Paquetage IPV6 - Amélioration des fonctions IPv6
Version 3.10.4

Christoph Schulz
courriel: fli4l@kristov.de

L'équipe fli4l
courriel: team@fli4l.de

25 octobre 2015

Table des matières

1. Documentation du paquetage IPV6 **3**
1.1. IPv6 - Internet protocole version 6 . . . 3
1.1.1. Introduction . . . 3
1.1.2. Format de l'adresse . . . 3
1.1.3. Configuration . . . 4
1.1.4. WebGUI . . . 15

A. Annexe du paquetage IPV6 **16**
A.1. IPV6 - Connexion Internet en utilisant un tunnel IPv6 SixXS . . . 16
A.1.1. Créer un compte . . . 16
A.1.2. Configurer le tunnel . . . 16
A.1.3. Configurer le sous-réseaux . . . 18

Table des figures **22**

Liste des tableaux **23**

Index **24**

1. Documentation du paquetage IPV6

1.1. IPv6 - Internet protocole version 6

1.1.1. Introduction

Ce paquetage permet au routeur fli4l avec bien des égards de rendre compatible l'IPv6. Les informations qui sont incluses dans le paquetage IPv6 pour le routeur sont les adresses IPv6, la gestion des (sous-)réseaux IPv6, la route IPv6 prédéfinie et les règles de pare-feu. Vous pouvez aussi configurer le service IPv6 par le DHCPv6. Enfin, il est possible de construire un tunnel automatiquement avec des fournisseurs IPv6. Maintenant cela fonctionne correctement, mais, seulement avec des tunnels 6in4, le fournisseur "SixXS" prend en charge cette technologie. Les autres technologies comme (AYIYA, 6to4, Teredo) ne sont pas encore prisent en charge.

IPv6 est le successeur du protocole Internet IPv4. Il a été principalement conçu pour augmenter la quantité relativement faible des adresses Internet formelles : IPv4 supporte environ 2^{32} d'adresses,[1] avec IPv6 on a déjà 2^{128} d'adresses. Avec la communication IPv6, on peut attribuer une adresse unique pour chaque hôte, et nous ne sommes plus sur des techniques telles que le NAT, le PAT, le Masquerading, etc.

Outre cet aspect, les sujets comme l'autoconfiguration et la sécurité ont aussi joué un rôle lors du développement du protocole IPv6. Ces questions seront traitées dans les sections suivantes.

Le plus gros problème avec IPv6 est sa distribution : Actuellement, l'IPv6 – par rapport à IPv4 – est très peu utilisés. La raison est que le protocole IPv6 et IPv4 ne sont pas techniquement compatibles l'un avec l'autre et par conséquent tous les composants matériels et logiciels, qui sont impliqués dans la transmission de paquets sur Internet pour l'IPv6 doit être installé. Certains services comme le DNS (Domain Name System) pour IPv6 doivent être ouverts en conséquence.

Un cercle vicieux s'ouvre alors : la faible propagation des IPv6 chez les fournisseurs d'accès Internet amène l'indifférence de la part des fabricants à équiper les routeurs d'un dispositif pour le fonctionnement IPv6, cela signifie que les fournisseurs d'accès ont peur de la transition vers IPv6, parce qu'ils craignent qu'un tel effort ne vos pas la peine. Ce n'est que lentement que le vent tourne en faveur de l'IPv6, car des réserves d'adresses IPv4 s'épuisent. [2]

1.1.2. Format de l'adresse

Une adresse IPv6 se compose de huit valeurs de deux octets, elles sont classées en hexadécimal :

Exemple 1 : `2001:db8:900:551:0:0:0:2`

Exemple 2 : `0:0:0:0:0:0:0:1` (IPv6-Loopback-Adresse)

1. c'est seulement approximatif, car certaines adresses ont un objectif bien spécifique, comme le Broadcasting et le Multicasting,
2. Maintenant les derniers blocs d'adresses IPv4 ont été attribués par l'IANA.

Pour réduite l'encombrement des adresses, on peut fusionnée une suite de zéros successifs, en les supprimant et en ajoutant seulement une paire de deux points. Les adresses ci-dessus peuvent également être écrites comme ceci :

Exemple 1 (compacté) : `2001:db8:900:551::2`

Exemple 2 (compacté) : `::1`

Une telle réduction est uniquement autorisée d'une fois, pour éviter toute ambiguïté. L'adresse `2001:0:0:1:2:0:0:3` peut être réduite comme ceci `2001::1:2:0:0:3` ou `2001:0:0:1:2::3`, mais pas comme ceci `2001::1:2::3`, parce que, il serait maintenant difficile de savoir comment les quatre zéros doivent être répartis sur les zones de réductions.

Une autre ambiguïté existe, si une adresse IPv6 doit être combinée avec un port (TCP ou UDP) : dans ce cas, il ne faut pas joindre le port directement avec les deux-points à l'adresse, parce que ces deux-points seront intégrés à l'intérieur de l'adresse et donc dans certains cas, il serait difficile de savoir si la spécification du port est peut-être ou pas un composant de l'adresse. Il faut donc, dans ce cas mettre l'adresse IPv6 entre deux crochets. Cette syntaxe est demandée dans les URL (par exemple lorsque l'utilisation doit indiquer une adresse IPv6 au format numérique dans le navigateur Web).

Exemple 3 : `[2001:db8:900:551::2]:1234`

Voici l'adresse sans mettre les crochets `2001:db8:900:551::2:1234`, correspond à l'adresse intégrale `2001:db8:900:551:0:0:2:1234` vous voyez quelle ne possède aucune indication de port.

1.1.3. Configuration

Paramètres généraux

Les paramètres généraux contiennent d'abord, l'activation du support IPv6, d'autre part l'attribution optionnelle d'une adresse IPv6 sur le routeur.

OPT_IPV6 Avec cette variable, vous pouvez activer le support IPv6.

Configuration par défaut : `OPT_IPV6='no'`

HOSTNAME_IP6 (optionnelle) Cette variable règle explicitement l'adresse IPv6 du routeur. Si la variable n'est pas définie, l'adresse IPv6 est placé sur la configuration de la premier adresse du sous-réseau IPv6 (`IPV6_NET_x`, voir ci-dessous).

Exemple : `HOSTNAME_IP6='IPV6_NET_1_IPADDR'`

Configuration du sous-réseau

Dans ce paragraphe, nous allons décrire la configuration d'un ou plusieurs sous-réseaux IPv6. Un sous-réseau IPv6 est une adresse IPv6 étendue qui est spécifiée par un préfixe et qui est liée à une interface réseau spécifique. Les autres paramètres concernent l'édition du préfixe et le service DNS dans le sous-réseau, ainsi que le nom du routeur optionnel à l'intérieur du sous-réseau.

IPV6_NET_N Dans cette variable, vous indiquez le nombre de sous-réseaux IPv6 à utiliser. Vous devez définir Au moins un sous-réseau IPv6, pour utiliser l'IPv6 dans le réseau local.

Configuration par défaut : `IPV6_NET_N='0'`

IPV6_NET_x Dans cette variable, vous indiquez l'adresse IPv6, contenu dans le sous-réseau IPv6 du routeur, ainsi que la taille du masque de sous-réseau en utilisant la notation CIDR. Si le sous-réseau est un routage public, il provient en générale d'Internet ou d'un prestataire de tunnel.

Important: *Si vous activez la configuration automatique sans-état dans le même sous-réseau (voir la section* `IPV6_NET_x_ADVERTISE` *ci-dessous), la longueur du préfixe du sous-réseau doit faire 64 bits !*

Important: *Si le sous-réseau est connecté à un tunnel (voir* `IPV6_NET_x_TUNNEL` *ci-dessous), vous devez indiquer seulement une partie de l'adresse du routeur, mais* pas *le préfixe du sous-réseau associé au tunnel (qui se trouve dans* `IPV6_TUNNEL_x_PREFIX`*), avec ce préfixe, l'adresse pourra être combiné ! Dans la version précédente du paquetage IPv6, la variable* `IPV6_TUNNEL_x_PREFIX` *n'existait pas, le préfixe et le sous-réseau de l'adresse du routeur étaient ensemble dans la variable* `IPV6_NET_x`*. Toutefois, cela ne s'applique pas si le préfixe du sous-réseau est assigné dynamiquement par le fournisseur à la construction du tunnel. De plus, la longueur du préfixe du sous-réseau (dans ce cas : /48) est cachée, si bien que le routage prédéfini ne peut pas être correctement réglé et que la route vers les destinations spécifiques conduit alors à des effets étranges.*

Exemples :

```
IPV6_NET_1='2001:db8:1743:42::1/64'       # sans Tunnel~: adresse complette
IPV6_NET_1_TUNNEL=''

IPV6_NET_2='0:0:0:42::1/64'               # avec Tunnel~: adresse partielle
IPV6_NET_2_TUNNEL='1'
IPV6_TUNNEL_1_PREFIX='2001:db8:1743::/48' # voir section "configuration du Tunnel"
```

IPV6_NET_x_DEV Avec cette variable, vous indiquez le nom de l'interface du sous-réseau IPv6 sur laquelle l'adresse IPv6 sera associée. Cette interface réseau n'entre *pas* en collision avec l'interface réseau qui a été attribuée dans la configuration de base (`base.txt`), les deux adresses IPv4 et IPv6 pourront être affectées sur cette interface réseau.

