

# Модули ядра и настройки ядра Linux

# Ядра ОС и Linux

**Ядро** — центральная часть ОС, обеспечивающая приложениям координированный доступ к ресурсам компьютера, таким как процессорное время, память, внешнее аппаратное обеспечение, внешнее устройство ввода и вывода информации.

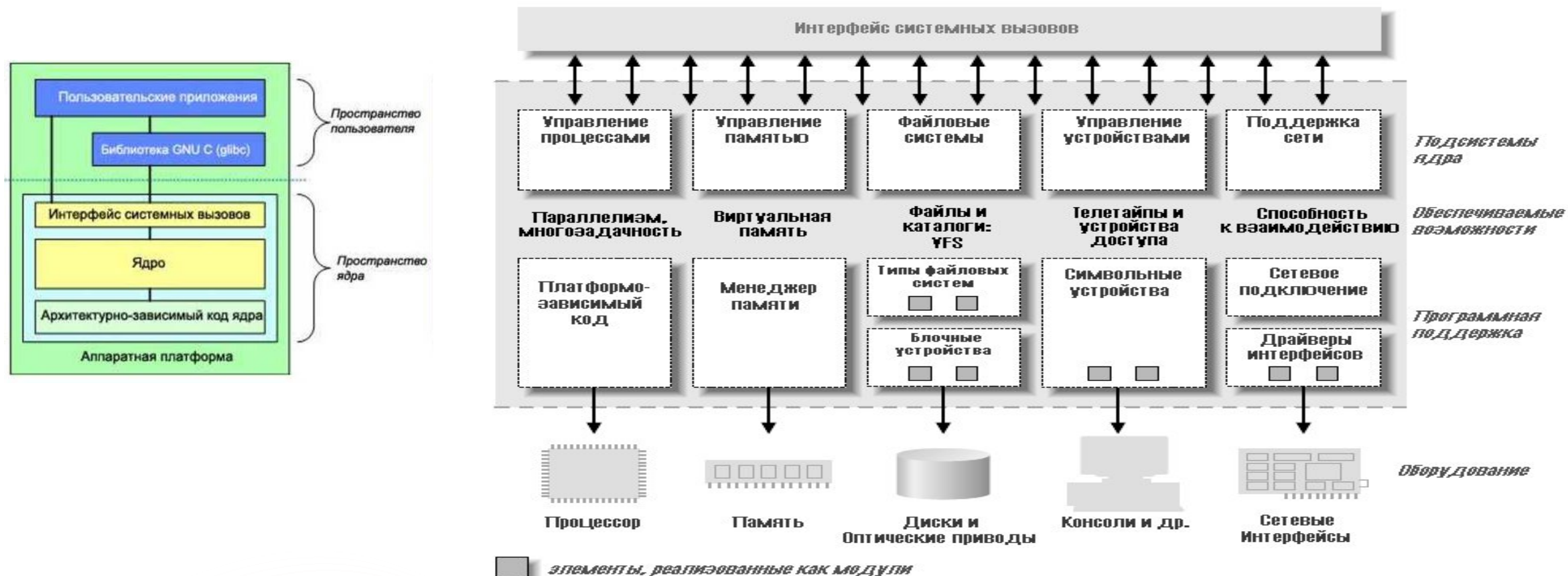
**Ядро Linux** — монолитное, с поддержкой загружаемых модулей.

Загружаемый модуль ядра (loadable kernel module, LKM) — объектный файл, содержащий код, расширяющий возможности ядра операционной системы.

Может быть подключен к работающему ядру. Обычно так реализованы драйверы устройств.

# Архитектура ядра

Устройства (или некоторые системы) для своей работы требуют загруженных в память модулей ядра Linux (в нотации Windows — драйверов).



