Arranque y parada del sistema

Joaquín Seoane Pascual
joaquin@dit.upm.es

Departamento de Ingeniería de Sistemas Telemáticos
Universidad Politécnica de Madrid

5 de noviembre de 2001

Índice General

Proceso de arranque

- Una (P)ROM (BIOS) toma control.
- Se carga y ejecuta algo de disco (sector 0) o de red.
- Puede haber varios pasos en disco.
- Toma control el núcleo (se le pueden pasar parámetros).
- Toma control el arrancador de servicios (`init`).
- Se arrancan ordenadamente servicios hasta un *nivel*.

Arranque inicial en PROM

- Comprueba hardware básico.
- Puede comprobar (P)ROMs auxiliares.
- Suele haber una memoria no volátil con configuración.
- Da control a cargador en disquete, disco, CD-ROM o red.

Arranques de diskete

- Directo:

  `cp vmlinuz-2.2.19 /dev/fd0 .`

  – Sistema de ficheros raíz en otro sitio (hay que informar donde está): `rdev /dev/fd0 /dev/hda1`.

  – No se le pueden pasar otros parámetros.

- Indirecto:

  – Cargador de disquete.

  – Sistema de ficheros con núcleo, cargador del núcleo parametrizable y quizá disco RAM:

    * `syslinux.sys` con `syslinux.cfg`.
    * `MSDOS` y `loadlin.exe` y sus parámetros,
    * Se puede adornar el arranque, elegir núcleo y pasarle parámetros.

Arranques de disco duro

Casi siempre es indirecto:

- Puede haber cargador para elegir sistema operativo.
- Cada sistema operativo puede tener su cargador en partición.
- Un cargador puede conocer o no el sistema de ficheros (eg: `grub` frente a `lilo`).
- Puede haber parámetros (en disco o interactivos).

Arranques de CD-ROM

Siempre es indirecto:

- Formato *El Torito*.
- Una o varias imágenes de disquete de 1.44 o 2.88 Meg.
- Luego como arranque de disquete.

Arranques de Red

Siempre es indirecto:

- Parámetros por `bootp` o `dhcp`.
- Transferencia por `tftp` o extensión.
- Se da control a lo que llegue.

Parámetros de arranque

Un cargador recibe parámetros para:

- Sí mismo:
  - Globales (modo de pantalla, consola, menú, ...)
  - Para cada imagen (lugar, selector, ...)
- Núcleo (si lo conoce: partición raíz, modo de montaje, puertos y direcciones reservados, RAM, parámetros de manejadores, programa inicial).
- Primer proceso (`init`: monousuario o multiusuario, nivel).
- Procesos sucesivos: variables de entorno.

Cargador `lilo`

- Configurable por fichero, con parámetros de carga e instrucciones al configurador.
- Sabe cargar y pasar parámetros a núcleo de linux.
- No conoce sistema de ficheros de linux: requiere mapa de bloques.
- El cargador carga el núcleo (comprimido) con llamadas a la BIOS.
- Se descomprime el núcleo y se le da control.
- El núcleo recibe parámetros: la partición raíz, programa inicial, etc..
- Arranaca programa inicial o `init` (proceso 1) pasando modo (`single` o `auto` o nivel determinado).

Configuración de `lilo`

- Configuración en `/etc/lilo.conf`. Ejemplo:
- ```
  prompt                # siempre pregunta
  timeout=50            # espera de respuesta
  message=/etc/mensajelilo
  boot=/dev/hda1        # lugar de lilo
  image=/vmlinuz        # núcleo a cargar
    root=/dev/hda1      # raiz de sistema de ficheros
    read-only           # modo de montaje de la raíz
    append="mem=127M"   # parámetros al núcleo
    label=Linux         # nombre
  other=/dev/hda2       # otro sistema operativo
    label=Win           # nombre
  ```
- Editar `/etc/mensajelilo`

  ```
  Seleccione Linux o Win.
  ```

- Ejecutar `lilo` y rearrancar (`CONTROL-ALT-SUPR`).
  ```

El proceso inicial

- Lanzado por el núcleo.
- Lanza los demás.
- Padre de todos los procesos.
- Puede ser cualquier programa llamado `init` o el que se le diga al núcleo.
- El más usado y sofisticado es el de Sistema-V.
- Se acostumbra a implementar *niveles de ejecución*:
  - El sistema está en un *nivel de ejecución* en cada momento:
  - Un *nivel de ejecución* determina qué procesos bajo su control pueden existir.

El `init` de Sistema-V

- Niveles:

S	Monousuario de mantenimiento
1	Para pasar a monousuario
2-5	Multiusuario
0	Para parar
6	Para rearrancar
A, B, C	Seudoniveles de mantenimiento

- El cambio manda ejecutar procesos y luego manda señales `SIGTERM` (terminación ordenada) y, a los 5 segundos, `SIGKILL` (asesinato) a procesos fuera del nivel.
- Los procesos lanzados reciben nivel actual (`RUNLEVEL`) y previo (`PREVLEVEL`).
- Se fuerzan niveles con `telinit`.
- Configurable con `/etc/initab`.

/etc/inittab

Líneas de la forma: `etiqueta:niveles:acción:proceso`

- La *etiqueta* identifica la línea.
- Los *niveles* enumeran aquéllos donde se aplica.
- La *acción* determina lo que se hace con el proceso:
  - `wait` ejecuta una vez y espera.
  - `once` ejecuta una vez y no espera.
  - `respawn` rearranca cuando termina.
- Ignoran nivel:
  - `initdefault`: nivel por omisión (si no la hay, lo pregunta).
  - `sysinit` se ejecuta al leer `inittab`.
  - `bootwait` y `boot` después de `sysinit` (no Debian).
  - `ctrlaltdel` cuando `SIGINT` (la manda el núcleo).
  - `powerwait, powerfail, powerfailnow, powerokwait` ...

/etc/inittab de Debian

