Arranc i Parada

- *Arrancada del maquinari*
- *Arrancada en Unix System V*
- *Arrancada manual*
- *Procediment de parada*
- *Caiguda del sistema*
- *Personalització dels scripts d'inicialització*
- *Daemons / Serveis debian*

Objectius

L'estudiant ha de ser capaç de :

- Enumerar i descriure el procediments necessaris per realitzar l'arrancada i l'aturada del sistema operatiu.
- Modificar els procediments d'arrancada per variar el comportament del sistema.
- Instal·lar, configurar i activar serveis
- Engegar, aturar, reengegar, rellegir els fitxers de configuració d'un servei
- Programar serveis nous

Arrancada del sistema (bootstrapping)

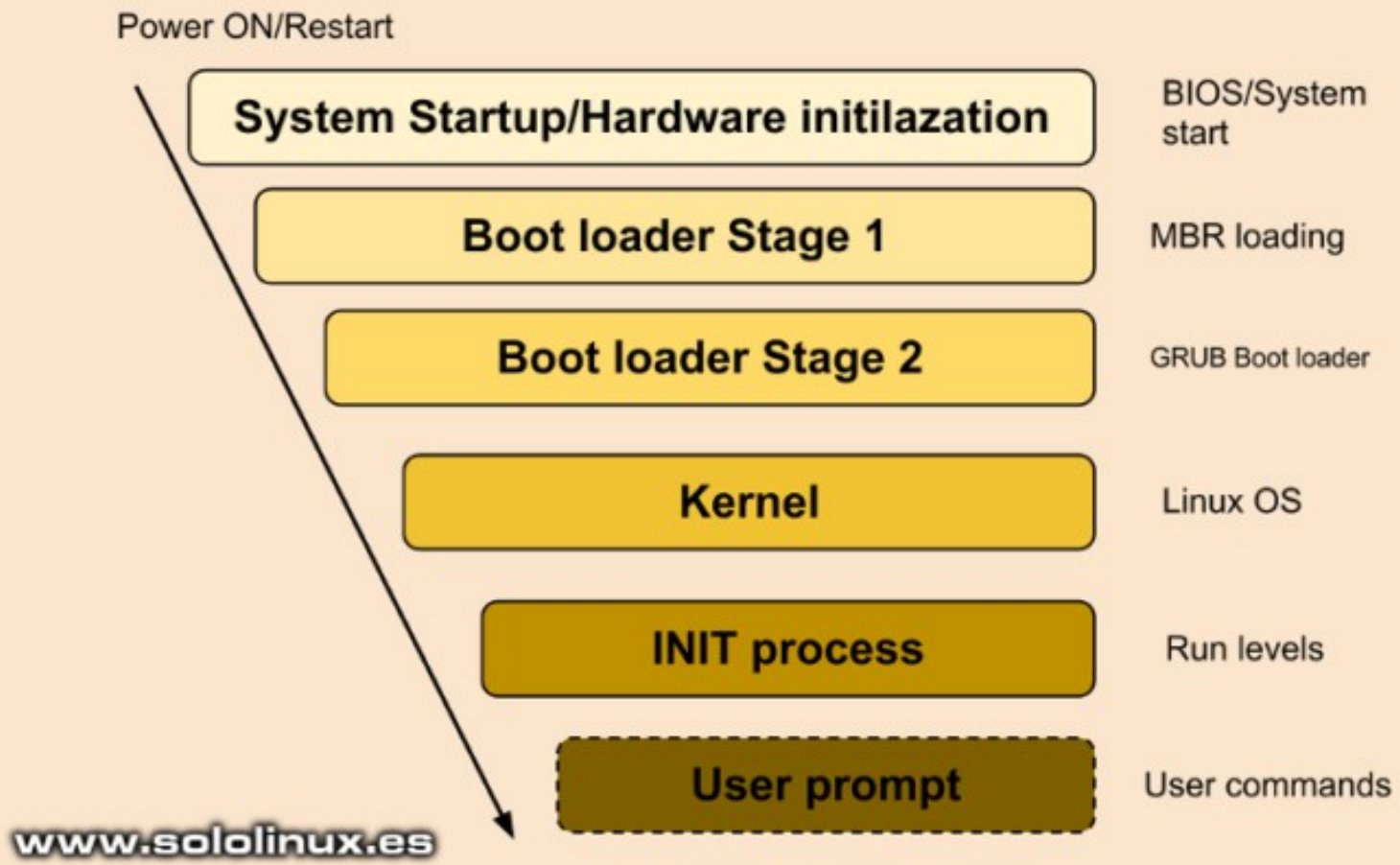
- L'arrencada del sistema es pot dividir en tres fases:

  1) Execució firmware o BIOS a mem no volatil
     - detecta dispositius mínims (disc, teclat, xarxa)
       - Leds o beeps
       - https://en.wikipedia.org/wiki/Power-on_self-test
     - carrega programa de boot (gestor d'arranc) indicat en el MBR (primers 512 B del disc dur)

  2) Execució programa de boot (gestor d'arranc): Lilo o grub
     - verifica i detecta resta de hardware
     - carrega el nucli del SO (el de linux, localitzat a /kernel anomenat unix, vmlinux, vmlinuz,...)

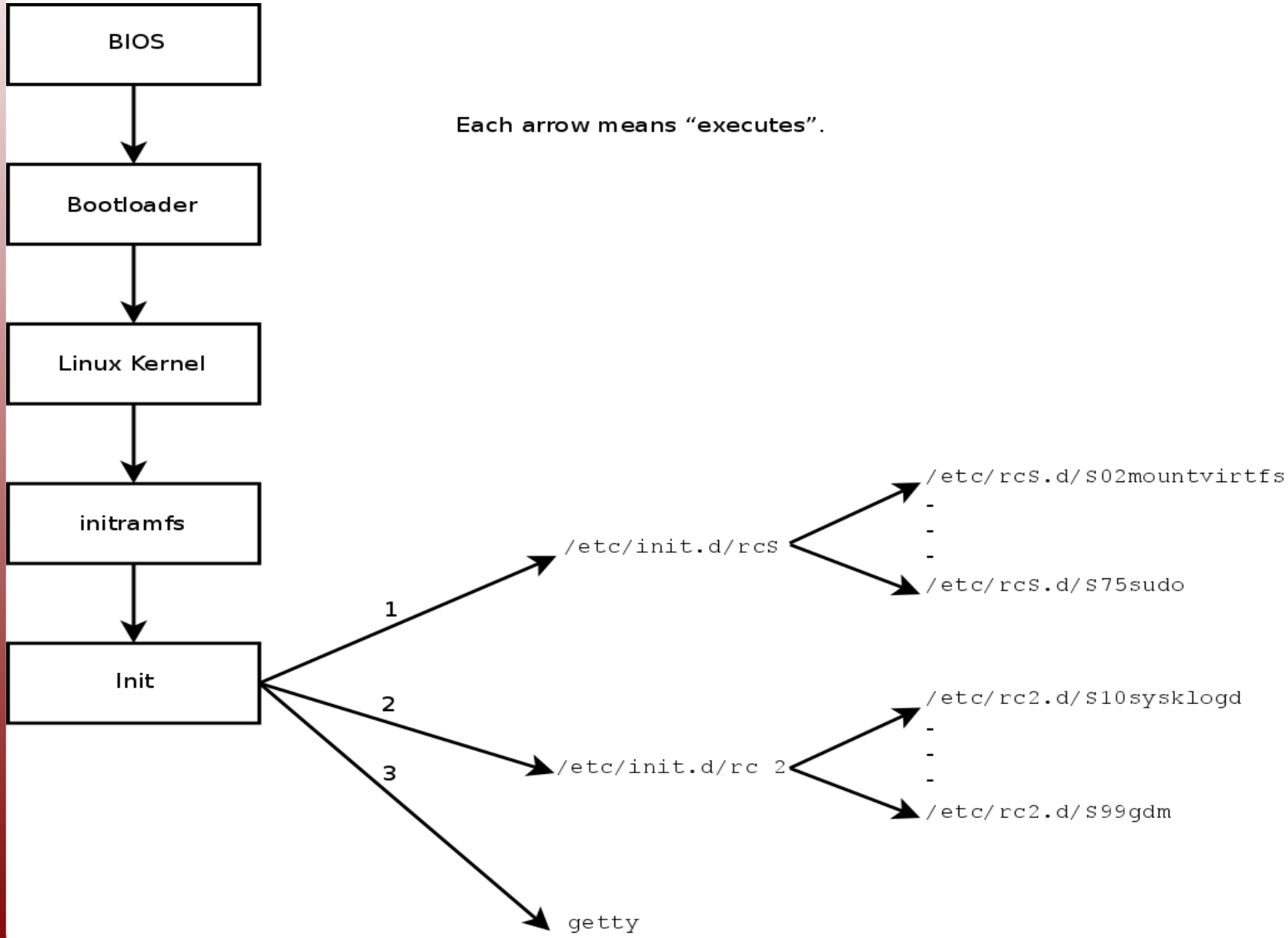
  3) Execució del Sistema Operatiu – programa INIT.

