Лабораторная работа №12. Управление устройствами и модулями Ядро ОС

Введение

Из этой лабораторной вы узнаете, что такое служба systemd-udevd и как с помощью ее правил заставить работать 4G-модем, который упорно притворяется, что он USB-флешка. Мы посмотрим внимательнее, что такое модули ядра, и научимся не только загружать драйверы, но и намертво блокировать работу отдельных устройств, а то вдруг заклеить камеру стикером нам покажется недостаточной мерой.

Управление устройствами

В операционной системе Linux наиболее полная информация об устройствах доступна в псевдофайловой системе sysfs, которая монтируется в каталог /sys. Что такое псевдофайловые системы и как они работают мы уже рассматривали в лабораторной по управлению процессами, и, как вы помните, sysfs пришла на замену procfs, но вот порядка в ней на порядок больше.

По мере развития системы procfs ее файлы потеряли однообразие и используют теперь совершенно разные форматы, в то время как в системе sysfs одному параметру соответствует строго один файл, поэтому намусорить там значительно сложнее. Имя файла в sysfs является именем параметра, а содержимое файла — его значением, и в подавляющем большинстве случаев это действительно так.

Псевдофайловая система sysfs

В sysfs находятся следующие каталоги:

• Каталог `/sys/block` — тут содержатся символические ссылки на блочные устройства, найденные в системе, по одной ссылке на каждое устройство. Символические ссылки указывают на соответствующие каталоги в `/sys/devices`.

• Каталог `/sys/bus` — содержит каталоги для всех типов шин ядра. Один каталог на один тип шины. При этом каждый из этих каталогов содержит два подкаталога:

- Каталог `/sys/bus/devices` — содержит символические ссылки на элементы в `/sys/devices`, соответствующие устройствам этого типа шины;
- Каталог `/sys/bus/drivers` — содержит каталоги с драйверами устройств, подгруженных для работы с этой шиной. По одному каталогу на один драйвер.

• Каталог `/sys/class` — содержит структуру из подкаталогов, по одному на каждый класс устройств, зарегистрированных в системе. Внутри размещаются символические ссылки на элементы в `/sys/devices`, соответствующие этим устройствам.

• Каталог `/sys/dev` — содержит два подкаталога `block` и `char` для блочных и символьных устройств соответственно. Внутри этих подкаталогов находятся символические ссылки на каталоги в `/sys/devices`, соответствующие этим устройствам. Имена ссылок имеют вид X:Y, где X – это основной ID, а Y – дополнительный ID устройства.

• Каталог `/sys/devices` — содержит отражение структур device в ядре ОС в виде дерева каталогов и файлов.

• Каталог `/sys/digsig` — содержит данные, связанные с работой в замкнутой программной среде (ЗПИ).

• Каталог `/sys/firmware` — содержит интерфейсы для просмотра и изменение параметров прошивок, например, UEFI.

• Каталог `/sys/fs` — содержит подкаталоги для некоторых файловых систем, в том числе и cgroup. Этот каталог обычно используется как точка монтирования для tmpfs, в которой содержатся точки монтирования файловых систем контрольных групп.

• Каталог `/sys/hypervisor` — используется в случае работы гипервизора на компьютере.

• Каталог `/sys/kernel` — содержит интерфейсы для просмотра информации о работе ядра ОС.

• Каталог `/sys/module` — содержит подкаталоги для всех подгруженных модулей ядра, по одному на каждый модуль.

• Каталог `/sys/power` — содержит интерфейсы для просмотра и изменения параметров управления питанием и режимами энергосбережения.

Правила udev

Служба udev является диспетчером устройств ядра Linux, который обрабатывает все события пользовательского пространства, возникающие при добавлении/удалении аппаратных устройств и изменении их характеристик.

Обработчики событий службы udev называют также правилами, и с их помощью, например, служба управления сетевыми подключениями NetworkManager автоматически перенастраивает сетевые интерфейсы при перемещении портативного устройства в другой сегмент сети. Вы можете создать собственные правила, с помощью которых можно выполнить переименование устройства или загрузку необходимых драйверов.

На рисунке представлена схема работы подсистемы udev, которая включает следующие шаги:

• Во время загрузки системы еще на самых ранних стадиях инициализации ядра создается псевдофайловая система **tmpfs**, которая монтируется в каталог `/dev`.

• В определенный момент стартует служба **systemd-udevd**, и в первую очередь она загружает в оперативную память базу данных правил (обработчиков), опираясь на файлы из каталогов `/usr/lib/udev/rules.d/` и `/etc/udev/rules.d/` (последние в приоритете).

Файлы правил имеют суффикс `*.rules` и загружаются в алфавитном порядке, поэтому в названиях файлов первые два символа используют в качестве числа, которое задает приоритет.

• После загрузки правил служба отправляет ядру запрос на передачу всех событий **uevent**, возникших в системе еще до запуска **systemd-udevd**. Обмен информацией между ядром и пользовательским пространством осуществляется через сокет **netlink**.