# Ядра ОС и Linux

Находятся все модули в папке **/lib/modules/**. Учитывая, что модули рассчитаны только для **определенной версии ядра**, то в этой папке создается отдельная подпапка, для каждой установленной в системе версии ядра. В этой папке находятся сами модули и дополнительные конфигурационные файлы, модули **отсортированы по категориям**, в зависимости от назначения основные команды для управления модулями

**lsmod** — посмотреть загруженные модули

**modinfo** — информация о модуле

**insmod** — загрузить модуль

**rmmod** — удалить модуль

Работа с модулями ядра Linux выполняется, в основном, с помощью этих команд, но могут использовать и другие



# Ядра ОС и Linux

Все модули записаны в конфигурационном файле **/lib/modules/modules.aliases**, поэтому мы можем посмотреть его содержимое. Это все известные модули:

```
find /lib/modules/$(uname -r) -name *.ko
```

или

```
modprobe -c
```

Все информация о загруженных модулях хранится в файле **/proc/modules**:

```
cat /proc/modules
```

Более цивилизованный способ утилита **lsmod** и **modinfo**. Чтобы посмотреть загруженные модули ядра linux выполните:

```
sudo lsmod
```

Загружен ли модуль

```
sudo lsmod | grep vbox
```

Или более подробно

```
modinfo fuse
```

# Ядра ОС и Linux

Загрузить модуль ядра Linux можно с помощью команд `modprobe` или `insmod`:

**`modprobe pcspkr`**

или

**`insmod /lib/modules/6.1.44-1.el7.3.x86_64/kernel/drivers/input/misc/pcspkr.ko`**

Первый способ проще и предпочтительнее, т. к. **`modprobe` автоматически загружает зависимости.**

# Ядра ОС и Linux

Выгрузить модуль ядра Linux можно с помощью команд **modprobe** или **rmmod**:

**modprobe -r vboxdrv**

или

**rmmod vboxdrv**

Выгрузить можно только неиспользуемый модуль.

Блокировать загрузку модулей можно через файл **/etc/modprobe.d/blacklist.conf**:

**blacklist b43**

Автоматическую загрузку модулей можно организовать с помощью файлов в каталоге **/etc/modules-load.d/**

# Ядра ОС и Linux

Для отображения состояния процессов и устройств существуют виртуальные папки в корневом каталоге Linux:

**/dev** файловая система для устройств

**/proc** кроме информации о каждом запущенном процессе, добавлена всевозможную информацию о состоянии работающего ядра.

**/sys** информация, не относящаяся к процессу, которая попала в дерево /proc, дублируется в /sys.

# Ядра ОС и Linux

Виртуальная ФС **/proc** также позволяет просматривать и временно менять параметры ядра от пользователя **root**. Многие параметры ядра могут быть изменены «на лету» либо в файлах, либо утилитой **sysctl**.

**cat /proc/sys/vm/swappiness** или **sysctl vm.swappiness**

Изменение параметра до перезагрузки:

**sysctl vm.swappiness=50** или **echo 50 >/proc/sys/vm/swappiness**

Конфигурационный файл - **/etc/sysctl.conf** и файлы в каталоге **/etc/sysctl.d**. Чтобы изменить параметр на постоянной основе, нужно прописать его в файл **/etc/sysctl.conf** или в каталоге **/etc/sysctl.d/** создать любой файл с расширением **.conf** и прописать его туда. И применить настройки оттуда: **sysctl -p**

Просмотр всех параметров ядра: **sysctl -a**

# Некоторые параметры ядра

**net.ipv4.ip\_forward** — включает возможность проброса пакетов

**net.ipv4.tcp\_keepalive\_time** — таймаут после которого неактивное соединение будет считаться разорванным

**net.ipv4.icmp\_echo\_ignore\_all** — ядро игнорирует все ICMP-запросы (используются командой ping)

**fs.file-max** — максимальное количество открытых файлов

**kernel.pid\_max** — максимальное значение PID (идентификатор процесса)

**vm.max\_map\_count** — максимальное количество отображений в памяти

**kernel.threads-max** — максимальное количество параллельных потоков

**vm.dirty\_background\_ratio** — процент от оперативной памяти, при заполнении которой страничным кэшем начнется его сброс на диск

**vm.dirty\_ratio** — максимальный объем системной памяти, которую можно заполнить страничным кэшем

**vm.swappiness** — процент свободной памяти, по достижении которого данные начинают переноситься на swar раздел

**kernel.panic** — таймаут в секундах после ошибки в ядре (kernel panic) до перезагрузки

# Каталог /dev

Этот каталог содержит файловую систему, структура которой инициализируются при загрузке системы сервисом **systemd-udevd**.