Exemple : `IPV6_NET_1_DEV='eth0'`

IPV6_NET_x_TUNNEL Dans cette variable, vous indiquez un sous-réseau IPv6 spécifique, à l'index du tunnel. Le préfixe du tunnel spécifié sera combiné avec l'adresse du routeur pour obtenir l'adresse IPv6 complète pour le routeur. Si la variable est vide ou non définie, aucun sous-réseau ne fera partie du tunnel, dans la variable `IPV6_NET_x` vous devez indiquer une 'adresse IPv6 complète pour le routeur, y compris le masque de réseau (voir plus haut).

Un tunnel peut être attribué à plusieurs sous-réseaux, le préfixe du sous-réseau du tunnel est généralement assez grand pour qu'il puisse être divisé en plusieurs sous-réseaux (/56 ou plus). Bien sûr, ce n'est pas possible dans l'autre sens, attribuer un sous-réseau à plusieurs préfixes du sous-réseau du tunnel, car l'adresse du sous-réseau serait ambiguë.

Exemple : `IPV6_NET_1_TUNNEL='1'`

IPV6_NET_x_ADVERTISE Avec cette variable, vous déterminez si le préfixe du sous-réseau sera distribué par "l'intermédiaire du routeur" dans le LAN. Cela est utilisé pour une "stateless autoconfiguration" (ou configuration automatique sans état) et ne doit pas être confondu avec le DHCPv6. Les valeurs possibles sont "yes" ou "no".

Il est recommandé d'activer ce paramètre, à moins que toutes les adresses dans le réseau soient affectées statiquement ou qu'un autre routeur est déjà compétent pour notifier le préfixe du sous-réseau.

Important: *La distribution automatique des sous-réseaux fonctionne seulement si le sous-réseau est un réseau /64, c.-à-d., si la longueur du préfixe du sous-réseau est de 64 bits ! La raison est que les hôtes du réseau calculent l'adresse IPv6 à partir du préfixe et de leur adresse MAC, si l'hôte ne partage pas les 64 bits cela ne fonctionne pas. Si la configuration automatique échoue, il faut vérifier le préfixe du sous-réseau, il a peut-être été spécifié de manière incorrecte (par exemple /48).*

Configuration par défaut : `IPV6_NET_1_ADVERTISE='yes'`

IPV6_NET_x_ADVERTISE_DNS Avec cette variable vous déterminez si le service DNS local sur le sous-réseau IPv6 sera distribué par "l'intermédiaire du routeur". Cela ne fonctionne que si la fonction IPv6 du service DNS est activé par le biais de la variable `DNS_SUPPORT_IPV6`='yes'. Les valeurs possibles sont "yes" ou "no".

Configuration par défaut : `IPV6_NET_1_ADVERTISE_DNS='no'`

IPV6_NET_x_DHCP Avec cette variable, vous activez le service DHCPv6 pour le sous-réseau IPv6. Les valeurs possibles sont "yes" ou "no". Le DHCPv6 est utilisé ici uniquement pour permettre aux hôtes du sous-réseau d'obtenir des informations sur le nom de domaine et l'adresse du serveur DNS à utiliser. Actuellement l'attribution d'adresse IPv6 via le DHCPv6 n'est pas possible avec fli4l.

L'adresse du serveur DNS ne sera pas publié par le DHCPv6, si le support IPv6 du service DNS via la variable `DNS_SUPPORT_IPV6` dans le paquetage dns_dhcp n'est pas activé.

Important: *La variable `IPV6_NET_x_ADVERTISE_DNS` et `IPV6_NET_x_DHCP` ne sont pas mutuellement exclusif, mais les deux peuvent être activés. Dans ce cas, l'adresse du serveur DNS peut être attribuée de deux manières différentes sur l'hôte du réseau local.*

Un sous-réseau IPv6 au maximum peut être attaché à une interface réseau, pour configurer le DHCPv6 !

Configuration par défaut : `IPV6_NET_1_DHCP='no'`

IPV6_NET_x_NAME (optionnelle) Dans cette variable, vous pouvez paramètrer un nom d'hôte spécifique pour chaque interface du sous-réseau IPv6 du routeur.

Exemple : `IPV6_NET_1_NAME='fli4l-subnet1'`

Configuration d'un Tunnel

Dans ce paragraphe nous allons présenter la configuration d'un tunnel IPv6-6in4 Un tel tunnel est utile lorsque votre propre fournisseur d'accès Internet ne supporte pas l'IPv6 par défaut. Ainsi, nous pouvons faire un tunnel-broker avec un hôte bien précis sur Internet, avec le soi-disant PoP (Point of Presence), il faut construit une connexion bidirectionnelle via IPv4, les paquets IPv6 seront ensuite empaquetés et acheminés (d'où 6 "in" 4 parce que les paquets IPv6 sont encapsulés dans les paquets IPv4). [3] Pour que le tunnel fonctionne, il faut configurer les routeurs avec le paquetage IPv6 des deux côtés de la connexion Internet. Le premier paragraphe décrit la configuration, le deuxième paragraphe décrit la connexion.

3. Il s'agit de l'IPv4 protocole 41, "encapsulation IPv6".

IPV6_TUNNEL_N Avec cette variable vous indiquez le nombre de tunnels 6in4 à mettre en place.

Exemple : `IPV6_TUNNEL_N='1'`

IPV6_TUNNEL_x_TYPE Avec cette variable, vous déterminez le type de tunnel. Actuellement, les valeurs possibles sont : "raw" pour un tunnel qui envoi des paquets "brut", "static" pour un tunnel statique, "sixxs" pour un tunnel Heartbeat dynamique du fournisseur SixXS et "he" pour un tunnel du fournisseur Hurricane Electric. Au sujet du tunnel Heartbeat voir le paragraphe plus bas.

Exemple : `IPV6_TUNNEL_1_TYPE='sixxs'`

IPV6_TUNNEL_x_DEFAULT Avec cette variable, vous déterminez si les paquets IPv6 qui ne sont pas adressés au niveau local ou aux réseaux locaux, doivent être routés sur un autre tunnel. Il ne peut y avoir qu'un seul tunnel (parce que seulement une route par défaut peut exister). Les valeurs possibles sont "yes" ou "no".

Important: *le tunnel doit exactement être une passerelle par défaut pour les données IPv6 sortantes, car la communication avec des hôtes IPv6 ne serait pas possible autrement sur Internet ! L'utilisation exclusive du tunnel pas par défaut, n'est utile que si le trafic IPv6 sortant est envoyé via une route par défaut configurée séparément et qui n'est pas en rapport avec un tunnel. Voir l'introduction du paragraphe "configuration de route" et aussi la description de la variable `IPV6_ROUTE_x` ci-dessous.*

Configuration par défaut : `IPV6_TUNNEL_1_DEFAULT='no'`

IPV6_TUNNEL_x_PREFIX Avec cette variable, vous indiquez le préfixe IPv6 du sous-réseau du tunnel dans la notation CIDR, c.-à-d. que vous indiquez la longueur du préfixe, mais aussi l'adresse IPv6. Cette information est précisée dans la convention du fournisseur de tunnel. En ce qui concerne certains fournisseurs de tunnel, si le préfixe est réaffecté à chaque construction du tunnel, alors cette information sera inutile. (Actuellement, de tels fournisseurs ne sont pas supportés).

