

Contents

- 1 SHORT BLUEPRINT: STEPS NEEDED TO USE GSMOPEN
- 2 UNLOCK THE DONGLE!!!
- 3 What Is GSMOpen
- 4 STATUS
- 5 Hardware Requirements
 - ◆ 5.1 Voice calls and SMS
 - ◆ 5.2 SMS ONLY (no voice calls)
- 6 Dialplan, and how to use GSMOpen for outbound voice calls
 - ◆ 6.1 Dialplan
 - ◆ 6.2 The "ANY" and "RR" interfaces, poor man interface grouping
- 7 CONFIGURATION FILE and incoming voice calls
 - ◆ 7.1 Incoming voice calls
 - ◆ 7.2 Multiple concurrent incoming calls to the same GSM number
- 8 API and CLI Commands
 - ◆ 8.1 integration with mod_sms
 - ◆ 8.2 gsm
 - ◆ 8.3 gsmopen
 - ◆ 8.4 gsmopen boost audio
 - ◆ 8.5 gsmopen_dump
 - ◆ 8.6 gsmopen_sendsms
 - ◆ 8.7 chat
- 9 Events
 - ◆ 9.1 Voice Calls
 - ◆ 9.2 MESSAGE (SMSs)
 - ◇ 9.2.1 integration with mod_sms
 - ◆ 9.3 gsmopen::dump_event
 - ◆ 9.4 gsmopen::alarm
- 10 MULTIPLE LINES HARDWARE SETUP
- 11 TO DO
- 12 KNOWN ISSUES
- 13 Building
 - ◆ 13.1 Linux, *BSD, etc
 - ◇ 13.1.1 Which Linux distro? Desktop or Server?
 - ◇ 13.1.2 Prerequisites
 - 13.1.2.1 PREREQUISITES CentOS 5.x
 - 13.1.2.2 PREREQUISITES CentOS 6.x, RHEL6.x, Scientific 6 Server 64 bit
 - 13.1.2.3 PREREQUISITES Ubuntu LTS 12.04 Server 64 bit
 - 13.1.2.4 PREREQUISITES Debian 6 (Squeeze) Server 64 bit
 - ◇ 13.1.3 Build and Install
 - ◇ 13.1.4 Configuration File
 - ◇ 13.1.5 Start FS and Load GSMOpen
 - ◆ 13.2 WINDOWS
 - ◇ 13.2.1 Prerequisites on Windows

GSMopen

- [13.2.1.1 Build FreeSWITCH on Windows](#)
- [13.2.1.2 PREREQUISITES GSMopen on Windows](#)
- [13.2.1.3 Build and Install GSMopen on Windows](#)
- ◇ [13.2.2 Configuration File on Windows](#)
- ◇ [13.2.3 Start FS and Load GSMopen on Windows](#)
- [14 How to Report Bugs and Feature Requests](#)
- [15 How to Find Help](#)
- [16 UNLOCK THE DONGLE!!!](#)
- [17 TROUBLESHOOTING](#)
- [18 Group Buy on Huawei E169's](#)

SHORT BLUEPRINT: STEPS NEEDED TO USE GSMOPEN

- Compile and install FreeSWITCH
- Compile and install the prerequisites (gsmlib and libctb)
- Compile and install mod_gsmopen
- Install and edit mod_gsmopen config file
- Connect one or more Huawei USB dongles (or other GSM modems) to the FreeSWITCH server machine
- Be sure the **dongle has the "voice" capability unlocked**, or unlock it with dc-unlocker (<http://www.dc-unlocker.com/>).
- Start FreeSWITCH
- Load mod_gsmopen in FreeSWITCH
- Profit!

UNLOCK THE DONGLE!!!

Be sure the **dongle has the "voice" capability unlocked**, or unlock it with dc-unlocker (<http://www.dc-unlocker.com/>).

What Is GSMopen

GSMopen is in "beta" status.

GSMopen is an endpoint (channel driver) that allows an SMS to be sent to and from FreeSWITCH as well as incoming and outgoing GSM voice calls (that can be bridged, originated, answered, etc. as in all other endpoints, e.g. sofia/SIP). An SMS on FreeSWITCH is handled following the CHAT API (like the text messaging in mod_sofia, mod_skypopen, and mod_dingaling).

Preferred GSM modem to use for voice calls (and SMSs) is Huawei E1550 dongle, or compatible.

GSMopen works in FreeSWITCH on Linux and Windows, native at 8khz (GSM is 8khz compressed audio). It probably works on *BSD and OSX too.

GSMopen

GSMopen operates on GSM cellular/mobile connections in the same way that OpenZAP operates on analog lines. One interface (a GSM modem) is needed for each channel. For example, two concurrent calls would need two channels as well as two USB GSM dongles connected to the FreeSWITCH server.

Obviously you must have credit on the SIM(s) inside the modems to make and receive calls, just like you need credits for your regular cell phone to work.

