KHOMP

www.khomp.com.br

Mod Khomp: User Guide

Sumário

First considerations

This document covers information about the *Endpoint* of Khomp as a whole, include the options, *applications*, **CLI** commands, among others.

For first installation procedures, please see the README.

Configuration

Configuring the Khomp Endpoint is a task that consists of three steps:

- Configuration of the boards through the K3L;
- Setting up the Endpoint;
- Setting up the FreeSWITCH.

These steps are described more fully below.

K3L API Configuration

This step is carried out in a semi-automated way using the **khompwizard** program, a wizard that configures the basic parameters of the system boards. This wizard initializes the configuration files using information from the user when they are needed, initializing the standard settings to default values.

Typically, this program runs automatically after installation of the system. However, you may need to run it manually if an update is being performed, or if new cards were added to the system after installing new drivers.

If you need to set advanced parameters of the board and/or signaling, the program **k3lconfig** allows you to access all the available settings for each installed card. For more information about this program, check the **k3lconfig User's Guide**. For solving synchronization issues, check the Troubleshooting section for manual configuration of the boards.

Endpoint Configuration

The system's default configuration normally meet most user's needs. However, the configuration of the Endpoint Khomp can be modified through the configuration file **'/usr/local/freeswitch/conf/autoload_configs/khomp.conf.xml'**.

The list of options is as follows:

<channels>

Define general settings of all channels of Khomp:

```
Sintaxe: <param name="..." value="..."/>
```

- **accountcode**: Sets the default account code to calls in the Endpoint. This option can be any alphanumeric string;
- **dialplan**: Name of the dialplan module in use.
- **auto-fax-adjustment**: Enable ("yes") or disables ("no") the automatic adjustment of the channel (disable the echo canceller and the suppression DTMF) tone to detect FAX (local option) ;

- **auto-gain-control**:Enable ("yes") or disables ("no") the activation of the automatic gain control (AGC) by the Endpoint (local option);
- **context-digital**: Context for incoming connections on digital boards (the default is "khomp-DD-LL", where "DD" will be replaced at the time of connection by the device number, "LL" by the number of the link, "CCC" by channel number and "SSSS" for the device serial number);
- **context-fxo**: Context for incoming connections on FXO cards (the default is "khomp-CC-DD", where "DD" will be replaced at the time of connection by the device number, "CC" by channel number, and "SSSS" by the device serial number);
- **context-fxs**: Context for incoming connections on FXS cards (the default is "khomp-CC-DD", where "DD" will be replaced at the time of connection by the device number, "CC" by channel number, and "SSSS" by the device serial number);
- **context-gsm-call** (or **context-gsm**): Context of entry for GSM boards (the default is "khomp-CC-DD", where "DD" will be replaced at the time of connection by the device number, "CC" by channel number, and "SSSS" by the device serial number);
- **context-gsm-sms**: Context for incoming SMSs (the default is "khomp-sms-CC-DD", where "DD" will be replaced by the number of device, "CC" by channel number and "SSSS" by the device's serial number);
- **context-pr**: Context for incoming connections on boards KPR (default is "khomp-CC-DD", where "DD" will be replaced at the time of connection by the device number, "CC "by channel number);
- **delay-ringback-co**: Sets the delay to enable the generation of call control tone (ringback) by the Endpoint Khomp when there is an ringback indication from signaling and there is no audio being sent by the channel which indicated the situation (local option);
- **delay-ringback-pbx**: Sets the delay to enable the generation of call control tone (ringback) by the Endpoint Khomp when there is an ringback indication, and the audio has no tone (silence) (local option);
- **disconnect-delay**: Sets the time in milliseconds to perform processing a disconnect event, to ignore situations where other equipment performing the double service to overthrow collect calls (local option);
- **drop-collect-call**: Enables/Disables the action of dropping collect calls. If enabled, all collect calls will be dropped no matter what KDropCollectCall is set to (the default value is "no");
- **echo-canceller** (former 'echocanceller): Active ("yes") or disables ("no") echo cancellation automatic Endpoint (local option);
- **flash-to-digits**: Defines the digits to be sent when the FLASH is detected on FXS channels;
- **fxo-send-pre-audio**: When enabled ("yes") releases audio channel before the outgoing call connection boards KFXO (the default value is "yes");
- **fxs-digit-timeout**: Defines the timeout, in seconds, between digits of a FXS board's extension;
- **fxs-global-orig**: Start number for sequencial branch numbering in FXS cards that are not listed in the **[fxs-branches]** section (the numbering follows ascending order from board number and physical channel number) (default is "0");
- **fxs-co-dialtone**: Sequences of numbers, separated by commas, which fires a continuous tone (of central office) in FXS branches (eg: "0,99" means that, when you dial "0" or "99", the user will hear a continuous dial tone) (default is empty);
- **fxs-bina**: When enabled ("yes"), calls to FXS lines will send digits corresponding to the source phone identification using BINA DTMF signaling (the default value is "no") (local option);
- **ignore-letter-dtmfs**: Defines if the channel should ignore some uncommon DTMF digits detected by the board (A, B, C and D). However, if you need to pass those digits througth the board, you may need to set this option to 'no' (the default value is "yes");
- **input-volume**: Sets the volume gain for incoming audio (entering the board), from -10 to +10 (local option);
- **kommuter-activation**: Sets whether to activate devices kommuter found in the system will be done automatically ("auto") by the channel, or manually ("manual") by the user through the command "khomp kommuter on/off"

- **kommuter-timeout**: Sets the timeout (in seconds) for initializing the kommuter devices. If this timeout is reached without receiving notification of the channel, the devices will switch back to "off" condition. The minimum value is "0", where the links will always remain switched "on", and the maximum is "255";
- **language**: Set language to Khomp board calls;
- **log-to-console**: Set log messages to be printed on the console;
- **log-to-disk** (old "log"): Set log messages to be saved to disk;
- **out-of-band-DTMF** (former **dtmfsuppression**): Activate ("yes") or disables ("no") the removal and DTMF sending these out-of-band (local option);
- **output-volume**: Define o volume de saída das ligações, varia de -10 a +10 ;
- **pulse-forwarding** (former 'pulsedetection): Active ("yes") or disables ("no") for the detection of pulses and converting them into DTMF (local option);
- **r2-preconnect-wait** (former 'r2preconnectwait): Sets the timeout sending the ringback signaling, protocol R2/MFC to start sending audio silence. Only used when "r2-strict-Behavior" is set to "no" (local option);
- **r2-strict-Behaviour**: Enable ("yes") or disables ("no") the behavior of signaling R2/MFC as the standard sets. The default is "no", and can be changed to "yes" if needed to receive / send data precise signaling protocol (condition B, for example) (local option);
- **suppression-delay** (former **suppressiondelay**): Activate ("yes") or disables ("no") the delay necessary to suppress DTMF. If disabled ("no"), also disables suppression of DTMF (local option);
- **trace**: Set debugging options. Should not be used in production unless absolutely necessary;
- **user-transfer-digits**: Defines a sequence of DTMF digits to initiate the transfer between FreeSWITCH® and another PBX (using user signaling, like QSig or FXO FLASH);

<groups>

Defines the groups to be used in channel allocation.

