

# **Paket ISDN**

## **Version 4.0.0-stable-x86-r60798**

Frank Meyer                      Das fli4l-Team  
E-Mail: [frank@fli4l.de](mailto:frank@fli4l.de)      E-Mail: [team@fli4l.de](mailto:team@fli4l.de)

19. Oktober 2022

# Inhaltsverzeichnis

|  |           |
|--|-----------|
| <b>1. Dokumentation des Paketes ISDN</b>                                   | <b>3</b>  |
| 1.1. ISDN - Kommunikation über aktive und passive ISDN-Karten . . . . .    | 3         |
| 1.1.1. Herstellen einer ISDN-Verbindung . . . . .                          | 3         |
| 1.1.2. ISDN-Karte . . . . .  | 4         |
| 1.1.3. OPT_ISDN_COMP (EXPERIMENTAL) . . . . .                              | 7         |
| 1.1.4. ISDN-Circuits . . . . .   | 7         |
| 1.1.5. OPT_TELMOND - telmond-Konfiguration . . . . .                       | 16        |
| 1.1.6. OPT_RCAPID - Remote CAPI Dämon . . . . .                            | 19        |
| <b>A. Anhang zum Paket ISDN</b>  | <b>21</b> |
| A.1. ISDN . . . . .  | 21        |
| A.1.1. Technische Details zu Einwahl und Routing bei ISDN . . . . .        | 21        |
| A.1.2. Fehlermeldungen des ISDN-Subsystems (englisch, i4l-Dokumentation) . | 22        |
| <b>Abbildungsverzeichnis</b>   | <b>25</b> |
| <b>Tabellenverzeichnis</b>   | <b>26</b> |
| <b>Index</b>   | <b>27</b> |

# 1. Dokumentation des Paketes ISDN

## 1.1. ISDN - Kommunikation über aktive und passive ISDN-Karten

fli4l ist vornehmlich zum Einsatz als ISDN- und/oder DSL-Router gedacht. Mit der Einstellung `OPT_ISDN='yes'` wird das ISDN-Paket aktiviert. Voraussetzung ist eine ISDN-Karte, die von fli4l unterstützt wird.

Soll kein ISDN verwendet werden, kann mit der Einstellung `OPT_ISDN='no'` die ISDN-Installation abgeschaltet werden. Dann werden alle in diesem Kapitel folgenden ISDN-Variablen ignoriert.

Standard-Einstellung: `OPT_ISDN='no'`

### 1.1.1. Herstellen einer ISDN-Verbindung

Das Einwählverhalten von fli4l wird von drei verschiedenen Variablen bestimmt, `DIALMODE`, `ISDN_CIRC_X_ROUTE_X`, `ISDN_CIRC_X_TIMES`. Es wird von `DIALMODE` (Seite ??) (in `<config>/base.txt`) bestimmt, ob bei Eintreffen eines Paketes auf einem aktiven Circuit automatisch eine Verbindung aufgebaut werden soll oder nicht. `DIALMODE` kann folgende Werte annehmen:

**auto** Trifft ein Paket auf einem ISDN-Circuit (bzw. dem daraus abgeleiteten ISDN-Interface `ippp*`) ein, wird automatisch eine Verbindung aufgebaut. Ob und wann ein Paket auf einem ISDN-Circuit eintrifft, wird von `ISDN_CIRC_X_ROUTE_X` und `ISDN_CIRC_X_TIMES` bestimmt.

**manual** Im manuellen Modus muß der Verbindungsaufbau über `imond/imonc` angestoßen werden. Wie das geht, steht im Abschnitt über `imonc/imond`.

**off** Es werden keine ISDN-Verbindungen hergestellt.

Auf welchem der konfigurierten Circuits Pakete eintreffen und damit eine Einwahl auslösen können, wird über `ISDN_CIRC_X_ROUTE_X` definiert. Standardmäßig ist es auf `'0.0.0.0/0'`, die sogenannte 'default route' gesetzt. Das heißt, dass alle Pakete, die das lokale Netz verlassen, über diesen Circuit gehen, wenn er aktiv ist. Ob und wann er aktiv ist, wird dabei von `ISDN_CIRC_X_TIMES` bestimmt, da fli4l über die definierten Circuits ein *least cost routing* durchführt (siehe Abschnitt Least-Cost-Routing - Funktionsweise (Seite ??) in der Dokumentation des Grundpaketes). Möchte man nicht alle Pakete über diesen Circuit leiten, sondern nur Pakete in ein bestimmtes Netz (z.B. Firmennetz), kann man hier ein oder mehrere Netze angeben. Dann richtet fli4l eine Route über das dem Circuit zugeordnete ISDN-Interface ein, die permanent aktiv ist. Wird nun ein Paket in dieses Netz geschickt, erfolgt automatisch der Verbindungsaufbau.

Wie schon erwähnt, beschreibt `ISDN_CIRC_X_TIMES` neben den Verbindungskosten eines Circuits auch, ob und wann ein Circuit mit einer 'default route' aktiv ist und damit einen Verbindungsaufbau auslösen kann. Das 'wann' spezifiziert man über die Zeit, die ersten beiden Elemente

einer time-info (z.B. Mo-Fr:09-18), das 'ob' durch den vierten Parameter lc-default-route (y/n). fli4l (bzw. imond) sorgt dann dafür, dass die Pakete, die das lokale Netz verlassen, immer über den zu diesem Zeitpunkt aktiven Circuit gehen und damit ein Herstellen der Verbindung zum Internet-Provider auslösen.

Zusammenfassend kann man also für die Standardanwendungsfälle folgendes sagen:

- Will man einfach nur ins Internet, stellt man DIALMODE auf auto, definiert 1-n Circuits, die als erste Route '0.0.0.0/0' haben und deren Zeiten (Zeiten mit lc-default-route = y) die gesamte Woche abdecken.

```
ISDN_CIRC_%_ROUTE_N='1'
ISDN_CIRC_%_ROUTE_1='0.0.0.0/0'
ISDN_CIRC_%_TIMES='Mo-Su:00-24:0.0148:Y'
```

- Will man nebenbei noch über eine spezielle Nummer ins Firmennetz, definiert man sich einen Circuit (oder mehrere Circuits) mit Route ungleich '0.0.0.0/0' und hat damit einen permanent aktiven Zugang zum Firmennetz.

```
ISDN_CIRC_%_ROUTE_N='1'
ISDN_CIRC_%_ROUTE_1='network/netmaskbits'
ISDN_CIRC_%_TIMES='Mo-Su:00-24:0.0148:Y'
```

### 1.1.2. ISDN-Karte

Der fli4l-Router unterstützt generell die gleichzeitige Verwendung mehrerer ISDN-Karten. Dies erfordert jedoch, dass alle ISDN-Karten *denselben Treibertyp* benötigen. Der Treibertyp leitet sich aus der Gruppe in der unten stehenden Tabelle ab, in welcher der betreffende Treiber zu finden ist. Es ist also beispielsweise kein Problem, mehrere mISDN-getriebene Adapter oder mehrere HiSax-getriebene Adapter zu verwenden (sofern genügend Ressourcen vorhanden sind), es ist aber nicht möglich, eine mISDN-getriebene und eine HiSax-getriebene Karte zur selben Zeit zu benutzen.

#### ISDN\_%\_TYPE ISDN\_%\_IO ISDN\_%\_IO0 ISDN\_%\_IO1 ISDN\_%\_MEM ISDN\_%\_IRQ

Hier sind die technischen Daten für die ISDN-Karte anzugeben.

Die im Beispiel aufgeführten Werte funktionieren für eine TELES 16.3, wenn die Karte auf IO-Adresse 0xd80 eingestellt ist (über Dip-Switches). Bei einer anderen Einstellung der Karte muss der Wert geändert werden.

#### Häufig gemachter Fehler (Beispiel):

```
ISDN_1_IO='240' -- richtig wäre: ISDN_1_IO='0x240'
```

Bei IRQ 12 muss man einen eventuell vorhandenen PS/2-Maus-Anschluss im BIOS abschalten. Sonst besser einen anderen IRQ wählen! „Gute“ sind meist 5, 10 und 11.

ISDN\_%\_TYPE entspricht prinzipiell den Typ-Nummern für den HiSax-Treiber. Ausnahme: nicht-HiSax-Karten wie z.B. AVM-B1. Für diese wurde der Nummernkreis für die Typen erweitert (siehe unten). Die Liste der möglichen HiSax-Typen basiert auf `linux-2.x.y/Documentation/isdn/README.HiSax`.