GSMopen uses *very* low CPU, so it works with the less powerful server platforms without problems (e.g. embedded appliances).

GSMopen has been contributed to the community by: Giovanni Maruzzelli (gmaruzz al-t gmail dot com) with a lot of help from the core developer's team, and hints, patches, suggestions, bug reports, features requests from the superlative FS community.

GSMOpen is fully integrated with http://wiki.freeswitch.org/wiki/Mod_sms so you can use it for your messaging system pleasure.

STATUS

GSMopen is in "beta" status.

But 'it works for me' and in various installations, without problems.

Preferred GSM modem to use with GSMopen is Huawei E1550 dongle, or compatible.

Hardware Requirements

The CPU load generated by the GSMopen endpoint is very low, so if a server is able to run FS, it will have no problems using GSMopen channels.

You will need at least one physical interface to connect to a GSM network.

Voice calls and SMS

Preferred GSM modem to use with GSMopen is Huawei E1550 dongle, or compatible.

MORE THAN ONE DEVICE on an USB bus without dedicated power supply can have **intermittent failure**, because dongles can use all of the power! If you use more than one device, use an external (or many, cascaded), powered from the wall, USB hub. That's perfectly OK.

If unsure, please check that the **dongle has the "voice" capability and that "voice" capability is unlocked**. To check the capability existence and status, and unlock it if needed, you can use dc-unlocker client for windows (<http://www.dc-unlocker.com/> , client and checking are free, small fee to unlock).

Compatibility list (please add to this list, probably most dongles are compatible):

GSMopen

- E1550
- E1552
- E169
- E1692
- E171
- E173
- E175X
- E1752C
- E1762
- E180
- K3520
- K3715
- K3765

Look for "Huawei modem" on Ebay or similar sites. Eg:

<http://www.aliexpress.com/wholesale?isFreeShip=y&SearchText=huawei%2Be1550&CatId=0&manual=y>

Anyway, if other brands or models have the basic USB interfaces (serial + audio) but requires different modem commands, I will add support to them. Open a Jira for it, or contact me directly.

Using more than one dongle *REQUIRES*** the usage of an (or many, cascaded) external powered USB hub** (so it can feed power to the dongles, 500mA/.5W each one).

SMS ONLY (no voice calls)

- Any GSM modem (secondhand cellphone or a professional modem) that accepts AT commands + its datacable to connect it to the server.

Dialplan, and how to use GSMopen for outbound voice calls

Dialplan

Like other endpoints it's easy to build up useful dialplans using GSMopen.

You can use the standard format with the interface name:

```
gsmopen/interface1/3472665618
```

To call the number "3472665618" using the gsmopen interface named "interface1"

Dialplan snippet:

```
<!-- dial 3472665618 via gsmopen using interface1 interface to go out -->
<extension name="gsmopen">
  <condition field="destination_number" expression="^2909$">
    <action application="bridge" data="gsmopen/interface1/3472665618"/>
  </condition>
```

```
</extension>
```

The "ANY" and "RR" interfaces, poor man interface grouping

You can also use the "ANY" or "RR" interfaces

```
gsmopen/ANY/3472665618
gsmopen/RR/3472665618
```

To call "3472665618" using the first available (idle, not in a call) gsmopen interface, automatically selected (thx Seven Du).

"ANY" and "RR" are now just aliases, and will choose an available idle interface based on a round robin algorithm (so to distribute calls more fairly between all the available interfaces).

Dialplan snippet:

```
<!-- dial 3472665618 via gsmopen RR interface -->
<extension name="gsmopen">
  <condition field="destination_number" expression="^2908$">
    <action application="bridge" data="gsmopen/RR/3472665618"/>
  </condition>
</extension>
```

CONFIGURATION FILE and incoming voice calls

GSMOpen is **very** configurable.

Almost any single AT command used to manage callflow and to understand signaling and status can be customized.

There are default values for all values, so you can leave the configuration file almost empty (you lazy!).

```
<configuration name="gsmopen.conf" description="GSMOpen Configuration">
  <global_settings>
    <param name="debug" value="8"/>
    <param name="dialplan" value="XML"/>
    <param name="context" value="default"/>
    <param name="hold-music" value="$$${moh_uri}"/>
    <param name="destination" value="9999"/>
  </global_settings>
  <!-- one entry here per gsmopen interface -->
  <per_interface_settings>
    <interface id="1" name="interface0">
      <param name="hold-music" value="$$${moh_uri}"/>
      <param name="dialplan" value="XML"/>
      <param name="context" value="default"/>
      <param name="destination" value="5000"/>
      <param name="controldevice_name" value="/dev/ttyUSB3"/>
      <param name="controldevice_audio_name" value="/dev/ttyUSB2"/>
    </interface>
    <interface id="3" name="interfaceNICE">
      <param name="hold-music" value="$$${moh_uri}"/>
    </interface>
  </per_interface_settings>
</configuration>
```

