

# **Paquetage PROXY**

## **Version 4.0.0-trunk-x86\_64-r60769**

Frank Meyer  
courriel: [frank@fli41.de](mailto:frank@fli41.de)

L'équipe fli4l  
courriel: [team@fli41.de](mailto:team@fli41.de)

1<sup>er</sup> octobre 2022

# Table des matières

<b>1</b>	<b>Documentation du paquetage PROXY</b>	<b>3</b>
1.1	PROXY - Différent Serveur proxy . . . . .	3
1.1.1	OPT_PRIVOXY - Filtrage de la publicité avec un proxy HTTP . . . . .	3
1.1.2	OPT_TOR - Système de communication anonyme pour Internet . . . . .	5
1.1.3	OPT_SS5 - Proxy Socks 4/5 . . . . .	7
1.1.4	OPT_TRANSPROXY (Expérimental) - Proxy HTTP transparent . . . . .	7
1.1.5	OPT_SIPROXD (Expérimental) - Proxy pour Session Initiation Protocol . . . . .	8
1.1.6	OPT_IGMPPROXY - Proxy pour Internet Group Management Protocol . . . . .	8
1.1.7	OPT_STUNNEL - Tunnel avec une connexion SSL/TLS . . . . .	15
	<b>Table des figures</b>	<b>22</b>
	<b>Liste des tableaux</b>	<b>23</b>
	<b>Index</b>	<b>24</b>

# 1 Documentation du paquetage PROXY

## 1.1 PROXY - Différent Serveur proxy

### 1.1.1 OPT\_PRIVOXY - Filtrage de la publicité avec un proxy HTTP

Privoxy "Privacy Enhancing Proxy" ("filtrage avancé, pour la protection de la vie privée") voir le site Web officiel de Privoxy (<http://www.privoxy.org/>). Privoxy filtre le contenu des pages web sur votre navigateur, en remplaçant par des images vides les bannières publicitaires et les Popups, Il gère les cookies dans une mémoire cache (petit paquet de données avec lesquels un site web peut reconnaître certain surfer) et empêche l'affichage de ce que l'on appelle bugs-Web (ce sont de grandes images 1x1 pixels, qui sont utilisées, pour espionner le comportement des utilisateurs sur le Net).

Pendant que Privoxy fonctionne, vous pouvez tout simplement configurer et activer les paramètres par l'intermédiaire de l'interface Web. L'interface Web se trouve à l'adresse <http://config.privoxy.org/> ou en abrégée <http://p.p/>.

Privoxy Internet Junkbusters à eu une évolution conséquente à partir de la version 2.1.0, voir le site Web (<http://www.junkbuster.com/>). L'innovation la plus importante, est que toutes les règles de filtrage sont centralisées dans un fichier `default.action`. Celui-ci se trouve dans le répertoire `fli4l/etc/privoxy`. Le grand avantage de cette méthode c'est que les nouvelles versions de ce fichier peuvent être télécharger séparément à cette adresse <http://sourceforge.net/projects/ijbswa/files/>.

Ainsi, chaque utilisateur fli4l peut tenir ce fichier à jour, sans mettre à jour le routeur-fli4l. (Actuellement, la version 1.8 de ce fichier est dans ce paquetage)

**PRIVOXY\_MENU** Avec cette variable, vous pouvez ajouter la section Privoxy au menu-`httpd`.

**PRIVOXY\_N** Vous indiquez dans cette variable le nombre de Privoxy qui doit être enregistré pour chaque interface.

**PRIVOXY\_x\_LISTEN** Vous indiquez dans cette variable, l'adresse-IP ou le nom symbolique, y compris le numéro de Port de l'interface, sur lequel le Privoxy doit écouter les connexions des clients. C'est une bonne idée d'indiquer ici, seulement les adresses des interfaces que l'on fait confiance, car tous les ordinateurs auront un accès complet à travers le Privoxy (avec bien sur le navigateur configuré et activé). En règle générale il est judicieux d'indiquer, la valeur par défaut qui est `IP_NET_1_IPADDR:8118`

Avec l'adresse indiquée ici, le Privoxy écoute et offre ses services. Le port par défaut est 8118. Vous devez utiliser cette information pour configurer le proxy dans votre navigateur. Pour plus de détail sur la configuration d'Internet Explorer et de Netscape Navigator, voir le site Web :

<http://www.privoxy.org/>

Vous devez enregistrer dans chaque navigateur, en tant que proxy l'ordinateur-fli4l, vous allez donc prendre le nom de la variable `HOSTNAME='fli4l'` ou l'adresse-IP (par ex.

Une configuration ainsi activer ne survit pas à un redémarrage du routeur fli4l... (tobig)

URL pré-cise

192.168.6.1) de la variable `HOST_x_IP='192.168.6.1'` qui est dans le fichier config de `fli4l`. Avec le Port par défaut, on a ici tous les paramètres nécessaires, pour configurer votre navigateur Web, pour l'utilisation du Privoxy.

**PRIVOXY\_x\_ALLOW\_N** Vous indiquez dans cette variable le nombre d'adresse réseau à installer.

**PRIVOXY\_x\_ALLOW\_x** Vous indiquez dans cette variable l'adresse réseau ou l'adresse-IP pour le quelle le filtrage de paquets doit être ouvert. Normalement il est logique d'indiquer ici le paramètre `IP_NET_1`.

**PRIVOXY\_x\_ACTIONDIR** Avec cette variable vous indiquez l'emplacement, ou vous pouvez paramétrer l'ensemble des règles Privoxy (pour les fichiers *default.action* et *user.action*) sur le routeur. Le chemin d'accès spécifié est évaluée par rapport au répertoire racine. Cette variable peut être utilisé pour deux choses différentes :

**Le déplacement dans la mémoire permanente de l'ensemble des règles** Si vous spécifiez un répertoire dans un emplacement autre que le disque-RAM, au démarrage de `fli4l` l'ensemble des règles défini par défaut seront copiés et utilisé à partir de cet emplacement. Les modifications apportées à ces ensembles de règles survivront à un redémarrage du routeur. On doit aussi tenir compte du fait qu'après une mise à jour du paquetage Privoxy, ces règles seront toujours utilisées donc l'ensemble des règles du paquetage de mise à jour sera simplement ignoré.

**l'utilisation de vos propres ensembles de règles** L'utilisateur `fli4l` permet d'écrire des règles spécifiques à la place des règles standard. Vous devez dans cette variable indiquer votre propre sous-répertoire qui sera dans le répertoire *config* (par exemple *etc/mon\_privoxy*, par contre vous ne devez pas indiquer *etc/privoxy*) ensuite vous placez dans ce sous-répertoire vos propres règles.

Le paramétrage de cette variable est optionnel.

**PRIVOXY\_x\_HTTP\_PROXY** Si vous voulez utiliser en plus du Privoxy un autre Proxy HTTP, c'est-à-dire utiliser également des pages Web en cache, vous pouvez paramétrer cette variable. Le Privoxy utilise alors ce proxy. Avec cette variable vous avez l'avantage utilisé plusieurs Proxys. Le paramètre peut ressembler à cela :

```
PRIVOXY_1_HTTP_PROXY='mon.provider.de:8000'
```

Ce paramètre est optionnel.

**PRIVOXY\_x SOCKS\_PROXY** Si vous voulez utiliser en plus du Privoxy un autre Proxy SOCKS. Pour augmenter la surveillance privée de la transmission de données du Privoxy, par exemple, envoyé les données par le réseau Tor, vous pouvez paramétrer cette variable. Pour plus de détails sur Tor, reportez-vous à la [Documentation Tor](#) (Page 5). Le paramètre pour utiliser Tor peut ressembler à cela :

```
PRIVOXY_x SOCKS_PROXY='127.0.0.1:9050'
```

Ce paramètre est optionnel.

