Breizh C@mp
Mix de technologies

Les développeurs aussi maîtrisent le systemd

#systemD

Jean-Eudes Couignoux - @Jean_Eudes

Objectif

Calendrier

- Système d'init ?
- SystemD : Une solution à quelles problématiques ?
- Décortiquons un peu systemD
- Les « scripts » SystemD

L’ère pré-systemD

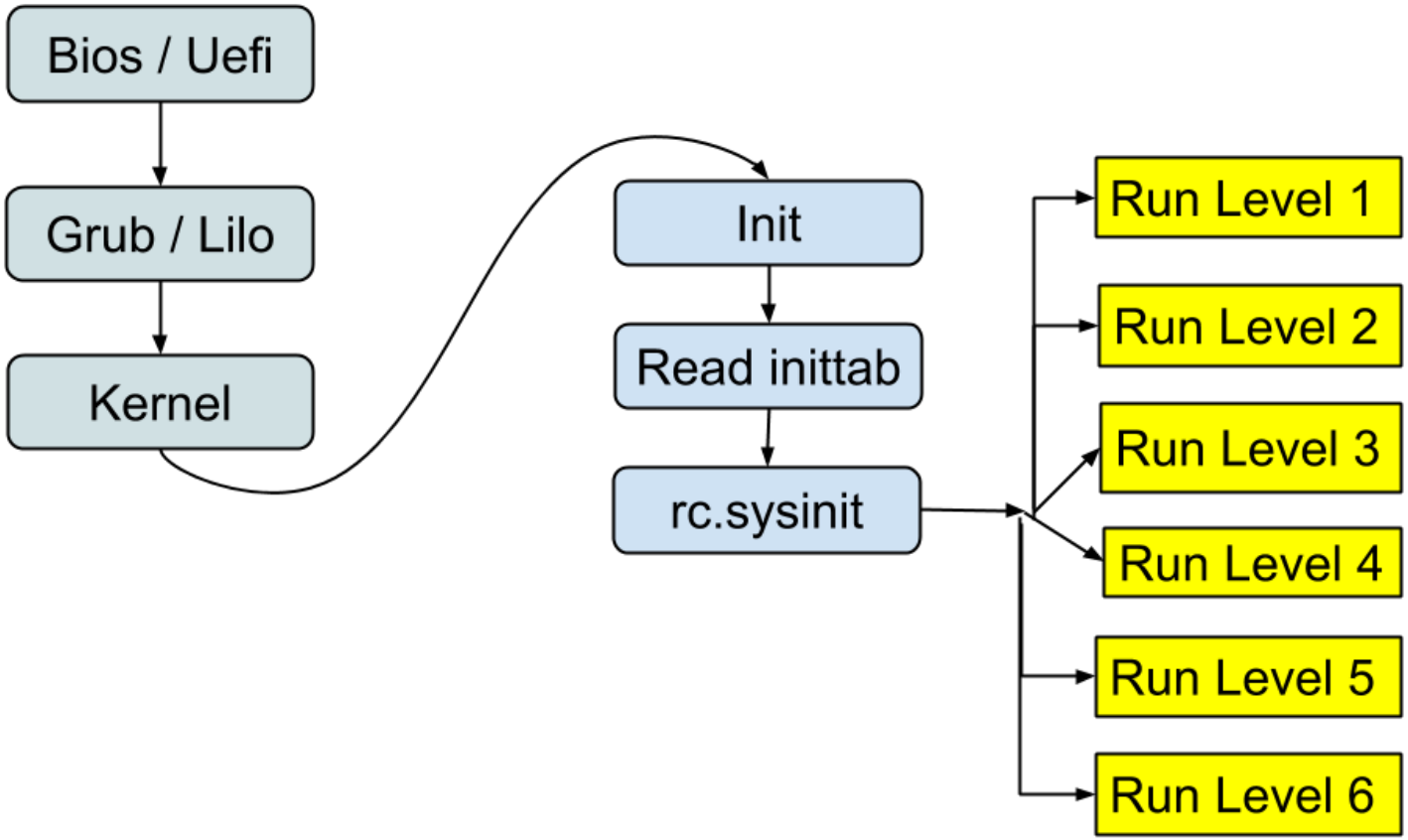
Responsabilité de l'init

- Initialisation des processus
- Gestion du cycle de vie des processus
- Resté en vie