Important: *Cette variable* peut *restée vide, si le tunnel n'a pas de préfixe de sous-réseau attribué. Toutefois, ce tunnel ne peut pas être affecté à un sous-réseau IPv6 par la variable (`IPV6_NET_x`), parce que les adresses IPv6 dans le sous-réseau ne peuvent pas être calculées. Il est logique d'une telle configuration ne soit que provisoire, en attendant l'activation du tunnel et avant que le fournisseur de tunnel attribue un préfixe du sous-réseau (cette procédure est par ex. courant chez le fournisseur de tunnel SixXS).*

Exemples :

```
IPV6_TUNNEL_1_PREFIX='2001:db8:1743::/48'      # /48-sous-réseau
IPV6_TUNNEL_2_PREFIX='2001:db8:1743:5e00::/56' # /56-sous-réseau
```

IPV6_TUNNEL_x_LOCALV4 Dans cette variable, vous indiquez l'adresse IPv4 locale du tunnel ou le paramètre 'dynamic' si l'adresse IPv4 est allouée dynamiquement par le circuit WAN actif. S'il s'agit d'un tunnel Heartbeat (voir `IPV6_TUNNEL_x_TYPE` ci-dessus).

Exemple :

```
IPV6_TUNNEL_1_LOCALV4='172.16.0.2'
IPV6_TUNNEL_2_LOCALV4='dynamic'
```

IPV6_TUNNEL_x_REMOTEV4 Dans cette variable, vous indiquez l'adresse IPv4 distant du tunnel. Cette information est habituellement déterminée par le fournisseur du tunnel. Exemple (Correspond au PoP deham01 d'Easynet) :

```
IPV6_TUNNEL_1_REMOTEV4='212.224.0.188
```

Important: *Si la variable* `PF_INPUT_ACCEPT_DEF` *est sur "no", c.-à-d que le pare-feu IPv4 est configuré manuellement, une règle est nécessaire pour accepter tous les paquets IPv6-in-IPv4 (Protocole-IP 41) de l'extrémité du tunnel. Surnommé point d'arrêt du tunnel, la règle correspondante est indiqué ci-dessous :*

```
PF_INPUT_x='prot:41 212.224.0.188 ACCEPT'
```

IPV6_TUNNEL_x_LOCALV6 Dans cette variable, vous indiquez l'adresse IPv6 local du tunnel avec le masque de sous-réseau, en utilisant la notation CIDR. Cette information est donnée par le fournisseur d'accès du tunnel. Lors d'une nouvelle configuration du tunnel, les fournisseurs de tunnel l'attribuent à chaque extrémité du tunnel. Cette information est inutile, (actuellement les fournisseurs ne supportent pas encore cette fonction).

Exemple : `IPV6_TUNNEL_1_LOCALV6='2001:db8:1743::2/112'`

IPV6_TUNNEL_x_REMOTEV6 Dans cette variable, vous indiquez l'adresse IPv6 distante du tunnel. Cette information est donnée par le fournisseur d'accès du tunnel. Le masque de sous-réseau n'est pas nécessaire, car il est récupéré dans La variable `IPV6_TUNNEL_x_LOCALV6`. Lors d'une nouvelle configuration du tunnel, les fournisseurs de tunnel l'attribuent à chaque extrémité du tunnel. Cette information est inutile, (actuellement les fournisseurs ne supportent pas encore cette fonction).

Exemple : `IPV6_TUNNEL_1_REMOTEV6='2001:db8:1743::1'`

IPV6_TUNNEL_x_DEV (optionnelle) Dans cette variable, vous indiquez le nom de l'interface réseau du tunnel à produire. Si vous avez plusieurs tunnels, ils doivent être nommés différemment, de sorte que tout fonctionne. Si la variable n'est pas définie, un nom pour le tunnel sera généré automatiquement ("v6tun" + index Tunnel).

Exemple : `IPV6_TUNNEL_1_DEV='6in4'`

IPV6_TUNNEL_x_MTU (optionnelle) Dans cette variable, vous indiquez la taille du MTU (Maximum Transfert Unit) en octets, c.-à-d. le plus grand paquet qui peut être envoyé sur le tunnel. en régle général cette information est précisée par le fournisseur de tunnel. Le réglage par défaut si non spécifié est de "1280", il doit être compatible avec tous les tunnels.

Configuration par défaut : `IPV6_TUNNEL_1_MTU='1280'`

Certains fournisseurs de tunnel exigent un signe de vie qui soit en permanence envoyée sur le routeur du fournisseur de tunnel, pour s'assurer que l'hôte sollicite le tunnel, bien que celui-ci n'est pas utilisé. En plus le soi-disant protocole Heartbeat ("battement de coeur") est utilisé. Les fournisseurs exigent généralement une ouverture de session réussie avec identifiant et mot de passe pour empêcher les abus. Si vous utilisez un tunnel Heartbeat (comme il est proposé avec SixXS), alors les informations appropriées doivent être renregistré, elles sont décrites plus bas.

IPV6_TUNNEL_x_USERID Dans cette variable, vous indiquez le nom d'utilisateur, nécessaires pour la connexion au tunnel.

Exemple : `IPV6_TUNNEL_1_USERID='ABCDE-SIXXS'`

IPV6_TUNNEL_x_PASSWORD Dans cette variable, vous indiquez le mot de passe pour le nom d'utilisateur spécifié ci-dessus. Il ne doit pas contenir d'espaces.

Exemple : `IPV6_TUNNEL_1_PASSWORD='passwort'`

IPV6_TUNNEL_x_TUNNELID Dans cette variable, vous indiquez, l'indenfication du tunnel. Le nom du tunnel SixXs commence toujours avec un grand "T".

Exemple : `IPV6_TUNNEL_1_TUNNELID='T1234'`

IPV6_TUNNEL_x_TIMEOUT (optionnelle) Dans cette variable, vous indiquez le temps d'attente en seconde, avant la construction du tunnel. La valeur par défaut dépend du fournisseur d'accès du tunnel.

Exemple : `IPV6_TUNNEL_1_TIMEOUT='30'`

Configuration des routes

Les routes sont des chemins pour rediriger les paquets IPv6. Cela signifie que le routeur doit savoir ou envoyer les paquets entrants, il s'appuie sur une table de routage pour trouver exactement les informations. Pour les paquets IPv6, il est important de savoir où sont envoyés les paquets qui ne font pas partie du réseau local. Pour cela, une route par défaut doit être configurée pour envoyer tous les paquets à l'autre extrémité du tunnel IPv6. Vous pouvez également ajouter d'autres routes qui relient les sous-réseaux IPv6 les uns aux autres.

IPV6_ROUTE_N Dans cette variable vous indiquez le nombre de routes IPv6. En général, aucune route supplémentaire IPv6 n'est nécessaire.

Configuration par défaut : `IPV6_ROUTE_N='0'`

IPV6_ROUTE_x Dans cette variable, vous indiquez la route sous la forme 'Réseau de destination Passerelle', le réseau de destination est écrit en utilisant la notation CIDR. Vous devez indiquer `::/0` pour la route par défaut du réseau de destination. Cependant, il n'est pas nécessaire de configurer la route par défaut qui passe par le tunnel (voir l'introduction de ce paragraphe).

Exemple : `IPV6_ROUTE_1='2001:db8:1743:44::/64␣2001:db8:1743:44::1'`

IPv6-Firewall

Comme pour les réseaux IPv4, les réseaux IPv6 ont besoin d'un pare-feu, ainsi le monde extérieur ne pourra pas joindre les ordinateurs du réseau local. Cela est d'autant plus important, car chaque ordinateur est remplacé dans le cas normal, d'une adresse IPv6 unique, cette adresse qui peut être affectée à l'ordinateur de façon permanente, car elle est basée sur l'adresse MAC de la carte d'interface réseau. [4] Par conséquent, le pare-feu interdira toute demande provenant de l'extérieur, dans ce paragraphe vous allez voir comment ouvrir les entrées correspondantes petit à petit – selon vos besoins –.

La configuration du pare-feu IPv6, correspond grosso modo à la configuration du pare-feu IPv4. Les différences particulières seront examinées séparément.

PF6_LOG_LEVEL La configuration du système de journalisation dans la variable `PF6_LOG_LEVEL` est utilisée pour toutes les chaînes ci-dessous sans distinction, leur contenu peut être réglé sur l'une des valeurs suivantes : debug, info, notice, warning, err, crit, alert, emerg.

4. Une exception existe, si "Privacy extension" est activé pour les hôtes du LAN, alors une partie de l'adresse IPv6 sera générée de façon aléatoire. Ces adresses par définition, ne sont pas connues du monde extérieur et donc la configuration du firewall sera partiellement ou pas du tout pertinente.

