Package OPENVPN
Version 3.10.18

Claas Hilbrecht
email: babel (+at+) fli4l dot de

the fli4l-Team
email: team@fli4l.de

September 15, 2019

Contents

1. Documentation Of Package OPENVPN

1.1. OpenVPN - VPN Support

As of version 2.1.5 package OpenVPN is part of fli4l.

Important: *For using OpenVPN over the Internet a flatrate or billing based on data volume is a must have! If the fli4l router is powered on the connection will never be hung up because a small amount of data is permanently transferred by OpenVPN. Using a VPN Tunnel over the Internet thus can cause high online costs. The same is applying for an ISDN connection being used for OpenVPN.*

Besides OpenVPN another VPN package exists: OPT_PoPToP (see opt-database http://www.fli4l.de/download/zusatzpakete/).

Deciding which VPN solution to use is driven by security and function concerns. No advices on security of the different packages are given by the team. In unsure, see

Linux-Magazine January 2004

http://diswww.mit.edu/bloom-picayune/crypto/14238

http://sites.inka.de/bigred/archive/cipe-l/2003-09/msg00263.html

Concerning functionality a clear advice can be given to use OpenVPN which outperforms both CIPE and poptop here. OpenVPN supports tunnel mode, bridge mode, data compression and is more solid than CIPE on a fli4l router. OpenVPN has a Windows version to be used as of Windows 2000. Only disadvantages against CIPE are the sheer size in opt archive and missing OpenVPN support for fli4l version 2.0.x.

1.1.1. OpenVPN - Introductive Example

To introduce you to OpenVPN's configuration at first a small example. Two networks that both use a fli4l router shall be connected over the Internet. OpenVPN establishes an encrypted tunnel on both fli4l routers to let computers from both nets communicate with each other. The configuration variables shown in picture 1.1 are used for this purpose.

local net, remote net represent the two nets to be connected by the tunnel. They have to be in different TCP/IP ranges and should have different non interfering net masks. The settings from `IP_NET_x` (Page **??**) in both routers base.txt configuration files hence have to be different. Thus it is not possible to connect two nets over a tunnel that both use IP range 192.168.6.0/24.

transport net The transport network consists of two elements:

- the connection between two OpenVPN daemons, described by *remote_host:remote_port* and *(local_host:)local_port*. This is an equivalent to the OpenVPN settings in `OPENVPN_x_REMOTE_HOST`, `OPENVPN_x_REMOTE_PORT`, `OPENVPN_x_LOCAL_HOST` and `OPENVPN_x_LOCAL_PORT`.

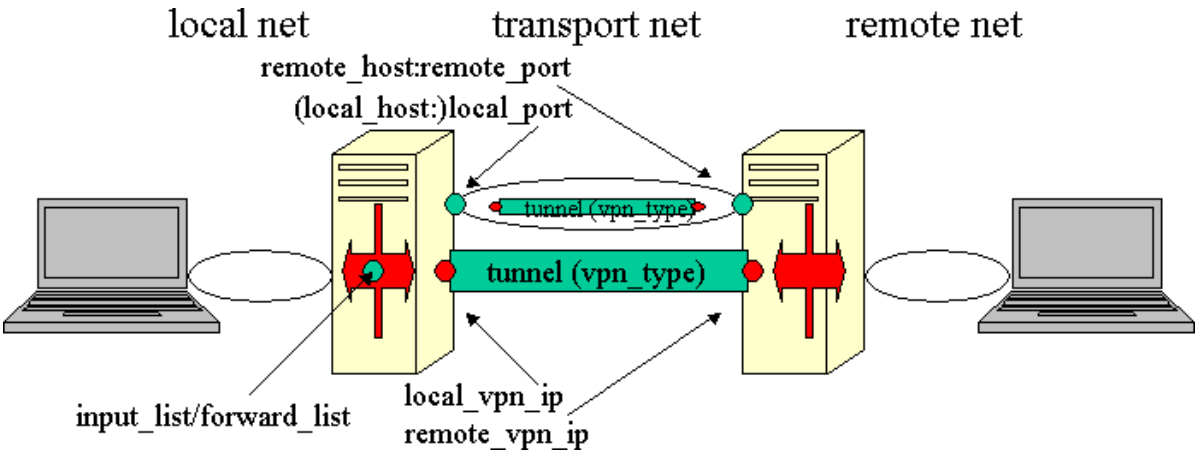

Figure 1.1.: VPN configuration example — tunnel between two routers

- and a tunnel over which the connection between the two OpenVPN-Daemons is established, described by *local_vpn_ip*/*remote_vpn_ip*. This again is an equivalent to the OpenVPN settings in `OPENVPN_x_LOCAL_VPN_IP` and `OPENVPN_x_REMOTE_VPN_IP`. Both VPN IP addresses have be set to values of non existent nets on both routers.

input_list, forward_list Packets that should be sent over this tunnel have to pass the packet filter at first. It will only allow ICMP messages (i.e. ping) as the default which can be used to test the tunnel. Everythihng else has to be allowed explicitly. In the simplest case this is done by

```
OPENVPN_x_PF_INPUT_POLICY='ACCEPT'
OPENVPN_x_PF_FORWARD_POLICY='ACCEPT'
```

Plese note that „accepting“ a complete VPN connection is very critical in terms of security. Better use the tmpl: syntax of the packet filter to only allow those services needed.

No more settings are required for a simple VPN tunnel. All other configuration handles extended functions or special use cases. You should use those after establishing a working tunnel with this minimal configuration.

1.1.2. OpenVPN - Configuration

Because of the complexity of OpenVPN we start by explaining settings required for any VPN connection. Don't try extended configurations for OpenVPN before establishing a connection with minimal settings.

OPT_OPENVPN Default: `OPT_OPENVPN='no'`

`'yes'` activates package OpenVPN. `'no'` deactivates package OpenVPN completely.

OPENVPN_N Default: `OPENVPN_N='0'`

How many OpenVPN configurations are active in the configuration file?

OPENVPN_x_REMOTE_HOST Default: `OPENVPN_x_REMOTE_HOST="`

IP address or DNS address of the remote OpenVPN. For a Roadwarrior (Page 23) this line has to be completely omitted. If omitted OpenVPN waits for connection establishment and doesn't try to connect by itself.

OPENVPN_x_REMOTE_HOST_N Default: `OPENVPN_x_REMOTE_HOST_N='0'`

Using dynamic DNS services is not alsways 100% reliable. You may simply use two ore more of those DynDNS services to register your current IP address with all of them at the same time. To enable OpenVPN to go through the whole DynDNS names a list of *additional* DNS names has to be set. By the help of `OPENVPN_x_REMOTE_HOST` OpenVPN will try to contact these addresses in random order. Hence `OPENVPN_x_REMOTE_HOST` has to exist and be configured correctly!

OPENVPN_x_REMOTE_HOST_x Default: `OPENVPN_x_REMOTE_HOST_x="`

Same description as above applies here `OPENVPN_x_REMOTE_HOST` (Page 5).

OPENVPN_x_REMOTE_PORT Default: `OPENVPN_x_REMOTE_PORT="`

Each OpenVPN connection does need an unused port address on the fli4l router. It is adviced to use ports above 10000 for those are not commonly used. If configuring a connection for a remote station with dynamically changing IP address that has no DynDNS address omit this entry as well as `OPENVPN_x_REMOTE_HOST`.

OPENVPN_x_LOCAL_HOST Default: `OPENVPN_x_LOCAL_HOST="`

Specifies to what IP address OpenVPN will bind. For connections over the Internet this entry should be completely omitted. If an address is set here OpenVPN will only listen for incoming traffic on this IP. If you want to secure a WLAN connection you should set the IP address of fli4l's WLAN interface card here.

