

Re: Creating a 'load simulator' by calling Perl Programs – or Forking?

Re: Creating a 'load simulator' by calling Perl Programs – or Forking?

Source: <http://coding.derkeiler.com/Archive/Perl/comp.lang.perl.misc/2008-04/msg00204.html>

- *From:* pgodfrin <pgodfrin@xxxxxxxx>
 - *Date:* Thu, 3 Apr 2008 21:12:39 -0700 (PDT)
-

On Apr 3, 4:22 pm, xhos...@xxxxxxxx wrote:

pgodfrin <pgodf...@xxxxxxxx> wrote:

...

Question #1: What is the proper way to execute the perl programs?

You gave two examples, sequentially and in parallel. Which one is appropriate depends on what kind of load you want to simulate, which in turn depends on why are you are doing the simulation.

Question #2: Is there a way to echo to the screen that the wait() is happening and provide some indication of progress?

To clarify this question, I had originally tried:
do {\$x=wait; print ".";} until \$x==1;

Thinking I would get a period printed during the loop. (silly me – wait only comes back when a child process dies...).

So you do get a period printed, once per child–exit. This is what I would want to do, anyway, but I seem to be in a philosophical minority in this regard (I hate progress indicators that merely indicate that the system clock is ticking, and nothing about the actual progress of the actual program.) I'm not sure what you originally thought it might do instead. Periods to be printed as fast as Perl possible can print them?

Re: Creating a 'load simulator' by calling Perl Programs – or Forking?

Of course if this is much to complicated a way to so a simulated load, then so be it – just a plain old fork and wait works fine... And I could always remove the /dev/null and actually see the called programs output, which will tell me that it's still running... But I was curious if there is a more elegant or at least interesting way of doing this?

If the waiting program is still running, and is still waiting, then the program it is waiting on must still be running. What other possibilities are there?

If you really want the "time has not come to a halt" thing, then something like this:

```
perl -le '$|=1; fork or do {sleep 4*$_; exit} foreach (1..3); \
$SIG{ALRM}=sub{print "."; alarm 1}; alarm 1; \
do {$x=wait; print "$x";} until $x== -1; \
alarm 0'
.
.
.
20287
.
.
.
20288
.
.
.
20289
-1
```

Xho

-----<http://NewsReader.Com/>-----

The costs of publication of this article were defrayed in part by the payment of page charges. This article must therefore be hereby marked advertisement in accordance with 18 U.S.C. Section 1734 solely to indicate this fact.

Hi Xho,
As always – good points. I am looking to load up the system, so a parallel simulation seems to be the best approach. And frankly, the period printing was just a way to see that indeed something was happening, but clearly and wholly unnecessary – although I did get a chuckle over the time halting comment...

Re: Creating a 'load simulator' by calling Perl Programs – or Forking?

And anyway 'watch ps u' in anotehr session celarly shows me the processes still running.

But – the question still remains – what's the best way to call other perl programs? Exec, system or something else?

pg

.