PF6_INPUT_POLICY Cette variable définit la politique par défaut pour les paquets entrants sur le routeur avec la (chaîne INPUT). Les valeurs possibles sont "REJECT" (par défaut, rejette tous les paquets), "DROP" (rejette en secret tous les paquets), "ACCEPT" (accepte tous les paquets). Pour une description plus détaillée, voir la documentation de la variable `PF_INPUT_POLICY`

Configuration par défaut : `PF6_INPUT_POLICY='REJECT'`

PF6_INPUT_ACCEPT_DEF Dans cette variable vous pouvez activer les règles prédéfinies pour la chaîne INPUT du pare-feu IPv6. Les valeurs possibles sont "yes" ou "no".

La règle par défaut pour l'ouverture entrante du trafic pings-ICMPv6 (un ping par seconde en tant que limite), ainsi que pour les paquets NPD (Neighbour Discovery Procotol) sur le pare-feu, qui sont nécessaires pour l'auto-configuration sans état des réseaux IPv6. La communication localhost et la réponse des paquets entre la communication d'origine locale, sont également autorisés. Enfin, le pare-feu IPv4 est réglé de telle sorte que pour chaque tunnel IPv6 encapsulé dans le paquet IPv4, la communication avec l'extrémité du tunnel sera acceptée.

Configuration par défaut : `PF6_INPUT_ACCEPT_DEF='yes'`

PF6_INPUT_LOG Cette variable active le fichier journal il enregistre tous les paquets entrants et rejetés.Les valeurs possibles sont "yes" ou "no". Pour une description plus détaillée, voir la documentation de la variable `PF_INPUT_LOG`.

Configuration par défaut : `PF6_INPUT_LOG='no'`

PF6_INPUT_LOG_LIMIT On configure avec cette variable une limite pour le fichier journal de la chaîne INPUT du pare-feu IPv6, pour garder le fichier journal en lecture. Pour une description plus détaillée, voir la documentation de la variable `PF_INPUT_LOG_LIMIT`.

Configuration par défaut : `PF6_INPUT_LOG_LIMIT='3/minute:5'`

PF6_INPUT_REJ_LIMIT On configure avec cette variable une limite pour le rejet des paquets TCP entrants. Les paquets TCP dépassant cette limite, seront rejetés avec la méthode douce (DROP). Pour une description plus détaillée, voir la documentation de la variable `PF_INPUT_REJ_LIMIT`.

Configuration par défaut : `PF6_INPUT_REJ_LIMIT='1/second:5'`

PF6_INPUT_UDP_REJ_LIMIT On configure avec cette variable une limite pour le rejet des paquets UDP entrants. Les paquets UDP dépassant cette limite, seront rejetés avec la méthode douce (DROP). Pour une description plus détaillée, voir la documentation de la variable `PF_INPUT_UDP_REJ_LIMIT`.

Configuration par défaut : `PF6_INPUT_UDP_REJ_LIMIT='1/second:5'`

PF6_INPUT_ICMP_ECHO_REQ_LIMIT Avec cette variable, vous définissez la façon de répondre à une demande de requête écho ICMPv6 commune. La fréquence et la limite de restriction est décrite analogiquement comme ceci 'n/unité de tempsrafales' par exemple, '3/minute :5'. Une fois que la limite est dépassée, le paquet est tout simplement ignoré (DROP). S'il la variable est vide, la valeur par défaut utilisé sera la suivante '1/seconde :5' si la variable contient 'none', alors, aucune limite ne sera effectuée.

Configuration par défaut : `PF6_INPUT_ICMP_ECHO_REQ_LIMIT='1/second:5'`

PF6_INPUT_ICMP_ECHO_REQ_SIZE Avec cette variable, vous définissez la taille (en octets) que peut recevoir la demande de requête écho ICMPv6. Ce chiffre vient "ajouter"

des donnés à l'en-tête du paquet à prendre en considération. La valeur par défaut est de 150 octets.

Configuration par défaut : `PF6_INPUT_ICMP_ECHO_REQ_SIZE='150'`

PF6_INPUT_N Dans cette variable vous indiquez le nombre de règles du pare-feu IPv6 pour les paquets entrants (chaîne INPUT). Par défaut, deux règles sont activées : la première permet l'accès au routeur par tous des hôtes locaux via l'adresse du niveau de lien et la seconde permet la communication des hôtes du premier sous-réseau IPv6 défini avec le routeur.

Si plusieurs sous-réseaux IPv6 locaux sont définis, la seconde règle doit être reproduite au temps de fois que nécessaire. Voir le fichier de configuration.

Exemple : `PF6_INPUT_N='2'`

PF6_INPUT_x Dans cette variable, vous indiquez la règle pour la chaîne INPUT du pare-feu IPv6. Pour une description plus détaillée, voir la documentation de la variable `PF_INPUT_x`.

Les différences par rapport au pare-feu IPv4 :

- Au lieu de `IP_NET_x` vous devez mettre `IPV6_NET_x`.
- Au lieu de `IP_ROUTE_x` vous devez mettre `IPV6_ROUTE_x`.
- Les adresses IPv6 doivent être placées entre deux crochets. (y compris le masque de sous réseau, s'il est disponible).
- Tous les adresses IPv6 que vous indiquez (y compris les variables `IPV6_NET_x`, etc.) doivent être placées entre deux crochets, si l'adresse est suivi d'un port ou d'une plage de ports.

Exemple :

```
PF6_INPUT_1='[fe80::0/10] ACCEPT'
PF6_INPUT_2='IPV6_NET_1 ACCEPT'
PF6_INPUT_3='tmpl:samba DROP NOLOG'
```

PF6_INPUT_x_COMMENT Dans cette variable, vous pouvez indiquer une description ou un commentaire associé à la règle INPUT.

Exemple : `PF6_INPUT_3_COMMENT='no␣samba␣traffic␣allowed'`

PF6_FORWARD_POLICY Avec cette variable vous définissez la stratégie par défaut pour les paquets transmis par le routeur avec la (chaîne FORWARD). Les valeurs possibles sont "REJECT" (par défaut, rejette tous les paquets), "DROP" (rejette en secret tous les paquets), "ACCEPT" (accepte tous les paquets). Pour une description plus détaillée, voir la documentation de la variable `PF_FORWARD_POLICY`.

Configuration par défaut : `PF6_FORWARD_POLICY='REJECT'`

PF6_FORWARD_ACCEPT_DEF Cette variable active les règles prédéfinies pour la chaîne FORWARD du pare-feu IPv6. Les valeurs possibles sont "yes" ou "no".

Ouverture des règles par défaut sur le pare-feu pour les ping ICMPv6 sortants (un ping par seconde comme limite). Les paquets de réponses au ping seront également autorisés.

Configuration par défaut : `PF6_FORWARD_ACCEPT_DEF='yes'`

PF6_FORWARD_LOG Cette variable active le fichier journal il enregistre tous les paquets entrants et rejetés. Les valeurs possibles sont "yes" ou "no". Pour une description plus détaillée, voir la documentation de la variable `PF_FORWARD_LOG`.

Configuration par défaut : `PF6_FORWARD_LOG='no'`

PF6_FORWARD_LOG_LIMIT On configure avec cette variable une limite pour le fichier journal de la chaîne FORWARD du pare-feu IPv6, pour garder le fichier journal en lecture. Pour une description plus détaillée, voir la documentation de la variable `PF_FORWARD_LOG_LIMIT`.

Configuration par défaut : `PF6_FORWARD_LOG_LIMIT='3/minute:5'`

PF6_FORWARD_REJ_LIMIT On configure avec cette variable une limite pour le rejet des paquets TCP entrants. Les paquets TCP dépassant cette limite, seront rejetés en secret avec (DROP). Pour une description plus détaillée, voir la documentation de la variable `PF_FORWARD_REJ_LIMIT`.

Configuration par défaut : `PF6_FORWARD_REJ_LIMIT='1/second:5'`

PF6_FORWARD_UDP_REJ_LIMIT On configure avec cette variable une limite pour le rejet des paquets UDP entrants. Les paquets UDP dépassant cette limite, seront rejetés avec la méthode douce (DROP). Pour une description plus détaillée, voir la documentation de la variable `PF_FORWARD_UDP_REJ_LIMIT`.