Procés d'arrancada de linux



https://www.sololinux.es/como-arranca-linux-explicado-paso-a-paso/

Processos que formen el Sistema de boot



Procés d'arrancada en un PC:

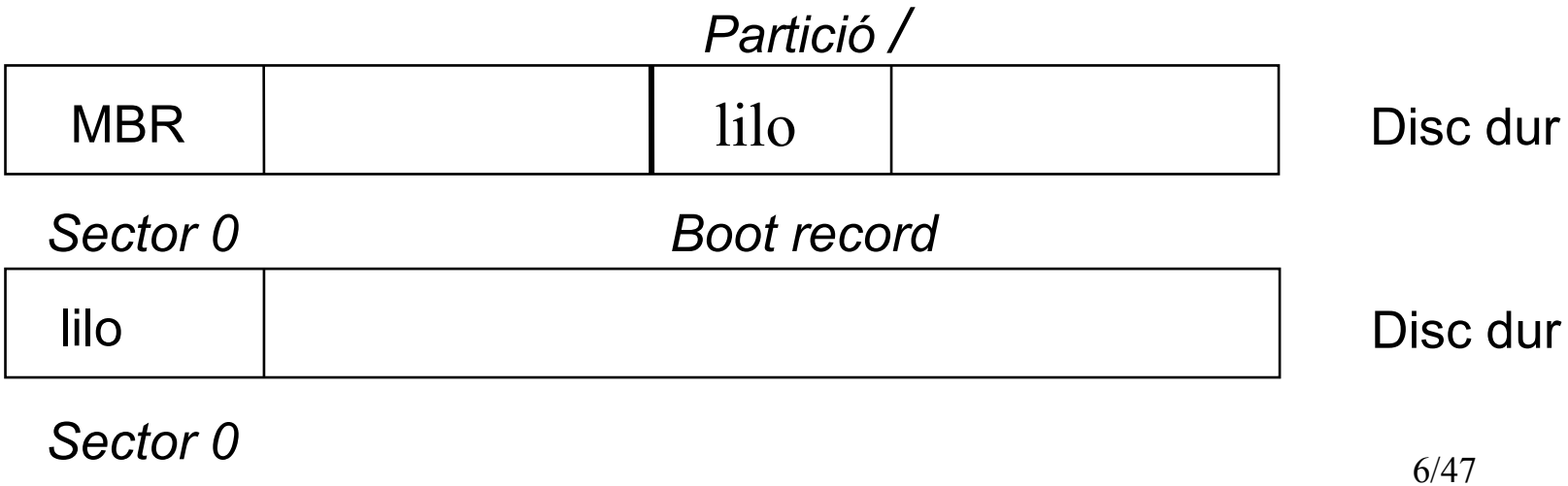
Format per tres o quatre fases:

1) Execució BIOS: carrega master boot (situat al MBR).

2) Execució master boot: carrega boot secundari(lilo,grub).

3) Execució boot secundari: carrega el sistema operatiu.

