JitterBuffer

About

The jitter buffer is implemented in the Sort Transportable Framed Utterances (STFU) library. The jitter buffer is not enabled by default.

As of FreeSWITCH™ version 1.6 all variables relating to media have been normalized to begin with the string "rtp_"

If you see variables beginning with sip_ that clearly deal with media streams, you are reading outdated documentation. Please let the docs team know by opening a JIRA ticket or leave a comment at the bottom of the affected page. Thanks.

Interesting Information

The following information came from Anthony Minessale in a freeswitch-users thread:

If the jitterbuffer is already on, calling it again will just resize it, so setting it to the same is redundant but harmless.

If you are surprised by why the jitterbuffer is paused during bridge:

If both sides of a bridge are RTP and both sides have a jb, its fairly useless. In fact if anything, it can worsen call quality.

You should only run jitterbuffers at points of termination change of protocol. Examples, if FS was hosting a conference or IVR, if you are bridging the call to a phone for instance, you want to not use a jitterbuffer because you want to preserve the original timestamps so your phone can use its own jitterbuffer.

For special examples where you are using FS jitterbuffer in front of something else that may not have one or some other special circumstance you can use the setting chris mentioned to leave it running.

Activation instructions

The jitter buffer can be activated via channel variable, dialplan app, or sofia param.

The jitter buffer has three params that control its behavior: length, max length, and max drift. Length is the initial size of the jitter buffer in milliseconds. Max length is the upper bound for how big the jitter buffer can grow. Max drift controls how much delay the jitter buffer will tolerate before dropping frames to make up ground.

Dialplan app

Enable 60 ms jitter buffer:

```
<action application="jitterbuffer" data="60"/>
```

Enable 60 ms jitter buffer with 200ms max length and 20 ms max drift:

```
<action application="jitterbuffer" data="60:200:20"/>
```

Sofia profile param

This param is the initial size of the jitter buffer in milliseconds. The max length and max drift values can't be set with this param.

```
<param name="auto-jitterbuffer-msec" value="60"/>
```

jitterbuffer_msec

Activates the jitter buffer. The jitter buffer has three params: length, max length, and max drift.

Usage:

```
<action application="set" data="jitterbuffer_msec=60:200:20"/>
<action application="answer"/>
```

Or to set it on the subsequent outbound (Leg B) call: export sets a variable on both the current channel and on any channels it creates, while the 'nolocal:' disables setting it on the current channel and only sets it on the subsequent outbound channels.

```
<action application="export" data="nolocal:jitterbuffer_msec=60"/>
<action application="bridge" data="sofia/default/888@conference.freeswitch.org"/>
```

You can also activate the Jitter Buffer in the bridge as follows:

```
<action application="bridge" data="{jitterbuffer_msec=60}sofia/gateway/$1@gateway.com"/>
```

This will add a jitter buffer to packets flowing from a remote gateway towards a local freeswitch user. The network would look like this:

```
(local sip user) -----> FreeSWITCH -----> (remote gateway)
```

Where the link between FreeSWITCH and the remote gateway has jitter, and say the local SIP user has no jitter buffering on their IP-phone. This will help the voice quality for the incoming audio.

Other usage

rtp_jitter_buffer_plc

Enables/disables packet loss concealment (PLC) when using the jitter buffer. PLC is enabled by default when the jitter buffer is enabled. Set this variable before enabling the jitter buffer for it to have an effect.

Usage:

```
<action application="set" data="rtp_jitter_buffer_plc=true"/>
```

or to disable PLC:

```
<action application="set" data="rtp_jitter_buffer_plc=false"/>
```

rtp_jitter_buffer_during_bridge

Enables/disables the jitter buffer during bridge.

Usage:

```
<action application="set" data="rtp_jitter_buffer_during_bridge=true"/>
```

or,

```
<action application="set" data="rtp_jitter_buffer_during_bridge=false"/>
```

To enable jitter buffer on only the B-leg of the call, issue commands based on these examples in this sequence:

```
<action application="export" data="rtp_jitter_buffer_during_bridge=true"/>
```

```
<action application="export" data="nolocal:jitterbuffer_msec=60:120"/>
```

Pause

The jitter buffer can be paused mid-call

```
<action application="jitterbuffer" data="pause"/>
```

Resume

The jitter buffer can be resumed mid-call

```
<action application="jitterbuffer" data="resume"/>
```

Debug

Jitter buffer debugging can be enabled/disabled.

```
<action application="jitterbuffer" data="debug:${uuid}"/>
<action application="jitterbuffer" data="debug:off"/>
```

Lookahead

If using the Opus codec (popular with WebRTC) and the far end is sending F.E.C. (Forward Error Correction) information, you can enable a look-ahead jitter buffer in the codec configuration:

autoload_configs/opus.conf.xml

```
<param name="use-jb-lookahead" value="1"/>
```

On FreeSWITCH version 1.6 and later this greatly improves the audio performance even with heavy packet loss.