In this case, the options are used to define names for *strings allocation of channels*. The format follows the standard **<group name> = <allocation string>**, where the **allocation string** is the same string used in the bridge application, and **group name** is an arbitrary name chosen by the user.

> For example, to define the group **pstn** as the channels 0 and 5 of the board 0, the following line could be used:

```
<param name="pstn" value="b0c0 + b0c5"/>
```

This group, in turn, could be used in the *bridge application* as **<action application="bridge" data="Khomp/Gpstn/..."/>**.

> You can associate a given input context to a channel group, simply specify a name of context string after the allocation, separated by ':'.

For example, to define the same group **pstn** as channels 0 to 20 of card 0, and defining the incoming context to for channels in this groups to **from-pstn**, one could use the line:

```
<param name="pstn" value="b0c0-20:from-pstn"/>
```

This group would be used the same way as before in the *bridge application*, and all the calls coming from these channels would be treated in context **from-pstn**.

<cadences>

Defines settings for the Endpoint cadences.

In this case, the options are names cadences, followed by one or two pairs of numbers - that define the ranges of tone and silence to be used in cadences. For details, please refer to the configuration file for examples.

<fxs-branches>

Defines source numbers for the board KFXS.

In this case, the options are sequences of prefixes of branches and serial numbers of the boards, which define the basic numbers of source addresses, and the numerical order of the boards. The format of the options is:

```
<param name="prefixo" value="serial1, serial2, ...."/>
```

For example, if two **KFXS-300 SPX** boards with serial numbers **K0374** and **K2352** must be numbered sequentially, starting from branch 200, you may write:

```
<param name="200" value="374, 2352"/>
```

This will define the first branch of board '*K0374* as number 200, the second as 201, and so one. The first branch from board K2352 will have number 230 (as K0374 has 30 channels), the second will be numbered 231, and so on - until the last channel, numbered 259.

For more details, please refer to the configuration file for other exemples.

<fxs-hotlines>

Sets hotlines for the KFXS based boards

In this case, the options are sequences of branches and sequences of destination numbers, which define branches to be treated as "hotlines" and numbers to be dialed when they are take off hook. For instance:

```
<param name="100" value="1234"/>
<param name="200" value="4321"/>
```

In the first line, the branch numbered 100 will call extension 1234 when taken off hook, while in the second one, branch 200 will call number 4321 when taken off hook.

<fxs-options>

Allows you to set specific settings for FXS extension.

In this case, the settings are extension numbers (based on those defined in the **<fxs-branches>**), and the options and their values.

- context;
- input-volume;
- output-volume;
- calleridnum;
- calleridname;
- language;
- accountcode;
- flash-to-digits.

Each option is separated from each other by a pipe "|" or a slash "/" and defined after the colon ":". Example:

```
<param name="200" value="input-volume:1|context:master-branch" />
```

For more information on the syntax and examples, please refer to the configuration file.

For more information visit the configuration file '**khomp.conf.xml**'.

FreeSWITCH Configuration

When connections are received on the boards and devices Khomp, they are forwarded by the *Endpoint* to specific contexts within the dialplan of FreeSWITCH®. These settings can be changed via the configuration file **khomp.conf.xml**, available on the FreeSWITCH configuration directory (by default, "**/usr/local/freeswitch/conf/autoload_configs**").

For details about the specific contexts, see section Endpoint configuration.

Below are details of how to configure the settings for incoming calls

Contexts for E1 channels

For E1 boards, inbound contexts are predefined as option **context-digital** with default value as following:

```
<param name="context-digital" value="khomp-DD-LL"/>
```

This standard defines the context that links will be redirected in accordance with the number of the board and number of the link: **DD** is the device number (two digits), and **LL** is the number of the link (also with two digits).

However, it is possible to configure other inbound contexts, with different formats. There is format **CCC**, which means the channel number on the card (three digits), and **SSSS**, which represents the serial board number (with four digits).

Examples for configuration entries (**khomp.conf.xml**):

```
<!-- Serial board number and sequential link (ex: khomp-01-00) -->
<param name="context-digital" value="khomp-DD-LL"/>

<!-- Serial board number and sequential link (ex: khomp-3049-00) -->
<param name="context-digital" value="khomp-SSSS-LL"/>

<!-- Sequential board number and the channel (ex: khomp-00-001) -->
<param name="context-digital" value="khomp-DD-CCC"/>

<!-- Receive all calls in one context (khomp-digital) -->
<param name="context-digital" value="khomp-digital"/>
```

Follows an example the context usage inside the dialplan:

```
<!--
The present context in 'extensions.conf' will handle calls
that come from the link 0 (first link) of the board 0.
-->
<context name="khomp-00-00">
             .
             .
             .
</context>
```

Another example, using the same format:

```
<!--
The present context in 'extensions.conf' will handle calls
that come from the link 1 (second link) of the board 0.
-->
<context name="khomp-00-01>
             .
             .
             .
</context>
```

A complete example, with a few simple actions:

```
<context name="khomp-00-00">
    <extension name="exemplo_1">
        <condition field="destination_number" expression="^1234$">
            <action application="bridge" data="Khomp/b0L1/2345"/>
        </condition>
    </extension>
    <extension name="exemplo_2">
        <condition field="destination_number" expression="^23(\d{2})$">
            <action application="bridge" data="sofia/${use_profile}/11$1@${sip_from_host}"/>
        </condition>
    </extension>
</context>
```

```
<context name="khomp-00-01">
    <extension name="exemplo_3">
        <condition field="destination_number" expression="^1111$">
            <action application="bridge" data="Khomp/b0L0/2345"/>
        </condition>
    </extension>
</context>
```

This dialplan defines that:

1. The incoming calls on the link **0** of the board **0** will have the following handling:
    - Calls to the extension 1234 for will be redirected to the second link on the first board (b0L1), calling number **2345**;
    - Calls to any four digit number starting with**23** will be redirected to SIP phones numbered **11** plus the last two digits of the number dialed.
2. The incoming calls on the link **1** of the board **0** for the number **1111** will be redirected to the first link of the first board (b0L0) calling number **2345**.

Contexts of FXS/FXO/GSM channels

Just as in the context of E1 cards, inbound links are forwarded to the Endpoint FreeSwitch.