System V

- Lecture du fichier inittab
- Chargement des différents niveaux

System V


Bios / Uefi
Grub / Lilo
Kernel
Init
Read inittab
rc.sysinit
Run Level 1
Run Level 2
Run Level 3
Run Level 4
Run Level 5
Run Level 6

System V

```
. /lib/lsb/init-functions
NAME=cocktail
EXEC=$CATALINA_HOME/bin/startup.sh
PIDFILE=$CATALINA_BASE/logs/cocktail.pid
export CATALINA_PID=$PIDFILE
do_start()
{
start-stop-daemon --start --quiet --pidfile $PIDFILE --exec $EXEC -c tomcat
}
do_stop()
{
start-stop-daemon --stop --quiet --retry=TERM/30/KILL/5 --pidfile $PIDFILE
rm -f $PIDFILE
}
do_status()
{
status_of_proc -p $PIDFILE $EXEC $NAME && exit 0 || exit $?
}
```

System V

- Comment démarrer un processus

```
$ /etc/init.d/myService start
```

System V

- Comment arrêter un processus

```
$ /etc/init.d/myService stop
```

System V

- Comment arrêter un processus

```
$  kill $PID
```

System V

- Comment arrêter un processus

```
$  kill $PID
```

System V

- Connaître le status d'un processus

```
$  ps aux | grep service
```

System V

- Connaître le status d'un processus

```
$  /etc/init.d/myService status
```

Les problématiques avec SystemV

- Les dépendances entre services
- Gestion des daemons
- Gestion des processus zombie
- Duplication du code

