udev

devfs done right

Managing Linux devices names in userspace

Armen Baghumian
armen@OpenSourceClub.org

How devices are accessed in Linux

- Everything is a file (one of the Unix creeds)
  - link, device node, pipe ....

- Device are accessed via device nodes

- Device nodes behave like normal file
  - reside on file system
  - file operations like open, read, write seek are possible
  - if you write to the device node the kernel writes to the device
  - if you read from the device node the kernel reads from the device

- You can backup the whole drive

```
# dd if=/dev/hdb of=~/backup/hdb.img
```

How devices are accessed in Linux

All devices are distinguished by their names, types, major numbers, and minor numbers

All major and minor numbers are assigned a name that matches up with a type of device

- This allocation is done by The LANANA [1]
- Current device list always is accessible at lanana.org [2]

The kernel cares only about type and numbers

Most applications only care about the name of the device node

[1] Linux Assigned Names And Numbers Authority
[2] http://www.lanana.org/docs/device-list/devices.txt

Special Properties

- Device type
  - block device or character device
- Major number
- Minor number

```
$ ls -l /dev/hda
brw-rw---- 1 root disk 3, 0 2007-11-29 16:43 /dev/hda
```

Description of block device with major number 3 at LANANA

```
3 block  First MFM, RLL and IDE hard disk/CD-ROM interface
                   0 = /dev/hda            Master: whole disk (or CD-ROM)
                  64 = /dev/hdb            Slave: whole disk (or CD-ROM)

                 For partitions, add to the whole disk device number:
                   0 = /dev/hd?            Whole disk
                   1 = /dev/hd?1           First partition
                   2 = /dev/hd?2           Second partition
                     ...
                  63 = /dev/hd?63          63rd partition
```

/dev

No dynamic allocation of entries

/dev is big (18 thousand entries)

No persistent naming

devfs

- Kernel module
- Dynamic and automatic population of /dev
- Hard to configure from the userspace
- Developed by Richard Gooch
  - Richard has stopped maintaining it
  - Contribution continued by Andrey Borzenkov
- Not maintained anymore

udev

Developed by Greg Kroah-Hartman and Kay Sievers

Runs in userspace

- Has userspace configuration
- Has userspace toolkit

Creates a dynamic /dev

Provides consistent device naming if wanted

Runs as daemon (udevd)

- Listens to uevents the kernel sends out
- Udev depends on the latest version of the uevent introduced with Linux kernel 2.6.13

udev tools

udevcontrol (located at /sbin/udevcontrol)

udevinfo (located at /usr/bin/udevinfo)

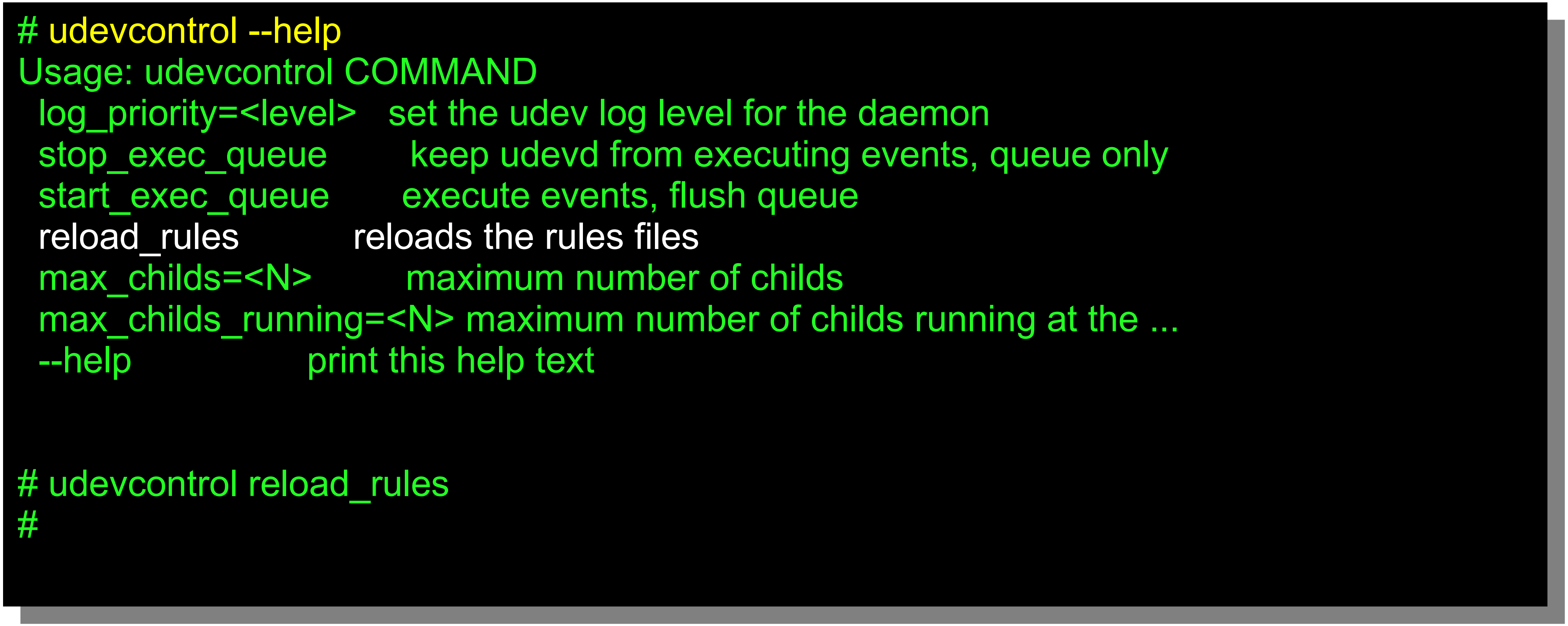
udevtest (located at /usr/bin/udevtest)