## 1. Dokumentation des Paketes ISDN

| Typ                                  | Karte   | Benötigte Parameter |
|--------------------------------------|---|---------------------|
| Dummy Typ-Nummer:                    |   |                     |
| 0                                    | no driver (dummy)   | none                |
| Typ-Nummern für Remote-CAPI-Treiber: |   |                     |
| 160                                  | AVM Fritz!Box Remote CAPI   | ip,port             |
| 161                                  | Melware Remote CAPI (rcapi)   | ip,port             |
| Typ-Nummern für mISDN-Treiber:       |   |                     |
| 301                                  | HFC-4S/8S/E1 multiport cards  | no parameter        |
| 302                                  | HFC-PCI based cards   | no parameter        |
| 303                                  | HFCS-USB Adapters   | no parameter        |
| 304                                  | AVM Fritz!Card PCI (v1 and v2) cards  | no parameter        |
| 305                                  | cards based on Infineon (former Siemens) chips:<br>- Dialogic Diva 2.0<br>- Dialogic Diva 2.0U<br>- Dialogic Diva 2.01<br>- Dialogic Diva 2.02<br>- Sedlbauer Speedwin<br>- HST Saphir3<br>- Develo (former ELSA) Microlink PCI (Quickstep 1000)<br>- Develo (former ELSA) Quickstep 3000<br>- Berkcom Scitel BRIX Quadro<br>- Dr.Neuhaus (Sagem) Niccy | no parameter        |
| 306                                  | NetJet TJ 300 and TJ320 cards   | no parameter        |
| 307                                  | Sedlbauer Speedfax+ cards   | no parameter        |
| 308                                  | Winbond 6692 based cards  | no parameter        |

Meine Karte ist eine Teles 16.3 NON-PNP ISA, also ist Type=3.

Für eine ICN-2B-Karte müssen IO und MEM gesetzt werden, zum Beispiel `ISDN_1_IO='0x320'`, `ISDN_1_MEM='0xd0000'`.

Bei neueren Teles-PCI-Karten muss type=20 (statt 21) verwendet werden. Die Dinger melden sich bei "cat /proc/pci" mit "tiger" oder so ähnlich. Sonst kann ich nichts zu diesen Werten beitragen, sorry.

Um die ISDN-Typen 105 bis 114 verwenden zu können, ist es vorher nötig, die passenden Treiberdateien von <http://www.fli4l.de/download/stabile-version/avm-treiber/> herunterzuladen und in das fli4l-Verzeichnis zu entpacken. Da diese Treiber nicht der GPL unterliegen, können sie leider nicht mit dem ISDN Paket mitgeliefert werden.

Für den ISDN-Typ 303 ist es nötig USB Support zu installieren und zu aktivieren. Siehe dazu USB - Support für USB-Geräte (Seite ??).

Tips zu den Typ-Nummern bekommt man auch über die i4l-FAQ oder Mailingliste, wenn man wirklich nicht weiß, was da für eine Karte im PC steckt.

Die Kartentypen, die mit „from isapnp setup“ gekennzeichnet sind, müssen mit dem PnP-Tool isapnp initialisiert werden - wenn es sich tatsächlich um eine PnP-Karte han-

delt. Siehe dazu die Beschreibung im Kapitel OPT\_PNP - Installation von isapnp tools (Seite ??).

Der ISDN-Typ 0 wird dann benötigt, wenn man das ISDN-Paket installieren will ohne ISDN-Karte; z.B. um imond verwenden zu können bei einem Netzwerkrouter.

**ISDN\_%\_IP ISDN\_%\_PORT** Für die ISDN-Typen 160 und 161 werden über diese Variablen die IP-Adresse (ISDN\_%\_IP) bzw. der Port (ISDN\_%\_PORT) des Gerätes eingestellt, das eine entfernte CAPI-Schnittstelle anbietet. Die IP-Adresse ist zwingend erforderlich, die Port-Nummer kann hingegen weggelassen werden: Je nach gewähltem Typ wird dann ein Standard-Port eingestellt (Typ 160: 5031, Typ 161: 2662).

Beispiel:

```
ISDN_1_TYPE='160' # AVM Fritz!Box
ISDN_1_IP='192.168.177.1'
```

**ISDN\_DEBUG\_LEVEL** Dieses gibt den Debug-Level für den HiSaX-Treiber an. Der Debug-Level setzt sich dabei durch Addition der folgenden Werte zusammen (Zitat aus der Original-Doku):

| Number | Debug-Information                           |
|--------|---|
| 1      | Link-level <-> hardware-level communication |
| 2      | Top state machine                           |
| 4      | D-Channel Q.931 (call control messages)     |
| 8      | D-Channel Q.921                             |
| 16     | B-Channel X.75                              |
| 32     | D-Channel l2                                |
| 64     | B-Channel l2                                |
| 128    | D-Channel link state debugging              |
| 256    | B-Channel link state debugging              |
| 512    | TEI debug                                   |
| 1024   | LOCK debug in callc.c                       |
| 2048   | More debug in callc.c (not for normal use)  |

Die Standardeinstellung (ISDN\_DEBUG\_LEVEL='31') sollte den meisten reichen.

**ISDN\_VERBOSE\_LEVEL** Hiermit kann man die "Geschwätzigkeit" des ISDN-Subsystems im fli4l-Kernel einstellen. Jeder Verbose-Level schließt die Level mit niedrigerer Nummer mit ein. Die Verbose-Level sind:

|              |   |
|--------------|---|
| '0'          | keine zusätzlichen Informationen                              |
| '1'          | Es wird protokolliert, was eine ISDN-Verbindung ausgelöst hat |
| '2' und '3'  | Anrufe werden protokolliert                                   |
| '4' und mehr | Die Datentransferrate wird regelmäßig protokolliert.          |

Die Meldungen werden über das Kernel-Logging-Interface ausgegeben, erscheinen also bei aktiviertem OPT\_SYSLOGD (Seite ??) dort.

**Wichtig:** Sollen Anrufe mit telmond protokolliert werden, diesen Wert nicht kleiner als 2 einstellen, da telmond sonst die Informationen fehlen, um Anrufe zu protokollieren.

Standardeinstellung: `ISDN_VERBOSE_LEVEL='2'`

**ISDN\_FILTER** Aktiviert den Filtermechanismus des Kerns, um ein ordnungsgemäßes Auflegen nach dem angegebenen Hangup-Timeout zu gewährleisten. Siehe <http://www.fli4l.de/hilfe/howtos/basteleien/hangup-problem-loesen/> für genauere Informationen.

**ISDN\_FILTER\_EXPR** Hier steht der zu nutzende Filter, wenn `ISDN_FILTER` auf 'yes' gesetzt ist.

### 1.1.3. OPT\_ISDN\_COMP (EXPERIMENTAL)

Mit `OPT_ISDN_COMP='yes'` werden die LZS- und die BSD-Kompression aktiviert. Das Kompressionspaket hat freundlicherweise Arwin Vosselman (E-Mail: [arwin\(at\)xs4all\(dot\)nl](mailto:arwin(at)xs4all(dot)nl)) zusammengestellt. Dieses Zusatzpaket hat Experimental-Status.

Standard-Einstellung: `OPT_ISDN_COMP='no'`

Hier im Einzelnen die erforderlichen Parameter für LZS-Kompression:

**ISDN\_LZS\_DEBUG (EXPERIMENTAL)** Debug-Level-Einstellung:

- '0' keine Debugging Information
- '1' normale Debugging Information
- '2' erweiterte Debugging Information
- '3' schwere Debugging Information (inkl. Dumping der Datenpakete)

Standard-Einstellung: `ISDN_LZS_DEBUG='1'`

Wer bei Problemen mit der Komprimierung noch mehr Debugmeldungen sehen möchte, setzt diese Variable auf '2'.

**ISDN\_LZS\_COMP (EXPERIMENTAL)** Stärke der Kompression (nicht Dekompression!). Bitte erst einmal auf dem Wert '8' stehen lassen. Werte von 0 bis 9 sind möglich.

Grössere Zahlen geben bessere Kompression, 9 erzeugt aber übermässige CPU-Last.

Standard-Einstellung: `ISDN_LZS_COMP='8'`

**ISDN\_LZS\_TWEAK (EXPERIMENTAL)** Auch diese Variable erst einmal auf '7' stehen lassen.

Standard-Einstellung: `ISDN_LZS_TWEAK='7'`

Außer diesen 3 Werten muss noch die Variable `ISDN_CIRC_x_FRAMECOMP` angepasst werden, s. nächstes Kapitel.

### 1.1.4. ISDN-Circuits

In der fli4l-Konfiguration können mehrere Verbindungen über ISDN definiert werden. Davon sind maximal 2 Verbindungen auch zur gleichen Zeit über eine ISDN-Karte möglich.

Die Definition solcher Verbindungen geschieht über sogenannte Circuits. Dabei wird pro Verbindung ein Circuit verwendet.

In der Beispiel-Datei `config.txt` sind zwei solcher Circuits definiert:

## 1. Dokumentation des Paketes ISDN

- Circuit 1: Dialout über Internet-By-Call-Provider Microsoft Network, Sync-PPP
- Circuit 2: Dialin/Dialout zu einem ISDN-Router (das könnte beispielsweise auch fli4l sein) über Raw-IP, z.B. als Zugang zum Firmen-Netz von irgendwo. Bei mir konkret ist das eine Linux-Box mit isdn4linux als "Gegner".

Soll der fli4l-Router lediglich als Internet-Gateway dienen, ist nur ein Circuit notwendig. Ausnahme: Man will die Least-Cost-Router-Features von fli4l nutzen. Dann sind sämtliche erlaubten Circuits für verschiedene Zeitbereiche zu definieren, siehe unten.

