

1 Arranque. SysVinit

1.1 Sumario

- 1 NOTA
- 2 Inicio y parada Sysvinit
 - ◆ 2.1 Niveles de init
 - ◆ 2.2 Archivos de configuración de init
 - ◆ 2.3 Uso del sistema de arranque init
- 3 Linux Standard Base (LSB)

1.2 NOTA

El sistema de gestión de servicios que se explica a continuación está "deprecated" (marcado como obsoleto) en la mayoría de distribuciones GNU/Linux, las cuales han optado por sustituir su gestión mediante el sistema **systemd**. Por cuestiones de compatibilidad algunas distribuciones todavía lo soportan, aunque la tendencia es su eliminación.

1.3 Inicio y parada Sysvinit

Cuando un Sistema Linux arranca crea una serie de procesos automáticamente durante esa fase. Estos procesos serán el mínimo de servicios necesarios para que el sistema pueda funcionar adecuadamente. Es posible gestionar, es decir, añadir, eliminar y cambiar el orden en el que esos procesos se inician. Existe un proceso fundamental en todo Sistema Linux, el proceso `init` (inicio), cuya configuración nos permite establecer qué procesos serán los que se iniciarán durante el arranque. Del mismo modo se usa este mecanismo para especificar el orden de detención de éstos cuando el sistema se apague.

1.3.1 Niveles de init

Linux reconoce varios niveles, denominados **runlevels**, que definen el perfil de arranque del sistema en ese nivel, es decir, qué procesos (servicios o demonios) se iniciarán cuando el sistema arranque. Por ejemplo, en Debian se reconocen los siguientes:

```
Nivel de ejecución 0: Apagado.
Nivel de ejecución 1: Monousuario (sólo usuario root; no es necesaria la contraseña). Se suele usar para analizar y reparar prob
Nivel de ejecución 2: Multiusuario sin soporte de red.
Nivel de ejecución 3: Multiusuario con soporte de red.
Nivel de ejecución 4: Como el runlevel 3, pero no se suele usar
Nivel de ejecución 5: Multiusuario en modo gráfico (X Windows).
Nivel de ejecución 6: Reinicio.
```

En el caso de arrancar en entorno gráfico debemos utilizar, como puede verse en el listado anterior, el Nivel de ejecución (runlevel) 5, pero no siempre es así, por ejemplo en Debian/Ubuntu el nivel de entorno gráfico es el 2. Por supuesto que cuando se arranca en nivel 1, 2 o 3, el número de procesos que se levantarán será diferente, al no ser usado en esos niveles el servidor gráfico X Window. Éste es el motivo por el cual disponemos de distintos runlevels.

Comando para visualizar el nivel de ejecución de la sesión actual:

```
runlevel
```

Puede especificarse un runlevel por defecto, es decir, aquel en el que el sistema arrancará al no especificar ningún nivel concreto. En los Linux desktop lo lógico es arrancar en runlevel correspondiente al entorno gráfico. Puede especificarse ésto en el archivo **/etc/inittab** (sino existe puede crearse) y añadir una línea del tipo en ese archivo:

```
id:5:initdefault:
```

En el caso de Ubuntu 12.04 no existe ese archivo, utiliza la configuración definida en el archivo **/etc/init/rc-sysinit.conf**. En cualquier caso el mecanismo es el mismo, buscar una línea en la que se indique la opción `DEFAULT`, en este caso la línea es:

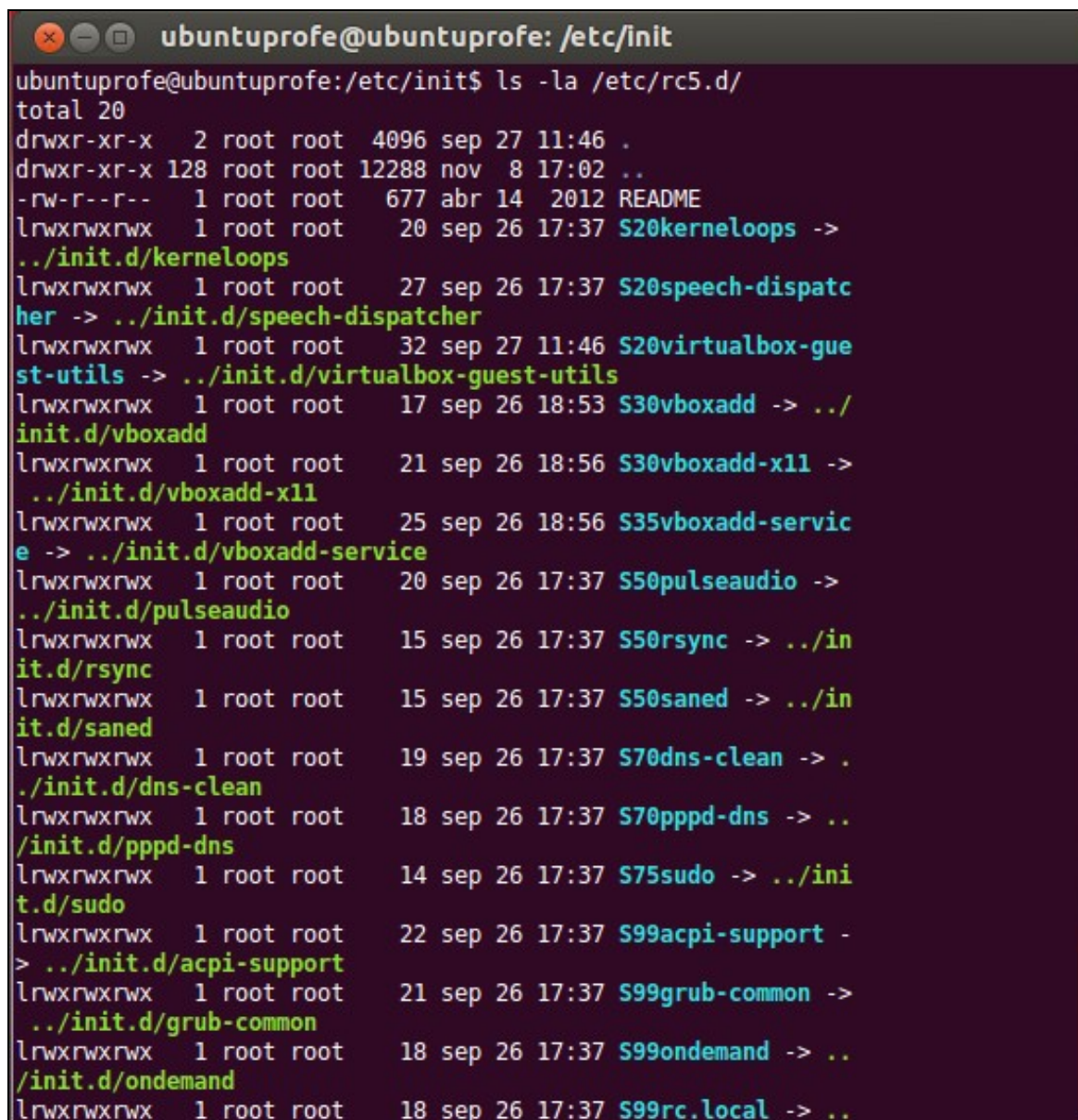
```
env DEFAULT_RUNLEVEL = 2
```

1.3.2 Archivos de configuración de init

Existe un directorio, `/etc/init.d`, dónde se incluyen distintos scripts de arranque y parada de procesos del sistema. Para verlos podéis hacer un `ls -la /etc/init.d` y observar el contenido de esa salida. Veréis que los archivos allí incluidos tienen permiso de ejecución. El objetivo de esto es tener de un modo centralizado los scripts de control de todos aquellos procesos, para poder invocar su arranque, parada, reinicio, o recarga directamente desde `/etc/init.d` (`start`, `stop`, `restart`, `reload`). De este modo solo tenemos que recordar esa ubicación para poder saber cómo invocar a un script indirectamente.

Paralelamente disponemos de los directorios `/etc/rcN.d` dónde `N=0,1,2,3,4,5,6` indica el runlevel correspondiente. En estos directorios disponemos de enlaces simbólicos a los scripts dentro de `/etc/init.d` que se iniciarán en el arranque del runlevel correspondiente. Es decir en los directorios `/etc/rcN.d` tenemos enlaces simbólicos que apuntan por lo general a scripts de gestión de servicios en `/etc/init.d`.