The context is pre-defined as follows, according to the file **khomp.conf.xml**:

```
<param name="context-gsm" value="khomp-DD-CC"/> ;placas GSM

<param name="context-fxs" value="khomp-DD-CC"/> ; placas FXS

<param name="context-fxo" value="khomp-DD-CC"/> ; placas FXO
```

For these options, **DD** is the device number (two digits), and **CC** is the channel number (also two digits). There is also the **SSSS** format, which represents the serial number board.

- **NOTE**: In the **KGSM** board, incoming calls are **always** redirected to the **"s"** extension, since the GSM protocol does not identify the target number, only the originator - if not omitted.

Priority settings on the FXS branches

On calls originated from an FXS branch, the Endpoint searches for a valid extension (digits sent) after the DTMF **#** or after the *timeout* (option **fxs-digit-timeout**). That search is done in the context defined in section **<fxs-options>**, or if no context configured, the search is done in context defined in **context-fxs**.

Contexts for SMS messages (GSM only)

SMS messages are received by the Khomp *Endpoint* and forwarded to FreeSWITCH as a normal connection but no audio, which has some variables set with information received in the message - for more information on these variables, see the documentation of the variables of the *Endpoint*. This context can also be modified in the same way as the above contexts.

The default value for this option follows (**khomp.conf.xml**):

```
<param name="context-gsm-sms" value="khomp-sms-DD-CC"/>
```

Where **DD** is the device number (two digits), and **CC** is the channel number (also with two digits). For example:

```
<context name="khomp-sms-00-01">
    <extension name="sms">
        <condition field="destination_number" expression="^s$">
            <action application="log" data="DEBUG KSmsType=${KSmsType}"/>
            <action application="log" data="DEBUG KSmsBody=${KSmsBody}"/>
        </condition>
    </extension>
</context>
```

Contexts for Khomp_PR channels (KPR)

For these cards, inbound links have a pre-defined context, as shown below:

```
<param name="context-pr" value="khomp-DD-CC"/>
```

In this case, **DD** is the device number (two digits), and**CC** is the channel number (also two digits). The name and format of this context can also be changed through the "context-pr" in the configuration file.

Groups contexts

The section **groups**, in the configuration file **khomp.conf.xml**, can be used to define specific settings for certain groups of channels.

This section is detailed in the section **Endpoint Configuration**.

Using the *bridge* application

The **bridge** application is responsible for generating calls from the FreeSWITCH from a dialplan. This application can be used to generate calls from different Endpoints technologies, following a specific format to define destination, dialing options and define the communication *Endpoint* to be used.

Fields relating to the Khomp *Endpoint*

When used for **Khomp** channels, the *bridge string* can have two, three or four fields separated by slash (/). Some example strings:

```
<action application="bridge" data="Khomp/B2L0/32625644"/>
<action application="bridge" data="Khomp/*B2L0/32625644"/>
<action application="bridge" data="Khomp/S0411/99991234"/>
<action application="bridge" data="Khomp/Gpstn/99991234"/>
<action application="bridge" data="Khomp/*Gpstn/99991234"/>
<action application="bridge" data="Khomp/B2C58/32625644/category=4:orig=4855553232"/>
<action application="bridge" data="Khomp/b0c9"/>
<action application="bridge" data="Khomp/b0c1+b0c14"/>
<action application="bridge" data="Khomp/r304"/>
```

In the first five examples, three fields have been specified; in the sixth, four fields are used; in the last three examples, just two are used.

The fields description for the Khomp Endpoint:

- **1st** field: **Khomp**: identifying the type of *Endpoint* in question;
- **2nd** field: **B2L0**, **S0411**, **Gpstn**, etc: represents the **Policy for channel allocation** (detailed below);
- **3rd** field: **32625644** and **99991234**: the destination numbers (missing for calls to **KFXS** channels);
- **4th** field: **category=4:orig=4855553232**: additional options, detailed below.

NOTE: The *bridge* allocation string with only two fields is specific to the KFXS boards, where the destination is the channel itself.

Policy for channel allocation

The policy for allocation of channels on the **Khomp** Endpoint can be specified in the **bridge** string itself or in the **groups** section (inside the configuration file **khomp.conx.conf**). To specify boards, links and channels, the following syntax is available (considering X, Y and Z as any numbers):

- b**X** -- search the channels on the board **X**, ascending order;
- b**X**L**Y** -- search channel on the link **Y** from **X** board, ascending order;
- b**X**c**Y** -- try to allocate channel **Y** from board **X**;
- b**X**c**Y**-**Z** -- search for channels starting from channel **Y** to channel **Z** (inclusive) of board **X**, ascending order;
- B**X**c**Y**-**Z** -- same as above, but descending order;
- s**X** -- search the channels on the board of serial number **X**, ascending order;
- s**X**L**Y** -- search channel on the link **Y** from board of serial number **X**, ascending order;
- s**X**c**Y** -- try to allocate channel **Y** from board of serial number **X**;
- s**X**c**Y**-**Z** -- search for channels starting from channel **Y** to channel **Z** (inclusive) from board of serial number **X**, ascending order;
- S**X**c**Y**-**Z**-- same as above, but descending order.

To search for extensions of cards**KFXS** according to the extension number, can be used the following syntax (whereas X and Y valid extension numbers):

- r**X**- search branch numbered **X**;
- R**X**- equivalent to the above;
- r**X**-**Y**- search from branch **X** to **Y**, ascending order;
- R**X**-**Y**- search from branch **X** to **Y**, descending order.

The capitalization of the letter 'B', 'S' or 'R' defines the search order of the channels: if capitalized, order is descending; otherwise, ascending.

As for the allocation of channels across groups, the following syntax is available:

- ggroupname - uses the string defined the group "groupname" in the configuration file (detailed in the configuration section of the Endpoint).
- Ggroupname - equivalent to the above.

Grouping channel allocations

There are cases where you need to get more channels for a particular device or particular group of extensions. For this, there is an extension available in string allocation, with respect to the use of token sum (**+**) to concatenate multiple strings*binding*, as follows:

```
<action application="bridge" data="Khomp/B1L0+B2L0/32332933"/>
<action application="bridge" data="Khomp/*B2+B3+B4/99887766"/>
<action application="bridge" data="Khomp/S0411+B1L0/99887766"/>
<action application="bridge" data="Khomp/Gpstn1+Gpstn2/99991234"/>
<action application="bridge" data="Khomp/*gOperadora1+gOperadora2/98891234"/>
```

This grouping is available for the *application* **bridge** and on the specification of groups. The processing of allocation takes place from left to right - except when using cyclic channel allocation, where**all** the specified channels are scanned simultaneously.

