OTUS

ОНЛАЙН-ОБРАЗОВАНИЕ

OTUS

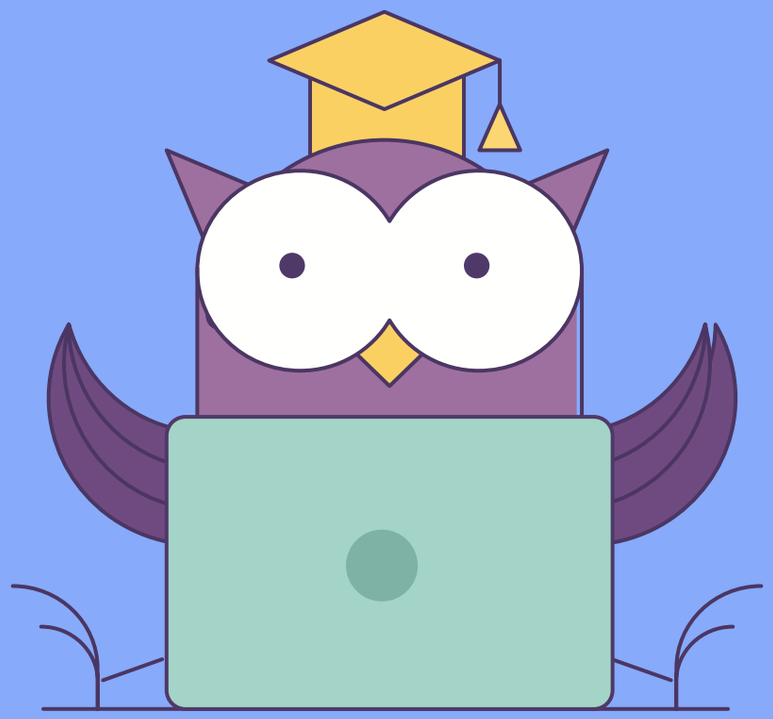
LVM и файловая система

Курс «Администратор Linux»

Занятие № 3

Александр Румянцев Алексей Цыкунов Альбрехт Леонид

Меня хорошо слышно && видно?



Напишите в чат, если есть проблемы!

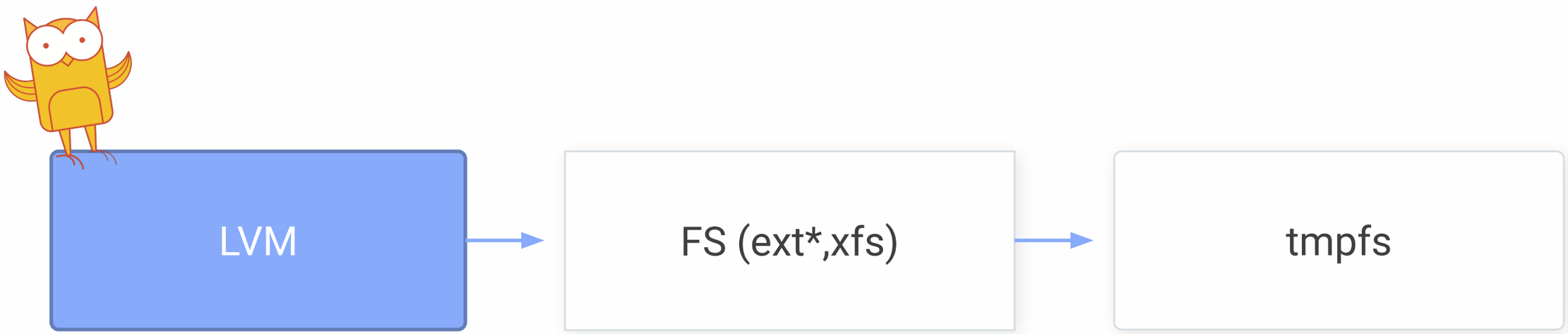
Ставьте + если все хорошо
Ставьте - если есть проблемы

Цели вебинара

По итогам занятия вы сможете:

- Рассказать о том, что такое LVM и его use cases
- Рассказать о некоторых особенностях файловых систем

Маршрут вебинара



LVM

- **LVM** - Logical Volume Manager. Специальная подсистема ядра, которая добавляет дополнительный уровень абстракции от “железа”, позволяя управлять дисковым пространством и решать разнообразные задачи по управлению дисковой подсистемой:
  - Группировка физических томов
  - Создание и изменение размеров логических томов (на лету)
  - Snapshots ("снимки")
  - Thin provisioning
  - Cache volumes
  - LVM mirror
  - LVM stripes