Configuration par défaut : `PF6_FORWARD_UDP_REJ_LIMIT='1/second:5'`

PF6_FORWARD_N Dans cette variable vous indiquez le nombre de règles du pare-feu IPv6 pour les paquets entrants (chaîne FORWARD). Par défaut, deux règles sont activées : la première empêche la transmission de tous les paquets samba dans d'autre réseaux qui ne proviennent pas du réseau local et la seconde permet la communication à partir des hôtes du premier sous-réseau IPv6 défini dans le routeur.

Si plusieurs sous-réseaux IPv6 locaux sont définis, la seconde règle doit être reproduite au temps de fois que nécessaire. Voir le fichier de configuration.

Exemple : `PF6_FORWARD_N='2'`

PF6_FORWARD_x Dans cette variable, vous indiquez la règle pour la chaîne FORWARD du pare-feu IPv6. Pour une description plus détaillée, voir la documentation de la variable `PF_FORWARD_x`.

Les différences par rapport au pare-feu IPv4 :

- Au lieu de `IP_NET_x` vous devez mettre `IPV6_NET_x`.
- Au lieu de `IP_ROUTE_x` vous devez mettre `IPV6_ROUTE_x`.
- Les adresses IPv6 doivent être placées entre deux crochets. (y compris le masque de sous réseau, s'il est disponible).
- Tous les adresses IPv6 que vous indiquez (y compris les variables `IPV6_NET_x`, etc.) doivent être placées entre deux crochets si l'adresse est suivi d'un port ou d'une plage de ports.

Exemple :

```
PF6_FORWARD_1='tmpl:samba DROP'
PF6_FORWARD_2='IPV6_NET_1 ACCEPT'
```

PF6_FORWARD_x_COMMENT Dans cette variable, vous pouvez indiquer une description ou un commentaire associé à la règle FORWARD.

Exemple : `PF6_FORWARD_1_COMMENT='no␣samba␣traffic␣allowed'`

PF6_OUTPUT_POLICY Cette variable définit la stratégie par défaut pour les paquets sortants du routeur(chaîne OUTPUT). Les valeurs possibles sont "REJECT" (par défaut, pour tous les paquets), "DROP" (rejette secrètement tous les paquets) et "ACCEPT" (accepte tous les paquets). Pour plus de détails, reportez-vous à la documentation de la variable `PF_OUTPUT_POLICY`.

Configuration par défaut : `PF6_OUTPUT_POLICY='REJECT'`

PF6_OUTPUT_ACCEPT_DEF Cette variable active les règles pré-réglées pour la chaîne OUTPUT du pare-feu IPv6. Les valeurs possibles sont "yes" ou "no". À l'heure actuelle, il n'existe pas de règle prédéfinie.

Configuration par défaut : `PF6_OUTPUT_ACCEPT_DEF='yes'`

PF6_OUTPUT_LOG Cette variable permet l'enregistrement tous les paquets sortants rejetés. Les valeurs possibles sont "yes" ou "no". Pour plus de détails, reportez-vous à la documentation de la variable `PF_OUTPUT_LOG`.

Configuration par défaut : `PF6_OUTPUT_LOG='no'`

PF6_OUTPUT_LOG_LIMIT On configure avec cette variable une limite pour le journal de la chaîne OUTPUT du pare-feu IPv6, pour garder le fichier journal en lecture. Pour une description plus détaillée de la documentation voir la variable `PF_OUTPUT_LOG_LIMIT`.

Configuration par défaut : `PF6_OUTPUT_LOG_LIMIT='3/minute:5'`

PF6_OUTPUT_REJ_LIMIT On configure avec cette variable une limite pour le rejet des paquets TCP sortants. Les paquets TCP dépassant cette limite, seront rejetés en secret avec (DROP). Pour une description plus détaillée, voir la documentation de la variable `PF_OUTPUT_REJ_LIMIT`.

Configuration par défaut : `PF6_OUTPUT_REJ_LIMIT='1/second:5'`

PF6_OUTPUT_UDP_REJ_LIMIT On configure avec cette variable une limite pour le rejet des paquets UDP sortants. Les paquets UDP dépassant cette limite, seront rejetés en secret avec (DROP). Pour une description plus détaillée, voir la documentation de la variable `PF_OUTPUT_UDP_REJ_LIMIT`.

Configuration par défaut : `PF6_OUTPUT_UDP_REJ_LIMIT='1/second:5'`

PF6_OUTPUT_N Dans cette variable vous indiquez le nombre de règles du pare-feu IPv6 pour les paquets entrants (chaîne OUTPUT). Par défaut, deux règles sont activées : la première permet l'accès au routeur par tous des hôtes locaux via l'adresse du niveau de lien et la seconde permet la communication des hôtes du premier sous-réseau IPv6 défini avec le routeur.

Si plusieurs sous-réseaux IPv6 locaux sont définis, la seconde règle doit être reproduite au temps de fois que nécessaire. Voir le fichier de configuration.

Exemple : `PF6_OUTPUT_N='1'`

PF6_OUTPUT_x Dans cette variable, vous indiquez la règle pour la chaîne OUTPUT du pare-feu IPv6. Pour une description plus détaillée, voir la documentation de la variable `PF_OUTPUT_x`.

Les différences par rapport au pare-feu IPv4 :

- Au lieu de `IP_NET_x` vous devez mettre `IPV6_NET_x`.
- Au lieu de `IP_ROUTE_x` vous devez mettre `IPV6_ROUTE_x`.
- Les adresses IPv6 doivent être placées entre deux crochets. (y compris le masque de sous réseau, s'il est disponible).
- Tous les adresses IPv6 que vous indiquez (y compris les variables `IPV6_NET_x`, etc.) doivent être placées entre deux crochets si l'adresse est suivi d'un port ou d'une plage de ports.

Exemple :

```
PF6_OUTPUT_1='tmpl:ftp IPV6_NET_1 ACCEPT HELPER:ftp'
```

PF6_OUTPUT_x_COMMENT Dans cette variable, vous pouvez indiquer une description ou un commentaire associé à la règle OUTPUT.

Exemple : `PF6_OUTPUT_3_COMMENT='no␣samba␣traffic␣allowed'`

PF6_USR_CHAIN_N Dans cette variable, vous indiquez le nombre de chaînes, qui seront définies par l'utilisateur dans la table du pare-feu IPv6. Pour une description plus détaillée, voir la documentation de la variable `PF_USR_CHAIN_N`.

Configuration par défaut : `PF6_USR_CHAIN_N='0'`

PF6_USR_CHAIN_x_NAME Dans cette variable, vous indiquez le nom personnalisé de la table du pare-feu IPv6. Pour une description plus détaillée, voir la documentation de la variable `PF_USR_CHAIN_x_NAME`

Exemple : `PF6_USR_CHAIN_1_NAME='usr-myvpn'`

PF6_USR_CHAIN_x_RULE_N Dans cette variable, vous indiquez le nombre de règles personnalisées pour pare-feu IPv6 associé à la table du pare-feu IPv6. Pour une description plus détaillée, voir la documentation de la variable `PF_USR_CHAIN_x_RULE_N`.

Exemple : `PF6_USR_CHAIN_1_RULE_N='0'`

PF6_USR_CHAIN_x_RULE_x dans cette variable, vous indiquez la règle définie par l'utilisateur de la table du pare-feu IPv6. Pour une description plus détaillée, voir la documentation de la variable `PF_USR_CHAIN_x_RULE_x`

Les différences par rapport au pare-feu IPv4 :

- Au lieu de `IP_NET_x` vous devez mettre `IPV6_NET_x`.
- Au lieu de `IP_ROUTE_x` vous devez mettre `IPV6_ROUTE_x`.
- Les adresses IPv6 doivent être placées entre deux crochets. (y compris le masque de sous réseau, s'il est disponible).
- Tous les adresses IPv6 que vous indiquez (y compris les variables `IPV6_NET_x`, etc.) doivent être placées entre deux crochets si l'adresse est suivi d'un port ou d'une plage de ports.

PF6_USR_CHAIN_x_RULE_x_COMMENT Dans cette variable, vous pouvez indiquer une description ou un commentaire associé à la règle.

Exemple : `PF6_USR_CHAIN_1_RULE_1_COMMENT='some␣user-defined␣rule'`

PF6_POSTROUTING_N Dans cette variable vous indiquez le nombre de règles du pare-feu IPv6 pour le masquage des paquets (chaîne POSTROUTING). Pour plus de détails, reportez-vous à la documentation de la variable `PF_POSTROUTING_N`.

