

# Contents

- [1 General](#)
- [2 Conference configuration](#)
  - ◆ [2.1 <advertise>](#)
  - ◆ [2.2 <caller-controls>](#)
  - ◆ [2.3 <chat-permissions>](#)
  - ◆ [2.4 <profiles>](#)
    - ◇ [2.4.1 Conference Parameters](#)
- [3 Conference Dialplan Application](#)
  - ◆ [3.1 Syntax](#)
  - ◆ [3.2 Examples](#)
    - ◇ [3.2.1 Adding Callers To The Conference With DTMFs](#)
    - ◇ [3.2.2 Announcing Caller Count While In Conference](#)
- [4 Conference set auto outcall Dialplan Application](#)
  - ◆ [4.1 Syntax](#)
  - ◆ [4.2 Example](#)
- [5 Application Reference](#)
- [6 API Reference](#)
- [7 Event Socket Use](#)
- [8 Channel variables set by mod\\_conference](#)
  - ◆ [8.1 conference\\_last\\_matching\\_digits](#)
  - ◆ [8.2 conference\\_member\\_id](#)
  - ◆ [8.3 conference\\_moderator](#)
  - ◆ [8.4 conference\\_name](#)
  - ◆ [8.5 conference\\_recording](#)
  - ◆ [8.6 conference\\_uuid](#)
  - ◆ [8.7 last\\_transferred\\_conference](#)
- [9 Channel variables to control mod\\_conference behavior](#)
  - ◆ [9.1 conference\\_auto\\_outcall\\_announce](#)
  - ◆ [9.2 conference\\_auto\\_outcall\\_caller\\_id\\_name](#)
  - ◆ [9.3 conference\\_auto\\_outcall\\_caller\\_id\\_number](#)
  - ◆ [9.4 conference\\_auto\\_outcall\\_flags](#)
  - ◆ [9.5 conference\\_auto\\_outcall\\_maxwait](#)
  - ◆ [9.6 conference\\_auto\\_outcall\\_prefix](#)
  - ◆ [9.7 conference\\_auto\\_outcall\\_profile](#)
  - ◆ [9.8 conference\\_auto\\_outcall\\_timeout](#)
  - ◆ [9.9 conference\\_controls](#)
  - ◆ [9.10 conference\\_enforce\\_security](#)
  - ◆ [9.11 conference\\_enter\\_sound](#)
  - ◆ [9.12 conference\\_exit\\_sound](#)
- [10 Sound files](#)
- [11 FAQ](#)
- [12 See Also](#)

## General

The mod\_conference module provides multi-user conference features. You can create as many conferences as you like, as long as there still are free system resources (i.e. memory, CPU power) left.

Conferences are created according to configuration within the dialplan.

Conferences created in the dialplan use profiles that are defined in conf/autoload\_configs/conference.conf.xml, if you are using the standard configuration with mod\_dialplan\_xml.

Community member Stanislav Sinyagin has posted a nice [personal blog entry](#) on how he got his conference bridge all configured.

## Conference configuration

The configuration has the following structure, each "[... config here ...]" should be replaced with a configuration.

```
<configuration name="conference.conf" description="Audio Conference">
  <advertise>
    [... config here ...]
  </advertise>

  <caller-controls>
    <group name="default">
      [... config here ...]
    </group>
  </caller-controls>

  <profiles>
    <profile name="default">
      [... config here ...]
    </profile>
  </profiles>

  <chat-permissions>
    <profile name="default">
      [... config here ...]
    </profile>
  </chat-permissions>
</configuration>
```

### <advertise>

Specifies whether changes in the conference (as leaving and entering) should be advertised... The following information will be advertised to a Jabber/XMPP server.

(someone please complete this list)

- Enter - Someone entered the conference.
- Leave - someone left the conference.

## Mod\_conference

- Mute - Someone muted/unmuted the conference.

### Example configuration

```
<advertise>
  <room name="888@${subdomain}" status="FreeSWITCH"/>
</advertise>
```

The name (888 in this case) should be the profile name that you specified in <profiles> section, \${subdomain} will be replaced with the subdomain that you specified in freeswitch.xml.

"status" is advertised as whatever you pass to it (identifier) or "Available" if none is passed.

