v.3.7

КУРС LL-102

Системное администрирование Linux

Yuriy Lebedev

SOFTLINE EDUCATION

http://edu.softline.ru/

2018

Оглавление

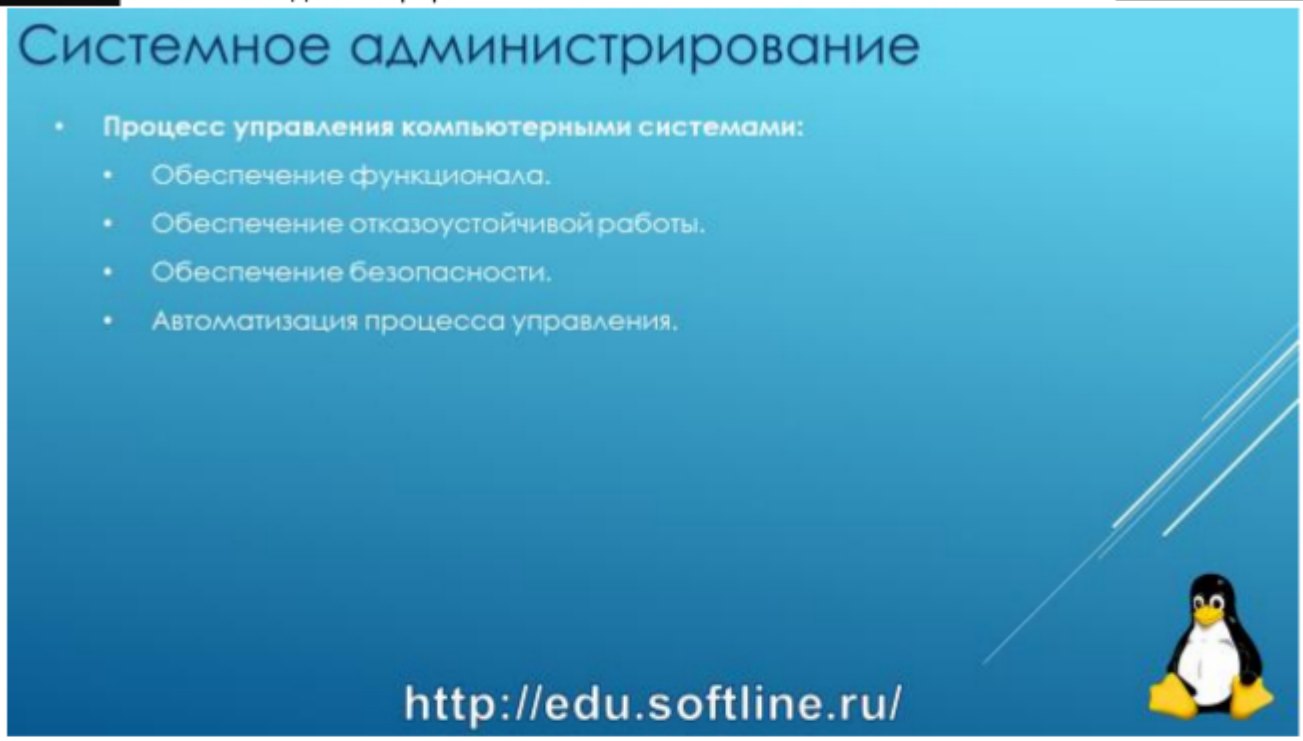
Модуль 01: Установка Linux. 2
Лабораторная работа 33
Модуль 02: Загрузка системы 42
Лабораторная работа 90
Модуль 03: Работа с оборудованием 95
Лабораторная работа 112
Модуль 04: Файловые системы. 115
Лабораторная работа 179
Модуль 5: Управление учетными записями. 198
Лабораторная работа 226
Модуль 6: Конфигурирование сети 234
Лабораторная работа 250
Модуль 07: Инструментальные средства системного администрирования 262
Лабораторная работа 339
Модуль 8: Пакетные менеджеры и сборка ПО. 349
Лабораторная работа. 431
Модуль 9: Ядро операционной системы 442
Лабораторная работа 458
Модуль 10: Настройка X-сервера 466
Лабораторная работа 486
Модуль 11: Устранение неполадок 493
Лабораторная работа 500

Модуль 01: Установка Linux.



В модуле представлены:

- Значение и принципы системного администрирования.
- Аппаратные архитектуры.
- Установка ОС Linux.
- Удаленное управление установкой Linux (VNC).
- Автоматическая установка Linux (Kickstart Installation).
- Установка Linux по сети (PXE).



Системное администрирование – процесс управления компьютерными системами.

Системный администратор – человек, выполняющий ряд процедур, направленных на обеспечение безотказной работы компьютерной системы.

В настоящее время системное администрирование становится очень ответственной профессией, от которой напрямую зависит жизнедеятельность компании. Основными задачами системных администраторов являются:

- Предоставление пользователям функционала, достаточного для выполнения своих прямых обязанностей.
- Проектирование компьютерной системы таким образом, чтобы обеспечить высокий уровень информационной безопасности.
- Обеспечение отказоустойчивой работы системы.
- Обеспечение защиты от вредоносных программ.
- Обеспечение защиты от действий компьютерных взломщиков.
- Автоматизация процесса администрирования.

Все эти задачи являются крайне важными для успешной и плодотворной деятельности предприятия.

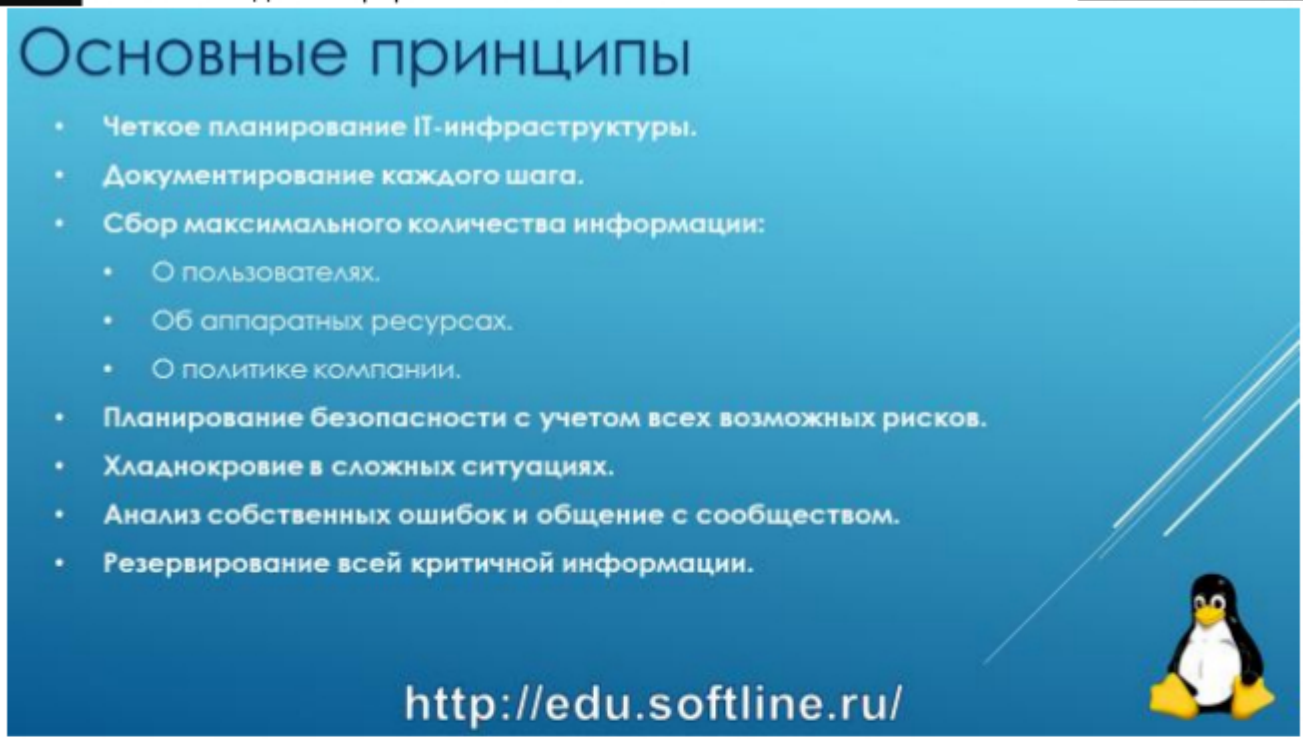
Каждый пользователь должен иметь тот функционал, который необходим ему для выполнения задач, поставленных перед ним. Отсутствие этого функционала повлечёт за собой простой в работе того или иного подразделения компании, а соответственно и материальные потери.

Профессиональный системный администратор должен уметь строить максимально безотказные информационные системы. Нельзя забывать, что все программы пишутся людьми и в этих программах существуют ошибки, когда ошибки начинают проявляться слишком часто, то при плохом подходе к проектированию, система даст сбой.