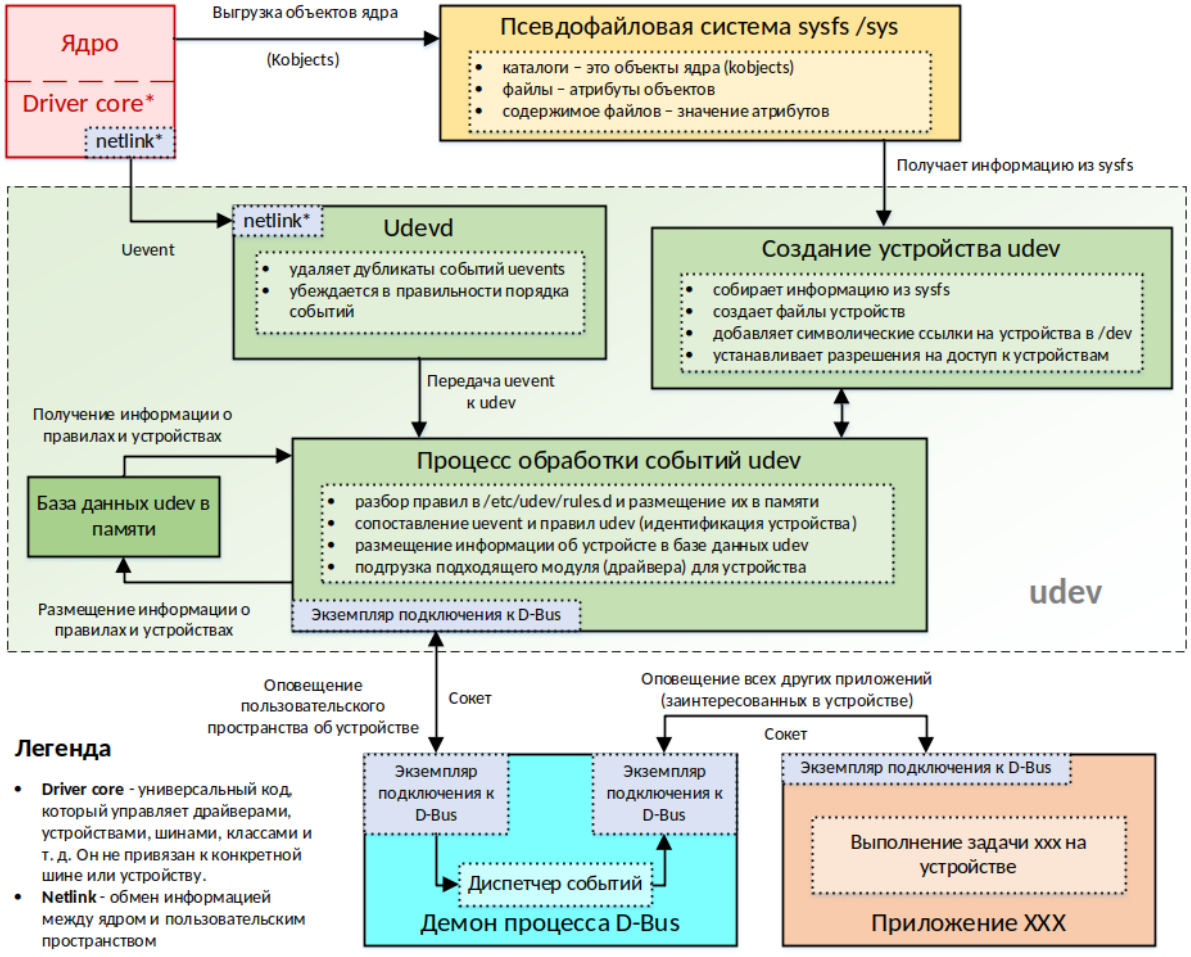


Схема работы подсистемы udev

• При подключении/отключении периферийного устройства (например, usb-накопителя), ядро ОС производит обновление своих структур и отражает эти изменения в псевдофайловой системе **sysfs**. При этом происходит оповещение службы **systemd-udevd** о произошедшем событии (**uevent**).

• Служба **systemd-udevd** удаляет дубликаты событий uevent и обеспечивает их сортировку в правильном порядке, чтобы события о подключении родительских устройств (например, шины usb) обрабатывались раньше, чем события о подключении дочерних устройств (например, usb-накопителя).

• Далее в соответствии с информацией из базы **udev** проводит поиск и применение правил, соответствующих пришедшему событию. В правилах описаны действия (скрипты), которые должны быть выполнены.

Кроме этого, udev собирает информацию об устройствах из **sysfs** и создает файлы устройств, выдает на них необходимые разрешения и добавляет символические ссылки в каталоге `/dev`. При необходимости udev отправляет ядру запросы на подгрузку необходимых модулей (драйверов устройств) и размещает информацию об устройствах в своей базе.

• На последнем этапе процесса служба **udev** через шину **D-Bus** оповещает о появлении нового устройства или изменении его физического состояния все приложения, заинтересованные в получении этой информации.

Примечание

Обработка событий в udev происходит параллельно, поэтому в файле `/etc/fstab` диски монтируются не по именам *sda* или *sdb*, а по их **UUID**, чтобы исключить ошибку, которая может возникнуть в связи с тем, что диски поменяются именами. Но в то же время с помощью правил udev имя устройства можно сделать постоянным.

Синтаксис правил

Правила **udev** хранятся в каталогах:

• Каталог `/usr/lib/udev/rules.d` — системные правила, созданные при установке системы или ПО.

• Каталог `/etc/udev/rules.d` — пользовательские правила, созданные администратором, более приоритетные (при совпадении имен файлов в этих двух каталогах).

Пример файла с правилами */usr/lib/udev/rules.d/50-udev-default.rules*:

```
# do not edit this file, it will be overwritten on update

# run a command on remove events
ACTION=="remove", ENV{REMOVE_CMD}!="", RUN+="$env{REMOVE_CMD}"
ACTION=="remove", GOTO="default_end"

SUBSYSTEM=="virtio-ports", KERNEL=="vport\*", ATTR{name}=="?\*", SYMLINK+="virtio-
ports/$attr{name }"

# select "system RTC" or just use the first one
SUBSYSTEM=="rtc", ATTR{hctosys}=="1", SYMLINK+="rtc"
SUBSYSTEM=="rtc", KERNEL=="rtc0", SYMLINK+="rtc", OPTIONS+="link_priority=-100"
...
```

Файлы правил имеют суффикс `*.rules` и состоят из множества строк, где каждая строка – это отдельное правило за исключением пустых строк с комментариями, начинающихся с символа решетки.

Каждое правило состоит из серии выражений `{ключ}{оператор}{значение}`, которые отделяются друг от друга запятыми. Выражения должны задавать как минимум один ключ соответствия и один ключ присвоения, где:

- Ключ соответствия — это условие, по которому определяется, соответствует ли конкретное устройство данному правилу или нет. Если ключей соответствия несколько, то правило применяется только в том случае, если устройство удовлетворяет сразу всем указанным критериям.
- Ключ присвоения — это действие, которое должно быть выполнено в системе, если текущее устройство удовлетворяет критериям ключей соответствия.

Ключи соответствия:

Ключи соответствия могут определять критерии как для самого устройства, так и для его родительских устройств. Например, для жесткого диска родителем будет устройство **SCSI**, а для него, в свою очередь, **шина BUS**).

Большинство ключей поддерживают шаблоны подстановки (*от англ. wildcard pattern*) в стиле оболочки: «*», «?», «[]», «|».

Основные ключи для критериев соответствия самого устройства:

- **KERNEL** — соответствие с именем устройства в ядре ОС;
- **SUBSYSTEM** — соответствие с подсистемой устройства;
- **DRIVER** — соответствие с именем драйвера, обслуживающего устройство;
- **DEVPATH** — соответствие пути до устройства в *sysfs*;
- **ATTR{атрибут}** — соответствие содержимому файла для этого атрибута в каталоге устройства *sysfs*;
- **ENV{переменная}** — соответствие значению переменной окружения.

Аналогичные ключи для определения соответствия как самому устройству, так и устройствам-предкам (родителям, дедушкам и т.д.):

- **KERNELS** — соответствие с именем устройства и/или его предка в ядре ОС;
- **SUBSYSTEMS** — соответствие с подсистемой устройства и/или его предка;
- **DRIVERS** — соответствие с именем драйвера, обслуживающего устройство и/или его предка;

• **ATTRS{атрибут}** — соответствие содержимому файла для этого атрибута в каталоге *sysfs* устройства и/или его предка.

С ключами соответствия используются два простых оператора:

• Оператор «==» — устройство соответствует критерию;
• Оператор «!=» — устройство не соответствует критерию.

Ключи присвоения (действий)

Ниже представлены основные ключи для действий, которые будут выполнены в случае, если правило применимо:

• **NAME** — устанавливает имя файла устройства в каталоге `/dev`;
• **SYMLINK** — создает символические ссылки, который действуют как альтернативные имена для этого устройства;
• **RUN** — задает список команд для запуска (это может быть имя скрипта с необходимыми операциями);
• **OWNER** — устанавливает владельца для файла устройства;
• **GROUP** — устанавливает группу-владельца для файла устройства;
• **MODE** — устанавливает права доступа на файл устройства.

Все вышеперечисленные ключи действий поддерживают шаблоны подстановки, подобные в команде `printf`:

• Шаблон `%k` — имя устройства, данное ему ядром,
• Шаблон `%n` — номер устройства (например, для sda2, номером является число 2),
• Шаблон `%p` — путь к файлу устройства,
• Шаблон `$name` — текущее имя устройства,
• Шаблон `$env{ключ}` — значение переменной окружения,
• Шаблон `$driver` — драйвер устройства.

С ключами присвоения используются следующие операторы:

• Оператор «=» — очистить список и добавить указанное действие;
• Оператор «+=» — добавить указанное действие без очистки списка;
• Оператор «-=» — удалить указанное действие из списка;
• Оператор «:=» — присвоить значение и запретить его изменение другими правилами.

