

A systemd rendszer- és szolgáltatáskezelő démon

Vajna Miklós

2011. szeptember 23.

Miről lesz szó?

- A systemd projektről röviden
- Killer feature-ök
- Funkciók bemutatása
- Migrációs tapasztalatok
- Bekapcsolódás a fejlesztésbe

- Rendszer- és szolgáltatáskezelő démon
- sysvinit leváltására
- SysV és LSB init scriptekkel kompatibilis
- Legfontosabb funkciók:
 - agresszív párhuzamosítási képességek a függőség-alapú szolgáltatásvezérlő logikának köszönhetően
 - igény szerinti szolgáltatás-indítás socket/D-Bus activation segítségével
 - rendszer állapotának mentése, monitorozása és visszaállítása



- Szerzők: Lennart Poettering (Red Hat), Kay Sievers (Attachmate/SUSE)
- Nem ismeretlenek: udev, pulseaudio, avahi, ifplugd, stb.
- Első kiadás: 2010. április 30.
- Ötletek más init rendszerekből: Solaris SMF, Apple launchd, Canonical upstart
- Csak Linux kernelen
- Mára a legtöbb nagy disztribúció vagy váltott rá, vagy tervezi a váltást a következő verzióban.

A systemd projektről röviden

Ki váltott már?



- Fedora 15: ez az alapértelmezett
- openSUSE: 11.4-ben választható, 12.1-re tervezett gyárilag
- Mandriva 2011: alapértelmezett
- Arch Linux: közösségi repóból telepíthető
- Frugalware 1.5: alapértelmezett
- Debian GNU/Linux: SID-ből telepíthető
- Gentoo: testing-ből elérhető

A systemd projektről röviden

Miért csak Linuxon?

Mert olyan kernel funkciókat használ, ami jelenleg más rendszereken nem érhető el. Példák:

- cgroups
- autofs4
- libudev
- capabilities
- /sys
- IP_FREEBIND
- oom score
- binfmt_misc

- A rendszerindítás építőkövei az rc scriptek voltak, példa a smartd rc scriptjére: 653 sor.
- A systemd ezek helyett service file-okat használ, például:

```
[Unit]
Description=Self Monitoring and Reporting Technology (SMART) Daemon

[Service]
ExecStart=/usr/sbin/smartd -n

[Install]
WantedBy=multi-user.target
```

Killer feature-ök

Minden szolgáltatásnak saját cgroup

- A monitorozás a sysvinit mellett nehézkes: honnan tudjuk, hogy egy szolgáltatás leállt?
- Tipikus hack: pidfile + bízni, hogy a megfelelő pid íródik bele, a szolgáltatás tényleg leáll
- Honnan tudjuk, hogy egy process melyik szolgáltatáshoz tartozik? (zypper ps)
- Probléma logout, shutdown esetén
- Megoldás: minden szolgáltatásnak saját cgroup, az onnan fork()olt processek is benne maradnak
- Ha már cgroup: limitek (cpu, memória, io, stb.) cgroup-szinten

Killer feature-ök

Párhuzamosítás, implicit függőségek

- Ismert: párhuzamosítással gyorsítható a rendszerindítás
- Probléma: függőségek kellene hozzá, kézzel meg kell adni, elég egy hiba és nem bootol a rendszer
- Megoldás: Tegyük le a szolgáltatások socketjeit előre, majd indítsunk el mindent egyszerre, a többit a kernel megoldja (socket activation)
- Ha már a socketek külön életet élnek: restart egyetlen socketre beérkezett kérés elmulasztása nélkül
- Innen adódik az igény szerinti szolgáltatás-indítás is
- Hasonló ötlet alkalmazható device, path, DBus esetén

Killer feature-ök

Disztribúció-független megközelítés

- init rendszer tipikus terület ahol a disztribúciók igenis különböztek (eddig; a telepítő és csomagkezelő mellett)
- Az rc scripteknél ez teljes FAIL: különböző név, könyvtárak, runlevelek, paraméterek, konfigurációk
- Az LSB ezen segítene, de a gyors boothoz ez se volt elég, aki implementál LSB-t, az is használ egyedi kiegészítéseket
- Itt: a service file legyen disztribúció-független és az upstream tarball része
- Ugyanez egyéb alacsony szintű config file-okra:
/etc/os-release és társai