OPENVPN_x_LOCAL_PORT Default: `OPENVPN_x_LOCAL_PORT="`

Specifies the port number the local OpenVPN daemon will listen to. For each OpenVPN connection you need a reserved port that only can be used by this connection. Other software on the router is not allowed to use this port. `OPENVPN_x_REMOTE_PORT` and `OPENVPN_x_LOCAL_PORT` of each OpenVPN connection have to match! If setting `OPENVPN_x_REMOTE_PORT='10111'` on one side of the tunnel `OPENVPN_x_LOCAL_PORT='10111'` *has to be set* on the other side as well.

Again: It is very important to match these settings to the according remote OpenVPN station otherwise a connection is not possible between OpenVPN partners.

To enable OpenVPN to listen to incoming connections OpenVPN itself opens the ports in the packet filter set in `OPENVPN_x_LOCAL_PORT`. If this is not your wish then you may change this behavior in `OPENVPN_DEFAULT_OPEN_OVPNPORT` (Page 11). It is *not* necessary to set `OPENVPN_DEFAULT_OPEN_OVPNPORT='yes'` because this is the default behavior!

OpenVPN does not work with ports lower than 1025. If i.e. OpenVPN should work as a tcp-server on port 443 (https port) you have to forward this port via the packet filter to a port above 1024. If i.e. OpenVPN is listening on port 5555 and port 443 should be forwarded there `PF_PREROUTING` has to be set like this:

```
PF_PREROUTING_5='tmpl:https dynamic REDIRECT:5555'
```

OPENVPN_x_SECRET Default: `OPENVPN_x_SECRET=''`

OpenVPN needs a keyfile for encrypting an OpenVPN connection. This keyfile can be generated unter Windows or Linux by OpenVPN itself. Beginners may install OpenVPN's Windows software or use OpenVPN's WebGUI. If you do not want to use OpenVPN under Windows but only generate the needed keyfiles it is enough to install *OpenVPN User-Space Components*, *OpenSSL DDLs*, *OpenSSL Utilities*, *Add OpenVPN to PATH* and *Add Shortcuts to OpenVPN*. With choosing *Generate a static OpenVPN key* from the OpenVPN start menu the keyfiles needed can be generated. At the end the message „Randomly generated 2048 bit key written to `C:/Program files/OpenVPN/config/key.txt`" will appear. The file `key.txt` is the one we need. Copy this file into the directory `<config>/etc/openvpn` and change its name `key.txt` to something more meaningful. Keyfiles can also be generated automatically by the fli4l router if you set `OPENVPN_CREATE_SECRET` to `'yes'` and reboot fli4l. If configuring OpenVPN for the first time enter all data in the config file and either set `OPENVPN_DEFAULT_CREATE_SECRET` (Page 11) to `'yes'` if one keyfile should be used for all connections or if a keyfile for only one connection should be generated set `OPENVPN_x_CREATE_SECRET` to `'yes'`. After boot of the fli4l router one or more keyfiles will be created automatically and saved to `/etc/openvpn` with the name specified. Keyfile(s) can be copied via scp or other medias. After creation of keyfiles change the setting back to 'no' and build a new boot media for fli4l with the configuration and keyfiles you just created. If you forget to change 'yes' to 'no' fli4l will generate new keyfiles with each reboot but no OpenVPN daemon will be started and thus no tunnels can be established. If you set `OPENVPN_x_CREATE_SECRET` to `'webgui'` you can use the web interface to generate keyfiles. Use OpenVPN's WebGUI in detail view for connections and choose 'Keymanagement'. For reference see 1.1.6

Hint: By executing

```
openvpn --genkey --secret <filename>
```

you can generate a keyfile by hand via fli4l's console.

Keyfiles have to be copied to the directory `<config>/etc/openvpn` as seen in the following picture. The file name of the keyfiles without path has to set in `OPENVPN_x_SECRET`. In this way keyfiles will be copied to the opt-archive while creating the boot media.

OPENVPN_x_TYPE Default: `OPENVPN_x_TYPE=''`

An OpenVPN connection either can be used as a tunnel or as a bridge. Through an OpenVPN tunnel only IP traffic can be routed. A bridge transfers ethernet frames i.e. not only IP traffic but also IPX or NetBEUI or else. For using OpenVPN to transfer ethernet frames package advanced_networking is needed in addition. Please note that a bridge over a DSL line can be really slow!

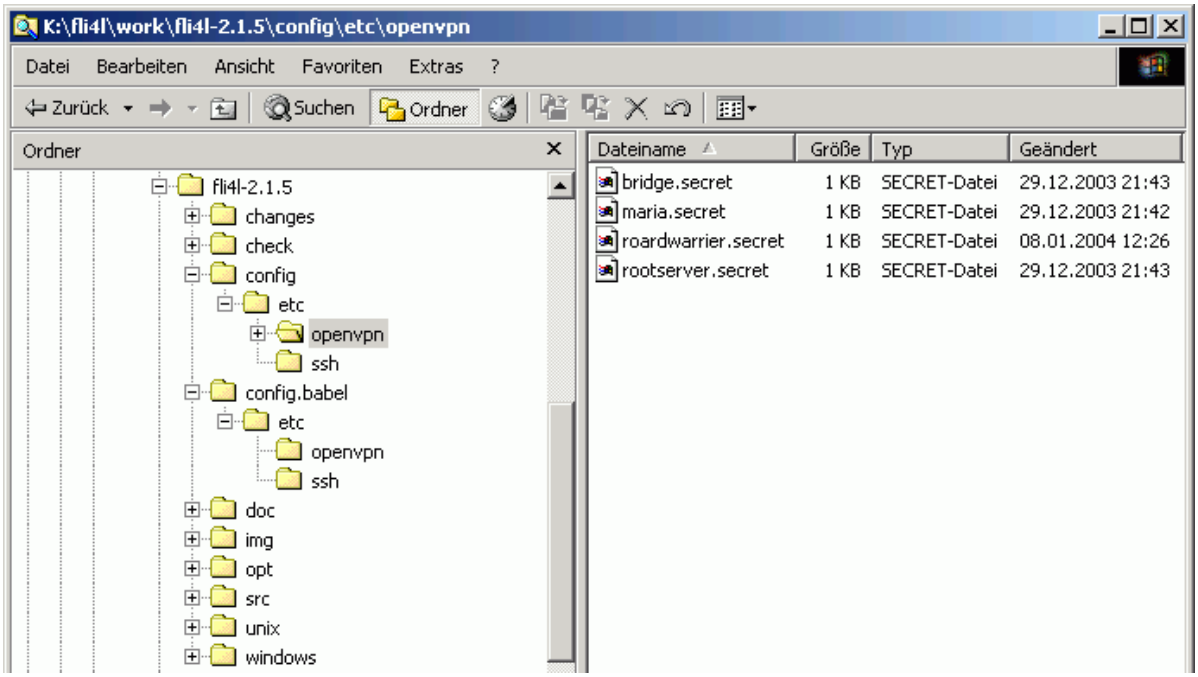

Figure 1.2.: fli4l config directory with OpenVPN *.secret files

1.1.3. OpenVPN - Bridge configuration

For using OpenVPN as a bridge the following entries are valid. Please note that when using a bridge over the Internet broadcast traffic uses already a rather high bandwidth without any real data being transferred.

Remember that the following settings are only valid if for this connection `OPENVPN_x_TYPE` (Page 6) is set to `'bridge'`! A configured bridge from package advanced_networking to which the VPN connection can bind is needed additionally.