### Example 'advertise' Event via mod\_event\_multicast

```
proto: conf
login: 888@example.com
from: 888@example.com
status: FreeSWITCH
rpid: idle
event_type: presence
Event-Name: PRESENCE_IN
Core-UUID: c76e2d7d-39d7-dc11-93bf-0090fb0792c6
FreeSWITCH-Hostname: example.com
FreeSWITCH-IPv4: 192.168.1.5
FreeSWITCH-IPv6: 127.0.0.1
Event-Date-Local: 2008-02-09 13:04:44
Event-Date-GMT: Sat, 09 Feb 2008 18:04:44 GMT
Event-Date-timestamp: 1202580284348009
Event-Calling-File: mod_conference.c
Event-Calling-Function: send_presence
Event-Calling-Line-Number: 5037
Multicast-Sender: example.com
```

### <caller-controls>

Caller controls are used to modify the state of the conference. It can be lowering the volume, mute the conference and such. Below are the possible commands that can be assigned to digits and executed during a conference... NOTE: As of Oct 3 2011 (1936c2b) , there is a new "moderator-controls" parameter. See below.

### Actions

- mute - Turn on/off sound in the conference.
- deaf mute - performs both a mute and a deaf in one action
- energy up - increase background energy threshold
- energy equ - reset background energy threshold
- energy dn - decrease background energy threshold
- vol talk up - Increase your mic volume
- vol talk zero - Change back to default setting
- vol talk dn - Decrease your mic volume
- vol listen up - Increase your speaker volume
- vol listen zero - Switch back to default setting
- vol listen dn - Decrease your speaker volume
- hangup - Leave the conference

```
<advertise>
```

## Mod\_conference

- lock - Toggle the conference locked/unlocked (no new members can enter when locked)
- event - Send the dtmf event via CUSTOM conference::maintenance subclass to the event system (even to event socket)
- transfer - transfer caller to a given extension dialplan context
- execute\_application - executes a dialplan application

### Reserved Group Names

- none - Use this name to prevent installing caller-controls for callers of a conference.
- default - Use this name to utilize the hard-coded set of controls built-in to mod\_conference. Do NOT name a custom set of conference-controls "default" as they will be overridden with the hard-coded set. The behavior of the "default" group is defined below:

```
<group name="default">
  <control action="vol talk dn"      digits="1"/>
  <control action="vol talk zero"    digits="2"/>
  <control action="vol talk up"      digits="3"/>
  <control action="vol listen dn"    digits="4"/>
  <control action="vol listen zero"  digits="5"/>
  <control action="vol listen up"    digits="6"/>
  <control action="energy dn"        digits="7"/>
  <control action="energy equ"       digits="8"/>
  <control action="energy up"        digits="9"/>
  <control action="mute"             digits="0"/>
  <control action="deaf mute"        digits="*/>
  <control action="hangup"           digits="#"/>
</group>
```

### Example Configuration

```
<caller-controls>
  <group name="somekeys">
    <control action="vol talk dn"      digits="1"/>
    <control action="vol talk zero"    digits="2"/>
    <control action="vol talk up"      digits="3"/>
    <control action="transfer"         digits="5" data="100 XML default"/>
    <control action="execute_application" digits="0" data="playback conf_help.wav"/>
    <control action="execute_application" digits="#" data="execute_dialplan conference-menu"/>
  </group>
</caller-controls>
```

### Limitations

Be aware that the caller-controls are applied across the entire conference. You cannot enter one member of the conference using caller-controls ABC and then enter a second member using caller-controls XYZ.

### <chat-permissions>

A conference can be controlled through Jabber/XMPP. This profile exists as part of the conference entity so you can have multiple profiles to limit access. When the conference publishes its presence (above) to a Jabber server, anyone on a federated server can talk to it and issue commands by typing messages via their favorite Jabber client (even if that is another FreeSWITCH™ box). To send commands to ConferenceA, from your

```
<caller-controls>
```

## Mod\_conference

jabber client send a message to conf+ConferenceA@freeswitch.mydomain.com

Note: As of revision 3789 all commands except list have been disabled "until there is security"

Another Note: After spending much time trying to get control of mod\_conference via Openfire XMPP server, asked around, and "bougyman" suggested that XMPP muc functionality is very limited at present (January 2012). So watch this space.

Here is an example configuration

```
<configuration name="conference.conf" description="Audio Conference">
  <profiles>
    <profile name="default">
      <param name="chat-permissions" value="default"/>
    </profile>
  </profiles>
  <chat-permissions>
    <profile name="default">
      <!-- both of these users have a functionally equivalent capability set -->
      <user name="bob@somewhere.com" commands="all"/>
      <!-- individually specified commands must be book-end'ed with the | character -->
      <user name="harry@somewhere.com" commands="|deaf|dial|energy|kick|list|lock|mute|norecord|pla
    </profile>
  </chat-permissions>
</configuration>
```

### <profiles>

You can specify a number of different profiles in the profiles section, these will let you apply a number of settings to a conference. Please note that the profiles are not conference rooms, but settings applied to conference rooms. The dialplan section in this document will describe how you create conference rooms and apply these settings.