Exemple : `PF6_POSTROUTING_N='2'`

PF6_POSTROUTING_x PF6_POSTROUTING_x_COMMENT

Vous indiquez dans ces variables la liste de règles qui décrivent les paquets IPv6 qui seront masqués par le routeur (ou transmis non masqué). Pour plus de détails, reportez-vous à la documentation de la variable `PF_POSTROUTING_x`

PF6_PREROUTING_N Dans cette variable, vous indiquez le nombre de règles du pare-feu IPv6 pour transmettre les paquets vers une autre destination (chaîne PREROUTING). Pour plus de détails, reportez-vous à la documentation de la variable `PF_PREROUTING_N`.

Exemple : `PF6_PREROUTING_N='2'`

PF6_PREROUTING_x PF6_PREROUTING_x_COMMENT

Vous indiquez dans ces variables la liste de règles qui décrivent la transmission des paquets IPv6 du routeur vers une autre destination. Pour plus de détails, reportez-vous à la documentation de la variable `PF_PREROUTING_x`.

1.1.4. WebGUI

Ce paquetage installe un menu supplémentaire dans le mini-HTTPD pour le "filtrage de paquets (IPv6)", sous lequel vous pourrez voir les enregistrements du filtrage de paquets de votre configuration IPv6.

A. Annexe du paquetage IPV6

A.1. IPV6 - Connexion Internet en utilisant un tunnel IPv6 SixXS

Cette section décrit comment le paquetage IPv6 peut être utilisé pour vous aidez à créer un tunnel avec le fournisseur SixXS (https ://www.sixxs.net/) pour connecter votre propre réseau à Internet IPv6.

A.1.1. Créer un compte

Tout d'abord, un compte SixXS doit être enregistré dans "Signup for new users". Une fois que vous avez surmonter cet obstacle, vous avez un nom d'utilisateur de la forme SIxXS-YYYYY et un mot de passe associé. Ces données sont nécessaires plus tard pour la configuration du tunnel.

A.1.2. Configurer le tunnel

Préparations

Mais d'abord, vous devez vous connecter au tunnel. Cela sera fait après l'enregistrement via le menu "Request tunnel". Ici c'est important, car vous devez créer le type de tunnel dans cette deuxième partie de l'enregistrement, vous devez choisir "Dynamic IPv4 Endpoint using Heartbeat protocol", parce que cette configuration est directement pris en charge par fli4l. Troisième partie, le paramètre "Static IPv4 Endpoint" est également possible, si vous avez attribué une adresse IPv4 fixe, qui ne change jamais. Pour le moment le paramètre du tunnel "Dynamic NAT-traversing IPv4 Endpoint using AYIYA" n'est pas supportée par le paquetage IPv6.

Une fois que vous avez indiqué les autres champs, sur l'emplacement du routeur, avec le paramètre "Next step" de la deuxième page, vous devez choisir un ou plusieurs PoP (Points of Presence), c'est important pour la construction du tunnel. Il faut prendre la personne qui est le plus proche, pour tester le tunnel avec les paquets IPv6 aussi efficacement que possible.

Si toutes les informations ont été enregistrées, vous pouvez activer "Place request for new tunnel", vous recevrez un E-Mail avec les données importantes nécessaires au tunnel. Il comportera :

1. le numéro d'identification du tunnel (T...)
2. Le nom des PoPs associés
3. L'adresse IPv4 du PoPs attribuée par ("SixXS IPv4")
4. l'adresse IPv6 locale du tunnel, y compris le masque de sous-réseau (typique à SixXS /64), c'est à dire l'adresse du routeur ("Votre" IPv6)
5. l'adresse IPv6 distant du tunnel, y compris le masque de sous-réseau (qui est identique au sous-réseau de l'adresse IPv6 locale), c.-à-d. l'adresse du PoP ("SixXS IPv6")

Configuration

Maintenant, le tunnel peut être configuré ! En premier, vous devez placer la variable `IPV6_TUNNEL_N` sur "1" parce que vous devez construire exactement un tunnel :

```
IPV6_TUNNEL_N='1'
```

Les détails SixXS seront enregistrées dans la configuration IPv6 suivante :

1. Le numéro d'identification du tunnel est entré dansla variable `IPV6_TUNNEL_1_TUNNELID`.
2. ne pas enregistrer (le nom du PoP est sans intérêt).
3. L'adresse IPv4 du POP est à enregistrer dans la variable `IPV6_TUNNEL_1_REMOTEV4`.
4. L'adresse IPv6 locale du tunnel *avec* le masque sous-réseau doit être indiqué dans la variable `IPV6_TUNNEL_1_LOCALV6`.
5. L'adresse IPv6 distant du tunnel *sans* le masque sous-réseau doit être indiqué dans la variable `IPV6_TUNNEL_1_REMOTEV6`.

En outre, vous devez indiquer le nom d'utilisateur et le mot de passe dans la configuration du tunnel dans les variables `IPV6_TUNNEL_1_USERID` et `IPV6_TUNNEL_1_PASSWORD`. Enfin, il faut noter dans la variable `IPV6_TUNNEL_1_TYPE` que la configuration du tunnel est un tunnel SixXS :

```
IPV6_TUNNEL_1_TYPE='sixxs'
```

Si vous avez créé un PoP SixXS "deham01", avec l'adresse IPv4 `212.224.0.188`, les points d'arrêt de tunnel (distant) `2001:db8:900:551::1/64`, le paramètre (local) `2001:db8:900:551::2/64`, l'ID du tunnel "T1234", le nom d'utilisateur "USER1-SIXXS" et le mot de passe "sixxs" (n'utiliser *pas* ce mot de passe s'il vous plaît) alors configuration ressemblera à ceci :

```
IPV6_TUNNEL_N='1'
IPV6_TUNNEL_1_LOCALV4='dynamic' # ou l'adresse fixe IPv4 locale
IPV6_TUNNEL_1_REMOTEV4='212.224.0.188'
IPV6_TUNNEL_1_LOCALV6='2001:db8:900:551::2/64'
IPV6_TUNNEL_1_REMOTEV6='2001:db8:900:551::1'
IPV6_TUNNEL_1_TYPE='sixxs'
IPV6_TUNNEL_1_USERID='USER1-SIXXS'
IPV6_TUNNEL_1_PASSWORD='sixxs'
IPV6_TUNNEL_1_TUNNELID='T1234'
```

Test

Si vous avez réussi, vous pouvez commencer la mettre à jour et la configuration du routeur fli4l. Ensuite vous pouvez vous connecter au routeur (directement ou par ex. via SSH), vous devriez peut être pouvoir pingger l'extrémité du tunnel distant. Si cela a fonctionné, vous devriez voir les données circuler comme l'exemple ci-dessous :

```
garm 3.6.0-revXXXXX # ping -c 4 2001:db8:900:551:0:0:0:1
PING 2001:db8:900:551::1 (2001:db8:900:551::1): 56 data bytes
64 bytes from 2001:db8:900:551::1: seq=0 ttl=64 time=67.646 ms
64 bytes from 2001:db8:900:551::1: seq=1 ttl=64 time=72.001 ms
64 bytes from 2001:db8:900:551::1: seq=2 ttl=64 time=70.082 ms
64 bytes from 2001:db8:900:551::1: seq=3 ttl=64 time=67.996 ms
```

```
--- 2001:db8:900:551::1 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 67.646/69.431/72.001 ms
```

La dernière ligne est importante "0% packet loss", cela signifie que les paquets pingger ont tous reçu une réponse. Si vous n'avez aucune réponse de l'autre bout du tunnel, le résultat est différent :

```
garm 3.6.0-revXXXXX # ping -c 4 2001:db8:900:551:0:0:0:1
PING 2001:db8:900:551::1 (2001:db8:900:551::1): 56 data bytes

--- 2001:db8:900:551::1 ping statistics ---
4 packets transmitted, 0 packets received, 100% packet loss
```

Ici aucun paquet pingger n'a reçu de réponse ("100% packet loss"). Cela signifie que soit la configuration n'est pas correcte, ou que la connexion au tunnel SixXS n'a pas encore été totalement établie. Dans le second cas, vous devriez attendre un certain temps parce que la configuration sur les POP peut ainsi prendre quelques heures. Si vous avez vérifié la configuration deux ou trois fois et que vous n'avez trouvé aucune erreur et un certain temps, c'est écoulé, vous devez contacter par e-mail SixXS et décrire le problème avec le plus de détail possible.