OPENVPN_x_BRIDGE Default: `OPENVPN_x_BRIDGE=""`

Holds the name of the bridge this OpenVPN connection should bind to. If `BRIDGE_DEV_x_NAME='cuj-br'` is given and the OpenVPN connection should bind to that bridge `'cuj-br'` has to be set in accordance.

OPENVPN_x_BRIDGE_COST Default: `OPENVPN_x_BRIDGE_COST=""`

If using spanning tree protocol (STP, see http://de.wikipedia.org/wiki/Spanning_Tree or documentation for package advanced_networking) you can specify the connection costs here.

OPENVPN_x_BRIDGE_PRIORITY Default: `OPENVPN_x_BRIDGE_PRIORITY=""`

If using STP (spanning tree protocol, see http://de.wikipedia.org/wiki/Spanning_Tree or documentation for package advanced_networking) you can specify connection priority here.

1.1.4. OpenVPN - Tunnel configuration

OPENVPN_x_REMOTE_VPN_IP Default: `OPENVPN_x_REMOTE_VPN_IP=""`

This setting is only valid if `OPENVPN_x_TYPE` (Page 6) is set to `'tunnel'` for this OpenVPN connection!

VPN IP address of the OpenVPN remote station. VPN IP addresses are needed only for routing and can be chosen nearly free. The following restrictions apply:

- IP address may not be used in local network and thus can't be in the subnet of the fli4l router.
- IP address may not be used for any local network device.
- IP address may not belong to any network routed by `IP_ROUTE_x`.
- IP address may not belong to any network routed by `ISDN_CIRC_ROUTE_x`.
- IP address may not belong to any network routed by `CIPE_ROUTE_x`.
- IP address may not belong to any network routed by `OPENVPN_ROUTE_x`.
- IP address may not belong to any network used or routed by fli4l in any other way.

As you see VPN IP addresses can't be used anywhere else. Before beginning to configure OpenVPN you should look for an unused net in both local and remote address ranges. It should belong to private address ranges (see http://ftp.univie.ac.at/netinfo/rfc/rfc1597.txt).

OPENVPN_x_LOCAL_VPN_IP Default: `OPENVPN_x_LOCAL_VPN_IP=""`

This setting is only valid if `OPENVPN_x_TYPE` (Page 6) is set to `'tunnel'` for this OpenVPN connection.

IP address for the local OpenVPN device tunX. Same restrictions as in `OPENVPN_x_REMOTE_VPN_IP` (Page 8) apply here.

By the way, it is possible to use the same IP address as in `OPENVPN_x_LOCAL_VPN_IP` for all local OpenVPN connections. This enables a host host to use the same IP address in all VPNs. Packet filter rules are drastically easier to configure this way.

OPENVPN_x_IPV6 Default: `OPENVPN_x_IPV6='no'`

This enables native IPv6 support in OpenVPN. Consider this as an experimental feature because of the code being brandnew. Of course OPT_IPV6 has to be activated and configured as well then. For `OPENVPN_x_IPV6='no'` and/or `OPT_IPV6='no'` all relevant variables are ignored.

ATTENTION!!! These settings are not checked for overlapping with other parts of the configuration! This applies to `OPENVPN_x_LOCAL_VPN_IPV6`, `OPENVPN_x_REMOTE_VPN_IPV6` and `OPENVPN_x_ROUTE_x`.

OPENVPN_x_REMOTE_VPN_IPV6 Default: `OPENVPN_x_REMOTE_VPN_IPV6=""`

For IPv6 the same restrictions apply as for `OPENVPN_x_REMOTE_VPN_IP` (Page 8).

```
OPENVPN_X_REMOTE_IPV6='FD00::1'
```

OPENVPN_x_LOCAL_VPN_IPV6 Default: `OPENVPN_x_LOCAL_VPN_IPV6=''`

For IPv6 applies the same as for `OPENVPN_x_LOCAL_VPN_IP` (Page 8). If no subnet is set /64 will be used as a default.

```
OPENVPN_X_LOCAL_IPV6='FD00::2/112'
```

OPENVPN_x_ROUTE_N Default: `OPENVPN_x_ROUTE_N=''`

This setting is only valid if `OPENVPN_x_TYPE` (Page 6) is set to `'tunnel'` for this OpenVPN connection.

Routes are being set automatically by OpenVPN when starting up. Up to 50 nets can be routed over a single OpenVPN connection. For every net to be routed a valid `OPENVPN_x_ROUTE_x` entry must be created.

Please note that the packet filter rules necessary have to be set manually in `OPENVPN_PF_FORWARD_x` `OPENVPN_PF_INPUT_x` res. `OPENVPN_PF6_FORWARD_x` `OPENVPN_PF6_INPUT_x`. OpenVPN only allows ICMP over a VPN connection and denies all other data traffic. Details can be found at `OPENVPN_x_PF_INPUT_N` (Page 16) and `OPENVPN_x_PF_FORWARD_N` (Page 17) res. at `OPENVPN_x_PF6_INPUT_N` (Page 17) and `OPENVPN_x_PF6_FORWARD_N` (Page 18).

OPENVPN_x_ROUTE_x Default: `OPENVPN_x_ROUTE_x=''`

Specify the nets to be reached over the OpenVPN remote station here. If on the remote side i.e. the nets 192.168.33.0/24 and 172.18.0.0/16 can be reached and should be accessed through the OpenVPN tunnel both of them have to be entered under `OPENVPN_x_ROUTE_x`. Host routes (`/32`) may be set here as well.

If the default route should be reached through an OpenVPN tunnel specifiy 0.0.0.0/0 res. ::/0 for IPv6 and an optional flag as routes here. For IPv6 routes OPT_IPv6 has to be activated, local and remote IPv6 addresses for the tunnel have to be set and OPENVPN_x_IPV6 must be 'yes'. OpenVPN has several alternative ways to set a default route which can be chosen by a flag. Each method has its own advantages and disadvantages. At the moment the following flags are supported:

local The *local* flag should be chosen if the OpenVPN remote station is located in a subnet that can be reached directly by the fli4l router. This may be the case for example for an OPENVPN default route over WLAN.

def1 With this flag two new routes 0.0.0.0/1 and 128.0.0.0/1 will be defined in addition to a host route to the OpenVPN remote station. This routes act as default routes for the complete (encrypted) traffic to the OpenVPN remote station (which can be reached over the host route).

If omitting the optional flag OpenVPN will choose the method of setting default routes. Methods will be picked by the OpenVPN version. At the moment *local* is the default advised.

```
OPENVPN_1_ROUTE_N='3'
OPENVPN_1_ROUTE_1='192.168.33.0/24'
OPENVPN_1_ROUTE_2='172.18.0.0/16'
OPENVPN_1_ROUTE_3='2001:db8:/32'
```

OpenVPN - Delegation Of DNS and Reverse-DNS

OPENVPN_x_DOMAIN Default: `OPENVPN_x_DOMAIN=""`

This parameter sets the remote domain. The variable can hold multiple domains which have to be separated by spaces then. If only this parameter is set (without mentioning of an additional DNS server) it will be assumed that a DNS server is listening on the IP of the other end of the tunnel (see `OPENVPN_x_REMOTE_VPN_IP` (Page 8)). On the remote router incoming DNS queries have to allowed in this case. (i.e. via `OPENVPN_x_INPUT_y='tmpl:dns ACCEPT'`)

OPENVPN_x_ROUTE_x_DOMAIN Default: `OPENVPN_x_ROUTE_x_DOMAIN=""`

Different subnets can have different domains assigned. Per `OPENVPN_x_ROUTE_y` one according domain can be configured. If a `OPENVPN_x_ROUTE_y_DNSIP` exists for the domain, it will be used, else the one set at `OPENVPN_x_DNSIP`. The effect is the same as with `OPENVPN_x_DOMAIN` but this method allows better documentation.

