udev

Gestion des périphériques dynamiques

Description

udev fournis un système d'évènements périphériques, de gestion des permissions des nœuds périphériques, et peut créer des liens symboliques additionnels dans /dev, ou renommer des interfaces réseaux. Le kernel assigne simplement des noms de périphériques basés sur l'ordre de découverte.

udevd reçois les uevents directement du kernel quand un périphérique est ajouté ou enlevé du système, ou quand son état change. Quand udev reçois un évènement périphérique, il regarde si des règles dans sa configuration correspondent, peut fournir des informations additionnelles à stocker dans la base de données udev, ou des informations pour créer des noms de liens. Tous les processus d'information de périphériques sont stockés dans la base de données udev et envoyés au souscripteur d'évent.

Configuration

Les fichiers de configuration sont dans **/etc/udev** et **/lib/udev**. Toutes les lignes vides, ou commençant par '#' seront ignorés

Fichier de configuration

Le fichier de configuration principal est **/etc/udev/udev.conf**. Il consiste de variables permettant à l'utilisateur d'écraser les valeurs

udev par défaut. Les variables sont :

udev_root Spécifie où placer les nœuds de périphériques dans le système de fichier (défaut /dev)

udev_log Le logging priority.

Fichiers de règles

Les règles par défaut de **udev** sont lues depuis les fichiers dans **/lib/udev/rule.d/**, les fichiers personnalisés dans **/etc/udev/rules.d/**, et les fichiers temporaires dans **/dev/.udev/rules.d/**. Tous les fichiers de règles sont triés et traités dans l'ordre lexical, sans regarder dans quel répertoire ils sont. Les fichiers dans **/etc/udev/rules.d** ont une précédence sur les fichiers de même nom dans /lib/udev/rules.d.

Les fichiers de règles doivent se terminer par **.rules**, sinon ils sont ignorés. Chaque ligne dans un fichier de règle contient au moins une paire de clé/valeur. Il y'a 2 types de clé, les clés de correspondance et les clés d'assignement. Si toutes les clés de correspondance correspondent avec leurs valeurs, la règle s'applique et les clés d'assignement reçoivent la valeur spécifiée. Une règle de correspondance peut renommer une interface réseau, ajouter les liens pointant vers des nœuds de périphériques, ou lancer un programme spécifique.

Une règle consiste d'une ou plusieurs paires de clé/valeur séparés par un ','. Chaque clé a une opération distincte, dépendant de l'opérateur utilisé

Opérateurs

== compare l'égalité

!= Compare l'inégalité

= Assigne une valeur à une clé. Les clés qui représentent une liste, sont réinitialisées avec cette valeur simple.

+= Ajoute la valeur à une clé qui maintient une liste

:= Assigne une valeur à une clé, puis empêche de modifier cette valeur ultérieurement.

Les noms des clés peuvent être utilisés pour faire correspondre les propriétés des périphériques. Certains périphériques matchent également des clés avec les propriétés des périphériques parent dans sysfs. Si plusieurs clés qui match un périphérique parent sont spécifié dans une simple règle, toutes ces clés doivent matcher à un et le même périphérique parent.

ACTION Match le nom de l'action

DEVPATH Match le devpath du périphérique

KERNEL Match le nom du périphérique

NAME Match le nom du nœud ou de l'interface réseau

SYMLINK Match le nom d'un lien ciblant le nœud

SUBSYSTEM Match le nom du pilote du périphérique

ATTR{filename} Match les valeurs d'attribut sysfs du périphérique.

KERNELS Cherche le devpath supérieur pour un nom de périphérique correspondant

SUBSYSTEMS Cherche le devpath supérieur pour un nom de sous-système de périphérique correspondant

DRIVERS Cherche le devpath supérieur pour un nom de pilote de périphérique correspondant

ATTRS{filename} Cherche le devpath supérieur pour un périphérique qui match les valeurs d'attributs sysfs. Si plusieurs ATTRS sont spécifiés, tous doivent matcher sur le même périphérique.

ENV{key} Match avec une valeur de propriété de périphérique.

TAG Match avec un tag de périphérique

TEST{masque octal} Test l'existence d'un fichier. Un masque octal peut être spécifié.

PROGRAM Exécute un programme. La clé est vraie, si le programme retourne un succès. Les propriétés du périphérique sont disponibles pour le programme. La sortie du programme affichée dans stdout est redirigée dans la clé RESULT

RESULT Match la chaîne retournée par un appel PROGRAM.

De nombreux champs supportent les motifs type shell :

***** Match aucun, ou un certain nombre de caractères

? Match un simple caractère

[] Match un simple caractère spécifié dans les crochets.

Les clés suivantes peuvent être assignées :

NAME Le nom du nœud de périphérique. Généralement, le kernel définis le nom du nœud par défaut, ou créé et supprime le nœud avant que udev reçoive l'event. Changer le nom du nœud créés des inconsistances et n'est pas supportées. Si le kernel et NAME spécifient des noms différents, une erreur sera loggée. Udev ne modifie que les permissions des nœuds et crée des liens additionnels, qui est une bonne manière pour renommer un nœud. Les liens ne doivent jamais être en conflit avec des noms de nœud.