Cyclical and/or *fair* allocation

Another way for allocation of channels is the cyclic and/or *fair* allocation, which chooses the channel that has **completed** the the lowest number of **outgoing** calls. This mode of allocation may be used by passing an asterisk (*****) before the allocation string of channels (as can be seen in the section above, in the second and fifth examples).

When started with an asterisk (*****), other forms of allocation (increasing, decreasing, etc) are used to decide what channel will be allocated when there are two or more channels with less number of outgoing calls.

- **WARNING: The use of fair and/or cyclic is recommended only for analog (KFXO), branches (KFXS) and cellular interface (KGSM) boards**. E1 connections should allocate channels in one way (ascending/descending) from one side and the opposite on the other to avoid problems of double seizure (which may occur in R2/MFC signaling). Fair/cyclic allocation also costs more in memory and processor footprint, which tends to be a higher cost in E1 due to the higher number of channels (30 in each link).

  For these reasons, fair/cyclic allocations should only be used on signalizations where it can represent any real difference, like equalizing the charge costs of the lines, the total usage, or the number of connections received by each branch.

Available options

- **orig**: Sets the originator number, **without changing the variable ${CALLERID(num)}**. That is, the option **orig** serves **only** to pass a number of different source of${origination_caller_id_number}. If FreeSWITCH has already set the variable ${origination_caller_id_number}, which is the default behavior, Endpoint automatically uses this value as a reference to the number of origin, without having to pass any additional options.

  On the boards **KGSM**, is set to **restricted**, omits the number of origin. Example:

```
<action application="bridge" data="Khomp/b0/99887766/orig=restricted"/>
```

- **category**: When set to a numeric value, sets the category of outgoing call to this value (available only in R2/MFC signaling);
- **uui**: When adjusted for a number and a string of text, separated by hash ("#"), sends a "UserToUser" to the other end before making the call - the first value will be the descriptor and the second one will be the message as the text (available only in ISDN signaling);
- **ring_cadence**: When set to a cadence name (listed in the **[cadences]** section), uses this for ringing FXS channels;
- **ring**: When set to two numbers separated by a dot ("."), defines the cadences to be used while ringing a FXS channel - the first time is the ringing time, and the second one, the silence time;
- **ring_ext**: When set to two numbers separated by a dot ("."), defines the extended cadences to be used while ringing a FXS channel, executed after the **ring** specification - the first time
- **usr_xfer**: Defines a group of DTMF digits to initiate a transfer between PBXes (using QSig or FXO FLASH, for instance);
- **drop_on**: When set to "message_box", "human_answer", "answering_machine", "carrier_message", "unknown" or a list them - separated by plus sign ("+") or dot (".") - drops the call when detect voice mail box, human answer, answering machine messages, operator messages, or unknown answer pattern - respectively. Available in digital signals (E1 links and boards KGSM). Additionally, the information service is reported to the user in the variable**KCallAnswerInfo**;
- **answer_info**: When specified (take no parameters), report answer information to the user through the variable**KCallAnswerInfo**
- **pre**: When set to a string of DTMF digits, uses these to pre-allocate an output channel in an analog PABX, dial the desired number of B below. Only available for signaling analog (FXO);
- **pre_answer**: When set (need no value), answers the channel before the connection is completed - allowing, for instance, DTMF tones to sent (useful for use in a **DISA**);
- **output_volume**: Sets the output volume of the link (ranges from -10 to +10);
- **input_volume**: Sets the volume of inbound link (ranges from -10 to +10);

List of variables

Here's a list of variables available in the Endpoint:

- **KDropCollectCall**: When set before ringing or answer an incoming call, enables ("yes") or disables ("no", default) the drop of collect calls based on signaling received from the central public, double answer, audio tone recognition (can be defined globally);

- **KR2SendCondition**: When set to a numeric value, before the ringing an incoming call, adjusts the condition for this Endpoint to this value (available only on R2 signaling);
- **KR2GotCategory**: Adjusted by the Endpoint when an incoming call is received, with the category number of the caller (only available on R2 signaling);
- **KR2GotCondition ***: Adjusted by the Endpoint, available after returning from a call made by FreeSWITCH (**bridge** application). Has the condition of the remote end received when making the call (available only for R2 signaling);
- **KISDNGotCause ***: Adjusted by the Endpoint, available after returning from a call made by FreeSWITCH (**bridge** application). Has the ISDN "cause" code received when making the call (available only for ISDN signaling);
- **KCallAnswerInfo ***: Adjusted by the Endpoint, available after returning from a call made by FreeSWITCH (**bridge** application). Contains the service information identified to make the call (available only for digital signaling - E1 and GSM);
- **KSmsDelivered**: Adjusted by the Endpoint when sending a SMS message with the application **KSendSMS**, saying whether the message was delivered successfully ("yes") or not ("no");
- **KSmsErrorCode**: Adjusted by the Endpoint when sending a SMS message with the application **KSendSMS**, containing the error code that happened when sending the message;
- **KSmsErrorName**: Adjusted by the Endpoint when sending a SMS message with the application **KSendSMS**, contains the name of the error or "None" if there has been no error;
- **KSmsType**: Adjusted for the input Endpoint in the context of SMS messages, defines the type of message received (can contain the values "message", "confirm" or "broadcast";
- **KSmsFrom**: Adjusted for the input Endpoint in the context of SMS messages, sets the number of origin of the received message (available on types" message "and" confirm ");
- **KSmsDate**: Adjusted for the input Endpoint in the context of SMS messages, sets the date of sending the message (available on types "message" and "confirm");
- **KSmsSize**: Adjusted for the input Endpoint in the context of SMS messages, contains the size (in bytes) of the received message (available on types "message" and "broadcast");
- **KSmsMode**: Adjusted for the input Endpoint in the context of SMS messages, contains the encoding type of the received message (available on types "message" and "broadcast");
- **KSmsBody**: Adjusted for the input Endpoint in the context of SMS messages, contains the text of the received message (available in types "message" and "broadcast");
- **KSmsDelivery**: Adjusted for the input Endpoint in the context of SMS messages, containing the date of delivery of the message sent earlier (available in type "confirm");
- **KSmsStatus**: Adjusted for the input Endpoint in the context of SMS messages, contains the status of the message sent earlier (available on the type "confirm");
- **KSmsSerial**: Adjusted for the input Endpoint in the context of SMS messages, contains the serial number of the received message (available on the type "broadcast");
- **KSmsPage**: Adjusted for the input Endpoint in the context of SMS messages, contains the page number of the received message (available on the type broadcast");
- **KSmsPages**: Adjusted for the input Endpoint in the context of SMS messages, contains the total number of pages to be received (available in type "broadcast");
- **KUserInfoDescriptor**: Sets/reports protocol descriptor of the User-to-User Information message (ISDN).
- **KUserInfoData**: Sets/reports data in the User-to-User Information message (ISDN).
- **KFaxSent**: Adjusted by the Endpoint when sending FAX with **KSendFax** application, and determines whether the fax was successfully sent ("yes") or not ("No");
- **KFaxReceived**: Adjusted by channel when receiving FAX with **KReceiveFax** application, and determines whether the fax was successfully received ("yes") or not ("no");

- **KFaxResult**: Adjusted by the channel when sending or receiving FAX with the application **KSendFax** or **KReceiveFax** (respectively), and defines the result of execution.

