udev & sysfs : fonctionnement et administration

UFR des Sciences Versailles - M2 SeCReTS

CAUMES Clément & DEBROUASSE Kevin &
HEQUET Jonathan & Mehdi MTALSI-MERIMI

Lundi 25 Novembre 2019

udev, sysfs : présentation du fonctionnement général le plus simplement possible et exploitation pour l'administration et la compréhension des messages associés dans les log's.

- Préambule et définitions des bases
- Historique avant udev
- Définitions précise de udev et sysfs
- Ecriture de règles udev
- Exploitation pour l'administration

Kernel

Le kernel d'un système d'exploitation est le logiciel permettant :

- la gestion des systèmes de fichiers
- la communication entre les logiciels et les périphériques connectés à l'appareil
- la gestion et la communication entre les processus

Pilote

Un pilote est un programme qui permet au système d'exploitation d'interagir avec un périphérique.

Périphérique

Tout élément dans un OS Linux est représenté par un fichier. De cette manière, les périphériques sont stockés sous forme de fichiers dans le dossier /dev. Ces fichiers vont donc permettre l'accès aux périphériques et sont appelés device nodes.

Device Node

Un device node est un point d'entrée vers le noyau caractérisé par un type (bloc ou char) et deux nombres: le major et le minor. Cela définit de façon unique quel périphérique matériel est accédé via ce fichier. Le majeur permet au noyau de savoir quel driver doit gérer le périphérique et le mineur permet au driver de savoir quel périphérique parmi ceux qu'il gère est utilisé.

ls -l /dev

Permet de voir les différents node devices de la machine.

Qu'est ce que makedev?

Dans les premières versions de Linux, les numéros de major et minor sont codés dans le noyau. De plus, l'espace de valeurs possibles pour les numéros major et minor était trop petit. Il fallait donc trouver une autre solution pour les versions suivantes de Linux.

Qu'est ce que devfs?

Devfs est un système de fichiers contenant les device nodes et dont les noeuds sont créés par les pilotes des périphériques lors de leur détection. Cependant, il y avait toujours certaines limites (espace de numéros trop petit notamment).

udev

udev est un gestionnaire de périphériques du dossier /dev. Il va créer des nœuds dynamiquement pour les périphériques connectés au système. Udev détecte lorsqu'un nouveau périphérique est connecté.

udev

udev est charger de faire le lien entre les informations de sysfs et les règles de l'utilisateur. Pour manipuler et connaitre le traitement du périphérique défini par l'utilisateur, udev se base sur plusieurs propriétés.

Interprétation des propriétés importantes

Les propriétés dépendent du type exact de périphérique mais certaines propriétés sont toujours présentes:

- ACTION : Le type d'événement à traiter
- MAJOR, MINOR : Les numéro majeur et mineur du périphérique concerné.
- SEQNUM : un numéro croissant pour ordonner les événements,
- SUBSYSTEM : Le sous-système noyau ayant causé l'événement,
- DEVPATH : Le fichier dans /sys correspondant au périphérique.

udevadm monitor -k -p

```
$ udevadm monitor -k -p
monitor will print the received events for: KERNEL - the kernel uevent
KERNEL[8414.073713] add /devices/pci0000:00/0000:00:0b.0/usb1/1-1 (usb)
ACTION=add
DEVPATH=/devices/pci0000:00/0000:00:0b.0/usb1/1-1
SUBSYSTEM=usb
DEVNAME=/dev/bus/usb/001/027
DEVTYPE=usb_device
PRODUCT=1e3d/2093/100
TYPE=0/0/0
BUSNUM=001
DEVNUM=027
SEQNUM=2455
MAJOR=189
MINOR=26
```

udevadm info -a -p

```
$ udevadm info -a -p /dev/sdb
...
looking at device
'/devices/pci0000:00/0000:00:06.0/0000:07:00.1/host4/rport-4:0-5/target4:0:0/4:0:0:5/block/sdd':
KERNEL==sdb
SUBSYSTEM==block
DRIVER==
ATTR{range}==16
ATTR{ ext_range} ==256
ATTR{removable} ==0
ATTR{ro} ==0
ATTR{size} ==35163230208
ATTR{alignment_offset} ==0
ATTR{discard_alignment} ==0
ATTR{capability} ==52
...
```

Fonctions

- Sysfs est un système de fichier virtuel, c'est à dire couche d'abstraction au dessus du système de fichier physique
- Il introduit en même temps que udev avec le noyau 2.6 de linux
- Sysfs exporte depuis l'espace noyau vers l'espace utilisateur les informations sur les périphériques du système. Ainsi, il va créer un dossier associé au système de fichiers contenant une suite de fichiers représentant les attributs du périphérique en question.

ls /sys/block/sdb

Permet de visualiser les fichiers représentant les attributs du premier périphérique USB connecté.

Les règles permettent d'automatiser un certain nombre de taches en fonction des évènements recensées par le noyau.