GSMOpen

```
<param name="dialplan" value="XML"/>
<param name="context" value="default"/>
<param name="destination" value="9996"/>
<param name="controldevice_name" value="/dev/ttyUSB7"/>
<param name="controldevice_audio_name" value="/dev/ttyUSB6"/>
</interface>
</per_interface_settings>
</configuration>
```

Following are all the various configurable parameters you can set for each interface (with their default values):

```
context = "default"
dialplan = "XML"
destination = "5000"
controldevice_name = NULL
controldevice_audio_name = NULL
digit_timeout = NULL
max_digits = NULL
hotline = NULL
dial_regex = NULL
hold_music = NULL
fail_dial_regex = NULL
enable_callerid = "true"
at_dial_pre_number = "ATD"
at_dial_post_number = ";"
at_dial_expect = "OK"
at_hangup = "ATH"
at_hangup_expect = "OK"
at_answer = "ATA"
at_answer_expect = "OK"
at_send_dtmf = "AT+VTS"
at_preinit_1 = ""
at_preinit_1_expect = ""
at_preinit_2 = ""
at_preinit_2_expect = ""
at_preinit_3 = ""
at_preinit_3_expect = ""
at_preinit_4 = ""
at_preinit_4_expect = ""
at_preinit_5 = ""
at_preinit_5_expect = ""
at_postinit_1 = "at+cmic=0,9"
at_postinit_1_expect = "OK"
at_postinit_2 = "AT+CKPD=\"EEE\""
at_postinit_2_expect = "OK"
at_postinit_3 = "AT+CSSN=1,0"
at_postinit_3_expect = "OK"
at_postinit_4 = "at+sidet=0"
at_postinit_4_expect = "OK"
at_postinit_5 = "at+clvl=99"
at_postinit_5_expect = "OK"
at_query_battchg = "AT+CBC"
at_query_battchg_expect = "OK"
at_query_signal = "AT+CSQ"
at_query_signal_expect = "OK"
at_call_idle = "+MCST: 1"
at_call_incoming = "+MCST: 2"
at_call_active = "+CSSI: 7"
at_call_failed = "+MCST: 65"
```

GSMOpen

```
at_call_calling = "+CSSI: 1"
at_indicator_noservice_string = "CIEV: 20"
at_indicator_nosignal_string = "CIEV: 50"
at_indicator_lowsignal_string = "CIEV: 5;1"
at_indicator_lowbattchg_string = "CIEV: 0;1"
at_indicator_nobattchg_string = "CIEV: 0;0"
at_indicator_callactive_string = "CIEV: 3;1"
at_indicator_nocallactive_string = "CIEV: 3;0"
at_indicator_nocallsetup_string = "CIEV: 6;0"
at_indicator_callsetupincoming_string = "CIEV: 6;1"
at_indicator_callsetupoutgoing_string = "CIEV: 6;2"
at_indicator_callsetupremoteringing_string = "CIEV: 6;3"
alsaname = "plughw:1"
alsaname = "plughw:1"
at_early_audio = "0"
at_after_preinit_pause = "500000"
at_initial_pause = "500000"
at_has_clcc = "0"
at_has_ecam = "0"
alsa_period_size = "160"
alsa_periods_in_buffer = "4"
gsmopen_sound_rate = "8000"
alsa_play_is_mono = "1"
alsa_capture_is_mono = "1"
capture_boost = "5"
playback_boost = "10"
no_sound = "0"
portaudiocindex = "1"
portaudiopindex = "1"
speexecho = "1"
speexpreprocess = "1"
gsmopen_serial_sync_period = "300"
```

Incoming voice calls

Each incoming voice call that arrives on the interface will be directed to the *destination* extension in the *context* context.

So, please edit or add those fields to the config file to adapt it to your needs (the default config file works with the default, out-of-the-box, demo FreeSWITCH dialplan).

Multiple concurrent incoming calls to the same GSM number

This is possible, if carrier support call forwarding. You must set up call forwarding on BUSY, NOT REACHABLE state. Remember to switch off CALL WAITING.

Each physical interface (eg: GSM modem) has its own SIM, with just one number. You must have one interface for each concurrent call.

When a call is made by a remote party to a number, the carrier sends the call to the SIM that bears that number. If SIM is busy or unreachable, carrier will redirect call to forwarded number. Same way you can add additional SIM cards / phones - forward them.

GSMOpen

Carrier can limit maximal call forwarding chain length. If you experience that situation, please add that information here.

API and CLI Commands

GSMOpen adds various commands to the standard FreeSWITCH API and Commands.

They can all be used both through the command line and via API/socket/ESL/whatever.

integration with mod_sms

GSMOpen is fully integrated with http://wiki.freeswitch.org/wiki/Mod_sms so you can use it for your messaging system pleasure.

gsm

"gsm" commands are intended to be used from the FS command line ("gsm remove" and "gsm reload" can be useful from Event socket as well).