Hacemos un `ls -la /etc/rc5.d`



```
ubuntuprofe@ubuntuprofe: /etc/init
ubuntuprofe@ubuntuprofe:/etc/init$ ls -la /etc/rc5.d/
total 20
drwxr-xr-x  2 root root  4096 sep 27 11:46 .
drwxr-xr-x 128 root root 12288 nov  8 17:02 ..
-rw-r--r--  1 root root   677 abr 14  2012 README
lrwxrwxrwx  1 root root    20 sep 26 17:37 S20kerneloops ->
../init.d/kerneloops
lrwxrwxrwx  1 root root    27 sep 26 17:37 S20speech-dispatcher ->
../init.d/speech-dispatcher
lrwxrwxrwx  1 root root    32 sep 27 11:46 S20virtualbox-guest-utils ->
../init.d/virtualbox-guest-utils
lrwxrwxrwx  1 root root    17 sep 26 18:53 S30vboxadd ->
../init.d/vboxadd
lrwxrwxrwx  1 root root    21 sep 26 18:56 S30vboxadd-x11 ->
../init.d/vboxadd-x11
lrwxrwxrwx  1 root root    25 sep 26 18:56 S35vboxadd-service ->
../init.d/vboxadd-service
lrwxrwxrwx  1 root root    20 sep 26 17:37 S50pulseaudio ->
../init.d/pulseaudio
lrwxrwxrwx  1 root root    15 sep 26 17:37 S50rsync ->
../init.d/rsync
lrwxrwxrwx  1 root root    15 sep 26 17:37 S50saned ->
../init.d/saned
lrwxrwxrwx  1 root root    19 sep 26 17:37 S70dns-clean ->
../init.d/dns-clean
lrwxrwxrwx  1 root root    18 sep 26 17:37 S70pppd-dns ->
../init.d/pppd-dns
lrwxrwxrwx  1 root root    14 sep 26 17:37 S75sudo ->
../init.d/sudo
lrwxrwxrwx  1 root root    22 sep 26 17:37 S99acpi-support ->
../init.d/acpi-support
lrwxrwxrwx  1 root root    21 sep 26 17:37 S99grub-common ->
../init.d/grub-common
lrwxrwxrwx  1 root root    18 sep 26 17:37 S99ondemand ->
../init.d/ondemand
lrwxrwxrwx  1 root root    18 sep 26 17:37 S99rc.local ->
..
```

Podéis ver en esa salida cómo todos los archivos de ese listado **son de tipo link y apuntan a archivos dentro de `/etc/init.d`**. Los links dentro de los directorios `/etc/rcN.d` tienen un código de denominación concreto. Pueden empezar por S o K, los que empiezan por **S hacen referencia a procesos que arrancarán** en el runlevel correspondiente. A continuación en el nombre va un número decimal de 2 dígitos que indica el orden, en una secuencia, en la que arranca el proceso. Por ejemplo un link con nombre `S99xxx`, arrancará después que un link con nombre `S88xxx`, pues el número 99 es mayor que 88 y la secuencia de arranque es creciente. Los enlaces cuyo nombre empieza por la letra K corresponde a aquellos servicios que serán detenidos (Kill) en el runlevel correspondiente.

En el ejemplo anterior vemos como la sección header especifica el nombre del servicio ofrecido por el script, **Provides**, que será usado para hacer referencia a él por otros scripts. Las otras secciones especifican dependencias externas con otros scripts. Por ejemplo el **Required-Start** indica que para que el script inicie deberán ser previamente iniciados los scripts con sección **Provides postgresql y networking**. También indica en que runlevels arranca el servicio, **Default-Start**, y en que runlevels debe ser detenido **Default-Stop**. Por último ofrece la posibilidad de ofrecer una descripción corta y larga del servicio, **Short-Description y Description** respectivamente.

La información de la sección **header** será utilizada por comandos como **update-rc** e **inserv** para definir el modo de construir los enlaces en los directorios **/etc/rcx.d** según se vió en la sección anterior

[Volver](#)

JavierFP 16:31 08 ene 2019 (CET)