

Seminario. Gestor de dispositivos udev de Linux

Duración: 1 sesión

1. Objetivos del seminario

Los **objetivos** concretos de este seminario son:

- Conocer cómo funciona el gestor de dispositivos udev de Linux.
- Configurar una regla sencilla para detectar la conexión o desconexión de un dispositivo USB en el sistema.

2. Introducción

udev es un gestor de dispositivos genérico que se ejecuta como un demonio en un sistema Linux y escucha (a través de un socket netlink) los eventos que el kernel envía si se inicializa un nuevo dispositivo o se elimina un dispositivo del sistema. Este gestor de dispositivos se ejecuta en el espacio de usuario.

Las reglas de **udev** se leen de los archivos ubicados en el directorio de reglas del sistema `/lib/udev/rules.d`

cuando se agrega o quita un dispositivo (o cambia su estado de alguna forma) el kernel lo informa al servicio **systemd-udevd**. Este último puede configurarse para responder a dichos eventos mediante reglas en archivos con la extensión **.rules**

Por otro lado, la herramienta **udevadm** controla el funcionamiento de **udev**. Entre otros usos, destaca la monitorización de los eventos detectados por el kernel.

Hay otras herramientas adicionales que nos permiten examinar la información sobre los dispositivos conectados al sistema. Por ejemplo, para ver la información sobre discos USB usaremos:

```
lsblk
```

Ahora, para consultar los atributos de uno de esos dispositivos, p.ej. el sdb, usaremos:

```
udevadm info /dev/sdb
```

Una vez que tengamos la información específica sobre algún dispositivo, podemos crear una regla en el sistema para ejecutar una acción ante un evento relacionado con ese dispositivo.

3. Crear una regla de udev

Queremos crear una configuración en nuestra máquina de forma que al retirar un dispositivo USB del ordenador se lance una "acción udev".

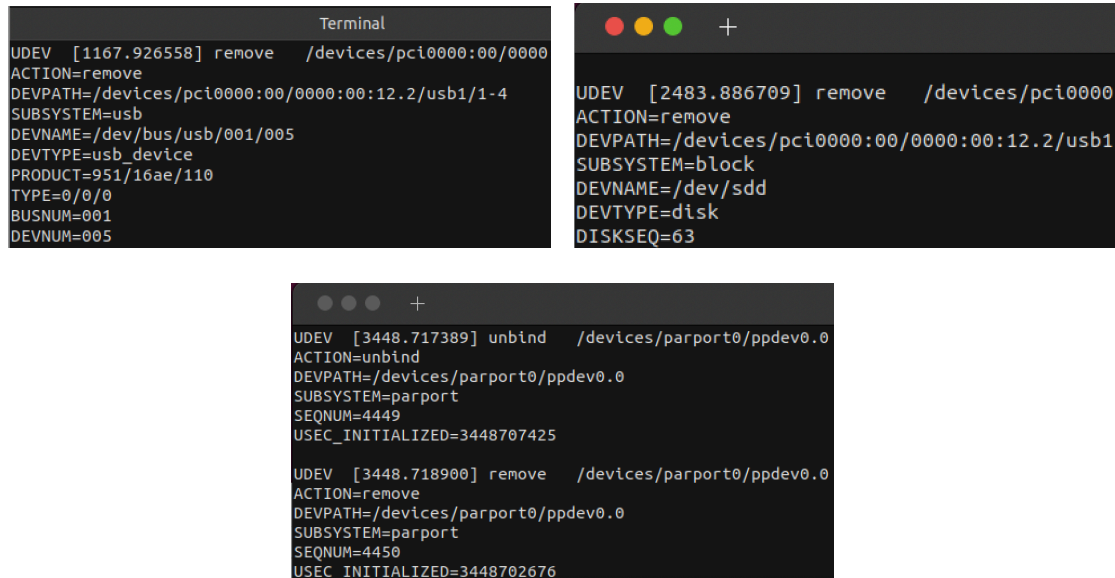
Para configurarlo vamos a añadir una nueva regla al archivo de reglas udev, localizado en `/etc/udev/rules.d/`

La idea es crear una regla sencilla que ejecute un proceso (p.ej. lanzar el salvapantallas para bloquear la pantalla) cada vez que se retire cualquier dispositivo

USB de uno de los puertos. Debemos tener siempre en cuenta que el proceso que se ejecute lo hará con privilegios de root.