You begin typing "gsm console interface_name" to direct the "current console" to sending messages to interface_name. Starting now on, you can type "gsm AT_command" and AT_command string will be sent to the modem related to interface interface_name.

```
gsm console interface1
gsm ATI
```

"gsm list" gives the list and status of all the running GSMOpen interfaces (a star marks the interface from which "RR" - see below - will start hunting an IDLE one), statistics about inbound failed and total calls, outbound failed and total calls per each interface.

```
gsm list
```

Muhammad Shahzad and Seven Du contributed code for adding and removing interfaces on the fly.

remove

```
gsm remove <interface_name | interface_id>
```

This command remove the gsmopen interface with name interface_name or with id interface_id, if that interface is idle.

```
gsm remove interface1
```

reload

```
gsm reload
```

GSMopen

This command re-reads GSMopen configuration file `gsmopen.conf.xml` and adds ONLY the non running interfaces it found in that file. All existing running interfaces are not affected.

gsmopen

"gsm remove" and "gsm reload" (see before) can be useful from API/socket/ESL/whatever as well.

"gsmopen" commands are intended to be used by programs (API/socket/ESL/whatever) and have the format: "gsmopen interface_name AT_command_string". They send the AT_command string to the modem related to interface interface_name.

```
gsmopen interface2 ATI
```

This allow you to use directly the entire power of the AT command set of your GSM modem or cellphone, for eg. to prototype a new feature, do customization, etc etc. Typing "console loglevel 9" at the FS command line allows you to see the AT answers from the GSM modem.

gsmopen_boost_audio

This command affects the volumes of incoming and outbound audio at the sample level, in code. This command is may be useful to interactively (trial and error during a call) find the best audio boost setting for your setup, then you write the found values in the config file. Boost can be for playback or for capture, and can be negative or positive (expressed in decibel units). Syntax is:

```
gsmopen_boost_audio interface_name [<play|capt> <in decibels: -40 ... 0 ... +40>]
```

eg:

```
gsmopen_boost_audio interface3 play -10
```

The example will lower by 10 decibels the volume of the playback in interface3

gsmopen_dump

This command generates (fires) a CUSTOM event of type `gsmopen::dump_event` that reports a lot of useful information about the interface interface_name.

If interface_name is "list", `gsmopen_dump` will fire as many events as the number of running interfaces, one for each of them.

```
gsmopen_dump interface1
```

Or,

```
gsmopen_dump list
```

For the event description, see below, **Events**.

gsmopen_sendsms

It supports full utf8 SMS text, although the FS command line only accepts ASCII. Please use ESL or API to send utf8 text.

```
gsmopen_sendsms interface_name destination_number SMS_text
```

eg:

```
gsmopen_sendsms interface1 3472665618 this is a nice SMS text
```

chat

GSMopen answers to the FreeSWITCH standard "chat" command too, and uses its arguments to execute a gsmopen_sendsms command. So, if you have a messaging application that uses the chat command with Sofia/SIP or Jingle, no need to recode it with special cases for SMS messages :).

It uses SMS as protocol specification. e.g., from command line:

```
chat SMS|interface3|3472665618|ciao amore
```

Events

GSMopen generates (fires) various CUSTOM events in addition to the standard FreeSWITCH events on voice calls (like the other endpoints) and MESSAGE (chat) events on incoming SMSs (like Sofia and Jingle).

Voice Calls

Standard CODEC and CHANNEL_* events.

See [Event list](#)

MESSAGE (SMSs)

The Event type generated by an incoming SMS is of type MESSAGE (like in Jingle and Sofia).

The interesting fields are:

```
login:                the interface name that received the SMS
from:                 the sender's number, urlencoded
date:                 the date of the received message, urlencoded
datacodingscheme:    which kind of alphabet was used to send the message
servicecentreadress: address of SC used to send the message
messagetype:         numeric, usually 0, kind of message
during-call:         bool, the message was received while a voice call was ongoing?
```

And obviously the body, encoded in utf8, that contains the SMS's text.

GSMopen

NB: The body is UTF8 encoded, gives you ASCII for ASCII, and UTF8 for all the rest.