Администратор должен всегда развиваться и работать над проектами, которые помогут работе предприятия, если администратор будет постоянно решать проблемы, связанные с информационной

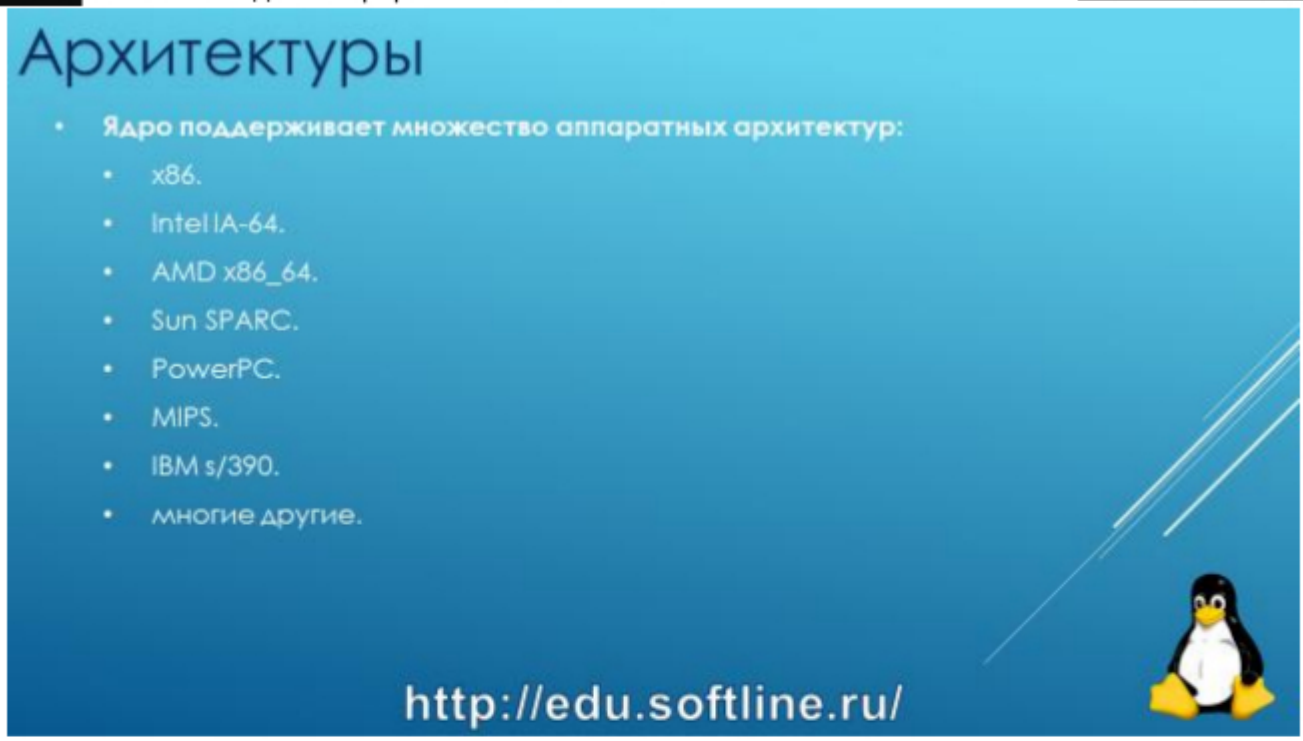
системой, то у него не останется времени на крупные проекты, соответственно, и на развитие тоже. Именно поэтому автоматизация процесса администрирования является одной из наиболее важных частей IT-системы.

Помните, вы всегда должны быть готовы к неожиданностям, в век информационных технологий, любая оплошность системного администратора может привести к крупным убыткам.



Проектирование IT-инфраструктуры требует очень тщательного продумывания работы всех элементов:

- Получите информацию обо всех рабочих процессах, происходящих в компании.
- Изучите всё аппаратное оснащение, которое будет участвовать в проектируемой системе.
- Одним из наиболее важных этапов в подготовке является сбор данных о требованиях пользователей, такую информацию может дать либо отдел кадров, если предприятие имеет большой штат сотрудников, либо сами сотрудники, если их не много.
- Составьте схему вашей инфраструктуры, она поможет вам заранее определить узкие места.
- Воплощая планируемые действия в жизнь, обязательно документируйте каждый шаг. Идеальный вариант, если администратор документирует каждое изменение, производимое в системе. Эта информация окажется крайне важной при сбоях в системе.
- При возникновении сбоев, никогда не паникуйте. Если это взлом, то спокойно отключите сервер от сети, налейте чашку кофе и начинайте разбираться, что произошло. Если у вас нет на это время, то первым делом восстановите работоспособность системы, сохранив "дефектную" составляющую. Всегда изучайте все сбои, свои ошибки, чем больше вы их будете разбирать, тем меньше вероятность их повторения.
- Никогда не забывайте, что абсолютно-безотказная система та, для которой есть резервные копии. Как бы администратор не старался, а потеря всех данных может произойти. Всегда делайте резервные копии важной информации.



Поддержка различных аппаратных платформ является одной из причин растущей популярности Linux.

Среди поддерживаемых архитектур можно увидеть, как привычные x86, x86_64, так и менее распространённые, и использующиеся в различных сегментах, SPARC или PowerPC.

Вы всегда можете взять исходные коды ядра и скомпилировать их под необходимую платформу.

В отличии от CentOS 5 и 6, которые традиционно распространялись в двух архитектурах (i386 и x86_64), начиная с 7-ой версии CentOS выпускается на основной архитектуре x86_64.

Дополнительно к основной архитектуре часть сообщества CentOS, которая называется Alternative Architecture Special Interest Group (AltArch SIG) выпускает CentOS 7 под другие архитектуры:

Версия CentOS	Архитектура
7 (1603)	aarch64
7 (1511)	armhfp (Arm32)
7 (1511)	i386
7 (1511)	ppc64le
7 (1511)	ppc64

Также у CentOS присутствуют сборки для Docker, Amazon и многих других платформ.

ПРИМЕЧАНИЕ:
В данном курсе все внимание уделяется архитектуре по умолчанию (x86_64).

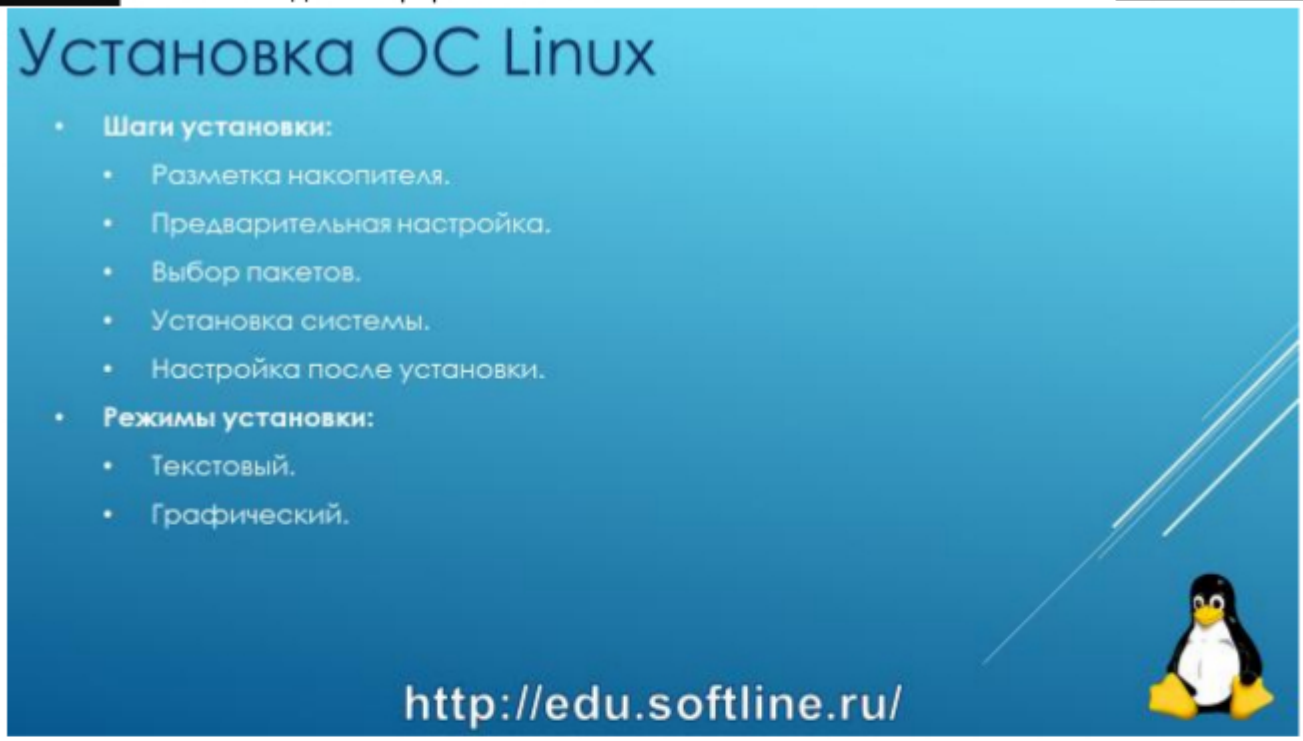


По ссылке https://www.centos.org/download/ вы можете получить установочные образы CentOS при помощи ближайшего к вам зеркала.