Инструменты управления udev

Для управления udev предназначена утилита `udevadm`. Она управляет поведением **systemd-udevd** во время выполнения, запрашивает события ядра, управляет очередью событий и предоставляет простые механизмы отладки.

Основные опции утилиты udevadm:

• Команда `udevadm info <имя_устройства или путь_до_устройства>` — выводит информацию об устройстве из базы данных udev.

• Команда `udevadm test <путь_до_устройства_в_sysfs>` — имитирует запуск события udev для выбранного устройства и выводит отладочную информацию. Полезно для проверки работоспособности правил.

• Команда `udevadm monitor` — выводит события uevent в режиме реального времени.

• Команда `udevadm settle` — наблюдает за очередью событий udev. Если все события обработаны, через определенное время (по умолчанию 120 секунд) команда завершает свою работу.

• Команда `udevadm control` — управляет внутренним состоянием запущенного демона udev.

udevadm info

Для вывода информации об устройствах используется команда `udevadm info <имя_устройства/путь_до_устройства>`. Если необходимо вывести информацию и об устройствах-предках, то следует добавить параметр `-a` в команду `udevadm info -a <имя_устройства/путь_до_устройства>`.

Например, выведем информацию о диске `udevadm info /dev/sda`.

```
sa@astra:~$ udevadm info /dev/sda
P: /devices/pci0000:00/0000:00:0d.0/ata1/host0/target0:0:0/0:0:0:0/block/sda
N: sda
L: 0
S: disk/by-id/ata-VBOX_HARDDISK_VB6a0f90fb-c5f1dd19
S: disk/by-path/pci-0000:00:0d.0-ata-1
E: DEVPATH=/devices/pci0000:00/0000:00:0d.0/ata1/host0/target0:0:0/0:0:0:0/block/sda
E: DEVNAME=/dev/sda
...
```

udevadm test

Имитировать событие **udev** для выбранного устройства и вывести отладочную информацию можно командой `udevadm test <путь_до_устройства_в_sysfs>` (**DEVPATH**). Реальных изменений не происходит, полезно для проверки работоспособности правил.

```
sa@astra:~$ udevadm test /sys/block/sda
```

```
This program is for debugging only, it does not run any program
specified by a RUN key. It may show incorrect results, because
some values may be different, or not available at a simulation run.

Load module index
Network interface NamePolicy= disabled on kernel command line, ignoring.
Parsed configuration file /usr/lib/systemd/network/99-default.link
Created link configuration context.
Reading rules file: /usr/lib/udev/rules.d/50-firmware.rules
Reading rules file: /usr/lib/udev/rules.d/50-udev-default.rules
Reading rules file: /usr/lib/udev/rules.d/55-dm.rules
Reading rules file: /usr/lib/udev/rules.d/56-hpmud.rules
Reading rules file: /usr/lib/udev/rules.d/60-block.rules
Reading rules file: /usr/lib/udev/rules.d/60-cdrom_id.rules
. . .
Reading rules file: /usr/lib/udev/rules.d/99-systemd.rules
Rules contain 196608 bytes tokens (16384 * 12 bytes), 26851 bytes strings
19410 strings (154657 bytes), 16959 de-duplicated (130258 bytes), 2452 trie nodes used
Invalid inotify descriptor.
Starting 'ata_id --export /dev/sda'
Process 'ata_id --export /dev/sda' failed with exit code 1.
Starting 'scsi_id --export --whitelisted -d /dev/sda'
'scsi_id --export --whitelisted -d /dev/sda'(out) 'ID_SCSI=1'
'scsi_id --export --whitelisted -d /dev/sda'(out) 'ID_VENDOR='
'scsi_id --export --whitelisted -d /dev/sda'(out) 'ID_VENDOR_ENC='
'scsi_id --export --whitelisted -d /dev/sda'(out) 'ID_MODEL='
'scsi_id --export --whitelisted -d /dev/sda'(out) 'ID_MODEL_ENC='
'scsi_id --export --whitelisted -d /dev/sda'(out) 'ID_REVISION='
'scsi_id --export --whitelisted -d /dev/sda'(out) 'ID_TYPE='
Process 'scsi_id --export --whitelisted -d /dev/sda' succeeded.
DEVPATH=/devices/pci0000:00/0000:00:0d.0/ata1/host0/target0:0:0/0:0:0:0/block/sda
DEVNAME=/dev/sda
DEVTYPE=disk
DISKSEQ=9
MAJOR=8
MINOR=0
ACTION=add
SUBSYSTEM=block
ID_SCSI=1
ID_BUS=scsi
ID_PATH=pci-0000:00:0d.0-ata-1
ID_PATH_TAG=pci-0000_00_0d_0-ata-1
DEVLINKS=/dev/disk/by-path/pci-0000:00:0d.0-ata-1
TAGS=:systemd:
USEC_INITIALIZED=2547531
Unload module index
Unloaded link configuration context.
```

udevadm monitor

Команда `udevadm monitor` используется для вывода событий **uevent** в режиме реального времени. Если запустить утилиту и после этого вставить оптический диск в виртуальную машину, то мы увидим соответствующее событие.

```
sa@astra:~$ udevadm monitor
monitor will print the received events for:
UDEV - the event which udev sends out after rule processing
KERNEL - the kernel uevent

KERNEL[237008.787520] change
/devices/pci0000:00/0000:00:01.1/ata3/host2/target2:0:0/2:0:0:0/b lock/sr0 (block)
UDEV [237008.836023]  change
/devices/pci0000:00/0000:00:01.1/ata3/host2/target2:0:0/2:0:0:0/b lock/sr0 (block)
```

udevadm control

Для управления внутренним состоянием запущенного демона **udev** и диагностики его работы используются команды `sudo udevadm control [параметры]`.

• Команда `sudo udevadm control --ping` — проверяет, запущена ли служба **systemd-udevd** и отвечает ли она на запросы.

• Команда `sudo udevadm control --reload` — заставляет службу перечитать правила **udev** и других баз данных (например, индекс модуля ядра). Изменения не повлияют на уже существующие устройства. Новые правила будут применяться только к новым событиям **uevent**.

• Команда `sudo udevadm control --stop-exec-queue` — приостановит обработку новых событий службой **systemd-udevd**. События будут помещаться в очередь до востребования.

• Команда `sudo udevadm control --start-exec-queue` — возобновит обработку событий службой **systemd-udevd** после выполнения предыдущей команды `stop-exec-queue`.

• Команда `sudo udevadm control -log-priority=<уровень_журналирования>` — установит уровень журналирования службы **systemd-udevd**. Может принимать одно из следующих значений: **emerg, alert, crit, err, warning, notice, info, debug**.

Проверим работу команд на следующем примере:

• остановим обработку сообщений **uevent** командой `sudo udevadm control --stop-exec-queue`;

• Далее необходимо переподключить оптический диск;

• запустим `udevadm settle` в отдельном терминале и убедимся, что команда не завершила свою работу мгновенно, так как в очереди есть необработанные сообщения;

• возобновим обработку сообщений `sudo udevadm control --start-exec-queue` и убедимся, что команда `udevadm settle` завершила свою работу, т.к. все сообщения из очереди были обработаны.

Создание правила udev

Создадим правило **udev** для usb-накопителя, которое позволит логировать серийный номер и время подключения устройства к компьютеру.