**ISDN\_CIRC\_N** Gibt die Anzahl der verwendeten ISDN-Circuits an. Wird fli4l lediglich als ISDN-Anrufmonitor eingesetzt, ist einzustellen:

```
ISDN_CIRC_N='0'
```

Soll der fli4l-Router lediglich als einfaches ISDN-Gateway in das Internet verwendet werden, reicht ein Circuit. Ausnahme: LC-Routing, siehe unten.

**ISDN\_CIRC\_x\_NAME** Hier sollte ein Name für den Circuit vergeben werden - max. 15 Stellen lang. Dieser wird dann im imon-Client `imonc.exe` statt der angewählten Telefonnummer gezeigt. Erlaubte Zeichen sind die Buchstaben 'A' bis 'Z' (Klein- und Großschreibung), die Zahlen '0' bis '9' und der Bindestrich '-', wie z.B.

```
ISDN_CIRC_x_NAME='msn'
```

Der Name kann außerdem im Paketfilter oder bei OpenVPN benutzt werden. Wenn z.B. der Paketfilter einen ISDN Circuit regeln soll, muß ein 'circuit\_' dem Circuit Namen vorangesetzt werden. Heißt ein ISDN Circuit z.B. 'willi', so wird daraus folgendes im Paketfilter:

```
PF_INPUT_3='if:circuit_willi:any prot:udp 192.168.200.226 192.168.200.254:53 ACCEPT'
```

**ISDN\_CIRC\_x\_USEPEERDNS** Hiermit wird festgelegt, ob die vom Internet-Provider bei der Einwahl übergebenen Nameserver für die Dauer der Onlineverbindung in die Konfigurationsdatei des lokalen Nameservers eingetragen werden sollen. Sinnvoll ist die Nutzung dieser Option also nur bei Circuits für Internet-Provider. Inzwischen unterstützen fast alle Provider diese Art der Übergabe.

Nachdem die Nameserver-IP-Adressen übertragen wurden, werden die in der `base.txt` unter *DNS\_FORWARDERS* eingetragenen Nameserver aus der Konfigurationsdatei des lokalen Nameservers entfernt und die vom Provider vergebenen IP-Adressen als Forwarder eingetragen. Danach wird der lokale Nameserver veranlaßt, seine Konfiguration neu einzulesen. Dabei gehen bis dahin aufgelöste Namen nicht aus dem Nameserver-Cache verloren.

Diese Option bietet den Vorteil, immer mit den am nächsten liegenden Nameservern arbeiten zu können, sofern der Provider die korrekten IP-Adressen übermittelt - dadurch geht die Namensauflösung schneller.



Im Falle eines Ausfalls eines DNS-Servers beim Provider werden in der Regel die übergebenen DNS-Server-Adressen sehr schnell vom Provider korrigiert.

Trotz allem ist vor jeder ersten Einwahl die Angabe eines gültigen Nameservers in *DNS\_FORWARDERS* der base.txt zwingend erforderlich, da sonst die erste Anfrage nicht korrekt aufgelöst werden kann. Außerdem wird beim Beenden der Verbindung die originale Konfiguration des lokalen Nameservers wieder hergestellt.

Standard-Einstellung: `ISDN_CIRC_x_USEPEERDNS='yes'`

**ISDN\_CIRC\_x\_TYPE** `ISDN_CIRC_x_TYPE` gibt den Typ der x-ten IP-Verbindung an. Dabei sind folgende Werte möglich:

'raw' RAW-IP  
'ppp' Sync-PPP

In den meisten Fällen wird PPP verwendet, jedoch ist Raw-IP etwas effizienter, da hier der PPP-Overhead entfällt. Eine Authentifizierung ist zwar bei Raw-IP nicht möglich, es kann jedoch über die Variable `ISDN_CIRC_x_DIALIN` (s.u.) eine Zugangsbeschränkung auf ganz bestimmte ISDN-Nummern (Stichwort "Clip") eingestellt werden. Wird `ISDN_CIRC_x_TYPE` auf 'raw' gestellt wird analog zu den PPP up/down Scripten in `/etc/ppp` ein raw up/down Script ausgeführt.

**ISDN\_CIRC\_x\_BUNDLING** Für die ISDN-Kanalbündelung wird das verbreitete MPPP-Protokoll nach RFC 1717 verwendet. Damit gelten folgende Einschränkungen, die aber in der Praxis meist nicht relevant sind:

- Nur mit PPP-Verbindungen möglich, nicht bei Raw-Circuits
- Kanalbündelung nach neuerer RFC 1990 (MLP) ist nicht möglich

Der 2. Kanal kann entweder mit dem Client `imonc` manuell hinzugeschaltet werden oder über die Bandbreitenanpassung automatisch aktiviert werden, siehe auch die Beschreibung zu `ISDN_CIRC_x_BANDWIDTH`.

Standard-Einstellung: `ISDN_CIRC_x_BUNDLING='no'`

Vorsicht: bei Verwendung von Kanalbündelung zusammen mit Kompression kann es zu Problemen kommen, siehe auch die Beschreibung zu `ISDN_CIRC_x_FRAMECOMP`.

**ISDN\_CIRC\_x\_BANDWIDTH** Ist die ISDN-Kanalbündelung über `ISDN_CIRC_x_BUNDLING='yes'` aktiviert, kann mit dieser Variablen eine automatische Hinzuschaltung des 2. ISDN-Kanals eingestellt werden. Dabei sind 2 numerische Parameter anzugeben:

1. Schwellenwert in Byte/Sekunde (S)
2. Zeitintervall in Sekunden (Z)

Wird der Schwellenwert S für Z Sekunden überschritten, schaltet der Steuerprozeß `imond` den 2. Kanal automatisch hinzu. Wird der Schwellenwert S für Z Sekunden unterschritten, wird der 2. Kanal automatisch wieder deaktiviert. Die automatische Bandbreitenanpassung kann mit `ISDN_CIRC_1_BANDWIDTH=""` abgeschaltet werden. Dann ist lediglich eine manuelle Kanalbündelung über den Client `imonc` möglich.

Beispiele:

- `ISDN_CIRC_1_BANDWIDTH='6144 30'`

Überschreitet die Transferrate den Wert von 6 kibibyte/second für 30 Sekunden, wird der 2. Kanal zugeschaltet.

- `ISDN_CIRC_1_BANDWIDTH='0 0'`

Der zweite ISDN-Kanal wird sofort, spätestens 10 Sekunden nach dem Verbindungsaufbau zugeschaltet und bleibt solange bestehen, bis die komplette Verbindung beendet wird.

- `ISDN_CIRC_1_BANDWIDTH=""`

Der zweite ISDN-Kanal kann nur manuell zugeschaltet werden, Voraussetzung ist jedoch weiterhin, dass `ISDN_CIRC_1_BUNDLING='yes'` eingestellt ist.

- `ISDN_CIRC_1_BANDWIDTH='10000 30'`

Eigentlich sollte hiermit der 2. Kanal nach 30 Sekunden zugeschaltet werden, wenn während dieser Zeitspanne 10000 B/s erreicht werden. Dazu wird es aber nicht kommen, da mit ISDN nicht mehr als 8 kB/s erreicht werden können.

Ist `ISDN_CIRC_x_BUNDLING='no'` eingestellt, ist der Wert in der Variablen `ISDN_CIRC_x_BANDWIDTH` belanglos.

Standard-Einstellung: `ISDN_CIRC_x_BANDWIDTH=""`

**ISDN\_CIRC\_x\_LOCAL** In dieser Variablen wird die lokale IP-Adresse auf der ISDN-Seite hinterlegt.

Bei dynamischer Adresszuweisung sollte dieser Wert **leer** sein. Beim Verbindungsaufbau wird dann die IP-Adresse ausgehandelt. In den meisten Fällen vergeben Internet-Provider diese Adresse dynamisch. Soll jedoch eine fest vergebene IP-Adresse verwendet werden, ist diese hier einzutragen. Diese Variable ist optional und muss bei Bedarf in das Konfigfile eingetragen werden.

**ISDN\_CIRC\_x\_REMOTE** In dieser Variablen wird die entfernte IP-Adresse und die zugehörige Netzmaske auf der ISDN-Seite hinterlegt. Dazu muss die CIDR (Classless Inter-Domain Routing) Schreibweise benutzt werden. Details zu CIDR (Seite ??) ist in der Dokumentation des Baseispaketes bei `IP_NET_x` zu finden.

Bei dynamischer Adresszuweisung sollte dieser Wert **leer** sein. Beim Verbindungsaufbau wird dann die IP-Adresse ausgehandelt. In den meisten Fällen vergeben Internet-Provider diese Adresse dynamisch. Soll jedoch eine fest vergebene IP-Adresse verwendet werden, ist diese hier einzutragen. Diese Variable ist optional und muss bei Bedarf in das Konfigfile eingetragen werden.