A.1.3. Configurer le sous-réseaux

Préparations

Si le tunnel fonctionne, vous avez réussi de paramétrer la première partie. Mais ce n'est pas encore terminé. Car le routeur n'a pas encore la possibilité de renvoyer les paquets IPv6 venant d'Internet, vers l'hôte du réseau local. Il faut configurer seulement un sous-réseau IPv6 pour intégrer les hôtes du réseau local.

Voici une petite différence, mais significatif sur la façon de configurer un réseau IPv4 : À cause du manque d'adresse un seul hôte est directement connecté à Internet. Les autres hôtes dans le réseau local, transmettent seulement dans le réseau interne, c.-à-d. que les adresses de domaine `192.168.*.*`, `172.16.*.*`, bis `172.31.*.*`, et `10.*.*.*`, sont non routable (ils ne peuvent pas être connectées à Internet). [1] Avec les adresses IPv6 il y en a en abondance, il n'y a donc pas besoin d'utiliser des adresses réseau internes. Toutefois, en raison de la nature mondiale des sous-réseaux locaux on doit s'assurer que les adresses des hôtes locaux n'entrent pas en conflit avec les autres adresses sur Internet. Par conséquent, un sous-réseau doit être attribué par le fournisseur IPv6 afin d'éviter de telles collisions.

Sur SixXS, allez dans le menu "Request subnet". Ici vous devez principalement indiquer le tunnel à utiliser, qui est facile parce que jusqu'à présent un seul tunnel a été configuré. Ensuite, envoyer le formulaire avec "Place request for new subnet", vous recevrez un e-mail après un certain temps avec les informations suivantes :

1. L'adresse du sous-réseau IPv6, y compris le masque de sous-réseau ("Subnet IPv6")
2. L'adresse IPv6 du routeur dans le tunnel, où le sous-réseau transmettra vers SixXS ("routé vers")

1. voir RFC 1918 (http://tools.ietf.org/html/rfc1918) pour plus de détails

3. L'adresse IPv4 du routeur ("Votre IPv4") [2]

Maintenant ces données sont suffisantes pour configurer fli4l avec votre propre sous-réseau IPv6. Cependant, il faut savoir une chose : le sous-réseau attribué est généralement très grand. SixXS attribue habituellement /48e de sous-réseaux, c.-à-d. dans une adresse IPv6 128 bits une partie est utilisé pour le préfixe du réseau avec /48 bits et le reste pour l'adressage des hôtes, donc 128 - 48 = 80 Bits. Un tel grand sous-réseau a deux inconvénients majeurs. Le premier inconvénient est sa taille : on peut configurer $2^{80} \approx 1209$ trillard d'adresses pour les hôtes. Il ne paraît pas conseillé de gérer cela sans utiliser une structure supplémentaire pour les adresses des hôtes. Le second inconvénient est plus grave : utiliser *l'autoconfiguration IPv6* dans un si grand sous-réseau, n'est pas possible. L'autoconfiguration est un processus, auquel un hôte IPv6 reçoit un protocole déterminé, avec un préfixe et l'adresse Mac de la carte réseau, ce bricolage crée l'adresse IPv6 de sous-réseau. L'adresse Mac se compose de six octets et avec l'aide du standard EUI-64, on peut étendre l'adresse à huit octets. Cela fait 64 bits, et puis c'est fini, il y a tout simplement pas assez d'informations pour arriver au 80 bits pour l'adressage de l'hôte.

Voici un long discours, en quelques mots : l'adressage du sous-réseau doit être plus petite, afin de pouvoir faire une configuration automatique. Il faudrait arriver à /64 bits pour adresser le sous réseau. C'est assez simple : Le masque de sous-réseau est facilement modifiable à /64. Par ex. le sous-réseau distribué par SixXS est `2001:db8:123::/48`, ensuite le sous-réseau pour lequel vous souhaitez configurer fli4l est `2001:db8:123::/64`. Cela signifie dans le détail qu'avec un masque de sous-réseau de /48, nous avons $2^{(64-48)} = 2^{16} = 65536$ sous-sous-réseaux à répartir, la premier adresse de sous-réseaux n'a que des zéros qui devrait être utilisé par fli4l. On doit se souvenir, que l'abréviation de l'adresse`2001:db8:123::` est en faite `2001:db8:123:0:0:0:0:0`. Les trois premiers chiffres est la partie attribuée clairement par le fournisseur IPv6 pour alloués le sous-réseau, les quatre chiffres suivant est le sous-sous-réseau qui est représente par des "zéros", [3] ces quatre derniers chiffres sont réservés pour la partie hôte. Le résultats du (sous-) sous-réseau est encore énorme, il peut accueillir jusqu'à $2^{64} \approx 18,4$ billions d'hôtes. Merci à l'autoconfiguration IPv6, pour obtenir les adresses réelles qui ne seront jamais en contact. C'est une bonne chose ...

Configuration

Retour à la configuration ! Premièrement, la variable `IPV6_NET_N` est réglée sur "1", parce qu'un sous-réseau local IPv6 doit être mis en place. Vous indiquez l'adresse IPv6 de sous-réseau avec le masque de sous réseau /64 dans la variable `IPV6_NET_1`. Mais ce n'est pas tout à fait juste : Car l'adresse IPv6 *de sous-réseau à l'intérieur du routeur*, n'a *pas* de préfixe de sous-réseau qui est affecté au tunnel. Celui-ci est configuré ailleurs, notamment dans la partie configuration du tunnel. Maintenant, vous devez placer le préfixe de sous-réseau dans la variable `IPV6_TUNNEL_1_PREFIX`

Si vous avez reçu de SIXXS /48 de sous-réseau IPv6 `2001:db8:123::/48`. Nous allons ajouter le nombre '456' dans la "1", adresse de sous-réseau à l'intérieur du routeur pour avoir un sous-sous-réseau de /64. nous allons ensuite créer la configuration suivante :

```
IPV6_NET_N='1'
IPV6_NET_1='0:0:0:456::1/64'              # Adresse-IPv6 du Routeur (aucun préfixe
```

2. Si l'adresse IPv4 du routeur est dynamique, remplacer celle-ci par "heartbeat"
3. Bien sûr, vous pouvez opter pour un sous-sous-réseau différent !

```
                                        # de sous-réseau + Masque de sous réseau)
IPV6_TUNNEL_1_PREFIX='2001:db8:123::/48' # /48-préfixe de sous-réseau
```

Il convient de noter, que les trois premiers zéros dans la variable `IPV6_NET_1` sont pour ainsi dire attribués au tunnel avec un préfixe de sous-réseau de /48. Conjointement avec le préfixe de sous-réseau de /48 qui est attribué par le fournisseur de tunnel, il sera notifié en sous-réseau de /64 par l'adresse `2001:db8:123:456::/64` et l'adresse IPv6 du routeur `2001:db8:123:456::1`.

Le nom de l'interface réseau sur laquelle ce sous-réseau est attaché n'est pas encore enregistré. Il faut affecter pour chaque sous-réseau une interface réseau. S'il n'y a pas plusieurs cartes réseau configurées dans le routeur, le nom de l'interface réseau de la manière typique sera "eth0" ou "wlan0" pour un périphérique wifi. Regardez en cas de doute le paramètre dans la variable `IP_NET_1_DEV` ("IP" sans "6") et recopiez simplement le contenu :

```
IPV6_NET_1_DEV='eth0' # Interface réseau pour le sous-réseau IPv6
```

Enfin, nous devons paramétrer toutes les variables pour l'autoconfiguration-IPv6 :

```
IPV6_NET_1_ADVERTISE='yes'     # /64-préfixe de sous-réseau et Route par défaut par RA
IPV6_NET_1_ADVERTISE_DNS='yes' # Serveur-DNS par RA (exige le
                               # DNS_SUPPORT_IPV6='yes'!)
IPV6_NET_1_DHCP='yes'          # Nom-Domaine et Serveur-DNS par DHCPv6
                               # (celui-ci exige DNS_SUPPORT_IPV6='yes')
```

Les deux dernières variables ne sont pas nécessaires pour le fonctionnement du sous-réseau IPv6 , mais très utiles. Elles sont utilisées pour diffuser des informations supplémentaires sur le sous-réseau IPv6, à savoir l'adresse IPv6 du serveur DNS et le nom de domaine utilisé. Le serveur DNS peut être même publié de deux façons différentes. Parce que les systèmes différents, mettent ici les préférences diverses à jour, il est avantageux à la fois d'activer le (RDNSS via les annonces de routeur et le DHCPv6).

