

# Операционная система Linux: Загрузка ОС



Александр  
Гришин



**Александр Гришин**

Инженер, компания YADRO

---

# Предисловие

На этом занятии мы рассмотрим:

- процесс загрузки ОС Linux;
- основные функции BIOS/UEFI;
- работу загрузчика GRUB;
- начнем изучение процесса инициализации `init`.

---

# План занятия

1. [Предисловие](#)
2. [Загрузка ОС](#)
3. [BIOS](#)
4. [UEFI](#)
5. [GRUB](#)
6. [init](#)
7. [Итоги](#)
8. [Домашнее задание](#)



# Загрузка ОС

# Загрузка, boot

**Загрузка** (boot, booting) — процесс запуска устройства и загрузки ОС. В этот момент происходит обнаружение и (само)настройка всех КОМПОНЕНТОВ СИСТЕМЫ.

```
[ OK ] Reached target Local File Systems (Pre).  
        Mounting Temporary /etc/pacman.d/gnupg directory...  
[ OK ] Started Entropy Daemon based on the HAVEGE algorithm.  
        Starting Journal Service...  
        Starting udev Kernel Device Manager...  
[ OK ] Mounted Temporary /etc/pacman.d/gnupg directory.  
[ OK ] Reached target Local File Systems.  
        Starting Rebuild Dynamic Linker Cache...  
[ OK ] Finished Load Kernel Modules.  
        Mounting Kernel Configuration File System...  
        Starting Apply Kernel Variables...  
[ OK ] Mounted Kernel Configuration File System.  
[ OK ] Finished Apply Kernel Variables.  
[ OK ] Finished Load/Save Random Seed.  
[ OK ] Started Journal Service.  
        Starting Flush Journal to Persistent Storage...  
[ OK ] Finished Flush Journal to Persistent Storage.  
        Starting Create Volatile Files and Directories...  
[ OK ] Started udev Kernel Device Manager.  
[ OK ] Finished Create Volatile Files and Directories.  
        Starting Rebuild Journal Catalog...  
        Starting Update UTMP about System Boot/Shutdown...  
[ OK ] Finished Update UTMP about System Boot/Shutdown.  
[ OK ] Finished Rebuild Journal Catalog.
```

---

# Этапы загрузки

1. Загрузка BIOS/UEFI (MBR/GPT).
2. Поиск, загрузка в память и запуск загрузчика (GRUB).
3. Поиск, загрузка в память и запуск ядра ОС.
4. Запуск системных служб.
5. Запуск пользовательских служб.



**BIOS**



# BIOS

**BIOS** (Basic Input/Output System) — набор системных программ (firmware), используемых для процесса проверки и инициализации аппаратного обеспечения с последующим запуском загрузчика ОС.



---

# История BIOS

Изначально BIOS была собственностью IBM и применялась только IBM PC. После реверс-инжиниринга стала применяться во всех PC-совместимых компьютерах.

Ранее, помимо загрузки, BIOS использовалась для обработки прерываний от устройств ввода-вывода. Сейчас данный механизм полностью реализован на уровне операционной системы.

Также, в первых компьютерах BIOS была аппаратно реализована в виде ROM (ПЗУ), в последних — это flash-память с возможностью перезаписи.

# POST

**POST** (power-on self-test) — процесс первоначальной проверки оборудования сразу после его включения.

- Если во время POST обнаруживаются ошибки, то они выдаются пользователю как набор звуковых сигналов и номер кода ошибки, доступный через специальные отладочные платы.
- POST запускается после включения компьютера или нажатия кнопки «Reset».
- Если перезагрузка происходит по нажатию *Alt-Ctrl-Del*, то процедура POST не запускается.