Die angegebene Netzmaske wird bei der Configuration des Interfaces nach der Einwahl verwendet. Während dieser Configuration wird auch eine Route zum Host erzeugt, in den man sich einwählt. Da man diese Route in der Regel nicht braucht, ist es günstig, hier nur eine Route direkt zum Einwahlrechner zu erzeugen. Dazu setzt man die Netzmaske auf /32, indem man hier 32 als Anzahl der gesetzten Bits in der Netzmaske spezifiziert. Für Details siehe [Kapitel: Technische Details zum Dialin](#) (Seite 21).

**ISDN\_CIRC\_x\_MTU** **ISDN\_CIRC\_x\_MRU** Mit diesen optionalen Variablen können die sog. **MTU** (maximum transmission unit) und die **MRU** (maximum receive unit) eingestellt werden. Optional bedeutet, die Variable muß nicht in der Konfigurationsdatei

stehen, sie ist bei Bedarf durch den Benutzer einzufügen!

Normal beträgt die MTU 1500 und die MRU 1524. Diese Einstellung sollte nur in Sonderfällen geändert werden!

**ISDN\_CIRC\_x\_CLAMP\_MSS** Hier sollte man ein yes setzen, wenn man synchrones ppp verwendet (ISDN\_CIRC\_x\_TYPE='ppp') und eines der folgenden Symptome auftritt:

- der Webbrowser verbindet sich mit dem Webserver, aber es wird keine Seite angezeigt und kommt auch keine Fehlermeldung; es passiert einfach nichts mehr,
- das Versenden kleiner E-Mails funktioniert, bei großen gibt es Probleme oder
- ssh funktioniert, scp hängt nach dem initialen Verbindungsaufbau.

Provider, bei denen solche Probleme auftreten, sind z.B. Compuserve und andere Media-ways basierte Zugänge.

Standard-Einstellung: ISDN\_CIRC\_x\_CLAMP\_MSS='no'

**ISDN\_CIRC\_x\_HEADERCOMP** Mit ISDN\_CIRC\_x\_HEADERCOMP='yes' kann die Van-Jacobson-Komprimierung oder Headerkomprimierung eingestellt werden. Nicht alle Provider unterstützen das. Sollte es daher bei eingeschalteter Komprimierung zu Einwahlproblemen kommen, sollte ISDN\_CIRC\_x\_HEADERCOMP='no' eingestellt werden.

Standard-Einstellung: ISDN\_CIRC\_x\_HEADERCOMP='yes'

**ISDN\_CIRC\_x\_FRAMECOMP (EXPERIMENTAL)** Dieser Parameter wird nur berücksichtigt, wenn OPT\_ISDN\_COMP='yes' eingestellt wird. Er regelt die Frame-Komprimierung.

Folgende Werte sind möglich:

|             |  |
|-------------|--|
| 'no'        | Keine Frame-Komprimierung  |
| 'default'   | LZS according RFC1974(std) and BSDCOMP 12                            |
| 'all'       | Negotiate lzs and bsdcomp  |
| 'lzs'       | Negotiate lzs only   |
| 'lzsstd'    | LZS according RFC1974 Standard Mode ("Sequential Mode")              |
| 'lzsext'    | LZS according RFC1974 Extended Mode                                  |
| 'bsdcomp'   | Negotiate bsdcomp only   |
| 'lzsstd-mh' | LZS Multihistory according RFC1974 Standard Mode ("Sequential Mode") |

Welcher Wert für den jeweiligen Provider verwendet werden kann, muss ausprobiert werden. So weit bekannt geht bei T-Online nur 'lzsext'. Bei den meisten anderen Providern sollte man mit 'default' auskommen.

Vorsicht: Bei verwendung von Kanalbündelung in zusammenhang mit 'lzsext' kann es zu Problemen kommen. Diese Probleme sind, so weit bekannt, Einwahlserverpezifisch und damit meistens Providerspezifisch. Es können aber bei einem Provider auch verschiedene Typen Einwahlserver im Einsatz sein, es kann in dem Fall zu Unterschieden zwischen Einwahlknoten kommen.

'lzsstd-mh' ist für Router-zu-Routerbetrieb (r2r) gedacht. Das Verfahren wird von Providern nicht eingesetzt aber bringt bei Verwendung zwischen zwei fl4l-Router erhebliche Verbesserung bei gleichzeitigen Übertragung von mehreren Dateien. Die Headerkompression ist dazu erforderlich und wird deshalb automatisch eingeschaltet.

**ISDN\_CIRC\_x\_REMOTENAME** Diese Variable ist normalerweise lediglich bei der Konfiguration von fli4l als Einwahlrouter relevant. Hier kann ein Name des Remote-Hosts eingetragen werden, muß aber nicht.

Standard-Einstellung: `ISDN_CIRC_x_REMOTENAME=""`

**ISDN\_CIRC\_x\_PASS** Hier sind die Provider-Daten einzutragen. Im Beispiel oben handelt es sich um die Daten des Providers Microsoft Network.

`ISDN_CIRC_x_USER` enthält die Benutzerkennung, `ISDN_CIRC_x_PASS` das Password.

WICHTIG: Für einen T-Online-Zugang ist folgendes zu beachten:

Der Username `AAAAAAAAAAAAATTTTTT#MMMM` setzt sich aus der zwölfstelligen Anschlußkennung, der T-Online-Nummer und der Mitbenutzernummer zusammen. Hinter der T-Online-Nummer muß ein '#' angegeben werden, wenn die Länge der T-Online-Nummer kürzer als 12 Zeichen ist.

Sollte dies in Einzelfällen nicht zum Erfolg führen (offenbar abhängig von der Vermittlungsstelle), muß zusätzlich zwischen der Anschlußkennung und der T-Online-Nummer ein weiteres '#'-Zeichen eingefügt werden.

Ansonsten (T-Online-Nr ist 12stellig) sind keine '#'-Zeichen anzugeben.

Beispiel: `ISDN_CIRC_1_USER='123456#123'`

Bei Raw-IP-Circuits haben diese Variablen keine Bedeutung.

**ISDN\_CIRC\_x\_ROUTE\_N** Die Anzahl der Routen die dieser ISDN Circuit bedient. Wenn dieser Circuit eine Default-Route definiert muss der Eintrag auf '1' gesetzt werden.

**ISDN\_CIRC\_x\_ROUTE\_X** Die Route oder die Routen für diesen Circuit. Für einen Internet-Zugang sollte man hier im ersten Eintrag '0.0.0.0/0' (default route) angeben. Das Format ist immer 'network/netmaskbits'. Eine Hostroute würde z.B. so aussehen: '192.168.199.1/32'. Bei Einwahl in den Firmen- oder Uni-Router ist lediglich das oder die Netze anzugeben, die man dort erreichen will, z.B.

```
ISDN_CIRC_%_ROUTE_N='2'
ISDN_CIRC_%_ROUTE_1='192.168.8.0/24'
ISDN_CIRC_%_ROUTE_2='192.168.9.0/24'
```

Bei mehreren Netzen müssen diese jeweils in einen eigenen Eintrag, also für jede Route muss eine `ISDN_CIRC_x_ROUTE_y=""` Zeile angelegt werden.

Möchte man die LC-Routing-Features von fli4l nutzen, kann \*mehreren\* Circuits eine Default-Route zugewiesen werden. Welcher Circuit dann tatsächlich verwendet wird, wird über `ISDN_CIRC_x_TIMES` eingestellt, siehe unten.

**ISDN\_CIRC\_x\_DIALOUT** `ISDN_CIRC_x_DIALOUT` gibt die zu wählende Telefonnummer an. Es ist möglich, hier mehrere Nummern anzugeben (falls eine besetzt ist, wird die nächste angewählt) - die Trennung erfolgt dabei durch Leerzeichen. Laut Berichten in der Newsgroup dürfen maximal fünf Nummern angegeben werden.

**ISDN\_CIRC\_x\_DIALIN** Soll der Circuit (auch) zum Einwählen genutzt werden, wird in ISDN\_CIRC\_x\_DIALIN die Rufnummer des Anrufenden eingesetzt - und zwar mit Vorwahl, aber *\*ohne\** die erste 0. Bei Anschlüssen hinter Telefonanlagen kann dies anders sein, eventuell müssen dann eine oder sogar zwei führende Nullen angegeben werden.

Soll es mehreren Teilnehmern ermöglicht werden, sich über diesen Circuit einzuwählen, können diese Nummern durch Leerzeichen getrennt angegeben werden. Besser ist es aber, jedem Gegner einen extra Circuit zuzuweisen. Sonst könnte es bei Einwahl von zwei Gegnern zu gleicher Zeit (ist über die 2 ISDN-Kanäle durchaus möglich) auf demselben Circuit zu Kollisionen bzgl. IP-Adressen kommen.

Falls der Anrufende keine Rufnummer überträgt, hier eine '0' setzen. Aber Vorsicht: damit wird jedem eine Anwahl gestattet, der keine Rufnummer überträgt!

Möchte man eine Einwahl unabhängig von der MSN des Anrufenden realisieren, ist als Wert '\*' anzugeben.

