Useful systemd functionalities without systemd

Bartosz Golaszewski
bgolaszewski@baylibre.com
Bartosz Golaszewski

➢ > 6.5 years professional experience in embedded Linux
➢ Contributor to the linux kernel, busybox, buildroot, uClibc, sigrok
➢ Part of the firmware team in Project Ara
➢ Maintainer of ACME software
Agenda

➢ ACME, sigrok & faster boot
➢ Why not systemd?
➢ Recreating functionalities
  ➢ Parallel startup
  ➢ Readahead
  ➢ Security
  ➢ Watchdogs
ACME, sigrok & faster boot
ELC 2015 – San Jose

« Tunning systemd for embedded » by Alison Chaiken

- systemd philosophy & design
- Pros & cons
- Demoed systemd's cgroups support, privilage limitation, watchdogs
ACME Cape

- Low-cost multi-channel power monitor
- Based on TI ina226 power monitor
- Open hardware/open source
- Standard BeagleBone Black cape
- USB, Jack & HE10 connectors
- http://baylibre.com/acme/
- ACME wiki
Sigrok

- Portable, cross-platform, free signal analysis software suite
- Broad hardware support
- Various frontends
- Reusable libraries
- http://sigrok.org/
Sigrok + ACME + PulseView

➢ X.org, fluxbox x 2, Qt5, Xvfb, x11vnc, PulseView x 2
➢ Boost, uClibc, etc.
➢ Startup scripts in /etc/init.d
➢ No parallelization
Why not systemd?
Buildroot/busybox init

- Minimalistic version of SysV init
- No runlevels in busybox init
- Scripts in /etc/init.d with start/stop commands
- No restart, status, etc. commands
- Startup scripts privided by buildroot packages
systemd

- Available as buildroot package
- Painless configuration
- Mostly just works
- Drastic improvement in boot-time
- X.org is integrated and supports socket activation
Reasons not to use systemd in embedded devices

➢ Merciless deprecation of features
  ➢ udev firmware loader
  ➢ readahead
➢ Dependence on newer kernels
➢ Dependence on glibc
➢ Size & attack surface
➢ Level of complication
Dependence on newer kernels

- Forcing kernel updates is great... but not always feasible
- Old kernels often in use by companies
- Out-of-tree SoC support
- Upgrade not always possible
Size, attack surface & level of complication

➢ Unnecessarily many functionalities in PID 1
  ➢ Could be avoided with `prctl(PR_SET_CHILD_SUBREAPER, ...)`

➢ Privileged process with many inputs and code paths

➢ >330000 LOC (src/core alone >36000 LOC)
Recreating systemd functionalities
Parallel service startup

➢ **systemd**:  
  ➢ several startup notification types  
  ➢ Requires=, After= & Before= configuration options

➢ **Busybox**:  
  ➢ Block runit startup scripts to wait for dependencies  
  ➢ Use inotifyd to find out when the pidfile has been created
Demo: parallel startup
(with busybox runit and inotifyd)
Readahead

➢ Obsolete on modern desktops
➢ Still useful on boards with slow eMMC's
➢ systemd implementation nuked...
➢ busybox applet
  ➢ works out-of-the-box
  ➢ can be run as init
Demo: readahead

(busybox applet)
Security features

➢ Extensive cgroups usage for security and resource management in systemd
➢ Not all these features needed in embedded systems
➢ Use cgroup-tools & unshare to achieve similar results
Demo: security features

(cgroups + unshare)
Software watchdogs

➢ systemd:
  ➢ `ShutdownWatchdogSec=` option for hw watchdog
  ➢ `WatchdogSec=` and `Restart=on-failure` for sw watchdog
  ➢ process required to call `sd_notify("WATCHDOG=1")` periodically

➢ Alternative:
  ➢ Use some trivial action to determine whether a process is still alive
Demo: software watchdogs
(without dbus/libsystemd)
Conclusion

➢ Philosophy of systemd is: 
*Extract duplicate functionality from daemons and move it to systemd core or kernel.*

➢ Most of this functionality already exists in a tiny busybox binary and also *mostly* does the job.
References

➢ http://she-devel.com/ELC_systemd.pdf
➢ http://0pointer.net/blog/
➢ busybox documentation
➢ systemd manuals
➢ http://ewontfix.com/14/
➢ http://busybox.net/~vda/
Q & A

Bartosz Golaszewski
bgolaszewski@baylibre.com