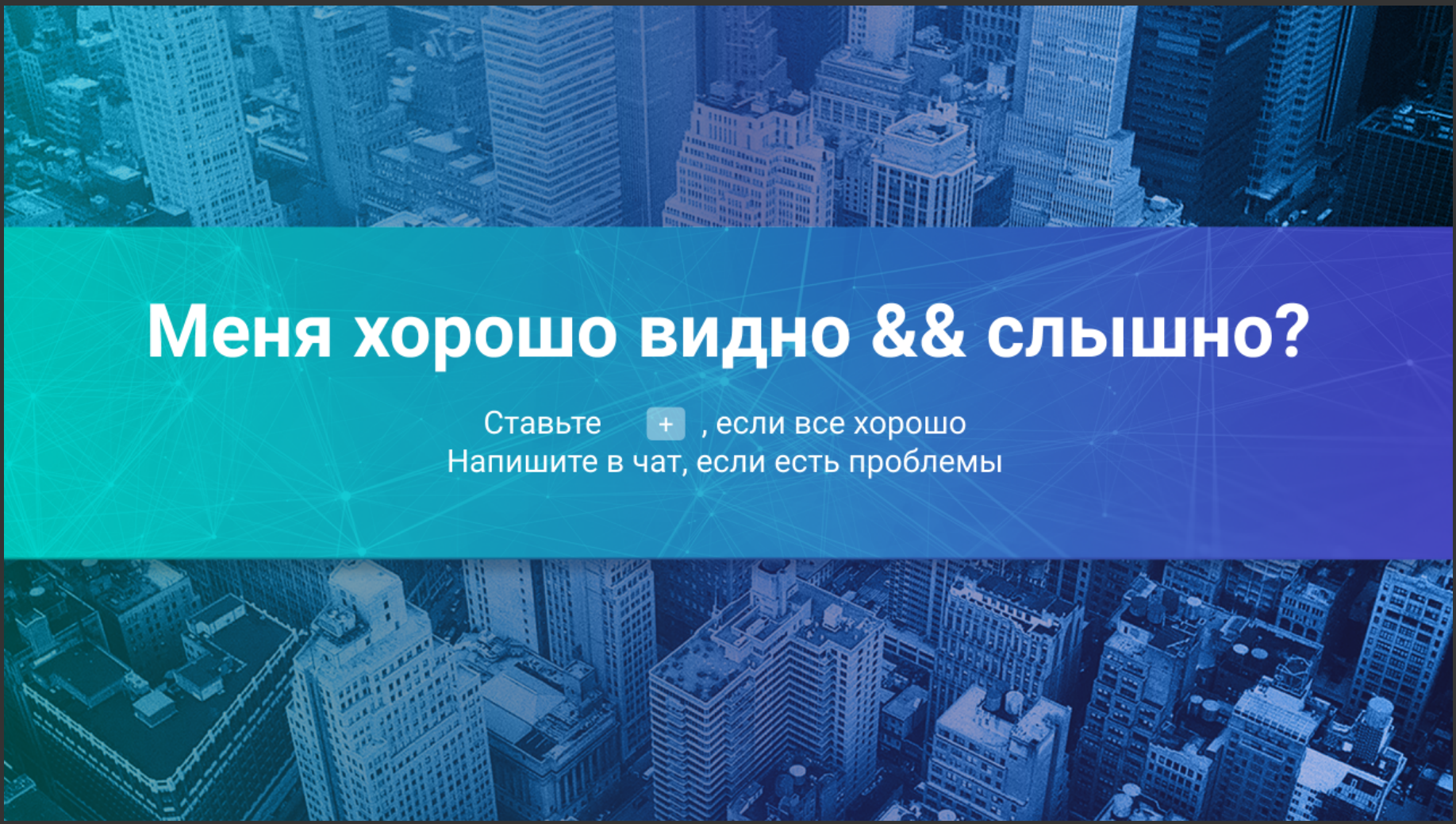




Онлайн-образование



Меня хорошо видно && слышно?