**udev** — это менеджер устройств для ядра Linux. Его **основная задача** — **обслуживание файлов устройств** и обработка действий, выполняемых в пространстве пользователя при добавлении или отключении внешних устройств.

Некоторые интересные устройства:

**/dev/null** — представляет собой что-то типа «мусорной корзины». Чтение /dev/null возвращает символы end-of-file — конец файла

**/dev/zero** — чтение устройства возвращает \0 символы. Поэтому /dev/zero обычно используется для создания пустых файлов.

**/dev/random** и **/dev/urandom** — создают «случайные» данные.



# Основные особенности udev

**udev** поддерживает **неизменное именование** устройств, не зависящее от порядка включения устройств в систему

**udev** работает целиком в **пространстве пользователя**

состоит из 3х частей:

1.Библиотека **libudev**, позволяющая получать доступ к информации об устройствах.

2.Демон **systemd-udevd**, управляющий в пространстве пользователя содержимым /dev.

3.Программа **udevadm**, используемая для отладки и диагностики.

Главный файл конфигурации для udev — это **/etc/udev/udev.conf**, а для управления поведением демона udev во время выполнения вы можете использовать утилиту **udevadm**.

# Демон udev, systemd-udevd.service

Демон **systemd-udevd.service** взаимодействует с ядром и получает события устройства непосредственно от него каждый раз, когда вы добавляете или удаляете устройство из системы, или когда устройство меняет своё состояние.

Udev основан на правилах — **правила гибкие и очень мощные.**

Событие устройства сопоставляется с набором правил хранимых в папках:

**/etc/udev/rules.d**  
**/usr/lib/udev/rules.d**  
**/run/udev/rules.d.**

Когда подключается устройство, ядро обнаруживает и инициализирует его, и в каталоге **/sys/** создаётся каталог с атрибутами устройства.

# Утилита **udevadm monitor**

**udevadm** - инструмент управления udev

**info** — отображение информации об устройстве

**trigger** — используется для воспроизведения событий

**settle** — отслеживает очередь событий udev

**control** — изменение состояние демона

**monitor** — события ядра и события udev

**test** — моделирование поведения правил

**test-builtin** — запустить встроенную команду

**wait** — ожидание реакции устройства

**lock** — тестовая блокировка устройств

# Утилита `udevadm monitor`

Чтобы отобразить полученные события ядра и события `udev` (которые `udev` отправляет после обработки правила), запустите `udevadm` с командой `monitor`. Затем подключите устройство к вашей системе и наблюдайте с терминала, как обрабатывается событие устройства.

**UDEV** — то что `udev` отправляет после обработки правила

**KERNEL** — событие ядра

# Утилита `udevadm test`

**`udevadm test <путь к имени устройства>`**

Служит для проверки правил написанных для устройства. Например:

**`udevadm test /sys/block/sda`**

Если же в правиле допустить синтаксическую ошибку, например, UBSYSTEM вместо SUBSYSTEM, `udevadm test` выдаст что то подобное:

```
unknown key 'UBSYSTEM' in /etc/udev/rules.d/10-local.rules:2  
invalid rule '/etc/udev/rules.d/10-local.rules:2'
```

Здесь мы видим саму причину ошибки, неверный ключ, а также файл и строку в которой допущена ошибка.

# Утилита **udevadm test**

**udevadm info <путь к имени устройства>**

Посмотреть все возможные Udev параметры для устройства можно с помощью запроса **info**. Например, для диска **/dev/sda**

Опция **-n** задает имя устройства, **-p** путь в **sysfs**. Например, то же самое получим если выполнить:

```
udevadm info /dev/sda1  
udevadm info -a -n sda1  
udevadm info -a -p /sys/block/sda/sda1
```

Вывести весь список устройств **udevadm info --export-db** можно использовать для анализа

# Утилита **udevadm info**

**udevadm info <путь к имени устройства>**

Посмотреть все возможные Udev параметры для устройства можно с помощью запроса info. Например, для диска /dev/sda

Опция -n задает имя устройства, -p путь в sysfs. Например, то же самое получим если выполнить:

**udevadm info /dev/sda1**

**udevadm info -a -n sda1**

**udevadm info -a -p /sys/block/sda/sda1**

Вывести весь список устройств **udevadm info --export-db** можно использовать для анализа



# Пользовательские правила udev

Пользовательские правила работы с устройствами находятся в папке **/etc/udev/rules.d**

Правила udev могут:

- переименовать устройство;
- создать дополнительное имя для устройства;
- установить владельца и группу;
- поменять права доступа к устройству;
- выполнить устройства.