Во-первых, создадим сам файл для правила в каталоге `/etc/udev/rules.d/` с помощью команды:

```
sudo touch /etc/udev/rules.d/99-usb-drive.rules
```

Во-вторых, создадим в текстовом редакторе скрипт `/usr/bin/usb_disk_log.sh`, который будет записывать необходимую нам информацию в файл.

```
#!/bin/bash
/bin/echo "В $(/bin/date) был вставлен usb-диск. Серийный номер: $1" >> /var/log/usb_disk.log
```

Разрешим выполнение файла `usb_disk_log.sh` командой

```
sa@astra:~$ sudo chmod 744 /usr/bin/usb_disk_log.sh
```

Проверим работоспособность скрипта командой `sudo /usr/bin/usb_disk_log.sh 1234` — в файле `/var/log/usb_disk.log` должна появиться соответствующая строка.

```
sa@astra:~$ sudo /usr/bin/usb_disk_log.sh 1234
sa@astra:~$ cat /var/log/usb_disk.log
В Сб апр 19 13:55:46 MSK 2025 был вставлен usb-диск. Серийный номер: 1234
```

Теперь посмотрим параметры нашего usb-накопителя. Для этого подключим его к хостовой и гостевой машине и выведем информацию о пути до этого устройства `udevadm info /dev/sdb1 | grep DEVPATH | awk -F '=' '{print $2}'`.

```
sa@astra:~$ udevadm info /dev/sdb1 | grep DEVPATH | awk -F '=' '{print $2}'
/devices/pci0000:00/0000:00:0c.0/usb1/1-2/1-2:1.0/host3/target3:0:0/3:0:0:0/block/sdb/sdb1
```

И информацию о его атрибутах:

```
sa@astra:~$ udevadm info --attribute-walk /dev/sdb1 | grep -P '(usb|block)'
 looking at device '/devices/pci0000:00/0000:00:0c.0/usb1/1-2/1-
2:1.0/host3/target3:0:0/3:0:0:0/block/sdb/sdb1':
   SUBSYSTEM=="block"
 looking at parent device '/devices/pci0000:00/0000:00:0c.0/usb1/1-2/1-
2:1.0/host3/target3:0:0/3:0:0:0/block/sdb':
   SUBSYSTEMS=="block"
 looking at parent device '/devices/pci0000:00/0000:00:0c.0/usb1/1-2/1-
2:1.0/host3/target3:0:0/3:0:0:0':
   ATTRS{device_blocked}=="0"
 looking at parent device '/devices/pci0000:00/0000:00:0c.0/usb1/1-2/1-
2:1.0/host3/target3:0:0':
 looking at parent device '/devices/pci0000:00/0000:00:0c.0/usb1/1-2/1-2:1.0/host3':
 looking at parent device '/devices/pci0000:00/0000:00:0c.0/usb1/1-2/1-2:1.0':
   SUBSYSTEMS=="usb"
   DRIVERS=="usb-storage"
 looking at parent device '/devices/pci0000:00/0000:00:0c.0/usb1/1-2':
   SUBSYSTEMS=="usb"
   DRIVERS=="usb"
 looking at parent device '/devices/pci0000:00/0000:00:0c.0/usb1':
   KERNELS=="usb1"
   SUBSYSTEMS=="usb"
   DRIVERS=="usb"
```

В созданный нами файл правила `/etc/udev/rules.d/99-usb-drive.rules` добавим следующий текст:

```
SUBSYSTEM=="block", DRIVERS=="usb-storage", RUN+="/usr/bin/usb_disk_log.sh $env{ID_SERIAL}"
```

Протестируем наше правило:

```
sa@astra:~$ udevadm test `udevadm info /dev/sdb1 | grep DEVPATH | awk -F '=' '{print $2}'`
Load module index
Network interface NamePolicy= disabled on kernel command line, ignoring.
Parsed configuration file /usr/lib/systemd/network/99-default.link
Created link configuration context.
```

```
Reading rules file: /usr/lib/udev/rules.d/50-firmware.rules
Reading rules file: /usr/lib/udev/rules.d/50-udev-default.rules
Reading rules file: /usr/lib/udev/rules.d/55-dm.rules
. . .
USEC_INITIALIZED=14642379
run: '/usr/bin/usb_disk_log.sh SanDisk_Cruzer_Micro_3514820EDED27F47-0:0'
Unload module index
Unloaded link configuration context.
```

Теперь перезагрузим правила командой `sudo udevadm control --reload` и повторно подключим «флешку» к нашей гостевой машине. В журнале `usb_disk.log` появится новая информация:

```
sa@astra:~$ cat /var/log/usb_disk.log
В Сб апр 19 13:55:46 MSK 2025 был вставлен usb-диск. Серийный номер: 1234
В Sat Apr 19 13:58:32 MSK 2025 был вставлен usb-диск. Серийный номер:
SanDisk_Cruzer_Micro_3514820EDED27F47-0:0
```

Еще несколько практических примеров:

Некоторые модели USB-модемов могут по ошибке определяться как USB-накопители. Чтобы исправить эту проблему, можно создать правило udev и настроить его на выполнение команды для смены режима работы модема. Для этого нужно:

1. Создать файл правила `/etc/udev/rules.d/10-modem_mode_switch.rules` с содержимым:

```
ACTION=="add",SUBSYSTEM=="usb",ATTRS{idVendor}=="12d1",ATTRS{idProduct}=="1f01",RUN+="/usr/sbi
n/usb_modeswitch --quiet --config-file /usr/share/usb_modeswitch/12d1:1f01"
```

2. Создать конфигурационный файл `/usr/share/usb_modeswitch/12d1:1f01` с содержимым:

```
DefaultVendor = 0x12d1
DefaultProduct = 0x1f01
TargetVendor = 0x12d1
TargetProduct = 0x14dc
# switch to 12d1:14dc (default HiLink CDC-Ether mode)
MessageContent="555342431234567800000000000000a11062000000000000100000000
000000"
```

После выполнения указанных действий при подключении устройства будет выполняться требуемая команда, и он будет работать как USB-модем.

Когда на сервере несколько сетевых карт, с помощью udev правил сетевым интерфейсам можно присвоить собственные имена, чтобы было проще ориентироваться. Например, создадим файл `/etc/udev/rules.d/10-interface_rename.rules` с содержимым:

```
SUBSYSTEM=="net", ACTION=="add", ATTR{address}=="08:00:27:48:15:07", NAME="localnet"
```

Это правило переименует сетевой интерфейс с MAC-адресом 08:00:27:48:15:07 в localnet, и вы будете точно знать, что интерфейс «смотрит» в локальную сеть.