Популярные образы доступные для загрузки перечислены в таблице ниже:

Имя образа	Описание
CentOS-7-x86_64-DVD-1511.iso	Образ DVD-диска, содержит все пакеты, которые могут быть установлены при помощи установщика. Рекомендованный образ.
CentOS-7-x86_64-NetInstall-1511.iso	Образ для сетевой установки и восстановления системы. Установщик запросит расположение, из которого необходимо произвести установку. Этот образ очень полезен в случае использования локального зеркала.
CentOS-7-x86_64-Everything-1511.iso	Этот образ содержит полный набор пакетов для CentOS 7. Он может быть использован для установки системы или создания локального зеркала. Для этого образа необходим двухсторонний DVD или флэш накопитель USB объемом 8 Gb.
CentOS-7-x86_64-LiveGNOME-1511.iso **CentOS-7-x86_64-LiveKDE-1511.iso**	Эти образы – это живые образы CentOS 7. В зависимости от имени они содержат соответствующий оконный менеджер. Они предназначены для тестирования и знакомства с окружением CentOS. Живые образы не изменяют содержимое жесткого диска по умолчанию, за исключением опции установки живого образа на жесткий диск. Эти образы содержат ограниченный набор пакетов, который не может быть изменен в процессе установки.
CentOS-7-x86_64-Minimal-1511.iso	Назначение данного образа – это установка самой базовой системы CentOS 7 с минимальным набором пакетов необходимых для функционирования системы.

ПРИМЕЧАНИЕ:
В данном курсе будет использоваться рекомендованный образ CentOS-7-x86_64-DVD-1511.iso.



Перед установкой убедитесь, что всё оборудование, необходимое для работы, поддерживается ядром операционной системы.

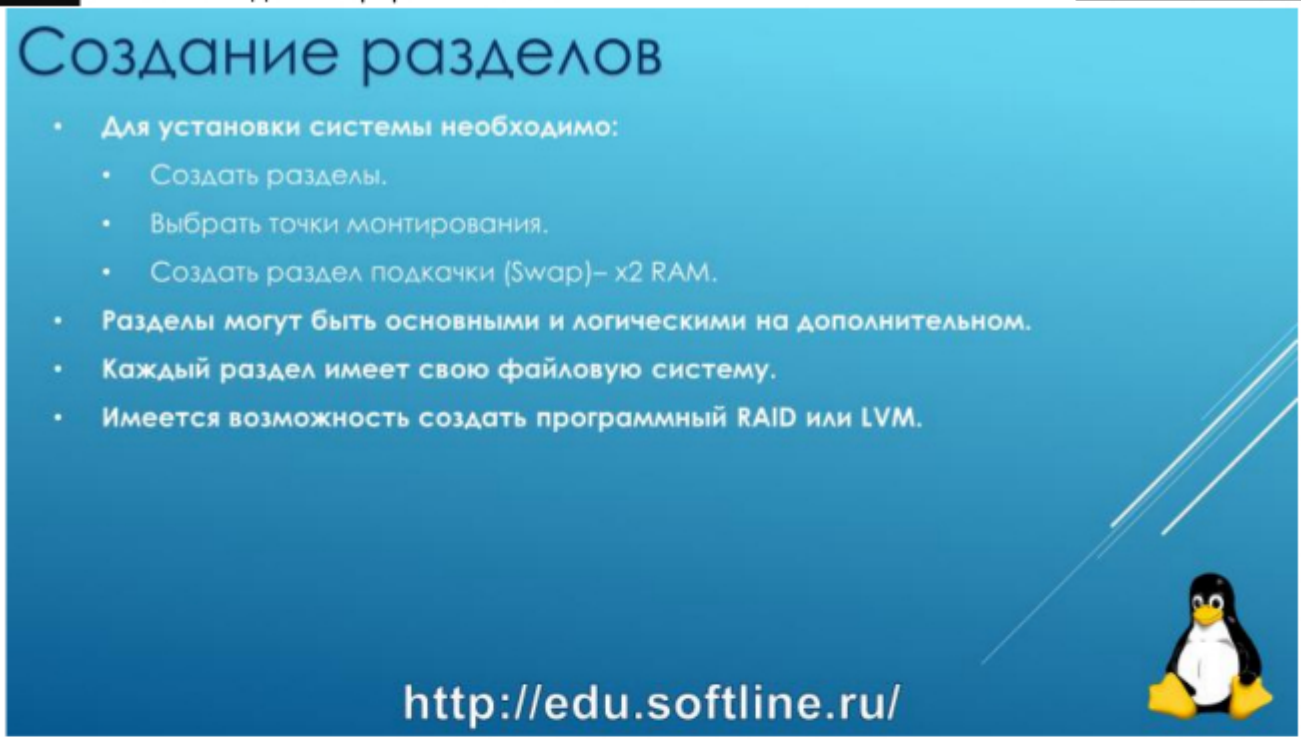
Установить дистрибутив операционной системы, построенной на ядре Linux, очень просто. На сегодняшний день, практически все программы установки предлагают пользователям, помимо классического – текстового, графический установщик. Кроме упомянутых двух типов установки, существует ещё один – сетевая установка, интересна администраторам, под управлением которых находятся удалённые компьютеры, либо при потребности установки на большое число систем.

Чтобы приступить к установке вставьте первый загрузочный диск в привод. Убедитесь, что в BIOS системы указан верный приоритет загрузочных устройств, первым должен стоять привод компакт-дисков. У каждого дистрибутива оригинальное приглашения к началу инсталляции, CentOS предлагает небольшое меню с описанием каждого типа. Если вы желаете начать установку в текстовом режиме, то необходимо добавить к командной строке:

```
... text
```

Если же вы хотите начать установку в графическом режиме, то просто нажмите клавишу <Enter>.

Система установки Anaconda задаст ряд вопросов, касающихся раскладки клавиатуры, даты и времени и т.д. Важной составляющей процесса установки является создание разделов на жёстком диске.



Физические диски делятся на разделы, которые могут содержать либо файловую систему, либо выделяться в качестве подкачки, как составляющая виртуальной памяти. Создание разделов позволяет изолировать данные друг от друга, это может быть множество операционных систем, либо домашние каталоги пользователей. Первые четыре раздела являются первичными разделами. При создании более чем 4-х разделов, выделяется специальный extended раздел, в котором находятся логические разделы.

Разделы могут быть объединены в некую сущность, массив. Такое объединение осуществляется с помощью аппаратного RAID-контроллера, либо его программного аналога. Существует несколько уровней RAID-массивов. Массив уровня RAID0 предназначен для обеспечения отказоустойчивости, когда все данные, записываемые на один раздел, будут отражаться на втором "зеркальном" разделе. Массив уровня RAID1 предназначен для увеличения скорости операций записи на диск, запись будет производиться поочерёдно, сначала на один, а затем на другой раздел, входящий в массив. При этом размер массива уровня RAID1 будет равен суммарному размеру разделов, входящих в него.

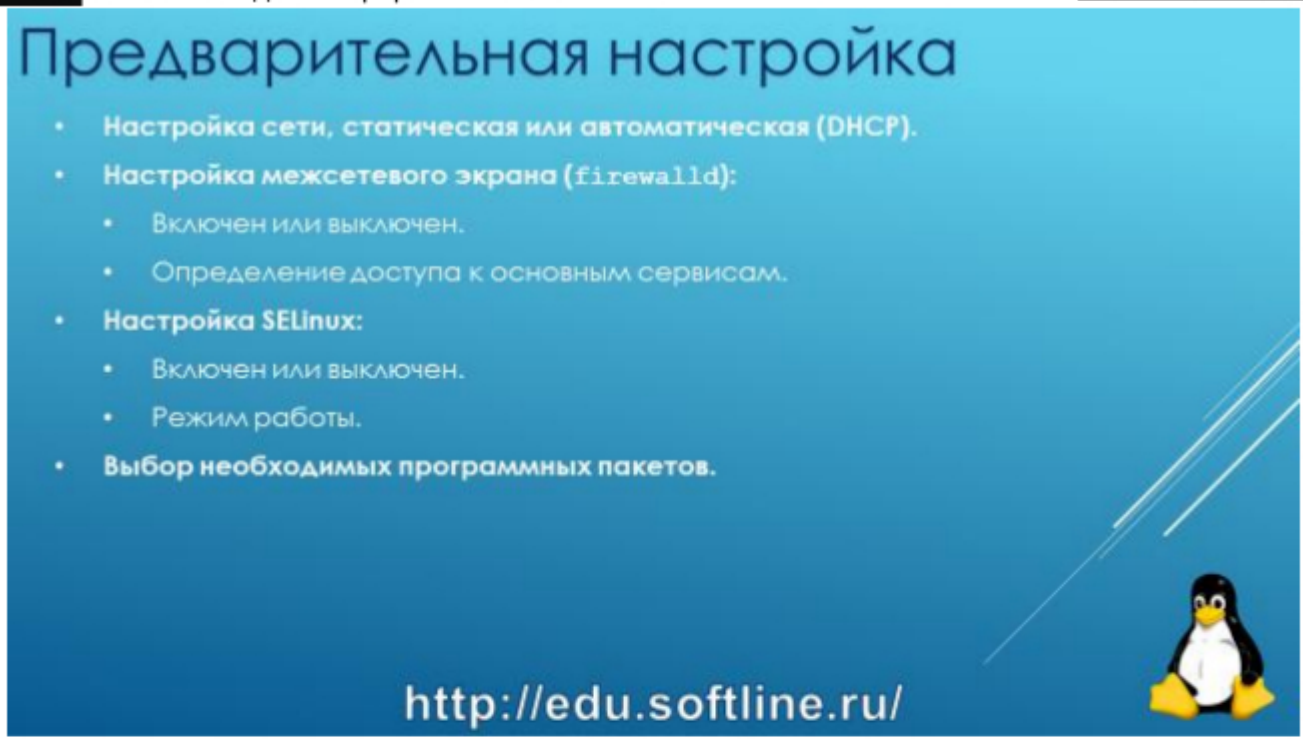
Помимо RAID-массивов существует система LVM (Logical Volume Manager), которая объединяет несколько разделов в один с целью создания логической группы, в которой администратор может создавать логические тома. Использование LVM делает возможным выполнения таких операций, как изменение размера тома, создания снимков. Далее мы более подробно рассмотрим обе технологии. Графический режим установки системы позволяет администратору не только создавать простые разделы, но и организовать программный RAID-массив или LVM-группы.

