Subject: Skylark timer jobs...
Posted by mirek on Mon, 07 Oct 2013 07:23:50 GMT
View Forum Message <> Reply to Message

This is a follow up to Dolik's idea of Skylark jobs, see:

http://www.ultimatepp.org/redmine/issues/523

Quote:
I was thinking about similar interface to what is in CtrlCore (or using bazaar/Timer). I decided to drop support for that because in preforked mode there would be no simple way to communicate the commands back to the timer thread. (Or is there some shared memory or other IPC mechanism in U++?).

Well, 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

Related 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?

Mirek

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

Subject: Re: Skylark timer jobs...
Posted by mirek on Tue, 08 Oct 2013 17:55:33 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Tue, 08 October 2013 12:13mirek 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?

Well, of course, you would have to somehow make sure that jobs scheduled before prefork are not inherited to child processes. Other than that, I think everything is OK (or not?

Quote:
What do you think about the solution I proposed in RM? Do you have any specific arguments against this?

Well, perhaps, but it sounds quite complicated and 'feels' error-prone.

Quote:
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.

Thats not mutually exclusive. We run many cron jobs in our server environment. All are implemented with single binary, which has many modes represented on commandline... That way, all code is easily shared.

Mirek

Subject: Re: Skylark timer jobs...
Posted by dolik.rce on Tue, 08 Oct 2013 19:10:49 GMT
View Forum Message <> Reply to Message

mirek wrote on Tue, 08 October 2013 19:55dolik.rce wrote on Tue, 08 October 2013 12:13Wouldn't separate timer in each preforked process could cause same job running in each process?

Well, of course, you would have to somehow make sure that jobs scheduled before prefork are not inherited to child processes. Other than that, I think everything is OK (or not? All the preforked processes are equal, so they'd all run same jobs over time, because they wouldn't know it has already been taken care of by some other process. Or am I missing something?

mirek wrote on Tue, 08 October 2013 19:55Quote:
What do you think about the solution I proposed in RM? Do you have any specific arguments against this?

Well, perhaps, but it sounds quite complicated and 'feels' error-prone.It depends how complex the functionality would be. Simple running and stopping non-parameter callbacks should be quite easy to do right. More advanced stuff would probably get trickier.

mirek wrote on Tue, 08 October 2013 19:55Quote: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.

Thats not mutually exclusive. We run many cron jobs in our server environment. All are implemented with single binary, which has many modes represented on commandline... That way, all code is easily shared.I admit I didn't think of that Having multiple apps in single binary sounds little obscure at first, but I guess it is not bad in any sense (thinking of it, busybox uses this trick for years...).

So the last argument for this seems to be ability to run in environments where cron is not available. Having little dependencies was always considered good thing in U++ But I'm not sure that this argument is valid, because I haven't met any cron-less such system for very long time... Perhaps some very cheap vps providers, but it's probably very rare. Or windows, but even there, there is the task scheduler or what it's called...

Honza

Subject: Re: Skylark timer jobs...
Posted by mirek on Tue, 08 Oct 2013 19:58:01 GMT
View Forum Message <> Reply to Message

[quote title=dolik.rce wrote on Tue, 08 October 2013 15:10][quote title=mirek wrote on Tue, 08 October 2013 19:55]dolik.rce wrote on Tue, 08 October 2013 12:13Wouldn't separate timer in each preforked process could cause same job running in each process?

I guess that quite depends on usage (and that would be limited by what is "illegal").

Anyway, after some thinking about real-life examples, I would say making it illegal would work, but would not be really practical. Now I agree that we need sharing

One ugly issue though... By adding jobs to the main process, you make it more vulnerable to crashing. Plus, perhaps crashing timer process should not delete timer queue.

So to sum it, storing the timer queue into file system suddenly starts to seem like quite a good option!

Mirek

Subject: Re: Skylark timer jobs...
Posted by dolik.rce on Tue, 08 Oct 2013 21:11:39 GMT
View Forum Message <> Reply to Message

mirek wrote on Tue, 08 October 2013 21:58Anyway, after some thinking about real-life examples, I would say making it illegal would work, but would not be really practical. Now I agree that we need sharing

One ugly issue though... By adding jobs to the main process, you make it more vulnerable to

crashing. Plus, perhaps crashing timer process should not delete timer queue.

So to sum it, storing the timer queue into file system suddenly starts to seem like quite a good option!

I'll try to prepare some basic implementation, we can then try it out and discuss it further...

The problem with crashing main process can probably be solved by making all the processes execute the jobs, they just need to access the queue items carefully enough...

Honza

Subject: Re: Skylark timer jobs...
Posted by mirek on Wed, 09 Oct 2013 05:37:44 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Tue, 08 October 2013 17:11mirek wrote on Tue, 08 October 2013 21:58Anyway, after some thinking about real-life examples, I would say making it illegal would work, but would not be really practical. Now I agree that we need sharing

One ugly issue though... By adding jobs to the main process, you make it more vulnerable to crashing. Plus, perhaps crashing timer process should not delete timer queue.

So to sum it, storing the timer queue into file system suddenly starts to seem like quite a good option!

I'll try to prepare some basic implementation, we can then try it out and discuss it further...

The problem with crashing main process can probably be solved by making all the processes execute the jobs, they just need to access the queue items carefully enough...

Honza

..or you can fork (and re-fork) one process just for callbacks...