Протестируем созданное нами правило. Для этого сначала найдем сетевое устройство в каталоге /sys

```
sa@astra:~$ find /sys -name *eth*
...
/sys/devices/pci0000:00/0000:00:03.0/net/eth0
...
```

Затем запустим тест и убедимся, что правило отработает:

```
sa@astra:~$ sudo udevadm test /sys/devices/pci0000:00/0000:00:03.0/net/eth0
...
SYSTEMD_ALIAS=/sys/subsystem/net/devices/localnet
...
```

Теперь можно либо перезаргузить машину, либо дать такой набор команд:

```
sa@astra:~$ sudo ip link set eth0 down
sa@astra:~$ sudo udevadm control --reload-rules
sa@astra:~$ sudo udevadm trigger
sa@astra:~$ sudo ip link set localnet up
```

Проверяем:

```
sa@astra:~$ sudo ip link show
```

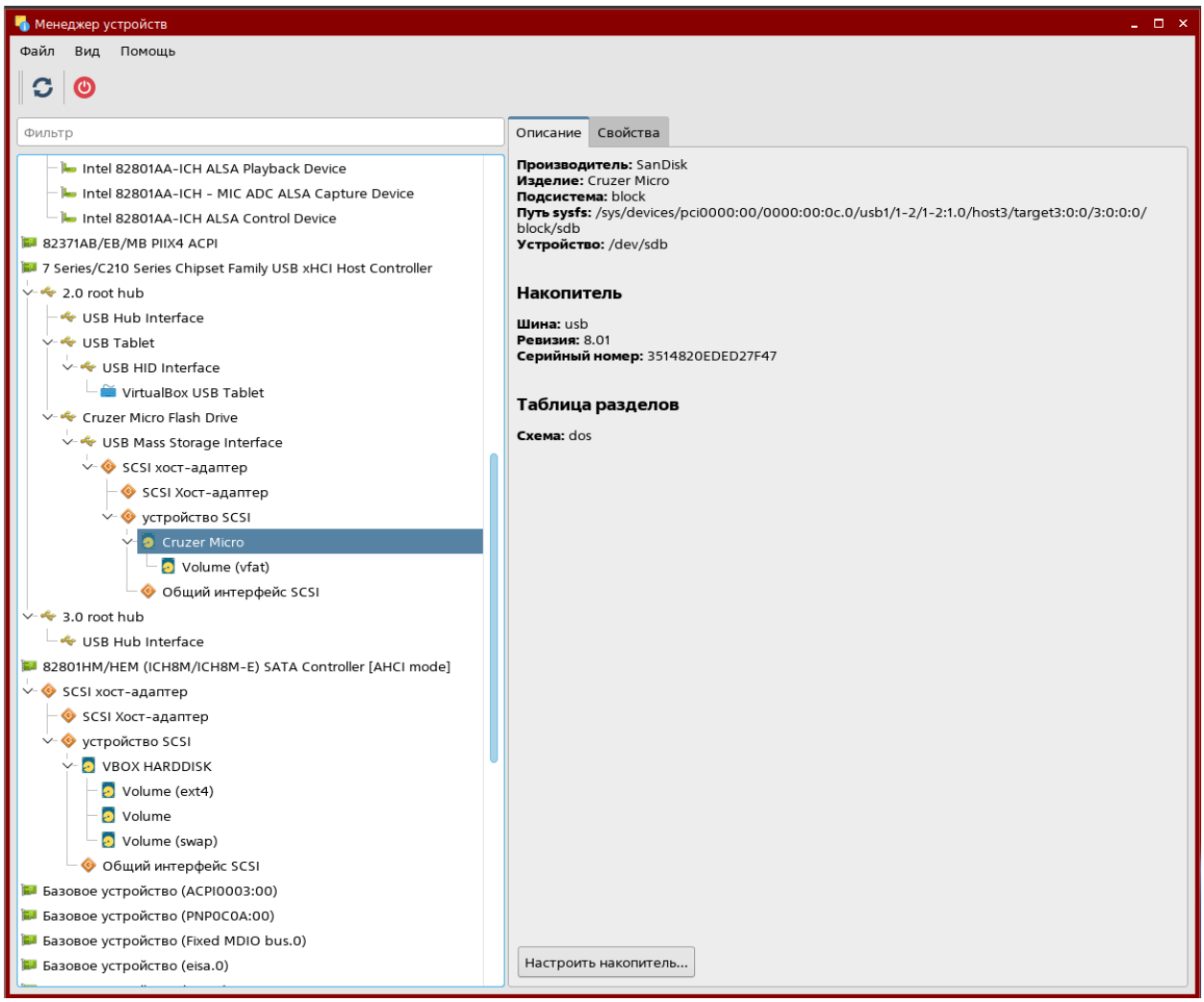
По тому же принципу можно создать правило для интерфейса в Интернет. Не забудьте только изменить сетевые настройки в соответствии с новыми именами интерфейсов.

Просмотр информации об устройствах

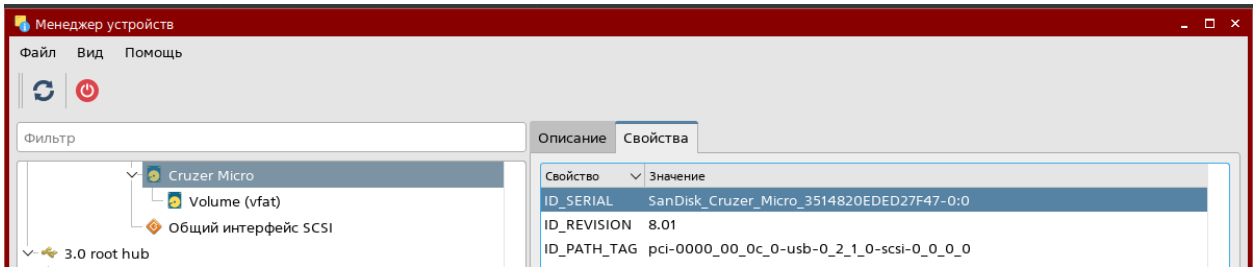
Для просмотра информации об устройствах кроме утилиты udevadm и псевдофайловой системы sysfs существуют и другие способы.

Графический менеджер устройств

Менеджер устройств (fly-admin-device-manager) позволяет просматривать и управлять устройствами, которые зарегистрированы в системе, так же удобно, как мы это могли делать в диспетчере устройств Windows. Запустить его можно через меню пуск **Пуск ▸ Системные ▸ Менеджер устройств**.



Графический менеджер устройств



Свойства выбранного устройства

Утилиты для просмотра информации об устройствах

Для командной строки тоже есть ряд удобных утилит.

Утилита `lspci` выводит информацию об устройствах, подключенных к шине PCI.

```
sa@astra:~$ lspci
00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)
00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]
00:01.1 IDE interface: Intel Corporation 82371AB/EB/MB PIIX4 IDE (rev 01)
00:02.0 VGA compatible controller: VMware SVGA II Adapter
00:03.0 Ethernet controller: Intel Corporation 82540EM Gigabit Ethernet Controller (rev 02)
```

```
00:04.0 System peripheral: InnoTek Systemberatung GmbH VirtualBox Guest Service
00:05.0 Multimedia audio controller: Intel Corporation 82801AA AC'97 Audio Controller (rev 01)
00:07.0 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 08)
00:0c.0 USB controller: Intel Corporation 7 Series/C210 Series Chipset Family USB xHCI Host
Controller
00:0d.0 SATA controller: Intel Corporation 82801HM/HEM (ICH8M/ICH8M-E) SATA Controller [AHCI
mode] (rev 02)
```

Утилита `lsusb` выводит информацию об устройствах, подключенных к шине USB.

```
sa@astra:~$ lsusb
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 003: ID 0781:5151 SanDisk Corp. Cruzer Micro Flash Drive
Bus 001 Device 002: ID 80ee:0021 VirtualBox USB Tablet
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

Утилита `lscpu` выводит информацию о процессорах.

```
sa@astra:~$ lscpu
Архитектура:             x86_64
CPU op-mode(s):          32-bit, 64-bit
Порядок байт:            Little Endian
Address sizes:           39 bits physical, 48 bits virtual
CPU(s):                  1
. . .
```

Утилита `lshw` выводит информацию о всех устройствах в системе. Доступна в расширенном репозитории для установки.