⚠ *ПРИМЕЧАНИЕ:*
Начиная с CentOS 6 по умолчанию предлагается установка на LVM.

Для создания раздела необходимо выбрать физическое устройство и нажать кнопку создания нового раздела. Установщик предложит указать следующие параметры:

- Точка монтирования. Каталог, в иерархии файловой системы, к которому будет подключен раздел, достаточно указать корневой раздел - **/**.

- Тип файловой системы. По умолчанию используется файловая система **xfs.** Помните, что у раздела **swap** нет точки монтирования, чтобы его создать, необходимо выбрать в качестве файловой системы – **swap**.
- Размер файловой системы.



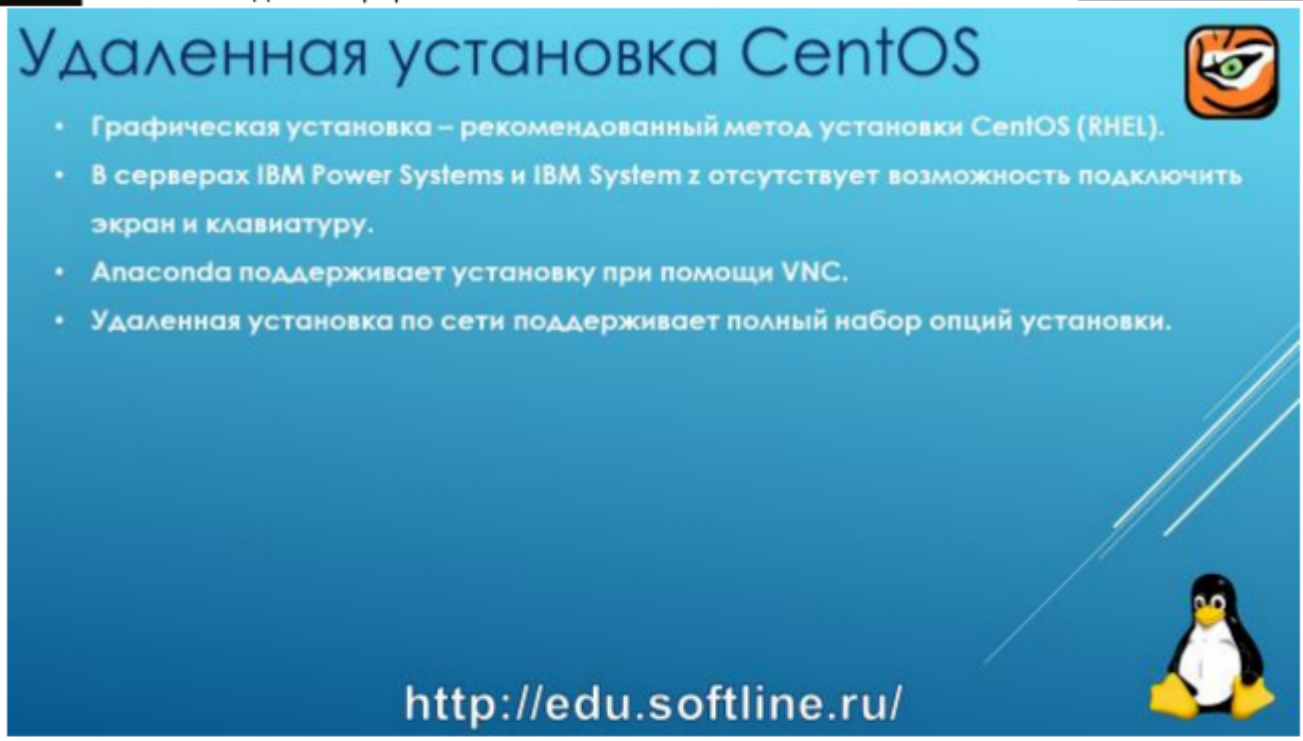
Вы можете при установке указать сетевые реквизиты, либо указать на использование DHCP для автоматического получения сетевых реквизитов.

Программа установки предложит включить или выключить системный брандмауэр. Если пользователь указал на включение, то будет предложено указать минимальные настройки:

- Разрешить весь сетевой трафик для определённого интерфейса.
- Разрешить входящий трафик для определённых сервисов.

SELinux – Secure Enhanced Linux, система управления доступом, основанная на концепции MAC (Mandatory Access Control). В отличие от традиционной дискреционной системы контроля доступа, когда пользователь определяет уровень доступа к объекту, SELinux обеспечивает контроль на уровне системных политик, которые описывают, кто и какие действия может совершать над тем или иным объектом. Установщик предлагает на выбор три режима работы SELinux:

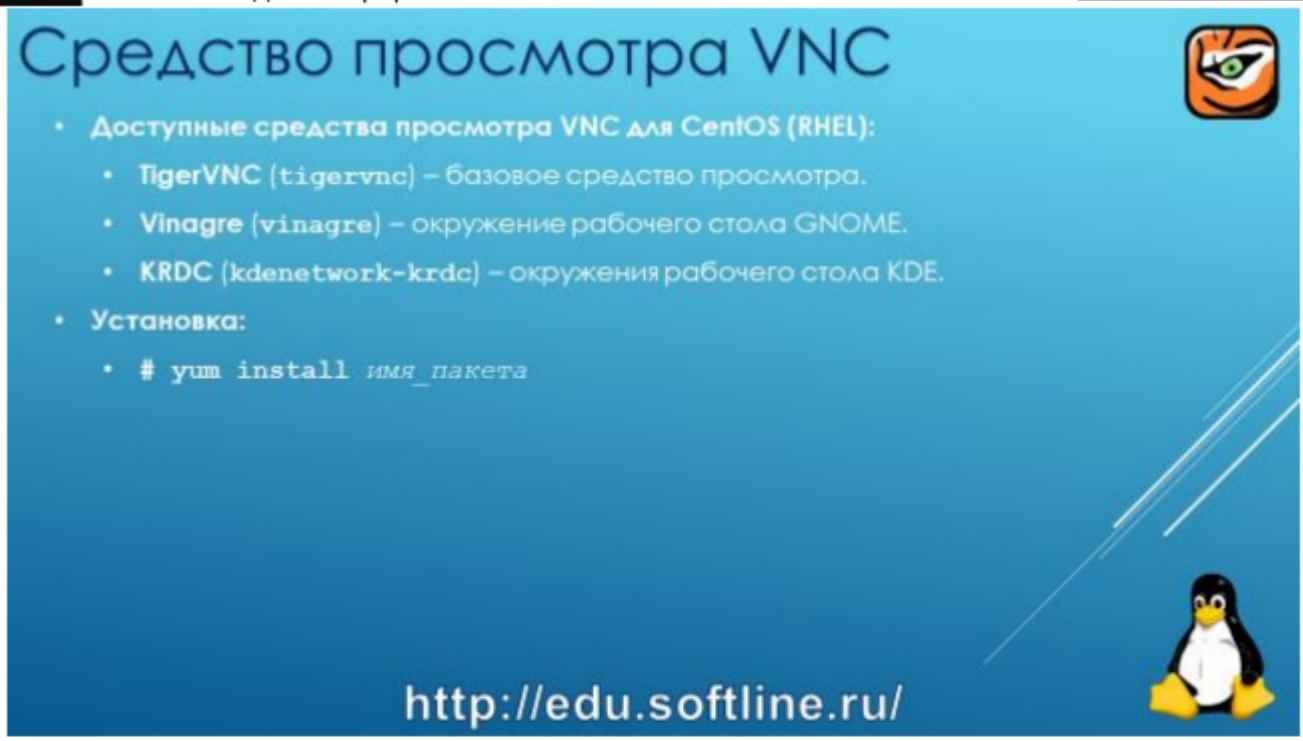
- Disabled – отключен.
- Permissive – включён, но производимые ограничения только регистрируются в журнальных файлах, не внося реальных ограничений в действия пользователей.
- Enforcing – SELinux полностью включён и применяет системные политики по их прямому назначению.