Description of variables

Below, follows an explanation on how to use variables of Khomp Endpoint in the dialplan, to communicate and/or to receive information:

KDropCollectCall

When activated, causes the Endpoint to drop Khomp collect calls through dual service (available for signaling 'R2 Digital' and FXO), through information available on the ISDN protocol and R2/MFC, or by detecting the audio call collect (available for any digital signage for E1, and GSM signaling).

This variable is useful to filter collect calls to certain extensions, and **must** be set before making any type of answer - applications such as **playback** and **bridge** should always be executed after setting this variable, for example.

For better functionality, is also recommended that no call status (*ringback*) is sent before this variable is set, so applications should be performed only after the correct setting of this variable.

This variable can be set locally and globally, both to **yes** or **no**. The adjustment of global variable to **yes** will drop all the collect calls, unless the particular call is set to **no** - this allows the creation of a global filter of collect calls, with few exceptions.

Enabling the variable in context **default**:

<context name="default">

```
<extension name="example">
 .
 .
 .
 <action aplication="set" data="KDropCollectCall=yes"/>
 .
 .
 .
</extension>
```

</context>

Enabling the variable in the global context, remembering that it must be configured in the file **vars.xml**:

```
<X-PRE-PROCESS cmd="set" data="KDropCollectCall=yes"/>
```

KR2SendCondition

When you receive a call, can be set before sending ringback by FreeSWITCH (ie, before the run FreeSWITCH applications **answer**, **bridge**). When used in signaling R2/MFC, this variable sets the condition for B to the numeric value desired.

Exemplo:

```
<!-- Condition "NUMBER CHANGED" warns the caller that the number of B has changed. -->
<action application="KR2SendCondition" data="3"/>
```

KR2GotCategory

When you receive a call, is set by the Endpoint with the category received from the number that originated the call. It is set in signaling R2/MFC, and can be found anywhere in the dialplan.

Example:

```
<action application="log" data="DEBUG KR2GotCategory [${KR2GotCategory}]"/>
```

KR2GotCondition

Variable adjusted by the Endpoint, and available after returning from a call made by FreeSWITCH. Has the condition of B received when making the call. Available only for signaling R2/MFC.

Example:

```
<action application="log" data="DEBUG KR2GotCondition [${KR2GotCondition}]"/>
```

KUserInfoDescriptor

Variable adjusted by the Endpoint in the context of entry, from information received by the ISDN network functionality through user-to-User Information. Contains the descriptor number of the protocol used by the other end, and usually contains the value '0 ', but this is dependent on application.

For further information, consult the specification ITU-T Q931 (more precisely, the specification table 4-26).

Example (working with the descriptor number of the protocol):

```
<action application="log" data="DEBUG KUserInfoDescriptor [${KUserInfoDescriptor}]"/>
```

KUserInfoData

Variable adjusted by the channel in the context of entry, from information received by the ISDN network functionality through user-to-User Information. Contains the actual data, which were received in the form of a string of text.

More information about this feature, see the specification ITU-T Q931.

Example (working with the data received):

```
<action application="log" data="DEBUG KUserInfoData [${KUserInfoData}]"/>
```

It is important to note that the variables are sensitive to the capitalization of letters (case sensitive).

KCallAnswerInfo

Variable adjusted by the Endpoint. It is set in outbound connections, representing the type of answer performed by the other end. May contain the following values:

- ◊ "MessageBox" (*): detected mailbox of a cell phone;
- ◊ "CarrierMessage": message sent before the service provider;
- ◊ "AnsweringMachine" (**): answering answering machine;
- ◊ "HumanAnswer" (**): human service;
- ◊ "Fax": reported when a fax tone is detected.
- ◊ "Unknown": unknown type of care;

(*) This type of service is detected by signals at certain frequencies that are sent before the call comes into a mailbox, and vary by operator. The algorithm captures most of the mailboxes, but can fail if there is not a clear signal, or if it is not within the standards most commonly used;

(**) The difference between these two types of care depends on the specific configuration using the program **k3lanswerinfoconfig**, with the detection only based on heuristics and **never** with an accuracy of 100%.

Console commands

List of available commands in the console for the FreeSWITCH Endpoint of Khomp:

- **khomp channels disconnect** : Disconnect one or more channels. This command sends a message directly to the physical channel of the card in question, requesting a disconnection. Use with caution;
- **khomp channels unblock** : Unlock blocked channels for input or output. Only available in digital signage via E1;
- **khomp clear links**: Clears error counters on the links;
- **khomp clear statistics**: Clears the statistics of channel connections, or statistics for a particular channel;
- **khomp get** : Gets the number of options Endpoint Khomp;
- **khomp kommuter** : Enables or disables kommuters connected via USB on this machine. Only accessible when the configuration "kommuter-activation" is set to "manual".
- **khomp kommuter count**: Gets the amount of kommuters connected via USB on this machine;
- **khomp log console**: Sets options in the console logs;
- **khomp log disk**: Sets logging options to disk;
    - **khomp log console** and **khomp log disk** have auxiliary options **No**, which reverses the selection of messages, and **just**, which generalizes the choice. Examples:
        - **khomp log disk just commands events** (Enables **only** logging to the disk of commands and events);
        - **khomp log disk no commands** (Disable logging to the disk of commands sent to board);
        - **khomp log disk warnings** (Enables also logging to the disk of warnings from Endpoint).
    - More information on options for the **log** command, type: "**help khomp log disk**" or "**help console log khomp**".
- **khomp log rotate**: Rotate log files from the Endpoint;
- **khomp log status**: Shows log messages currently being written to disk and displayed on the console;
- **khomp log trace isdn**: Enable Debugging ISDN signaling;
- **khomp log trace k3l** : Enables debugging low-level API K3L;
- **khomp log trace r2** : Enables debugging low-level signaling R2/MFC;
- **khomp reset links**: Sends a reset command for a specific **E1** of a particular *card*;
- **khomp revision**: Shows version number and revision of the Endpoint;
- **khomp select sim**: Select the SIM card, available on the boards KGSM;
- **khomp send command** : Sends command API K3L directly to the board (only for debugging, may compromise the stability of the system if used improperly);
- **khomp send raw command** : Sends a command directly to the DSP board (only for debugging, may compromise the stability of the system if used improperly);
- **khomp set** : Sets various options of the Endpoint Khomp;
- **khomp show calls** : Shows states for calls, may also listing specific channels or boards;
- **khomp show channels** : Shows the status of the channels Khomp and may also list specific adapter;
- **khomp show links**: Display states of E1 links available.
- **khomp show statistics** : Shows the statistics of channel connections, or statistics for a particular channel;
- **khomp sms** : Send an SMS message for a given number, using KGSM channels;
- **khomp summary** : Prints a summary of system boards and their features;

Additional features

This chapter discusses additional features of the Endpoint, related to the special features present in certain signs.