---

# Процесс загрузки BIOS

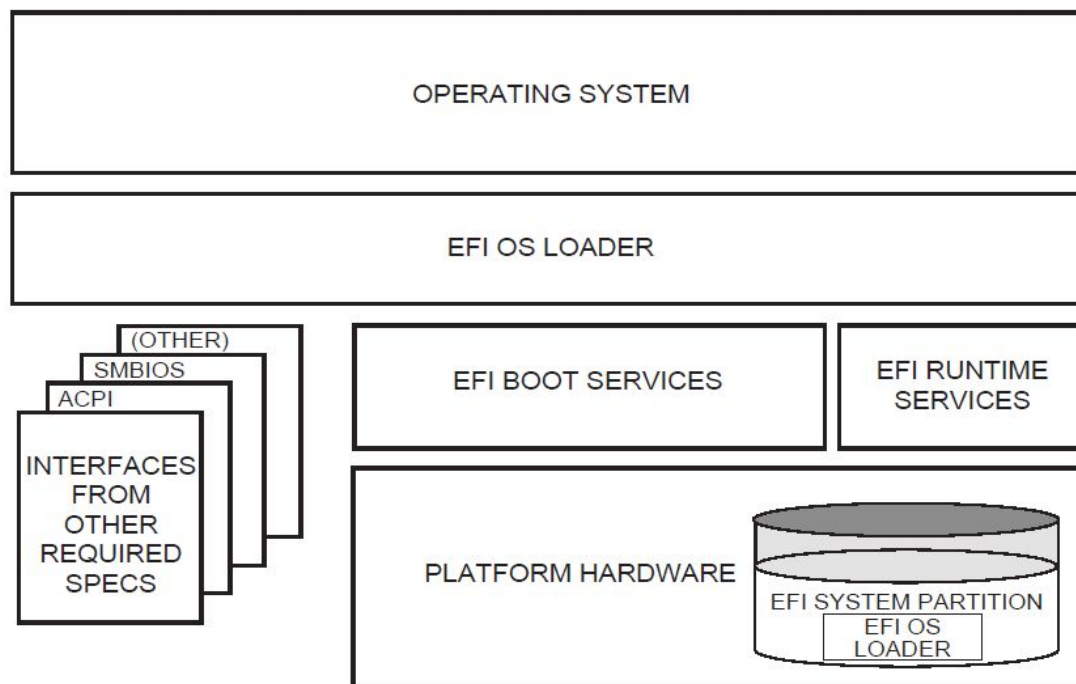
1. Инициализация видеокарты (как option BIOS);
  2. Запуск и выполнение POST;
  3. Поиск загрузчика ОС на доступных устройствах (проверка сигнатуры загрузочного диска);
  4. Передача управления на найденный загрузочный сектор.
- ➡ В случае обнаружения дополнительных (option) микросхем BIOS (например, на сетевой карте, материнской плате, видео-карте), управление загрузкой может передаваться на данное устройство.



**UEFI**

# UEFI

**UEFI, Unified Extensible Firmware Interface** (интерфейс расширяемой прошивки) — спецификация интерфейса между ОС и аппаратными прошивками (firmware).



---

# Преимущества UEFI

- возможность использовать разделы больше 2 ТБ GUID Partition Table (GPT);
- работа напрямую с ФС;
- работа с сетью;
- удобное и функциональное графическое окружение;
- 32- или 64-битное окружение;
- разработка на языке C;
- модульная архитектура;
- большие возможности по совместимости и модернизации.

---

## Недостатки UEFI

- бóльшая возможность для внедрения вредоносных программ;
- проблемы с гибкостью, т.к. каждая ОС должна иметь свой собственный драйвер в составе UEFI;
- плохая поддержка старых ОС;
- усложнение архитектуры усложняет разработку;
- возможна ситуация, когда устанавливается только определенная ОС (например, Windows).



---

## Фазы UEFI

1. **SEC** (Security) — проверяет цифровые подписи и передает управление доверенному коду.
2. **PEI** (Pre EFI Initialization) — инициализация устройств.
3. **DXE** (Driver eXecution Environment) — загрузка сервисов UEFI.
4. **BDS** (Boot Device Select) — поиск устройств загрузки.
5. **RT** (Run Time) — GRUB.

---

# UEFI Boot Manager

UEFI, в отличие от BIOS, содержит свой **собственный загрузчик** — **UEFI Boot Manager**.

*Boot Manager осуществляет загрузку:*

- UEFI-загрузчиков ОС;
- UEFI-драйверов;
- UEFI-приложений.

Boot Manager имеет свою собственную конфигурацию, записанную в NVRAM (Non-volatile RAM).



**GRUB**

---

# GRUB

**GRand Unified Boot** (GRUB) — стандарт де-факто для загрузчиков Linux.

*Существует две версии GRUB:*

- GRUB Legacy;
- GRUB 2 (с 2012 года).

*Основное назначение:*

загрузка ядра ОС и выбор параметров загрузки.