In den beiden letzten Fällen ist ein Authentifizierungsverfahren (siehe ISDN\_CIRC\_x\_AUTH) unumgänglich.

**ISDN\_CIRC\_x\_CALLBACK** Einstellung Callbackverfahren, mögliche Werte:

|         |   |
|---------|---|
| 'in'    | fli4l wird angerufen und ruft zurück                          |
| 'out'   | fli4l ruft an, hängt jedoch wieder ein und wartet auf Rückruf |
| 'off'   | kein Callback   |
| 'cbcp'  | CallBack Control Protocol                                     |
| 'cbcp0' | CallBack Control Protocol 0                                   |
| 'cbcp3' | CallBack Control Protocol 3                                   |
| 'cbcp6' | CallBack Control Protocol 6                                   |

Bei den CallBack Control Protokolle (auch 'Microsoft CallBack' genannt) ist cbcp6 das meist übliche Protokoll.

Standard-Wert: 'off'

**ISDN\_CIRC\_x\_CBNUMBER** Hier kann man für die Protokolle cbcp, cbcp3 und cbcp6 eine Rückrufnummer einsetzen (bei cbcp3 Pflicht).

**ISDN\_CIRC\_x\_CBDELAY** Diese Variable gibt eine Verzögerung in Sekunden an, die bei Callback gewartet werden soll. Je nachdem, in welcher Richtung der Callback erfolgen soll, hat diese Variable eine etwas andere Bedeutung:

- ISDN\_CIRC\_x\_CALLBACK='in':

Wird fli4l angerufen und soll zurückrufen, ist ISDN\_CIRC\_x\_CBDELAY die Wartezeit, bis der Rückruf erfolgen soll. Ein guter Erfahrungswert ist ISDN\_CIRC\_x\_CBDELAY='3'. Je nach "Gegner" kann aber auch ein geringerer Wert funktionieren, welches dann den Verbindungsaufbau beschleunigen kann.

- ISDN\_CIRC\_x\_CALLBACK='out':

In diesem Fall gibt ISDN\_CIRC\_x\_CBDELAY die Zeit an, wie lange das "Anklingeln des Gegners" erfolgen soll, bis auf den Rückruf gewartet wird. Auch hier ist ISDN\_CIRC\_x\_CBDELAY='3' ein guter Erfahrungswert. Was mir dazu aufgefallen ist: Bei Fernverbindungen muß man bis zu 3 Sekunden "klingeln" lassen, bevor der andere

Router überhaupt den Anruf sieht. Bei Ortsverbindungen kann meist dieser Wert kleiner gewählt werden. Im Zweifel: Ausprobieren!

Ist die Variable `ISDN_CIRC_x_CALLBACK='off'` eingestellt, wird `ISDN_CIRC_x_CBDELAY` ignoriert. Auch beim CallBack Control Protocol hat diese Variable keine Bedeutung.

**ISDN\_CIRC\_x\_EAZ** Im Beispiel ist die MSN (hier EAZ genannt) auf 81330 gesetzt. Hier sollte die eigene MSN \*ohne\* Vorwahl eingetragen werden.

Bei Anschlüssen hinter einer Telefonanlage mit Anlagenanschluss ist meistens nur die Durchwahl anzugeben. Ich habe aber auch schon gelesen, dass eine '0' weiterhelfen kann, wenn es Probleme mit der verwendeten Telefonanlage geben sollte.

Dieses Feld kann auch leer sein. Dies soll bei besonders hartnäckigen TK-Anlagen helfen. Um Feedback wird an dieser Stelle gebeten.

**ISDN\_CIRC\_x\_SLAVE\_EAZ** Ist der fli4l-Router am internen S0-Bus einer Telefonanlage angeschlossen und möchte man Kanalbündelung verwenden, ist bei manchen Telefonanlagen die Angabe einer 2. Durchwahlnummer für den Slave-Kanal einzutragen.

Im Normalfall kann diese Variable jedoch leer bleiben.

**ISDN\_CIRC\_x\_DEBUG** Soll ipppd zusätzliche Debug-Informationen ausgeben, muss man `ISDN_CIRC_x_DEBUG` auf 'yes' setzen. In diesem Fall schreibt ipppd zusätzlichen Informationen über die syslog-Schnittstelle.

WICHTIG: Damit diese auch über syslogd ausgegeben werden, muss die Variable `OPT_SYSLOGD` (Siehe `OPT_SYSLOGD` - Programm zum Protokollieren von Systemfehlermeldungen (Seite ??)) ebenso auf 'yes' gesetzt sein.

Weil manche Meldungen über klog ausgegeben werden sollte man beim Debuggen von ISDN auch `OPT_KLOGD` (Siehe `OPT_KLOGD` - Kernel-Message-Logger (Seite ??)) auf 'yes' setzen.

Bei Raw-IP-Circuits hat `ISDN_CIRC_x_DEBUG` keine Bedeutung.

**ISDN\_CIRC\_x\_AUTH** Wird dieser Circuit auch zum Einwählen verwendet und soll eine Authentifizierung über PAP oder CHAP vom einwählenden "Gegner" gefordert werden, ist `ISDN_CIRC_x_AUTH` auf 'pap' oder 'chap' zu setzen - und \*nur\* dann. Anderenfalls immer leer lassen!

Grund: Ein angewählter Internet-Provider wird es immer ablehnen, sich selbst auszuweisen! Ausnahmen bestätigen die Regel, wie ich erst kürzlich in der i4l-Mailingliste las ...

Als Benutzername und Passwort werden die Einträge von `ISDN_CIRC_x_USER` und `ISDN_CIRC_x_PASS` benutzt.

Bei Raw-IP-Circuits hat diese Variable keine Bedeutung.

**ISDN\_CIRC\_x\_HUP\_TIMEOUT** Mit der Variablen `ISDN_CIRC_x_HUP_TIMEOUT` wird die Zeit gesteuert, nach der der fli4l-Rechner die Verbindung zum Provider beenden soll, wenn nichts mehr über die Leitung geht. Im Beispiel wird die Verbindung nach 40 Sekunden Idle-Time abgebaut, um Geld zu sparen. Bei einem erneuten Zugriff in's Netz wird die

Verbindung in Sekundenschnelle wieder aufgebaut. Sinnvoll bei Providern, die sekunden-genau abrechnen!

Man sollte zumindest in der Testphase das automatische Wählen/Einhängen des fli4l-Routers beobachten (entweder auf der Console oder im imon-Client für Windows). Nicht, dass durch eine fehlerhafte Konfiguration der ISDN-Anschluss zur Standleitung wird.

Wird der Wert auf '0' gestellt, wird keine Idle-Zeit mehr berücksichtigt, d.h. fli4l hängt von sich aus die Leitung nicht mehr ein. Bitte mit Vorsicht anwenden.

**ISDN\_CIRC\_x\_CHARGEINT** Charge-Interval: Hier ist der Zeittakt in Sekunden anzugeben. Dieser wird dann für die Kosten-Berechnung verwendet.

Die meisten Provider rechnen minutengenau ab. In diesem Fall ist der Wert '60' richtig. Compuserve verwendet einen 3-Minuten-Takt (Stand Juni 2000), also `ISDN_CIRC_x_CHARGEINT='180'`. Bei Providern mit sekundengenaue Abrechnung (z.B. Planet-Interkom) setzt man besser `ISDN_CIRC_x_CHARGEINT` auf '1'.

Erweiterung für `ISDN_CIRC_x_CHARGEINT >= 60` Sekunden:

Wurde `ISDN_CIRC_x_HUP_TIMEOUT` Sekunden lang kein Traffic bemerkt, wird ca. 2 Sekunden vor Ablauf des Taktes eingehängt. Die vom Provider berechnete Zeit wird also fast komplett ausgenutzt. Ein wirklich tolles Feature von isdn4linux!

Bei sekundengenau abgerechneten Verbindungen hat das natürlich keinen Sinn - daher gilt diese Regelung erst ab Zeittakten von 60 Sekunden.

**ISDN\_CIRC\_x\_TIMES** Hier werden die Zeiten angegeben, wann dieser Circuit aktiviert werden soll und wann er wieviel kostet. Dadurch wird es möglich, zu verschiedenen Zeiten verschiedene Circuits mit Default-Routen zu verwenden (Least-Cost-Routing). Dabei kontrolliert der Daemon imond die Routen-Zuweisung.

Aufbau der Variablen:

```
ISDN_CIRC_x_TIMES='times-1-info [times-2-info] ...'
```

Jedes Feld times-?-info besteht aus 4 Unterfeldern - durch Doppelpunkt (':') getrennt.

1. Feld: W1-W2

Wochentag-Zeitraum, z.B. Mo-Fr oder Sa-Su usw. Schreibweise in Englisch oder deutsch möglich. Soll ein einzelner Wochentag eingetragen werden, ist zu W1-W1 schreiben, also z.B. Su-Su.

2. Feld: hh-hh

Stunden-Bereich, z.B. 09-18 oder auch 18-09. 18-09 ist gleichbedeutend mit 18-24 plus 00-09. 00-24 meint den ganzen Tag.