Les alternatives à SystemV

- Upstart
- OpenRC

La philosophie SystemD

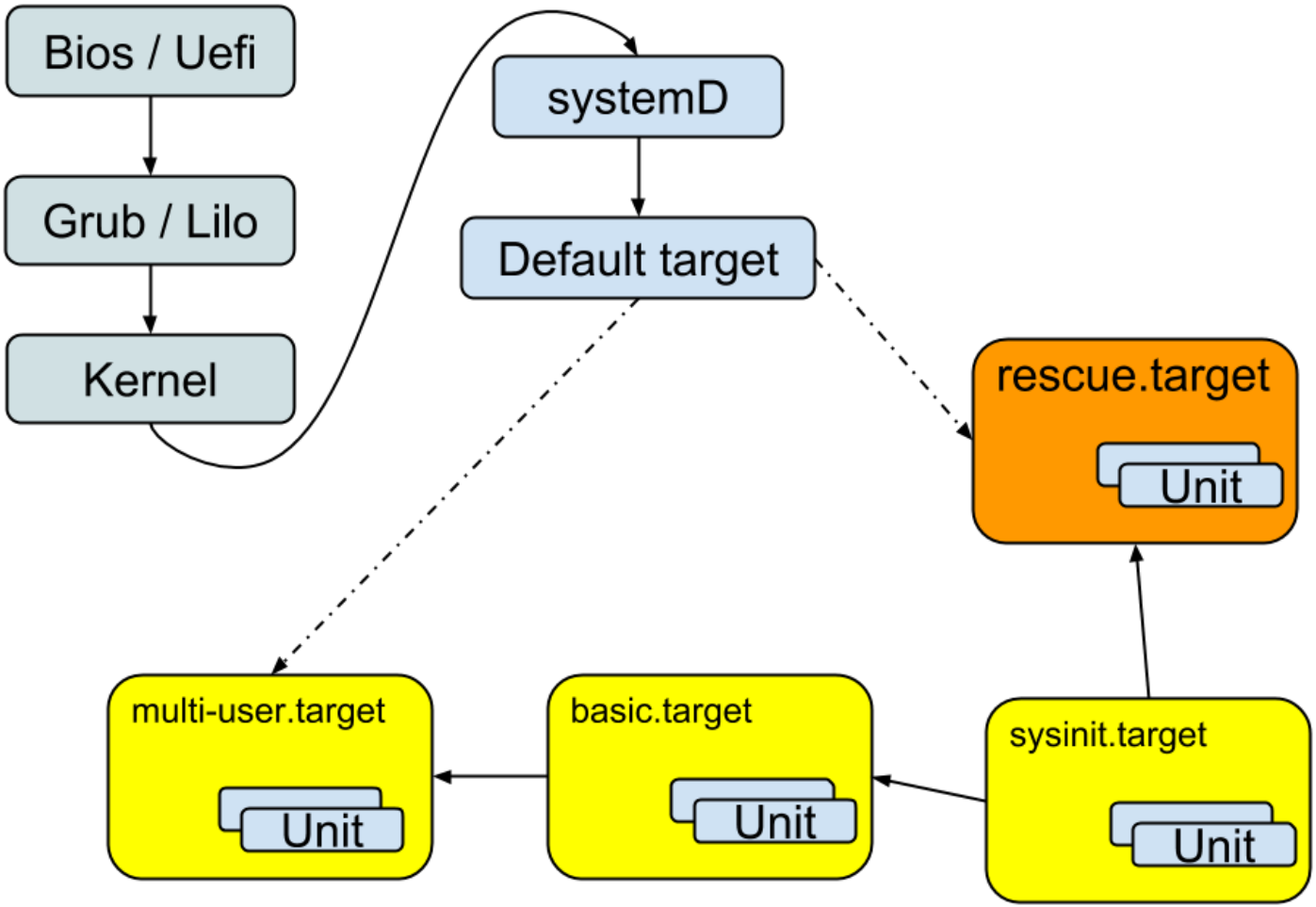
Fiche signalétique

- Projet commence en 2010
- Écrit en C
- maintenu par Lennart Poettering

Adoption

- Fedora / Red Hat / Centos
- Archlinux
- Debian / Ubuntu (en cours)