Aplicações (applications) e canais

The Endpoint Khomp, and to record a type of communication channel "Khomp" also records the following items:

"KUserTransfer" application

Performs the transfer process from the current channel *number* for the extension using the signaling protocol QSig (Single Step Call Transfer) for boards configured with E1 ISDN signaling (ISDN), or use the FLASH command for FXO.

The syntax follows:

```
 <action application="KUserTransfer" data="number[,options])"/>
```

Example:

```
<action application="answer"/>
<action application="KUserTransfer" data="2345"/>
```

The fields have the following meanings:

- **number**: Number where the link should be transferred.
- **options**: Sets the transfer options to be used, which are:
    - **N**: Wait until the channel is disconnected.

Application "KSendSMS"

This application has the function of sending SMS messages through the boards of KGSM Khomp using modules and SIM cards in the board to do so. The syntax of the application is as follows:

```
<action application="KSendSMS" data="resource | destination | message" />
```

Each field can be summarized in:

- **resource**: The following is a syntax identical to the allocation of channels Dial application, and defines what modem use;
- **destination**: Number where to send the message, may be preceded or succeeded by**!** to request a confirmation of transmission;

- **message**: Text (without quotes) to be sent to**destination**.

After sending the message, the variables **KSmsDelivered** and **KSmsErrorCode** will contain the result of the post. For more information about these, please consult the section on variables used in the Endpoint.

Examples of use of this application are as follows:

- Sends "Test message." phone for "99887766" using the modem "1" (second modem) card "0":

```
<action application="log" data="DEBUG Sending SMS ..." />
<action application="KSendSMS" data="b0c1|99887766|Test message" />
```

- Sends "Test message." phone for "99887766" using the first free modem card "0", and checks the return shipment:

```
<action application="log" data="DEBUG  Sending SMS ..." />
<action application="KSendSMS" data="b0|99887766|Test message" />
<action application="log" data="DEBUG Sent? ${KSmsDelivered}" />
<action application="log" data="DEBUG Code: ${KSmsErrorCode}" />
<action application="log" data="DEBUG Desc: ${KSmsErrorName}" />
```

- Sends "Test message." phone for "99887766" using the first free modem card "0", or the first free channel board "1" (if no free channel at the first sign):

```
<action application="log" data="DEBUG Sending SMS ..." />
<action application="KSendSMS" data="b0+b1|99887766|Test message" />
```

- Sends "Test message." phone for "99887766" using the first free modem card "0", requesting confirmation:

```
<action application="log" data="DEBUG Sending SMS ..." />
<action application="KSendSMS" data="b0|99887766!|Test message" />
```

Application "KEchoCanceller"

This application has the function to enable or disable the echo canceller channel.

```
<action application="KEchoCanceller" data="action[,options])"/>
```

Where:

- **actions**: It is **on** to enable the echo canceller, and **off** to disable;

Example usage of this application:

```
<action application="KEchoCanceller" data="off"/>
```

Application "KAutoGainControl"

This application has the function to enable or disable the automatic gain control in the channel.

```
<action application="KAutoGainControl" data="action[,options])"/>
```

Where:

- **actions**: It is **on** to enable the automatic gain control, and **off** to disable;

Example usage of this application:

```
  <action application="KAutoGainControl" data="on"/>
```

Application "KDTMFSuppression"

This application has the function to enable or disable the suppression of DTMF channel. The syntax of the application is as follows:

```
<action applicatin="KDTMFSuppression" value="action[,options])"/>
```

Where:

- **actions**: It is**on** to enable DTMF suppression, and **off** to disable;

It is important to note that when disabled suppression of DTMF, DTMF tones are passed inband *and* will not be reported to FreeSWITCH. Thus FreeSWITCH does not recognize the DTMF tones, which may result in malfunction of applications such as IVR.

Example usage of this application:

```
<action applicatin="KDTMFSuppression" value="off"/>
```

Application "KSetVolume"

This application has the function to adjust the volume of incoming and outgoing channels Khomp, and its syntax as follows:

```
<action application="KSetVolume" data="<volume>"/>
<action application="KSetVolume" data="<output-volume>|<input-volume>"/>
```

Where the fields have the following meanings:

- **volume**: Sets the volume of input and output (-10 to +10);
- **output-volume**: Sets the output volume (-10 to +10, "none" for no change);
- **input-volume**: Sets the input level (-10 to +10, "none" for no change).

Application "KAdjustForFax"

This application has the function of setting a channel for receiving Khomp signal FAX/modem, optimizing the communication channel for data traffic. Syntax:

```
<action application="KAdjustForFax" data=""/>
```

This application does not receive parameters. Example of use:

```
<action application="KAdjustForFax" data=""/>
```

Application "KSendFax"

This application has the function to send faxes using digital channels or FXO connections Khomp in pre-established, and its syntax as follows:

```
<action application="KSendFax" data="<file>[:<file2>[:...]][|<faxid>]"/>
```

This application requires a license purchased separately to be used in digital (non-FXO) channels. Fields have the following meanings:

- **file**: Files to be sent to the fax should be encapsulated in TIFF format and have a resolution of 98, 196 or 392 dpi;
- **faxid**: the fax number. If not specified, the value is obtained by the id of the link, and if this also is not valid, the fax number will be set as default in K3L.

Example usage of this application:

```
<action application="KSendFax" data="/tmp/fax.tif:/home/root/fax2.tif,1234"/>
```

Application "KReceiveFax"

This application has the function of receiving digital channels or fax using the FXO Khomp, and its syntax as follows:

```
<action application="KReceiveFax" data="<file>[|<faxid>]/>
```

This application requires a license purchased separately to be used in digital (non-FXO) channels. Fields have the following meanings:

- **file**: Name that file will be assigned to incoming fax.
- **faxid**: the fax number. If not specified, the value is obtained by the id of the link, and if this also is not valid, the fax number will be set as default in K3L.