The profiles section has this structure:

```
<profiles>
  <profile name="default">
    <param name="paramName" value="paramValue"/>
  </profile>
</profiles>
```

You can have any number of <profile> tags, and each <profile> can have any number of <param> tags (as long as there are parameters to configure ;-).

### Conference Parameters

The parameters that you can specify are the ones below. Keep in mind that if TTS is enabled all audio-file params beginning with 'say:' will be considered text to say with TTS.

Name	Description	Example value
announce-count		5

<chat-permissions>

## Mod\_conference

	Requires TTS. The system will speak the total number of callers in the conference when a new person joins, but only once the threshold specified in this parameter is reached.	
auto-gain-level	Enables Automatic Gain Control (AGC). If the parameter is set to true, then the default AGC value is used, otherwise if set to a number it will override the default.	1100
auto-record	Put a value in for this parameter in order to toggle recording every conference call. Within mod_conference there is a special parameter named <code>\${conference_name}</code> . All channel variables are accessible as well. <b>NOTE: auto-record doesn't begin recording until two parties are on the line.</b>	<p><b>Example 1:</b> /var/myNFSshare/\${conference_name}_\${strftime(%Y-%m-%d)}</p> <p><b>Example 2:</b> shout://user:pass@server.com/live.mp3</p>
caller-controls	Name of the caller control group to use for this profile. One of the ones you specified in <caller-controls>. The profile named "default" will be used if none is specified.	somekeys
moderator-controls	Name of the moderator control group to use for this profile. One of the ones you specified in <caller-controls>. The profile named "default" will be used if none is specified.	somekeys
cdr-log-dir	Target directory for conference auto CDRs to be written. Use "auto" to store in \$PREFIX/log/conference_cdr. An absolute path is acceptable as is a relative path. A relative path will yield \$PREFIX/log/<value> for the conference CDR directory.	
caller-id-name	Default Caller ID Name for outbound calls	FreeSWITCH
caller-id-number	Default Caller ID Number for outbound calls	8777423583
comfort-noise		1-10000, "true" (1400), -1 (total silence)

## Mod\_conference

	Sets the volume level of background white noise to add to the conference.	
conference-flags	Can be any combination of <b>wait-mod</b> , <b>audio-always</b> , <b>video-bridge</b> and <b>video-floor-only</b> . <b>wait-mod</b> is used to force conference participants to wait (with music) for a moderator/leader to enter the conference. The moderator is determined by setting the <b>moderator</b> member-flag from the dialplan where the call is sent to the conference. This might be used in conjunction with an IVR where the moderators are authenticated with an extra pass-code. By default it only mix audios of members who are considered "talking", <b>audio-always</b> will always mix audios of all members.	wait-mod
domain	???	\$\$ {domain}
energy-level	Energy level required for audio to be sent to the other users. The energy level is a threshold of 'voice energy' that must be present before audio is bridged into the conference. Useful if a participant is in a noisy environment, so you only hear background noise when they speak. 0 disables the detection and will bridge all packets even if they are only background noise.	300
interval	Number of milliseconds per frame. Which may be different from ptime in SIP SDP, or driver with TDM. Higher numbers require less CPU but can cause conversation quality issues, so experimentation with your setup is best. The default is good for conversation quality.	20
max-members	Sets a maximum number of	20

## Mod\_conference

	participants in conferences with this profile.	
member-flags	Can be any combination of: <b>mute, deaf, mute-detect, dist-dtmf</b> (dtmf pass through), <b>moderator, nomoh, endconf, mintwo</b> (ends conference if only 1 participant), <b>waste</b> (always bridge). <b>"moderator"</b> should probably not be set within the XML profile, but rather from the dialplan, and should be used in combination with the wait-mod conference-flag. <b>Endconf</b> causes a conference to be torn down when a member with this flag leaves the conference. (If more than one member has the endconf flag set, the conference will be torn down when the last endconf member leaves.)	deaf
pin	Pin code that must be entered before user is allowed to enter the conference.	12345
moderator-pin	Pin code that must be entered before moderator is allowed to enter the conference.	12345
pin-retries	Max number of times the user or moderator can be asked for PIN	3
rate	Sample Rate	8000
sound-prefix	Set a default path here so you can use relative paths in the other sound params.	/soundfiles
suppress-events	For use with the event socket. This parameter specifies which events will NOT be sent to the socket when getting CUSTOM conference::maintenance events.	add-member
terminate-on-silence	If conference is silent for x number of seconds it will automatically terminate and disconnect all callers.	300
timer-name	???	???
tts-engine		cepstral