---

# Настройка GRUB

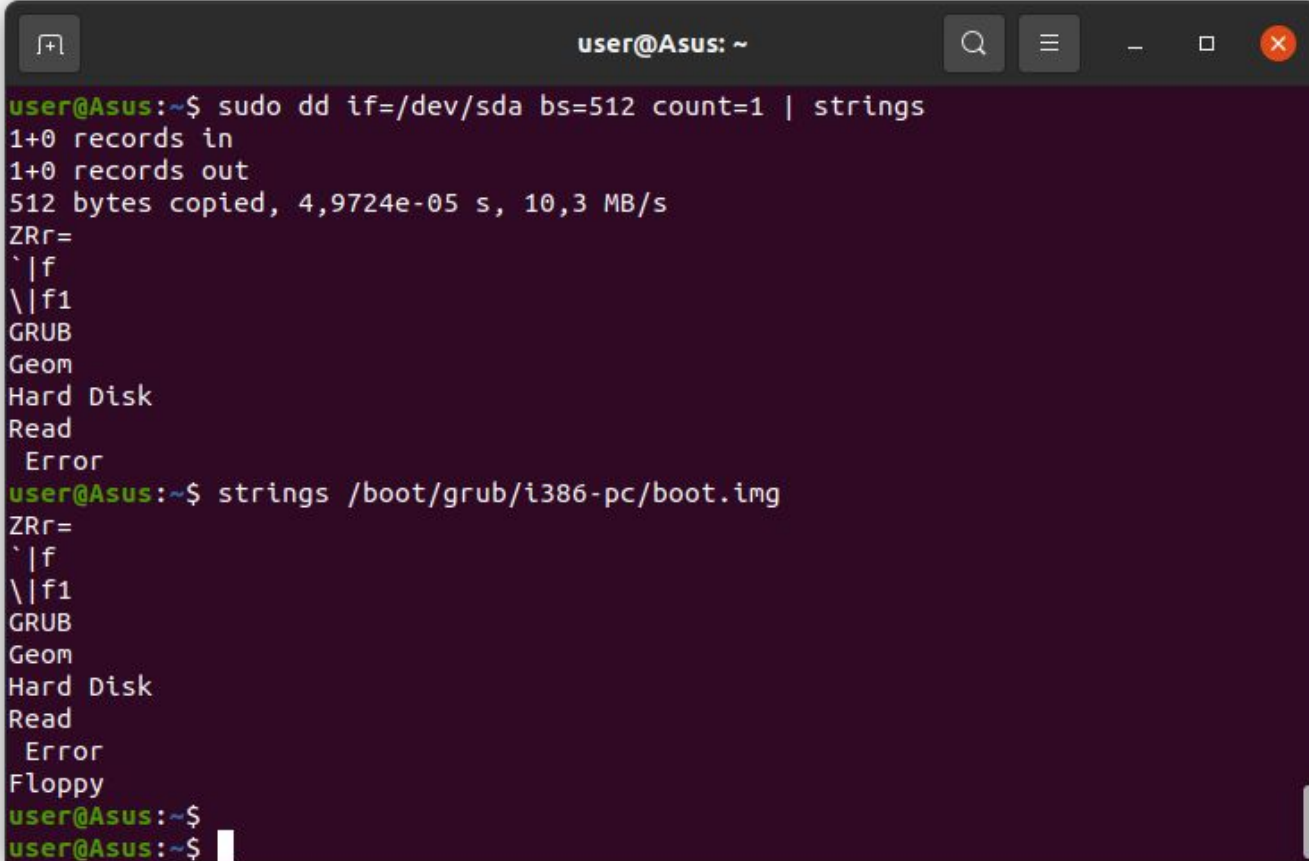
*grub.cfg:*

- /boot/grub/ — Debian, Ubuntu;
- /boot/grub2/ — Red Hat, CentOS.

*Внесение изменений в конфигурацию GRUB:*

- update-grub/grub2-mkconfig
- /etc/default/grub

# Просмотр загрузчика на диске



```
user@Asus: ~  
user@Asus:~$ sudo dd if=/dev/sda bs=512 count=1 | strings  
1+0 records in  
1+0 records out  
512 bytes copied, 4,9724e-05 s, 10,3 MB/s  
ZRR=  
`|f  
\\|f1  
GRUB  
Geom  
Hard Disk  
Read  
Error  
user@Asus:~$ strings /boot/grub/i386-pc/boot.img  
ZRR=  
`|f  
\\|f1  
GRUB  
Geom  
Hard Disk  
Read  
Error  
Floppy  
user@Asus:~$  
user@Asus:~$
```

# Редактирование GRUB

Отредактируем `/etc/default/grub`

user@user:~\$ `sudo nano /etc/default/grub`

```
# GRUB_BACKGROUND=/home/user/Pictures/cat_for_grub.jpg
GRUB_DEFAULT=0
GRUB_TIMEOUT_STYLE=hidden
# GRUB_TIMEOUT=10
GRUB_DISTRIBUTOR=`lsb_release -i -s 2> /dev/null || echo Debian`
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash"
GRUB_CMDLINE_LINUX=""
```

user@user:~\$ `sudo update-grub`

user@user:~\$ `shutdown -r now`

# Командная строка GRUB

Выполнять команды grub возможно интерактивно, во время загрузки.