udevmonitor (located at /usr/sbin/udevmonitor)

Other tools

- udevsettle
- udevtrigger

udevcontrol



```
# udevcontrol --help
Usage: udevcontrol COMMAND
  log_priority=<level>   set the udev log level for the daemon
  stop_exec_queue        keep udevd from executing events, queue only
  start_exec_queue       execute events, flush queue
  reload_rules           reloads the rules files
  max_childs=<N>         maximum number of childs
  max_childs_running=<N> maximum number of childs running at the ...
  --help                 print this help text



# udevcontrol reload_rules
#
```

udevinfo

```
$ udevinfo --help
Usage: udevinfo OPTIONS
  --query=<type>     query database for the specified value:
    name             name of device node
    symlink          pointing to node
    path             sysfs device path
    env              the device related imported environment
    all              all values

  --path=<devpath>   sysfs device path used for query or chain
  --name=<name>      node or symlink name used for query

  --root             prepend to query result or print udev_root
  --attribute-walk   print all SYSFS_attributes along the device chain
  --export-db        export the content of the udev database
  --version          print udev version
  --help             print this text
```

udevinfo example

```
$ udevinfo --query=all --name=sdb
P: /block/sdb
N: sdb
S: disk/by-id/usb-CREATIVE_Zen_V_Plus__UMS__828A490D0002FA93
S: disk/by-path/pci-0000:00:10.4-usb-0:8:1.0-scsi-0:0:0:0
E: ID_VENDOR=CREATIVE
E: ID_MODEL=Zen_V_Plus_(UMS)
E: ID_REVISION=0001
E: ID_SERIAL=CREATIVE_Zen_V_Plus_(UMS)_828A490D0002FA93
E: ID_TYPE=disk
E: ID_BUS=usb
E: ID_PATH=pci-0000:00:10.4-usb-0:8:1.0-scsi-0:0:0:0
```

udevtest example

```
$ udevtest /sys/block/hda/
...
main: looking at device '/block/hda' from subsystem 'block'
...
udev_rules_get_name: add symlink 'disk/by-path/pci-0000:00:0f.0-ide-0:0'
run_program: 'vol_id --export /dev/.tmp-3-0'
run_program: '/lib/udev/vol_id' (stderr) '/dev/.tmp-3-0: error open volume'
run_program: '/lib/udev/vol_id' returned with status 2
udev_rules_get_name: no node name set, will use kernel name 'hda'
unlink_secure: chown(/dev/.tmp-3-0, 0, 0) failed: No such file or directory
unlink_secure: chmod(/dev/.tmp-3-0, 0000) failed: No such file or directory
udev_device_event: device '/block/hda' already in database, validate currently present symlinks
udev_node_add: creating device node '/dev/hda', major = '3', minor = '0', mode = '0660', ...
udev_node_add: creating symlink '/dev/creative' to 'hda'
udev_node_add: creating symlink '/dev/disk/by-path/pci-0000:00:0f.0-ide-0:0' to '../../hda'
udev_node_remove_symlinks: removing symlink '/dev/disk/by-id/ata-Maxtor_6Y080L0_Y36C4VQE'
delete_path: rmdir(/dev/disk/by-id) failed: Permission denied
main: run: 'socket:/org/kernel/udev/monitor'
main: run: 'socket:/org/freedesktop/hal/udev_event'
```