SIMLINK Le nom de lien ciblant un nœud. Chaque règle qui match va ajouter cette valeur à la liste des liens à créer. Plusieurs liens peuvent être créés en les séparant par un espace. Si plusieurs périphériques demandent le même nom de lien, le lien pointera toujours vers le périphérique avec la plus haute link_priority. Si ce périphérique est supprimé, le lien sera réévalué. Sans link_priority spécifié, le périphérique obtenant le lien ne peut être prédit.

OWNER, GROUP, MODE Les permissions pour le nœud périphérique.

ATTRKEY La valeur qui devrait être écrite dans un attribut sysfs du périphérique de l'évènement

ENVKEY Définis une valeur de propriété du périphérique. Les noms de propriété avec un '.' ne sont pas stockés dans la base ou exportés vers les outils externes ou évènements.

TAG Attache un tag à un périphérique. C'est utilisé pour filtrer les évènements pour les utilisateurs de la fonctionnalité de monitoring de libudev. L'implémentation peut seulement fonctionner efficacement si seulement quelques tags sont attachés aux périphériques.

RUN Ajoute un programme à la liste des programmes à exécuter pour un périphérique spécifique. Cela peut seulement être utilisé pour des tâche très courtes. Les tâches longues bloquent le processus d'évènement. Si RUNfail_event_on-error est spécifié, et que le programme ne retourne pas 0, l'évènement sera marqué comme échoué. Sans chemin absolu, le programme doit être dans /lib/udev. Les chemins avec des espaces peuvent être spécifié entre ”

LABEL Label nommé où un GOTO peut sauter

GOTO Saute à LABEL

IMPORT{type} Importe des variables comme propriétés de périphérique, en fonction de son type :

program Exécute un programme externe.

file Importe un fichier texte comme valeur assignée, qui doit être au format de clé d'environnement

db Importe une simple propriété spécifiée comme valeur assignée la base du périphérique courrant. Ne fonctionne que si la base est déjà remplie par un précédent évènement.

cmdline Importe une simple propriété depuis la ligne de commande du kernel. Pour de simples flags la valeur de la propriété sera à 1.

parent Import les clés stockées du périphérique parent.

Sans option, udev choisis entre program et file en fonction du bit exécutable des permissions de fichier

WAIT_FOR Attend qu'un fichier devienne disponible ou 10sec. Le chemin est relatif au périphérique sysfs. Sans chemin spécifié, attend qu'un attribut apparaisse.

Options

link_priority=value Spécifie la priorité des liens crées. (Défaut : 0)

event_timeout= Nombre de seconde qu'un event attend les opérations avant de se terminer de lui-même.

string_escape=none|replace Les caractères non-sûres et de contrôle sont remplacés dans les chaînes utilisé pour le nommage des périphériques.

static_node= Applique les permissions spécifiée dans cette règle à un nœud de périphérique statique avec le nom spécifié. Les nœuds statique peuvent être des modules kernel, ou copiés depuis /lib/udev/devices. Ces nœuds peuvent ne pas avoir de module kernel correspondant au moment où udev est démarré, et permet de piloter les modules chargeable du kernel.

watch Recherche un nœud avec inotify, quand il est fermé après avoir été ouvert en écriture, un uevent de changement sera synthétisé.

nowatch Désactive la recherche de nœud avec inotify

Les champs **NAME, SYMLINK, PROGRAM, OWNER, GROUP, MODE** et **RUN** supportent les substitutions de chaîne dans le style printf. Les substitutions sont :

Options

$kernel, %k Le nom kernel pour ce périphérique

$number, %n Le numéro kernel pour ce périphérique. Par exemple 'sda3' a comme nombre kernel 3.

$devpath, %p Le devpath du périphérique

$id, %b Le nom du périphérique matché pendant la recherche de devpath supérieur pour SUBSYSTEMS, KERNELS, DRIVERS et ATTRS

$driver Le nom du pilote du périphérique matché pendant la recherche du devpath supérieur pour SUBSYSTEMS, KERNELS, DRIVERS et ATTRS

$attrfile, %sfile La valeur d'un attribut sysfs trouvé au périphérique, où toutes les clés de la règle ont matchés.

$envkey, %Ekey Une valeur de propriété de périphérique

$major, %M Le numéro majeur kernel pour le périphérique

$minor, %m Le numéro mineur kernel pour le périphérique

$result, %c La chaîne retournée par le programme externe lancé avec PROGRAM. Une seule partie de la chaîne peut être sélectionnée en spécifiant le numéro comme attribut %cN. Si le numéro est suivi d'un '+', cette partie + plus celles restantes sont substituées.

$parent, %P Le nom du nœud parent

$name Le nom courant du nœud

$links La liste des liens courants, séparés par un espace.

$root, %r La valeur udev_root

$sys, %S Le point de montage sysfs

$tempnode, %N Le nom d'un nœud créé temporairement pour fournir l'accès au périphérique depuis un programme externe avant que le vrai nœud soit créé.

%% Le caractère %

$$ Le caractère $