Для этого необходимо нажать «C» в меню GRUB\*.

```
GNU GRUB version 2.02

Minimal BASH-like line editing is supported. For the first word,
TAB lists possible command completions. Anywhere else TAB lists
possible device or file completions. ESC at any time exits.

grub> reboot_
```

\*для виртуальной машины может потребоваться при загрузке нужно нажать «Shift«» и держать, пока не появится меню GRUB



---

## Восстановления пароля ОС

1. Выбираем расширенные параметры загрузки.
2. Выбираем `recovery mode`.
3. Выбираем `root`.
4. Вводим: `mount -o remount,rw /`
5. Вводим: `passwd user`
6. Вводим: `exit` или `shutdown -r now`

# Параметры ядра

```
user@user:~$ cat /proc/cmdline
```

```
BOOT_IMAGE=/boot/vmlinuz-5.4.0-58-generic  
root=UUID=d7c016dd-ca7d-457a-8cad-46caff118014 ro quiet splash  
vt.handoff=7
```

## *Параметры:*

- **BOOT\_IMAGE** — образ ядра, которое загружено в данный момент;
- **root** — корневая ФС;
- **vt.handoff** — параметр специфичный для Ubuntu, нужен для сокрытия вывода загрузчика.

---

# Параметры ядра

- *Просмотр параметров ядра:*

user@user:~\$ `sysctl -a`

- *Просмотр конфигурационного файла:*

user@user:~\$ `cat /etc/sysctl.conf | more`

- *Просмотр сетевых настроек IPv4:*

user@user:~\$ `cat /etc/sysctl.conf | grep ipv4`

- *Включение пересылки пакетов (функций шлюза):*

user@user:~\$ `sudo sysctl net.ipv4.ip_forward=1`

# Initrd

**Initrd** (Initial RAM Disk) — образ, содержащий модули для инициализации ядра.

➡ GRUB передает ядру данный образ как параметр загрузки, поэтому, следует обратить внимание на то, что файл *initrd* должен соответствовать загружаемому ядру.

➡ Если ядро содержит все необходимые модули, то файл **initrd не нужен**.

---

# Создание initrd

```
user@user:~$ sudo update-initramfs -c -k 5.4.0-56-generic
```

*Параметры:*

- **-c** создать новый initramfs;
- **-k** версия ядра (может быть «all»);
- **-b** указать путь, отличный от boot;
- **-u** обновить initramfs;
- **-d** удалить initramfs.

*Базовая конфигурация:*

`/etc/initramfs-tools/initramfs.conf`



**init**

---

# init

**init** — специальный процесс (демон) управления системой и службами.

*Расположение:*

`/sbin/init`

*Режимы работы init:*

- однопользовательский (службы не запускаются);
- многопользовательский (режим запуска по умолчанию);
- сервер (аналогичен многопользовательскому, но без GUI).

---

# Варианты init

- **System V:**
  - Все службы запускаются последовательно.
- **BSD init:**
  - FreeBSD, NetBSD, OpenBSD.
- **systemd:**
  - упрощенный процесс загрузки;
  - параллельный запуск служб;
  - запись событий в системный журнал.



---

# Процесс запуска Init-V

- GRUB загружает и запускает ядро;
- ядро запускает `/sbin/init`;
- `init` разбирает `/etc/inittab` и выполняет сценарий для инициализации системы;
- `init` выполняет скрипт `/etc/rc.d/rc` или `/etc/init.d/rc`;
- скрипты из `/etc/rcn.d` или `/etc/init.d/rcn.d` запускают различные службы.

---

## Уровни запуска Init-V\*

0. остановка работы с выключением;
1. (S) — однопользовательский режим;
2. многопользовательский режим без выхода в сеть;
3. многопользовательский режим с сетью, но без запуска X;
4. обычно не применяется;
5. многопользовательский режим с сетью и запуском X (по умолчанию);
6. остановка работы компьютера с перезагрузкой.