Test

Voici un exemple de l'ensemble de la configuration IPv6 (on suppose que la variable est sur `DNS_SUPPORT_IPV6='yes'`!) :

```
IPV6_NET_N='1'
IPV6_NET_1='0:0:0:456::1/64'   # Adresse-IPv6 du Routeur (aucun préfixe de
                               # sous-réseau + Masque de sous réseau)
IPV6_NET_1_DEV='eth0'          # Interface réseau pour le sous-réseau IPv6
IPV6_NET_1_ADVERTISE='yes'     # Préfixe du sous-réseau et la route par défaut par RA
IPV6_NET_1_ADVERTISE_DNS='yes' # Serveur-DNS par RA
IPV6_NET_1_DHCP='yes'          # Nom de domaine et Serveur-DNS par DHCPv6

IPV6_TUNNEL_N='1'
IPV6_TUNNEL_1_PREFIX='2001:db8:123::/48' # /48-Masque de sous-réseau
IPV6_TUNNEL_1_LOCALV4='dynamic'          # ou une adresse local IPv4 fixe
IPV6_TUNNEL_1_REMOTEV4='212.224.0.188'
IPV6_TUNNEL_1_LOCALV6='2001:db8:900:551::2/64'
IPV6_TUNNEL_1_REMOTEV6='2001:db8:900:551::1'
IPV6_TUNNEL_1_TYPE='sixxs'
IPV6_TUNNEL_1_USERID='USER1-SIXXS'
IPV6_TUNNEL_1_PASSWORD='sixxs'
IPV6_TUNNEL_1_TUNNELID='T1234'
```

Ce routeur fli4l est normalement configuré pour un hôte sous Windows 7, avec une configuration automatiquement des adresses IPv6 et une route par défaut, un serveur DNS et un domaine, rendant ainsi l'ordinateur apte pour l'IPv6. Maintenant vous pouvez par exemple tester avec un simple ping de l'ordinateur Windows vers Internet IPv6. Dans l'exemple suivant, nous essayons d'atteindre le serveur Web fli4l.de à partir d'un hôte Windows (nous utilisons directement l'adresse IPv6, afin de ne pas procéder à la fonctionnalité de correction DNS) :

```
C:\>ping 2001:bf0:c000:a::2:132

Ping exécuté pour 2001:bf0:c000:a::2:132 avec 32 octets de données~:

Réponse de 2001:bf0:c000:a::2:132: Temps=104ms
Réponse de 2001:bf0:c000:a::2:132: Temps=102ms
Réponse de 2001:bf0:c000:a::2:132: Temps=106ms
Réponse de 2001:bf0:c000:a::2:132: Temps=106ms

Statistique du Ping pour 2001:bf0:c000:a::2:132:
    Paquets~: Envoyés = 4, Reçus = 4, Perdus = 0 (Perte 0%),
Durée approximative de boucle en milliseconde~:
    Minimum = 102ms, Maximum = 106ms, Meyenne = 104ms
```

Enfin, nous allons utiliser l'outil "traceroute" (dans Windows : "tracert") pour déterminer si le paquet est routé correctement. Un exemple du réseau local de l'auteur est indiqué ci-dessous. Cela montre bien qu'un seul paquet vient du routeur fli4l (première ligne), puis descend de l'autre côté du tunnel (deuxième ligne) et enfin dans le monde de l'Internet IPv6 (à partir de la troisième ligne) :

```
C:\>tracert 2001:bf0:c000:a::2:132


Identification de la route vers virtualhost.in-berlin.de [2001:bf0:c000:a::2:132]
avec un maximum de 30 sauts~:

  1    <1 ms    <1 ms    <1 ms  garm.example.org [2001:db8:13da:1::1]
  2    70 ms    79 ms    71 ms  gw-1362.ham-01.de.sixxs.net [2001:db8:900:551::1]
  3    67 ms    71 ms    76 ms  2001:db8:800:1003::209:55
  4    68 ms     *       70 ms  2001:db8:1:0:87:86:71:240
  5    69 ms     *       71 ms  2001:db8:1:0:87:86:77:67
  6    72 ms     *       71 ms  2001:db8:1:0:86:87:77:81
  7    71 ms     *       71 ms  2001:db8:1:0:87:86:77:83
  8    90 ms     *       81 ms  2001:db8:1:0:87:86:77:62
  9    84 ms     *       88 ms  2001:db8:1:0:87:86:77:71
 10    99 ms    83 ms    83 ms  2001:db8:1:0:87:86:77:249
 11    94 ms    87 ms    87 ms  20gigabitethernet4-3.core1.fra1.he.net [2001:7f8::1b1b:0:1]
 12    96 ms    99 ms    99 ms  10gigabitethernet1-4.core1.ams1.he.net [2001:470:0:47::1]
 13   105 ms   108 ms   107 ms  2001:7f8:8:5:0:73e6:0:1
 14   106 ms   107 ms   104 ms  virtualhost.in-berlin.de [2001:bf0:c000:a::2:132]

La route est identifiée.
```

Table des figures

Liste des tableaux

Index

HOSTNAME_IP6, 4

IPV6_NET_N, 4
IPV6_NET_x, 4
IPV6_NET_x_ADVERTISE, 5
IPV6_NET_x_ADVERTISE_DNS, 6
IPV6_NET_x_DEV, 5
IPV6_NET_x_DHCP, 6
IPV6_NET_x_NAME, 6
IPV6_NET_x_TUNNEL, 5
IPV6_ROUTE_N, 9
IPV6_ROUTE_x, 9
IPV6_TUNNEL_N, 6
IPV6_TUNNEL_x_DEFAULT, 7
IPV6_TUNNEL_x_DEV, 8
IPV6_TUNNEL_x_LOCALV4, 7
IPV6_TUNNEL_x_LOCALV6, 8
IPV6_TUNNEL_x_MTU, 8
IPV6_TUNNEL_x_PASSWORD, 8
IPV6_TUNNEL_x_PREFIX, 7
IPV6_TUNNEL_x_REMOTEV4, 7
IPV6_TUNNEL_x_REMOTEV6, 8
IPV6_TUNNEL_x_TIMEOUT, 9
IPV6_TUNNEL_x_TUNNELID, 9
IPV6_TUNNEL_x_TYPE, 7
IPV6_TUNNEL_x_USERID, 8

OPT_IPV6, 4

PF6_FORWARD_ACCEPT_DEF, 11
PF6_FORWARD_LOG, 11
PF6_FORWARD_LOG_LIMIT, 11
PF6_FORWARD_N, 12
PF6_FORWARD_POLICY, 11
PF6_FORWARD_REJ_LIMIT, 12
PF6_FORWARD_UDP_REJ_LIMIT, 12
PF6_FORWARD_x, 12
PF6_FORWARD_x_COMMENT, 12
PF6_INPUT_ACCEPT_DEF, 10
PF6_INPUT_ICMP_ECHO_REQ_LIMIT, 10
PF6_INPUT_ICMP_ECHO_REQ_SIZE, 10
PF6_INPUT_LOG, 10
PF6_INPUT_LOG_LIMIT, 10
PF6_INPUT_N, 11
PF6_INPUT_POLICY, 9
PF6_INPUT_REJ_LIMIT, 10
PF6_INPUT_UDP_REJ_LIMIT, 10
PF6_INPUT_x, 11
PF6_INPUT_x_COMMENT, 11
PF6_LOG_LEVEL, 9
PF6_OUTPUT_ACCEPT_DEF, 12
PF6_OUTPUT_LOG, 13
PF6_OUTPUT_LOG_LIMIT, 13
PF6_OUTPUT_N, 13
PF6_OUTPUT_POLICY, 12
PF6_OUTPUT_REJ_LIMIT, 13
PF6_OUTPUT_UDP_REJ_LIMIT, 13
PF6_OUTPUT_x, 13
PF6_OUTPUT_x_COMMENT, 13
PF6_POSTROUTING_N, 14
PF6_POSTROUTING_x, 14
PF6_POSTROUTING_x_COMMENT, 14
PF6_PREROUTING_N, 14
PF6_PREROUTING_x, 14
PF6_PREROUTING_x_COMMENT, 14
PF6_USR_CHAIN_N, 14
PF6_USR_CHAIN_x_NAME, 14
PF6_USR_CHAIN_x_RULE_N, 14
PF6_USR_CHAIN_x_RULE_x, 14
PF6_USR_CHAIN_x_RULE_x_COMMENT, 14