Arranque. SysVinit

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)

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.

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.

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 espeficiarse 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:

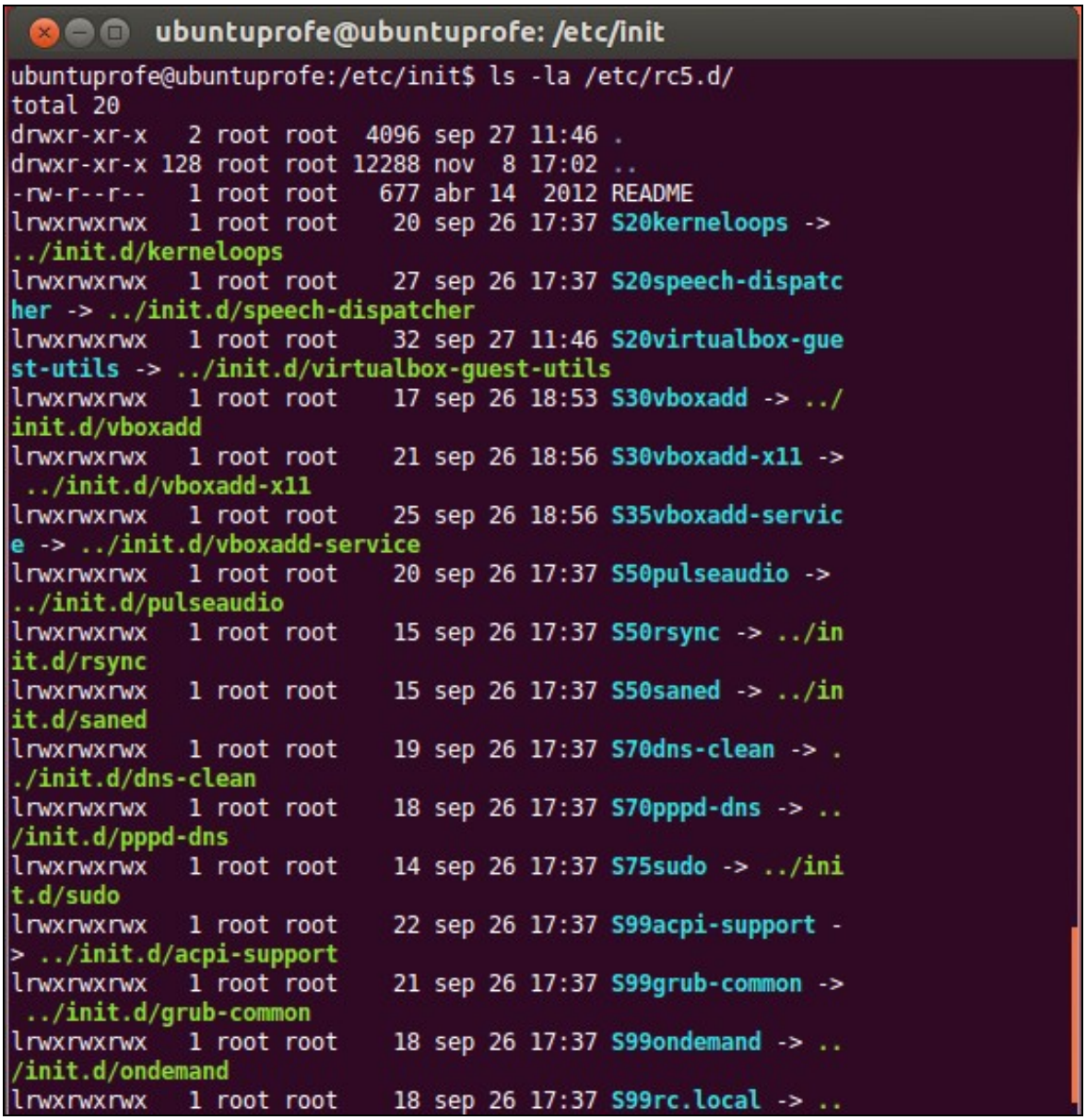
```
env DEFAULT_RUNLEVEL = 2
```

Archivos de configuración de init

Existe un diretorio, /**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 incluídos tienen permiso de ejecución. El objetivo de ésto 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



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.