OPENVPN_x_DNSIP Default: `OPENVPN_x_DNSIP=""`

If the tunnel end point is not the appropriate DNS server set the IP of the appropriate one here. If this is empty the one at `OPENVPN_x_REMOTE_VPN_IP` (Page 8) will be used.

OPENVPN_x_ROUTE_x_DNSIP Default: `OPENVPN_x_ROUTE_x_DNSIP=""`

Multiple subnets routed can also have different DNS servers - define one per `OPENVPN_x_ROUTE_x` (Page 9) here.

1.1.5. Expert Settings

Settings described in this chapter are all optional and should only be changed if the OpenVPN connection is working but should be optimized (for example by the use of another encryption algorithm).

All settings in `OPENVPN_DEFAULT_` are otional. This means they don't have to be written in the config file. If an entry is missing in openvpn.txt the OpenVPN start script will use the default value described here. If you don't want to change this defaults do not write them to the openvpn.txt config file!

General Settings

OPENVPN_DEFAULT_CIPHER Default: `OPENVPN_DEFAULT_CIPHER='BF-CBC'`

One of the available encryption methods. Method `'BF-CBC'` is used as a default by all OpenVPN versions (also non-fli4l specific versions).

OPENVPN_DEFAULT_COMPRESS Default: `OPENVPN_DEFAULT_COMPRESS='yes'`

OpenVPN uses adaptive LZO data compression to enlarge the bandwidth of a connection. Adaptive means OpenVPN recognizes by itself when i.e. already compressed zip files are sent over an OpenVPN connection. In such case data compression will be switched off until data is sent that will benefit from data compression. There is nearly no cause for deactivating data compression because this enlarges bandwidth at nearly no cost. Only disadvantage of data compression is a small increase of latency by some milliseconds.

For online games via VPN which need a ”good” ping, i.e. low latency it may be wise to deactivate data compression.

OPENVPN_DEFAULT_CREATE_SECRET Default: `OPENVPN_DEFAULT_CREATE_SECRET='no'`

This setting will cause OpenVPN to automatically generate keyfiles on boot of the fli4l router. An OpenVPN connection won't be started then. For details see OPENVPN_x_SECRET (Page 6).

OPENVPN_DEFAULT_DIGEST Default: `OPENVPN_DEFAULT_DIGEST='SHA1'`

Enter available checksums her. OpenVPN uses `'SHA1'` as default.

OPENVPN_DEFAULT_FLOAT Default: `OPENVPN_DEFAULT_FLOAT='yes'`

OpenVPN remote stations that use DynDNS addresses can change their IP address at any time. To make OpenVPN accept this changed IP address set `OPENVPN_DEFAULT_FLOAT` to `'yes'`. If `'no'` is set changing of an IP address is not allowed. This only makes sense with WLAN connections or connections to remote stations with static IP addresses (i.e. some provider's root servers). This setting can be superseded be a per connection setting as all other `OPENVPN_DEFAULT_` settings can.

OPENVPN_DEFAULT_KEYSIZE Default: `OPENVPN_DEFAULT_KEYSIZE="`

Keysize depends on the encryption method used. Only change this setting when connecting to an OpenVPN remote station that does not use default settings and which you have no influence on. If keysize can be determined by you this value should stay empty. OpenVPN will use the optimal keysize for the encryption method used then.

OPENVPN_DEFAULT_OPEN_OVPNPORT Default: `OPENVPN_DEFAULT_OPEN_OVPNPORT='yes'`

fli4l's packet filter rules have to be changed to enable OpenVPN connections. For all TCP or UDP ports (see `OPENVPN_x_PROTOCOL`) OpenVPN should listen on `PF_INPUT_x` (Page **??**) in base.txt has to be adapted. By specifiying `'yes'` these packet filter rules will be generated automatically. For some connections it may make sense to set `'no'` and define the rules yourself.

OPENVPN_DEFAULT_ALLOW_ICMPPING Default: `OPENVPN_DEFAULT_ALLOW_ICMPPING='yes'`

`'yes'` configures the packet filter for the connection to let pass ping data packets. If there is no really good cause ICMP ping should be allowed at any time. This setting is *not* equivalent to OpenVPN's ping option!

OPENVPN_DEFAULT_PF_INPUT_LOG Default: `OPENVPN_DEFAULT_PF_INPUT_LOG='BASE'`

`'yes'` or `'no'` sets whether the packet filter should protocol denied incoming packets for the VPN connection in the INPUT list or not. By specifiying `'BASE'` the setting from `'PF_INPUT_LOG='` in base.txt will be used.

OPENVPN_DEFAULT_PF_INPUT_POLICY Default: `OPENVPN_DEFAULT_PF_INPUT_POLICY='REJECT'`

This setting equals `'PF_INPUT_POLICY='` (Page **??**) in base.txt. By specifiying `'BASE'` the setting from `'PF_INPUT_POLICY='` in base.txt will be used.

OPENVPN_DEFAULT_PF_FORWARD_LOG Default: `OPENVPN_DEFAULT_PF_FORWARD_LOG='BASE'`

`'yes'` or `'no'` sets whether the packet filter should protocol denied incoming packets for the VPN connection in the FORWARD list or not. By specifiying `'BASE'` the setting from `'PF_FORWARD_LOG='` in base.txt will be used.

OPENVPN_DEFAULT_PF_FORWARD_POLICY Default: `OPENVPN_DEFAULT_PF_FORWARD_POLICY='REJECT'`

This setting equals `'PF_FORWARD_POLICY='` (Page **??**) in base.txt. By specifiying `'BASE'` the setting from `'PF_FORWARD_POLICY='` in base.txt will be used.

OPENVPN_DEFAULT_PING Default: `OPENVPN_DEFAULT_PING='60'`

To keep an established tunnel open and to recognize if the OpenVPN remote station can still be reached an encrypted ping will be sent over the line in the interval in seconds specified here. `'off'` does not send pings over the line but only real user data.

OPENVPN_DEFAULT_PING_RESTART Default: `OPENVPN_DEFAULT_PING_RESTART='180'`

If in the time interval set here no ping or other data is transferred succesfully the VPN connection concerned will be restarted. The value in `OPENVPN_DEFAULT_PING_RESTART` has to be greater than the one in `OPENVPN_DEFAULT_PING`. `'off'` disables automatic restart.

OPENVPN_DEFAULT_RESOLV_RETRY Default: `OPENVPN_DEFAULT_RESOLV_RETRY='infinite'`

If `OPENVPN_x_REMOTE_HOST` or `OPENVPN_x_LOCAL_HOST` holds DNS names instead of IP addresses they have to be resolved to IP addresses when starting an OpenVPN connection. If this fails OpenVPN will retry to resolve the DNS name for the timespan set here. If this doesn't work within the time limit set here no OpenVPN connection will be established. With `'infinite'` OpenVPN will try forever to resolve the DNS name. Only change this setting if if you know what you're doing!

OPENVPN_DEFAULT_RESTART Default: `OPENVPN_DEFAULT_RESTART='ip-up'`

After disconnection of a tunnel an immedeate restart should be done in order to keep disconnection time as small as possible. For all OpenVPN connections made over dial-in lines like DSL or ISDN `'ip-up'` should be specified here. `'never'` should be set instead for OpenVPN connections over WLAN because of reconnection being independent of dial-ins. For OpenVPN tunnels over an ISDN dial-in connection being established with `ISDN_CIRC_x_TYPE='raw'` `'raw-up'` has to be set here.