3. Feld: Charge

Hier werden in Euro-Werten die Kosten pro Minute angegeben, z.B. 0.032 für 3.2 Cent pro Minute. Diese werden unter Berücksichtigung der Taktzeit umgerechnet für die tatsächlich anfallenden Kosten, welche dann im imon-Client angezeigt werden.

4. Feld: LC-Default-Route

Der Inhalt kann Y oder N sein. Dabei bedeutet:

## 1. Dokumentation des Paketes ISDN

- Y: Der angegebene Zeitbereich wird beim LC-Routing als Default-Route verwendet. Wichtig: In diesem Fall muss auch `ISDN_CIRC_x_ROUTE='0.0.0.0/0'` sein!
- N: Der angegebene Zeitbereich dient nur zum Berechnen von Kosten, er wird beim automatischen LC-Routing jedoch nicht weiter verwendet.

Beispiel:

```
ISDN_CIRC_1_TIMES='Mo-Fr:09-18:0.049:N Mo-Fr:18-09:0.044:Y Sa-Su:00-24:0.044:Y'  
ISDN_CIRC_2_TIMES='Mo-Fr:09-18:0.019:Y Mo-Fr:18-09:0.044:N Sa-Su:00-24:0.044:N'
```

Die Bedeutung ist dabei wie folgt: Circuit 1 (Provider Planet-Interkom) soll abends an Werktagen und komplett am Wochenende verwendet werden, jedoch tagsüber an Werktagen der Circuit 2 (Provider Compuserve).

**Wichtig:** Die bei `ISDN_CIRC_x_TIMES` angegebenen Zeiten müssen die ganze Woche abdecken. Ist das nicht der Fall, kann keine gültige Konfiguration erzeugt werden.

*Wenn die Zeitbereiche aller LC-Default-Route-Circuits ("Y") zusammengenommen nicht die komplette Woche beinhalten, gibt's zu diesen Lückenzeiten keine Default-Route. Damit ist dann Surfen im Internet zu diesen Zeiten ausgeschlossen!*

Beispiel:

```
ISDN_CIRC_1_TIMES='Sa-Su:00-24:0.044:Y Mo-Fr:09-18:0.049:N Mo-Fr:18-09:0.044:N'  
ISDN_CIRC_2_TIMES='Sa-Su:00-24:0.044:N Mo-Fr:09-18:0.019:Y Mo-Fr:18-09:0.044:N'
```

Hier wurde für die Werktage von 18-09 Uhr "N" gesetzt. Zu diesen Zeiten gibt es keine Route in's Internet: Surfen verboten!

Noch ein ganz einfaches Beispiel:

```
ISDN_CIRC_1_TIMES='Mo-Su:00-24:0.0:Y'
```

für diejenigen, die eine Flatrate nutzen.

Und noch eine letzte Bemerkung zum LC-Routing:

Deutsche Feiertage werden wie Sonntage behandelt.

### 1.1.5. OPT\_TELMOND - telmond-Konfiguration

Mit `OPT_TELMOND` kann man einstellen, ob der telmond-Server aktiviert werden soll. Dieser horcht auf eingehende Telefonanrufe und teilt über TCP-Port 5001 die anrufende und die angerufene Telefonnummer mit. Diese Information kann vom Windows- und Unix/Linux-Client `imonc` abgefragt und angezeigt werden (s.a. Kapitel "Client-/Server-Schnittstelle imond").

Zwingende Voraussetzung ist hierfür die Installation einer ISDN-Karte und die Konfiguration von `OPT_ISDN` und den dazugehörenden Konfigurations-Variablen.

Im laufenden Betrieb kann die korrekte Funktion von telmond unter Linux/Unix/Windows überprüft werden mit:

```
telnet fli4l 5001
```



Dann sollte der letzte Anruf gezeigt und anschließend die telnet-Verbindung sofort wieder geschlossen werden.

Der Port 5001 ist nur vom LAN aus erreichbar. Standardmäßig wird ein Zugriff von außen über die Firewall-Konfiguration abgeblockt. Möchte man jedoch auch den Zugriff innerhalb des LANs anders regeln, kann dies über die weitere telmond-Konfigurationsvariablen eingestellt werden, siehe unten.

Standard-Einstellung: `START_TELMOND='yes'`

**TELMOND\_PORT** TCP/IP-Port, auf dem telmond auf Verbindungen horcht. Der Standardwert '5001' sollte nur in Ausnahmefällen geändert werden.

**TELMOND\_LOG** Bei `TELMOND_LOG='yes'` werden sämtliche einkommenden Telefonanrufe in der Datei `/var/log/telmond.log` gespeichert. Der Inhalt der Datei kann mit dem imond-Client `imonc` unter Unix/Linux und Windows abgefragt werden.

Abweichende Pfade bzw. nach Clients aufgeteilte Log-Dateien können weiter unten konfiguriert werden.

Standard-Einstellung: `TELMOND_LOG='no'`

**TELMOND\_LOGDIR** Ist das Protokollieren eingeschaltet, kann über `TELMOND_LOGDIR` ein alternatives Verzeichnis statt `/var/log` angegeben werden, z.B. `'/boot'`. Dann wird die LOG-Datei `telmond.log` auf dem Bootmedium angelegt. Dazu muß dann das Bootmedium auch Read/Write "gemounted" sein. Wird hier `'auto'` angegeben, befindet sich das Logfile je nach Konfiguration unter `/boot/persistent/isdn` oder an einem anderen durch `FLI4L_UUID` bestimmten Pfad. Wenn `/boot` nicht Read/Write gemountet ist, wird das File in `/var/run` angelegt.

**TELMOND\_MSN\_N** Sollen bestimmte Anrufe nur auf einigen PC-Clients im `imonc` sichtbar werden, kann ein Filter eingestellt werden, mit dem Anrufe auf spezielle MSNs nur für diese PC-Clients protokolliert werden.

Ist so etwas notwendig, z.B. in einer WG, wird die Variable `TELMOND_MSN_N` auf die Anzahl der MSN-Filter eingestellt.

Standard-Einstellung: `TELMOND_MSN_N='0'`

**TELMOND\_MSN\_x** Für jeden MSN-Filter ist eine Liste von IP-Adressen anzugeben, für welche die Anrufe auf eingetragene MSN sichtbar werden sollen.

Die Variable `TELMOND_MSN_N` bestimmt die Anzahl solcher Konfigurationen, siehe oben.

Der Aufbau der Variablen ist:

```
TELMOND_MSN_x='MSN IP-ADDR-1 IP-ADDR-2 ...'
```

Ein einfaches Beispiel:

```
TELMOND_MSN_1='123456789 192.168.6.2'
```

Soll ein Anruf auf eine bestimmte MSN auf mehreren Rechnern sichtbar werden, z.B. Fax, sind die IP-Adressen der Rechner hintereinander anzugeben, z.B.

```
TELMOND_MSN_1='123456789 192.168.6.2 192.168.6.3'
```

**TELMOND\_CMD\_N** Sobald ein Telefonanruf (Voice) auf einer bestimmten MSN hereinkommt, können optional bestimmte Kommandos auf dem fli4l-Router ausgeführt werden. Mit TELMOND\_CMD\_N gibt man die Anzahl der konfigurierten Kommandos an.

**TELMOND\_CMD\_x** Mit TELMOND\_CMD\_1 bis TELMOND\_CMD\_n können Kommandos angegeben werden, welche ausgeführt werden, wenn ein Telefonanruf eintrifft.

Die Variable TELMOND\_CMD\_N bestimmt die Anzahl solcher Kommandos, siehe oben.

Die Variable hat folgenden Aufbau:

```
MSN CALLER-NUMBER  COMMAND ...
```

Dabei ist die MSN ohne Vorwahl einzutragen. Als CALLER-NUMBER gibt man die komplette Telefonnummer - also mit Vorwahl - an. Schreibt man als Wert für CALLER-NUMBER ein einfaches Sternchen (\*), wird von telmond die Telefonnummer des Anrufers nicht ausgewertet.

Hier ein Beispiel:

```
TELMOND_CMD_1='1234567 0987654321 sleep 5; imonc dial'
TELMOND_CMD_2='1234568 * switch-on-coffee-machine'
```

Im ersten Fall wird die Kommandofolge "sleep 5; imonc dial" durchgeführt, wenn der Anrufer mit der Telefonnummer 0987654321 die MSN 1234567 anruft. Tatsächlich werden hier 2 Kommandos ausgeführt. Zunächst wird 5 Sekunden gewartet, damit der ISDN-Kanal wieder frei wird, auf dem der Anruf hereinkam. Anschließend wird der fli4l-Client imonc mit dem Argument "dial" gestartet. imonc gibt dieses Kommando 1:1 an den Server imond weiter, welcher dann auf dem Default-Circuit eine Netzverbindung herstellt, z.B. ins Internet. Welche Kommandos das imonc-Client-Programm an den Server imond weitergeben kann, ist im Kapitel "Client-/Server-Schnittstelle imond" erklärt. Damit diese Einstellung funktioniert, muss OPT\_IMONC aus dem Paket "tools" installiert sein.