```
id:2:initdefault:                 # Normalmente a nivel 2
si::sysinit:/etc/init.d/rcS       # Arranque (excepto emergencia)
~~:S:wait:/sbin/sulogin           # Monousuario
l0:0:wait:/etc/init.d/rc 0        # Niveles numéricos
l1:1:wait:/etc/init.d/rc 1        # Gestionados fuera
l2:2:wait:/etc/init.d/rc 2
l3:3:wait:/etc/init.d/rc 3
....
1:2345:respawn:/sbin/getty 38400 tty1
2:23:respawn:/sbin/getty 38400 tty2
....
ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now
pf::powerwait:/etc/init.d/powerfail start
pn::powerfailnow:/etc/init.d/powerfail now
po::powerokwait:/etc/init.d/powerfail stop
```

Programillas de arranque y parada

- Todos los servicios deben poder arrancarse, pararse y, posiblemente recargar configuración o rearrancarse.
- Al cambiar de nivel, `init` debe parar los servicios que no deben estar y arrancar los que deben estar.
- Para uniformizar el arranque y la parada, cada servicio tiene un programilla en `/etc/init.d`.
- Los programillas de `/etc/init.d` deben reaccionar a los parámetros `start` y `stop`, aunque suelen tener más: `restart`, `reload`.
- Debe poderlos ejecutar `init` cuando proceda.
- Los servicios tienen un orden natural de arranque y parada.

Enlace con `init`

- A través de programillas de la `inittab`: `/etc/init.d/rcS` y `/etc/init.d/rc`.
- Reciben como parámetro un nivel.
- Ejecuta todos los `/etc/rc`*nivel*`.d/K??*` en orden con parámetro `stop`.
- Ejecuta todos los `/etc/rc`*nivel*`.d/S??*` de los servicios no parados en orden con parámetro `start`.

Mantenimiento de programillas de arranque y parada

- Todos los `/etc/rc?.d/*` son enlaces simbólicos a ficheros de `/etc/init.d`
- Todo paquete instala programillas de arranque y parada en `/etc/init.d` y luego actualiza enlaces
- Los programillas de `/etc/init.d` deben reaccionar a los parámetros `start` y `stop`, aunque suelen tener más: `restart`, `reload`
- Los programillas de `/etc/init.d` son los que se utilizan para controlar servicios
- Idealmente deberían ser invariables y usar ficheros de configuración externos.
- Los enlaces simbólicos se mantienen con editores de niveles de ejecución

```
update-rc.d [-n] [-f] <basename> remove
update-rc.d [-n] [-f] <basename> defaults [NN | sNN kNN]
update-rc.d [-n] [-f] <basename> start|stop NN runlvl runlvl .  ...
    -n: not really
    -f: force
```

Arranque de servicios de red bajo demanda

- Algunos servicios de red se arrancan desde el principio, pero:
  - Puede haber muchos posibles: sobrecarga de procesos.
  - Deben implementar, cada uno de ellos, una política de seguridad que hay que configurar.
  - Un servicio mal implementado puede caerse.
- Uso de *superservidores* de internet:
  - Escuchan en una serie de puertos configurables.
  - Conocen qué programas los atienden.
  - Los arrancan bajo demanda y los mantienen el tiempo necesario.
  - Los pueden proteger sistemáticamente.

Algunos superservidores

- Combinación de `inetd` y `tcpd`, configurables con `/etc/inetd.conf` y `/etc/hosts.allow` y `/etc/hosts.deny`.
- Sistema monolítico como `xinetd`, que permite configurar por servicios en `/etc/xinetd.d/`*servicio*.

Ejemplo de `inetd.conf`

```
daytime stream  tcp  nowait  root \
    internal
telnet  stream  tcp  nowait  telnetd.telnetd  \
    /usr/sbin/tcpd  /usr/sbin/in.telnetd
pop-3   stream  tcp  nowait  root  \
    /usr/sbin/tcpd  /usr/sbin/in.qpopper
imap2   stream  tcp  nowait  root \
    /usr/sbin/tcpd  /usr/sbin/imapd
smtp    stream  tcp  nowait  mail \
    /usr/sbin/exim  exim -bs
```

Ejemplo de configuración de `tcpd`

```
/etc/hosts.deny:
      ALL: ALL

/etc/hosts.allow:
     ALL: LOCAL @some_netgroup
     ALL: .foobar.edu EXCEPT terminalserver.foobar.edu
     pop-3: ALL
```

Ejemplo de instalación: ratón

- `apt-get install gpm`
- Configuración según se instala: responder preguntas
- Configuracón a posteriori
  - Por programa de ayuda: `gpmconfig`
  - A mano, editando `/etc/gpm.conf`

Observar y controlar procesos

- `ps augxww`
- Ver número de proceso de `gpm`
- Ver que ratón funciona
- Matarlo con `kill` y ese número.
- Ver que no está.
- Ver que el ratón no funciona.
- Lanzarlo con `/etc/init.d/gpm start`
- Pararlo con `/etc/init.d/gpm stop`
- Lanzarlo de nuevo.
- Observar `/etc/init.d/gpm` como programa administrativo.
- Observar `/etc/gpm.conf` como fichero de configuración.