## Mod\_conference

	Text-To-Speech (TTS) Engine to use	
tts-voice	TTS Voice to use	david
verbose-events	Maximum verbosity for transcribing. Events related to the conference will send more data.	???
ack-sound	File to play to acknowledge success.	beep.wav
alone-sound	File to play if you are alone in the conference	yactopitc.wav
bad-pin-sound	File to play to when the pin is invalid	invalid-pin.wav
enter-sound	File to play when you join the conference.	welcome.wav
exit-sound	File to play when you leave the conference.	exit.wav
is-locked-sound	File to play when the conference is locked during the call to the members in the conference	is-locked.wav
is-unlocked-sound	File to play when the conference is unlocked during the call	is-unlocked.wav
kicked-sound	File to play when you are kicked from the conference.	kicked.wav
locked-sound	File to play when the conference is locked and someone goes to join	locked.wav
max-members-sound	If <b>max-members</b> has been reached, this sound plays instead of allowing new users to the conference.	max-members.wav
moh-sound	the given sound file/resource will be played only when the conference size is 1 member. It will loop over and over until the conference goes to 2+ members where it will stop. When the conference goes back to 1 member it will play again.	idlemusic.mp3
muted-sound	File to play when member is muted.	muted.wav
mute-detect-sound	If the <b>mute-detect</b> member-flag has been set, this sound plays when the user	mute-detect.wav

## Mod\_conference

	talks while muted.	
nack-sound	File to play to acknowledge failure.	beeperr.wav
	??? What is it acknowledging ???	
perpetual-sound	the given sound file/resource will be played on a loop forever. this can be used for some backgroundmusic.	holdmusinc.mp3
pin-sound	Played when the user should enter the conference pin code.	pin.wav
unmuted-sound	File to play when member is unmuted.	unmuted.wav
ivr-dtmf-timeout	How many ms to wait between DTMF digits to match caller-controls.	500
ivr-input-timeout	How many ms to wait for the first DTMF, zero = forever	0
enconf-grace-time	Defines how much time all members have before the conference will be torn down when the last member with <b>endconf</b> leaves. Default = 0.	600

## Conference Dialplan Application

The conference dialplan application is used to create/join conferences and to bind a profile to them.

### Syntax

```
<action application="conference" data="confname@profilename+[conference pin number]+flags{mute|de
```

The first time a conference name (confname) is used, it will be created on demand, and the pin will be set to what ever is specified at that time: the pin in the data string if specified, or if not, the "pin" setting in the conference profile, and if that is also unspecified, then there is no pin protection. Any later attempt to join the conference must specify the same pin number, if one existed when it was created.

profilename should be one of those that you specified in the conference configuration.

If the data value starts with "bridge:", then it is a bridging conference. The conference name should not be already in use. You can specify the special literal value of "\_uuid\_" for the conference name, and a session-specific unique id will be generated for you.

## Mod\_conference

Conferences stay alive until the number of members reaches the minimum number, and then falls below that. The minimum for bridging conferences is 2, and for other dynamically created conferences is 1.

### Examples

Action data	Description
confname	profile is "default", no flags or pin
confname+1234	profile is "default", pin is 1234
confname@profilename+1234	no flags
confname+1234+flags{mutelwaste}	profile is "default", multiple flags with pin
confname++flags{endconflmoderator}	profile is "default", no p.i.n., multiple flags
bridge:confname	a "bridging" conference

\*Note that while some parameters are optional, their order is very important

Or a simple example in an action tag:

```
<action application="conference" data="meeting@mykeys">
```

Bridging conference example that plays ringback while other party is bridged in:

```
<extension name="test_bridging_conference">
  <condition field="destination_number" expression="^(3000)$">
    <action application="answer"/>
    <action application="playback" data="connecting_your_call.wav"/>
    <action application="set" data="ringback=${us-ring}"/>
    <action application="conference" data="bridge:$1-${domain_name}@default:user/1000@${domain_
  </condition>
</extension>
```

### Adding Callers To The Conference With DTMFs

By combining several elements - the dialplan, API calls, `bind_digit_action` - you can create a simple system for a caller to add another user to the conference. Click [here](#) for the code and explanation.