El proceso de arranque es el siguiente: Dado un runlevel concreto (N=0,1,2,3,4,5,6) se ejecutarán para todos los niveles anteriores, por ejemplo para N=2, 0 y 1, los scripts dentro de /etc/rcN.d. Es decir, para runlevel 2 se ejecutarán los scripts a los que apunta /etc/rc0.d, /etc/rc1.d.

Dentro de estos directorios también existen links con nombre que empieza por K, seguidos de un número de secuencia de 2 dígitos. Éstos hacen referencia a aquellos procesos que se detendrán al iniciarse el runlevel correspondiente. En este caso el orden de ejecución es el inverso al de arranque, es decir, un link KXX con número de secuencia mayor se ejecutará antes que uno con número de secuencia menor.

Desde la versión del 2009 Ubuntu ha empezado a migrar el mecanismo de arranque los servicios del sistema a una herramienta denominada **upstart**. En la actualidad utiliza un sistema híbrido, algunos servicios pueden ser procesados mediante upstart y también está soportada la posibilidad de utilizar el sistema de init.

Uso del sistema de arranque init

Podemos controlar y configurar el mecanismo de arranque de los servicios o demonios (daemons) directamente editando los archivos, links, de /etc/rcN.d, utilizando la convención denominativa SXX o KXX, o bien utilizando el comando **update-rc.d**.

Ejemplo: Crear un link para que arranque el proceso con script en /etc/init.d/miprograma en el runlevel 2.

```
sudo ln -s /etc/init.d/miprograma /etc/rc2.d/S99miprograma
```

Al indicar un número de secuencia alto garantizamos que arrancará al final del proceso de arranque en el runlevel 2

Utilizando el comando **update-rc.d** podemos hacer lo mismo pero sin necesidad de crear explícitamente el enlace simbólico. Además con update-rc.d podemos indicar la configuración para varios runlevels simultáneamente y configuración de arranque (start) y parada (stop).

Ejemplos: Incorporar en arranque el servicio con script en /etc/init.d/miprograma para los niveles 0, 1 y 2 con secuencia 50:

```
update-rc.d miprograma start 50 0 1 2 .
```

Incorporar en arranque el servicio con script en /etc/init.d/miprograma para los niveles 2 a 5 con secuencia 40 y detención en los niveles 0 1 y 6:

```
update-rc.d miprograma start 40 2 3 4 5 . stop 40 0 1 6 .
```

Incorporar en arranque por defecto el servicio con script /etc/init.d/miprograma:

```
update-rc.d miprograma defaults
```

Linux Standard Base (LSB)

LSB es un estándar desarrollado por la Linux Software Foundation para reducir las diferencias en el modo de iniciar servicios en distintas distribuciones de Linux. Básicamente constituye un conjunto de reglas que deberían de cumplir todos los scripts relacionados con el arranque de servicios del Sistema.

Las 3 reglas principales son

- **Todo script LSB debe proporcionar las opciones de invocación start,stop,restart,force-reload,status**
- **Devolver un exit code (código de salida) en cualquier caso**
- **Documentar las runtime dependencies**

Cada script de inicio tendrá una sección **header** que puede ser utilizada para gestionar dependencias con otros scripts, por ejemplo:

```
*** BEGIN INIT INFO
* Provides:          my_daemon
* Required-Start:    postgresql networking
* Required-Stop:     postgresql networking
```

```
* Default-Start:     2 3 4 5
* Default-Stop:      0 1 6
* Short-Description: This is a test daemon
* Description:       This is a test daemon
*                    This provides example about how to
*                    write a Init script.
*** END INIT INFO
```

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 posgresql 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)