Установка при помощи VNC.

Графическая установка – рекомендованный метод установки CentOS (RHEL). Однако, в некоторых ситуациях, получение прямого доступа к графическому интерфейсу сопряжено с трудностями или невозможно. Множество корпоративных систем, особенно в серверах (IBM Power Systems и IBM System z) отсутствует возможность подключить экран и клавиатуру, что делает установку при помощи VNC обязательной, если не используется автоматическая установка (Kickstart).

Чтобы разрешить установку на «бездисплейной» системе (Headless Systems), программа установки Anaconda поддерживает установку при помощи Virtual Network Computing (VNC), которая позволяет графической установке запуститься локально и при этом отображаться на системах, подключенных к сети. Установка по сети поддерживает полный набор опций установки, в том числе для систем без поддержки экранов и устройств ввода.



Установка средства просмотра VNC.

Выполнение установки по VNC требует запущенного средства просмотра VNC на сторонней системе. Средство просмотра VNC доступно в репозиториях большинства дистрибутивов Linux. Бесплатное средство просмотра VNC также доступно и под другие операционные системы, например, Windows.

Следующие средства просмотра доступны для CentOS (RHEL):

- TigerVNC – базовое средство просмотра, не зависит от окружения рабочего стола. Устанавливается при помощи пакета **`tigervnc`**.
- Vinagre – средство просмотра для окружения рабочего стола GNOME. Устанавливается при помощи пакета **`vinagre`**.
- KRDC – средство просмотра для окружения рабочего стола KDE. Устанавливается при помощи пакета **`kdenetwork-krdc`**.

Для установки любого из перечисленных средств просмотра, воспользуйтесь следующей командой, запущенной от имени пользователя root:

```
# yum install имя_пакета
```

ПРИМЕЧАНИЕ:
В описанных далее процедурах будет использоваться TigerVNC.



Выполнение установки по VNC.

Программа установки Anaconda предлагает два режима установки по VNC. Это прямой режим (Direct Mode) и режим подключения (Connect Mode). В прямом режиме (Direct Mode) требуется средство просмотра VNC для инициации подключения к системе, на которой будет выполняться установка. Режим подключения (Connect Mode) требует инициации подключения к средству просмотра со стороны системы, на которую выполняется установка. После того, как подключение установлено оба режима не имеют различий. Режим, который вы выберите зависит от параметров вашего окружения.

Прямой режим (Direct Mode).

В данном режиме, Anaconda настроена на запуск установки и ждет средства просмотра VNC. IP-адрес и порт отображаются на системе на которую будет произведена установка. При помощи этой информации вы можете подключиться к устанавливаемой системе с другого компьютера. В данном случае вы должны иметь визуальный и интерактивный доступ к системе, на которую производится установка.

Режим подключения (Connect Mode).

В данном режиме, средство просмотра VNC запускается на удаленной системе в режиме прослушивания (Listening Mode). Средство просмотра VNC ожидает входящего подключения на указанном порту. Затем, Anaconda запускается, имя хоста и номер порта предоставляются при помощи опций загрузки или команды автоматической установки (Kickstart). Когда установка начинается, программа установки устанавливает соединения с прослушивающим средством просмотра VNC при помощи указанных имени хоста и номера порта. Для этой цели, удаленная система должна иметь возможность принимать входящие сетевые подключения.

Рекомендации по выбору режима установки VNC.

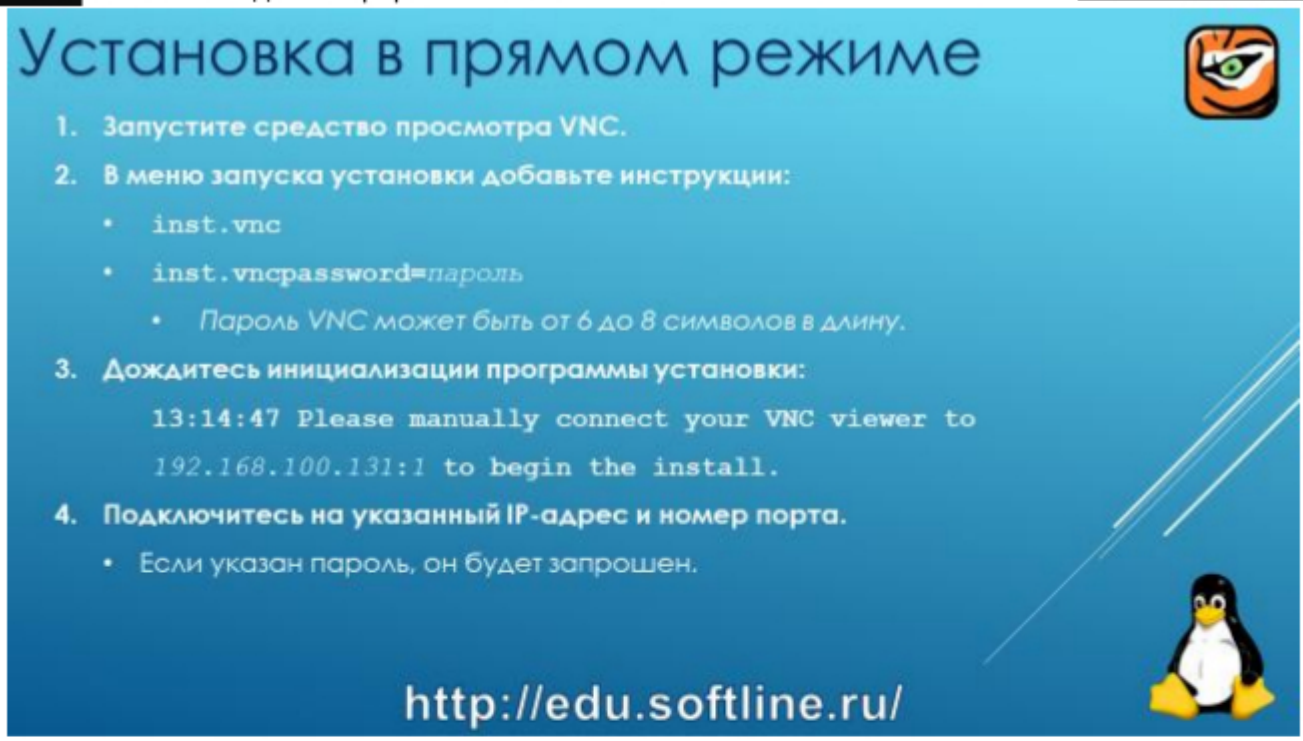
- Графически и интерактивный доступ к системе.
  - Если графический и интерактивный доступ к системе на которой производится установка отсутствует, в таком случае доступен только режим подключения (Connect Mode).

- Правила сетевых подключения и межсетевые экраны.
    - Если система, на которую производится установка не разрешает входящих подключений из-за межсетевого экрана, необходимо выбрать режим подключения (Connect Mode) или отключить межсетевой экран.
    - Если удаленная систем, на которой запущено средство просмотра VNC не разрешает входящих подключений, необходимо использовать прямой режим (Direct Mode) или отключить межсетевой экран.



ПРИМЕЧАНИЕ:

Для запуска установки с использованием VNC, необходимо указать опции загрузки. Правильное решение этой задачи зависит от архитектуры системы, так как опции загрузочного меню для систем на базе AMD64 и Intel 64 могут отличаться от серверов IBM Power Systems.

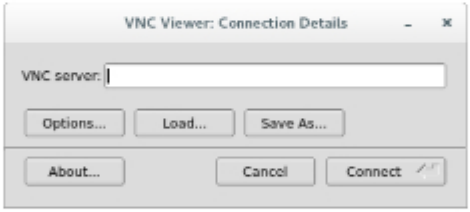


Установка в прямом режиме VNC (Direct Mode).

В прямом режиме (Direct Mode) используется средство просмотра VNC для инициации подключения к устанавливаемой системе. Anaconda запрашивает инициацию этого подключения.