```
sa@astra:~$ sudo apt install lshw
Чтение списков пакетов… Готово
Построение дерева зависимостей
. . .
Обрабатываются триггеры для man-db (2.8.5-2) …

sa@astra:~$ sudo lshw -short
H/W path                Device      Class       Description
============================================================
                                    system      VirtualBox
/0                                   bus         VirtualBox
/0/0                                 memory      128KiB BIOS
/0/1                                 memory      4GiB System memory
/0/2                                 processor   12th Gen Intel(R) Core(TM) i5-1240P
/0/100                               bridge      440FX - 82441FX PMC [Natoma]
/0/100/1                             bridge      82371SB PIIX3 ISA [Natoma/Triton II]
/0/100/1.1              scsi2        storage     82371AB/EB/MB PIIX4 IDE
...
```

Модули ядра операционной системы

Модули ядра — это объектный файл, который содержит код, расширяющий функциональность базового ядра операционной системы. Модули бывают встроены в ядро или могут загружаться во время работы операционной системы. Самый простой пример модуля — это драйверы различных устройств.

Размещение модулей ядра

Все модули для установленных ядер хранятся директории `/lib/modules/`.

```
sa@astra:~$ ls -l /lib/modules
итого 12
drwxr-xr-x 6 root root 4096 апр 19 13:42 5.15.0-33-generic
drwxr-xr-x 3 root root 4096 мар 22  2023 5.15.0-33-hardened
drwxr-xr-x 3 root root 4096 мар 22  2023 5.15.0-33-lowlatency
```

Для каждой версии ядра создается свой подкаталог с необходимыми для него модулями. Содержимое каталога для текущий версии ядра можем посмотреть командой:

```
sa@astra:~$ ls -l /lib/modules/$(uname -r)
итого 6332
lrwxrwxrwx  1 root root      40 окт 25  2022 build -> /usr/src/linux-headers-5.15.0-33-generic
drwxr-xr-x  2 root root    4096 апр 19 13:42 initrd
drwxr-xr-x 14 root root    4096 мар 22  2023 kernel
drwxr-xr-x  2 root root    4096 апр 19 13:42 misc
-rw-r--r--  1 root root 1491044 апр 19 13:42 modules.alias
-rw-r--r--  1 root root 1464259 апр 19 13:42 modules.alias.bin
-rw-r--r--  1 root root   10441 окт 25  2022 modules.builtin
-rw-r--r--  1 root root   26058 апр 19 13:42 modules.builtin.alias.bin
-rw-r--r--  1 root root   12859 апр 19 13:42 modules.builtin.bin
-rw-r--r--  1 root root   81997 окт 25  2022 modules.builtin.modinfo
-rw-r--r--  1 root root  704625 апр 19 13:42 modules.dep
-rw-r--r--  1 root root  966046 апр 19 13:42 modules.dep.bin
-rw-r--r--  1 root root     314 апр 19 13:42 modules.devname
-rw-r--r--  1 root root  241182 окт 25  2022 modules.order
-rw-r--r--  1 root root    1786 апр 19 13:42 modules.softdep
-rw-r--r--  1 root root  649608 апр 19 13:42 modules.symbols
-rw-r--r--  1 root root  785484 апр 19 13:42 modules.symbols.bin
drwxr-xr-x  3 root root    4096 мар 22  2023 vdso
```

У файлов модулей расширение `*.ko`. Посмотреть все модули для текущей версии можно командой:

```
sa@astra:~$ find /lib/modules/$(uname -r) -name *.ko | head
/lib/modules/5.15.0-33-generic/kernel/kernel/kheaders.ko
/lib/modules/5.15.0-33-generic/kernel/arch/x86/kernel/cpuid.ko
/lib/modules/5.15.0-33-generic/kernel/arch/x86/kernel/msr.ko
/lib/modules/5.15.0-33-generic/kernel/arch/x86/kernel/cpu/mce/mce-inject.ko
/lib/modules/5.15.0-33-generic/kernel/arch/x86/platform/atom/punit_atom_debug.ko
/lib/modules/5.15.0-33-generic/kernel/arch/x86/kvm/kvm.ko
/lib/modules/5.15.0-33-generic/kernel/arch/x86/kvm/kvm-amd.ko
/lib/modules/5.15.0-33-generic/kernel/arch/x86/kvm/kvm-intel.ko
/lib/modules/5.15.0-33-generic/kernel/arch/x86/events/amd/amd-uncore.ko
/lib/modules/5.15.0-33-generic/kernel/arch/x86/events/rapl.ko
```

При изменении состава модулей выполняется команда `depmod`, которая пересоздает базу данных установленных модулей и их зависимостей. Файл с базой можно посмотреть командой:

```
sa@astra:~$ cat /lib/modules/$(uname -r)/modules.dep | head
kernel/arch/x86/events/amd/amd-uncore.ko:
kernel/arch/x86/events/intel/intel-cstate.ko:
kernel/arch/x86/events/rapl.ko:
kernel/arch/x86/kernel/cpu/mce/mce-inject.ko:
kernel/arch/x86/kernel/msr.ko:
kernel/arch/x86/kernel/cpuid.ko:
```

```
kernel/arch/x86/crypto/twofish-x86_64.ko: kernel/crypto/twofish_common.ko
kernel/arch/x86/crypto/twofish-x86_64-3way.ko: kernel/arch/x86/crypto/twofish-x86_64.ko
kernel/crypto/twofish_common.ko
kernel/arch/x86/crypto/twofish-avx-x86_64.ko: kernel/crypto/crypto_simd.ko
kernel/crypto/cryptd.ko kernel/arch/x86/crypto/twofish-x86_64-3way.ko kerne
l/arch/x86/crypto/twofish-x86_64.ko kernel/crypto/twofish_common.ko
kernel/arch/x86/crypto/serpent-sse2-x86_64.ko: kernel/crypto/serpent_generic.ko
kernel/crypto/crypto_simd.ko kernel/crypto/cryptd.ko
```

Просмотр модулей ядра

Для просмотра модулей ядра, загруженных в данный момент, служит команда `lsmod`.

```
sa@astra:~$ lsmod
Module                  Size  Used by
vboxvideo              36864  0
drm_ttm_helper         16384  1 vboxvideo
intel_rapl_msr         20480  0
intel_rapl_common      32768  1 intel_rapl_msr
intel_powerclamp       20480  0
. . .
psmouse               172032  0
usbhid                 65536  0
i2c_piix4              28672  0
hid                   147456  2 usbhid,hid_generic
drm_kms_helper        303104  2 vmwgfx,vboxvideo
syscopyarea            16384  1 drm_kms_helper
sysfillrect            20480  1 drm_kms_helper
sysimgblt              16384  1 drm_kms_helper
e1000                 151552  0
fb_sys_fops            16384  1 drm_kms_helper
cec                    61440  1 drm_kms_helper
rc_core                61440  1 cec
drm                   598016  7 vmwgfx,drm_kms_helper,vboxvideo,drm_ttm_helper,ttm
pata_acpi              16384  0
video                  53248  0
parsec                249856  0
```

Она выводит таблицу, в которой три столбца:

- **Module** – название модуля.
- **Size** – размер модуля в байтах.
- **Used by** – показывает количество экземпляров модуля, используемое в настоящее время, и список зависимостей, указываются через запятую. Значение ноль говорит о том, что модуль не используется.

Для вывода подробной информации о загруженном модуле, используется команда:

```
modinfo <имя_модуля>
```

Выведем информацию о модуле, который реализует драйвер сетевого адаптера e1000 с помощью команды:

```
sa@astra:~$ sudo modinfo e1000
[sudo] пароль для sa:
ilename:       /lib/modules/5.15.0-33-generic/kernel/drivers/net/ethernet/intel/e1000/e1000.ko
license:        GPL v2
description:    Intel(R) PRO/1000 Network Driver
```

```
author:         Intel Corporation, <linux.nics@intel.com>
srcversion:     B4D2D3AB3E4A089C43DAE56
alias:          pci:v00008086d00002E6Esv*sd*bc*sc*i*
. . .
alias:          pci:v00008086d00001000sv*sd*bc*sc*i*
depends:
retpoline:      Y
intree:         Y
name:           e1000
vermagic:       5.15.0-33-generic SMP mod_unload modversions
parm:           TxDescriptors:Number of transmit descriptors (array of int)
parm:           RxDescriptors:Number of receive descriptors (array of int)
parm:           Speed:Speed setting (array of int)
parm:           Duplex:Duplex setting (array of int)
parm:           AutoNeg:Advertised auto-negotiation setting (array of int)
parm:           FlowControl:Flow Control setting (array of int)
parm:           XsumRX:Disable or enable Receive Checksum offload (array of int)
parm:           TxIntDelay:Transmit Interrupt Delay (array of int)
parm:           TxAbsIntDelay:Transmit Absolute Interrupt Delay (array of int)
parm:           RxIntDelay:Receive Interrupt Delay (array of int)
parm:           RxAbsIntDelay:Receive Absolute Interrupt Delay (array of int)
parm:           InterruptThrottleRate:Interrupt Throttling Rate (array of int)
parm:           SmartPowerDownEnable:Enable PHY smart power down (array of int)
parm:           copybreak:Maximum size of packet that is copied to a new buffer on receive (uint)
parm:           debug:Debug level (0=none,...,16=all) (int)
```

Управление модулями ядра

Для управления модулями ядра есть несколько инструментов:

- команды для загрузки и выгрузки модулей ядра;
- конфигурационные файлы загрузки модулей;
- конфигурационные файлы для автозагрузки модулей;
- конфигурационные файлы для запрета загрузки модулей (черные списки);
- конфигурационные файлы для задания параметров модулей.

Конфигурационные файлы находятся в следующих каталогах:

- Каталоги `/etc/modules-load.d/` и `/usr/lib/modules-load.d/` – каталоги с конфигурационными файлами *.conf для автозагрузки модулей при старте системы;
- Каталоги `/etc/modprobe.d/` и `/lib/modprobe.d/` – каталоги с конфигурационными файлами `*.conf` для блокировки модулей, создания псевдонимов и настройки параметров модулей.

Загрузка и выгрузка модулей ядра

Загрузка модулей ядра осуществляется командой:

`modprobe [параметры] <имя_модуля>` или `insmod <полный_путь_до_файла_модуля>`.

Выгрузка модулей осуществляется командой:

`modprobe -r <имя_модуля>` или `rmmode <имя_модуля>`.

Выгрузим модуль e1000 и загрузим его заново:

```
sa@astra:~$ sudo modprobe -r e1000 && sudo modprobe e1000
```

При выполнении команды вы могли заметить, что сетевое соединение прерывалось.

У команды `modprobe` кроме ключа `-r` есть еще много параметров. С полным списком вы можете ознакомиться в справке `man modprobe`. Рассмотрим только пару наиболее часто используемых:

• Ключ `--show-depends` – вывод зависимостей модуля (в том числе и файл самого модуля), например:

```
sa@astra:~$ sudo modprobe --show-depends ttm
insmod /lib/modules/5.15.0-70-generic/kernel/drivers/gpu/drm/drm.ko
insmod /lib/modules/5.15.0-70-generic/kernel/drivers/gpu/drm/ttm/ttm.ko
```

• Ключ `--showconfig` – вывод эффективных настроек модулей, например:

```
sa@astra:~$ sudo modprobe --showconfig | wc
   45581  136646 2145499
sa@astra:~$ sudo modprobe --showconfig | head
b lacklist ast
blacklist mgag200
blacklist ath_pci
blacklist ohci1394
blacklist sbp2
blacklist dv1394
blacklist raw1394
blacklist video1394
blacklist aty128fb
blacklist atyfb
sa@astra:~$ sudo modprobe --showconfig | tail
alias symbol:zgid ib_core
alias symbol:zl10036_attach zl10036
alias symbol:zl10039_attach zl10039
alias symbol:zl10353_attach zl10353
alias symbol:zpa2326_isreg_precious zpa2326
alias symbol:zpa2326_isreg_readable zpa2326
alias symbol:zpa2326_isreg_writeable zpa2326
alias symbol:zpa2326_pm_ops zpa2326
alias symbol:zpa2326_probe zpa2326
alias symbol:zpa2326_remove zpa2326
```

Автозагрузка модулей ядра

Для автозагрузки модулей ядра службой udev при старте системы их имена достаточно добавить в один из файлов с суффиксом .conf в каталоге /etc/modules-load.d/.

```
sa@astra:~$ ls -l /etc/modules-load.d/
итого 8
- rw-r--r-- 1 root root 123 мар 22  2023 cups-filters.conf
lrwxrwxrwx 1 root root  10 окт  3  2022 modules.conf -> ../modules
-rw-r--r-- 1 root root  30 мая  5  2022 usb-over-ip-load.conf
```

Пустые строки и комментарии игнорируются `/etc/modules-load.d/modules.conf`:

```
# /etc/modules: kernel modules to load at boot time.
#
# This file contains the names of kernel modules that should be loaded
# at boot time, one per line. Lines beginning with "#" are ignored.

sa@astra:~$ cat /etc/modules-load.d/cups-filters.conf
# Parallel printer driver modules loading for cups
# LOAD_LP_MODULE was not 'yes' in /etc/default/cups
lp
ppdev
parport_pc
```

Кроме этой директории существует еще директория `/usr/lib/modules-load.d/`, конфигурационные файлы которой тоже учитываются.

Модули, внесенные в эти конфигурационные файлы, будут автоматически загружены при следующем старте ОС.

Черный список модулей

Если требуется заблокировать работу устройства или избежать конфликтов между модулями, какие-то из них можно внести в так называемый черный список.

Черные списки находятся в файлах *.conf в следующих каталогах:

• Каталог `/lib/modprobe.d/` – содержит настройки, заданные при установке ОС или внесенные при установке ПО;

• Каталог `/etc/modprobe.d/` – в эту папку можно сохранять свои собственные настройки, которые потребовались вам в процессе эксплуатации системы.