**PRIVOXY\_x\_TOGGLE** Avec cette variable vous pouvez arrêter le proxy par l'interface Web. Si le Privoxy est mis hors circuit, il réagira simplement comme Proxy-Forwarding et ne modifiera plus le contenu des pages Web transférées. Vous devez considérer, que ce réglage vaut pour TOUS les utilisateurs du Proxy, c.-à-d. que si un utilisateur arrête Privoxy, le Privoxy sera coupé pour tous les autres utilisateurs Web qui transfert par le Proxy.

**PRIVOXY\_x\_CONFIG** Avec cette variable, les utilisateurs ont la possibilité de configurer le proxy par l'interface web. Pour plus de détails, je vous demande de consulter la documentation Privoxy qui est ici.

**PRIVOXY\_x\_LOGDIR** Dans cette variable vous pouvez indiquer le répertoire du fichier log (ou journal) pour le privoxy. Cela peut être utile, par ex. lorsque l'utilisateur veut enregistrer les accès des sites Web. Si rien n'est spécifié (par défaut), les principaux messages seront enregistrés sur la console, de plus la variable **PRIVOXY\_LOGLEVEL** sera ignorée.

Vous pouvez aussi indiquer 'auto', le chemin du fichier log sera alors déplacé dans le répertoire système, pour avoir des données persistantes. S'il vous plaît, assurez-vous que la variable **FLI4L\_UUID** soit dans ce cas configuré correctement. Comme on peut s'attendre une grandes quantités de données sera enregistrées et le fichier log dans le /boot ou dans le Disque-RAM sera rempli rapidement.

**PRIVOXY\_x\_LOGLEVEL** On indique dans cette variable les valeurs, pour que Privoxy puisse enregistrer les événements dans le fichier log. Il est possible d'ajouter plusieurs valeurs à la suite, vous devez les séparer par un espace. Les valeurs suivantes peuvent être ajoutées.

Valeur	ce qui sera enregistré ?
1	Chaque Requête (GET/POST/CONNECT).
2	Le statut de chaque connexion
4	Le statut-I/O
8	Header-Parsing
16	<b>Toutes</b> les données
32	Debug force-feature
64	Debug regular expression filters
128	Debug redirects
256	Debug GIF animation
512	Common Log Format (Analyse fichier-log)
1024	Debug kill pop-ups
2048	CGI (server web) user interface
4096	Startup banner and warnings
8192	Non-fatal errors

Pour produire un fichier log (ou journal) avec Common Log Format, vous devez indiquer SEULEMENT la valeur 512, si vous indiquez d'autres valeurs le fichier log sera "pollué" par d'autres enregistrements et vous aurez des problèmes pour l'analyser.

Privoxy offre de très nombreuses options de configurations. Cependant pour des raisons compréhensibles nous ne pouvons pas développer toutes ces options dans le fichier de configuration de fli4l. Beaucoup de ces options peuvent être paramétrées sur l'interface Web de Privoxy. Vous trouverez des infos plus précises pour la configuration de ces fichiers sur la page d'accueil de Privoxy. Les fichiers de configuration de Privoxy se trouvent dans le répertoire <fli4l-Version>/opt/etc/privoxy/. Ce sont des fichiers originaux du Paquetage-Privoxy, toutefois, pour gagner de la place, tous les commentaires ont été supprimés.

### 1.1.2 OPT\_TOR - Système de communication anonyme pour Internet

Tor est l'outil d'un grand nombre d'organismes et de simples citoyens, qui veulent améliorer leur protection et leur sécurité sur Internet. L'utilisation de Tor vous aide à être anonymes

lors de la navigation et de la publication sur le Web, messagerie instantanée, IRC, SSH et autres applications basées sur TCP. En outre, Tor fournit une plate-forme sur laquelle les développeurs de logiciels peuvent créer de nouvelles applications pour plus d'anonymat, sur la sécurité et la protection de la vie privée.

<https://www.torproject.org/index.html.fr>

## **TOR\_LISTEN\_N**

**TOR\_LISTEN\_x** Dans la première variable, vous indiquez le nombre d'adresse réseau, dans la deuxième variable vous indiquez l'adresse-IP ou le nom symbolique, y compris le numéro de Port de l'interface, sur lequel Tor doit écouter les connexions des Clients. C'est une bonne idée, d'indiquer ici seulement les adresses des interfaces que l'on fait confiance, car tous les ordinateurs auront un accès complet à travers Tor (avec bien sur le navigateur configuré et activé). En règle générale il est judicieux d'indiquer, la valeur par défaut qui est `IP_NET_1_IPADDR:9050`

Avec l'adresse indiquée ici, Tor écoute et offre ses services. Le port par défaut est 9050. Vous devez utiliser cette information pour configurer le proxy dans votre navigateur.

Vous devez indiquer dans chaque navigateur en tant que proxy l'ordinateur-fli4l, vous allez donc prendre le nom de la variable `HOSTNAME='fli4l'` ou l'adresse-IP (par ex. 192.168.6.1) de la variable `HOST_x_IP='192.168.6.1'` qui est dans le fichier config de fli4l. Avec le Port par défaut, on a ici tous les paramètres nécessaires, pour configurer votre navigateur Web, pour l'utilisation de Tor.

**TOR\_ALLOW\_N** Vous indiquez dans cette variable le nombre d'adresse réseau à installer.

**TOR\_ALLOW\_x** Vous indiquez dans cette variable l'adresse réseau ou l'adresse-IP pour le quelle le filtrage de paquets doit être ouvert. Normalement il est logique d'indiquer ici le paramètre `IP_NET_1`.

**TOR\_CONTROL\_PORT** Vous indiquez dans cette variable, le port TCP que Tor doit utiliser, pour le contrôle d'accès via le protocole Tor. Cette variable est optionnelle, si rien n'ai indiqué cette fonction sera désactivée.

**TOR\_CONTROL\_PASSWORD** Vous spécifier dans cette variable, un mot de passe pour le contrôle d'accès.

**TOR\_DATA\_DIR** Cette variable est optionnelle. Si rien n'est indiqué, le dossier par défaut `/etc/tor` est utilisé.

**TOR\_HTTP\_PROXY** Si vous voulez utiliser en plus de Tor un autre Proxy http, Tor pourra alors utiliser ce proxy. Avec cette variable vous avez l'avantage utilisé plusieurs Proxys. Le paramètre peut ressembler à cela :

```
TOR_HTTP_PROXY='mein.provider.de:8000'
```

Ce paramètre est optionnel.

**TOR\_HTTP\_PROXY\_AUTH** Une authentification peut-être nécessaire, vous devez la spécifier dans cette variable. Ainsi le mandataire sera enregistré sous la forme Nom d'utilisateur :Mot de passe.

**TOR\_HTTPS\_PROXY** Vous pouvez enregistrer dans cette variable, un Proxy-HTTPS. Voir [TOR\\_HTTP\\_PROXY](#).

**TOR\_HTTPS\_PROXY\_AUTH** Voir pour ce sujet [TOR\\_HTTP\\_PROXY\\_AUTH](#).

**TOR\_LOGLEVEL** On indique dans cette variable les valeurs pour que Tor puisse enregistrer les événements dans le fichier log. Les valeurs suivantes sont possibles : debug, info, notice, warn ou err. Les valeurs debug et info ne devraient pas si possible être utilisées, pour des raisons de sécurité.

**TOR\_LOGFILE** Si vous voulez utiliser un autre système que syslog pour enregistrer les événements de Tor, vous devez l'indiquer dans cette variable.

Vous pouvez aussi indiquer 'auto', le chemin du fichier log sera alors déplacé dans le répertoire système, pour avoir des données persistantes. S'il vous plaît, assurez-vous que la variable FLI4L\_UUID soit dans ce cas configuré correctement. Comme on peut si attendre une grandes quantités de données sera enregistrées et le fichier log dans le /boot ou dans le Disque-RAM sera rempli rapidement.

### 1.1.3 OPT\_SS5 - Proxy Socks 4/5

Il est nécessaire d'installer le Proxy-Socks pour certains programmes, nous mettons à disposition ici le protocole SS5. Voir le site Web.