OPENVPN_DEFAULT_PROTOCOL Default: `OPENVPN_DEFAULT_PROTOCOL='udp'`

This variable sets which protocol should be used as default. UDP is a a good choice normally but sometimes only TCP is allowed, which has a remarkable overhead. Possible values are `'udp'`, `'udp6'`, `'tcp-server'`, `'tcp-server6'`, `'tcp-client'` or `'tcp-client6'`. Settings `'tcp-server'` or `'tcp-client'` make only sense if a VPN tunnel has to be established through a number of paket filters or other tunnels. If no special case should be handled *always* use the default setting `'udp'`. By adding '6' the tunnel will be IPv6 capable (WAN) and can be reached over IPv6-Internet.

OPENVPN_DEFAULT_START Default: `OPENVPN_DEFAULT_START='always'`

OpenVPN connections can either be started `'always'` or `'on-demand'`. Particular OpenVPN connections can be started with the OpenVPN WebGUI (see 1.1.6) only when

needed. They can also be started via fli4l console at any time. Login to the fli4l console and execute the following command:

```
cd /etc/openvpn
openvpn --config name.conf --daemon openvpn-name
```

This start an OpenVPN tunnel running in background. Instead of `name.conf` use the name of your configuration file in directory `/etc/openvpn`.

OPENVPN_DEFAULT_VERBOSE Default: `OPENVPN_DEFAULT_VERBOSE='2'`

This variable sets the verbosity of OpenVPN. If a VPN connection is running flawlessly you can set this to `'0'` to avoid all messages. For testing purposes a value of `'3'` is adviced. Higher values may be useful for debugging. Maximum value is `'11'`.

OPENVPN_DEFAULT_MANAGEMENT_LOG_CACHE Default:
`OPENVPN_DEFAULT_MANAGEMENT_LOG_CACHE='100'`

This value controls how many log lines should be saved. Logs can be reviewed in the WebGUI (Page 18).

OPENVPN_DEFAULT_MUTE_REPLAY_WARNINGS Default:
`OPENVPN_DEFAULT_MUTE_REPLAY_WARNINGS='no'`

This variable controls if a warning is posted to the log when receiving double packet for this could point out security problems in a network. When using poor WLAN connections doubled packets may occur rather often. In this case it makes sense to switch the warning off to avoid flooding logfiles. This setting has *no* impact on the security of an OpenVPN connection.

OPENVPN_DEFAULT_MSSFIX Default: `OPENVPN_DEFAULT_MSSFIX="`

Setting MSSFIX defines the size of TCP packets for the VPN connection. `OPENVPN_DEFAULT_MSSFIX='0'` disables this option. If fragment sizes are given and MSSFIX entry is empty fragment sizes will be used automatically. This setting only works with `OPENVPN_x_PROTOCOL='udp'`.

OPENVPN_DEFAULT_FRAGMENT Default: `OPENVPN_DEFAULT_FRAGMENT='1300'`

Activates internal fragmentation of OpenVPN packets with a size of x bytes. This setting only works with `OPENVPN_x_PROTOCOL='udp'`.

`OPENVPN_DEFAULT_FRAGMENT='0'` completely deactivates fragmentation.

OPENVPN_DEFAULT_TUN_MTU Default: `OPENVPN_DEFAULT_TUN_MTU='1500'`

Sets the MTU of the virtual OpenVPN adapter to x bytes. Only change this setting if if you know what you're doing! Usually it is more reasonable to try fragment or MSSFIX options at first.

OPENVPN_DEFAULT_TUN_MTU_EXTRA Default: `OPENVPN_DEFAULT_TUN_MTU_EXTRA="`

If `OPENVPN_x_PROTOCOL='bridge'` is set 32 bytes will be reserved as extra memory for managing the buffers for the tap device. With `OPENVPN_x_PROTOCOL='tunnel'` no extra memory is reserved. This only affects the memory footprint in the router and has no influence on the amount of data sent over the tunnel.

OPENVPN_DEFAULT_LINK_MTU Default: `OPENVPN_DEFAULT_LINK_MTU="`

Sets the MTU of an OpenVPN connection to x bytes. Only use this setting if if you know what you're doing! Usually it is more reasonable to try fragment or MSSFIX options at first.

OPENVPN_DEFAULT_SHAPER Default: `OPENVPN_DEFAULT_SHAPER="`

Restricts *outgoing* bandwidth of the tunnel to the specified value of bytes per second. Possible range is from 100 up to 100000000 bytes. For values up to 1000 bytes per second reduce MTU of the connection otherwise ping times will increase significantly. If you want to restrict a tunnel to a certain bandwidth in both directions you have to configure this option on both OpenVPN end points separately.

In modern OpenVPN versions shaping is not working correctly. Data transfer rates in tunnels using shaping may be extremely fluctuating or even not work at all. Problems may occur in completely different ways depending on the hardware used and lead to unpredictable behavior. Please use shaping with care at the moment. If in doubt deactivate or at least test shaping extensively.

OPENVPN_EXPERT Default: `OPENVPN_EXPERT='no'`

Expert mode enables you to use native Openvpn config files. These have to be stored in the config directory etc/openvpn and etc/openvpn/scripts. All files found there will be transferred to the router.

Expert mode ignores all config settings thus `OPENVPN_N='0'` has to be set.

Expert mode creates no firewall rules. You will have to place them in base.txt by yourself.

Connection-specific Settings

The following OpenVPN options only are valid for the connection mentioned. Only a few of them are mandatory while the most can be omitted. All default settings are taken from `OPENVPN_DEFAULT_x`. Changing values in `OPENVPN_DEFAULT_` applies to all connections that do not explicitely change defaults.

OPENVPN_x_NAME Default: `OPENVPN_x_NAME="`

Defines a name for the OpenVPN connection with up to 16 characters. A config file with this name and suffix .conf will be created in directory /etc/openvpn. This name will appear in syslogs as well. Example: if the name `'peter'` is entered in syslog the connection will appear as 'openvpn-peter'. This helps to identify connections. A name may contain characters, numbers and the '-'.

OPENVPN_x_ACTIV Default: `OPENVPN_x_ACTIV='yes'`

If you want to deactivate an OpenVPN connection but keep the config file it can be disabled by specifying 'no'. Config files will be written to rc.cfg but no corresponding connection will be created.

OPENVPN_x_CHECK_CONFIG Default: `OPENVPN_x_CHECK_CONFIG='yes'`

OpenVPN's extended config file checks are too stringent in rare cases. For example if an ISDN backup connection uses the same routing entries as a connection over the Internet

extended checks will complain. In this case extended checking should be disabled for the backup connection. Set OPENVPN_x_CHECK_CONFIG='no' to switch off extended checking for this connection.

OPENVPN_x_CIPHER Default see: OPENVPN_DEFAULT_CIPHER

See OPENVPN_DEFAULT_CIPHER (Page 10). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_COMPRESS Default see: OPENVPN_DEFAULT_COMPRESS

See OPENVPN_DEFAULT_COMPRESS (Page 10). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_CREATE_SECRET Default see: OPENVPN_DEFAULT_CREATE_SECRET='no'

See OPENVPN_x_SECRET (Page 6). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_DIGEST Default see: OPENVPN_DEFAULT_DIGEST

See OPENVPN_DEFAULT_DIGEST (Page 11). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_FLOAT Default see: OPENVPN_DEFAULT_FLOAT

See OPENVPN_DEFAULT_FLOAT (Page 11). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_KEYSIZE Default see: OPENVPN_DEFAULT_KEYSIZE

See OPENVPN_DEFAULT_KEYSIZE (Page 11). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_ISDN_CIRC_NAME Default OPENVPN_x_ISDN_CIRC_NAME=”

Specifies on which ISDN circuit the OpenVPN connection will be established. Enter the name of the ISDN circuits defined in ISDN_CIRC_x_NAME=" (Page **??**). The ISDN Circuit has to be of type 'raw'.