udevmonitor

```
# udevmonitor --help
Usage: udevmonitor [--help] [--env]
  --env     print the whole event environment
  --help    print this help text
```

udevmonitor example

```
# udevmonitor
udevmonitor prints the received event from the kernel [UEVENT]
and the event which udev sends out after rule processing [UDEV]

UEVENT[1196797419.657986] add@/devices/pci0000:00/0000:00:10.4/usb5/5-8
...
UDEV  [1196797419.670502] add@/devices/pci0000:00/0000:00:10.4/usb5/5-8/usbdev5.33_ep00
UDEV  [1196797419.670562] add@/devices/pci0000:00/0000:00:10.4/usb5/5-8/5-8:1.0
UEVENT[1196797419.682588] add@/class/scsi_host/host20
UEVENT[1196797419.684583] add@/devices/pci0000:00/0000:00:10.4/usb5/5-8/5-8:1.0/usbdev5.33_ep02
UEVENT[1196797419.684625] add@/devices/pci0000:00/0000:00:10.4/usb5/5-8/5-8:1.0/usbdev5.33_ep82
UEVENT[1196797419.684637] add@/class/usb_device/usbdev5.33
UDEV  [1196797419.922531] add@/class/scsi_host/host20
UDEV  [1196797419.922592] add@/devices/pci0000:00/0000:00:10.4/usb5/5-8/5-8:1.0/usbdev5.33_ep02
UDEV  [1196797419.922608] add@/devices/pci0000:00/0000:00:10.4/usb5/5-8/5-8:1.0/usbdev5.33_ep82
UDEV  [1196797420.056263] add@/class/usb_device/usbdev5.33
UEVENT[1196797424.718671] add@/devices/pci0000:00/0000:00:10.4/usb5/5-8/5- ...
UEVENT[1196797424.718717] add@/class/scsi_disk/20:0:0:0
UEVENT[1196797424.754980] add@/block/sdb
UEVENT[1196797424.755029] add@/class/scsi_device/20:0:0:0
UDEV  [1196797424.770187] add@/devices/pci0000:00/0000:00:10.4/usb5/5-8/5- ...
UDEV  [1196797424.810219] add@/class/scsi_disk/20:0:0:0
UDEV  [1196797424.810271] add@/block/sdb
UDEV  [1196797424.810286] add@/class/scsi_device/20:0:0:0
UEVENT[1196797425.702388] mount@/block/sdb
UDEV  [1196797425.705172] mount@/block/sdb
```

udev rules and configuration files

All udev configuration files are placed in /etc/udev/*

udev's main configuration file is /etc/udev/udev.conf

By default rules are located at /etc/udev/rules.d (Debian)

Rules must have the .rules suffix

udev.conf

udev_root

Specifies where to place the device nodes in the filesystem.
Default value is /dev.

udev_rules

The name of the udev rules file
Directory to look for files with the suffix .rules.
Default value is /etc/udev/rules.d.

udev_log

The logging priority. (err, info and debug)