Device-mapper

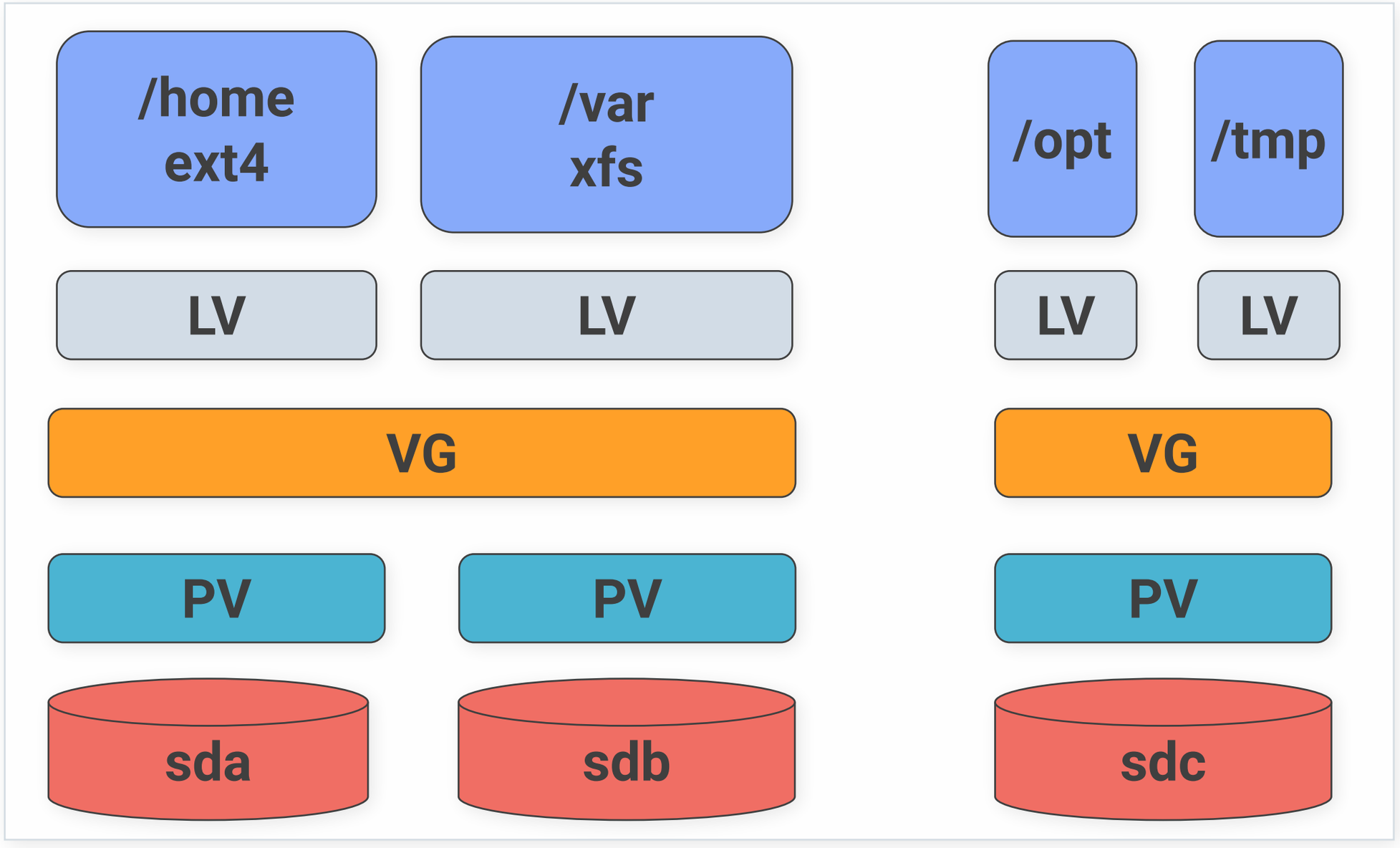
Device mapper (dm) - модуль ядра, позволяющий работать с виртуальными блочными устройствами (логическими дисками). По сути посылает информацию с виртуального устройства на реальное. Некоторые из его возможностей:

- Кэширование
- Шифрование
- Зеркалирование (mirror drives)
- Мультипасинг (multipath)
- Рэйд (mdadm)
- Thin-provision
- Stripe
- Snapshot

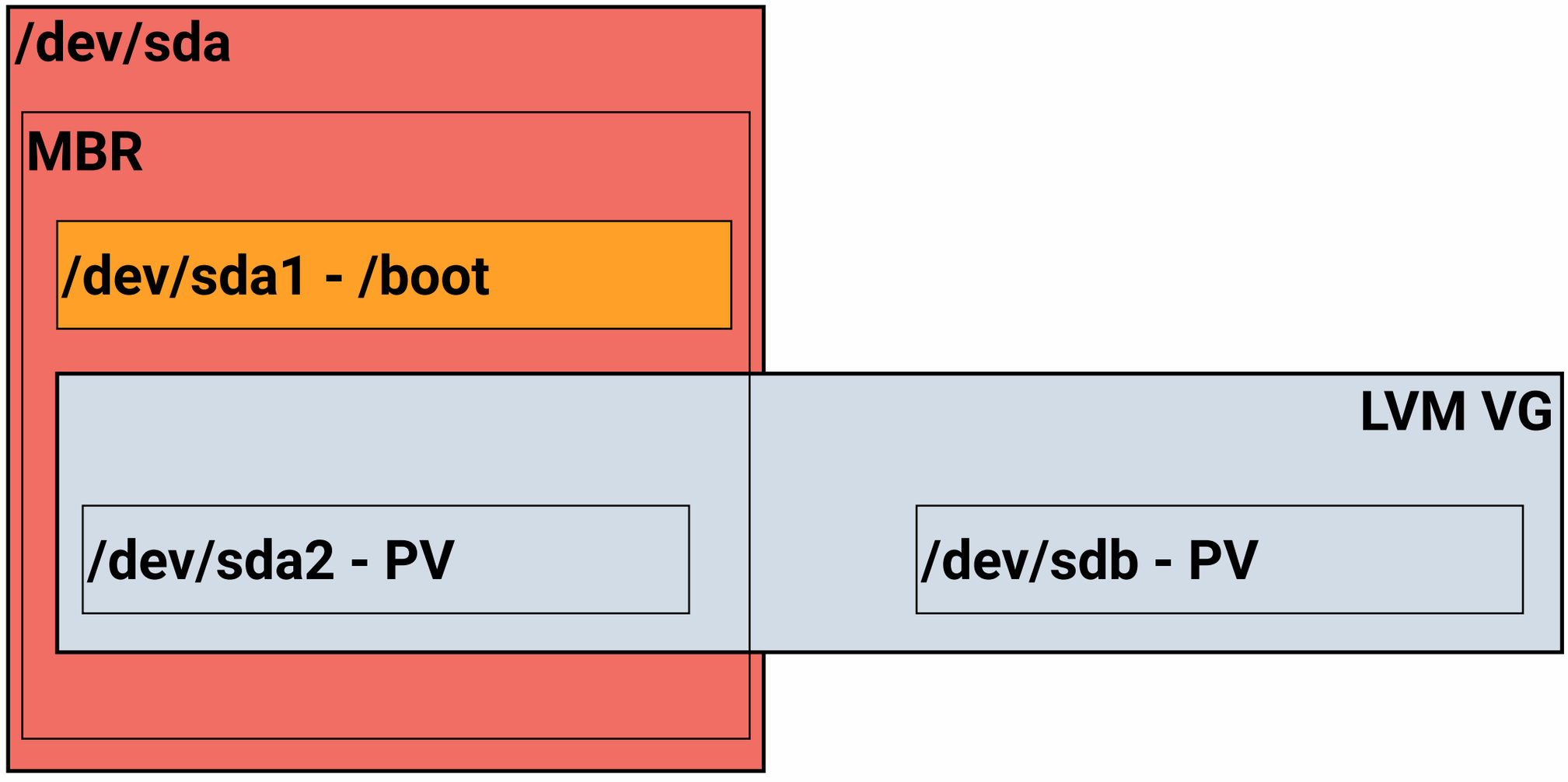
LVM Use Cases

- Для ежедневного использования. Ресайз томов
- Управление большой фермой дисков. Hot swap.
- На десктопах
- Для бэкапов. Не лучший метод, если держать снэпшот.
- HA-Cluster