Процедура запуска установки по VNC в прямом режиме (Direct Mode):

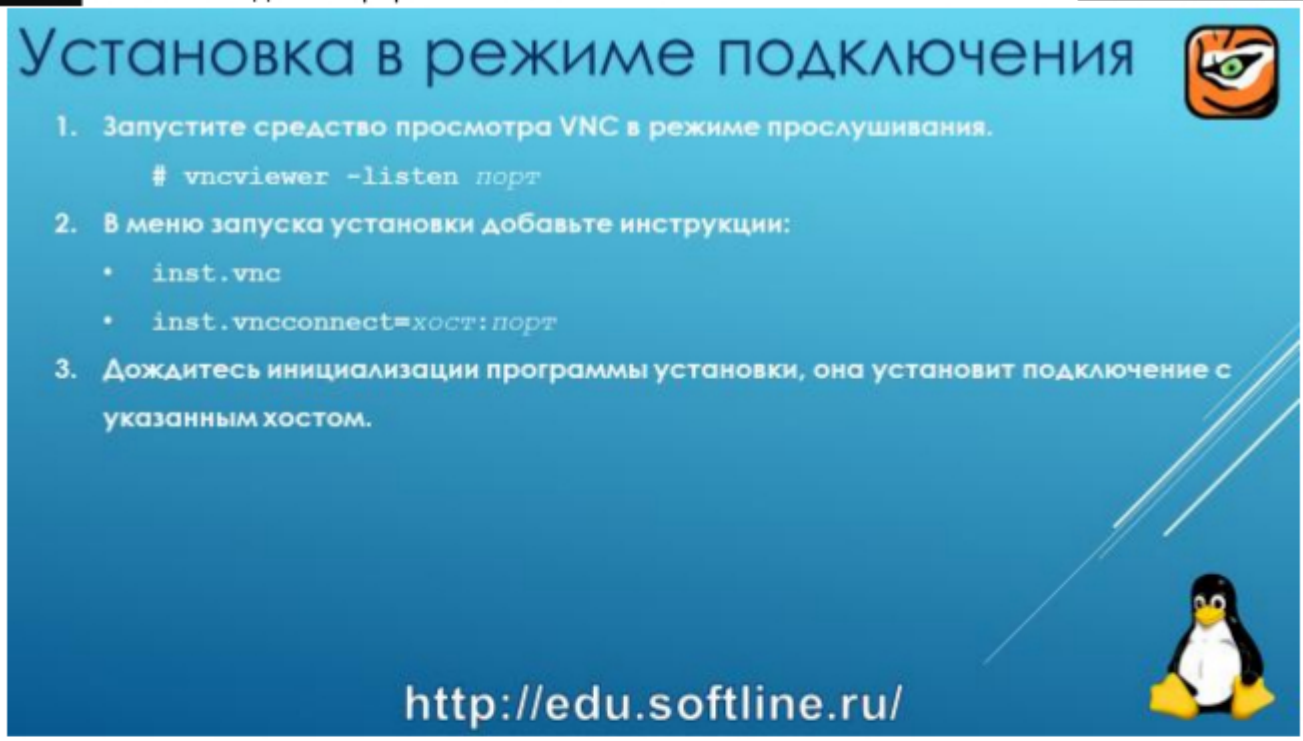
1. Запустите средство просмотра VNC. Вы получите окно со строкой, позволяющей указать IP-адрес.



2. Загрузите систему, на которую будет выполняться установка и дождитесь отображения загрузочного меню. В меню нажмите клавишу **<Tab>** для редактирования опций загрузки. Добавьте опцию **`inst.vnc`** в конец командной строки. Опционально, можно ограничить доступ к VNC на устанавливаемой системе добавив опцию загрузки **`inst.vncpassword=`***`пароль`*. Пароль VNC может быть от 6 до 8 символов в длину.
3. Нажмите клавишу **<Enter>** для запуска установки. Система инициализирует программу установки и запустит необходимые службы. Когда система будет готова, будет выведено следующее сообщение:

```
13:14:47 Please manually connect your VNC viewer to
192.168.100.131:1 to begin the install.
```

4. На системе с запущенным средством просмотра VNC, введите IP-адрес и номер порта, полученные на предыдущем шаге. Если вы указали пароль, после установки соединения он будет запрошен.



Установка в режиме подключения VNC (Connect Mode).

В режиме подключения (Connect Mode) система, на которой производится установка инициирует подключение к удаленной системе на которой запущено средство просмотра VNC. Прежде чем приступать к установке, необходимо убедиться, что удаленная система настроена на прием входящих подключений на порту, который определен для VNC.

Процедура запуска установки по VNC в режиме подключения (Connect Mode):

1. Запустите средство просмотра VNC в режиме прослушивания. Для TigerVNC команда выглядит следующим образом:

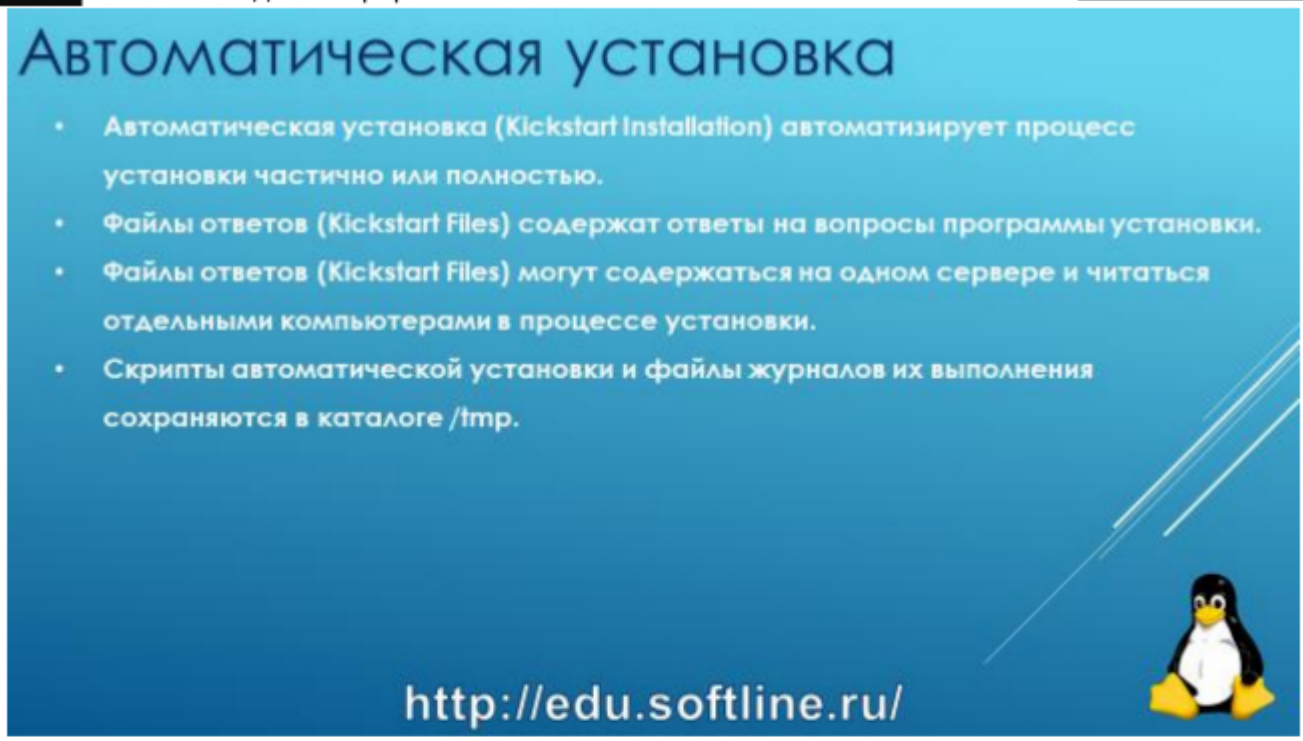
```
# vncviewer -listen порт
```

Терминал отобразит следующее сообщение:

```
TigerVNC Viewer 64-bit v1.3.0 (20130924)
Built on Sep 24 2013 at 16:32:56
Copyright (C) 1999-2011 TigerVNC Team and many others (see README.txt)
See http://www.tigervnc.org for information on TigerVNC.
Thu Feb 20 15:23:54 2014
main: Listening on port 5901
```

Теперь средство просмотра VNC готово и ожидает входящего подключения о устанавливаемой системы.

2. Загрузите систему, на которую будет выполняться установка и дождитесь отображения загрузочного меню. В меню нажмите клавишу **<Tab>** для редактирования опций загрузки. Добавьте опцию **inst.vnc** и **inst.vncconnect=***хост*:*порт* в конец командной строки.
3. Нажмите клавишу **<Enter>** для запуска установки. Система инициализирует программу установки и запустит необходимые службы. Когда система будет готова, она установит подключение с указанным хостом.



Описание автоматической установки (Kickstart Installation).