This is a session telnetting to the Events port (8021), asking for authorization, asking for events of type message in plain format, then an incoming SMS as reported with Events plain.

```
telnet 127.0.0.1 8021
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
Content-Type: auth/request

auth ClueCon

Content-Type: command/reply
Reply-Text: +OK accepted

events plain message

Content-Type: command/reply
Reply-Text: +OK event listener enabled plain

Content-Length: 888
Content-Type: text/event-plain

Event-Name: MESSAGE
Core-UUID: 3ebb22ce-0b58-11e2-9147-f53de47b3be1
FreeSWITCH-Hostname: vz139.gmaruzz.com
FreeSWITCH-Switchname: vz139.gmaruzz.com
FreeSWITCH-IPv4: 192.168.1.139
FreeSWITCH-IPv6: %3A%3A1
Event-Date-Local: 2012-10-03%2013%3A51%3A06
Event-Date-GMT: Wed,%2003%20Oct%202012%2011%3A51%3A06%20GMT
Event-Date-Timestamp: 1349265066811330
Event-Calling-File: mod_gsmopen.cpp
Event-Calling-Function: sms_incoming
Event-Calling-Line-Number: 3099
Event-Sequence: 29207
proto: sms
login: gsm01
from: %2B393472665618
date: 10/03/2012%2001%3A51%3A04%20PM%20(%2B0200)
datacodingscheme: default%20alphabet
servicecentreaddress: %2B393916263333
messagetype: 0
subject: SIMPLE%20MESSAGE
to: gsm01
hint: gsm01
to_proto: sms
from_user: %2B393472665618
to_user: gsm01
max_forwards: 70
DP_MATCH: gsm01
skip_global_process: true
dest_proto: GLOBAL
Delivery-Failure: true
Content-Length: 5

Test
```

MESSAGE (SMSs)

GSMOpen

integration with mod_sms

GSMOpen is fully integrated with http://wiki.freeswitch.org/wiki/Mod_sms so you can use it for your messaging system pleasure.

gsmopen::dump_event

CUSTOM events of type `gsmopen::dump_event` are fired in response to a `gsmopen_dump` command or API call (e.g., from command line or via script or through the ESL). This event reports a lot of useful information on the state of the interface which name was given as argument to the command (if that name is "list" the command will fire as many `gsmopen::dump_event` as interfaces are running, one for each of them).

This is an example of the `gsmopen::dump_event` fired in response to:

```
gsmopen_dump interface2001
```

Event:

```
Content-Length: 990
Content-Type: text/event-plain

Event-Subclass: gsmopen%3A%3Adump_event
Event-Name: CUSTOM
Core-UUID: 28d9e2e2-068d-11df-8f99-e9d7ea2264f4
FreeSWITCH-Hostname: hardy64
FreeSWITCH-IPv4: 192.168.0.12
FreeSWITCH-IPv6: %3A%3A1
Event-Date-Local: 2010-01-24%2017%3A51%3A02
Event-Date-GMT: Sun,%2024%20Jan%202010%2016%3A51%3A02%20GMT
Event-Date-Timestamp: 1264351862152937
Event-Calling-File: mod_gsmopen.cpp
Event-Calling-Function: dump_event_full
Event-Calling-Line-Number: 3008
interface_name: interface2001
interface_id: 1
active: 1
not_registered: 0
home_network_registered: 1
roaming_registered: 0
got_signal: 2
running: 1
imei: 353579017208923
imsi: 222018302196169
controldev_dead: 0
controldevice_name: /dev/ttyACM2
no_sound: 0
alsaname: plughw%3A2
alsaname: plughw%3A2
playback_boost: 1619.086162
capture_boost: 910.479058
dialplan: XML
context: default
destination: 2000
ib_calls: 0
ob_calls: 0
ib_failed_calls: 0
ob_failed_calls: 0
```

GSMOpen

```
interface_state: 0
phone_callflow: 0
session_uuid_str: _undef_
during-call: false
```

During a call (while a call is active on the interface) a lot of useful infos are added (courtesy of Math ;)).

Command used to generate the event is the same, but the interface is in an active call, executing a javascript app:

```
Content-Length: 2399
Content-Type: text/event-plain

Event-Subclass: gsmopen%3A%3Adump_event
Event-Name: CUSTOM
Core-UUID: 28d9e2e2-068d-11df-8f99-e9d7ea2264f4
FreeSWITCH-Hostname: hardy64
FreeSWITCH-IPv4: 192.168.0.12
FreeSWITCH-IPv6: %3A%3A1
Event-Date-Local: 2010-01-24%2017%3A55%3A40
Event-Date-GMT: Sun,%2024%20Jan%202010%2016%3A55%3A40%20GMT
Event-Date-Timestamp: 1264352140851653
Event-Calling-File: mod_gsmopen.cpp
Event-Calling-Function: dump_event_full
Event-Calling-Line-Number: 3008
interface_name: interface2001
interface_id: 1
active: 1
not_registered: 0
home_network_registered: 1
roaming_registered: 0
got_signal: 2
running: 1
imei: 353579017208923
imsi: 222018302196169
controldev_dead: 0
controldevice_name: /dev/ttyACM2
no_sound: 0
alsaname: plughw%3A2
alsaname: plughw%3A2
playback_boost: 1619.086162
capture_boost: 910.479058
dialplan: XML
context: default
destination: 2000
ib_calls: 1
ob_calls: 0
ib_failed_calls: 0
ob_failed_calls: 0
interface_state: 2
phone_callflow: 5
session_uuid_str: 45fbda50-0909-11df-8f99-e9d7ea2264f4
during-call: true
Channel-State: CS_EXECUTE
Channel-State-Number: 4
Channel-Name: gsmopen/interface2001
Unique-ID: 45fbda50-0909-11df-8f99-e9d7ea2264f4
Call-Direction: inbound
Presence-Call-Direction: inbound
Answer-State: answered
Channel-Read-Codec-Name: L16
```