<http://ss5.sourceforge.net/>

#### **SS5\_LISTEN\_N**

**SS5\_LISTEN\_x** Dans la première variable vous indiquez le nombre d'adresse réseau, dans la deuxième variable vous indiquez l'adresse-IP ou le nom symbolique, y compris le numéro de Port de l'interface, sur lequel SS5 doit écouter les connexions des Clients. C'est une bonne idée, d'indiquer ici seulement les adresses des interfaces que l'on fait confiance, car tous les ordinateurs auront un accès complet à travers SS5 (avec bien sur le navigateur configuré et activé). En règle générale il est judicieux d'indiquer, la valeur par défaut qui est IP\_NET\_1\_IPADDR:8050

Avec l'adresse indiquée ici, SS5 écoute et offre ses services. Le port par défaut est 8050. Vous devez utiliser cette information pour configurer le proxy dans votre navigateurs.

Vous devez indiquer dans chaque navigateur en tant que proxy l'ordinateur-fli4l, vous allez donc prendre le nom de la variable HOSTNAME='fli4l' ou l'adresse-IP (par ex. 192.168.6.1) de la variable HOST\_x\_IP='192.168.6.1' qui est dans le fichier config de fli4l. Avec le Port par défaut, on a ici tous les paramètres nécessaires, pour configurer votre navigateur Web, pour l'utilisation de SS5.

**SS5\_ALLOW\_N** Vous indiquez dans cette variable le nombre d'adresse réseau à installer.

**SS5\_ALLOW\_x** Vous indiquez dans cette variable l'adresse réseau ou l'adresse-IP pour le quelle le filtrage de paquets doit être ouvert. Normalement il est logique d'indiquer ici le paramètre IP\_NET\_1.

### 1.1.4 OPT\_TRANSPROXY (Expérimental) - Proxy HTTP transparent

Transproxy est un proxy "transparent", c'est une application qui permet, d'intercepter toutes les requêtes-HTTP qui passent par le routeur et de les transmettre à un Proxy-HTTP normal, par ex. au Privoxy. Pour parvenir à cette fonctionnalité, le filtre de paquets des requêtes HTTP qui doit aller sur Internet, passent par le transproxy celui-ci les traite et les transmet à un Proxy-HTTP. Iptables supporte cette fonction en utilisant le paramètre "REDIRECT" :

```
PF_PREROUTING_1='tprl:http IP_NET_1 REDIRECT:8081'
```

Cette règle transmet tous les paquets-HTTP du premier réseau défini (normalement c'est le LAN interne) au transproxy par le port 8081.

#### **TRANSPROXY\_LISTEN\_N**

**TRANSPROXY\_LISTEN\_x** Vous indiquez ici, les adresses IP ou les noms symboliques, ainsi que le numéro de port des interfaces, sur lesquels Transproxy doit écouter les connexions des clients. Si Toutes les interfaces spécifiées ici, utilise déjà un filtrage de paquets, Transproxy sera gêné par les paquets qui provient de ce filtrage. Avec la configuration par défaut `any:8081` Transproxy écouterait toutes les interfaces.

#### **TRANSPROXY\_TARGET\_IP**

**TRANSPROXY\_TARGET\_PORT** Grâce à cette option vous définissez le service pour lequel les requêtes HTTP doivent être redirigé. Cela peut être n'importe quel proxy standard HTTP (Squid, Privoxy, Apache, etc) et sur n'importe quel ordinateur (ou sur fli4l lui-même). Faire attention, que le Proxy ne se trouve pas dans le domaine du filtrage de paquet, dans lequel les requêtes http seront redirigées. Autrement un bouclage d'adresse apparaîtra.

#### **TRANSPROXY\_ALLOW\_N**

**TRANSPROXY\_ALLOW\_x** Liste des réseaux et/ou des adresses IP pour le filtrage de paquets ouvert. Cela devrait couvrir mêmes les réseaux, qui sont redirigés par le filtrage de paquets. Si aucun domaine n'est indiqué ici, vous devez indiquer les informations manuellement dans la configuration du filtrage de paquets.

### **1.1.5 OPT\_SIPROXD (Expérimental) - Proxy pour Session Initiation Protocol**

Si vous souhaitez utiliser plusieurs applications SIP (Ekiga, x-lite ou du matériel téléphonique SIP) qui fonctionnent derrière le routeur, il peut arriver que vous devez transmettre spécifiquement les ports réseau. Autrement, les connexions ne fonctionnent pas comme ils le devraient.

Pour éviter cela, on peut utiliser un proxy SIP spécial. Plusieurs de ces proxys sont en cours d'évaluation pour (fli4l V4.0.0). Si quelqu'un a des suggestions, il ne doit pas hésiter à contacter l'équipe!

### **1.1.6 OPT\_IGMPProxy - Proxy pour Internet Group Management Protocol**

Depuis quelques années Telekom AG Allemand utilise le VDSL 25/50 (bande passante : 25/50 Mbit/s) pour envoyer des paquets de divertissement. Ainsi, il est possible de recevoir la télévision par Internet (IPTV).

La distribution de la télévision sur IP est effectuée en utilisant le multicast (ou la multidiffusion), à savoir, un émetteur source unique vers un groupe (fermé). Pour l'organisation de la diffusion de groupe le protocole réseau IGMP (Internet Group Management Protocol) est nécessaire. L'IGMP ([http://fr.wikipedia.org/wiki/Internet\\_Group\\_Management\\_Protocol](http://fr.wikipedia.org/wiki/Internet_Group_Management_Protocol)) offre la possibilité de gérer dynamiquement des groupes multicast. L'administration ne se trouve pas dans la station d'émission, mais dans le routeur sur laquelle les destinataires du groupe multicast sont connectés directement. L'IGMP fournit des fonctions par lesquelles une station notifie au routeur qu'il peut recevoir des paquets IP multicast pour un groupe multicast particulier.



Les routeurs Speedport (actuellement W700V/W701V/W722) fournissent un support IGMP. Au lieu d'utiliser des routeurs Speedport, vous pouvez utiliser fli4l avec le support IPTV, pour cela vous devez installer le proxy IGMP sur le routeur fli4l.

La documentation du paquetage OPT\_IGMP décrit la configuration de fli4l avec une connexion VDSL et IPTV, vous devez avoir un décodeur (ou STB) X300T/X301T ou MR-303 derrière le routeur fli4l pour fonctionner. Dans cette documentation, on utilise une carte supplémentaire pour l'installation de l'IPTV via le réseau.

### Condition préalable

Telekom VDSL Allemand est présenté comme un VLAN. Dans la phase de lancement (démarrage du réseau) un seul lien VLAN (ID7) a été utilisé, sur lequel tout le trafic circulait. Après des modifications (sur le réseau) deux liens VLAN (ID7, ID8) sont utilisés, le lien ID7 reste pour le trafic Internet et le nouveau lien ID8 est utilisé exclusivement pour le trafic du multicast IPTV. Actuellement les modifications du VDSL pour le réseau (deux liens VLAN ID7/ID8) sont en grande partie terminées.

Hardware (avec un Set-Top-Box (ou boîtier décodeur) et un modem-VDSL) :

- Hardware pour fli4l : avec un VDSL 25/50 vous pouvez utiliser un processeur 486. Si vous avez des problèmes avec l'image/son le matériel utilisé est trop faible.
- Carte réseau haut de gamme (par exemple : 3Com, Intel PRO100). Le chipset Realtek est à mon avis un composant bas de gamme

Software :

- Paquetage : `advanced_networking`
- Paquetage : `dhcp_client` (pour le réseau et l'utilisation du lien ID8)

La configuration des fichiers de (`base.txt`, `dsl.txt`, `advanced_networking.txt`, `dhcp_client.txt`, `dns_dhcp.txt`) sont décrites dans ce manuel.

### Configuration du hardware

Recommandation avec le routeur Speedport et une connexion au boîtier décodeur IPTV directement sur le routeur sans autres éléments dans le réseau, bien sûr, cela s'applique également à fli4l. Néanmoins, si un nœud est interposés (comme un concentrateur, commutateur, pont, passerelle, routeur) entre le boîtier IPTV et le routeur, il doit être compatible avec le multicast, pour éviter les interférences.