### Announcing Caller Count While In Conference

Click [here](#) for a simple example of how to allow a caller to receive an announcement of how many members are in the conference.

## Conference\_set\_auto\_outcall Dialplan Application

Use `conference_set_auto_outcall` to have `mod_conference` call one or more endpoints when a conference starts. To have it call more than one endpoint, just repeat the `conference_set_auto_outcall` action in the dialplan.

## Syntax

```
<action application="conference_set_auto_outcall" data="dialstring"/>
```

## Example

Here is an example of using **conference\_set\_auto\_outcall** with some of the other `conference_auto_outcall_*` parameters to start a conference when someone dials **12345**. Extensions 1000 and 1001 will be dialed when the conference starts.

```
<extension name="Demonstrate conference_set_auto_outcall">
  <condition field="destination_number" expression="^(12345)$">

    <action application="answer"/>

    <action application="set" data="conference_auto_outcall_timeout=5"/>
    <action application="set" data="conference_auto_outcall_flags=none"/>
    <action application="set" data="conference_auto_outcall_caller_id_name=${effective_caller_id_number}"/>
    <action application="set" data="conference_auto_outcall_caller_id_number=${effective_caller_id_number}"/>
    <action application="set" data="conference_auto_outcall_profile=default"/>

    <action application="conference_set_auto_outcall" data="user/1000@${domain}"/>
    <action application="conference_set_auto_outcall" data="user/1001@${domain}"/>

    <action application="conference" data="$1@default"/>

  </condition>
</extension>
```

---

If you want, you can also autocall multiple destination at once, just remember to escape your variables if you have more than one or any non-escaped chars in it.

```
<action application="conference_set_auto_outcall" data="['var1=a,var2=b']user/1001@${domain}"/>
```

---

## Application Reference

The following command structure can be used from the CLI:

```
conference <conf_name>[@<conf_profile_name>]
conference bridge:<conf_name>[@<conf_profile_name>]|<_uuid>:<originate_url>
```

## API Reference

### deaf

Make a conference member deaf. Usage: conference <confname> deaf <[member\_id|all|last|non\_moderator]>

### dial

Dial a destination via a specific endpoint (ie. call mom from the conference).

Usage: conference <confname> dial [{dial string options}]<endpoint\_module\_name>/<destination>  
[<callerid\_number> [<callerid\_name>]]

See here for the list of dial string options available: [Channel Variables](#)

If the caller id values are not set, the variables in conference.conf.xml will be used. Specifically, the value for caller-id-number will be used for the number and the value for caller-id-name will be used for the name.

According to the issue logged as [MODAPP-104](#) last updated on June 23rd 2008: If the conference will be dynamically created as a result of this api call (ie this will be the first participant in the conference) - and the caller id name and number is not provided in the api call - the number and name will be "00000000" and "FreeSWITCH". This appears to be unaffected by the variables in conference.conf.xml.

```
conference testconf dial {originate_timeout=30}sofia/default/1000@softswitch 1234567890 FreeSWITCH
```

The above api call will dial out of a conference named "testconf" to the user located at the specified endpoint with a 30 second timeout. The endpoint will see the call as coming from "FreeSWITCH\_Conference" with a caller id of 1234567890.

Also worth noting: values provided in the dial string will overwrite the callerid\_number and callerid\_name variables provided at the end of the api call.

```
conference testconf dial {origination_caller_id_name=DialStringName,originate_timeout=30}sofia/d
```

The above api call is almost the same as the previous api call, except the endpoint will see "DialStringName" as the caller id name instead of "FreeSWITCH\_Conference".

### dtmf

Send DTMF to any member of the conference.

Usage: conference <confname> dtmf <[member\_id|all|last|non\_moderator]> <digits>

```
conference foo dtmf all 134
```

### energy

Adjusts the conference energy level for a specific member.

Usage: conference <confname> energy <[member\_id|all|last|non\_moderator]> [<newval>]

### enter\_sound

Changes the sound played while entering the conference

## Mod\_conference

Usage: conference <confname> enter\_sound enter\_sound onlofflnonelfile <filename>

### **exit\_sound**

Changes the sound played while leaving the conference

Usage: conference <confname> exit\_sound enter\_sound onlofflnonelfile <filename>

### **get**

Get runtime parameter of a given conference.

