systemd

Like it or not, it's here to stay

Alessandro Di Federico

ale@clearmind.me

Politecnico Open unix Labs

13 aprile 2016

Indice

Cos’è systemd?

Un init system per GNU/Linux

- Gestisce il boot
- Monta i dischi
- Gestisce i servizi
  - server grafico
  - cups (stampa)
  - servizio bluetooth
  - server web, di posta, VPN...

Perché ne parliamo?

- Nato nel 2010, è oggi estremamente popolare
- Init system di default su:
  - Debian (da Jessie)
  - Ubuntu (da 15.04)
  - Fedora (dalla 15)
  - Red Hat (dalla 7)
  - Arch Linux
- In altre distribuzioni è opzionale (e.g. Gentoo)

Pro e contro

- systemd mira ad essere un init system moderno
- Si ispira a launchd, l'init system di Mac OS X
- Affronta molte problematiche degli init system SysV

Idea chiave

Start less, start in parallel

Idea chiave

Start less, start in parallel

ovvero, blocca un servizio solo quando è davvero necessiario

Come?

- Abbiamo una webapp *W* e un servizio MySql *M*
- *W* dipende da *M*
- Comunicano tramite una connessione TCP sulla 3306
- Tradizionalmente *W* è avviato dopo *M*

Enters systemd

1. Il processo di systemd ascolta sulla 3306
2. I due servizi vengono avviati contemporaneamente
3. La webapp viene caricata finché non necessita di MySql
4. Quando pronto, MySql prende controllo della porta 3306
5. La webapp si collega a MySql e procede con il suo avvio

Workflow del servizio

- systemd effettua una fork
- Apre un socket TCP in ascolto sulla 3306
  File descriptor 4
- systemd imposta una variabile d'ambiente:

  $LISTEN_FDS numero di file descriptor aperti
- systemd effettua una exec su MySql
- Prima di aprire un socket MySql controlla se c'è già
- Se sì, usa quello

sd_listen_fds

- MySql usa sd_listen_fds per verificare se il socket è già pronto
- sd_listen_fds controlla $LISTEN_FDS
- Questo metodo di avvio deve essere supportato dal servizio
  - Supportato: dovecot, cupsd, apache
  - Non supportato: exim, mysql, nginix, lighttpd, openvpn

Anche con pezzi di file system

- Un servizio può richiedere che una certa directory sia montata
  Ad esempio se si ha /home su una partizione distinta
- Se invece non ne ha bisogno, non aspetta!
- Quindi, il boot procede mentre è in corso un fsck su /home

Start less

- systemd non solo avvia in parallelo, ma avvia anche meno
- Ad esempio, se nessuno si collega a MySql non viene avviato
- Alla prima connessione viene lanciato on-demand
  - Utile su desktop per servizi usati di rado (bluetooth)
  - Poco utile in un contesto server
- autofs permette di montare anche file system on-demand

cgroup

- I cgroup sono una feature specifica di Linux (non-POSIX)
- Limita cumulativamente le risorse di un set di processi
- Simile a setrlimit ma:
  - setrlimit: nessun processo può usare più del 50% della CPU
  - cgroup: questi processi, assieme, non possono usare più del 50% della CPU
- Utile per limitare (e uccidere) tutti i figli creati da un servizio
- TODO: cgtop

Altri vantaggi

- Script di avvio dei servizi uniformi su tutte le distro
- Script di avvio mantenuti dagli sviluppatori
- Il formato per descrivere un servizio è dichiarativo
- TODO: no more turing complete input languages
- Sistema di boot in C (niente più shell script)

I contro: dev con posizioni controverse

- TODO: link efi vars
- TODO: link altri WONTFIX
- L'init system è il processo più importante del sistema

1.6 MiB di PID 1

- L'init system è il processo più importante del sistema
- Se crasha, l'intero sistema va in kernel panic
- La filosofia UNIX lo vorrebbe piccolo, specifico è semplice
- systemd è grosso e sofisticato
- Addirittura si interfaccia alla rete
- Un bug in systemd può avere esiti catastrofici

Dipendenze e accoppiamento

- systemd, come vedremo, ingloba molti progetti
- Crea forti dipendenze
- Altri progetti sono tentati di richiederlo
  Ad esempio, GNOME
- Specifico a GNU/Linux
  GNOME per *BSD?

Non solo un init system

- logind
- systemd-udev
- colord
- journald

Like it or not, va conosciuto

Indice

Unit file

Una unit rappresenta una risorsa gestita da systemd:

.service	Un servizio, ad esempio server web o di stampa
.socket	Un socket gestito da sytemd, usato per attivare servizi
.device	Un device gestito da systemd (udev)
.mount	Un punto di mount, ad esempio /home
.automount	Un punto di mount da attivare on-demand
.swap	Un device usato come swap
.target	Meta-unit di alto livello, ad esempio sessione-grafica-pronta.target
.path	Un percorso nel file system, si attiva quando diventa disponbile
.timer	Un evento da eseguire periodicamente (cron)

Locazione degli unit file

- Si trovano principalmente in tre locazioni:

  /{etc,run,lib}/systemd/system

- In ordine di priorità decrescente

  1. /etc: configurazioni specifiche del sistema attuale
  2. /run: effimero, per unit generate automaticamente
  3. /lib: persistente, unit scritte da sviluppatori

Esempio di unit: cups.service

```
[Unit]
Description=CUPS Scheduler
Documentation=man:cupsd(8)

[Service]
ExecStart=/usr/sbin/cupsd -l
Type=simple

[Install]
Also=cups.socket cups.path
WantedBy=printer.target
```