Example usage of this application:

```
<action application="answer" />
<action application="KReceiveFax" data="/tmp/fax.tif"/>
```

Channel "Khomp_SMS"

This communication channel is used to receive SMS and create incoming links in FreeSWITCH for each message received. This channel does not have any treatment or audio processing, and is called with five variables set:

- **KSmsFrom**, containing the number of source who sent the message;
- **KSmsDate**, which sets the date/time of receipt of the message;
- **KSmsSize**, representing the message size (in bytes);
- **KSmsMode**, containing the encoding used to transmit the message;
- **KSmsBody**, that is the message itself.

The FreeSWITCH dialplan processing can be used to store this message in a database, run any application, among others. However, the only action accepted by this channel is shutdown (Hangup), so this incoming call should be considered a special dialplan execution without audio streams or channel allocation.

Channel "Khomp_PR"

This communication channel is used to receive calls on boards passive recording (**KPR family** and **KFXO-HI**), creating incoming links on FreeSWITCH for each incoming call. This channel allows only receiving audio captured the *link*by not allowing both the transmission of audio signalings as the control.

The FreeSWITCH dialplan processing can be used to record data on this link in a database, perform some special application and/or some recording application (such as **record**), among others. However, the only action accepted by this channel is shutdown (Hangup), so this should not be considered a common call.

Codes and meanings

This chapter presents the codes present in the channel Khomp and their meanings, used in both events as in the AMI console commands:

Channel state

Reflect the state of the channel on the board. In the case of E1 links, the state may have one or more of the following:

- **Free**: the channel is free;
- **Busy**: the channel is not free (or occupied, or failure);
- **Outgoing**: the channel has an output connection;
- **Incoming**: the channel has an input connection;
- **Locked**: the channel is blocked;
- **Outgoing Lock**: The channel is blocked for outgoing calls;
- **Local Fail**: The channel has a fault (at this point);
- **Incoming Lock**: the channel is blocked for incoming calls;
- **Remote Lock**: there is a remote lock (at the other end) in this channel.

In the case of a FXS channel, the state is defined by one of these values:

- **On Hook**: the phone connected to this channel is on-hook or disconnected;
- **Off Hook**: the phone connected to this channel is off the hook;
- **Ringing**: the channel is being called;
- **Failure**: the channel is in failure due to communication problems between the central and the plate.

In the case of a GSM channel, the state is defined by one of the following values:

- **Idle**: the channel is free and available for calls;
- **Call In Progress**: the channel is busy on a call;
- **SMS In Progress**: the channel is busy sending / receiving SMS messages;
- **Modem Error**: an error occurred communicating with the modem channel;
- **SIM Card Error**: The SIM card is not present or is not inserted / detected correctly;
- **Network Error**: an error occurred while communicating with the network;
- **Not Ready**: The modem is initializing the channel.

And in the case of an FXO channel, the states are as follows:

- **Disabled**: the channel is disabled;
- **Enabled**: the channel is enabled.

Call state

Defines the logical state for each channel, which can be:

- **Free**: the channel is free;
- **Incoming**: the channel is receiving a call;
- **Outgoing**: the channel is making a call;
- **Failure**: the channel is in fault.

FreeSWITCH call states

Directly reflects the call state controlled by FreeSWITCH, which can be:

- **new**: Channel is newly created;
- **init**: Channel has been initialized;
- **routing**: Channel is looking for an extension to execute;
- **execute**: Channel is executing its dialplan;
- **ex_media**: Channel is exchanging media with another channel;
- **cs_media**: Channel is consuming all media;
- **hangup**: Channel is flagged for hangup and ready to end.

GSM Codes

The following numeric codes are reported:

SMS codes (SMS causes)

```
1       Unassigned number
8       Operator determined barring
10      Call barred
21      SMS transfer rejected
27      Destination out of service
28      Unidentified subscriber
29      Facility rejected
30      Unknown subscriber
38      Network out of order
41      Temporary failure
42      Congestion
47      Resources unavailable
50      Facility not subscribed
```

69	Facility not implemented
81	Invalid SMS transfer reference value
95	Invalid message
96	Invalid mandatory information
97	Message type non existent
98	Message not compatible with SMS protection state
99	Information element non existent
111	Protocol error
127	Interworking
128	Telematic interworking not supported
129	SMS type zero not supported
130	Cannot replace SMS
143	Unspecified TPPID error
144	Alphabet not supported
145	Message class not supported
159	Unspecified TPDCS error
160	Command cannot be actioned
161	Command unsupported
175	Unspecified TP command error
176	TPDU not supported
192	SC busy
193	No SC subscription
194	SC system failure
195	Invalid SME address
196	Destination SME barred
197	SM rejected duplicate SM
198	TPVPF not supported
199	TPVP not supported
208	SIM SMS storage full
209	No SMS storage capability in SIM
210	Error in SMS
211	Memory capatity exceeded
213	SIM data download error
255	Unspecified error
300	Phone failure
301	SMS service reserved
302	Operation not allowed
303	Operation not supported
304	Invalid PDU mode parameter
305	Invalid text mode parameter
310	SIM not inserted
311	SIM PIN necessary
312	Phone SIM PIN necessary
313	SIM failure
314	SIM busy
315	SIM wrong
320	Memory failure
321	Invalid memory index
322	Memory full
330	SMSC address unknown
331	No network service
332	Network timeout
500	Unknown error
512	Network busy
513	Invalid destination address
514	Invalid message body length
515	Phone is not in service
516	Invalid preferred memory storage
517	User terminated

Call codes (call causes)

1	Unallocated number
3	No route to destination
6	Channel unacceptable
8	Operator determined barring
16	Normal call clear
17	User busy
18	No user responding
19	No answer from user
21	Call rejected
22	Number changed
26	Non Selected user clear
27	Destination out of order
28	Invalid number format
29	Facility rejected
30	Response status enquiry
31	Normal, unspecified
34	No circuit channel available
38	Network out of order
41	Temporary failure
42	Switch congestion
43	Access information discarded
44	Requested channel unavailable
47	Resource unavailable
49	QoS unavailable
50	Request facility not subscribed
55	Call barred with UG
57	Bearer capability not authorized
58	Bearer capability not available
63	Service not available
65	Bearer capability not implemented
69	Request facility not implemented
70	Only restricted bearer capability available
79	Service not implemented
81	Invalid call reference value
82	User not member of UG
88	Incompatible destination
91	Invalid transit network selected
95	Invalid message
96	Missing mandatory information element
97	Message type not implemented
98	Message incompatible with state
99	Information element not implemented
100	Invalid information element
101	Message incompatible with state (2)
102	Recovery on timer expiry
111	Protocol error
127	Interworking

General codes (mobile causes)