OPENVPN_x_PING Default see: OPENVPN_DEFAULT_PING

See OPENVPN_DEFAULT_PING (Page 12). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_PROTOCOL Default: OPENVPN_x_PROTOCOL='udp'

Specifies the protocol to be used for establishing an OpenVPN tunnel. Possible values are 'udp', 'udp6', 'tcp-server', 'tcp-server6', 'tcp-client' or 'tcp-client6'. Settings 'tcp-server' or 'tcp-client' make only sense if a VPN tunnel has to be established through a number of paket filters or other tunnels. If no special case should be handled *always* use the default setting 'udp'. By adding '6' the tunnel will be IPv6 capable (WAN) and can be reached over IPv6-Internet.

OPENVPN_x_RESOLV_RETRY Default see: OPENVPN_DEFAULT_RESOLV_RETRY

See OPENVPN_DEFAULT_RESOLV_RETRY (Page 12). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_PING_RESTART Default see: OPENVPN_DEFAULT_PING_RESTART

See OPENVPN_DEFAULT_PING_RESTART (Page 12). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_START Default see: OPENVPN_DEFAULT_START

See OPENVPN_DEFAULT_START (Page 12). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_VERBOSE Default see: OPENVPN_DEFAULT_VERBOSE

See OPENVPN_DEFAULT_VERBOSE (Page 13). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_MANAGEMENT_LOG_CACHE Default see: OPENVPN_DEFAULT_MANAGEMENT_LOG_CACHE

See OPENVPN_DEFAULT_MANAGEMENT_LOG_CACHE (Page 13). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_MUTE_REPLAY_WARNINGS Default see: OPENVPN_DEFAULT_MUTE_REPLAY_WARNINGS

See OPENVPN_DEFAULT_MUTE_REPLAY_WARNINGS (Page 13). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_RESTART Default see: OPENVPN_DEFAULT_RESTART

See OPENVPN_DEFAULT_RESTART (Page 12). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_ALLOW_ICMPPING Default see: OPENVPN_DEFAULT_ALLOW_ICMPPING

See OPENVPN_DEFAULT_ALLOW_ICMPPING (Page 11). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_OPEN_OVPNPORT Default see: OPENVPN_DEFAULT_OPEN_OVPNPORT

See OPENVPN_DEFAULT_OPEN_OVPNPORT (Page 11). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_PF_INPUT_LOG Default see: OPENVPN_DEFAULT_PF_INPUT_LOG

See OPENVPN_DEFAULT_PF_INPUT_LOG (Page 11). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_PF_INPUT_POLICY Default see: OPENVPN_DEFAULT_PF_INPUT_POLICY

See OPENVPN_DEFAULT_PF_INPUT_POLICY (Page 11). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_PF_INPUT_N Default: OPENVPN_x_PF_INPUT_N='0'

Sets the count of the following OPENVPN_x_PF_INPUT_x= entries.

OPENVPN_x_PF_INPUT_x Default: OPENVPN_x_PF_INPUT_x="

Like in package base this variables contain the packet filter rules. The same syntax like in base.txt is used, tmpl: and host aliased are possible as well. In addition you can use some special symbolic names which are:

VPNDEV actual tun device of the der respective OpenVPN connection.

LOCAL-VPN-IP Uses IP addresses from `OPENVPN_x_LOCAL_VPN_IP`.

REMOTE-VPN-IP Uses IP addresses from `OPENVPN_x_REMOTE_VPN_IP`.

REMOTE-NET Uses IP addresses from `OPENVPN_x_REMOTE_VPN_IP` and in addition all nets specified in OPENVPN_x_ROUTE_x.

OPENVPN_x_PF_FORWARD_LOG Default see: `OPENVPN_DEFAULT_PF_FORWARD_LOG`

See `OPENVPN_DEFAULT_PF_FORWARD_LOG` (Page 12). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_PF_FORWARD_POLICY Default see: `OPENVPN_DEFAULT_PF_FORWARD_POLICY`

See `OPENVPN_DEFAULT_PF_FORWARD_POLICY` (Page 12). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_PF_FORWARD_N Default: `OPENVPN_x_PF_FORWARD_N='0'`

Holds the count of the following `OPENVPN_x_PF_FORWARD_x=` entries.

OPENVPN_x_PF_FORWARD_x Default: `OPENVPN_x_PF_FORWARD_x=""`

See `OPENVPN_x_PF_INPUT_x` (Page 16).

OPENVPN_x_PF_PREROUTING_N Default: `OPENVPN_x_PF_PREROUTING_N='0'`

Holds the count of the following `OPENVPN_x_PF_PREROUTING_x=` entries.

OPENVPN_x_PF_PREROUTING_x Default: `OPENVPN_x_PF_PREROUTING_x=""`

See `OPENVPN_x_PF_INPUT_x` (Page 16).

OPENVPN_x_PF_POSTROUTING_N Default: `OPENVPN_x_PF_POSTROUTING_N='0'`

Holds the count of the following `OPENVPN_x_PF_POSTROUTING_x=` entries.

OPENVPN_x_PF_POSTROUTING_x Default: `OPENVPN_x_PF_POSTROUTING_x=""`

As of fli4l version 3.5.0 (or 3.5.0-rev18133 for tarball users) behavior has changed here. Prior to this entries like

```
OPENVPN_1_PF_POSTROUTING_1='MASQUERADE'
```

were valid. As of now giving a source and a target address is mandatory. This was necessary to use the full extent of POSTROUTING rules. In most cases you will only have to adapt rules `IP_NET_x` (Page **??**) and REMOTE-NET.

See `OPENVPN_x_PF_INPUT_x` (Page 16).

OPENVPN_x_PF6_INPUT_N Default: `OPENVPN_x_PF6_INPUT_N='0'`

Holds the count of the following `OPENVPN_x_PF6_INPUT_x=` entries.

OPENVPN_x_PF6_INPUT_x Default: `OPENVPN_x_PF6_INPUT_x=""`

Here the packet rules have to be set like in package IPv6. Syntax is the same as in ipv6.txt. Also tmpl: and host aliases are possible. In addition you can use some special symbolic names. See `OPENVPN_x_PF_INPUT_x` (Page 16) for details.

OPENVPN_x_PF6_FORWARD_N Default: `OPENVPN_x_PF6_FORWARD_N='0'`

Holds the count of the following `OPENVPN_x_PF6_FORWARD_x=` entries.

OPENVPN_x_PF6_FORWARD_x Default: `OPENVPN_x_PF6_FORWARD_x=''`

See `OPENVPN_x_PF6_INPUT_x` (Page 17).

OPENVPN_x_MSSFIX Default see: `OPENVPN_DEFAULT_MSSFIX`

See `OPENVPN_DEFAULT_MSSFIX` (Page 13). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_FRAGMENT Default see: `OPENVPN_DEFAULT_FRAGMENT`

See `OPENVPN_DEFAULT_FRAGMENT` (Page 13). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_TUN_MTU Default see: `OPENVPN_DEFAULT_TUN_MTU`

See `OPENVPN_DEFAULT_TUN_MTU` (Page 13). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_TUN_MTU_EXTRA Default see: `OPENVPN_DEFAULT_TUN_MTU_EXTRA`

See `OPENVPN_DEFAULT_TUN_MTU_EXTRA` (Page 13). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_LINK_MTU Default see: `OPENVPN_DEFAULT_LINK_MTU`

See `OPENVPN_DEFAULT_LINK_MTU` (Page 14). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

OPENVPN_x_SHAPER Default see: `OPENVPN_DEFAULT_SHAPER=''`

See `OPENVPN_DEFAULT_SHAPER` (Page 14). In contradiction to the default setting this setting only affects the OpenVPN connection mentioned.