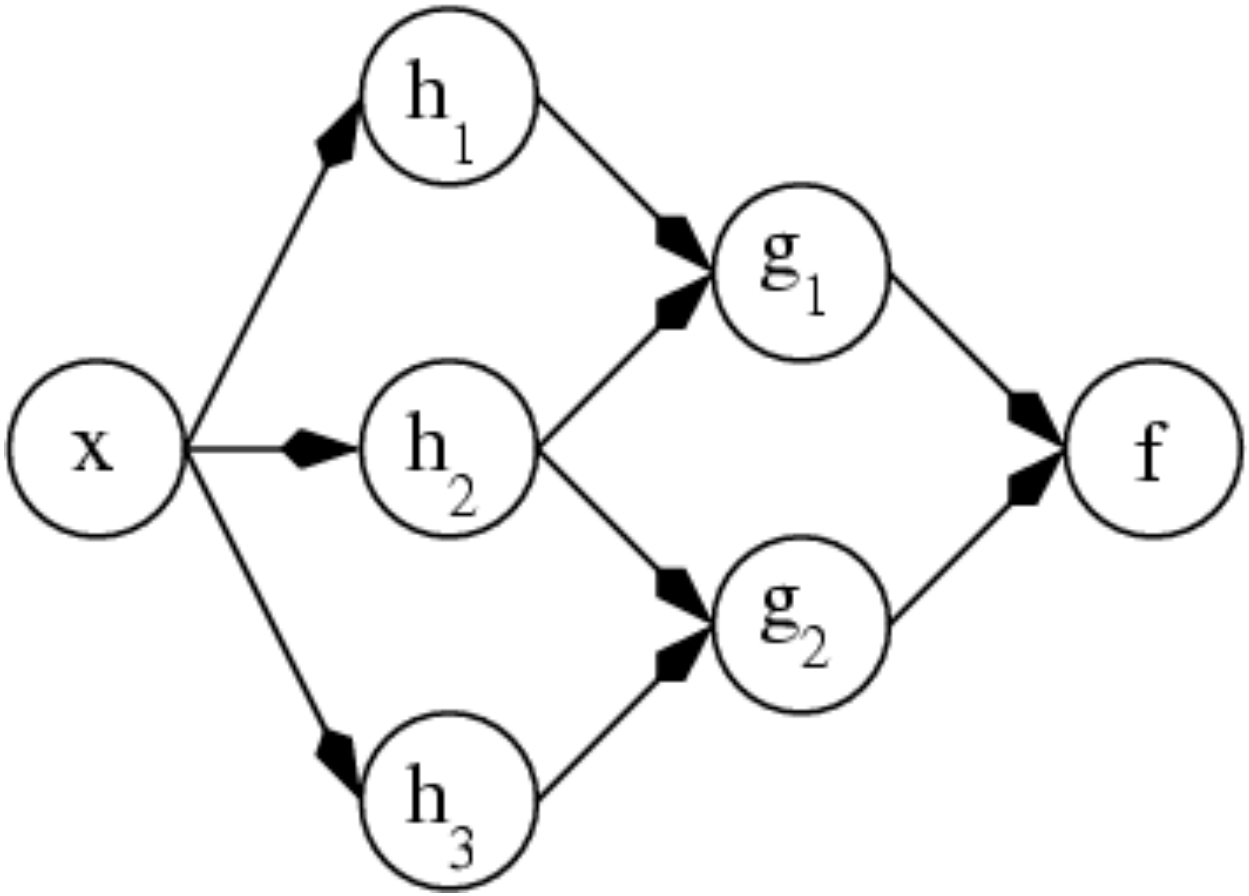
Démarrage


Bios / Uefi
Grub / Lilo
Kernel
systemD
Default target
rescue.target
Unit
multi-user.target
Unit
basic.target
Unit
sysinit.target
Unit

Améliorer la vitesse de démarrage

- Paralléliser le démarrage des services
- Démarrer moins de services

Gérer les dépendances entre services



Est ce vraiment la bonne solution ?

Est ce vraiment la bonne solution ?

Une autre alternative

- Supprimer la dépendance au service
- Dépendre uniquement de la socket

Comment suivre les processus : cgroup

- Regrouper des processus
- Contrôler l'usage des ressources matériels
- Comptabiliser les ressources utilisées

Gestion des logs

- Un nouvel outil : journald
- Récupérer facilement les logs des services
- Requêter plus facilement les logs
- zip, rotation, ...

All is Unit

- Un service : unit
- Monter un système de fichier : unit
- Créer une socket : unit
- Créer un cron : unit
- ...

La boîte à outil systemD

Gérer les services

```
$ systemctl
```

Connaître la liste des services actifs

```
$  systemctl list-units -t service
```

Connaître la liste des services actifs

```
$  systemctl list-units -t service
```

Gestion des unités

```
$ systemctl start <unit>
$ systemctl stop <unit>
$ systemctl restart <unit>
$ systemctl reload <unit>
```

activer une unité au démarrage

```
$ systemctl enable <unit>
$ systemctl disable <unit>
```

Mask une unité

```
$ systemctl mask <unit>
$ systemctl unmask <unit>
```

Lire les logs

```
$ journalctl
```

Lire les logs pour un service donné

```
$ journalctl -u sshd
```

```
$ tail -f /var/log/messages | grep sshd
```

Lire les logs pour un pid donné

```
$ journalctl _PID=1
```

Lire les logs de la dernière heure

```
$ journalctl --since=”2015-06-12 16:30:00”
```

Connaître la liste de tous les services

```
$  systemctl list-units -t service --all
```

De très nombreux critères de recherches

- service
- date
- utilisateur, groupe
- ...

Des formats d'export

- texte
- json

Mais encore

- loginctl
- machinectl
- ...

