers.edu. Pour s'abonner il suffit d'envoyer un message à *Majordomo@vger.rutgers.edu* avec le texte « subscribe linux-diald » **dans le corps du message**.

Une archive de cette liste peut être trouvée à *http://www.geocrawler.com* .

Il existe également de nombreux documents RFC (*Request For Comments – demande de commentaires*) qui définissent la manière dont les lignes encapsulées de PPP et les protocoles associés (LCP, IPCP, PAP, CHAP, ...) doivent fonctionner. Vous pouvez trouver ces documents dans le répertoire `/usr/doc/doc-rfc` et sur certains sites de la grande toile mondiale comme par exemple *http://metalab.unc.edu* et *http://nic.mil/RFC* . Vous pouvez demander des informations concernant les RFC en envoyant un courrier électronique à *RFC-INFO@ISI.EDU* .

Les « Howto » suivants peuvent vous être utiles :

- DNS-HOWTO – *http://www.freenix.org/unix/linux/HOWTO/DNS-HOWTO.html*
- Firewall-HOWTO – *http://www.freenix.org/unix/linux/HOWTO/Firewall-HOWTO.html*
- IP-Masquerade-HOWTO – *http://www.linuxdoc.org/HOWTO/IP-Masquerade-HOWTO.html*
- IPCHAINS-HOWTO – *http://www.freenix.org/unix/linux/HOWTO/IPCHAINS-HOWTO.html*
- Modems-HOWTO – *http://www.freenix.org/unix/linux/HOWTO/Modems-HOWTO.html*
- NET4-HOWTO – *http://www.freenix.org/unix/linux/HOWTO/NET4-HOWTO.html*
- PPP-HOWTO – *http://www.freenix.org/unix/linux/HOWTO/PPP-HOWTO.html*
- Serial-HOWTO – *http://www.freenix.org/unix/linux/HOWTO/Serial-HOWTO.html*