Ставьте , если все хорошо
Напишите в чат, если есть проблемы

НЕ ЗАБЫТЬ ВКЛЮЧИТЬ
ЗАПИСЬ!!!

Загрузка ОС Linux

Цели

- Понять основные различия BIOS vs UEFI
- Разобраться в процессе загрузки
- Познакомиться с GRUB
- Познакомиться с initrd

Шаги процесса загрузки

- Включение питания
 - Загрузка BIOS/UEFI из NVRAM
 - Тест “железа”
 - Выбор загрузочного устройства (диск, сеть)
- Запуск загрузчика
 - GRUB
 - Выбор в какое ядро грузиться
- Загрузка ядра
 - Создание структуры данных ядра
 - Старт PID 1 (init/systemd)

Basic Input-Output System (BIOS)

- Остался только в виртуалках (SeaBIOS)
- Требуется полуторный загрузчик из-за недостатка флеш памяти. BIOS должен работать в 16-битном режиме процессора и ему доступен всего 1 Мб памяти.
- Проблемы с одновременной инициализацией нескольких устройств

Загрузка из MBR

- Ищет в первом секторе диска (первые 512 байт) MBR загрузочного диска информации о полноторном загрузчике
- Загружает эту запись в NVRAM и приступает к ее исполнению
- Т.к. первый этап ДОЛЖЕН влезать в 1 сектор на диске он занимает мало места и вследствие этого не очень “умный”
- обнаружить и загрузить этап 1.5 - он располагается в пространстве между самой загрузочной записью и первым разделом на диске.
- После загрузки 1.5 этапа в RAM этап 1 передает ему контроль

1-st stage

- находится на 1-63 секторе жесткого диска.
- там хранится core.img файл GRUB-а ~25389 байт
- поддерживает некоторые файловые системы (EXT, FAT, NTFS)
- этап 2 может находиться, например, в ext файловой системе, но не в логическом томе
- стандартное положение файлов этапа 2 - /boot/grub
- задача 1,5-го этапа в том чтобы загрузиться с необходимыми драйверами ФС и найти в файловой системе /boot

1.5 stage

- На этом этапе загружаются (по необходимости) модули из директории `/boot/grub2/i386-pc`
- Конечная же задача - обнаружить и загрузить ядро Linux в RAM и передать ему управление
- Ядро и связанные с ним файлы находятся в директории `/boot: vmlinuz-*.x86_`
- GRUB поддерживает загрузку одного или нескольких ядер предоставляя администратору выбирать из заранее сформированного меню

2-st stage

- Загрузчик первой стадии - ищет в первом секторе (первые 512 байт диска) загрузочного диска информации о полуторном загрузчике
- Полуторный загрузчик - выделенная секция на ФС (1-62 секторы), зарезервированная область под загрузчик. Ищет файлы загрузки второй стадии.
- Загрузчик второй стадии - файлы в папке /boot с конфигами и ядрами

UEFI преимущества

Придумали в IBM для своей же архитектуры Power в конце 80-х. По сути эта та же маленькая ОС со своей спецификацией

Улучшения по сравнению с BIOS:

- Стартует в защищенном режиме
- Знает что такое ФС сразу и знает про GPT. Умеет грузиться с дисков > 2TB
- Имеет модульную архитектуру - можно использовать свои приложения и загружать свои драйвера
- Встроенный менеджер загрузки

UEFI Secure Boot

- Он защищает от выполнения неподписанного кода не только на этапе загрузки, но и на этапе выполнения ОС, например, как в Windows, так и в Linux проверяются подписи драйверов/модулей ядра, таким образом, вредоносный код в режиме ядра выполнить будет нельзя. Но это справедливо только, если нет физического доступа к компьютеру, т.к., в большинстве случаев, при физическом доступе ключи можно заменить на свои.
- Secure Boot призван защитить от буткитов, от атак типа Evil Maid.