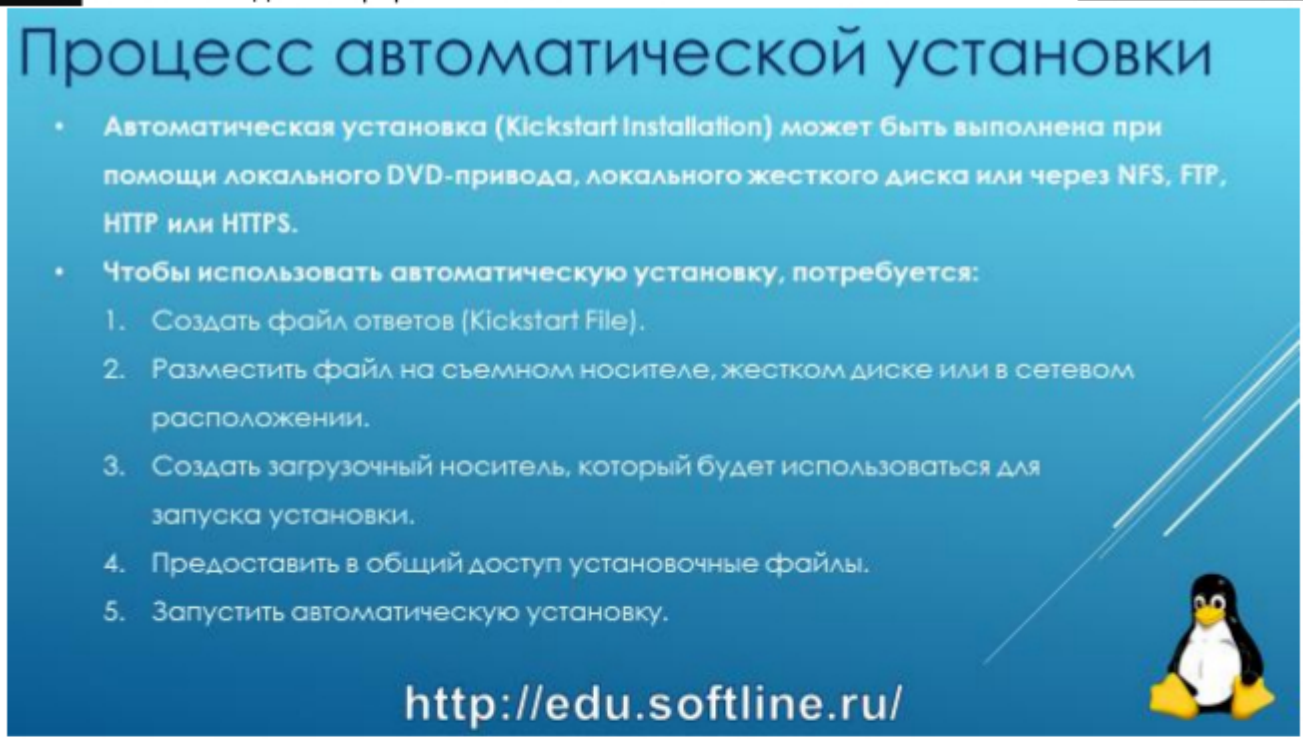
Автоматическая установка (Kickstart Installation) автоматизирует процесс установки частично или полностью. Файлы автоматической установки содержат ответы на все вопросы, которые задает программа установки, такие как выбор временной зоны, разметка разделов или выбор пакетов для установки. Предоставление подготовленного файла ответов в момент запуска установки, позволяет выполнить установку без дальнейшего участия пользователя. Это особенно полезно, если необходимо выполнить одновременную установку CentOS (RHEL) на множестве систем.

Файлы автоматической (Kickstart Files) установки могут содержаться на одном сервере и читаться отдельными компьютерами в процессе установки.

Все скрипты автоматической установки и файлы журналов их выполнения сохраняются в каталоге **`/tmp`**, что помогает отлаживать ошибки установки.

ПРИМЕЧАНИЕ:

В предыдущих версиях CentOS (RHEL), автоматическая установка также позволяла обновлять систему. В CentOS 7 (RHEL 7) этот функционал удален, вместо этого обновление системы производится при помощи специализированных инструментов.

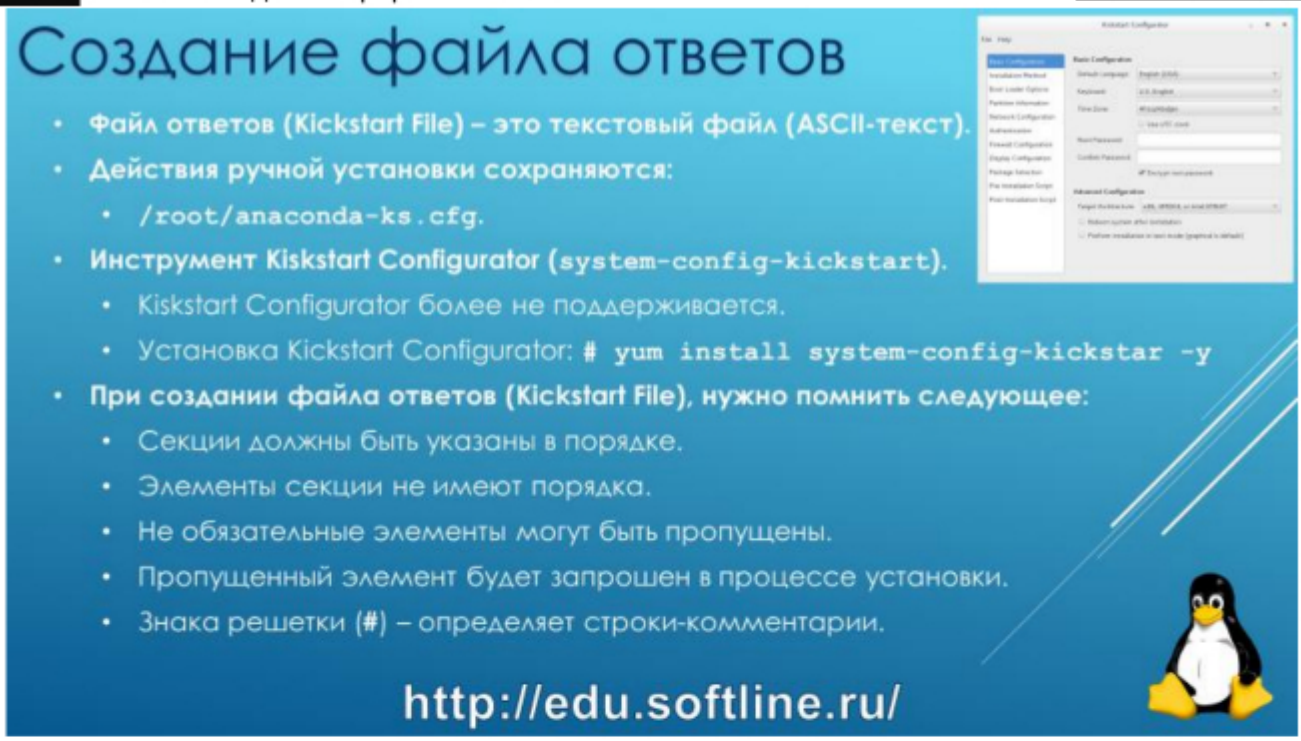


Процесс выполнения автоматической установки (Kickstart Installation).

Автоматическая установка (Kickstart Installation) может быть выполнена при помощи локального DVD-привода, локального жесткого диска или через NFS, FTP, HTTP или HTTPS.

Чтобы использовать автоматическую установку, потребуется:

1. Создать файл ответов (Kickstart File).
2. Разместить файл на съемном носителе, жестком диске или в сетевом расположении.
3. Создать загрузочный носитель, который будет использоваться для запуска установки.
4. Предоставить в общий доступ установочные файлы.
5. Запустить автоматическую установку.



Создание файла ответов (Kickstart File).

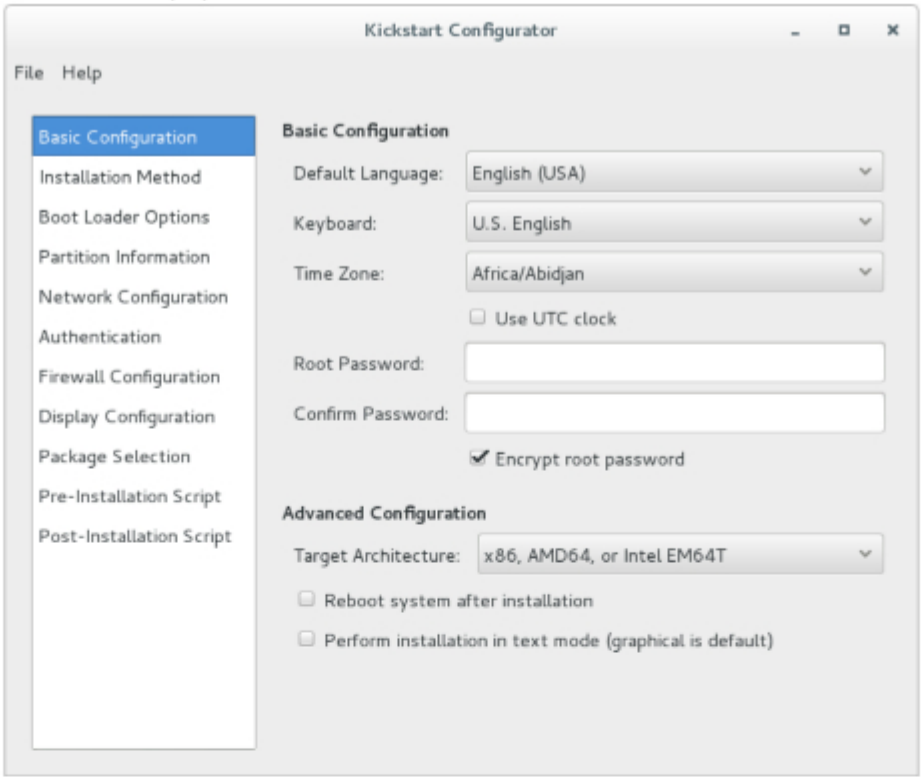
Файл ответов (Kickstart File) – это текстовый файл, содержащий поддерживаемые ключевые слова, которые используются в качестве директив в процессе установки. Любой текстовый редактор, способный сохранить файлы с ASCII-текстом, такой как Gedit или Vim в Linux или Блокнот (Notepad) в Windows, может быть использован для создания файла ответов (Kickstart File). Имя файла, который будет содержать ответы для программы установки не имеет значения, однако рекомендуется использовать простое имя, так как вам придется использовать его позже в конфигурационных файлах или диалоговых окнах.

