Subject: Re: Skylark timer jobs...
Posted by dolik.rce on Tue, 08 Oct 2013 16:13:52 GMT
View Forum Message <> Reply to Message

mirek wrote on Mon, 07 October 2013 09:23Well, do we really need to communicate commands back?

I mean, jobs started before prefork can stay in the main process. Each forked process then can start its own timer queue. The only thing impossible then would be killing job created in the main process in forked process or job id that is somehow transfered (via IPC) from another process. But there is a simple solution to that: make it illegal Wouldn't separate timer in each preforked process could cause same job running in each process? Unless some locking is applied, which is almost same thing as having IPC. I'd be happier if all the jobs were in one place. Even better would be if you could query the queue to get status of such jobs and pause/resume/add/remove/... them. But as said before, this would probably require IPC.

What do you think about the solution I proposed in RM? Do you have any specific arguments against this?
Quote:one simple solution I can imagine is to represent jobs as files (or optionally store them in database, as with sessions). But you can't simply store a callback, so all "schedulable" callbacks would have to be registered at the application start. At best, you could pass some simple arguments through the file (strings, numbers etc.).This would provide the necessary IPC with little overhead.

mirek wrote on Mon, 07 October 2013 09:23Related question (issue?): for the task you suggest (deleting obsolete sessions), I would normally use standard cron job. I mean I really do not see advantage of proposed job queue to normal cron job. Is there any?Simplicity Many users (including me) like that U++ allows you to deploy whole application by deploying single executable. Of course anything that can be done via jobs as discussed here can be in theory either implemented as separate app or called from cron via http (more on this below) as well. Having it in application brings benefit of easy monitoring and management of such jobs from the application itself (automatic or by user via web interface).

Also, if there is a task that needs to be done periodically, but you also want to allow user to perform it immediately (via web interface), then you'd have to implement the same logic in two places (once for cron, once in app). It is possible to overcome this by implementing this in server only and call it from cron via http interface. That is what I do now, but it feels bit unnatural to me. Perhaps it could also have some security implications for some tasks.

Does anyone else have any opinion about this proposed feature? Any input will be greatly appreciated

Best regards,
Honza