4) Execució del nucli del sistema operatiu.



Configuració del carregador lilo

- En l'arxiu */etc/lilo.conf*:

```
default=linux
boot=/dev/sda
prompt
timeout=50
image=/boot/vmlinuz-2.2.5-15
            label=linux
            initrd=/initrd.img
            read-only
image=/boot/vmlinuz-2.2.5-12
            label=linux2
            initrd=/initrd.img.old
            read-only
other=/dev/sda1
            label=win2k
            table=/dev/sda
```


http://www.juntadeandalucia.es/empleo/recursos/material_didactico/especialidades/materialdidactico_administrador_servidores/Content/1-so/4-AnexoSistemasArranque.pdf

Configuració del carregador grub 2

tutorial del grub 2

- instal·lar-lo: `grub-install /dev/sd`
- fitxers de configuració:
  - /boot/grub/device.map
  - /etc/grub.d -> scripts del grub
  - /etc/default/grub
  - /boot/grub/grub.cfg

- update-grub

Arrancada del Sistema Operatiu

El sistema operatiu inicialment és l'encarregat de:

- ✓ Inicialitzar els controladors de dispositiu (drivers)
- ✓ Inicialitzar les taules internes (PCB, descriptors de dispositius...)
- ✓ Crea els processos que gestionen el sistema:
  - BSD:
    - Swapper (PID=0)
    - Init (PID=1)
  - System V
    - Scheduler (PID=0)
    - Init (PID=1)
  - Linux:
    - Init (PID=1)
    - Systemd
- ✓ Init / systemd posa en marxa la resta de processos

  És el pare de tots els processos del sistema

El procés init (1/2)

Té dos modes bàsics de funcionament:

Mode multiusuari:

- ✓ Munta el S.F. / i els S.F. essencials i verifica la integritat dels S.F
- ✓ Munta els S.F. Locals
- ✓ Activa la paginació ⇒ `swapon`
- ✓ scripts de neteja (/tmp, /var)
- ✓ Verifica consistència fitxer quotes i activa quotes
- ✓ Configura interfície xarxa
- ✓ Activa serveis del sistema: D-bus, cron, serveis xarxa, ..
- ✓ Munta discs remots
- ✓ Activa dimoni *getty permeten l'entrada d'usuaris*

El procés init (2/2)

Mode monousuari

- ✓Munta el S.F. /
- ✓No posa en marxa els serveis (dimonis)
- ✓Posa en marxa una Bourne Shell (sh) amb privilegis de root
- ✓S'executa `sulogin`: demana password de root (SysV)

En linux mode desprotegit, a no ser que existeixi el fitxer /etc/securesingle o configurat a /etc/inittab.

Automàticament en mode multiusuari, però quan es detecta un problema en l'arranc ⇒ mode monosuari.

El procés init – model BSD

Procés Init executa els scripts per configurar el sistema

- Organització dels scripts segons BSD:
  - ✓scripts localitzats a /etc o /sbin
  - ✓El script principal es rc: rc.boot, rc.network, ...
  - ✓Els fitxers de configuració de rc:
    - /etc/rc.conf
    - /etc/rc.conf.local
  - ✓Arranc en mode monousuari i multiusuari

El procés init – model sysV

Procés Init executa els scripts per configurar el sistema

- Organització dels scripts segons la sysV
  - ✓El fitxer de configuració principal és /etc/inittab
  - ✓Scripts localitzats a /etc/rc.d
  - ✓Hi ha 7 nivells de run possibles
  - ✓En /etc/rcN.d hi ha un soft link dels serveis a executar on:
    - La primera lletra (S,K) indica si engegar o parar
    - El número de 2 dígits indica ordre
    - El nom del link fa referència a què fa

El procés init – model sysV

Procés Init executa els scripts per configurar el sistema

- Organització dels scripts segons la sysV
  - ✓El fitxer de configuració principal és /etc/inittab
  - ✓Scripts localitzats a /etc/rc.d
  - ✓Hi ha 7 nivells de run possibles
  - ✓En /etc/rcN.d hi ha un soft link dels serveis a executar on:
    - La primera lletra (S,K) indica si engegar o parar
    - El número de 2 dígits indica ordre
    - El nom del link fa referència a què fa