1.1.6. OpenVPN - WebGUI

As of fli4l version 2.1.10 a WebGUI is present to start and stop OpenVPN connections and for some more basic functions. You will need to activate package mini_httpd. If you set variable `OPENVPN_WEBGUI` in openvpn.txt to 'yes' a menu for OpenVPN will be added to the web interface. An overview of the configured connections is displayed when selecting it along with the state and actions possible. (see figure 1.3).

OpenVPN - WebGUI - Connection Overview

Status: State of a connection is symbolized by traffic light symbols. A red man means no OpenVPN process is running a yellow man means the process is running but no connection could be established (yet) and a green man means that a remote station is „connected“. Details about the connection can be displayed as tool tips for the men. This can be of interest at least for „yellow“ status.

Figure 1.3.: Connection Overview

Name: In this column the name of the OpenVPN connection is displayed as defined in configuration. A click on the name leads to an overview showing more informations concerning the connection. See below for details.

Aktion: The actions provided are symbolized by buttons with a small definition of each realized by a tool tip. These buttons exist:

OpenVPN - WebGUI - Detail View Of A Connection

Statistics: Some interesting statistics are shown here if the connection is started and not on 'hold'.

Log: last 20 lines of the connection logfile. If more lines should be displayed enter the number and click 'show'. If 'all' is selected the whole logfile will be shown. This tab is only shown for started connections.

Debug-Log: Shows the output of the start process. OpenVPN connections are started and the output is shown. This is handy if the connection can't be started by the start button and no normal log can be shown. This tab is only shown for connections not started.

Packet filter: Shows the configuration of the the packet filter relevant to this connection. Packet filter is only configured if the connection is started and configured as a tunnel.

Symbol	Description
	restart OpenVPN process and try to connect.
	stop OpenVPN process.
	reset connection.
	reset connection and put it in 'hold'. No data can be transferrred.
	free connection again. Data can be transferrred.

Table 1.1.: Actions of the OpenVPN-Webgui

Figure 1.4.: Detail view of a connection (Keymanagement)

Bridge: Shows the configuration of bridges on the router. Tab is only shown if the connection is configured as a bridge.

Configuration: Shows the configuration generated on boot for this connection.

Keymanagement: Creates a key for the connection and makes it available for download (see figure 1.4). If no key is present on first start it will be generated and displayed automatically. You may download it via the corresponding symbol or copy/paste it to a text file. To save a newly generated keyfile on the router click the disk symbol. Saving can be undone by clicking the restore symbol.

Support informations: Shows all informations relevant when problems occur. You may copy&paste these informations i.e. for a post on the newsgroups.

1.1.7. OpenVPN - Collaboration Of Different OpenVPN Versions

Please note that different versions of OpenVPN may use different default parameters for a connection. In particular MTU fragment and MSSFIX settings may differ. If values „don't match" connection establishment is not possible or no reliable connection can be made. Typical error messages can be:

```
FRAG_IN error flags=0xfa2a187b: FRAG_TEST not implemented
FRAG_IN error flags=0xfa287f34: spurrious FRAG_WHOLE flags
```

Crucial parameters for a connection are:

OPENVPN_x_TUN_MTU MTU Values of the TUN device were set to 1300 for OpenVPN 1.x. As of OpenVPN 2.0 1500 is the default here.

OPENVPN_x_LINK_MTU Byte size of the connection in both OpenVPN daemons. The default is depending on OpenVPN Version and operating system version.

OPENVPN_x_FRAGMENT Data packets (UDP or TCP) with a size bigger than the fragment size will be fragmented to packets not bigger than byte size provided in `OPENVPN_x_FRAGMENT`.

OPENVPN_x_MSSFIX To avoid fragmentation of data packets for TCP connections over VPN a maximum size for TCp data packets can be set here. Up-to-date operating systems will honorate this setting and make fragmentation unnecessary.

Different OpenVPN versions use the following settings as default values. Please obey these values when connecting OpenVPN in varying versions. Default settings on fli4l routers are shown in the second table.

OpenVPN Version/Option	1.xx	2.00
OPENVPN_x_TUN_MTU	1300	1500
OPENVPN_x_TUN_MTU_EXTRA	unknown	32
OPENVPN_x_FRAGMENT	unknown	not configured
OPENVPN_x_MSSFIX	not configured	1450

Table 1.2.: Different MTU parameters in different OpenVPN versions.

Based on this settings the defaults for your network should be determined and written to config/openvpn.txt explicitely. These are the best values for your tests to start with:

fli4l Version/Option	up to 2.1.8	from 2.1.9 on
OPENVPN_x_TUN_MTU	1300	1500
OPENVPN_x_TUN_MTU_EXTRA	64	32
OPENVPN_x_FRAGMENT	not configured	1300
OPENVPN_x_MSSFIX	not configured	1300

Table 1.3.: Different MTU parameters in different fli4l router versions.

```
OPENVPN_DEFAULT_TUN_MTU='1500'
OPENVPN_DEFAULT_MSSFIX='1300'
OPENVPN_DEFAULT_FRAGMENT='1300'
```

For fli4l versions prior to 2.1.9 „tun-mtu“ parameters can't be specified directly. But they can be influenced indirectly with `OPENVPN_x_LINK_MTU`. tun-mtu values are about 45 byte smaller than the values in `OPENVPN_x_LINK_MTU`. To get exact values only trying will help.

1.1.8. OpenVPN - Examples

Some examples will clarify the configuration of package OpenVPN.

Example - Joining Two Nets Using fli4l Routers

In the first example two fli4l routers will be connected. Nets behind each fli4l router should gain access to each other. Peter and Maria want to connect their nets over their fli4l routers. Peter uses a private net 192.168.145.0/24 and a DynDNS address 'peter.eisfair.net'. Marias setup is similar while she is using 10.23.17.0/24 and DynDNS address 'maria.eisfair.net'. Both trust in each other so they allow unlimited access to their complete nets for each other.

OpenVPN Option	Peter	Maria
OPENVPN_1_NAME=	'maria'	'peter'
OPENVPN_1_REMOTE_HOST=	'maria.eisfair.net'	'peter.eisfair.net'
OPENVPN_1_REMOTE_PORT=	'10000'	'10001'
OPENVPN_1_LOCAL_PORT=	'10001'	'10000'
OPENVPN_1_SECRET=	'pema.secret'	'pema.secret'
OPENVPN_1_TYPE=	'tunnel'	'tunnel'
OPENVPN_1_REMOTE_VPN_IP=	'192.168.200.202'	'192.168.200.193'
OPENVPN_1_LOCAL_VPN_IP=	'192.168.200.193'	'192.168.200.202'
OPENVPN_1_ROUTE_N=	'1'	'1'
OPENVPN_1_ROUTE_1=	'10.23.17.0/24'	'192.168.145.0/24'
OPENVPN_1_PF_INPUT_N=	'1'	'1'
OPENVPN_1_PF_INPUT_1=	'ACCEPT'	'ACCEPT'
OPENVPN_1_PF_FORWARD_N=	'1'	'1'
OPENVPN_1_PF_FORWARD_1=	'ACCEPT'	'ACCEPT'

Table 1.4.: OpenVPN Configuration with 2 fli4l routers

Example - Two Nets Connected By A Bridge

In the next example a bridge over a wi-fi connection will be configured. Packet filters are not of use here because usually ethernet frames will be forwarded but no IP packets. Please remember that with a bridge a common net is used. Thus no IP address can exist twice.