Le commutateurs (ou switch) est en général pas utilisé dans le réseau domestique, le réseau virtuel (VLAN) sert à soulager le trafic avec le lien (ID7) et le lien (ID8) pour le trafic multicast de l'IPTV.

Vous pouvez aussi, pour la configuration du hardware de fli4l utiliser des cartes réseaux séparés (une carte d'interface réseau = LAN ou carte Ethernet) et une carte pour la connexion du boîtier décodeur pour le trafic multicast, cela soulagera le reste de votre réseau domestique et écartera les problèmes par rapport à la configuration ci-dessus. Pour ceux qui préfèrent la méthode avec une 'simple' carte réseau, ils doivent savoir se qu'ils font (car elle n'est pas décrite ici).

Voici deux schémas un avec le routeur par défaut et un avec 3 cartes réseaux installées dans le routeur fli4l :

- Configuration par défaut :

- La carte réseau eth0 est configurée dans base.txt pour le LAN interne de la maison/bureau.
- La carte réseau eth1 est configurée dans dsl.txt pour l'interface DSL.



FIGURE 1.1 – fli4l avec la configuration par défaut

- Configuration avancée avec une carte réseau pour l'IPTV :
- Après l'installation d'une carte réseau supplémentaire dans le routeur fli4l, vous devez configurer la carte supplémentaire eth2 dans base.txt

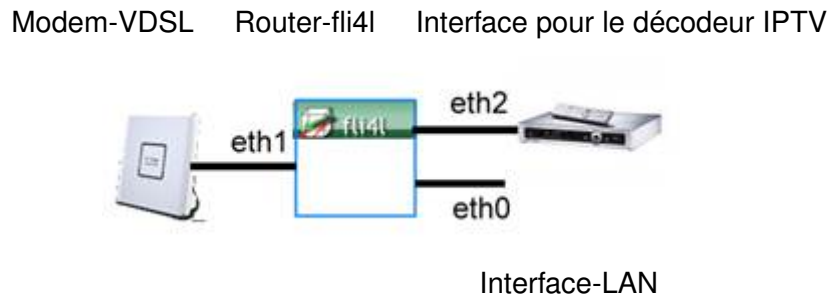


FIGURE 1.2 – fli4l avec la configuration IPTV

## Configuration du VLAN

Tout d'abord : l'OPT\_IGMP ne dépend pas du VLAN. Le VLAN est plutôt utilisé par Deutsche Telekom pour le VDSL et doit être supporté par le routeur. Si le VLAN est nécessaire pour d'autres fournisseurs Internet (Arcor, Alice, etc ..), la configuration est actuelle au-delà de mes connaissances.

Pour que Internet fonctionne avec le VDSL 25/50 de T-Home, la carte réseau qui est connectée au modem VDSL doit être configuré comme une interface VLAN.

*Une note pour ceux qui ont seulement le 'DSL normal', c'est à dire ADSL, ADSL2, ADSL2+ : le VLAN est nécessaire uniquement avec le VDSL, mais pas avec le 'DSL normal'. La configuration du VLAN ne doit pas être installé avec l'utilisation du 'DSL normal'.*

Si vous utilisez deux lien VLAN (voir ci-dessus) le trafic se répartit comme ceci :

- VLAN ID7 : Trafic-Internet
- VLAN ID8 : Trafic-IPTV Multicast

Donc, le trafic Internet fonctionne indépendamment du trafic IPTV. La principale différence, il est nécessaire d'utiliser le VLAN ID7 pour les accès entrant PPPoE. Le VLAN ID8 est fournie par un serveur DHCP sans accès entrant. Dans cette architecture, il n'y a pas de redémarrage

forcé après 24 heures.

Pour le VLAN la configuration suivante est requise (la carte réseau est indiquée à la section configuration du matériel) :

#### **advanced\_networking.txt**

```
VLAN_DEV_N='2'
VLAN_DEV_1_DEV='eth1'      # interface of VDSL-Modem; example: eth1
                             # in our example 'eth1' connects to the VDSL modem
VLAN_DEV_1_VID='7'         # ID7 to support VLAN for internet
VLAN_DEV_2_DEV='eth1'      # interface of VDSL modem; example: eth1
VLAN_DEV_2_VID='8'         # ID8 to support VLAN for IPTV
```

La carte réseau virtual eth1.7 doit être paramétré dans le fichier de configuration DSL :

#### **dsl.txt**

```
PPPOE_ETH='eth1.7'        # eth<number of the card connecting the vdsl modem>.7'
                             # i.e. 'eth1.7'
```

Pour la carte réseau virtual eth1.8 vous aurez besoin d'un client\_dhcp, car le VLAN ID8 est configuré par un serveur DHCP sans accès entrant.

#### **dhcp\_client**

```
OPT_DHCP_CLIENT='yes'
DHCP_CLIENT_TYPE='dhcpcd'
DHCP_CLIENT_INTERFACES='IP_NET_3_DEV' # listen on interface eth1.8
DHCP_CLIENT_USEPEERDNS='no'
DHCP_CLIENT_HOSTNAME=''
```

Depuis la v3.3 de fli4l, on ne peut plus définir l'interface avec cette valeur **eth1.8**, mais vous devez utiliser **IP\_NET\_x\_DEV** pour définir l'interface depuis le fichier base.txt ; Ici **IP\_NET\_3\_DEV**.

Facultatif :

Si la carte réseau que vous utilisez a des problèmes avec la taille du MTU, vous pouvez là régler dans la variable **DEV\_MTU**. Pendant les testes, la carte Intel Pro/100 (e100) et la 3-Com ont montrées aucun problème, d'autres utilisateurs ont signalés des problème avec la carte 3Com '3c59x', ils ont modifiés la valeur MTU à 1496.

```
DEV_MTU_1=''               # Adjust MTU size of NIC on VDSL-Modem
                             # Example: DEV_MTU_1='eth1 1496'
```

Les fichiers de configuration base.txt et dns\_dhcp.txt doivent être modifiées, comme décrit dans le chapitre suivant.

## Configuration de la carte réseau supplémentaire pour l'IPTV

Vous devez configurer base.txt et dns\_dhcp.txt pour paramétrer la deuxième carte réseau et le VLAN.

Paramétrage de la deuxième carte réseau pour l'IPTV :

```
NET_DRV_N='2'
NET_DRV_1='via-rhine'      # 1. NIC interface for LAN
NET_DRV_2='3c59x'         # 2. NIC - here 3Com for IPTV SetTopBox
```

Maintenant nous devons spécifier la plage d'adressage pour la deuxième carte réseau. Nous allons utiliser pour le réseau local 192.168.2.0/24 et 192.168.3.0/24 pour la deuxième carte réseau. Nous avons besoin également de paramétrer les cartes virtuelle pour eth1.7 et eth1.8 :

```
IP_NET_N='4'
IP_NET_1='192.168.2.1/24'  # home/office LAN
IP_NET_1_DEV='eth0'
IP_NET_2='192.168.3.1/24'  # iptv LAN
IP_NET_2_DEV='eth2'
IP_NET_3='dhcp'           # dhcp client - IP via dhclient
IP_NET_3_DEV='eth1.8'
IP_NET_3_MAC='00:40:63:da:cf:32' # new MAC (not the MAC of eth1)
IP_NET_4='dhcp'           # eth1.7 connecting to the modem
IP_NET_4_DEV='eth1.7'
IP_NET_4_MAC='00:40:63:da:cf:33' # new MAC (not the MAC of eth1)
```

Il est important de changer les adresses MAC pour eth1.7 et eth1.8, elles ne doivent pas coïncider avec eth1, sinon - selon le réseau VDSL - il peut éventuellement se produire des perturbations après la déconnexion forcée.