Arranc en Unix SysV

- ***Nivells d'execució (1/2)***
  - Nivell d'execució: mode de funcionament del sistema.
  - Cada nivell d'execució identificat per un nº o lletra:

Nivell	*mode*
0	Parada del sistema
1, s, S	Mode monousuari
2	Mode multiusuari
3	Mode multiusuari+serveis xarxa
4	Mode multiusuari+servei estadístiques
5	Mode reservat
6	Parada i arranc (reboot)

Nivells d'execució (2/2)

- Les accions a realitzar en cada nivell en /etc/inittab.
- Indica en quin nivell s'ha de engegar
- Per canviar de nivell

  ⇒`telinit` <nivell> o `init`
- Per saber el nivell d'execució actual:

  `$who -r` ó `runlevel` (linux)

- Per a que init rellegeixi el fitxer inittab:

  `$kill -SIGHUP 1`

  ⇒`telinit q`

El fitxer de configuració /etc/inittab

- Format: etiqueta:llista_nivells:acció:programa

```
id:3:initdefault:
si::sysinit:/etc/init.d/rc.sysinit
l0:0:wait:/etc/init.d/rc 0
l1:1:wait:/etc/init.d/rc 1
l2:2:wait:/etc/init.d/rc 2
l3:3:wait:/etc/init.d/rc 3
l4:4:wait:/etc/init.d/rc 4
l5:5:wait:/etc/init.d/rc 5
l6:6:wait:/etc/init.d/rc 6
ud::once:/sbin/update
ca::ctrlaltdel:/sbin/shutdown -t3 -r now
pf::powerfail:/sbin/shutdown -f -h +2 "Power
            Failure; System Shutting Down"
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6
x:5:respawn:/etc/X11/prefdm -nodaemon
```

Organització dels script d'inicialització

- Init consulta el fitxer /etc/inittab quan:
  - ✓Arranca el sistema
  - ✓Un dels processos que crea acaba
  - ✓Al rebre SIGPWR, SIGINT (Ctrl+ALT+DEL), SIGHUP.
  - ✓Al canviar de nivell d'execució

- Scripts
  - ✓Programació del servei
  - ✓Posar script en /etc/init.d
  - ✓Fer soft link en el directori del nivell corresponent
    - update-rc.d

Nou sistema d'inici: Systemd

- Gestor de sistema i serveis per a Linux
- Compatible amb els serveis del SysV o init
- Paral·lelitza els processos
- Utilitza D-bus i Sockets per activar serveis
- Executa daemons a demanda
- Optimitza l'ús de recursos usant cgroups
- Suporta snapshots i restauració del sistema a un punt definit
- Administra punts de muntatge i munta unitats d'emmagatzemament

https://wiki.archlinux.org/index.php/systemd_(Español)

Targets similars al Runlevel

- Targets amb noms en lloc de números
- Proposit especific amb possibilitat de fer més d'una acció al mateix temps en paral·lel
- Poden heretat serveis d'altres targets i implementar serveis addicionals
- Podem passar d'un target a un altre usant
  - telinit RUNLEVEL
  - systemctl isolate level.target

Equivalència Runlevel - Targets

- 0: runlevel0.target, poweroff.target
- 1,s,single: runlevel1.target, rescue.target
- 2,4 : runlevel2.target, runlevel4.target, multi-user.target
- 3: runlevel3.target, multi-user.target
- 5: runlevel5.target, ghaphical.target
- 6: runlevel6.target, reboot.target
- Emergency: emergency.target

- **Treballa amb unitats, cada component una unitat**
  - ✓ Arxiu que conté informació del que es vol activar
  - ✓ sufixes possible: .service, .mount, .socket, .device
  - ✓ Segueix la sintaxi: nom_unitat.sufixe Exemple:sshd.socket
- **Comandes: systemctl**
  - ✓ list-units | list-unit-files
  - ✓ start | stop | restart | reload | status | enable | disable | help unitat
  - ✓ Systemctl --failed
  - ✓ Exemple: sudo systemctl disable bluetooth.service
  - ✓ Hostnamectl
  - ✓ Timedatectl
  - ✓ Journalctl
  - ✓ Systemd-analyze / systemd-analyze time