# Пользовательские правила udev

Файл правил обязательно должен иметь расширение .rules

Правило udev состоит из нескольких пар **ключ — значение**, разделённых запятой.

Для проверки соответствия ==

Для указания действия =

Получаем информацию о сетевой карте с помощью udevadm:

```
udevadm info -a -p /sys/class/net/enp4s0
```

```
[sa@localhost rules.d]$ cat /etc/udev/rules.d/usb.rules
SUBSYSTEM=="usb", ACTION=="add", ENV{DEVTYPE}=="usb_device", RUN+="/bin/device_added
.sh"
SUBSYSTEM=="usb", ACTION=="remove", ENV{DEVTYPE}=="usb_device", RUN+="/bin/device_re
moved.sh"
```

# Пользовательские правила udev

Ключи соответствия:

**SUBSYSTEM** - подсистема устройства;

**KERNEL** - имя, выдаваемое устройству ядром;

**DRIVER** - драйвер, обслуживающий устройство;

**ATTR** - sysfs-атрибут устройства;

**SUBSYSTEMS** - подсистема родительского устройства.

Устройство может иметь родительские устройства, например, жёсткий диск <<< SCSI <<< шина BUS. Для получения информации от родительского устройства, используются ключи SUBSYSTEMS, KERNELS, DRIVERS, ATTRS.

# Пользовательские правила udev

Ключи действия:

**NAME** - установить имя файла устройства;

**SYMLINK** - альтернативное имя устройства;

**RUN** - выполнить скрипт при подключении устройства;

**GROUP** - группа, у которой есть доступ к файлу;

**OWNER** - владелец файла устройства;

**MODE** - маска прав доступа.

# Пользовательские правила udev

Ключ ATTR. Он позволяет получить информацию об устройстве. Посмотреть все возможные sysfs-параметры для устройства можно непосредственно в файловой системе /sys. Например, для диска /dev/sda есть каталог /sys/block/sda/, в котором можно найти файлы size, stat, ro, range и т.д.

Также посмотреть все возможные udev - параметры для устройства можно с помощью утилиты udevadm. Например, для диска /dev/sda:

```
udevadm info -a -n sda1
```

# Пользовательские правила udev

## ЗАПУСКАЕМ СКРИПТ

Хотим автоматически запускать скрипт при монтировании флешки? Если флешка будет называться /dev/sdb, тогда можно создать правило udev такого вида:

**KERNEL=="sdb", RUN+=" /usr/bin/my\_script"**

При подключении флешки выполнится скрипт /usr/bin/my\_script и сделает необходимые действия. Скрипт не должен выполняться слишком долго, так как udev остановится и будет ожидать завершения его работы.

# Пользовательские правила udev

КОНТРОЛЬ ПОДКЛЮЧЕНИЯ USB

Создадим два скрипта:

device\_added.sh

```
1 #!/bin/bash
2
3 echo "USB device added at $(date)" >>/tmp/scripts.log
```

device\_removed.sh

```
1 #!/bin/bash
2
3 echo "USB device removed at $(date)" >>/tmp/scripts.log
```

Делаем оба сценария исполняемыми. Затем создаём правило для запуска выполнения вышеуказанных сценариев.

```
SUBSYSTEM=="usb", ACTION=="add", ENV{DEVTYPE}=="usb_device",
    RUN+="/bin/device_added.sh"
```

```
SUBSYSTEM=="usb", ACTION=="remove", ENV{DEVTYPE}=="usb_device",
    RUN+="/bin/device_removed.sh"
```

Если правила не перезагружаются автоматически, выполните:

**udevadm control --reload-rules**



# Команды работы с оборудованием

lspci -vv <опции> — информация о pci устройствах

lsusb <опции> — Подключенные устройства USB

lsusb.py <опции> — Подключенные устройства USB

usb-devices — Подробная информация об USB устройствах

lsscsi <опции> — Смотрим подключенные жесткие диски

lscpu <опции> — Информация о процессоре

lsmem — Для просмотра информации о используемой памяти

lshw <опции> — Утилита выводит подробную информацию по устройствам

dmidecode – Разнообразная информация о железе

inxi — Скрипт, позволяющий получить информации о системе

hwinfo — Утилита выводит информацию об устройствах

HardInfo — Графическая утилита

# Команды работы с оборудованием


**hw-probe** — утилита от создателей сайта <https://linux-hardware.org/>, позволяет одной лаконичной командой собрать в одном файле все основные сведения о системе.