Pour que la nouvelle carte réseau puisse accéder à Internet, bien sûr, tout comme pour la première carte réseau. Ces paramètres supplémentaires sont nécessaires :

```
PF_INPUT_1='IP_NET_1 ACCEPT'
PF_INPUT_2='IP_NET_2 ACCEPT'
PF_INPUT_3='any 224.0.0.0/4 ACCEPT'
[...]
PF_FORWARD_3='any 224.0.0.0/4 ACCEPT'
PF_FORWARD_5='IP_NET_1 ACCEPT'
PF_FORWARD_6='IP_NET_2 ACCEPT'
[...]
PF_POSTROUTING_1='IP_NET_1 MASQUERADE'
PF_POSTROUTING_2='IP_NET_2 MASQUERADE'
```

Pour que cela fonctionne vous devez paramétrer l'adressage DHCP dynamique sur la carte réseau IPTV, pour accéder à la Set-Top Box (ou décodeur) et lui donner un nom. Les paramètres suivants dans le dns\_dhcp.txt sont nécessaires :

```
HOST_10_NAME='igmp'
HOST_10_IP4='192.168.3.1'
HOST_11_NAME='iptv'
HOST_11_IP4='192.168.3.4'
HOST_11_MAC='00:D0:E0:93:49:34'          # MAC Adr T-Home X300T
[...]
```

```
DHCP_RANGE_2_NET='IP_NET_2'
DNSDHCP_RANGE_2_START='192.168.3.10'
DNSDHCP_RANGE_2_END='192.168.3.20'
DNSDHCP_RANGE_2_DNS_SERVER1=''
DNSDHCP_RANGE_2_DNS_SERVER2=''
DNSDHCP_RANGE_2_NTP_SERVER=''
DNSDHCP_RANGE_2_GATEWAY=''
```

Après avoir configuré la nouvelle carte réseau, il est judicieux de tester l'accès Internet avec un PC connecté au routeur. Si vous arrivez à vous connecter, la seconde carte réseau sera configuré correctement.

## Fonctions de l'IGMP

Lors du démarrage du routeur fli4l les paramètres du fichier de configuration proxy.txt sont écrit dans le fichier /etc/igmpproxy.conf, ils sont ensuite lu lors du démarrage du proxy IGMP.

Contrairement aux versions antérieures le paquetage opt\_igmp est lancé au démarrage, il est exécute aussi longtemps que la connexion Internet physique est disponible. Le proxy IGMP n'est pas affectée par la déconnexion forcé après 24 heures ou par la connexion/déconnexion manuel de l'accès Internet.

## Configuration de l'IGMP

**OPT\_IGMPPROXY** Si vous indiquez 'yes', vous activez le proxy IGMP. avec 'no' vous désactivez l'ensemble du paquetage.

**IGMPPROXY\_DEBUG** Si vous indiquez 'yes' les messages du proxy IGMP sont envoyés à syslog.

**IGMPPROXY\_DEBUG2** Si vous indiquez 'yes' vous pouvez augmenter le niveau des messages du proxy IGMP.

**IGMPPROXY\_QUICKLEAVE\_ON** Avec Quickleave vous pouvez abaisser la charge de la liaison montante. Si vous avez indiqué 'yes', le multicast sera arrêté plus rapidement après un changement de canal et la charge de la liaison descendante sera réduite par le proxy IGMP, il se comporte comme un récepteur.

Si vous utilisez deux décodeurs et qu'il sont sur le même canal, il peut arriver (avec l'activation de Quickleave) que le programme soit interrompu sur l'un des décodeurs, vous devez alors changer le programme. Si vous utilisez un seule décodeur Quickleave peut être activé en toute sécurité.

```
IGMPPROXY_QUICKLEAVE_ON='yes'          # activate Quickleave mode
                                         # yes or no; Default: yes
```

**IGMPPROXY\_UPLOAD\_DEV** Pour le fonctionnement de l'IPTV avec le proxy IGMP vous avez besoin d'une interface avec une liaison montante et descendante. L'interface de

la liaison montante est l'interface de la carte réseau qui sera connecté au modem VDSL. Normalement cela doit toujours être la même.

Le transfert de l'IPTV doit se faire sur le lien ID8 avec eth1.8, au lieu de ppp0. Cela doit être paramétré dans le fichier de configuration.

```
IGMPPROXY_UPLOAD_DEV='eth1.8'      # Upstream interface; Default: ppp0
                                     # eth1.8 for T-Home/VDSL with id7/id8
```

**IGMPPROXY\_DOWNLOAD\_DEV** L'interface de la liaison descendante (carte réseau pour le décodeur de l'IPTV) dépend de la configuration matériel. Dans fli4l la deuxième carte réseau eth2 est l'interface pour le décodeur.

```
IGMPPROXY_DOWNLOAD_DEV='eth2'      # Downstream Interface
```

**IGMPPROXY\_ALT\_N** Dans cette variable vous indiquez le nombre de plages d'adresse IP pour le flux multicast.

**IGMPPROXY\_ALT\_x\_NET** Dans la variable IGMPPROXY\_ALT\_NET vous indiquez les adresses IP pour le trafic multicast provenant de extérieur pour le réseau local, ainsi que l'adresse IP local du décodeur.

```
IGMPPROXY_ALT_N='3'                # Number of Multicast sources
IGMPPROXY_ALT_1_NET='239.35.0.0/16' # IPTV streams - always needed
IGMPPROXY_ALT_2_NET='217.0.119.0/24' # needed for T-Home
IGMPPROXY_ALT_3_NET='193.158.34.0/23' # needed for T-Home
                                     # before May 2013 '193.158.35.0/24'
# IGMPPROXY_ALT_4_NET='192.168.3.0/24' # Address range IPTV SetTop-Box/not
                                     # needed anymore
```

**IGMPPROXY\_WLIST\_N** Dans cette variable vous indiquez le nombre de listes blanches pour le rapport IGMP.

**IGMPPROXY\_WLIST\_x\_NET :**

Si vous utilisez IGMPv3 toutes les adresses sont regroupées dans un rapport, (<http://grinch.itg-em.de/entertain/artikel/zielnetzarchitektur-und-igmpproxy/>). Ces adresses seront ensuite complètement ignorées. Cela conduira à un arrêt complet de tout le trafic multicast par l'émetteur IGMP, en supposant qu'ils ne sont plus nécessaires. Pour éviter cela, la configuration de listes blanches est utilisée. Seuls les groupes multicast de cette liste seront épargnés par le WAN.

```
IGMPPROXY_WLIST_N='1'              # Number of Multicast sources
IGMPPROXY_WLIST_1_NET='239.35.0.0/16' # IPTV streams - always needed
                                     # see above
```

### Modification des autres fichiers de configuration

Avec la révision 32955 il n'est pas nécessaire de paramétrer les règles du pare-feu pour le proxy IGMP et pour le flux multicast si les règles standard (PF\_INPUT\_ACCEPT\_DEF='yes' et PF\_FORWARD\_ACCEPT\_DEF='yes') sont activés dans le fichier base.txt, le script de démarrage ajoutera automatiquement ces règles si la variable OPT\_IGMPPROXY='yes' est activée.

Il y a deux règles qui sont ajouté dans la chaîne INPUT pour permettre aux messages entrants d'atteindre le proxy IGMP :

```
Chain INPUT (policy DROP 0 packets, 0 bytes)
pkts bytes target  prot opt in  out  source      destination
[...]
0      0 ACCEPT  all  --  *   *    0.0.0.0/0    224.0.0.1    \
/* automatically added for IGMP Proxy */

0      0 ACCEPT  all  --  *   *    0.0.0.0/0    224.0.0.22   \
/* automatically added for IGMP Proxy */
[...]
```

Vous avez aussi une règle dans la chaîne FORWARD qui permet de transmettre le flux multicast entrant vers le récepteur de média :

```
Chain FORWARD (policy DROP 0 packets, 0 bytes)
pkts bytes target  prot opt in  out  source      destination
[...]
0      0 ACCEPT  all  --  *   *    0.0.0.0/0    239.35.0.0/16 \
/* automatically added for IPTV streams */
[...]
```

Si l'une des règles par défaut n'est pas activée, vous devez au moins paramétrer et insérer manuellement les règles suivantes.

```
PF_INPUT_x='any 224.0.0.1/32 ACCEPT'
PF_INPUT_x='any 224.0.0.22/32 ACCEPT'
[...]
PF_FORWARD_x='any 239.35.0.0/16 ACCEPT'
```

**Remarque :** contrairement aux versions précédentes de la documentation, les règles réellement nécessaires ont été écrites pour un réseau limité. Si l'IPTV ne fonctionne pas, n'hésitez pas à fournir des informations supplémentaires concernant le réseau que vous utilisez.