PV - Physical Volume - любое блочное устройство
VG - Volume Group - группа PV
LV - Logical Volume - часть VG, доступная в виде блочного устройства



Пример использования на VM



Для использования требуется пакет **lvm2** и **device-mapper**

/usr/sbin/lvm - основная утилита, всё остальное - симлинки

```
[root@otuslinux ~]# pvcreate /dev/sda2
[root@otuslinux ~]# vgcreate vg0 /dev/sda2 # -s <PE (Phys Extent)size>
[root@otuslinux ~]# pvcreate /dev/sdb
[root@otuslinux ~]# vgextend vg0 /dev/sdb
[root@otuslinux ~]# lvcreate -l 10 -n home vg0 # 10 extents of 4M
[root@otuslinux ~]# lvcreate -L 1G -n root vg0 # 1GiB
[root@otuslinux ~]# lvcreate -l 100%FREE -n var vg0 # rest of VG
```

Управление и конфигурирование

OTUS

```
[root@otuslinux ~]# vgscan
[root@otuslinux ~]# vgchange -ay
[root@otuslinux ~]# pvs | vgs | lvs
[root@otuslinux ~]# pvremove | vgremove | lvremove
```

Конфигурация

/etc/lvm/ - хранилище кэша конфигурации и резервных копий
/etc/lvm/lvm.conf - основная конфигурация для утилиты lvm

- archive/ - информация за все время
- backup/ - бэкап текущей конфигурации
- cache/ - кэш
- profile/ - готовые (или самописные профили)

LVM Snapshot

- Принцип: создание нового LV, зависимого от оригинального LV, куда будут копироваться оригинальные блоки данных перед тем, как в оригинальный том будут записаны новые блоки данных.
- Должно быть место в VG под snapshot
- В результате при использовании снапшотов мы получаем двойную запись → замедление дисковых операций
- Удаление снапшота - быстрое, откат на снапшот - медленный.
- Снапшот можно монтировать, и даже в read-write режиме

LVM Snapshot

Процесс создания снапшота:

```
[root@otuslinux ~]# lvcreate -L 500M -s -n test-snap /dev/otus/test

[root@otuslinux ~]# lvconvert --merge /dev/otus/test-snap

[root@otuslinux ~]# lvremove /dev/otus/test-snap
```

LVM Thin Provision

Overbooking для LVM: Возможность выделить места больше, чем есть.
Используется в виртуализации и контейнеризации

Принцип: создается основной LV (thin pool), после чего создаются зависимые LV с указанием виртуального размера.

Основные команды:

```
[root@otuslinux ~]# lvcreate -L 100G -T vg0/lv-thinpool
[root@otuslinux ~]# lvcreate -V 100G -T vg0/lv-thinpool -n lv1
[root@otuslinux ~]# lvcreate -V 100G -T vg0/lv-thinpool -n lv2
[root@otuslinux ~]# lvcreate -V 100G -T vg0/lv-thinpool -n lv3
```

LVM Cache

Вынос часто используемых данных на SSD

- В основном используется на десктопах
- Суть в том, что добавляется кеш LV
- Нельзя использовать снапшоты

Очень краткая и ёмкая документация:
http://man7.org/linux/man-pages/man7/lvmcache.7.html

LVM Mirror

Зеркалирование на уровне LVM.

Основные команды:

```
[root@otuslinux ~]# pvcreate /dev/sda /dev/sdb
[root@otuslinux ~]# vgcreate vg0 /dev/sda /dev/sdb
[root@otuslinux ~]# lvcreate -L 50M -m1 -n mirror vg0
```

Маршрут вебинара

Файловую систему можно разделить на четыре основных компонента:

- Именованное пространство (namespace) - то как вещи (файлы, директории) представлены и организованы (иерархия)
- API - набор системных вызовов для навигации и управления объектами
- Модель безопасности - схемы для защиты, скрытия и разделения информации
- Реализация - софт для переноса логической модели на “железо”

hier[archy]

Основная документация: man hier

- Файловая система в linux (да и unix в целом) имеет иерархическую/древовидную организацию. Принято минимизировать количество разделов в корне и размещать там стартовые ветки иерархий. В корне мы обычно видим следующие каталоги:

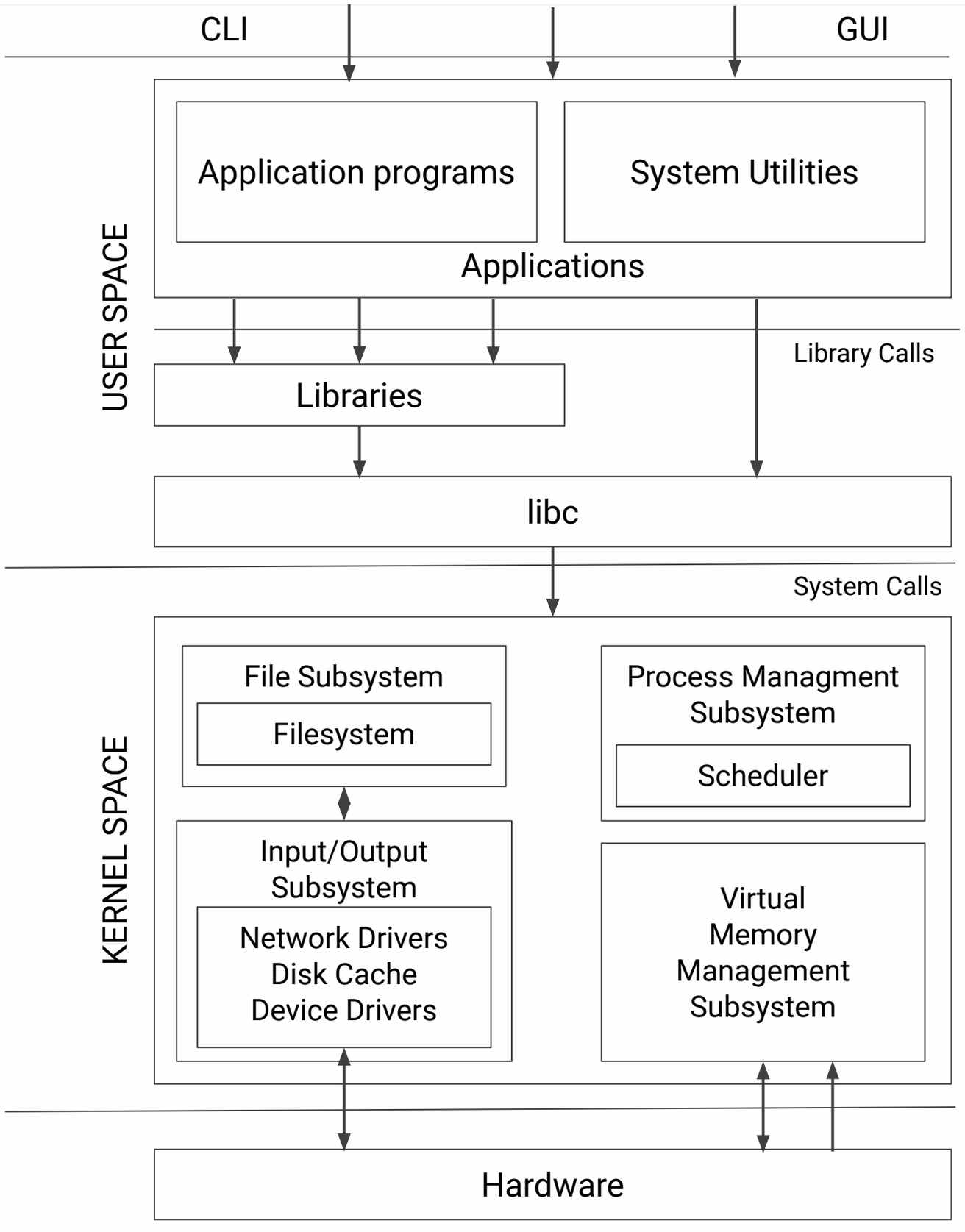
- **/boot** - информация необходимая для загрузки
- **/bin,/sbin** - системные исполняемые файлы.
- **/etc** - файлы конфигурации системы и приложений
- **/home** - домашние каталоги пользователей
- **/var** - динамически изменяемая информация (БД, Кэши, логи)
- **/tmp (tmpfs)** - временные файлы
- **/lib[64]** - системные библиотеки
- **/usr** - пользовательские (или системные) программы.