Использование: **sudo hw-probe [options]**

Пример: **sudo hw-probe -all -upload -id DESC**

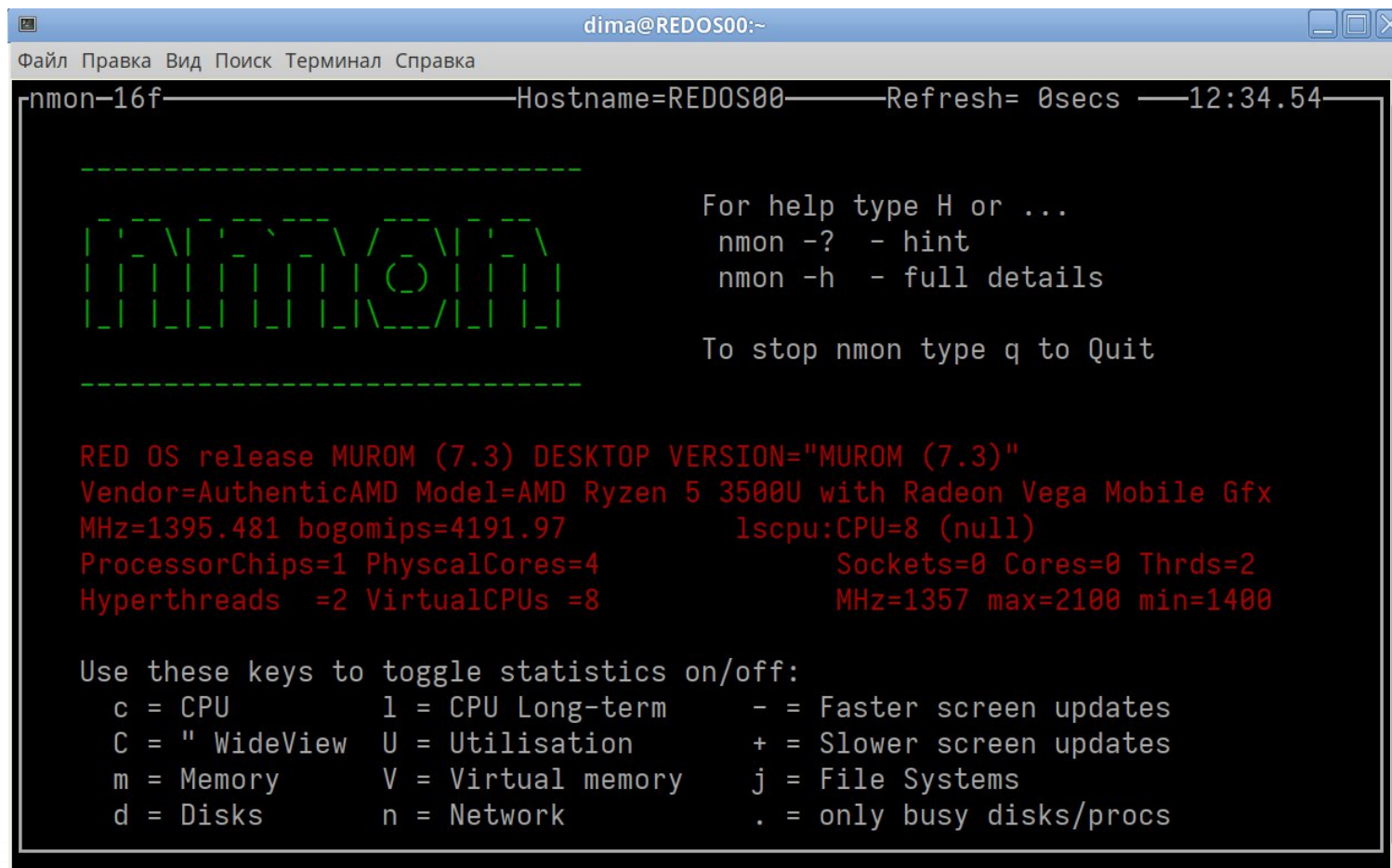
Выполняем и переходим по ссылке

## Host

System	 Red OS 7.3.1
Arch	x86_64
Kernel	5.15.10-1.el7.x86_64
Vendor	Lenovo »
Model	IdeaPad L340-15API 81LW »
Year	2019
HWid	8FB57 »
Type	notebook
DE	MATE (X11) - SDDM