0	Phone failure
1	No connection to phone
2	Phone adaptor link reserved
3	Operation not allowed
4	Operation not supported

5	Phone SIM PIN required
6	Phone FSIM PIN required
7	Phone FSIM PUK required
10	SIM not inserted
11	SIM PIN required
12	SIM PUK required
13	SIM failure
14	SIM busy
15	SIM wrong
16	Incorrect password
17	SIM PIN2 required
18	SIM PUK2 required
20	Memory full
21	Invalid index
22	Not found
23	Memory failure
24	Text string too long
25	Invalid character in text string
26	Dial string too long
27	Invalid character in dial string
30	No network service
31	Network timeout
32	Network not allowed
33	Command aborted
34	Number parameter instead of text parameter
35	Text parameter instead of number parameter
36	Numeric parameter out of bounds
37	Text string too short
40	Network PIN required
41	Network PUK required
42	Network subset PIN required
43	Network subset PUK required
44	Network service provider PIN required
45	Network service provider PUK required
46	Corporate PIN required
47	Corporate PUK required
60	SIM Service option not supported
100	Unknown
103	Illegal MS #3
106	Illegal MS #6
107	GPRS service not allowed #7
111	PLMN not allowed #11
112	Location area not allowed #12
113	Roaming not allowed #13
132	Service option not supported #32
133	Registration service option not subscribed #33
134	Service option temporary out of order #34
147	Long context activation
148	Unspecified GPRS error
149	PDP authentication failure
150	Invalid mobile class
151	GPRS disconnection TMR active
256	Too many active calls
257	Call rejected
258	Unanswered call pending
259	Unknown calling error
260	No phone number recognized
261	Call state not idle
262	Call in progress
263	Dial state error

264	Unlock code required
265	Network busy
266	Invalid phone number
267	Number entry already started
268	Cancelled by user
269	Number entry could not be started
280	Data lost
281	Invalid message body length
282	Inactive socket
283	Socket already open

Troubleshooting

In this section, errors and their most common solutions are presented.

Error during installation of kernel module

During installation of the *Endpoint*of Khomp may occur the following messages:

```
K3L: WARNING: Unable to find a module for [...]
```

or

```
install: ******  THE KERNEL MODULE HAS NOT BEEN INSTALLED: *******
install:
install: ** Please, untar the file kpdriver*.tar.gz located in: **
install: **                 '/usr/src/khomp/'                   **
install: **             then check the README.txt               **
install: **  for knowing how to proceed with the installation.  **
```

In this case, you must compile the drivers manually to your system. Proceed to the item below for more information.

Compiling the drivers and starting the services

Just follow to the directory **/usr/src/khomp**, unpack the file "kpdriver_2.0.0**XX**.tar.gz", and follow procedures described in the file **README_en.txt**.

After performing compilation and installation of the module, just load it into the system, configuring the boards and start the server processes Khomp.

To load the kernel driver, you must run the following command:

```
# /etc/init.d/khompdrv start
```

To set up the boards, in turn, must run the command:

```
# khompwizard
```

This will run a setup wizard that will ask the signaling used in the system, as well as other parameters of use of the boards.

If necessary configure other additional parameters you can use the following command:

```
# k3lconfig
```

This configurator, in turn, shows all possible options for card configuration. The parameters that are not configured automatically assume the default values, and are compatible with most systems. More details about this program can be obtained from the section number '2 '.

- **ATTENTION**: To start FreeSWITCH®, it is necessary that the Khomp board is configured and all modules are running (as shown above).

  If you want to run the system without the Khomp board, you need to configure FreeSWITCH for it does not load the module Khomp. To do this, open the **/usr/local/freeswitch/conf/autoload_configs/modules.conf.xml** and comment the line:

```
<!-- <load module="mod_khomp" /> -->
```

When the board Khomp is properly configured and loaded modules khomp (explained above), remember to uncomment this line in the file.

Finally, to load the server process, simply run the following command:

```
# kserver start
```

After performing these procedures, the Endpoint is already operational, and FreeSWITCH can now be loaded.

Setting up special parameters for audio or signaling

To set specific parameters of timing and/or signaling, you can use the program "k3lconfig": simply select the card you want, and options of the boards will be presented, divided into sections and subsections for easy access. It is not necessary to effect the configuration of all parameters: default values are assumed, if not configured.

To adjust the signaling link, simply - after selecting the card - enter the "Options signaling, and then in" Signs of the line. " To choose a particular signaling, just use the direction keys (arrows) to select it, press 'space', and confirm the option by pressing 'Enter' on the "Confirm" button.

Finally, to save the modified settings, just exit the program: it will display a window with options to save the changes made or not.

Please not that **the following options are required to be unchanged from the defaults**:

- Automatic echo cancellation;
- DTMF suppression;
- Automatic Gain Control (AGC).

These options are **controlled by Endpoint** and should be **disabled** in 'k3lconfig' (the default configuration).

Automatic load of services and kernel modules

If the loading of *kernel* module or Khomp services startup is not performed automatically at startup, you can perform this installation manually, creating a *link* for the *scripts* **/etc/init.d/khompdrv** and **/etc/init.d/kserver** in the system startup directory.

In the case of **Debian**-based distributions the *script* for loading *kernel* modules would be *linked* within the directory */etc/rcS.d*, while the services loader would be *linked* within the directories */etc/rc2.d*, */etc/rc3.d*, */etc/rc4.d*, */etc/rc5.d* as follows:

```
# ln -s /etc/init.d/khompdrv  /etc/rcS.d/S19khompdrv
# ln -s /etc/init.d/kserver   /etc/rc2.d/S20kserver
# ln -s /etc/init.d/kserver   /etc/rc3.d/S20kserver
# ln -s /etc/init.d/kserver   /etc/rc4.d/S20kserver
# ln -s /etc/init.d/kserver   /etc/rc5.d/S20kserver
```

Please check the rules of your distribution to initialize the services in accordance with what is expected by the start of it.

Apêndice

This section presents useful informations about Endpoint and related components.

Arrangement of installed files

The directories created/modified in this facility are:

```
/etc/init.d/                 -- Startup scripts;

/etc/khomp/                  -- Firmware files and settings;

/usr/local/freeswitch/conf/ -- Settings for FreeSWITCH and Endpoint;

/usr/doc/khomp/              -- Docs for the boards, Channel, and utilities;

/usr/sbin/                   -- Utilities and server processes;

/usr/lib/                    -- Shared libraries;

/usr/local/freeswitch/mod/  -- Endpoint module;

/var/log/khomp2.1/           -- Log directory for K3L and Endpoint;
```

The script **/etc/init.d/khompdrv** is responsible for loading modules **kpci9030.ko** and **kpex8311.ko** in the kernel, which should be carried out automatically at system startup. In case of problems, check the Troubleshooting section.

For more details: http://www.khomp.com.br