В современных системах имеет смысл выделять при установке:

/ - 8G
/home - 8G
/var - 16G

Для приложений стоит выделять отдельные тома (например для mysql - отдельный том в **/var/lib/mysql**)

Calls Arch


CLI
GUI
Application programs
System Utilities
Applications
USER SPACE
Library Calls
Libraries
libc
System Calls
File Subsystem
Filesystem
Process Managment Subsystem
Scheduler
Input/Output Subsystem
Network Drivers
Disk Cache
Device Drivers
Virtual Memory Management Subsystem
KERNEL SPACE
Hardware

Псевдофайловые системы

/proc

/sys

По сути две виртуальный файловые системы, которые предоставляют некий интерфейс от пользователя к ядру. А именно, /proc - к запущенным процессам, а /sys - информацию об устройствах, модулях ядра, файловых системах и пр.

http://man7.org/linux/man-pages/man5/sysfs.5.html#NOTES

Страничный кэш

Параметры работы **pdflush:**

/proc/sys/vm/dirty_expire_centisecs - время жизни dirty data в памяти
/proc/sys/vm/dirty_background_ratio - объем кеша, занятого dirty data (% кеша)
/proc/sys/vm/dirty_ratio - объем кеша, занятого dirty data (% общей памяти)
/proc/sys/vm/dirty_writeback_centisecs - период работы pdflush

fsync() - системный вызов, записывающий все измененные данные для конкретного файла на диск (man 2 fsync)

Сброс кешей:
[root@otuslinux ~]# sync
[root@otuslinux ~]# echo 3 > /proc/sys/vm/drop_caches

block. superblock. inode. hardlink.

Блок - минимальный адресуемый размер дискового пространства. Исторически - 512 байт. Так же это минимально аллоцируемый размер под файл.

Суперблок - информация о файловой системе:
- размер ФС
- размер блока
- битмап занятых блоков
- расположение и размер групп блоков и таблиц inode

```
[root@otuslinux ~]# dumpe2fs -h /dev/sda2
```

block. superblock. inode. hardlink.

Inode (индексный дескриптор) - структура, содержащая информацию о файле:
- размер файла в байтах;
- идентификатор владельца файла;
- идентификатор группы-владельца файла;
- режим доступа к файлу, определяющий кто и какой доступ имеет к файлу;
- дополнительные системные и пользовательские флаги, которые дополнительно могут ограничивать доступ к файлу и его модификацию;
- временные метки, отражающие время модификации индексного дескриптора (ctime, changing time), время модификации содержимого файла (mtime, modification time) и время последнего доступа к файлу (atime, access time);
- счётчик для учёта количества жёстких ссылок на файл;
- указатели на физические блоки диска, в которых хранится содержимое файла

Inode #2 - root

Директория - это такой-же файл, содержимое которого представляет индекс (B-tree), содержащий имя файла (ключ) и номер inode (значение)
- порядок файлов в директории не гарантируется
- перечисление всех файлов - очень медленно, но доступ к любому быстрый

Файловые системы

ext2 - исторически “стандартная” для Linux. файловая система решавшая много ограничений своих предтечей - ext и minix. Считается эталоном производительности. Поддерживается online resize

ext3 - Логическое продолжение ext2, расширены ограничения на размер файлов и тома, добавлена возможность журналирования

ext4 - Логическое продолжение ext3. Номинально сильно увеличены ограничения на размер тома, по факту из коробки все еще 4Тб на том, возможность хранить ext. attributes в Inode, увеличение inode (128->256b), решен вопрос со вложенными каталогами (>32000)

Файловые системы

XFS - высокопроизводительная журналируемая файловая система родом из SGI (Silicon Graphics).

Преимущества

+ динамическая аллокация inode
+ дефрагментация на лету
+ потенциально лучшая производительность
+ встроенные средства для резервного копирования/снапшотов (xfsdump/xfsrestore)
+ “отсутствие” жестких ограничений на размер файловой системы

Недостатки

- малая ремонтопригодность
- выше вероятность сбоя из-за хранения большого количества данных в памяти.
- невозможность уменьшить

Некоторые общие возможности:

- Copy-on-Write - ВАЖНО! Не то же самое что в LVM
- Error Detection - чексумма для каждого файла + внутренний функционал
- Производительность - в состоянии по умолчанию слегка выше, чем в традиционных
- Снапшоты
- RAID - обе поддерживают возможности создания рейдов

Журналирование

- Хранение логов изменения метаданных. Не гарантирует сохранение данных, но гарантирует консистентность файловой системы. В результате восстановление после сбоя - это "проигрывание" журнала вместо полного fsck.