EFI system partition

Что ему надо:

- Таблица разделов GPT EFI system partition
- Раздел \efi\boot\boot[название архитектуры].efi в FAT

Преимущества UEFI

Отличие в том, что в UEFI нет никакого полуторного загрузчика, в остальном же загрузка происходит так же как в BIOS

Можно грузиться с дисков > 2TB

Initrd. Общий принцип

Маленький образ ОС, единственная задача которого

- обработать модули (собрать LVM, собрать рейд),
- перемонтировать rootfs
- передать управление ядру:

Шаги в initrd

- инициализация ядра
- запуск /sbin/init
- загрузка модулей и некоторые этапы инициализации
- монтирование корня `mount / pivot_root()`

root fs:

- монтирование остальных разделов
- инициализация сети и запуск сервисов

GRUB

- Разработан проектом GNU
- GRUB Legacy и GRUB2 (текущий)
- GRUB является эталонной реализацией загрузчика, соответствующего спецификации Multiboot
- GRUB умеет передавать управление другому загрузчику - т.н. multichain booting
- GRUB позволяет пользователю при загрузке задавать произвольные параметры и передавать их в ядро Multiboot-совместимой ОС для дальнейшей обработки.

Файлы в /boot

- **/boot** - основной каталог. Отдельный затем, чтобы обеспечить работу с, например, рейдами. Сам по себе "полуторный" загрузчик не понимает софтверных рейдов
- **/boot/initramfs-*** - образ initrd. При обновлении ядра каждый раз собирается новый.
- **/boot/vmlinuz-*** - ядро Linux
- **/boot/System.map-*** - это ссылки на все экспортированные функции ядра. Нужен для утилиты insmod чтобы правильно скомпоновать ядро. Генерируется при сборке ядра

Файлы в `/boot/grub2`

- `/boot/grub2/` - основной каталог загрузчика
- `/boot/grub2/device.map` - список дисков/разделов в читаемом для GRUB-а формате
- `/boot/grub2/grub.cfg` - скрипт для подгрузки модулей, рисования менюшки пользователю. Файл генерируется автоматом. Править можно, но следует помнить что при обновлении ядра - будет перезаписан.
- `/boot/grub2/grubenv` - файл с небольшим кол-вом сохраненных состояний - переменными окружения. Во время загрузки в нее сохраняются эти переменные, а из работающей системы их можно использовать для редактирования окружения grub

Конфигурационные файлы grub

- **/boot/grub2/i386-pc/** - директория с разнообразными модулями
- **/etc/grub2.cfg** - симлинк в **/boot/grub2/grub.cfg**
- **/etc/grub.d/** - скрипты для формирования конфига GRUB-а
- **/etc/default/grub** - переменные для формирования grub.cfg. Те переменные что будут добавляться к параметрам ядра при генерации конфига, например, при обновлении самого ядра.
- **/etc/grub.d/40_custom** - с помощью этого скрипта можно создавать свои пункты меню с запуском кастомных ядер

Скрипты из grub.d

```
#!/bin/sh
exec tail -n +3 $
# This file provides an easy way to add custom menu entries. Simp
# menu entries you want to add after this comment. Be careful not
# the 'exec tail' line above.
menuentry 'OTUS Kernel' {
set root='(hd1,msdos1)'
linux /otus_kernel root=/dev/sdb1 ro quiet
initrd /initrd_otus_kernel.img
}
```

