

Udev

Udev (<https://www.freedesktop.org/software/systemd/man/udev.html>) es un subsistema (actualmente incluido dentro de Systemd en forma de demonio *systemd-udevd* -permanentemente encendido-) que sirve para recibir del kernel y en tiempo real los eventos relacionados con la conexión/desconexión de los dispositivos hardware. Una vez recibidos, Udev es quien se encarga realmente de cargar los módulos involucrados y de generar al vuelo sus archivos `/dev` correspondientes (a partir de la información obtenida de `/sys`, concretamente en `/sys/block` si se trata de dispositivos de bloques y `/sys/class` si son de carácter). De aquí se deduce que la carpeta `/dev`, si es gestionada por Udev, en realidad es una partición montada en la RAM (concretamente con el sistema de ficheros `devtmpfs`).

El uso de Udev tiene varias ventajas, como por ejemplo definir reglas que nos permitirán personalizar el comportamiento de nuestro sistema en el momento de detectar un dispositivo nuevo más o menos específico. Por ejemplo, podríamos cambiar el nombre de una partición de un pendrive USB para que fuera siempre `/dev/miusb` y así no tener que usar el nombre tan extraño asignado por el kernel como por ejemplo sería `/dev/sdc3` (el cual, además, podría cambiar entre conexión y desconexión según el orden en el que el kernel detecte dicho dispositivo cada vez.).

Las reglas básicamente constan de dos partes: se ha de especificar de qué dispositivo estamos hablando mediante filtros (los cuales pueden ser más generales o más exhaustivos según si queremos que la regla afecte de forma más general -"todos los discos duros"- o tan solo a dispositivos específicos -"solo el disco duro *tal*"-) y qué acción se le quiere aplicar (por ejemplo, una acción sería efectuar el cambio de nombre respecto al nombre inicial que le asigne el kernel, pero habría muchas más posibilidades). Las reglas se pueden encontrar en dos sitios en forma de ficheros de texto con extensión `.rules`:

`*/usr/lib/udev/rules.d` : reglas por defecto instaladas con los programas
`*/etc/udev/rules.d` : reglas que se editan/añaden localmente a "mano"

Tomará preferencia aquella regla que esté contenida en un fichero que tenga (de entre los que se llamen igual) un número mayor al principio de su nombre -ya que se leerá más tarde-. Si no tiene número inicial, se harán al final.

Reglas

-Las líneas que empiezan por `#` son comentarios. Cada regla va en una línea diferente

-Algunos de los valores que sirven para identificar de qué dispositivo estamos hablando son:

| | |
|---------------------------------|---|
| ACTION | =={"add" "remove" "change"} (tipus d'event que "dispararà" la regla) |
| KERNEL | (nom amb el que veu el kernel al dispositiu. Es poden usar comodins) |
| BUS | (sistema de drivers que ese dispositiu usa: usb, scsi...) |
| SUBSYSTEM | (subsistema en el que integra el dispositiu: usb, ata, net, tty...) |
| DRIVER | (nombre del driver que gobierna el dispositiu, como p.ej usb-storage) |
| ATTRS{atributdebusqueda} | (per exemple, "model", "size"...) |

Se pueden usar los símbolos `"=="` y `"!="`

-Algunos de los valores que sirven para indicar qué deseamos hacer con ese dispositivo concreto son:

| | |
|----------------|--|
| NAME | (da directamente un nombre nuevo al dispositivo dentro de <code>/dev</code>) |
| SYMLINK | (crea un link nuevo apuntando al nombre dado por el kernel en <code>/dev</code> a ese dispositivo) |
| OWNER | (establece el propietario del dispositivo) |
| GROUP | (establece el grupo propietario del dispositivo) |
| MODE | (establece los permisos del dispositivo -en formato numérico: "0666"-) |
| RUN | (indica la ruta del binario-y sus parámetros- que se ejecutará en detectarse el dispositivo) |
| GOTO | (va a la etiqueta indicada, dentro del propio fichero de reglas) |
| LABEL | (marca la etiqueta) |

En este caso, para asignar el valor deseado se utilizará en general el operador “=” . También existen los operadores “+=” (adición de un valor a los ya existentes) y “:=” (asignación de un valor y prohibición de que se cambie en otras reglas)