Les scripts systemD

System V

```
. /lib/lsb/init-functions
NAME=cocktail
EXEC=$CATALINA_HOME/bin/startup.sh
PIDFILE=$CATALINA_BASE/logs/cocktail.pid
export CATALINA_PID=$PIDFILE
do_start()
{
start-stop-daemon --start --quiet --pidfile $PIDFILE --exec $EXEC -c tomcat
}
do_stop()
{
start-stop-daemon --stop --quiet --retry=TERM/30/KILL/5 --pidfile $PIDFILE
rm -f $PIDFILE
}
do_status()
{
status_of_proc -p $PIDFILE $EXEC $NAME && exit 0 || exit $?
}
```

Un exemple simple

```
[Unit]
Description=Une application simple en Node.js

[Service]
ExecStart=/opt/nodejs/bin/node /var/www/myApp/main.js
```

```
$ systemctl start monAppli.service
```

Activation en service

```
[Unit]
Description=Une application simple en Node.js

[Service]
ExecStart=/opt/nodejs/bin/node /var/www/myApp/main.js

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

```
$ systemctl enable monAppli.service
```

Redémarrage automatique

```
[Unit]
Description=Une application simple en Node.js

[Service]
ExecStart=/opt/nodejs/bin/node /var/www/myApp/main.js
Restart=Always

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

Redémarrage automatique (autres options)

- no (par défaut)
- always
- on-success
- on-abort
- on-abnormal

Redémarrage automatique (quand ?)

- Définir un laps de temps : RestartSec
- 100 ms par défaut

Passer en mode production

```
[Unit]
Description=Une application simple en Node.js

[Service]
ExecStart=/opt/nodejs/bin/node /var/www/myApp/main.js
Restart=Always
Environment=NODE_ENV=production

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

Les variables d’environnement

- Utiliser un fichier : EnvironmentFile

Récupérons les logs

```
[Unit]
Description=Une application simple en Node.js

[Service]
ExecStart=/opt/nodejs/bin/node /var/www/myApp/main.js
Restart=Always
Environment=NODE_ENV=production
StandardOutput=journal

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

Récupérons les logs

- null
- tty
- journal
- syslog
- journal+console, syslog+console
- inherit

Récupérons les logs

On a la même option pour la sortie d’erreur : StandardError

Spécifier un user

```
[Unit]
Description=Une application simple en Node.js

[Service]
ExecStart=/opt/nodejs/bin/node /var/www/myApp/main.jsRestart=Always
Environment=NODE_ENV=production
StandardOutput=journal
User=nobody
Group=nobody

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

Utilisons chroot

```
[Unit]
Description=Une application simple en Node.js

[Service]
ExecStart=/nodejs/bin/node /www/myApp/main.js
Restart=Always
Environment=NODE_ENV=production
StandardOutput=journal
User=nobody
Group=nobody
RootDirectory=/opt/myApp

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

Arrêter mon service

```
$  systemctl stop myApp
```

Arrêter mon service

- Utilise un kill par défaut
- tue l’ensemble des processus généré par le processus parent
- Définir une commande ExecStop

Essayons avec un tomcat

```
[Unit]
Description=Une application simple en Java

[Service]
Type=forking
PIDFile=/var/run/tomcat.pid
Environment=CATALINA_PID=/var/run/tomcat.pid
ExecStart=/usr/bin/tomcat start

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

Les autres valeurs de type

- simple (valeur par défaut)
- forking
- oneshot
- dbus, notify, idle

Essayons avec un docker

```
[Unit]
Description=Une application simple avec docker

[Service]
Type=oneshot
ExecStart=/usr/bin/docker run --name busybox1 busybox /bin/sh -c
"while true; do echo Hello World; sleep 1; done"

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

Gestion des dépendances

- Requires, Wants
- Before, After

```
$  systemctl list-dependencies busybox1
```

Des listeners ?

```
[Unit]
Description=Une application simple avec docker

[Service]
Type=oneshot
ExecStartPre=/usr/bin/docker kill busybox1
ExecStartPre=/usr/bin/docker rm busybox1
ExecStartPre=/usr/bin/docker pull busybox1
ExecStart=/usr/bin/docker run --name busybox1 busybox /bin/sh -c
"while true; do echo Hello World; sleep 1; done"

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

Des listeners ?

- ExecStartPre
- ExecStartPost
- ExecStopPost

Mais encore ?

- Démarrer une socket
- Monter un système de fichier
- Démarrer des cron
- Lancer des containers (machinectl)

Pour aller plus loin

- man
- http://0pointer.de/blog/projects/systemd.html (lennart)
- http://www.freedesktop.org/wiki/Software/systemd/

Questions