Target predeterminat pel boot

- l'estàndart és: systemctl set-default multi-user.target
- Per canviar-lo, en el gestor d'arranc
  - systemd.unit=multi-user.target
- Canviar el target de defecte:
  - systemctl enable multi-user.target
  - Actualitza el fitxer /etc/systemd/system/default.target
- Quin nivell executo?
  - systemctl list-units –type=target

Aturar i engegar el sistema: systemctl

- systemctl reboot
- systemctl poweroff
- systemctl suspend | hibernate

- Serveis personalitzats:
  - Sintaxi d'arxius d'unitats inspirats en els arxius .ini de Microsoft
  - Dependències: en els arxius .target dins la secció [Unit]
    - Exemple: unitat A necessita/opcional la unitat B per a funcionar
    - Require=B | Wants=B
    - After=B
  - Type: tipus d'arranc
  - Dependències: en els arxius .target dins la secció [Service]
    - Simple | forking | oneshot | notify | dbus

Exemple servei ssh

```
[Unit]
Description=OpenBSD Secure Shell server
After=network.target auditd.service
ConditionPathExists=!/etc/ssh/sshd_not_to_be_run

[Service]
EnvironmentFile=-/etc/default/ssh
ExecStart=/usr/sbin/sshd -D $SSHD_OPTS
ExecReload=/bin/kill -HUP $MAINPID
KillMode=process
Restart=on-failure

[Install]
WantedBy=multi-user.target
Alias=sshd.service
```

```
systemctl start ssh.service
journalctl -u ssh.service
systemctl status ssh.service
```

Errors de systemd

- systemctl --state=failed
- systemctl status systemd-modules-load
- journalctl -b _PID=15630
- ls -Al /etc/modules-load.d/
  - Arreglar/comentar els serveis que poden donar problemes
- Reiniciar: systemctl start systemd-modules-load.service

- Serveis no existents que esperen un temporitzador

Què es D-Bus?

- Sistema de comunicació entre processos (IPC). Es comporta com una crida a procediment remot (RPC).
- Funciona amb tres capes
  - Una llibreria (libdbus) permet a dos aplicacions connectar-se i intercanviar missatges
  - Un dimoni al que poden connectar-se les aplicacions. El dimoni encaminarà els missatges des d'una aplicació a cap o a més aplicacions.
  - Llibreries adaptades per l'ús en entorns concrets.

- Quin ús:
  - Comunicació entre diverses aplicacions dins d'una mateixa sessió.
  - Comunicació entre el nucli, serveis i la sessió

Arranc manual

- Quan l'auto-arranc no configurat
- L'arranc del sistema s'atura abans de carregar el sistema operatiu
- Apareix el prompt
- S'ha d'introduir la comanda:
  - `boot, b` ó <CR>
  - Exemple:
    - boot –s (Solaris)
    - linux single (linux)
- L'auto-arranc es configura al carregador del sistema operatiu

init=/bin/sh

Procediment de parada (shutdown) (1/2)

- Parar el sistema de manera correcta
- Parar per realitzar canvis en el hardware
- Rearrancar per activar els canvis en scripts d'inicialització del sistema, actualitzar S.O, etc.
- Per aturar el sistema de manera correcta:
  - ✓notificar als usuaris de la parada
  - ✓enviar senyal de finalització als processos
  - ✓aturar els serveis del sistema
  - ✓treure als usuaris i matar als seus processos
  - ✓mantenir la integritat del S.F. (⇒ `sync`)
  - ✓aturar el sistema (`ls halt`).

