The s6 supervision suite

Laurent Bercot, 2017

What is an init system ?

- “init” is vague terminology. “init wars” happened because nobody had a clear vision on what an init system even **is** or **should be**.
- The 4 elements of an init system: /sbin/init, pid 1, process supervision, service management.
- Not necessarily in the same process.

Definition: process supervision

A long-lived process (daemon) is *supervised* when it’s spawned by the *supervision tree*, a set of stable, long-lived processes started at boot time by pid 1. (Often just pid 1.)

Supervision is a good pattern: the service is stable and launched in a reproducible env.

Supervision only applies to daemons.

Service management: definition

- Boot time: bring all services up
- Shutdown time: bring all services down
- More generally: change services’ states

Services can be oneshots (short-lived programs with side effects) or longruns (daemons). They have dependencies, which the service manager should enforce.

What features do “init”s offer ?

- Integrated init systems (systemd, launchd, upstart): “the big guys”. All four elements in one package, plus out-of-scope stuff.
- sysvinit, BSD init: /sbin/init, pid 1, supervision (/etc/inittab, /etc/gettys). Service manager not included: sysv-rc, /etc/rc
- OpenRC: service manager.
- Epoch: similar to sysvinit + sysv-rc

The “daemontools family”

- /etc/inittab supervision is impractical; nobody uses it for anything else than gettys.
- daemontools (DJB, 1998): the first project offering flexible process supervision. Realistic to supervise all daemons with it.
- daemontools-encore, runit, perp, s6: supervision suites.
- nosh: suite of tools similar to s6, in C++

Supervision suites are not enough

- Only ¼ of an init system. A full-featured init needs all 4 parts. (And nothing more.)
- runit provides /sbin/init and pid 1, but no service manager
- Void Linux uses runit without a service manager; it sometimes needs hacks (longruns doing nothing) to emulate oneshot services.

s6: a modern supervision suite

- Provides a (portable) pid 1.
- Provides hooks for service manager integration.
- There are two service managers designed to work with s6: anopa and s6-rc. They work on top of s6, not in the same layer.
- s6 is designed to be portable; /sbin/init cannot be. Script created by a system-specific program.

s6: technical aspects

- Design: modularity/layering done right.
- Design: s6-rc is a **parallel** service manager, with **reproducible env even for oneshots**.
- Portability: to any POSIX system.
- Pure C, all deps controlled, easy to bootstrap
- Uses notification; never polling.
- Lightweight: ~2 MB disk, negligible RAM/CPU.
- Very little code, short code paths

s6: availability and integration

- Packaged in all good distros - as mechanism, not policy.
- Used as pid 1 in Docker containers. s6-overlay predates s6-rc and anopa.
- Service manager integration requires joint work with the distribution.
- Plans to make s6 + s6-rc an alternative to busybox init + OpenRC in Alpine Linux.

s6: to learn more

http://skarnet.org/software/

skaware@list.skarnet.org

supervision@list.skarnet.org

Freenode IRC: #s6

Twitter: @laurentbercot

Or… come talk to me today or tomorrow!