# Команды работы с оборудованием

nmon – информация о чтении / записи процессами данных на диск.



```
dima@REDOS00:~
Файл Правка Вид Поиск Терминал Справка
nmon-16f-----Hostname=REDOS00-----Refresh= 0secs -----12:34.54-----

-----
  RED OS
-----

For help type H or ...
nmon -? - hint
nmon -h - full details

To stop nmon type q to Quit

RED OS release MUROM (7.3) DESKTOP VERSION="MUROM (7.3)"
Vendor=AuthenticAMD Model=AMD Ryzen 5 3500U with Radeon Vega Mobile Gfx
MHz=1395.481 bogomips=4191.97          lscpu:CPU=8 (null)
ProcessorChips=1 PhyscalCores=4        Sockets=0 Cores=0 Thrds=2
Hyperthreads  =2 VirtualCPUs =8       MHz=1357 max=2100 min=1400

Use these keys to toggle statistics on/off:
c = CPU          l = CPU Long-term      - = Faster screen updates
C = " WideView  U = Utilisation      + = Slower screen updates
m = Memory       V = Virtual memory   j = File Systems
d = Disks        n = Network          . = only busy disks/procs
```



# Команды работы с оборудованием

iostat – информация о чтении / записи процессами данных на диск

dima@REDOS00:~									
Файл Правка Вид Поиск Терминал Справка									
Total DISK READ :		0.00 B/s		Total DISK WRITE :		30.21 K/s			
Actual DISK READ:		0.00 B/s		Actual DISK WRITE:		0.00 B/s			
TID	PRI	USER	DISK READ	DISK WRITE	SWAPIN	IO>	COMMAND		
3140	be/4	dima	0.00 B/s	11.33 K/s	0.00 %	0.00 %	chromium-browser ~to-ssl-client-auth		
3159	be/4	dima	0.00 B/s	11.33 K/s	0.00 %	0.00 %	chromium-browser ~ [Chrome_IOThread]		
9343	be/4	dima	0.00 B/s	7.55 K/s	0.00 %	0.00 %	chromium-browser ~ [ThreadPoolForeg]		
1	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	systemd --switche~m --deserialize 30		
2	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[kthreadd]		
3	be/0	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[rcu_gp]		
4	be/0	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[rcu_par_gp]		
6	be/0	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[kworker/0:0H-events_highpri]		
2056	be/4	dima	0.00 B/s	0.00 B/s	0.00 %	0.00 %	goa-identity-service [gdbus]		
9	be/0	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[mm_percpu_wq]		
10	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[rcu_tasks_kthre]		

# Практическая работа

1. Подключить к компьютеру usb flash
2. Командой `udevadm monitor` определить какое устройство назначено диску и в какой каталог он подключен. Отмонтировать диск
3. Утилитой `lsmod` получить список модулей ядра. Вывод записать в файл `mod.log`.
4. Утилитой `modinfo` получить информацию по модулю `pcspkr`. Вывод в файл `mod.log`.
5. Выгрузить модуль `pcspkr`.
6. Вывести все параметры ядра командой `sysctl`. Вывод перенаправить в файл `sysctl.log`.
7. Командой `sysctl` вывести значение параметра `net.ipv4.ip_forward`.
8. Изменить значение этого параметра на противоположное.
9. Командой `sysctl` вывести значение параметра `net.ipv4.ip_forward`. Вывод дописать в файл `sysctl.log`.
10. Изменить его на постоянной основе, прописав в файле в каталоге `/etc/sysctl.d`



**Спасибо за внимание!**

**[www.red-soft.ru](http://www.red-soft.ru)**  
**[redos@red-soft.ru](mailto:redos@red-soft.ru)**