Procediment de parada (shutdown) (2/2)

Per rearrancar el sistema, aturar-lo o passar a mode monousuari ⇒ `shutdown`

- Permet indicar el temps d'espera
- Es pot avortar matant al procés shutdown
- Només el pot executar root
- En alguns sistemes el pot executar qui estigui al fitxer */etc/shutdown.allow*

- Altres maneres d'aturar el sistema:
  - ✓ `sync;sync;sync;killall;telinit 0`
  - ✓ `sync;sync;sync;killall;halt`

Caiguda del sistema (crash)

- Causes que ho provoquen:
  - ✓falles hardware: controladors, memòria, etc
  - ✓falles software: errors kernel (fatal error)
  - ✓bloqueig de recursos
  - ✓falles a tensió elèctrica
  - ✓falles a l'entorn: aire condicionat (màx 30..35º)
  - ✓problemes en la E/S: disc

- Consultar el fitxer /var/log/messages o `dmesg`
- Abans de caiguda es fa bolcat de memòria a àrea swap

  ⇒ `crash`, abans guardar amb `savecore` fitxer.

Personalització dels scripts d'inicialització

Regles per modificar els scripts:

- ✓Al afegir línies, aïllar els canvis, línies del nou script
- ✓No eliminar línies, només comentar-les
- ✓Abans de modificar un script copia de seguretat:

```
cp -p /etc/script /etc/script.save
chmod a-w /etc/script.save
```

- ✓La còpia de seguretat en d'altres dispositius
- ✓Mantenir els atributs del script!.

Daemons.

- Un daemon és un procés que s'executa en background
- Proporciona un servei
- S'inicia al engegar-se al sistema o manualment
- Pot realitzar les següents accions:
  - ✓Start
  - ✓Stop
  - ✓Restart
  - ✓Reload
  - ✓Force-reload

Serveis debian / Serveis Windows

- Comandes linux:
  - ✓service --status-all
  - ✓service nom_servei status
  - ✓service nom_servei stop
  - ✓service nom_servei start
- Comandes powerShell :
  - ✓Get-Service
  - ✓Get-Service nom_servei Format-List *
  - ✓Stop-Service nom_servei
  - ✓Start-Service nom_servei

Programar un dimoni. SysVinit (SysV)

- Copiar d'un servei que hagi a /etc/init.d/
- Modificar la capçalera en funció dels nivells que vulguem engegar o parar el servei
- Modificar el valor de les variables d'entorn definides, hi ha dimoni associat al servei?
- Modificar les comprovacions dels valors de les variables d'entorn
- Modificar les funcions que realitza el servei
- instal·lar el dimoni en els directoris corresponents
  - ✓Insserv update-rc.d/rcconf

Capçalera del script

```
#! /bin/sh  -x
### BEGIN INIT INFO
# Provides:          skeleton
# Required-Start:    $remote_fs $syslog
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
 # Short-Description: Example initscript
 # Description:       This file should be used to construct scripts to be
 #                    placed in /etc/init.d.
 ### END INIT INFO
```

- **Dóna la informació pel update-rc.d**
- **Per exemple:**
  - ✓ update-rc.d servei start 20 2 3 4 5 . start 30 5 . stop 80 0 1 6 .

Cos del script

```
case "$1" in
 start)
      do_start
      ;;
 stop)
   do_stop
   ;;
 status)
      status_of_proc "$DAEMON" "$NAME" && exit 0 || exit $?
      ;;
 #reload|force-reload)
      ;;
 restart|force-reload)
   do_stop
   do_start
   ;;
 *)
   #echo "Usage: $SCRIPTNAME {start|stop|restart|reload|force-reload}" >&2
   echo "Usage: $SCRIPTNAME {start|stop|status|restart|force-reload}" >&2
   exit 3
   ;;
esac
```

funció del script

```
do_start()
 {
    # Return
    #   0 if daemon has been started
    #   1 if daemon was already running
    #   2 if daemon could not be started
    start-stop-daemon --start --quiet --pidfile $PIDFILE --exec $DAEMON& --test
> /dev/null \
    start-stop-daemon --start --pidfile $PIDFILE --exec $DAEMON& -- \
       $DAEMON_ARGS \
       || return 2
    # Add code here, if necessary, that waits for the process to be ready
    # to handle requests from services started subsequently which depend
    # on this one.  As a last resort, sleep for some time.
 }
```

