

# Zap Tell

By  
Anthony Minessale II

This article is based on an observation in a Slashdot post that was met with smoke screen replies and not addressing the issue at hand.

<http://slashdot.org/comments.pl?sid=174812&threshold=1&commentsort=0&mode=thread&cid=14546443>

The original poster's point:

*The big screw-up is timing. Here I am, doing nothing but IAX2. There is no reason that I should have to load a zaptel driver! ("zaptel" being Digium's line of PCI cards) Asterisk refuses to use the normal Linux real-time clock features (POSIX timers, /dev/rtc, etc.) for timing. The zaptel driver is a crude piece of crap that was rejected by the kernel developers. It is unfit for serious use.*

After reading the post I felt obligated to explain the Zaptel dependency as I see it based on my past experience with Asterisk:

Looks like the thread full of insults and lookie heres RE: your timing post missed the point. I will assume you are referring to this:

```
grep ZAPTEL `find . -name \*.c`  
./apps/app_meetme.c:"Please note: A ZAPTEL INTERFACE MUST BE INSTALLED  
FOR CONFERENCING TO WORK!\n\n"  
./apps/app_meetme.c:"A ZAPTEL INTERFACE MUST BE INSTALLED FOR  
CONFERENCING FUNCTIONALITY.\n";  
./channel.c:#ifdef ZAPTEL_OPTIMIZATIONS  
./channel.c:#ifdef ZAPTEL_OPTIMIZATIONS  
./channel.c:#ifdef ZAPTEL_OPTIMIZATIONS  
./channel.c:#ifdef ZAPTEL_OPTIMIZATIONS  
./channel.c:#ifdef ZAPTEL_OPTIMIZATIONS  
./channel.c:#ifdef ZAPTEL_OPTIMIZATIONS  
./file.c:#ifdef ZAPTEL_OPTIMIZATIONS  
./file.c:#ifdef ZAPTEL_OPTIMIZATIONS  
./file.c:#ifdef ZAPTEL_OPTIMIZATIONS  
./file.c:#ifdef ZAPTEL_OPTIMIZATIONS
```

More or less, the timing lesson you were given as a response is not untrue but is also not relevant to your question. Where it may be true that for TDM you have an enormous

dependency on time sync you have much less with VoIP. The bigger reason for the Zaptel timing for non-hardware stuff comes from the hatred of threads that Asterisk has.

In a typical call in Asterisk, 2 channels exist within 1 thread. This thread must poll both channels and service them based on socket events. The first problem is that not every channel type has a socket (most do but IAX2, for example, does not which is another can of worms). The second problem is that most routines in Asterisk make the assumption that if you read from the channel in a loop you can write to the channel on each pass and have perfect timing. Sorry, thanks for playing, but you cannot do that. Many protocols (SIP for example) have VAD (Voice Activity Detection) support.

This means that the far end of your call may not send any voice because the other guy is not talking so why bother. When this occurs the poor Asterisk channel expecting reads to dictate writes simply halts in place DoH!

This problem is fixed by a friendly “Don’t use VAD” console message. And they are trying to improve on it by somehow trying to weave some logic in there to make that poll timeout and let it work still using good ol’ Zaptel timer but it’s like digging a hole too deep and just makes the mess bigger if both channels were in their own independent I/O thread the problem would be gone.

The next place Zaptel is wedged into place is in the channel I/O routines. A pipe is opened inside the channel to allow the queuing of frames meaning you can write them as fast as you want and the channel will deliver them at a steady interval. Thank goodness Asterisk is hard coded to only operate at 8000hz! This also compensates for the channels with no socket you can just queue all the inbound frames instead of write them. This constant pulse is also courtesy of Zaptel timer.

Next there is the file delivery system which is really the one place you really need a timer for sure since the file is simply a large collection of data so it must be fed at a timed interval this also is somewhere the Zaptel timer is used.

The timer is also used by the external music on hold system. The default system uses an external pipe to an mp3 player that also spits the data out too fast since it has no soundcard for timing so the Zaptel timer to the rescue again!

The one place you cannot avoid the timer is the MeetMe application. This is a conference system that depends on the Zaptel to do the audio buffering/muxing and the timing. There is, however, a purely soft-timed algorithm-based conference application available.

[http://www.voip-info.org/wiki-Asterisk+app\\_conference](http://www.voip-info.org/wiki-Asterisk+app_conference)

Nearly all of these uses can be replaced with low resolution timers and normalization techniques (gettimeofday with incremental waypoints) and they are if you compile Asterisk with ZAPTEL\_OPTIMIZATIONS undefined you will get this but you will be out of luck for the music on hold and the MeetMe.

So, based on these facts, the reason your IAX needs Zaptel is, as I mentioned earlier, it has no socket so all the frames are delivered via the queue system and if you compile the application on a system with no trace of zaptel you may be able to get it working properly because the build process will sprinkle the proper ifdefs and ifndefs to get rid of the Zaptel related code and replace it with soft timed equivalents.