Sezione [Unit]

Supponiamo di avere uno unit file *a*

Requires=*b*	*a* dipende strettamente da *b*
Wants=*b*	*a* dipende debolmente da *b*
BindsTo=*b*	Lo stato di esecuzione *a* segue quello di *b*
Conflicts=*b*	Se *a* è avviato, *b* viene arrestato

Ordine

- Requires non forza l'ordine di avvio
- Le unit vengono avviate in parallelo
- L'ordine di avvio può essere specificato con Before e After

Sezione [Install]

- Specifica cosa fare quando una unit viene *abilitata*
- *Abilitare* significa tipicamente avviare al boot
- Supponiamo di avere una unit *a* e un **target** *c*:

  WantedBy=*c* Come se in *c* ci fosse Wants=*a*
  RequiredBy=*c* Come se in *c* ci fosse Requires=*c*
  Also=... Unit «sorelle», ovvero che vengono gestite assieme

Tipi di servizio

Vari tipi (Type), a seconda di quando sono da considerasi avviati:

simple	Avvia un processo (ExecStart) che resta in esecuzione.
forking	Avvia un processo che crea un figlio (fork) ed esce.
oneshot	Avvia un processo che termina a breve.
dbus	Avvia un processo che si rende disponibile su dbus.
notify	Notifica a systemd di essere avviato con sd_notify.

Sezione [Service]

ExecStart=... Comando da invocare all'avvio.

ExecReload=... Comando da invocare per ricaricare la configurazione.

ExecStop=... Comando da invocare all'arresto del servizio.

Exec{Start,Stop}{Pre,Post}=... Comando da invocare prima/dopo l'avvio/arresto.

RemainAfterExit=yes In un servizio **oneshot**, è da considerasi in esecuzione anche dopo la terminazione.

systemctl

`systemctl` Mostra la lista delle unit attive.

`systemctl –all` Mostra tutte le unit disponibili.

`systemctl start unit` Attiva la unit.

`systemctl stop unit` Arresta la unit.

`systemctl reload unit` Ricarica la configurazione di una unit.

`systemctl enable unit` Installa una unit (tipicamente, avvia al boot).

`systemctl disable unit` Disinstalla una unit.

`systemctl status unit` Mostra lo stato di una unit (attiva, installata...)

`systemctl mask unit` Impedisce l'esecuzione di una unit.

`systemctl unmask unit` Permette l'esecuzione di una unit.

Scriviamo un servizio

```
[Unit]
Description=Create a file /hello.txt
RequiresMountsFor=/var/lib/systemd/random-seed
Conflicts=shutdown.target

[Service]
Type=oneshot
RemainAfterExit=yes
ExecStart=/usr/bin/touch /hello.txt
ExecStop=/bin/rm /hello.txt

[Install]
WantedBy=lightdm.service
```

Template di unit

- Le unit il cui nome contiene @ sono dei *template*
- Ovvero sono parametrici rispetto a quello che c'è dopo @
- Ad esempio:
  - openvpn@.service → openvpn@443.service
  - ifup@.service → ifup@eth0.service
  - user@.service → user@1000.service
- Il parametro vien espanso nella unit tramite %I
- È possibile istanziare più unit con parametri diversi.

Generatori di unit

- Un generatore ispeziona il sistema e crea le unit appropriate
- Può considerare file di configurazione, device disponibili...
- Al boot vengono eseguiti tutti quelli in:

  /lib/systemd/system-generators/

- E le unit vengono create in:

  /run/systemd/generator/

Esempio: systemd-fstab-generator

- Il file /etc/fstab contiene la configurazione dei punti di mount:

```
# <file system> <mount point>   <type>  <options>
<dump>  <pass>
/dev/sda1  /                    ext4     errors=remount-ro
1
/dev/sda5  none                 swap     sw
0       0
/dev/sr0           /media/cdrom0    udf,iso9660 user,no
0       0
```

- systemd-fstab-generator li traduce in unit di mount

Unit utente

- È possibile avere unit in un contesto utente
- L'utente può creare le proprie in:

  ~/.config/systemd/user

- Utilizzabili con systemctl –user

Indice

Il journal

- Il journal di sistema registra messaggi utili all'amministratore
- Tipicamente:
  - Messaggi da servizi
  - Messaggi dal kernel
- systemd ha un suo sistema di journaling: journald

journald

- Un punto unico per **tutti** gli eventi di sistema
- Caratteristiche del log
  - indicizzato
  - robusto alla corruzione (per lo più operazioni in append)
  - Forward Secure Sealing
- Registra gli eventi in formato binario
- Tipicamente non è persistente
  - mkdir /var/log/journal
  - Debian usa anche rsyslog (/var/log/*.log)
- Configurabile da /etc/systemd/journald.conf

journalctl

journalctl	Mostra tutti i log disponibili.
journalctl -b	Mostra solo i log di questo boot.
journalctl -b -1	Mostra solo i log del boot precedente.
journalctl –list-boots	Mostra i boot per cui sono disponibili log.
journalctl –since yesterday	Mostra i log da ieri.
journalctl -u my.service	Mostra solo i log per my.service.

Altri strumenti

machinectl	Gestisce container (systemd-nspawn e altri).
loginctl	Interagisce con logind (ad es. per listare le sessioni attive)
timedatectl	Permette di controllare l'ora e il client NTP.
hostnamectl	Permette di modificare hostname e nome di dominio.
localectl	Permette di impostare la lingua e il layout della tastiera.

License