OpenVPN Option	Peter	Maria
OPENVPN_2_NAME	'bridge'	'bridge'
OPENVPN_2_REMOTE_HOST	'10.1.0.1'	'10.2.0.1'
OPENVPN_2_REMOTE_PORT	'10005'	'10006'
OPENVPN_2_LOCAL_HOST	'10.2.0.1'	'10.1.0.1'
OPENVPN_2_LOCAL_PORT	'10006'	'10005'
OPENVPN_2_FLOAT	'no'	'no'
OPENVPN_2_RESTART	'never'	'never'
OPENVPN_2_SECRET	'bridge.secret'	'bridge.secret'
OPENVPN_2_TYPE	'bridge'	'bridge'
OPENVPN_2_BRIDGE	'pema-br'	'pema-br'

Table 1.5.: OpenVPN Configuration with 2 fli4l routers bridging nets over a wi-fi connection.

In addition to the settings for OpenVPN a bridge has to be configured in advanced_networking and base.txt has to be adapted to use the bridge device and not eth0 as the network device for the internal net. See the relevant entries in advanced_networking's and base's configuration files:

advanced_networking Option	Peter	Maria
OPT_BRIDGE_DEV	'yes'	'yes'
BRIDGE_DEV_BOOTDELAY	'no'	'no'
BRIDGE_DEV_N	'1'	'1'
BRIDGE_DEV_1_NAME	'pema-br'	'pema-br'
BRIDGE_DEV_1_DEVNAME	'br0'	'br0'
BRIDGE_DEV_1_DEV_N	'1'	'1'
BRIDGE_DEV_1_DEV_1_DEV	'eth0'	'eth0'

Table 1.6.: OpenVPN Configuration with 2 fli4l routers bridging nets over a wi-fi connection. Bridge configuration in advanced_networking.

base Option	Peter	Maria
IP_NET_N	'1'	'1'
IP_NET_1	'192.168.193.254/24'	'192.168.193.1/24'
IP_NET_1_DEV	'br0'	'br0'

Table 1.7.: OpenVPN Configuration with 2 fli4l routers bridging nets over a wi-fi connection. Bridge configuration in base.txt.

Example - Configure Access For A Road Warrior

For this example (Roadwarrior) access to a LAN behind fli4l should be configured for a notebook with Windows XP over GPRS. OpenVPN is installed on the notebook and the *.ovpn file is edited. Unfortunately the tun/tap driver for Windows is not as flexible as its Unix pendant. Point-to-Point addresses for VPN IP have to be in a 255.255.255.252 (or /30) net. If the road warrior should only access services in the LAN behind or on the fli4l router itself and does not have to be accessed by itself a route on fli4l's side is not necessary. The road warrior can be addressed on its virtual IP address (`OPENVPN_3_REMOTE_VPN_IP`) if necessary. If the road warrior has a fixed IP address a host route could be added if needed. If the road warrior i.e. has fixed IP address 192.168.33.33 you could simply add the following to fli4l's openvpn.txt:

```
OPENVPN_3_ROUTE_N='1'
OPENVPN_3_ROUTE_1='192.168.33.33/32'
```

With the configuration of the packet filter shown here complete communication in both directions is allowed. Only the fli4l router is not directly accessible for the road warrior. That would be needed if the road warrior should use the DNS server on the fli4l router.

```
OPENVPN_3_PF_FORWARD_N='1'
OPENVPN_3_PF_FORWARD_1='ACCEPT'
```

For allowing access to fli4l's internal DNS server add the following to the configuration of fli4l:

```
OPENVPN_3_PF_INPUT_N='1'
OPENVPN_3_PF_INPUT_1='if:VPNDEV:any tmpl:dns ACCEPT'
```

OpenVPN Option fli4l router	roadwarrior
OPENVPN_3_NAME='roadwarrior'	remote peter.eisfair.net
OPENVPN_3_LOCAL_PORT='10011'	rport 10011
OPENVPN_3_SECRET='roadwarrior.secret'	secret roadwarrior.secret
OPENVPN_3_TYPE='tunnel'	dev tun
OPENVPN_3_REMOTE_VPN_IP='192.168.200.238'	
OPENVPN_3_LOCAL_VPN_IP='192.168.200.237'	ifconfig 192.168.200.238 192.168.200.237
OPENVPN_3_ROUTE_N='0'	
OPENVPN_3_PF_FORWARD_N='1'	
OPENVPN_3_PF_FORWARD_1='ACCEPT'	
	route 192.168.145.0 255.255.255.0
	comp-lzo
	persist-tun
	persist-key
	ping-timer-rem
	ping-restart 60
	proto udp
	tun-mtu 1500
	fragment 1300
	mssfix

Table 1.8.: OpenVPN Configuration with a Windows Computer over GPRS.

Example - Secure A WLAN Connection

In this example a WLAN connection will be secured by the help of OpenVPN. The fli4l router has a LAN and a WLAN card it uses or an access point is connected to an additional fli4l NIC. This aims at WLAN clients only having access to the VPN port without establishing a VPN connection. After connecting succesfully to OpenVPN they should have unlimited access with cable nets. DNSMASQ DHCP server's settings have to be changed to achieve that. Package advanced_networking will be needed as well. Settings in base.txt: `IP_NET_1` is the cable LAN and `IP_NET_2` is the WLAN.

```
IP_NET_N='2'
IP_NET_1='192.168.3.254/24'
IP_NET_1_DEV='br0'
IP_NET_2='192.168.4.254/24'
IP_NET_2_DEV='eth2'
```

Set DHCP range to suit your needs. For `IP_NET_2` two settings are mandatory:

```
DHCP_RANGE_2_DNS_SERVER1='none'
DHCP_RANGE_2_NTP_SERVER='none'
DHCP_RANGE_2_GATEWAY='none'
```

Settings in advanced_networking.txt:

```
OPT_BRIDGE_DEV='yes'
BRIDGE_DEV_BOOTDELAY='yes'
BRIDGE_DEV_N='1'
BRIDGE_DEV_1_NAME='br'
BRIDGE_DEV_1_DEVNAME='br0'
BRIDGE_DEV_1_DEV_N='1'
BRIDGE_DEV_1_DEV_1_DEV='eth0'
```

OpenVPN Option Router	WLAN-Client
OPENVPN_4_NAME='wlan1'	
OPENVPN_4_LOCAL_HOST='192.168.4.254'	remote 192.168.4.254
OPENVPN_4_LOCAL_PORT='20001'	rport 20001
OPENVPN_4_SECRET='wlan1.secret'	secret wlan1.secret
OPENVPN_4_TYPE='bridge'	dev tap
OPENVPN_4_BRIDGE='br'	
OPENVPN_4_RESTART='never'	
OPENVPN_4_MUTE_REPLAY_WARNINGS='yes'	
	comp-lzo
	persist-tun
	persist-key
	ping-timer-rem
	ping-restart 60
	proto udp
	tun-mtu 1500
	fragment 1300
	mssfix

Table 1.9.: OpenVPN Securing a WLAN.

1.1.9. Additional Links On OpenVPN

At the end here are some links on OpenVPN configuration:
http://openvpn.net
http://de.wikipedia.org/wiki/OpenVPN
http://openvpn.se/
http://arnowelzel.de/wiki/de/fli4l/openvpn
http://wiki.freifunk.net/OpenVPN
http://w3.linux-magazine.com/issue/24/Charly.pdf
http://w3.linux-magazine.com/issue/25/WirelessLAN_Intro.pdf
http://w3.linux-magazine.com/issue/25/OpenVPN.pdf

A. Appendix For Package OPENVPN

List of Figures

List of Tables

Index