Das zweite Kommando "switch-on-coffee-machine" wird ausgeführt, wenn ein Anruf auf der MSN 1234568 hereinkommt, unabhängig, woher der Anruf kam. Natürlich gibt es das Kommando "switch-on-coffee-machine" (noch) nicht für fli4l!

Beim Aufruf der Kommandos können folgende Platzhalter verwendet werden:

|    |         |                            |
|----|---------|----------------------------|
| %d | date    | Datum                      |
| %t | time    | Uhrzeit                    |
| %p | phone   | Telefonnummer des Anrufers |
| %m | msn     | Eigene MSN                 |
| %% | percent | das Prozentzeichen selbst  |

Diese Daten können dann von den aufgerufenen Programmen weiter verwendet werden, z.B. zum Verschicken per E-Mail.

**TELMOND\_CAPI\_CTRL\_N** Wenn Sie einen CAPI-fähigen ISDN-Adapter oder eine Remote-CAPI (Typ 160 oder 161) verwenden, kann es sein, dass Sie die CAPI-Controller, an denen telmond auf Anrufe horcht, genauer konfigurieren wollen. Beispielsweise bietet die Fritz!Box Zugriff auf teilweise bis zu fünf verschiedene Controller an, von denen manche sich nicht unterscheiden (siehe hierzu die Informationen unter [http://www.wehavemorefun.de/fritzbox/CAPI-over-TCP#Virtuelle\\_Controller](http://www.wehavemorefun.de/fritzbox/CAPI-over-TCP#Virtuelle_Controller)). Um die zu verwendenden Controller einzuschränken, können Sie hier die Anzahl der zu nutzenden Controller angeben. In der nachfolgenden Array-Variable TELMOND\_CAPI\_CTRL\_% können Sie dann angeben, welche Controller genutzt werden sollen.

Wenn Sie diese Variable nicht nutzen, horcht telmond auf *allen* verfügbaren CAPI-Controllern.

**TELMOND\_CAPI\_CTRL\_x** Wenn sie TELMOND\_CAPI\_CTRL\_N ungleich Null gesetzt haben, müssen Sie in den Einträgen dieses Arrays die Indizes der CAPI-Controller angeben, an denen telmond auf eingehende Anrufe horchen soll.

Beispiel für die Remote-CAPI einer Fritz!Box mit "echtem" ISDN-Anschluss:

```
TELMOND_CAPI_CTRL_N='2'  
TELMOND_CAPI_CTRL_1='1' # horche auf eingehende ISDN-Anrufe  
TELMOND_CAPI_CTRL_2='3' # horche auf Anrufe auf dem internen S0-Bus
```

Beispiel für die Remote-CAPI einer Fritz!Box mit analogem Anschluss und SIP-Weiterleitung:

```
TELMOND_CAPI_CTRL_N='2'  
TELMOND_CAPI_CTRL_1='4' # horche auf eingehende analoge Anrufe  
TELMOND_CAPI_CTRL_2='5' # horche auf eingehende SIP-Anrufe
```

### 1.1.6. OPT\_RCAPID - Remote CAPI Dämon

Dieses OPT konfiguriert auf dem fli4l-Router das Programm rcapid, das Zugriff auf die ISDN-CAPI-Schnittstelle des Routers übers Netzwerk anbietet. Geeignete Anwendungen können somit die ISDN-Karte des Routers übers Netzwerk so nutzen, als ob sie lokal zur Verfügung stünde. Die Funktionalität ähnelt somit dem Paket "mtgcapri". Der Unterschied besteht jedoch darin, dass "mtgcapri" nur Windows-Systeme als Klienten unterstützt, während die Netzwerk-Schnittstelle des rcapid nach Kenntnis des Autors zur Zeit nur Linux-Systeme nutzen können. Insofern ergänzen sich in gemischten Umgebungen mit Windows- und Linux-Systemen beide Pakete ideal.

#### Konfiguration des Routers

**OPT\_RCAPID** Diese Variable aktiviert das Anbieten der auf dem Router verfügbaren ISDN-CAPI für entfernte Klienten. Mögliche Werte sind "yes" und "no". Wird diese Variable auf "yes" gesetzt, wird der Internet-Dämon inetd so konfiguriert, dass auf Anfragen am rcapid-Port 6000 der rcapid-Dämon gestartet wird (der Port kann mit Hilfe der Variablen RCAPID\_PORT geändert werden).

Beispiel: OPT\_RCAPID='yes'

**RCAPID\_PORT** Diese Variable enthält den TCP-Port, den der rcapid-Dämon benutzen soll.

Standard-Einstellung: RCAPID\_PORT='6000'

## Konfiguration der Linux-Klienten

Um auf einem Linux-Rechner die entfernte CAPI-Schnittstelle nutzen zu können, muss die modulare libcapi20-Bibliothek verwendet werden. Aktuelle Linux-Distributionen installieren bereits eine solche CAPI-Bibliothek (z.B. Debian Wheezy). Falls nicht, können die Quellen von [http://ftp.de.debian.org/debian/pool/main/i/isdnutils/isdnutils\\_3.25+dfsg1.orig.tar.bz2](http://ftp.de.debian.org/debian/pool/main/i/isdnutils/isdnutils_3.25+dfsg1.orig.tar.bz2) heruntergeladen werden; nach dem Auspacken und dem Wechsel ins Verzeichnis “capi20” kann die CAPI-Bibliothek mit dem üblichen Dreierschritt “./configure”, “make” und “sudo make install” übersetzt und installiert werden. Ist die Bibliothek erst einmal installiert, muss man nur noch in der Konfigurationsdatei `/etc/capi20.conf` vermerken, auf welchem Rechner das rcapid-Programm läuft. Ist der Router beispielsweise unter dem Namen “fli4l” erreichbar, sieht die Konfigurationsdatei folgendermaßen aus:

```
REMOTE fli4l 6000
```

Das war's! Ist auf dem Linux-Klienten das Programm “capiinfo” installiert (Teil des capi4k-utils-Pakets vieler Distributionen), dann kann man sofort die entfernte CAPI-Schnittstelle testen:

```
kristov@peacock ~ $ capiinfo
Number of Controllers : 1
Controller 1:
Manufacturer: AVM Berlin
CAPI Version: 1073741824.1229996355
Manufacturer Version: 2.2-00 (808333856.1377840928)
Serial Number: 0004711
BChannels: 2
[...]
```

Unter “Number of Controllers” wird die Anzahl der ISDN-Karten vermerkt, die auf dem Klienten nutzbar sind. Steht hier “0”, dann funktioniert zwar die Verbindung zum rcapid-Programm, aber auf dem Router wird/werden die ISDN-Karte(n) nicht erkannt. Funktioniert die Verbindung zum rcapid-Programm gar nicht (z.B. weil `OPT_RCAPID` auf “no” steht), dann erscheint die Fehlermeldung “capi not installed - Connection refused (111)”. In diesem Fall sollte man die Konfiguration noch einmal überprüfen.

# A. Anhang zum Paket ISDN

## A.1. ISDN

### A.1.1. Technische Details zu Einwahl und Routing bei ISDN

Dieses Kapitel ist nur für Leute interessant, die ein wenig verstehen wollen, was intern passiert, die spezielle Konfigurationswünsche haben oder die nach der Lösung für Probleme suchen. Andere sollten dieses Kapitel bitte *nicht* lesen.

Nach dem Herstellen einer Verbindung zum Provider konfiguriert der `ipppd`-Daemon, der diese Verbindung hergestellt hat, das Interface neu, um die ausgehandelten IP-Adressen zu setzen. Dabei werden vom Linux-Kern automatisch auch Routen gesetzt, die der Remote-IP und der Netzmaske entsprechen und vorhandene, spezielle Routen werden gelöscht. Wird keine Netzmaske vorgegeben, leitet der `ipppd` aus der Remote-IP die Netzmaske ab (er benutzt dazu die Überholte Einteilung in Class A,B und C Subnetze). Das Verschwinden der Routen und die automatisch gesetzten neuen Routen haben immer wieder für Probleme gesorgt:

- Firmennetze waren nicht mehr erreichbar, weil die Routen verschwunden waren oder von den gesetzten neuen Routen überlagert wurden
- Interfaces wählten sich scheinbar ohne Grund ein, da ein Paket statt über die default Route über die vom Kern generierte Route auf ein anderes Interface ging
- ...

Daher wird nunmehr versucht, diese unerwünschten Routen zu verhindern. Dazu werden folgende Dinge geändert:

- `remote ip` wird auf 0.0.0.0 gesetzt, wenn nichts anderes spezifiziert ist. Dadurch verschwinden die Routen, die beim Konfigurieren des Interfaces vom Kern eingerichtet werden.
- zusätzlich angegebene Routen über den Circuit werden in einer Datei zwischengespeichert
- wird eine Netzwerkmaske für den Circuit angegeben, wird diese an den `ipppd` weitergereicht, damit der sie nach Aushandeln der IP für die Konfiguration des Interfaces (und damit für die Generierung von Routen) nutzt.
- nach dem Einwählen werden die zwischengespeicherten Routen des Circuits ausgelesen und wieder gesetzt (sie wurden vom Kern beim Neukonfigurieren des Interfaces durch `ipppd` gelöscht)
- nach dem Auflegen wird das Interface wieder neu konfiguriert und die Routen werden neu gesetzt um die Ausgangssituation wieder herzustellen.