Que peut-on faire avec ?

- changer de nom un périphérique (durable ou lien symbolique)
- changer les permissions et les propriétés d'un périphérique
- lancer des scripts

Comment les écrire ?

- les règles de base se trouvent dans /lib/udev/rules.d/*
- les règles commencent toujours par "[0-9]*-titre.rules"
- écrire dans ce répertoire plutôt /etc/udev/rules.d/*

Quelle est la syntaxe globale ?

- système de clef/valeur (ex: KERNEL, SUBSYSTEM, DRIVER)
- commande (ex : SYMLINK, NAME, RUN)
- utilisation des expressions régulières
- utilisation de substitution de caractères

Opérateurs

- == : utilisé pour tester l'égalité
- != : utilisé pour tester la différence
- = : utilisé pour assigner une valeur à une clé
- += : utilisé pour ajouter la valeur à une suite de valeurs déjà assigné à la clé

Substitutions de caractères

- %n : représente le numéro kernel ex : sdb1 donne 1
- %k : nom kernel ex : sdb1
- %c : permet de récupérer la sortie de PROGRAM

Clés principales https://linux.die.net/man/8/udev

- ACTION : représente l'action du périphérique (connexion avec add et déconnexion avec remove) ex : ACTION=="add"
- DEVPATH : représente le chemin absolu d'accès au périphérique
- KERNEL : représente le nom du périphérique ex : KERNEL=="sd[b-z][0-9]"
- NAME : représente le nom du noeud du périphérique
- SYMLINK : représente le lien symbolique du noeud (il peut y avoir plusieurs lien symbolique par noeud)
- SUBSYSTEM : représente ke sous-système du périphérique ex : SUBSYSTEM=="usb"
- DRIVER : représente le nom du pilote du périphérique
- ATTR{filename} : représente l'attribut filename trouvé par sysfs lors de la connexion du périphérique.
- RUN : permet d'exécuter un script ou une commande
- ...

udevadm test

Pour visualiser quels scripts ont été lancés à la connexion d'un périphérique, il est possible d'utiliser la commande suivante :
udevadm test -a add [fichier sysfs]
ex : udevadm test /sys/block/sdb

Exemples

- KERNEL=="sdb[0-9]", ACTION=="add", RUN+="/usr/bin/program.sh" : va détecter la connexion d'un disque dur externe et va lancer le script program.sh
- KERNEL=="mice", ACTION=="add", NAME="souris" : va détecter la souris à sa connexion et va créer un unique node dans dev/souris
- KERNEL=="hdc", ACTION=="add", SYMLINK+="dev/cdrom" : va détecter le CD-ROM et va créer un lien symbolique dev/cdrom et qui pointera vers dev/hdc

Définition

Un log (ou logging) est un fichier permettant de stocker un historique des événements sur une machine. C'est donc un journal de bord qui est utilisé dans l'administration système pour garder une trace de ce qui s'est passé (pas forcément des incidents).

Informations utiles pour les logs

- Date et heure de l'action
- Identification de l'action
- Auteur de l'action (dans l'idéal)
- Identification de l'outil permettant d'effectuer l'action

Utilité de udev et de sysfs pour l'administration

udev permet de lancer des scripts lors de la connexion et déconnexion de périphériques sur la machine.
On peut donc écrire dans un fichier (de log) afin d'identifier l'action (connexion ou déconnexion) et le périphérique.

Extrait du résultat généré par une règle udev pour l'administration

Fri Nov 15 11:23:02 CET 2019 - CAUMES Generic_Flash_Disk_CCCB110423110435095297341 4-0:0 sdb1 : connexion
Fri Nov 15 11:23:04 CET 2019 - HEQUET Verbatim_STORE_N_GO_070126D061196C46-0:0 sdc1 : connexion
Fri Nov 15 11:23:09 CET 2019 - CAUMES Generic_Flash_Disk_CCCB110423110435095297341 4-0:0 sdb1 : deconnexion
Fri Nov 15 11:23:11 CET 2019 - HEQUET Verbatim_STORE_N_GO_070126D061196C46-0:0 sdc1 : deconnexion

Idées de règles udev pour l'administration

- Exercice de création de logs qui va détecter les différentes connexions et déconnexions de clés USB, ainsi que le montage/démontage de nouvelles partitions sur le disque dur (exercice 2).
- Exercice qui va créer un nouveau point de montage d'une clé USB et la démonter à sa déconnexion (exercice 3).
- Exercice qui va réaliser un backup sur une clé particulière (en fonction du numéro de série) (exercice 4).
- Exemple d'exercice d'administration qui va faire une analyse anti-virus automatique du contenu de la clé USB qui vient d'être connectée.

Références

- doc.ubuntu-fr.org/udev
- linuxconfig.org/tutorial-on-how-to-write-basic-udev-rules-in-linux
- www.linuxembedded.fr/2015/05/une-introduction-a-udev/