Funkciók bemutatása

Aktiválás

- Rendszerindítás során: targetek, azok dependjei (egy aktiválás egy symlink, mert egymásra épülnek a targetek, nem úgy, mint sysvinitnél)
- Socketre küldött üzenettel: syslog, cups, dbus, udev, stb.
- Dbus: Dbus által elérhető szolgáltatásokhoz (DBus-ban ehhez mapping)
- Eszközön keresztül: ha bedugom a nyomtatót, induljon el a cups, bluetooth/bluetoothd ugyanígy
- Path: munkakönyvtárba lerakott file (nem titkoltan az OSX-ből átvett ötlet)
- Timer: oneshot service-ek periodikus futtatása (/tmp kipucolása)

- A cgroup funkció miatt pontos kép a szolgáltatás folyamatairól
- Akció definiálható, hogy mi történjen, ha a szolgáltatás leáll
- API, ahol a szolgáltatás jelezheti az állapotát
- Indítás során minden kimenet logolódik, nincs „eltűnt” hibaüzenet

Funkciók bemutatása

Mount pontok kezelése

- Szükséges: ha minden egyszerre indul, szolgáltatásoknak kellhet valamilyen mount point.
- Lehetséges az aktiválás itt is: autofs.
- Explicit mount unitokkal, vagy fstab alapján.
- Ráépül az fsck service-ekre.
- A mount unitokra is épülnek automatikusan: pl. kvóta vizsgálata.

Funkciók bemutatása

Hogy néz ez ki a gyakorlatban?

- Minden kis egység unit, ezek fajtája lehet service, socket, path, mount, device, stb.
- Freedesktop projekt: .desktop formátumot használja újra
- Csomagból /lib/systemd/system alatt, /etc alá másolható és testreszabható
- Futási időben: /run alatt
- Generátorok: kompatibilitás miatt, valamint udev üzenetekből legyártott device unitok, stb.

- Létező megoldás jobb kezelése esetén: visszafele kompatibilitás
- fstab, rc scriptek, generátor interfész
- Új konfigurációs file-ok esetén disztró-specifikus `#ifdef`-ek *korlátozott* ideig megengedettek
- A `systemd` még mindig újdonságnak számít: leginkább referencia jellegű dokumentáció érhető csak el

- Első systemd csomag: 2010. november 23.
- Opcionálisan a -current ágban: 2010. december 20.
- Opcionálisan a -stable ágban: 2011. február 13.
- Alapértelmezett a -current ágban: 2011. február 21.
- Alapértelmezett a -stable ágban: 2011. augusztus 15.
- Mi vesz időt igénybe: disztró-specifikus funkciókhoz generátorok írása, tesztelés

Bekapcsolódás a fejlesztésbe

- Szabad szoftver, de vásárolható hozzá kereskedelmi támogatás
- Kezdetben: legtöbb változtatás, hogy a systemd olyan jól fusson más disztribúciókon mint Fedorán
- Szokásos „a mai felhasználók a jövő fejlesztői” elv
- A kód nagy részét Lennart írta: egységes forráskód, nincs elbonyolítva absztrakciós rétegekkel, szétgányolva #ifdef-ekkel
- Új funkciók: megbeszélés levlistán, majd TODO, majd implementáció
- Hackeléshez szükséges tudás minimális: C, libudev, libdbus ismeret kell csak hozzá

Bekapcsolódás a fejlesztésbe

Egy kis statisztika

```
$ git shortlog -s -n|head
2410  Lennart Poettering
 137  Kay Sievers
   26  Michal Schmidt
   20  Michael Biebl
   17  Andrey Borzenkov
   16  Miklos Vajna
   13  Harald Hoyer
   13  Tom Gundersen
   11  Fabiano Fidencio
   10  Bill Nottingham
```

- **Miért csak Linux-on fut a systemd:**
<http://mail.gnome.org/archives/desktop-devel-list/2011-May/msg00447.html>
- **Podcast:** <http://media.libsyn.com/media/linuxoutlaws/linuxoutlaws160.ogg> (1:10:50-nél kezdődik, 17 perc)
- **linux.conf.au 2011 videó:**
<http://linuxconfau.blip.tv/file/4696791/>

- **systemd honlap:**
`http://freedesktop.org/wiki/Software/systemd`
- **Levelezési lista:** `http://lists.freedesktop.org/mailman/listinfo/systemd-devel`
- **IRC:** `irc://irc.freenode.net/systemd`
- **A diák elérhetősége:** `http://vmiklos.hu/odp/`