Usage: conference <confname> get <parameter\_name>

Valid <parameter\_name>:

- run\_time (conference time, in seconds)
- count
- max\_members
- rate
- profile\_name
- sound\_prefix
- caller\_id\_name
- caller\_id\_number
- is\_locked
- endconf\_grace\_time (in seconds)

Output: current value of a given parameter:

- for run\_time, count, max\_members, rate, endconf\_grace\_time: a number (converted to string).
- for is\_locked: string "locked" if locked or empty if not.
- for others: string value.

It can be also used in a dialplan:

```
<action application="set" data="conf_runtime=${conference(${conference_name} get run_time)}/>
<action application="set" data="conf_sounddir=${conference(${conference_name} get sound_prefix)}/>
```

### **hup**

Kick without the kick sound

Usage: conference <confname> hup <[member\_id|all|last|non\_moderator]>

### **kick**

Kicks a specific member form a conference.

Usage: conference <confname> kick <[member\_id|all|last|non\_moderator]>

### **list**

Lists all or a specific conference members.

Usage: conference list [delim <string>]

## Mod\_conference

Usage: conference <confname> list [delim <string>]

Output: First line

- Conference <conference name> (<member\_count> member[s][ locked])
  - ◆ ?locked? - The lock/unlock status of the conference.

Following lines are csv separated list for each conference leg with the following items:

- id of participant(starts at 1 after FS restart)
- Register string of participant
- UUID of participants call leg
- Caller id number
- Caller id name
- Status (hear|speak|talking|video|floor)
  - ◆ ?hear? - The mute/unmute status of the member.
  - ◆ ?speak? - The ?deaf /undeaf? status of the member.
  - ◆ ?talking? - The input channel is providing some amount of sound energy.
  - ◆ ?video? - Providing video?
  - ◆ ?floor? - This member currently owns the floor.
- Volume In
- Volume Out
- Energy Level

### lock

Lock a conference so no new members will be allowed to enter.

Usage: conference <confname> lock

### mute

Mutes a specific member in a conference.

Usage: conference <confname> mute <[member\_id|all|last|non\_moderator]>

### nopin

Removes a pin for a specific conference.

Usage: conference <confname> nopin

### norecord

```
conference foo norecord all
```

Or,

```
conference foo norecord /tmp/foo.wav
```

### pause

Usage: conference <confname> pause <file\_path>

## Mod\_conference

### **pin**

Sets or changes a pin number for a specific conference.

Usage: conference <confname> pin <pin#>

Please note that if you set a conference pin and then issue a command like

```
conference <confname> dial sofia/default/123456@softswitch
```

123456 will not be challenged with a pin, but he will just joins the conference named <confname>.

### **play**

Play an audio file in a conference to all members or to a specific member. You can stop that same audio with the Stop command below.

Usage: conference <confname> play <file\_path> [<async>|<member\_id>]

### **record**

```
conference foo record /tmp/foo.wav
```

### **recording**

Alternative syntax for all recording related commands:

Usage:

```
conference <confname> recording start <file_path>
conference <confname> recording check <confname>
conference <confname> recording stop <file_path>|all
conference <confname> recording pause <file_path>
conference <confname> recording resume <file_path>
```

### **relate**

Mute or Deaf a specific member to another member.

Usage: conference <confname> relate <member\_id> <other\_member\_id> [nospeak|nohear|clear]

Examples:

```
conference my_conf relate 1 2 nospeak
```

Member 1 may now no longer speak to member 2, i.e. member 2 now cannot hear member 1.

```
conference my_conf relate 1 2 clear
```

Member 1 may now speak to member 2 again

```
conference my_conf relate 1 2 nohear
```

Member 1 now cannot hear member 2

## Mod\_conference

```
conference my_conf relate 1 2 clear
```

Member 1 can now hear member 2 again

### **resume**

Usage: conference <confname> resume <file\_path>

### **say**

Say text in a conference to all members.

Usage: conference <confname> say <text>

### **saymember**

Say text to a specific member in a conference.

Usage: conference <confname> saymember <member\_id> <text>

### **set**

Set runtime parameter of a given conference.

Usage: conference <confname> set <parameter\_name> <value>

Valid <parameter\_name>:

- max\_members
- sound\_prefix
- caller\_id\_name
- caller\_id\_number
- endconf\_grace\_time

Output: previous value of a given parameter.

