asemanfar - a blog about programming

- blog
- projects
- twitter
- admin

Why We Wrote Bluepill

October 31, 2009

At Serious Business, we use god to monitor our long-running processes (mongrel, background workers, and more recently unicorn). We had a basic god config setup that only checks for memory usage, cpu usage, and request queue length (for mongrels only). God was working fine for us except for one problem: the notorious memory leak. If you use god you probably know it leaks memory in correlation with the number of watches on the system. This became a problem for us when god hadn't been restarted for several days; its memory usage would climb and reach several gigs, causing the machine to swap and eventually lock-up. To prevent lock-ups we needed to manually monitor god (what does that make us?) and restart god daily via cron, gross.

Our frustration with this issue eventually reached a point where we decided to write our own process monitoring tool. Rohith Ravi, Gary Tsang, and I got together one weekend and built a first version of what we've come to call bluepill. We spent the next couple weeks massaging the DSL, expanding feature set, and fixing some bugs we found while using it for our apps. The current feature set is small but is sufficient for the most users:

- DSL for specifying processes and their respective watches
- Built-in support for monitoring memory usage and CPU usage
- Support for custom conditions to watch
- Daemoniziation of non-daemonized processes
- Monitoring child processes (especially useful for monitoring unicorn workers)
- Logging
- Support for triggers (flapping)

While both bluepill internal and the external interface is heavily influenced by god, we decided do some things differently in bluepill:

- Written with long-running daemon in mind (read: low resource consumption)
- Simplicity over flexibility:
- one process per application; forces separation between multiple apps on the same box
- simple state machine; does only what it needs to to keep the process up

This past week, we ran a test to see how well Bluepill will do in the wild compared to god, so we set up a basic bluepill config file and the equivalent god config on two identical machines and recorded their memory usage every 30 minutes for just over 4 days.

bluepill vs god memory usage

In addition to the memory leak issue, we sought to improve god in a few other ways: sequential CLI command processing and monitoring child processes:

CLI Command Processing
In god, CLI issued commands are sent to the long-running god daemon which starts a separate thread and returns to the CLI; this led to some race cases when you issue two commands sequentially and expected them to execute in that order (i.e god stop <process_name>; god start <process_name>). This is fixed in bluepill by handling CLI issued commands in a single thread fed by a queue.

Monitoring Child Processes
We recently switched in Unicorn which starts its own long-running child processes to handle requests. So in order to monitor the unicorn workers, we needed to add support for monitoring child processes. Child process monitoring differs from regular process monitoring because bluepill is not responsible for starting them back up and the PID comes from the parent process and not a PID file.

We're going to continue working on it to improve its feature set and iron outs any bugs that we find.

Read the readme for usage information. Read the design file for technical details.
Fork and contribute: http://github.com/arya/bluepill
Report bugs: http://github.com/arya/bluepill/issues

tagged with: ruby, programming, bluepill

Comments

posted by John Adams on 01/10/10 07:20 PM PST

Why not just fix the memory leak in god instead of writing yet another tool to monitor processes? In the ruby community there must be 30 different ways of solving the "keep this process up" problem.

It's yet another bandaid.

Also, why not monit?

posted by luis on 04/19/10 10:14 PM PDT

can god monitor unicorn workers ??

Thanks

posted by Eric Marden on 05/26/10 08:19 PM PDT

Sounds awesome. Can this be used for php/fastcgi?

posted by Graham Ashton on 07/27/10 11:12 AM PDT

I really like the sound of this; I've had various troubles with god over the years, and the latest

version doesn't seem very happy with resque's example god config.

John Adams - Why not write your own tools? It drives progress and teaches you a lot while you're at it. I tried monit once and found it horrible to work with when it wasn't behaving itself.

Leave a Comment

Name (required):

E-mail (required):

Website:

Make this box bigger dammit! - Want syntax highlighting?

You can surround a block of code with a <code> and </code> optionally specify a language for the that code of block using the lang attribute <code lang="ruby">. Some of the choices for langauge are: ruby, html, ruby_on_rails, html_rails, javascript, python, java, c++, c, php, sql, and probably anything else you try.

Submit