Ejemplos

Supondremos que creamos un fichero con un nombre cualquiera (por ejemplo, "lalala.rules") dentro de "/etc/udev/rules.d". Podríamos hacer, por ejemplo:

```
KERNEL=="ttyACM0", SYMLINK+="modem"  
KERNEL=="sda", SYMLINK+="midisco"
```

Podríamos hacer lo mismo para dar nombres personalizados a nuestras llaves Usb, por ejemplo...pero ¿cómo lo hacemos si no sabemos de antemano qué nombre le dará el kernel a nuestra llave Usb? Recordemos que cuando se enchufa un lápiz Usb, no se tiene la seguridad de que cada vez sea el mismo fichero de dispositivo el que lo represente: unas veces puede ser por ejemplo /dev/sdb1, otras /dev/sdc1,...depende de los dispositivos Usb que ya tengamos en ese momento enchufados. Para saber cuál es el fichero de dispositivo que toca en cada momento, se podría hacer un dmesg en el momento de conectarlo y escribir la regla correspondiente que le diera el nombre siempre fijo que deseamos cada vez que ese dispositivo y ningún otro se enchufara (por ejemplo, el nombre /dev/pepito), pero ¿cómo poner en el valor KERNEL de esa regla de qué dispositivo estamos hablando si cada vez puede ser detectado de forma diferente? Para ello, debemos conocer el comando udevadm:

udevadm info -q all -n /dev/disp: muestra la informació d'Udev demanada sobre el dispositiu especificat (com l'UUID,etc). La “P” indica el path, es decir, la ruta en /sys a ese dispositivo, la “N” es el nombre del nodo en /dev, la “S” indica si hay algún enlace creado por udev a ese dispositivo y “E” muestra datos adicionales si los hay. Altres opcions pel paràmetre -q (que serveix per dir quin tipus d'informació es vol) a part de “all” justament són: “name”, “symlink”, “path”, “property”.

udevadm info -a -p elpathdeldispositiu : mostra els atributs del dispositiu especificat via el seu path (valor “P” obtingut amb la comanda anterior). També es podria fer *udevadm info -a -n /dev/disp*

Lo que tenemos que hacer, pues es ver los atributos del dispositivo usb en cuestión, enchufándolo y ejecutando por ejemplo: *udevadm info -a -p \$(udevadm info -q path -n /dev/sdb)*. Saldrá una lista de ATTRS. Elegimos uno que tenga un valor no repetido (para identificar unívocamente el dispositivo, por ejemplo ATTRS{serial} -ó bien VendorID ó también ProductID...-) y procederemos a crear la regla udev correspondiente. Por ejemplo, si tenemos dos dispositivos Usb, podremos hacer:

```
KERNEL=="sd?", ATTRS{size}=="4127760", SYMLINK+="pen2GB"  
KERNEL=="sd?", ATTRS{size}=="390719855", SYMLINK+="disco200GB"
```

Pero lo que nos interesará más siempre es acceder a las particiones (“sda1”, “sdb2”...) y no tanto a los discos duros en sí (“sda”, “sdb”), ¡que es lo que hemos hecho en las líneas anteriores! Para dar nombre a las distintas particiones podemos usar varios métodos pero el más habitual es

```
KERNEL=="sd*", ATTRS{product}="loquesea", SYMLINK+="pepito%n"
```

donde %n cogerá los valores 1,2,3...según las particiones detectadas en el dispositivo. También existe %k para indicar el nombre del dispositivo que le daría el kernel por defecto (ver man udev para más información). Tras reiniciar el servicio Udev (o bien desenchufando y volviendo a enchufar el dispositivo), la regla se activará y al conectar de nuevo la memoria usb se creará automáticamente un fichero de dispositivo con el nombre deseado, incluyendo los fichero de dispositivos de las particiones que éste incluya (/dev/pepito, /dev/pepito1, etc), los cuales no serán más que enlaces blandos a los ficheros de dispositivo “reales” /dev/sdbX (que cada vez podrán ser diferentes).

NOTA: La diferencia entre ATTR y ATTRS es que el primero se refiere a los atributos del dispositivo (p.ej, una memoria) y el segundo se refiere a los atributos del dispositivo y además de todos sus dispositivos padres (p.ej, la memoria y su controladora). Esto se puede ver con *udevadm info*...además, se puede observar que el path de un dispositivo hijo incluye el