Rules

Each rule is constructed from a series of comma separated key-value pairs

Example:

```
KERNEL==”hdb”, NAME=”my_disk”
```

Different types of keys

- Match keys
- Assignment keys

Real world example

```
KERNEL=="hdc", NAME="hdd"
KERNEL=="hdd", NAME="hdc"
```

```
$ udevtest /sys/block/hdc
...
main: looking at device '/block/hdc' from subsystem 'block'
...
udev_rules_get_name: rule applied, 'hdc' becomes 'hdd'
...
udev_device_event: device '/block/hdc' already in database, validate currently present symlinks
udev_node_add: creating device node '/dev/hdd', major = '22', minor = '0', mode = '0660', ...
udev_node_add: creating symlink '/dev/disk/by-path/pci-0000:00:0f.0-ide-1:0' to '../../hdd'
udev_node_remove_symlinks: removing symlink '/dev/cdrom1'
udev_node_remove_symlinks: removing symlink '/dev/cdrw1'
udev_node_remove_symlinks: removing symlink '/dev/dvd1'
udev_node_remove_symlinks: removing symlink '/dev/dvdrw1'
...
```

```
22 block     Second IDE hard disk/CD-ROM interface
               0 = /dev/hdc            Master: whole disk (or CD-ROM)
              64 = /dev/hdd            Slave: whole disk (or CD-ROM)

             Partitions are handled the same way as for the first
             interface (see major number 3).
```

Real world example (My creative rule!)

```
$ udevinfo --attribute-walk --name=sdb
...
looking at parent device '/devices/pci0000:00/0000:00:10.4/usb5/5-8':
    KERNELS=="5-8"
    SUBSYSTEMS=="usb"
    DRIVERS=="usb"
    ATTRS{configuration}=="Configuration 1"
    ATTRS{serial}=="828A490D0002FA93"
    ATTRS{product}=="Zen V Plus _UMS_"
    ATTRS{manufacturer}=="Creative Technology Ltd"
...
```

```
SUBSYSTEMS=="usb", ATTRS{serial}=="828A490D0002FA93", \
ATTRS{manufacturer}=="Creative Technology Ltd", \
SYMLINK+="creative"
```

Is everything okay?

```
$ udevtest /sys/block/sdb
...
parse_file: reading '/etc/udev/rules.d/local.rules' as rules file
...
main: looking at device '/block/sdb' from subsystem 'block'
udev_rules_get_name: add symlink 'creative'
...
udev_rules_get_name: 2 untrusted character(s) replaced
udev_rules_get_name: add symlink 'disk/by-id/usb-CREATIVE_Zen_V_Plus__UMS__828A490D0002FA93'
udev_rules_get_name: add symlink 'disk/by-path/pci-0000:00:10.4-usb-0:8:1.0-scsi-0:0:0:0'
udev_rules_get_name: no node name set, will use kernel name 'sdb'
...
udev_node_add: creating device node '/dev/sdb', major = '8', minor = '16', mode = '0660', ...
udev_node_add: creating symlink '/dev/creative' to 'sdb'
udev_node_add: creating symlink '/dev/disk/by-id/usb-CREATIVE_Zen_V_Plus__UMS__828A490D0002FA93' to '../../sdb'
udev_node_add: creating symlink '/dev/disk/by-path/pci-0000:00:10.4-usb-0:8:1.0-scsi-0:0:0:0' to '../../sdb'
...

$ udevtest /sys/block/sdb
...
parse_file: reading '/etc/udev/rules.d/local.rules' as rules file
add_to_rules: unknown key 'UBSYSTEMS' in /etc/udev/rules.d/local.rules:4
...
```

Append this line in to the fstab and enjoy

```
# echo "/dev/creative /media/creative vfat rw,user 0  0" >> /etc/fstab
```

Lets write a “Backup” script

```
SUBSYSTEMS=="usb", ATTRS{serial}=="828A490D0002FA93", \
ATTRS{manufacturer}=="Creative Technology Ltd", \
ACTION=="mount", RUN+="/usr/bin/creative_otf_backup"
```

```
#! /bin/bash
if [ "$ACTION" = mount ]; then
    rm -rf /media/creative/arag
    svn export /home/armen/projects/arag /media/creative/arag
fi
exit 0;
```

Anything else?

Controlling permissions and ownership

Using external programs to name devices

Running external programs upon certain events

References

http://reactivated.net/writing_udev_rules.html

http://wiki.archlinux.org/index.php/Using_udev_to_map_multiple_entries_to_a_device

http://www.hantslug.org.uk/talks/2007-08-04/udev-1.0.0.pdf

http://en.wikipedia.org/wiki/Udev

http://www.kroah.com/linux/talks/ols_2003_udev_paper/Reprint-Kroah-Hartman-OLS2003.pdf

http://www.kroah.com/linux/talks/oscon_2004_udev

udev

devfs done right

Managing Linux devices names in userspace

Armen Baghumian
armen@OpenSourceClub.org