gsmopen::dump_event

GSMOpen

```
Channel-Read-Codec-Rate: 8000
Channel-Write-Codec-Name: L16
Channel-Write-Codec-Rate: 8000
Caller-Username: gsmopen
Caller-Dialplan: XML
Caller-Caller-ID-Name: GSMOpen%3A%20%2B393472665618
Caller-Caller-ID-Number: %2B393472665618
Caller-Destination-Number: 2000
Caller-Unique-ID: 45fbda50-0909-11df-8f99-e9d7ea2264f4
Caller-Source: mod_gsmopen
Caller-Context: default
Caller-Channel-Name: gsmopen/interface2001
Caller-Profile-Index: 1
Caller-Profile-Created-Time: 1264352127793104
Caller-Channel-Created-Time: 1264352127793104
Caller-Channel-Answered-Time: 1264352133281836
Caller-Channel-Progress-Time: 1264352127803152
Caller-Channel-Progress-Media-Time: 0
Caller-Channel-Hangup-Time: 0
Caller-Channel-Transfer-Time: 0
Caller-Screen-Bit: true
Caller-Privacy-Hide-Name: false
Caller-Privacy-Hide-Number: false
variable_read_codec: L16
variable_read_rate: 8000
variable_write_codec: L16
variable_write_rate: 8000
variable_channel_name: gsmopen/interface2001
variable_endpoint_disposition: ANSWER
variable_instance_id: 100
variable_current_application_data: freedomfone/leave_message/main.js%20100
variable_current_application: javascript
```

gsmopen::alarm

CUSTOM events of subtype `gsmopen::alarm` are automatically fired when something bad happens to an interface, usually resulting in the interface being unavailable for service.

Most interesting fields are:

<code>alarm_code</code>	integer	refers to the type of alarm
<code>alarm_message</code>	string	descriptive text

`alarm_code` possible values are defined below:

```
0 ALARM_FAILED_INTERFACE
1 ALARM_NO_NETWORK_REGISTRATION
2 ALARM_ROAMING_NETWORK_REGISTRATION
3 ALARM_NETWORK_NO_SERVICE
4 ALARM_NETWORK_NO_SIGNAL
5 ALARM_NETWORK_LOW_SIGNAL
```

Other fields as in the `dump_event` event. During a call, the alarm event also gets the additional fields.

This is an example of an alarm event for an interface that fails to initialize at startup (because the physical serial port does not exist):

`gsmopen::alarm`

GSMopen

```
Content-Length: 1023
Content-Type: text/event-plain

Event-Subclass: gsmopen%3A%3Aalarm
Event-Name: CUSTOM
Core-UUID: 28d9e2e2-068d-11df-8f99-e9d7ea2264f4
FreeSWITCH-Hostname: hardy64
FreeSWITCH-IPv4: 192.168.0.12
FreeSWITCH-IPv6: %3A%3A1
Event-Date-Local: 2010-01-24%2017%3A38%3A15
Event-Date-GMT: Sun,%2024%20Jan%202010%2016%3A38%3A15%20GMT
Event-Date-Timestamp: 1264351095120533
Event-Calling-File: mod_gsmopen.cpp
Event-Calling-Function: dump_event_full
Event-Calling-Line-Number: 3005
alarm_code: 0
alarm_message: gsmopen_serial_init%20failed
interface_name: interface4001
interface_id: 2
active: 0
not_registered: 0
home_network_registered: 0
roaming_registered: 0
got_signal: 0
running: 0
imei: _undef_
imsi: _undef_
controldev_dead: 0
controldevice_name: /dev/ttyACM0
no_sound: 0
alsaname: plughw%3A1
alsaname: plughw%3A1
playback_boost: 10.000000
capture_boost: 5.000000
dialplan: XML
context: default
destination: 2000
ib_calls: 0
ob_calls: 0
ib_failed_calls: 0
ob_failed_calls: 0
interface_state: 0
phone_callflow: 0
session_uuid_str: _undef_
during-call: false
```

MULTIPLE LINES HARDWARE SETUP

Because of limitations in the USB "regular" hubs, if you want to connect many USB modems to an USB hub it is of paramount importance that you use "Powered" USB 2.0 (or 3.0) Hubs, eg: the hubs that got a power supply to be plugged in the wall socket.