\* Во всех ОС, кроме Debian

## Уровни запуска Init-V\*

- S. инициализация компьютера непосредственно после запуска;
- 0. остановка работы компьютера с выключением;
- 1. однопользовательский режим с доступом к сети;
- 2–5. многопользовательский режим с сетью и запуском X (по умолчанию);
- 6. остановка работы компьютера с перезагрузкой.

\* В ОС Debian

---

# systemd

**systemd** — современный менеджер управления службами Linux.

*Позволяет выполнять следующие действия со службами:*

- запускать/останавливать;
- добавлять/удалять;
- редактировать;
- собирать логи.

# systemd

user@user:~\$ top

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1006	root	20	0	813384	341216	16196	S	7,3	8,5	1:37.42	Suricata-Main
855	root	20	0	94520	2192	2016	R	0,3	0,1	0:01.44	sdrplay_apiServ
874	redis	20	0	52660	3688	2620	S	0,3	0,1	0:00.74	redis-server
1252	user	20	0	465616	91632	51864	S	0,3	2,3	0:08.23	Xorg
1372	user	20	0	120500	2292	1920	S	0,3	0,1	0:00.83	VBoxClient
1409	user	20	0	2966580	231332	114028	S	0,3	5,7	0:26.11	gnome-shell
11015	user	20	0	43004	3984	3348	R	0,3	0,1	0:00.29	top
1	root	20	0	160392	9676	6728	S	0,0	0,2	0:02.87	systemd
2	root	20	0	0	0	0	S	0,0	0,0	0:00.00	kthreadd
3	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	rcu_gp
4	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	rcu_par_gp
5	root	20	0	0	0	0	I	0,0	0,0	0:00.33	kworker/0:0-ata
6	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/0:0H-kb

user@user:~\$ systemctl status

---

## Цели systemd

**Цель** (target) — нужное состояние системы; ссылка на файл, содержащий зависимости (службы).

➔ Systemd запускает все зависимости из соответствующего target-файла.

➔ Когда все зависимости будут запущены, то система будет работать на соответствующем target-уровне.

```
user@user:~$ systemctl get-default
```

```
graphical.target
```

# Модули systemd

**Модуль** (unit) — описывает запускаемую службу, устройство и т.п.

*Каждый модуль описан в своем файле (unit file):*

- [/usr/lib/systemd/system/](#) — модули из пакетов (Nginx, Apache, MySQL);
- [/run/systemd/system/](#) — модули, созданные во время работы ОС;
- [/etc/systemd/system/](#) — модули, созданные пользователем.

---

## Типы модулей

- модули служб — обычные службы ОС;
- модули монтирования — монтируют ФС;
- целевой модули/цели — группируют другие модули;

```
user@user:~$ systemd-analyze plot > test.svg
```

```
user@user:~$ sudo systemctl list-dependencies
```



# systemctl

systemctl **list-units** — список модулей

systemctl **list-units --type=service** — список модулей-служб

systemctl **status** <модуль> — состояние выбранного модуля

systemctl **enable\disable** <модуль> — разрешить/запретить модуль

systemctl **start\stop\restart** <модуль> — запустить/остановить/...  
модуль

systemctl **daemon-reload** — перезапуск конфигурации systemd

---

# journalctl

**Журнал** — база данных, в которой хранятся сообщения ядра и служб, начиная с загрузки и заканчивая завершением работы.

```
user@user:~$ journalctl
```

*Настройки журнала:*

```
user@user:~$ nano /etc/systemd/journald.conf
```

# journalctl

user@user:~\$ journalctl -u=sshd — сообщения для модуля ssh

user@user:~\$ journalctl -b 0 -u ssh — ...только в текущем сеансе

user@user:~\$ journalctl -n 100 /usr/sbin/sshd — показать внешний лог

user@user:~\$ journalctl --since=yesterday --until=now — временной период



# Итоги

---

# Итоги

## Сегодня мы:

- рассмотрели как происходит загрузка современного компьютера с ОС Linux;
- получили представление о работе [GRUB](#), научились сбрасывать пароль Linux и устанавливать котика на экран загрузки;
- начали рассмотрение менеджера [systemd](#), которому у нас будет посвящена отдельная лекция.

---

# Домашнее задание

Давайте посмотрим ваше [домашнее задание](#).

- Вопросы по домашней работе задавайте в чате учебной группы и/или в разделе “Вопросы по заданию”.
- Задачи можно сдавать **по частям**.
- Зачёт по домашней работе проставляется после того, как **приняты все задачи**.

**Задавайте вопросы и  
пишите отзыв о лекции!**

**Александр Гришин**