**Important !** Depuis la fin du mois de mai 2013, Telekom a introduit de nouvelles routes (sans classe prédéfini) pour améliorer le service (<http://www.onlinekosten.de/forum/showthread.php?t=116415&page=38>). Cela est une bonne chose, car plus de 256 émetteurs ou adresses sont utilisables. Maintenant le serveur DHCP transfère les routes, elles ne sont plus incluses dans le sous-réseau comme précédemment. Tant que Telekom ne change pas son sous-réseau du serveur-IPTV (193.158.34.0/23) la route statique peut être définie vers l'interface vlan8, vous devez faire attention au changement de celle-ci, sinon le multicast ne fonctionnera plus.

Solution : Dans le fichier base.txt, vous devez spécifier une route supplémentaire.

```
IP_ROUTE_N='1'
IP_ROUTE_1='193.158.34.0/23 eth1.8'
```

### 1.1.7 OPT\_STUNNEL - Tunnel avec une connexion SSL/TLS

Le programme "stunnel" permet d'encapsuler les connexions non cryptées dans un tunnel SSL/TLS crypté. Ce protocole permet d'échanger du texte clair en toute sécurisée. Grâce aux possibilités du protocole SSL/TLS, vous pouvez paramétrer les différents modes Client/Serveur.

## Configuration

**OPT\_STUNNEL** Cette variable vous permet d'utiliser un tunnel SSL/TLS.

Configuration par défaut : `OPT_STUNNEL='no'`

Exemple : `OPT_STUNNEL='yes'`

**STUNNEL\_DEBUG** Avec cette variable vous enregistrez paramètres de fonctionnement du "stunnel" les valeurs disponibles sont "yes" (tout est enregistré), "no" (les avertissements et les erreurs sont enregistrés), vous pouvez aussi indiquer une valeur comprise entre zéro et sept, cela spécifie la gravité maximale d'enregistrement des messages, si vous indiquez zéro vous enregistrez tous les messages d'urgences et sept tous les messages de débogage. Le paramètre "yes" correspond à sept pour la gravité maximale et "no" correspond à quatre pour la gravité maximale.

Configuration par défaut : `STUNNEL_DEBUG='no'`

Exemple 1 : `STUNNEL_DEBUG='yes'`

Exemple 2 : `STUNNEL_DEBUG='5'`

**STUNNEL\_N** Avec cette variable vous configurez le nombre de tunnel. Chaque instance de tunnel "écoute" sur un port du réseau "A" et utilise une connexion entrante sur un autre port du réseau "B" (qui peut également être situé sur une autre machine) avant tout trafic de "A" vers "B". Les données qui partent de "A" via le SSL/TLS sont cryptées, ils passent par "stunnel", les données sont décodées par "B" ou vice versa et ensuite il sont transmis, on paramètre cette fonction avec la variable `STUNNEL_x_CLIENT` (Page 16).

Configuration par défaut : `STUNNEL_N='0'`

Exemple : `STUNNEL_N='2'`

**STUNNEL\_x\_NAME** Dans cette variable vous indiquez le nom du tunnel. Ce nom doit être unique pour chaque tunnel configuré.

Exemple : `STUNNEL_1_NAME='imond'`

**STUNNEL\_x\_CLIENT** Dans cette variable vous pouvez régler le tunnel qui communiquera en SSL/TLS crypté. Il y a deux options :

- *Mode client* : il reçoit les données non cryptées qui proviennent du tunnel distant et enverra les données cryptées à l'autre extrémité du tunnel. Pour cela vous devez indiquer `STUNNEL_x_CLIENT='yes'`.
- *Mode serveur* : il reçoit les données cryptées via le SSL/TLS, il procède au décodage et renvoie le résultat à l'autre extrémité du tunnel. Pour cela vous devez indiquer `STUNNEL_x_CLIENT='no'`.

Le tunnel en mode client est particulièrement adapté pour des connexions qui vont vers l'extérieur, par ex. pour un accès Internet (non protégé), les données seront cryptées avant de quitter le réseau local. Le site incontournable distant doit bien sûr avoir également un serveur avec le service SSL/TLS pour recevoir les données chiffrées. Exemple d'un client e-Mail sur le LAN, le protocole POP3 "parle" en utilisant des données non cryptées, vous pouvez utiliser un compte POP3 avec le service SSL sur Internet.<sup>1</sup>

Le tunnel en mode serveur est à l'inverse adapté pour des connexions qui "proviennent de l'extérieur", par ex. des données qui viennent Internet (non protégé), les données seront décodées lorsqu'ils l'arriveront. Si le serveur distant n'a pas de service SSL/TLS

---

1. Voir [http://fr.wikipedia.org/wiki/Post\\_Office\\_Protocol](http://fr.wikipedia.org/wiki/Post_Office_Protocol)



les données doivent être décodées avant l'envoi. Exemple, pour accéder sur l'interface web de fli4l via le HTTP (HTTPS) avec le service SSL/TLS crypté à travers le tunnel, fli4l est configuré pour recevoir via le SSL/TLS les données cryptées sur le port 443, ils seront décodées puis les transmet au serveur Web du `mini_httpd`, qui écoute sur le port 80.

La configuration pour ces applications sont représentées un peu plus bas.

Exemple : `STUNNEL_1_CLIENT='yes'`

**STUNNEL\_x\_ACCEPT** Dans cette variable vous pouvez régler le port (et l'adresse) du tunnel pour "écouter" les connexions entrantes. Il y a deux possibilités :

- Si le tunnel écouter *toutes* les adresses (sur toutes les interfaces). Vous devez indiquer "any" dans cette variable.
- Si le tunnel écoute que sur des adresses spécifiques. Vous devez indiquer la référence appropriée avec l'IP du sous-réseau configurée, par exemple `IP_NET_1_IPADDR` (pour IPv4) ou `IPV6_NET_2_IPADDR` (pour IPv6).

En outre, vous *devez* indiquer derrière l'adresse le numéro de port, pour cela vous devez placer les deux-points (":") entre l'adresse et le port.

Exemple 1 : `STUNNEL_1_ACCEPT='any:443'`

Exemple 2 : `STUNNEL_1_ACCEPT='IP_NET_1_IPADDR:443'`

Exemple 3 : `STUNNEL_1_ACCEPT='IPV6_NET_2_IPADDR:443'`

Il convient de rappeler lors de l'utilisation d'une valeur dans la variable `IP_NET_x_IPADDR` ou dans `IPV6_NET_x_IPADDR` elle est de couche 3 du protocole (IPv4 ou IPv6). Le choix *doit* correspondre à l'affectation des variables `STUNNEL_x_ACCEPT_IPV4` et `STUNNEL_x_ACCEPT_IPV6`. Donc vous ne pouvez pas désactiver un tunnel IPv6 `STUNNEL_1_ACCEPT_IPV6='no'` et écouter sur une adresse IPv6 avec `STUNNEL_1_ACCEPT='IPV6_NET_2_IPADDR:443'` ; Ceci s'applique également pour la configuration inverse (`STUNNEL_1_ACCEPT_IPV4='no'` en utilisant `IP_NET_x_IPADDR`). En outre, cela dépend aussi du sens du paramètre "any", si vous utilisez la couche 3 du protocole (IPv4 ou IPv6), il écoutera seulement sur des adresses activées, via `STUNNEL_x_ACCEPT_IPV4` et `STUNNEL_x_ACCEPT_IPV6` du protocoles de couche 3.

