Conference Add Call Example

About

Allows a moderator to add a conferee to an established conference.

Discussion

Some folks like the idea that a conference moderator (or any caller with "proper access") should be able dynamically to add a call to a conference. There are many ways to solve this problem. More elaborate methods use scripts.

Presented here is a simple example using only the XML dialplan and some custom items in conference.conf.xml.

This solution employs bind_digit_action instead of conference caller-controls. Keep in mind that the digits assigned in conference.conf.xml for caller-controls will override any digits that you bind. This is the reason that we defined our own caller-controls instead of using the defaults. In our case, we simply remove the * caller-control and then use *1 and *2 for our bind_digit_action sequences.

The targets for bind_digit_actions (BDAs) are simply extensions that get executed. There is an extension for the dialog that asks the moderator for the phone number to call and an extension to remove the most recently added call. For the sake of simplicity, the moderator calls in to *46xx and has the controls enabled and regular callers just call in to 46xx.

We use create_uuid to make a predictable uuid to store in a channel variable and then we use that same uuid value so that we can do uuid_kill if the moderator chooses to disconnect the call. (This feature is useful in the case where you call and get someone’s voicemail or reach an automated attendant that plays music-on-hold forever and want to force a hangup.)

This method uses bgapi and originate to create the outbound call leg. This has the bonus of not blocking the moderator, that is, the moderator is immediately placed back into the conference while FreeSWITCH is dialing out. All the early media stuff is piped into the conference. You can tinker with ign ore_early_media if you wish.

There is no PIN checking or anything like that, however it is possible to add something like that. This is left as an exercise for the reader. If you do choose to create something like that then please add your code to this page and show the community how you accomplished the task.

One important point: after collecting the digits from the moderator you have to decide how you wish to dial the destination. In this case I took a four-digit number to mean that the moderator wants to add a local FS user, and anything more than a four-digit number is assumed to be a phone number to go out a gateway. This method is preferable to using the loopback channel: first, it provides strict control over what happens; second, the loopback channel adds a 3rd call leg which can get confusing, particularly for call detail records and billing.

Configuration

Add these entries to your $$conf_dir/autoload_configs/conference.conf.xml. First, add this group:

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

Then add this profile:
Next add this to your dialplan. In my case I just added this as "${conf_dir}/dialplan/default/01_Conf_Add_Caller.xml":

```
01_Conf_Add_Caller.xml
```
<!-- Sample dialplan for sending member or moderator to a conference where he/she can add/remove calls -->
<!-- Dial 46xx for member, *46xx for moderator -->
<extension name="Simple add-caller conf: member">
  <condition field="destination_number" expression="'(46\d{2})$'">
    <action application="answer"/>
    <action application="conference" data="$1@simple"/>
  </condition>
</extension>

<extension name="Simple add-caller conf: moderator">
  <condition field="destination_number" expression="'(\*46\d{2})$'">
    <action application="answer"/>
    <!-- set up a few bind_digit_action (BDA) bindings for the moderator -->
    <action application="bind_digit_action" data="moderator,*1,exec:execute_extension,ASK_FOR_NUMBER__$1 XML default"/>
    <action application="bind_digit_action" data="moderator,*2,exec:execute_extension,CANCEL_LAST_CALL__$1 XML default"/>
    <action application="digit_action_set_realm" data="moderator"/>
    <action application="conference" data="$1@simple+flags{moderator}"/>
  </condition>
</extension>

<extension name="Add new OB call to conference">
  <condition field="destination_number" expression="'^ASK_FOR_NUMBER__(\d+)$'">
    <!-- ask caller for a number + #, collect into ${target_num} variable -->
    <action application="play_and_get_digits" data="4 11 1 5000 # ivr/ivr-enter_destination_telephone_number.wav ivr/ivr-that_was_an_invalid_entry.wav target_num \d+"/>
    <!-- add this call to the conference -->
    <action application="execute_extension" data="ADD_CALL_TO_CONF__$1@${target_num}"/>
  </condition>
</extension>

<extension name="Remove last OB call added to conference">
  <condition field="destination_number" expression="'^CANCEL_LAST_CALL__(\d+)$'">
    <!-- remove a call from the conference -->
    <action application="play_and_get_digits" data="4 11 1 5000 # ivr/ivr-enter_destination_telephone_number.wav ivr/ivr-that_was_an_invalid_entry.wav target_num \d+"/>
    <action application="set" data="res=${uuid_kill ${hash(select/directory=${domain_name}/last_user_${target_num})}}"/>
  </condition>
</extension>