Die Konfiguration der Circuits sieht dann wie folgt aus:

- item default route

```
ISDN_CIRC_%_ROUTE='0.0.0.0'
```

Ist der Circuit ein lcr circuit und gerade “aktiv”, wird eine default route auf diesen Circuit (bzw. das dazugehörige Interface) gesetzt. Nach dem Einwählen erscheint eine Host-Route zum Provider, die nach dem Auflegen wieder verschwindet.

- spezielle Routen

```
ISDN_CIRC_%_ROUTE='network/netmaskbits'
```

Es werden die angegebenen Routen auf den Circuit (bzw. das dazugehörige Interface) eingerichtet. Nach dem Einwählen werden die von Kern gelöschten Routen wieder hergestellt und es gibt eine Host-Route zum Einwahlknoten. Nach dem Auflegen wird der Originalzustand wieder hergestellt.

- remote ip

```
ISDN_CIRC_%_REMOTE='ip address/netmaskbits'  
ISDN_CIRC_%_ROUTE='network/netmaskbits'
```

Beim Konfigurieren des Interfaces erscheinen Routen in das Zielnetz (entsprechend ip-adress AND netmask). Wird die spezifizierte IP nach dem Einwählen beibehalten (dass heißt, es wird keine andere ip während des Verbindungsaufbaus ausgehandelt), bleibt die Route bestehen.

Wird allerdings beim Einwählen eine andere IP ausgehandelt, ändert sich die Route entsprechend (new ip AND netmask).

Für die zusätzlichen Routen gilt das oben gesagte.

Das löst hoffentlich vorläufig *alle* Probleme, die mit speziellen Routen auftraten. Die Form der Korrektur mag sich in Zukunft noch ändern, an dem Prinzip ändert sich hoffentlich nichts mehr.

### A.1.2. Fehlermeldungen des ISDN-Subsystems (englisch, i4l-Dokumentation)

Im folgenden ein Auszug aus der Isdn4Linux Dokumentation (man 7 isdn\_cause).

Cause messages are 2-byte information elements, describing the state transitions of an ISDN line. Each cause message describes its origination (location) in one byte, while the cause code is described in the other byte. Internally, when EDSS1 is used, the first byte contains the location while the second byte contains the cause code. When using 1TR6, the first byte contains the cause code while the location is coded in the second byte. In the Linux ISDN subsystem, the cause messages visible to the user are unified to avoid confusion. All user visible cause messages are displayed as hexadecimal strings. These strings always have the location coded in the first byte, regardless if using 1TR6 or EDSS1. When using EDSS1, these strings are preceeded by the character 'E'.

**LOCATION** The following location codes are defined when using EDSS1:

## *A. Anhang zum Paket ISDN*

- 00 Message generated by user.
- 01 Message generated by private network serving the local user.
- 02 Message generated by public network serving the local user.
- 03 Message generated by transit network.
- 04 Message generated by public network serving the remote user.
- 05 Message generated by private network serving the remote user.
- 07 Message generated by international network.
- 0A Message generated by network beyond inter-working point.

**CAUSE** The following cause codes are defined when using EDSS1:

- 01 Unallocated (unassigned) number.
- 02 No route to specified transit network.
- 03 No route to destination.
- 06 Channel unacceptable.
- 07 Call awarded and being delivered in an established channel.
- 10 Normal call clearing.
- 11 User busy.
- 12 No user responding.
- 13 No answer from user (user alerted).
- 15 Call rejected.
- 16 Number changed.
- 1A Non-selected user clearing.
- 1B Destination out of order.
- 1C Invalid number format.
- 1D Facility rejected.
- 1E Response to status enquiry.
- 1F Normal, unspecified.
- 22 No circuit or channel available.
- 26 Network out of order.
- 29 Temporary failure.
- 2A Switching equipment congestion.
- 2B Access information discarded.
- 2C Requested circuit or channel not available.
- 2F Resources unavailable, unspecified.
- 31 Quality of service unavailable.
- 32 Requested facility not subscribed.
- 39 Bearer capability not authorised.
- 3A Bearer capability not presently available.
- 3F Service or option not available, unspecified.
- 41 Bearer capability not implemented.
- 42 Channel type not implemented.
- 45 Requested facility not implemented.
- 46 Only restricted digital information bearer.
- 4F Service or option not implemented, unspecified.
- 51 Invalid call reference value.
- 52 Identified channel does not exist.
- 53 A suspended call exists, but this call identity does not.
- 54 Call identity in use.
- 55 No call suspended.
- 56 Call having the requested call identity.
- 58 Incompatible destination.
- 5B Invalid transit network selection.

### *A. Anhang zum Paket ISDN*

- 5F Invalid message, unspecified.
- 60 Mandatory information element is missing.
- 61 Message type non-existent or not implemented.
- 62 Message not compatible with call state or message or message type non existent or not implemented.
- 63 Information element non-existent or not implemented.
- 64 Invalid information element content.
- 65 Message not compatible.
- 66 Recovery on timer expiry.
- 6F Protocol error, unspecified.
- 7F Inter working, unspecified.



# Abbildungsverzeichnis

# **Tabellenverzeichnis**

# Index

ISDN\_%\_IO, [4](#)  
ISDN\_%\_IO0, [4](#)  
ISDN\_%\_IO1, [4](#)  
ISDN\_%\_IP, [6](#)  
ISDN\_%\_MEM, [4](#)  
ISDN\_%\_PORT, [6](#)  
ISDN\_%\_TYPE, [4](#)  
ISDN\_CIRC\_N, [8](#)  
ISDN\_CIRC\_x\_AUTH, [14](#)  
ISDN\_CIRC\_x\_BANDWIDTH, [9](#)  
ISDN\_CIRC\_x\_BUNDLING, [9](#)  
ISDN\_CIRC\_x\_CALLBACK, [13](#)  
ISDN\_CIRC\_x\_CBDELAY, [13](#)  
ISDN\_CIRC\_x\_CBNUMBER, [13](#)  
ISDN\_CIRC\_x\_CHARGEINT, [15](#)  
ISDN\_CIRC\_x\_CLAMP\_MSS, [11](#)  
ISDN\_CIRC\_x\_DEBUG, [14](#)  
ISDN\_CIRC\_x\_DIALIN, [12](#)  
ISDN\_CIRC\_x\_DIALOUT, [12](#)  
ISDN\_CIRC\_x\_EAZ, [14](#)  
ISDN\_CIRC\_x\_FRAMECOMP, [11](#)  
ISDN\_CIRC\_x\_HEADERCOMP, [11](#)  
ISDN\_CIRC\_x\_HUP\_TIMEOUT, [14](#)  
ISDN\_CIRC\_x\_LOCAL, [10](#)  
ISDN\_CIRC\_x\_MRU, [10](#)  
ISDN\_CIRC\_x\_MTU, [10](#)  
ISDN\_CIRC\_x\_NAME, [8](#)  
ISDN\_CIRC\_x\_PASS, [12](#)  
ISDN\_CIRC\_x\_REMOTE, [10](#)  
ISDN\_CIRC\_x\_REMOTENAME, [11](#)  
ISDN\_CIRC\_x\_ROUTE\_N, [12](#)  
ISDN\_CIRC\_x\_ROUTE\_X, [12](#)  
ISDN\_CIRC\_x\_SLAVE\_EAZ, [14](#)  
ISDN\_CIRC\_x\_TIMES, [15](#)  
ISDN\_CIRC\_x\_TYPE, [9](#)  
ISDN\_CIRC\_x\_USEPEERDNS, [8](#)  
ISDN\_CIRC\_x\_USER, [12](#)  
ISDN\_DEBUG\_LEVEL, [6](#)  
ISDN\_FILTER, [7](#)  
ISDN\_FILTER\_EXPR, [7](#)  
ISDN\_LZS\_COMP, [7](#)  
ISDN\_LZS\_DEBUG, [7](#)  
ISDN\_LZS\_TWEAK, [7](#)  
ISDN\_VERBOSE\_LEVEL, [6](#)  
  
OPT\_ISDN, [3](#)  
OPT\_ISDN\_COMP, [7](#)  
OPT\_RCAPID, [19](#)  
OPT\_TELMOND, [16](#)  
  
RCAPID\_PORT, [19](#)  
  
TELMOND\_CAPI\_CTRL\_N, [18](#)  
TELMOND\_CAPI\_CTRL\_x, [19](#)  
TELMOND\_CMD\_N, [18](#)  
TELMOND\_CMD\_x, [18](#)  
TELMOND\_LOG, [17](#)  
TELMOND\_LOGDIR, [17](#)  
TELMOND\_MSN\_N, [17](#)  
TELMOND\_MSN\_x, [17](#)  
TELMOND\_PORT, [17](#)