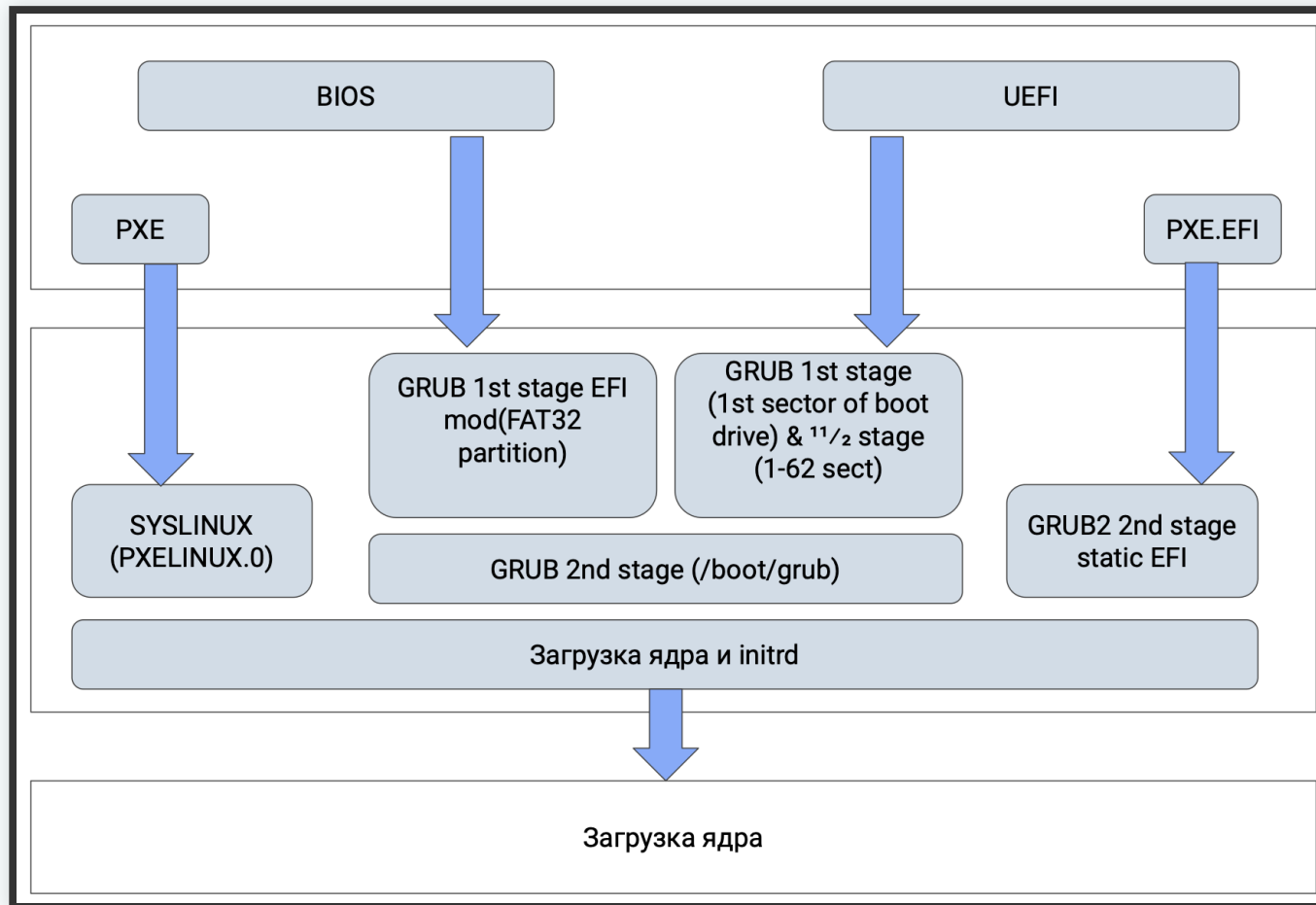
GRUB2 конфигурация

/etc/default/grub Часто модифицируемые опции:

- GRUB_CMDLINE_LINUX - параметры ядра
- GRUB_BACKGROUND - фоновая картинка
- GRUB_DEFAULT - пункт меню “по умолчанию”
- GRUB_DISABLE_RECOVERY - убирает автогенерацию рековери пунктов
- GRUB_PRELOAD_MODULES - перечень GRUB модулей
- GRUB_TIMEOUT - время в секундах до автовыбора

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```

Общий принцип



PXE

- Pre Boot Execution Environment Для организации загрузки системы в PXE используются протоколы IP, UDP, DHCP и TFTP. PXE-код, прописанный в сетевой карте, получает загрузчик из сети, после чего передаёт ему управление. По сути это такая маленькая ОС на PCI карте
- Cobbler - <https://cobbler.github.io>
- Foreman - <https://www.theforeman.org>