TO DO

Requests, suggestions, ideas (feel free to add here, but best is to add on Jira - see below 'BUGS and Feature Requests'):

KNOWN ISSUES

Be sure the **dongle has the "voice" capability unlocked**, or unlock it with dc-unlocker (<http://www.dc-unlocker.com/>).

Building

Linux, *BSD, etc

Which Linux distro? Desktop or Server?

Desktop operating systems and distros are completely unsupported.

If you want to use desktop operating systems you have to find the way yourself, sorry.

Only supported operating systems by Gsmopen are 64 bit servers: LTS ubuntu 12.04, centos 6, SL6, Debian 6, running directly on the hardware (eg: no Virtual Machines), or in OpenVZ containers running on Debian 6 or Centos 6

Prerequisites

PREREQUISITES CentOS 5.x

CentOS 5.x **do NOT WORKS** with huaweis and mod_gsmopen, **use CentOS 6.x**

PREREQUISITES CentOS 6.x, RHEL6.x, Scientific 6 Server 64 bit

Before building GSMopen module do (despite the name of the directory, works well particularly for CentOS ;)):

```
cd gsmlib/gsmlib-1.10-patched-13ubuntu
./configure
make
make install
ldconfig
```

Check if the library is added by running **ldconfig -p | grep gsm**

GSMopen

If you don't see any records, correct it after reading this post <http://linux.101hacks.com/unix/ldconfig/> e.g. adding new rule file in /etc/ld.so.d/. Then re-run **ldconfig**

then

```
cd /usr/src/freeswitch/src/mod/endpoints/mod_gsmopen/libctb-0.16/build
make DEBUG=0 GPIB=0
make DEBUG=0 GPIB=0 install
ldconfig
```

PREREQUISITES Ubuntu LTS 12.04 Server 64 bit

Before building GSMopen module do:

```
apt-get install gsm-utils
apt-get install libgsmme-dev
```

then

```
cd /usr/src/freeswitch/src/mod/endpoints/mod_gsmopen/libctb-0.16/build
make DEBUG=0 GPIB=0
make DEBUG=0 GPIB=0 install
ldconfig
```

PREREQUISITES Debian 6 (Squeeze) Server 64 bit

Before building GSMopen module do:

```
apt-get install gsm-utils
apt-get install libgsmme-dev
apt-get install usb-modeswitch-data usb-modeswitch
```

then

```
cd /usr/src/freeswitch/src/mod/endpoints/mod_gsmopen/libctb-0.16/build
make DEBUG=0 GPIB=0
make DEBUG=0 GPIB=0 install
ldconfig
```

You may need to reboot to have your dongle recognized

Build and Install

After installing prerequisites (see before), go into mod_gsmopen directory and type:

```
cd /usr/src/freeswitch/src/mod/endpoints/mod_gsmopen/
make clean
make install
```

GSMopen

Configuration File

Install and edit the gsmopen configuration file:

```
cd /usr/src/freeswitch/trunk/src/mod/endpoints/mod_gsmopen/configs/  
cp gsmopen.conf.xml /usr/local/freeswitch/conf/autoload_configs/  
vi /usr/local/freeswitch/conf/autoload_configs/gsmopen.conf.xml
```

Start FS and Load GSMopen

Launch FreeSWITCH:

```
/usr/local/freeswitch/bin/freeswitch
```

Then activate debug logging in console and logfile, and load mod_gsmopen:

```
freeswitch@machine> console loglevel 9  
freeswitch@machine> fsctl loglevel 9  
freeswitch@machine> load mod_gsmopen
```

WINDOWS

GSMopen runs very well on Windows.

GSMopen (mod_gsmopen) is NOT automatically built when you build FreeSWITCH on Windows.

Prerequisites on Windows

Build FreeSWITCH on Windows

You will need the Visual C compiler from Microsoft, commercial version, or the free (as in beer) Visual C Express (requires registration). They both give the same results in our case (eg: no need to buy the commercial version just for GSMopen).

After having downloaded the FS sources from svn or the packaged FS source release, follow the instruction on how to build FS on Windows. Using Visual C (Express or not):

- Open Freeswitch.sln
- Right click the main solution node at the top of the Solution Explorer
- Right click and select Build

This will build FreeSWITCH WITHOUT GSMopen. You must now build the prerequisites (see below) and after prerequisites are built, eventually build mod_gsmopen.

GSMOpen

PREREQUISITES GSMOpen on Windows

After having built FreeSWITCH, go into FreeSWITCH source directory, eg: c:\freeswitch, and then go to src/mod_gsmopen/gsmllib/gsmllib-1.10-patched-13ubuntu/win32 directory.

Using Visual C (Express or not):

- Open gsmllib.sln (Note: if you are using 2008 pro or higher or 2010 pro or higher this step is not needed see below)
- Right click the main solution node at the top of the Solution Explorer
- Right click and select Build

ERRORS ARE OK!, we're only interested in building the library, and it will be built ok. Errors come from application building, we're not interested in application.