Pero antes de comenzar con la configuración, **veamos qué ocurre en el kernel** cuando conectamos o desconectamos un dispositivo. Para ello ejecutamos el siguiente comando y vemos qué mensajes muestra el kernel ante cada tipo de evento:

```
udevadm monitor --environment --udev
```



The image shows three terminal windows displaying the output of the command `udevadm monitor --environment --udev`. The first window shows a 'remove' event for a USB device with details like `DEVNAME=/dev/bus/usb/001/005` and `DEVTYPE=usb_device`. The second window shows another 'remove' event for a USB device, with details like `DEVNAME=/dev/sdd` and `DEVTYPE=disk`. The third window shows an 'unbind' event for a parport device, followed by a 'remove' event for the same device.

Podemos ver diversa información sobre los dispositivos conectados/desconectados en ese momento. Además, en cada evento vemos que se aporta información sobre la acción, sobre el subsistema afectado, el tipo de dispositivo o el nombre del dispositivo:

- ACTION=unbind
- ACTION=remove
- ACTION=add
- SUBSYSTEM=usb
- SUBSYSTEM=block
- DEVTYPE=disk
- DEVTYPE=usb_device
- DEVNAME=/dev/sdd

Así pues, **usando esta información podemos crear una regla** que añadiremos a las que hay en el sistema de reglas para que se ejecute cuando se produzca uno de esos eventos.

Para el **primer ejemplo sencillo** (y comprobar así que todo funciona bien) vamos a crear un script llamado "retirar.sh" que simplemente añada un par de líneas a un archivo de texto llamado "log.txt". Ese script lo guardaremos en la carpeta `/root/`

```
#!/bin/bash
echo RETIRADO >> /root/log.txt
date >> /root/z.txt
echo " " >> /root/log.txt
```

Una vez tenemos el proceso que será lanzado cuando se desconecte un dispositivo USB, sólo tenemos que **crear un archivo con extensión .rules** en el directorio

/etc/udev/rules.d detallando el evento que queremos comprobar y especificando la acción que queremos realizar. Para ello podemos ejecutar lo siguiente:

```
cat << EOF | sudo tee /etc/udev/rules.d/busKill.rules
ACTION=="unbind", SUBSYSTEM=="usb", RUN+="/root/retirar.sh"
EOF
```

dónde:

- "==" es un operador para comparar igualdad
- "+=" es un operador para agregar el valor a una clave
- ACTION coincide con el nombre de la acción del evento
- SUBSYSTEM coincide con el subsistema del dispositivo de evento
- RUN especifica el script a ejecutar

En ese ejemplo hemos pedido que ante un evento de tipo **"unbind"** relacionado con el subsistema USB se ejecute el script que hemos creado antes.

Ahora sólo tenemos que **recargar los archivos de reglas** de **systemd-udev**:

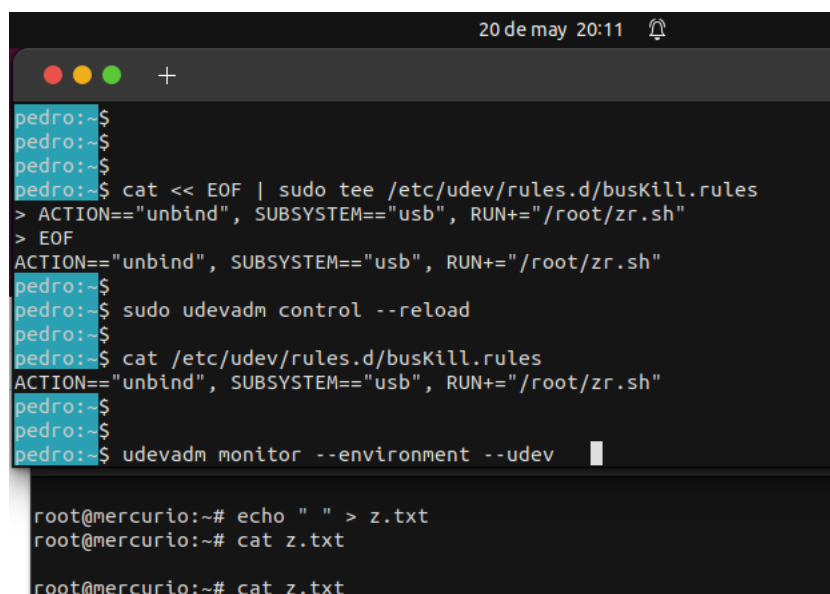
```
sudo udevadm control --reload
```

Si comprobamos el contenido del archivo .rules que hemos creado veremos nuestra regla:

```
cat /etc/udev/rules.d/busKill.rules
```

Para visualizar en tiempo real todos los eventos que suceden, debemos ejecutar el siguiente comando:

```
udevadm monitor --environment --udev
```