Параметры ядра

- Параметры, обрабатываемые самим ядром
 - Список и описание параметров ядра
 - <https://www.kernel.org/doc/Documentation/admin-guide/kernel-parameters.txt>
 - `man 7 bootparam`
 - Пример: `root= nomodeset ipv6.disable=1`
- Параметры, обрабатываемые системой инициализации
 - <https://www.freedesktop.org/software/systemd/man/kernel-command-line.html>
 - Пример: `quiet rhgb single`
- Параметры загрузки доступны через файл `/proc/cmdline`

Параметры ядра. Частный случай

Восстановление пароля

- https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/system_administrators_guide/terminal_menu_editing_during_boot#sec-Changing_and_Resetting_the_Root_Password

параметр **rd.break** запускает шелл перед `pivot_root()`.

Корневая файловая система будет находиться в `/sysroot`

Параметры ядра. Частный случай

Восстановление пароля, способ 2:

```
добавляем параметр init=/sysroot/bin/sh
```


Параметры ядра. Частный случай

SystemD Single User Mode:

```
systemd.unit=emergency.target  
systemd.unit=rescue.target
```

Initrd

Initrd - временная файловая система, используемая при начальной загрузке для монтирования корневой файловой системы для которой в свою очередь необходим модуль для работы с диском и ФС, а для чтения модуля необходима файловая система, с которой этот модуль читается.

Проблема курицы и яйца в полный рост)

Смотрим что внутри Initrd

Разборка:

```
zcat /boot/initrd-$(uname -r).img | cpio -i
```

Сборка:

```
find. -print0 |  
cpio -o --null --format=newc |  
gzip -q -9 > /boot/initrd-$(uname -r).img
```

Initrd. dracut

- <https://www.kernel.org/pub/linux/utils/boot/dracut/dracut.html>
- В CentOS для управления initrd используется dracut, который позволяет
 - модифицировать
 - просматривать содержимое initrd
 - вставлять свои скрипты в разные этапы загрузки.

Части модуля (функции-хуки)

исполняются в определенные этапы загрузки:

- `cmdline` - самое начало загрузки `initrd`
- `pre-udev` - перед запуском `udev`-подсистемы
- `pre-trigger` - в процессе запуска `udev`'а, возможность с ним взаимодействовать
- `pre-mount` - перед монтированием файловых систем
`mount` - смонтировать `root-filessystem`
- `pre-pivot` - после монтирования перед `pivot_rootcleanup` - перед `pivot_root` для “подчистки за собой”

Пример модуля

```
#!/bin/bash
check() {
return 0
}
depends() {
return 0
}
install() {
inst_hook cleanup 00 "${moddir}/test.sh"
}
```

Пример модуля

```
#!/bin/bash
exec 0<>/dev/console 1<>/dev/console
2<>/dev/console
cat <<'msgend'
Hello! You are in dracut module!
```

```
< I'm dracut module >
```

● ————— ●

$$\left| \begin{array}{cc} 0 & 0 \\ 0 & 0 \end{array} \right|$$

(1) (2) (3) (4)

(11)

— 3 —

Рефлексия



Отметьте 3 пункта, которые вам запомнились с вебинара



Что вы будете применять в работе из сегодняшнего вебинара?

The background of the slide is an aerial photograph of a city with many skyscrapers, overlaid with a semi-transparent blue layer. A network of white lines connects various points across the blue area, creating a digital or technological aesthetic.

Заполните, пожалуйста,
опрос о занятии по ссылке в чате

Домашнее задание

1. Попасть в систему без пароля несколькими способами
2. Установить систему с LVM, после чего переименовать VG
3. Добавить модуль в initrd
4. (*) Сконфигурировать систему без отдельного раздела с /boot, а только с LVM Репозиторий с пропатченным grub:
https://yum.rumyantsev.com/centos/7/x86_64/ PV
необходимо инициализировать с параметром --
bootloaderareaseize 1m