Then you must start a Visual Studio Command Prompt (from the "Visual Studio Tools" Start Menu). Inside the Command Prompt window go to

```
c:/freeswitch/src/mod/endpoints/mod_gsmopen/libctb-0.16/build
```

Inside the Command Prompt window execute:

```
nmake -f makefile.vc DEBUG=1 GPIB=0  
nmake -f makefile.vc DEBUG=0 GPIB=0
```

Note: If you have Visual Studio 2008 Pro or 2010 Pro you must execute the above nmake command then build mod_gsmopen (gsmllib will be built automatically as a dependency).

Build and Install GSMOpen on Windows

Go back to the Visual C compiler from Microsoft, commercial version, or the free (as in beer) Visual C Express (requires registration). They both give the same results in our case (eg: no need to buy the commercial version just for GSMOpen).

- Open Freeswitch.sln
- Right click the main solution node at the top of the Solution Explorer
- Click on "Add" and choose "Existing Project" (Note: If you have VS2008 Pro or higher or VS2010 Pro or higher this step is not needed)
- Navigate to c:/freeswitch/src/mod/endpoints/mod_gsmopen/ and select "mod_gsmopen.2008.vcproj"
- mod_gsmopen is added to the FreeSWITCH solution tree
- Right click on "mod_gsmopen" and choose "Build"

Configuration File on Windows

Install and edit the gsmopen configuration file:

```
copy c:/freeswitch/src/mod/endpoints/mod_gsmopen/configs/gsmopen.conf.xml c:/freeswitch/Debug/conf  
notepad c:/freeswitch/Debug/conf/autoload_configs/gsmopen.conf.xml
```

Start FS and Load GSMOpen on Windows

Launch FreeSWITCH:

```
c:/freeswitch/Debug/freeswitch.exe
```

Then activate debug logging in console and logfile, and load mod_gsmopen:

```
freeswitch@machine> console loglevel 9
freeswitch@machine> fsctl loglevel 9
freeswitch@machine> load mod_gsmopen
```

How to Report Bugs and Feature Requests

Be sure the **dongle has the "voice" capability unlocked**, or unlock it with dc-unlocker (<http://www.dc-unlocker.com/>).

You can file bug reports, hints, suggestions, feature requests, improvements, patches, etc to <http://jira.freeswitch.org> open an account there if you don't have it (it's free ;-).

That's the best way to give us info on bugs:

- 1) from the FS CLI: "console loglevel 9"
- 2) from the FS CLI: "fsctl loglevel 9"
- 3) from the FS CLI: "unload mod_gsmopen"
- 4) from the FS CLI: "load mod_gsmopen"
- 5) reproduce the bug
- 6) attach the **complete, since beginning** console output (or freeswitch.log file) **as a file attachment** to the Jira bug.
Please do not cut and paste console output or freeswitch.log in the Jira message. Attach it.

If the bug involves crashes, core dumps, etc, please read this guide on how to report it http://wiki.freeswitch.org/wiki/Reporting_Bugs , then file a Jira to mod_gsmopen with all relevant info.

How to Find Help

Be sure the **dongle has the "voice" capability unlocked**, or unlock it with dc-unlocker (<http://www.dc-unlocker.com/>).

You can drop in the IRC channels #freeswitch on irc.freenode.net to ask questions and discuss issues. The original developer of GSMOpen is called "gmaruzz" in the IRC channel.

You can also write to the FS users' and developers' mailing lists: <http://lists.freeswitch.org/mailman/listinfo>

UNLOCK THE DONGLE!!!

Be sure the **dongle has the "voice" capability unlocked**, or unlock it with dc-unlocker (<http://www.dc-unlocker.com/>).

TROUBLESHOOTING

MORE THAN ONE DEVICE on an USB bus without dedicated power supply can have **intermittent failure**, because dongles can use all of the power! If you use more than one device, use an external (or many, cascaded), powered from the wall, USB hub. That's perfectly OK.

Group Buy on Huawei E169's

There are some members in the community that would like to buy E169's (mostly because of the external antenna option). Because the sellers on Alibaba sell those (but only if you buy a minimum of X) I (bdfoster) would like to go buy them from Alibaba and distribute them (for the actual cost each, no 'markup'). **If you are interested** send an email to bdfoster@davri.com . If there are 20 people interested at \$25-\$30, let me know, and the more people sign up the cheaper they become.

US/Canada only. Buyer responsible for shipping costs. Void where prohibited. Must be 18 years or older. Payments accepted via PayPal only. I can't guarantee you will be able to use this but it has been tested to work with mod_gsmopen. I cannot do your research for you. If you have a suggestion for a different dongle send an email to the above address with your suggestion and why it would work out better. Payments will need to be made in two stages. The first stage will be for the actual product, the second will be for the shipping costs specifically to your address. .