**STUNNEL\_x\_ACCEPT\_IPV4** Dans cette variable vous pouvez indiquer le protocole IPv4 qui peut être utilisé pour la connexion *entrante* du tunnel. Généralement, cette variable devrait contenir la valeur "yes". Si vous indiquez "no" le protocole IPv6 sera utilisé pour la connexion entrante du tunnel. Cependant, cela nécessite une configuration IPv6 valide (reportez-vous à la documentation du paquetage IPv6).

Configuration par défaut : `STUNNEL_x_ACCEPT_IPV4='yes'`

Exemple : `STUNNEL_1_ACCEPT_IPV4='no'`

**STUNNEL\_x\_ACCEPT\_IPV6** Cette variable est analogue à la variable `STUNNEL_x_ACCEPT_IPV4`, si le protocole IPv6 est utilisé pour la connexion entrante vous devez indiquer la même valeur pour activer le protocole IPv6 dans le paquetage avec `OPT_IPV6='yes'`. Si vous indiquez "no" le protocole IPv4 sera utilisé pour la connexion entrante du tunnel.

Configuration par défaut : `STUNNEL_x_ACCEPT_IPV6=<valeur de OPT_IPV6>`

Exemple : `STUNNEL_1_ACCEPT_IPV6='no'`

**STUNNEL\_x\_CONNECT** Dans cette variable vous indiquez le nom ou l'adresse SSL/TLS du tunnel distant. Cela donne en principe trois possibilités, vous devez ajouter deux points ":" à la suite de ces trois possibilités et indiquer le numéro de port :

- L'adresse IPv4 ou IPv6 numérique.  
Exemple 1 : `STUNNEL_1_CONNECT='192.0.2.2:443'`
- Le nom du DNS d'un hôte interne.  
Exemple 2 : `STUNNEL_1_CONNECT='@webserver:443'`
- Le nom du DNS d'un hôte externe.  
Exemple 3 : `STUNNEL_1_CONNECT='@www.example.com:443'`

Si un hôte interne est indiqué, à la fois pour une adresse IPv4 et IPv6, l'adresse IPv4 sera privilégiée. Si un hôte externe est indiqué, à la fois pour une adresse IPv4 et IPv6, cela dépend du retour de l'adresse du protocole de couche 3, le premier DNS résolu sera utilisé.

**STUNNEL\_x\_OUTGOING\_IP** Dans cette variable optionnelle vous pouvez indiquer l'adresse locale pour la connexion sortante du tunnel. Cette variable est utilisée que si le tunnel distant a de multiples interfaces (ou routes) actif, par exemple si la cible utilise deux connexions Internet. Normalement, cette variable ne doit pas être configurée.

Exemple : `STUNNEL_1_OUTGOING_IP='IP_NET_1_IPADDR'`

**STUNNEL\_x\_DELAY\_DNS** Si cette variable optionnelle est activée "yes", le nom de DNS externe utilisé dans `STUNNEL_x_CONNECT` sera converti en une adresse IP si la connexion *sortante* du tunnel est construite. Ainsi client local sera connecté à la première adresse qui travers le tunnel. Cette variable est utile que si le tunnel distant, est un ordinateur qui ne peut être atteint que par le nom du DNS dynamique et si les changements d'adresse sont fréquent derrière ce nom ou si une connexion active empêche le démarrage du "stunnel".

Configuration par défaut : `STUNNEL_x_DELAY_DNS='no'`

Exemple : `STUNNEL_1_DELAY_DNS='yes'`

**STUNNEL\_x\_CERT\_FILE** Dans cette variable vous indiquez le nom du fichier du certificat pour le tunnel. Tunnel en mode serveur, la variable (`STUNNEL_x_CLIENT='no'`) est désactivée, c'est le certificat du serveur qui est validé par le client contre un "Certificate Authority" (CA) si nécessaire. Tunnel en mode client, la variable (`STUNNEL_x_CLIENT='yes'`) est activée, c'est le (généralement en option) certificat du client qui est validé par le serveur contre un CA si nécessaire.

Le certificat doit être dans le format dit PEM et doit être stocké sous le répertoire `<répertoire-config>/etc/ssl/stunnel/`. Seul le nom du fichier doit être stocké dans cette variable, pas le chemin.

Pour un tunnel en mode serveur le certificat est obligatoire!

Exemple : `STUNNEL_1_CERT_FILE='myserver.crt'`

**STUNNEL\_x\_CERT\_CA\_FILE** Dans cette variable vous indiquez le nom du fichier du certificat CA à utiliser pour valider le certificat de la station distante. Les clients valident généralement le certificat du serveur, cependant il est également possible de contourner la validation. Quelques détails pour la validation, s'il vous plaît lire la description de la variable [STUNNEL\\_x\\_CERT\\_VERIFY](#) (Page 18) ci-dessous.

Le certificat doit être dans le format dit PEM et doit être stocké sous le répertoire `<répertoire-config>/etc/ssl/stunnel/`. Seul le nom du fichier doit être stocké dans cette variable, pas le chemin.

Exemple : `STUNNEL_1_CERT_CA_FILE='myca.crt'`

**STUNNEL\_x\_CERT\_VERIFY** Dans cette variable vous commandez la validation du certificat de la station distante. Il y a cinq options :

- *none* : le certificat de la station distante n'est pas du tout validé. Dans ce cas, la variable STUNNEL\_x\_CERT\_CA\_FILE peut rester vide.
- *optional* : le certificat de la station distante est disponible, il sera vérifié en utilisant le certificat CA qui est configuré dans la variable STUNNEL\_x\_CERT\_CA\_FILE. Si la station distante n'a *pas* de certificat disponible, cela ne sera pas indiqué comme une erreur et la connexion sera toujours acceptée. Ce paramètre est utile que pour un tunnel en mode serveur, car le tunnel en mode client doit *toujours* obtenir un certificat à partir du serveur.
- *onlyca* : le certificat de la station distante est vérifié en utilisant le certificat CA, qui est configuré dans la variable STUNNEL\_x\_CERT\_CA\_FILE. Si la station distante ne diffuse aucun certificat ou s'il ne correspond pas au CA configuré, la connexion sera chargée. Ce paramètre est utile si vous utilisez votre propre CA et que vous connaissez tous les stations distantes potentiels.
- *onlycert* : le certificat de la station distante est comparé avec le certificat qui est configuré dans la variable STUNNEL\_x\_CERT\_CA\_FILE. Il ne sera *pas* vérifié en utilisant le certificat CA, mais il veillera que le certificat de la station distante *envoyé* est exactement le même entre le certificat du (serveur ou du client). Le fichier qui est référencé dans la variable STUNNEL\_x\_CERT\_CA\_FILE ne contient pas de CA, mais un certificat hôte. Ce paramètre garantit que seul un serveur déterminé connu (tunnel en mode serveur) peut être construite et connecté à un terminal connu (tunnel en mode client). Ce paramètre est utile pour les connexions peer-to-peer entre hôtes les deux sont sous contrôle et vous n'utilisez pas votre propre CA.
- *both* : le certificat de la station distante est comparé avec le certificat qui est configuré dans la variable STUNNEL\_x\_CERT\_CA\_FILE *et* sera également vérifié en utilisant le certificat CA qui est également dans cette variable. Les *deux* fichiers qui sont référencés dans la variable STUNNEL\_x\_CERT\_CA\_FILE, contient un CA *et* un certificat hôte. C'est une combinaison des options *onlycert* et *onlyca*. En comparaison avec le réglage *onlycert* la connexion sera refusée, si le certificat CA a expiré (même si le certificat de la station distante est validé).