Exercicis

- ✓ De quina manera es posen en marxa els diferents serveis que dóna el sistema operatiu i des de quin script ?
- ✓ Justifiqueu el contingut dels directoris i subdirectoris de */etc/rc<nivell>.d*.
- ✓ Que faríeu per afegir un nou servei en un determinat nivell d'execució ? I per eliminar-lo ?
- ✓ Com estan organitzats els scripts d'inicialització del sistema?

part principal del script */etc/init.d/rc*

```
#!/bin/bash
argv1="$1"
set `/sbin/runlevel`
runlevel=$2
previous=$1
export runlevel previous

[ -n "$argv1" ] && runlevel="$argv1"

# Is there an rc directory for this new runlevel?
if [ -d /etc/rc$runlevel.d ]; then
    # First, run the KILL scripts.
    for i in /etc/rc$runlevel.d/K*; do
        # Check if the script is there.
        [ ! -f $i ] && continue
            $i stop
    done

    # Now run the START scripts.
    for i in /etc/rc$runlevel.d/S*; do
            # Check if the script is there.
            [ ! -f $i ] && continue
               $i start
    done
fi
```

man runlevel

RUNLEVEL(8) Linux System Administrator's Manual RUNLEVEL(8) **NAME**

runlevel -- find the current and previous system runlevel. **SYNOPSIS**

runlevel [*utmp*]

DESCRIPTION Runlevel reads the system *utmp* file (typically */var/run/utmp*) to locate the runlevel record, and then prints the previous and current system runlevel on its standard output, separated by a single space. If there is no previous system runlevel, the letter **N** will be printed instead. If no *utmp* file exists, or if no runlevel record can be found, **runlevel** prints the word **unknown** and exits with an error. **Runlevel** can be used in *rc* scripts as a substitute for the System-V **who -r** command. However, in newer versions of init(8) this information is also available in the environ- ment variables **RUNLEVEL** and **PREVLEVEL**. **OPTIONS**

utmp The name of the *utmp* file to read.

SEE ALSO init(8), utmp(5)

AUTHOR Miquel van Smoorenburg, miquels@cistron.nl May 27, 1997

man bash

...

set [**--aefhknoptuvxldCH**] [*arg* ...]

n args are treated as values for the positional parameters and are assigned, in order, to $1, $2, ... $n. If no options or args are supplied, all shell variables are printed. The return status is always true unless an illegal option is encountered.

...

Exemple execució: `/sbin/runlevel`

```
N 3
```

Contingut /etc/init.d/crond pàgina 1

```
#! /bin/bash
RETVAL=0
# See how we were called.
prog="crond"
start() {
      echo -n $"Starting $prog: "
      daemon crond
      RETVAL=$?
      echo
      [ $RETVAL -eq 0 ] && touch /var/lock/subsys/crond
      return $RETVAL
}
stop() {
      echo -n $"Stopping $prog: "
      killproc crond
      RETVAL=$?
      echo
      [ $RETVAL -eq 0 ] && rm -f /var/lock/subsys/crond
      return $RETVAL
}
```

Contingut /etc/init.d/crond pàgina 2

```
rhstatus() {
   status crond
}

restart() {
   stop
   start
}

reload() {
   echo -n $"Reloading cron daemon configuration: "
   killproc crond -HUP
   retval=$?
   echo
   return $RETVAL
}
```

Contingut /etc/init.d/crond pàgina 3

```
case "$1" in
  start) start
         ;;
  stop)  stop
        ;;
  restart)restart
         ;;
  reload) reload
         ;;
  status) rhstatus
         ;;
  condrestart)
   [ -f /var/lock/subsys/crond ] && restart || :
         ;;
  *)
   echo $"Usage: $0 {start|stop|status|reload|
   restart|condrestart}"
   exit 1
esac
exit $?
```

Bibliografia

- Cap.4 “Essential System Administration”
- Cap. 2 “Unix System Administration Handbook”
- Cap. 8 i 9 “Administración Unix. System V. Redes TCP/IP”