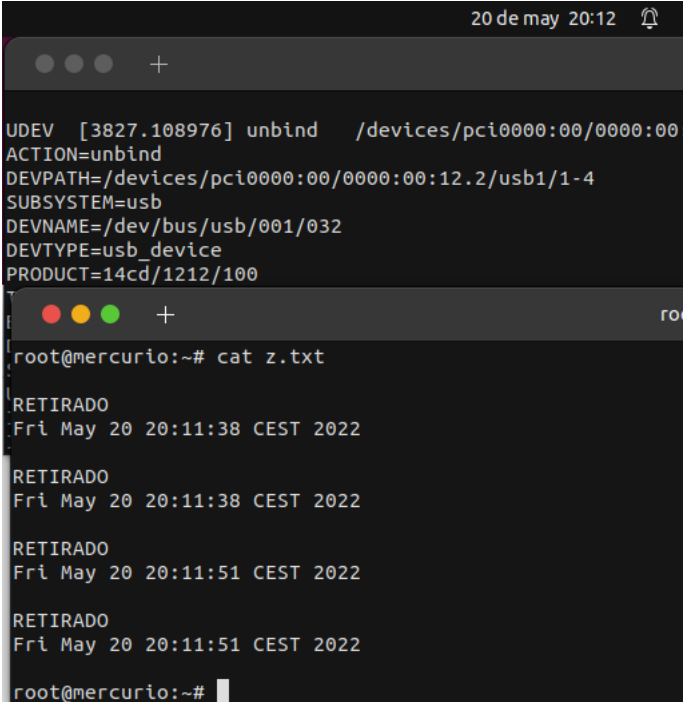
```
20 de may 20:11
pedro:~$
pedro:~$
pedro:~$
pedro:~$ cat << EOF | sudo tee /etc/udev/rules.d/busKill.rules
> ACTION=="unbind", SUBSYSTEM=="usb", RUN+="/root/zr.sh"
> EOF
ACTION=="unbind", SUBSYSTEM=="usb", RUN+="/root/zr.sh"
pedro:~$
pedro:~$ sudo udevadm control --reload
pedro:~$
pedro:~$ cat /etc/udev/rules.d/busKill.rules
ACTION=="unbind", SUBSYSTEM=="usb", RUN+="/root/zr.sh"
pedro:~$
pedro:~$
pedro:~$ udevadm monitor --environment --udev

root@mercurio:~# echo " " > z.txt
root@mercurio:~# cat z.txt
root@mercurio:~# cat z.txt
```

Conforme conectemos o desconectemos dispositivos USB veremos que se añade diversa información al sistema y se muestra en tiempo real en la consola.

Puesto que nuestro proceso escribía líneas de texto en el archivo "log.txt" ante cada desconexión de un dispositivo USB, podemos revisar dicho archivo para comprobar que nuestra regla funciona y se ejecuta ante cada evento de desconexión:

```
cat /root/z.txt
```



```

20 de may 20:12
[3827.108976] unbind /devices/pci0000:00/0000:00:
ACTION=unbind
DEVPATH=/devices/pci0000:00/0000:00:12.2/usb1/1-4
SUBSYSTEM=usb
DEVNAME=/dev/bus/usb/001/032
DEVTYPE=usb_device
PRODUCT=14cd/1212/100

root@mercurio:~# cat z.txt

RETIRADO
Fri May 20 20:11:38 CEST 2022

RETIRADO
Fri May 20 20:11:38 CEST 2022

RETIRADO
Fri May 20 20:11:51 CEST 2022

RETIRADO
Fri May 20 20:11:51 CEST 2022

root@mercurio:~#

```

Como vemos, la idea es sencilla y no es necesario que haya nada almacenado en la unidad USB.

4. Mejorando la regla udev

En la sección anterior hemos visto el funcionamiento más básico, sin embargo, esa regla es demasiado "amplia", ya que la acción será ejecutada sea cual sea el dispositivo USB que se desconecte...

Si quieres que una marca específica de unidad USB active el script, primero tenemos que hacer un poco de depuración para encontrar algunas propiedades que son activadas por la unidad USB específica al ser desconectada. Como hemos visto, al examinar los eventos con:

```
udevadm monitor --environment --udev
```

hemos obtenido información diversa sobre cada dispositivo conectado/desconectado. Si nos quedamos con algunas propiedades que puedan ser identificables (fabricante, modelo, número de producto, el UUID, etc) podremos hacer una regla específica para un dispositivo USB muy concreto.