Для ext4 есть три варианта (mount -o data=)

journal - пишем сначала в журнал, самый медленный метод
ordered - пишем сначала файловую систему, потом журнал
writeback - не гарантируется порядок изменений, в файле после сбоя может появиться блок данных, который был до сбоя удалён; самый быстрый метод.

Опции ФС

man mount

mount -o opt,opt=val /dev/dev /mountpoint

Самые распространенные:
noatime
data=
user
noexec
noauto

Знакомство с fstab

Конфигурационный файл, содержащий информацию о блочных устройствах, файловых системах на них и о том, как они будут интегрированы в систему.

```
[root@otuslinux ~]# cat /etc/fstab | grep -v '^#'
/dev/mapper/VolGroup00-LogVol00                 /                       xfs     defaults        0       0
UUID=570897ca-e759-4c81-90cf-389da6eee4cc       /boot                   xfs     defaults        0       0
/dev/mapper/VolGroup00-LogVol01                 swap                    swap    defaults        0       0

        <device>                                <mount_point>           <fs><options>       <dump>  <fsck>
```

Опции ФС

- Команда fuser -c точка монтирования выводит идентификаторы всех процессов, обращающихся к файлам или каталогам указанной файловой системы

```
[root@otuslinux ~]# fuser -c /usr
[root@otuslinux ~]# ps -fp “9182 3123”
```

Немного про fsck

Иногда мы можем получить ФС в непредсказуемом состоянии. Для починки используется утилита **fsck**. При перезагрузке проверяем ФС из /etc/fstab.

Если вы хотите проверить файловую систему вручную то вначале ее необходимо отмонтировать, а затем уже запустить **fsck**:

```
[root@otuslinux ~]# umount /data
[root@otuslinux ~]# fsck -y /dev/sdc
```

Журналируемые файловые системы проверяются быстрее за счет того, что **fsck** проверяет только последние записанные данные.

tmpfs. ramfs. rootfs. initramfs.

https://www.kernel.org/doc/Documentation/filesystems/ramfs-rootfs-initramfs.txt

Всё это - методы предоставления доступа к кеширующему механизму на уровне пользователя

```
[root@otuslinux ~]# mount -t tmpfs none /mnt -o size=100M
```

swap

swap - это процесс когда страницы памяти копируются на заранее размеченное место на диске. Делается это затем, чтобы освободить место в памяти.

```
[root@otuslinux ~]# swapon --show
[root@otuslinux ~]# free -h

[root@otuslinux ~]# mkswap /dev/sdb
[root@otuslinux ~]# swapon[off] /dev/sdb

/proc/sys/vm/swappiness
vm.swappiness=1
```

Полезные утилиты

Некоторые полезные утилиты:

```
[root@otuslinux ~]# df -Th
[root@otuslinux ~]# du
[root@otuslinux ~]# stat
[root@otuslinux ~]# ncdu
[root@otuslinux ~]# lsof
[root@otuslinux ~]# fuser
[root@otuslinux ~]# fsck
[root@otuslinux ~]# mkfs.*
[root@otuslinux ~]# mount | column -t
```

Ваши вопросы?

Домашнее задание

На имеющемся образе:
/dev/mapper/VolGroup00-LogVol00   38G 738M   37G   2% /

1) Уменьшить том под / до 8G
2) Выделить том под /home
3) Выделить том под /var - сделать в mirror
4) /home - сделать том для снэпшотов
5) Прописать монтирование в fstab. Попробовать с разными опциями и разными файловыми системами (на выбор)

Работа со снапшотами:
- сгенерить файлы в /home/
- снять снапшот
- удалить часть файлов
- восстановится со снапшота
- залоггировать работу можно с помощью утилиты script

* на нашей куче дисков попробовать поставить btrfs/zfs - с кешем, снапшотами - разметить здесь каталог /opt

Рефлексия

Напишите, пожалуйста, свое впечатление о вебинаре.

- Отметьте 3 пункта, которые вам запомнились с вебинара.
- Что вы будете применять в работе из сегодняшнего вебинара?

Заполните, пожалуйста, опрос в ЛК о занятии

Спасибо
за внимание!
До встречи в Slack и на вебинаре