It can be also used in a dialplan:

```
<action application="set" data="conf_oldsound=${conference(${conference_name} set sound_prefix $
<action application="set" data="void_result=${conference(${conference_name} set endconf_grace_ti
```

### **stop**

Stops any queued audio from playing.

Usage: conference <confname> stop <[currentlall]> [<member\_id>]

### **transfer**

Transfer a member from one conference to another conference.

Usage: conference <confname> transfer <conference\_name> <member\_id>

To transfer a member to another extension use the [api transfer](#) command with the uuid of their session.

## Mod\_conference

### **undeaf**

Allow a specific member to hear again.

Usage: conference <confname> undeaf <[member\_id|all|last|non\_moderator]>

### **unlock**

Unlock a conference so that new members can enter.

Usage: conference <confname> unlock

### **unmute**

Unmute a specific member of a conference.

Usage: conference <confname> unmute <[member\_id|all|last|non\_moderator]>

### **volume\_in**

Adjusts the input volume for a specific conference member.

Usage: conference <confname> volume\_in <[member\_id|all|last|non\_moderator]> [<newval>]

### **volume\_out**

Adjusts the output volume for a specific conference member.

Usage: conference <confname> volume\_out <[member\_id|all|last|non\_moderator]> [<newval>]

## Event Socket Use

You can subscribe to the following to receive conference events:

```
conference::maintenance
```

The "suppress-events" parameter can be added to the conference profile to prevent events from firing. e.g. if you're not interested in start or stop talking events:

```
<profile name="default">
  ...other options...
  <param name="suppress-events" value="start-talking,stop-talking"/>
</profile>
```

The events that can be suppressed are:

add-member, del-member, energy-level, volume-level, gain-level, dtmf, stop-talking, start-talking, mute-member, unmute-member, kick-member, dtmf-member, energy-level-member, volume-in-member, volume-out-member, play-file, play-file-member, speak-text, speak-text-member, lock, unlock, transfer, bgdial-result and floor-change.

## Channel variables set by mod\_conference

### conference\_last\_matching\_digits

Contains the last matching digits that the user on this channel sent into the conference.

**Usage:**

```
<action application="log" data="INFO Last digits sent by this user: ${conference_last_matching_digits}"/>
```

### conference\_member\_id

Contains the conference\_member\_id value for any conference to which the channel may be connected.

### conference\_moderator

Is true if the channel is connected to a conference as a moderator.

### conference\_name

The name of the last conference the channel joined.

**Usage:**

```
<action application="log" data="INFO Last conference joined by this user: ${conference_name}"/>
```

### conference\_recording

Contains the name of the file conference recording for any conference to which the channel may be connected.

## **conference\_uuid**

Every instance of a conference has its own UUID. This channel variable stores the conference UUID for the most recent conference in which the channel was a member. It is set as soon as the channel enters the conference, and will show up in XML CDRs and uuid\_dump calls, as well as any events that show channel variables.

## **last\_transferred\_conference**

Contains the name of the last conference that this channel was connected to.

### **Usage:**

```
<action application="log" data="INFO Last conference this person visited was [${last_transferred_
```

## **Channel variables to control mod\_conference behavior**

### **conference\_auto\_outcall\_announce**

Audio message to play to conference member joining conference via the conference\_set\_auto\_outcall application.

### **Usage:**

```
<action application="set" data="conference_auto_outcall_announce=sounds/soundfile.wav"/>
```

### **conference\_auto\_outcall\_caller\_id\_name**

Caller ID name to use when dialing endpoints to join the conference via the conference\_set\_auto\_outcall application.

### **Usage:**

```
<action application="set" data="conference_auto_outcall_caller_id_name=${effective_caller_id_nam
```

**conference\_auto\_outcall\_caller\_id\_number**

Caller ID number to use when dialing endpoints to join the conference via the conference\_set\_auto\_outcall application.

**Usage:**

```
<action application="set" data="conference_auto_outcall_caller_id_number=${effective_caller_id_n
```

**conference\_auto\_outcall\_flags**

Conference flags to set for members joining conference via the conference\_set\_auto\_outcall application

**Usage:**

```
<action application="set" data="conference_auto_outcall_flags=mute"/>
```

**conference\_auto\_outcall\_maxwait**

Maximum time in seconds that the channel that initiated the conference\_set\_auto\_outcall will wait for members to join the conference.

**Usage:**

```
<action application="set" data="conference_auto_outcall_maxwait=10"/>
```

**conference\_auto\_outcall\_prefix**

The value of conference\_auto\_outcall\_prefix is prepended to each of conference\_set\_auto\_outcall values, of which there can be more than one.