Supongamos que el ID_MODEL del dispositivo que estamos usando para este seminario es "Micromax_A74". Entonces podríamos construir una regla más específica como la siguiente y sólo ese dispositivo la activaría:

```

cat << EOF | sudo tee /etc/udev/rules.d/busKill.rules
ACTION=="unbind", SUBSYSTEM=="usb",
ENV{ID_MODEL}=="Micromax_A74", RUN+="/root/retirar.sh"
EOF

sudo udevadm control --reload

```

Por otro lado, supongamos que queremos detectar, no solo la desconexión de un dispositivo, sino también cuándo se conecta al sistema. En este caso necesitaríamos dos scripts y crear las correspondientes reglas en el archivo `.rules`:

```
/root/insertar.sh  
/root/retirar.sh
```

A continuación, creamos dos reglas para activar la ejecución de esos scripts:

```
sudo nano /etc/udev/rules.d/busKill.rules  
  
ACTION=="add", SUBSYSTEM=="usb", RUN+="/root/insertar.sh"  
ACTION=="unbind", SUBSYSTEM=="usb", RUN+="/root/retirar.sh"
```

Al igual que en los ejemplos previos, recargamos las reglas y comprobamos el funcionamiento:

```
sudo udevadm control --reload  
  
udevadm monitor --environment --udev
```

Ahora, al conectar y desconectar deberíamos ver que se añade texto al `log.txt`.

Por último, podríamos programar un sistema de seguridad, haciendo que el sistema funcione mientras que cierto dispositivo USB esté conectado, pero se bloquee en el momento en que sea retirado del puerto. Lo único que tenemos que hacer es, en lugar de recoger información en el archivo de texto, bloquear la pantalla con el salvapantallas ante la desconexión del dispositivo USB. Para ello podemos usar la siguiente regla:

```
ACTION=="unbind", SUBSYSTEM=="usb", RUN+="/usr/bin/xscreensaver-command -lock"
```

De esta forma tendríamos una "llave de seguridad" en nuestro sistema, que lo bloquearía en el momento en que nos retirásemos del mismo llevándonos nuestro dispositivo de seguridad:



5. Programar acciones ante eventos en Linux, macOS y Windows

Hemos presentado cómo llevar a cabo la tarea bajo Linux. Sin embargo, en cualquier sistema operativo moderno disponemos de herramientas para llevarla a cabo.

A continuación dejo varios tutoriales en los cuales se muestra cómo programar este tipo de reglas y acciones en Linux, Windows y macOS:

<https://tech.michaelaltfield.net/2020/01/02/buskill-laptop-kill-cord-dead-man-switch/>

<https://tech.michaelaltfield.net/2020/01/02/buskill-laptop-kill-cord-dead-man-switch/#comment-37530>

<https://stackoverflow.com/questions/7240117/execute-an-application-on-mac-os-x-when-a-particular-type-of-usb-device-is-conne/12259762#12259762>

Cuestiones a resolver

El objetivo principal es conocer cómo funciona el gestor de dispositivos **udev** de Linux.

El estudiante debe estudiar el gestor de dispositivos **udev** de Linux. A continuación configurará una regla sencilla para detectar la conexión o desconexión de un dispositivo USB en el sistema usando las herramientas estudiadas.

Como resultado **se mostrará** al profesor el funcionamiento correcto de la regla udev al desconectar un dispositivo USB del sistema.

En el documento a entregar se describirá cómo se ha configurado la regla udev y se incluirán varias capturas de pantalla mostrando el proceso y el resultado.

Normas de entrega

La práctica o seminario podrá realizarse de manera individual o por grupos de hasta 2 personas.

Se entregará como un archivo de texto en el que se muestre la información requerida. También se puede utilizar la sintaxis de Markdown para conseguir una mejor presentación e incluso integrar imágenes o capturas de pantalla. La entrega se realizará subiendo los archivos necesarios al repositorio “**PDIH**” en la cuenta de GitHub del estudiante, a una carpeta llamada “**S-reglas-udev**”.

Toda la documentación y material exigidos se entregarán en la fecha indicada por el profesor. No se recogerá ni admitirá la entrega posterior de las prácticas/seminarios ni de parte de los mismos.

La detección de copias implicará el suspenso inmediato de todos los implicados en la copia (tanto de quien realizó el trabajo como de quien lo copió).

Las faltas de ortografía se penalizarán con hasta 1 punto de la nota de la práctica o seminario.

Referencias

<https://www.compuhoy.com/su-pregunta-que-es-udev-ubuntu/>

<https://blog.carreralinux.com.ar/2018/02/conceptos-sobre-udev-dispositivos/>

<https://linux.die.net/man/8/udevadm>

<https://www.linuxparty.es/75-hardware/10488-como-usar-udev-para-la-deteccion-y-gestion-de-dispositivos-en-linux.html>

<https://tech.michaelaltfield.net/2020/01/02/buskill-laptop-kill-cord-dead-man-switch/>

<https://programmerclick.com/article/6259422049/>

<https://neilzone.co.uk/2022/01/implementing-buskill-shutting-down-a-debian-11-bullseye-machine-when-a-specific-usb-device-is-removed>