Configuration par défaut : STUNNEL\_x\_CERT\_VERIFY='none'

Exemple : STUNNEL\_1\_CERT\_VERIFY='onlyca'

## 1ère exemple d'application : connexion au WebGUI de fli4l via le SSL/TLS

Avec cet exemple, le WebGUI de fli4l est prolongé par une connexion SSL/TLS.

```
OPT_STUNNEL='yes'
STUNNEL_N='1'

STUNNEL_1_NAME='http'
STUNNEL_1_CLIENT='no'
STUNNEL_1_ACCEPT='any:443'
STUNNEL_1_ACCEPT_IPV4='yes'
STUNNEL_1_ACCEPT_IPV6='yes'
STUNNEL_1_CONNECT='127.0.0.1:80'
STUNNEL_1_CERT_FILE='server.pem'
STUNNEL_1_CERT_CA_FILE='ca.pem'
STUNNEL_1_CERT_VERIFY='none'
```

## 2ème exemple d'application : contrôle de deux routeurs fli4l distant via le SSL/TLS sécurisé, par le programme imonc

Avec cet exemple, vous contournez les faiblesses connues du Protocole imonc/imond (envoi des mots de passe en clair) pour une connexion WAN. (Naturellement, la connexion au LAN avec le tunnel peut continuer à être exploitée!)

Configuration local de fli4l dans le LAN (tunnel en mode client) :

```
OPT_STUNNEL='yes'
STUNNEL_N='2'

STUNNEL_1_NAME='remote-imond1'
STUNNEL_1_CLIENT='yes'
STUNNEL_1_ACCEPT='any:50000'
STUNNEL_1_ACCEPT_IPV4='yes'
STUNNEL_1_ACCEPT_IPV6='yes'
STUNNEL_1_CONNECT='@remote1:50000'
STUNNEL_1_CERT_FILE='client.pem'
STUNNEL_1_CERT_CA_FILE='ca+server1.pem'
STUNNEL_1_CERT_VERIFY='both'

STUNNEL_2_NAME='remote-imond2'
STUNNEL_2_CLIENT='yes'
STUNNEL_2_ACCEPT='any:50001'
STUNNEL_2_ACCEPT_IPV4='yes'
STUNNEL_2_ACCEPT_IPV6='yes'
STUNNEL_2_CONNECT='@remote2:50000'
STUNNEL_2_CERT_FILE='client.pem'
STUNNEL_2_CERT_CA_FILE='ca+server2.pem'
STUNNEL_2_CERT_VERIFY='both'
```

Configuration du premier fli4l distant (tunnel en mode serveur) :

```
OPT_STUNNEL='yes'
STUNNEL_N='1'

STUNNEL_1_NAME='remote-imond'
STUNNEL_1_CLIENT='no'
STUNNEL_1_ACCEPT='any:50000'
STUNNEL_1_ACCEPT_IPV4='yes'
STUNNEL_1_ACCEPT_IPV6='yes'
STUNNEL_1_CONNECT='127.0.0.1:5000'
STUNNEL_1_CERT_FILE='server1.pem'
STUNNEL_1_CERT_CA_FILE='ca+client.pem'
STUNNEL_1_CERT_VERIFY='both'
```

Configuration du second fli4l distant (tunnel en mode serveur) :

```
OPT_STUNNEL='yes'
STUNNEL_N='1'

STUNNEL_1_NAME='remote-imond'
STUNNEL_1_CLIENT='no'
STUNNEL_1_ACCEPT='any:50000'
```

```
STUNNEL_1_ACCEPT_IPV4='yes'  
STUNNEL_1_ACCEPT_IPV6='yes'  
STUNNEL_1_CONNECT='127.0.0.1:5000'  
STUNNEL_1_CERT_FILE='server2.pem'  
STUNNEL_1_CERT_CA_FILE='ca+client.pem'  
STUNNEL_1_CERT_VERIFY='both'
```

La connexion à "imond" distant est construite à travers une connexion fli4l local sur le port 5000 (première fli4l distant) et sur le port 5001 (seconde fli4l distant). fli4l se connecte via le tunnel SSL/TLS au fli4l distant respectif, le démon "imond" transmettra à son tour les données via un tiers (hôte interne) vers une connexion distante. Les réglages garantit la validation de chaque fli4l et acceptera uniquement l'autre fli4l comme partenaire de connexion.

# Table des figures

1.1	fi4l avec la configuration par défaut . . . . .	10
1.2	fi4l avec la configuration IPTV . . . . .	10

## Liste des tableaux

# Index

IGMPPROXY\_ALT\_N, [14](#)  
IGMPPROXY\_ALT\_x\_NET, [14](#)  
IGMPPROXY\_DEBUG, [13](#)  
IGMPPROXY\_DEBUG2, [13](#)  
IGMPPROXY\_DOWNLOAD\_DEV, [14](#)  
IGMPPROXY\_QUICKLEAVE\_ON, [13](#)  
IGMPPROXY\_UPLOAD\_DEV, [13](#)  
IGMPPROXY\_WLIST\_N, [14](#)  
IGMPPROXY\_WLIST\_x\_NET, [14](#)

OPT\_IGMPPROXY, [8](#), [13](#)  
OPT\_PRIVOX, [3](#)  
OPT\_SIPROXD, [8](#)  
OPT\_SS5, [7](#)  
OPT\_STUNNEL, [15](#), [16](#)  
OPT\_TOR, [5](#)  
OPT\_TRANSPROXY, [7](#)

PRIVOX\_MENU, [3](#)  
PRIVOX\_N, [3](#)  
PRIVOX\_x\_ACTIONDIR, [4](#)  
PRIVOX\_x\_ALLOW\_N, [4](#)  
PRIVOX\_x\_ALLOW\_x, [4](#)  
PRIVOX\_x\_CONFIG, [4](#)  
PRIVOX\_x\_HTTP\_PROXY, [4](#)  
PRIVOX\_x\_LISTEN, [3](#)  
PRIVOX\_x\_LOGDIR, [5](#)  
PRIVOX\_x\_LOGLEVEL, [5](#)  
PRIVOX\_x SOCKS\_PROXY, [4](#)  
PRIVOX\_x\_TOGGLE, [4](#)

SS5\_ALLOW\_N, [7](#)  
SS5\_ALLOW\_x, [7](#)  
SS5\_LISTEN\_N, [7](#)  
SS5\_LISTEN\_x, [7](#)  
STUNNEL\_DEBUG, [16](#)  
STUNNEL\_N, [16](#)  
STUNNEL\_x\_ACCEPT, [17](#)  
STUNNEL\_x\_ACCEPT\_IPV4, [17](#)  
STUNNEL\_x\_ACCEPT\_IPV6, [17](#)  
STUNNEL\_x\_CERT\_CA\_FILE, [18](#)  
STUNNEL\_x\_CERT\_FILE, [18](#)  
STUNNEL\_x\_CERT\_VERIFY, [18](#)  
STUNNEL\_x\_CLIENT, [16](#)  
STUNNEL\_x\_CONNECT, [17](#)  
STUNNEL\_x\_DELAY\_DNS, [18](#)  
STUNNEL\_x\_NAME, [16](#)  
STUNNEL\_x\_OUTGOING\_IP, [18](#)

TOR\_ALLOW\_N, [6](#)  
TOR\_ALLOW\_x, [6](#)  
TOR\_CONTROL\_PASSWORD, [6](#)  
TOR\_CONTROL\_PORT, [6](#)  
TOR\_DATA\_DIR, [6](#)  
TOR\_HTTP\_PROXY, [6](#)  
TOR\_HTTP\_PROXY\_AUTH, [6](#)  
TOR\_HTTPS\_PROXY, [6](#)  
TOR\_HTTPS\_PROXY\_AUTH, [6](#)  
TOR\_LISTEN\_N, [6](#)  
TOR\_LISTEN\_x, [6](#)  
TOR\_LOGFILE, [7](#)  
TOR\_LOGLEVEL, [6](#)  
TRANSPROXY\_ALLOW\_N, [8](#)  
TRANSPROXY\_ALLOW\_x, [8](#)  
TRANSPROXY\_LISTEN\_N, [8](#)  
TRANSPROXY\_LISTEN\_x, [8](#)  
TRANSPROXY\_TARGET\_IP, [8](#)  
TRANSPROXY\_TARGET\_PORT, [8](#)