**Usage:**

```
<extension name="mad_boss_intercom">
  <condition field="destination_number" expression="^0911$">
    <action application="set" data="conference_auto_outcall_caller_id_name=Mad Boss1"/>
    <action application="set" data="conference_auto_outcall_caller_id_number=0911"/>
    <action application="set" data="conference_auto_outcall_timeout=60"/>
    <action application="set" data="conference_auto_outcall_flags=mute"/>
    <action application="set" data="conference_auto_outcall_prefix={sip_auto_answer=true,execu
    <action application="set" data="sip_exclude_contact=${network_addr}"/>
    <action application="conference_set_auto_outcall" data="${group_call(sales) }"/>
    <action application="conference" data="madboss_intercom1@default+flags{endconf|deaf}"/>
```

## Mod\_conference

```
</condition>  
</extension>
```

### **conference\_auto\_outcall\_profile**

Conference profile to use for members joining the conference via the conference\_set\_auto\_outcall application.

#### **Usage:**

```
<action application="set" data="conference_auto_outcall_profile=default"/>
```

### **conference\_auto\_outcall\_timeout**

Originate timeout to use when joining a member to a conference via conference\_set\_auto\_outcall.

#### **Usage:**

```
<action application="set" data="conference_auto_outcall_timeout=60"/>
```

### **conference\_controls**

Set this variable to specify which conference controls profile to use when transferring a caller into a conference. This allows you, for example, to have a profile for the conference moderator and another profile for regular conference members. The profile for the moderator could include the ability to mute/kick people, etc.

NOTE: You must create the conference controls profile. Also, if this is not set then the *default* conference profile is used for the conference member.

#### **Usage:**

```
<action application="set" data="conference_controls=moderator"/>
```

### **conference\_enforce\_security**

Allows the conference security to be overridden. This applies in two different scenarios, one for inbound and one for outbound. By default, conference security is always applied to inbound calls and is always skipped for outbound calls. This channel variable allows the behavior to be modified.

## Mod\_conference

### Usage:

#### Inbound

```
<action application="set" data="conference_enforce_security=false"/>  
<action application="conference" data="3000"/>
```

#### Outbound

```
originate {conference_enforce_security=true}sofia/internal/1001 &conference(3000)
```

### conference\_enter\_sound

When set, this channel variable will override the enter-sound param on conference profile for any conferences into which the call leg is transferred.

### Usage:

```
<action application="set" data="conference_enter_sound=silence_stream://10"/>
```

### conference\_exit\_sound

### Usage:

```
<action application="set" data="conference_exit_sound=silence_stream://10"/>
```

## Sound files

Just about any format is supported, but currently it must be at the sample rate of the conference (no resampling is done). Since disk is cheaper than CPU, use a wav. Asterisk ships with some useful free sound files for conferences.

## FAQ

**Q: Are out-of-band DTMFs propagated to callers?**

## Mod\_conference

By default, out-of-band DTMFs ([RFC 2833?](#)) are absorbed by the conference. However, there are two ways to accomplish this:

- Set the dist-dtmf member flag in the conference configuration XML, eg: `<param name="member-flags" value="dist-dtmf"/>` With this set, all of the caller controls such as modifying voluming will be disabled and DTMF's will simply pass through to all other members of the conference.
- There is also an api call that will allow your application to send DTMF's to any conference member:

```
<confname> dtmf <[member_id|all|last]> <digits>
```

### Q: Does it require hardware, kernel modules, ztdummy, etc?

No.

### Q: How to create a dialplan to have callers speak their name before joining the conference and announce that to other participants when they join?

```
<extension name="Record Name and schedule conf announce">
  <condition field="destination_number" expression="^55(3\d\d\d)$">
    <action application="answer"/>
    <action application="set" data="namefile=/tmp/${uuid}-name.wav" inline="true"/>
    <action application="sleep" data="1000"/>
    <action application="playback" data="voicemail/vm-record_name1.wav"/>
    <action application="playback" data="tone_stream://%(1000,0,500)"/>
    <action application="record" data="${namefile} 1"/>
    <action application="playback" data="ivr/ivr-call_being_transferred.wav"/>
    <action application="set" data="res=${sched_api +1 none conference $1-${domain} play file_st">
    <action application="transfer" data="$1 XML default"/>
  </condition>
</extension>
```

## See Also

- [Conference Add Call Example](#)