```
sa@astra:~$ ls /lib/modprobe.d/
aliases.conf  blacklist_linux_5.15.0-33-generic.conf  fbdev-blacklist.conf  systemd.conf

sa@astra:~$ ls /etc/modprobe.d/
blacklist-astra.conf    blacklist.conf           blacklist-framebuffer.conf  blacklist-rare-
network.conf  parsec-cifs.conf
blacklist-ath_pci.conf  blacklist-firewire.conf  blacklist.local.conf        iwlwifi.conf

sa@astra:~$ cat /lib/modprobe.d/blacklist_linux_5.15.0-33-generic.conf | head
# Kernel supplied blacklist for linux 5.15.0-33-generic amd64
# modprobe.d/common.conf
# LP:1434842 -- disable OSS drivers by default to allow pulseaudio to emulate
blacklist snd-mixer-oss
blacklist snd-pcm-oss
# Autogenerated watchdog blacklist
blacklist acquirewdt
blacklist advantechwdt
blacklist alim1535_wdt
blacklist alim7101_wdt # Kernel supplied blacklist for linux 5.15.0-33-generic amd64
# modprobe.d/common.conf
# LP:1434842 -- disable OSS drivers by default to allow pulseaudio to emulate
blacklist snd-mixer-oss
blacklist snd-pcm-oss
# Autogenerated watchdog blacklist
blacklist acquirewdt
blacklist advantechwdt
```

```
blacklist alim1535_wdt
blacklist alim7101_wdt
```

Модули могут быть загружены как службой udev, так и на более раннем этапе из initramfs диска.

Для того чтобы запретить автозагрузку модуля, достаточно в каталоге `/etc/modprobe.d/` в файле с суффиксом `*.conf` создать строку, которая начиналась бы с директивы blacklist, за которой следовало бы имя модуля. Однако для соблюдения соглашения об именовании крайне желательно, чтобы имя такого файла начиналось со слова blacklist.

Для запрета автозагрузки модуля, загружаемого из initramfs, необходимо пересоздать initramfs после внесения изменений в конфигурационные файлы. Это можно сделать командой `sudo update-initramfs -u`.

Запрет, созданный с помощью директивы blacklist, запретит автозагрузку модуля. Однако модуль может быть все равно загружен, если от него зависят другие модули. Для полного запрета загрузки модуля в дополнение к директиве blacklist необходимо добавить команду `install <имя_модуля> /bin/true`.

Загрузку модулей можно запретить также с помощью параметров GRUB. Это может быть полезно, если один из модулей препятствует нормальной загрузке системы. Для этого необходимо добавить текст вида *module_blacklist=<имя_модуля1>,<имя_модуля2>* в строку с параметрами загрузки ядра.

Создадим файл blacklist-network.conf в /etc/modprobe.d/, добавим в него строку blacklist e1000 и перезагрузим систему.

```
sa@astra:~$ sudo nano /etc/modprobe.d/blacklist-network.conf
sa@astra:~$ sudo cat /etc/modprobe.d/blacklist-network.conf
blacklist e1000
sa@astra:~$ sudo reboot
```

Казалось бы, при перезагрузке системы драйвер сетевого адаптера не будет загружен. Выполним перезагрузку и убедимся в том, что драйвер по-прежнему загружается.

```
sa@astra:~$ lsmod | grep e1000
e1000                 151552  0
```

Дело в том, что этот драйвер загружается из initramfs. Выполним обновление образа Initrd, перезагрузим систему и убедимся, что теперь сеть нам не доступна:

```
sa@astra:~$ sudo update-initramfs -u
update-initramfs: Generating /boot/initrd.img-5.15.0-33-generic
sa@astra:~$ sudo reboot
```

После перезагрузки:

```
sa@astra:~$ lsmod | grep e1000
sa@astra:~$
```

Однако мы все еще можем загрузить этот модуль вручную командой:

```
sa@astra:~$ sudo modprobe e1000
sa@astra:~$ lsmod | grep e1000
e1000                 151552
```

При внесении дополнительной строки *install e1000 /bin/true* в конфигурационный файл `/etc/modprobe.d/blacklist-network.conf` мы полностью запретим загрузку этого модуля.

```
blacklist e1000
install e1000 /bin/true
```

Примечание

Строка *install e1000 /bin/true* указывает, что вместо загрузки модуля *e1000* надо выполнить команду */bin/true*, которая всегда завершает своё выполнение с кодом успеха (0). Таким образом, при попытке загрузить модуль *e1000* не возникает никаких ошибок — система считает, что модуль загружен успешно, хотя на самом деле он загружен не был.

Создайте файл `blacklist-network.conf` с содержимым, описанным выше, и перезагрузитесь:

```
sa@astra:~$ sudo nano /etc/modprobe.d/blacklist-network.conf
sa@astra:~$ sudo reboot
```

После перезагрузки проверим:

```
sa@astra:~$ sudo modprobe e1000
sa@astra:~$ lsmod | grep e1000
sa@astra:~$
```

Для восстановления работоспособности сети, удалим созданный нами файл черного списка `/etc/modprobe.d/blacklist-network.conf`, пересоздадим образ Initrd и перезагрузим систему.

```
sa@astra:~$ sudo rm /etc/modprobe.d/blacklist-network.conf
sa@astra:~$ sudo update-initramfs -u
update-initramfs: Generating /boot/initrd.img-5.15.0-33-generic
sa@astra:~$ sudo reboot
```

Как мы можем убедиться, сетевой адаптер снова работает, и сеть стала опять доступна:

```
sa@astra:~$ lsmod | grep e1000
e1000                 151552 0
```

Псевдонимы модулей

Псевдонимы модулей – это альтернативные имена модулей, которые могут быть созданы для удобства (если имя модуля, например, слишком длинное) или для нужд приложений.

Псевдонимы, как и черные списки, хранятся в файлах с суффиксом .conf в следующих каталогах:

• Каталог `/lib/modprobe.d/` - содержит настройки, заданные при установке ОС или внесенные при установке ПО.
• Каталог `/etc/modprobe.d/` - в эту папку можно сохранять свои собственные настройки, которые потребовались вам в процессе эксплуатации системы.
Однако в этом случае не нужно использовать слово blacklist в названии файлов.

Чтобы создать псевдоним, необходимо в файл с суффиксом `*.conf` (например, `aliases.conf` или `aliases-<что-то>.conf`) добавить строку вида:

```
alias <псевдоним_модуля> <имя_модуля>
```

В именах алиасов вам доступны также символы подстановки. Например, задав псевдоним через команду `alias my-mod* module_name`, к модулю можно будет обращаться через имена my-mod-1, my-mod-something и так далее.

Параметры модулей

Параметры модулей можно задавать как с помощью конфигурационных файлов, так и «на лету» с помощью утилиты `modprobe`:

• Параметры модулей, как и псевдонимы, задаются в файлах `*.conf` в каталогах `/lib/modprobe.d/` и `/etc/modprobe.d/`. Для этого необходимо добавить в файл строку вида:

```
options <имя_модуля> <имя_параметра>=<значение параметра>
```

• Для управления параметрами модулей при работе системы можно воспользоваться командой:

`modprobe <имя_модуля> <имя_параметра>=<значение параметра>`.

Практические задания

Задание 1.

1. По умолчанию группой файла /dev/tty8 является группа tty. Поменяйте группу на группу root для /dev/tty8.
2. После того как группа будет изменена, отмените свои изменения.

Задание 2.

1. Определите путь к сетевому устройству eth0 в каталоге /sys.
2. Выведите все атрибуты устройства eth0.

Задание 3.

1. Выгрузите модуль intel_rapl_msr.
2. Добавьте модуль intel_rapl_msr в черный список. Перезагрузите систему и убедитесь, что он не загружен.
3. Отмените свои изменения.

Вопросы

1. Какая директория отражает информацию об устройствах в ядре?

2. Есть два файла с одним названием в /etc/udev/rules.d и /usr/lib/udev/rules.d/. Какой из них будет иметь больший приоритет?

3. Какой командой можно посмотреть все PCI устройства?

4. Какая команда служит для просмотра только USB устройств?

5. Что делает команда udevadm info -e?

6. Какое расширение имеют файлы модулей ядра?

7. В какой директории находятся файлы модулей ядра?

8. В каком файле modpobe находит список всех встроенных модулей ядра?

9. Какой командой можно посмотреть список загруженных модулей ядра?

10. Что делает команда modprobe -r e1000?