<extension name="add that call">
  <!-- if we have a four-digit number then attempt to dial it as a user ... -->
  <condition field="destination_number" expression="'^ADD_CALL_TO_CONF__(\d{4})$'" break="on-true">
    <action application="set" data="new_uuid=${create_uuid foo}" inline="true"/>
    <action application="hash" data="insert/domain-${domain_name}/last_user_$1/${new_uuid}"/>
    <action application="set" data="res=${bgapi originate {origination_uuid=${new_uuid}}user/$1 &conference(${conference_name})}"/>
  </condition>
  <!-- if we have a five+ digit number then attempt to dial it as a gw connection ... -->
  <condition field="destination_number" expression="'^ADD_CALL_TO_CONF__(\d{5})$'" break="on-true">
    <action application="set" data="new_uuid=${create_uuid foo}" inline="true"/>
    <action application="hash" data="insert/domain-${domain_name}/last_user_$1/${new_uuid}"/>
    <action application="set" data="res=${bgapi originate {origination_uuid=${new_uuid}}sofia/gateway/mygw /$1 &conference(${conference_name})}"/>
    <!-- Alternatively, you can just use loopback, but that creates three call legs instead of one, so be warned -->
    <action application="set" data="res=${bgapi originate {origination_uuid=${new_uuid}}loopback/$1 &conference(${conference_name})}"/>
  </condition>
</extension>

<!-- ... otherwise inform moderator that the operation was not exactly successful -->
<condition field="destination_number" expression="'^ADD_CALL_TO_CONF__$'">
  <action application="playback" data="ivr/ivr-dude_you_suck.wav"/>
</condition>
</include>
Issue "reloadxml" or press F6 in fscli or the FS console and then you are ready to test.

**Operation**

**Entry**

Regular callers simply dial in (or get transferred to) 46xx. The Moderator dials *46xx. All default caller-controls are available to normal callers and moderators except for the " key which normally toggles deaf/mute. The " key does not respond to normal callers.

**Adding Conferees**

Moderator can dial *1 and then enter a destination phone number + # key. The system will attempt to dial the number and add the call to the conference. The moderator is put back into the conference as soon as the attempt is started.

**Dropping Conferees**

Moderator can dial *2 and the system will ask for a number to hang up. If more than one moderator is in the conference, they each have their own private "most recently added call" list and thus cannot remove another moderator's call. This kind of thing is doable, but is beyond the scope of this example.

**Ringback Issues**

With the above dialplan, local extensions and certain gateways being called from the conference did not send ringback tones back into the conference. There appears to be a way to fake this, but note that you will get a "joined" sound triggered both when the ringback is faked (because the loopback call joined) and when the originate is answered.

With loopback in earlier versions, all calls were squeezed through an L16 channel at 8k samples/s, so although loopback fixes the ringback issues, it meant that every conference member heard SD audio regardless of what the endpoint negotiated (however it should be possible since git HEAD of 4/23/12 to use "loopback_initial_codec=L16@16000h" in the originate).

Add these two extensions to your dialplan:

```xml
<extension name="confringback">
  <condition field="destination_number" expression="^confringback$">
    <action application="set" data="ringback=${uk-ring}"/>
    <action application="bridge" data="{ignore_early_media=true}loopback/confringback_media"/>
  </condition>
</extension>

<extension name="confringback_media">
  <condition field="destination_number" expression="^confringback_media$">
    <action application="pre_answer"/>
    <action application="sleep" data="200000"/>
    <action application="hangup"/>
  </condition>
</extension>
```

And adjust the above examples a little like this:
<extension name="Remove last OB call added to conference">
  <condition field="destination_number" expression="^remove_a_call$">
    <action application="play_and_get_digits" data="3 13 1 5000 # file_string://ivr/ivr-please_enter_the_phone_number.wav!ivr/ivr-followed_by_pound.wav ivr/ivr-that_was_an_invalid_entry.wav target_num \d+"/>
    <action application="set" data="res=${uuid_kill ${hash(select/domain-@domain_name)/last_user_${target_num}}}"/>
    <action application="set" data="res=${uuid_kill ${hash(select/domain-@domain_name)/last_user_ring_${target_num}}}"/>
  </condition>
</extension>

The first condition below matches calls that need simulated ringback, the second is as above (some extension names have been changed).

<extension name="add that call">
  <!-- if we have a four-digit number then attempt to dial it as a user ... -->
  <condition field="destination_number" expression="^ADD_CALL_TO_CONF__(2\d{2})$" break="on-true">
    <action application="set" data="new_uuid=${create_uuid foo}" inline="true"/>
    <action application="set" data="pb_uuid=${create_uuid foo}" inline="true"/>
    <action application="hash" data="insert/domain-@domain_name/last_user_$1/$new_uuid" />
    <action application="set" data="res=${bgapi originate {origination_uuid=${pb_uuid}}loopback/confringback &conference($@conference_name))}"/>
    <action application="set" data="res=${bgapi originate {origination_uuid=${new_uuid}, api_on_answer='uuid_kill ${pb_uuid}'}sofia/gateway/yourgateway.net/$1 &conference($@conference_name))}"/>
  </condition>
  <condition field="destination_number" expression="^ADD_CALL_TO_CONF__(6\d{2})$" break="on-true">
    <action application="set" data="new_uuid=${create_uuid foo}" inline="true"/>
    <action application="hash" data="insert/domain-@domain_name/last_user_$1/$new_uuid" />
    <action application="set" data="res=${bgapi originate {origination_uuid=${new_uuid}}sofia/gateway/anothergateway/$1 &conference($@conference_name))}"/>
  </condition>
</extension>

See Also
- Misc_Dialplan_Tools_bind_digit_action
- Mod_conference