Рекомендованный подход к созданию файла ответов (Kickstart File) – это выполнить установку вручную на одной из систем. После завершения установки, все выбранные в процессе установки действия будут сохранены в файле **`anaconda-ks.cfg`**, который располагается в каталоге **`/root/`** установленной системы. Вы можете скопировать файл, внести в него любые изменения и использовать для дальнейших установок.

В предыдущих версиях CentOS (RHEL) присутствовал инструмент Kiskstart Configurator (**`system-config-kickstart`**), который позволяет создать файл ответов (Kickstart File) при помощи графического мастера. Но этот инструмент более не поддерживается, поэтому вы можете его установить, но он не содержит в себе изменений в синтаксисе файлов ответов (Kickstart File) которые появились между CentOS 6 (RHEL 6) и CentOS 7 (RHEL 7).

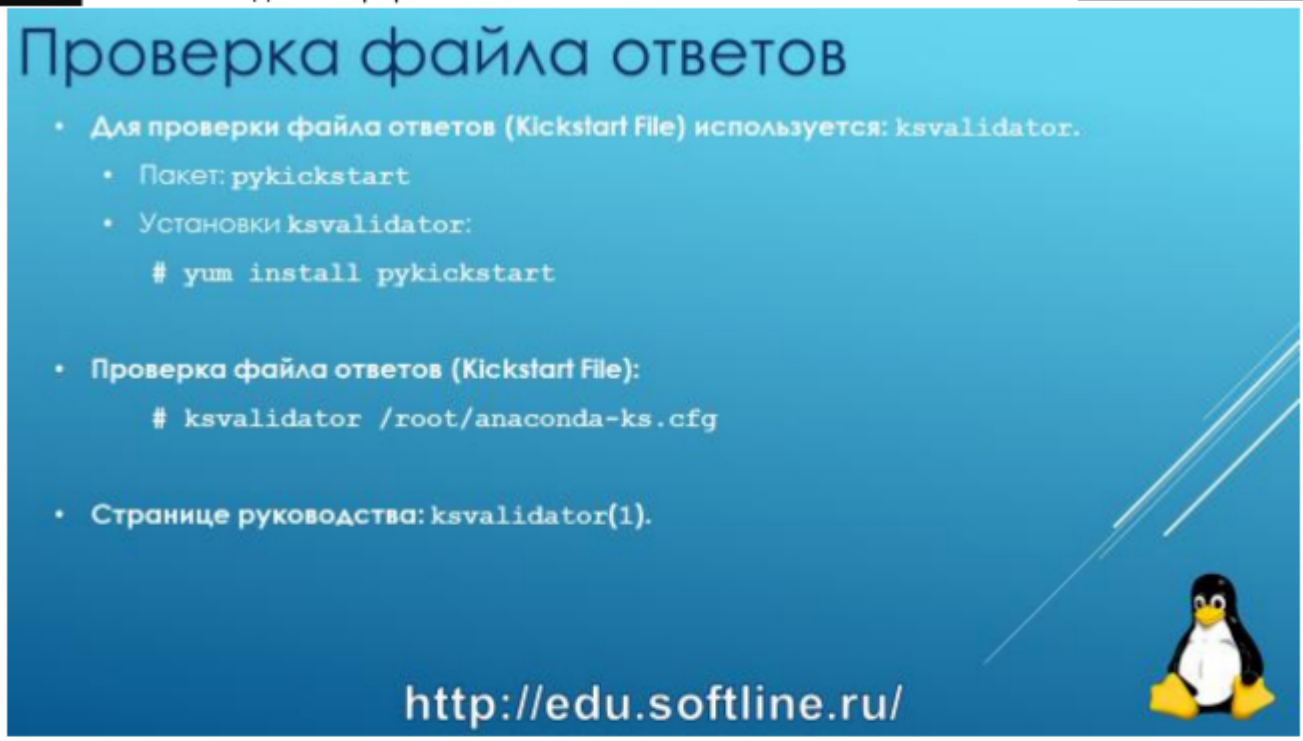
Для установки Kickstart Configurator, воспользуйтесь командой:

```
# yum install system-config-kickstart -y
```



При создании файла ответов (Kickstart File), нужно помнить следующее:

- Секции должны быть указаны в порядке.
- Элементы секции не имеют порядка.
- Не обязательные элементы могут быть пропущены.
- Пропуск обязательного элемента заставит программу установки запросить ответ на нужный вопрос.
- Строки, которые начинаются со знака решетки (#) – воспринимаются как комментарии и игнорируются.



Проверка файла ответов (Kickstart File).

После создания и настройки фала ответов (Kickstart File), полезно проверить файл перед началом выполнения установок с его использованием. CentOS 7 (RHEL 7) содержат утилиту командной строки **ksvalidator**, которая позволяет проверять файлы ответов. Для установки **ksvalidator**, воспользуйтесь следующей командой:

```
# yum install pykickstart
```

Для проверки файла ответов (Kickstart File) воспользуйтесь следующей командой:

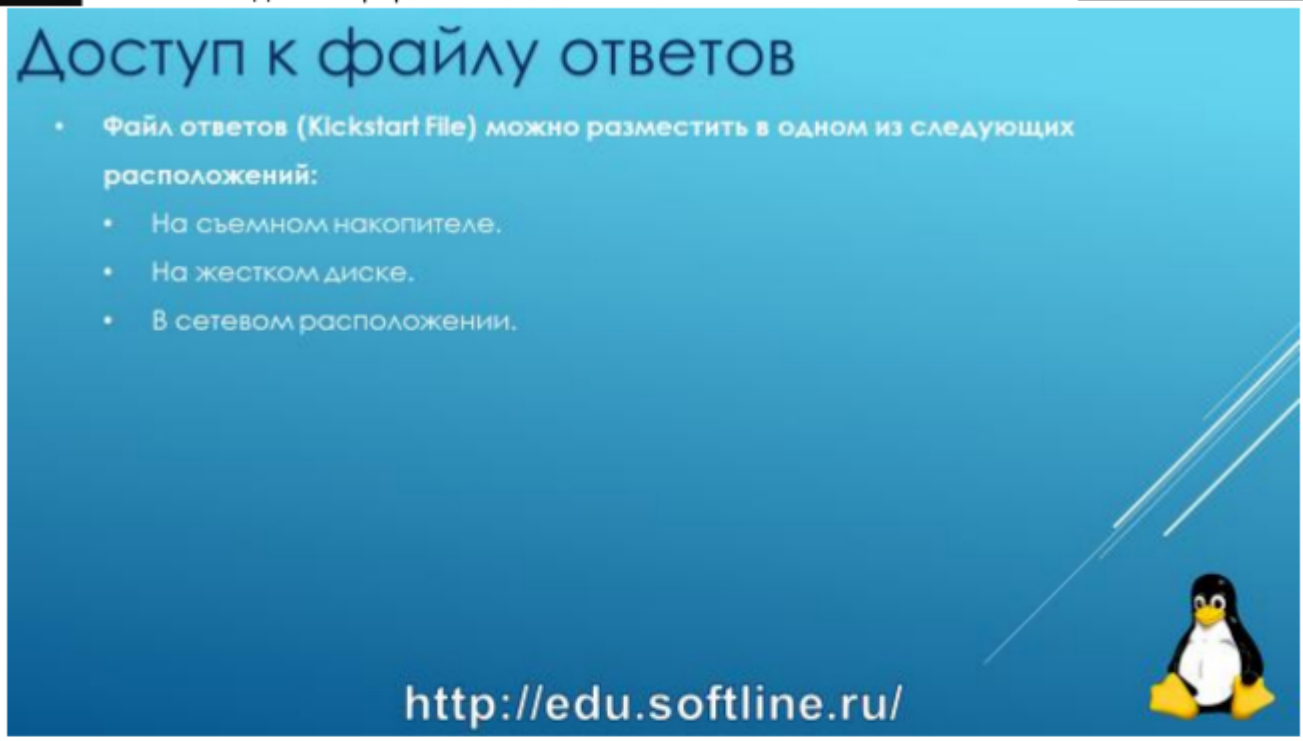
```
[root@c7-client01 ~]# ksvalidator /root/anaconda-ks.cfg
The following problem occurred on line 4 of the kickstart file:

Unknown command: LL-102
```

Для получения дополнительной информации об этом инструменте обратитесь к странице руководства **ksvalidator**(1).



ПРИМЕЧАНИЕ:

Утилита ***ksvalidator*** *имеет ряд ограничений. Файл ответов (Kickstart File) может быть очень сложным,* ***ksvalidator*** *может проверить корректность синтаксиса и отсутствие устаревших опций, но она не может гарантировать успешную установку с использованием этого файла. Также* ***ksvalidator*** *не может проверить секции* ***%pre****,* ***%post*** *и* ***%packsages*** *в файле ответов (Kickstart File).*



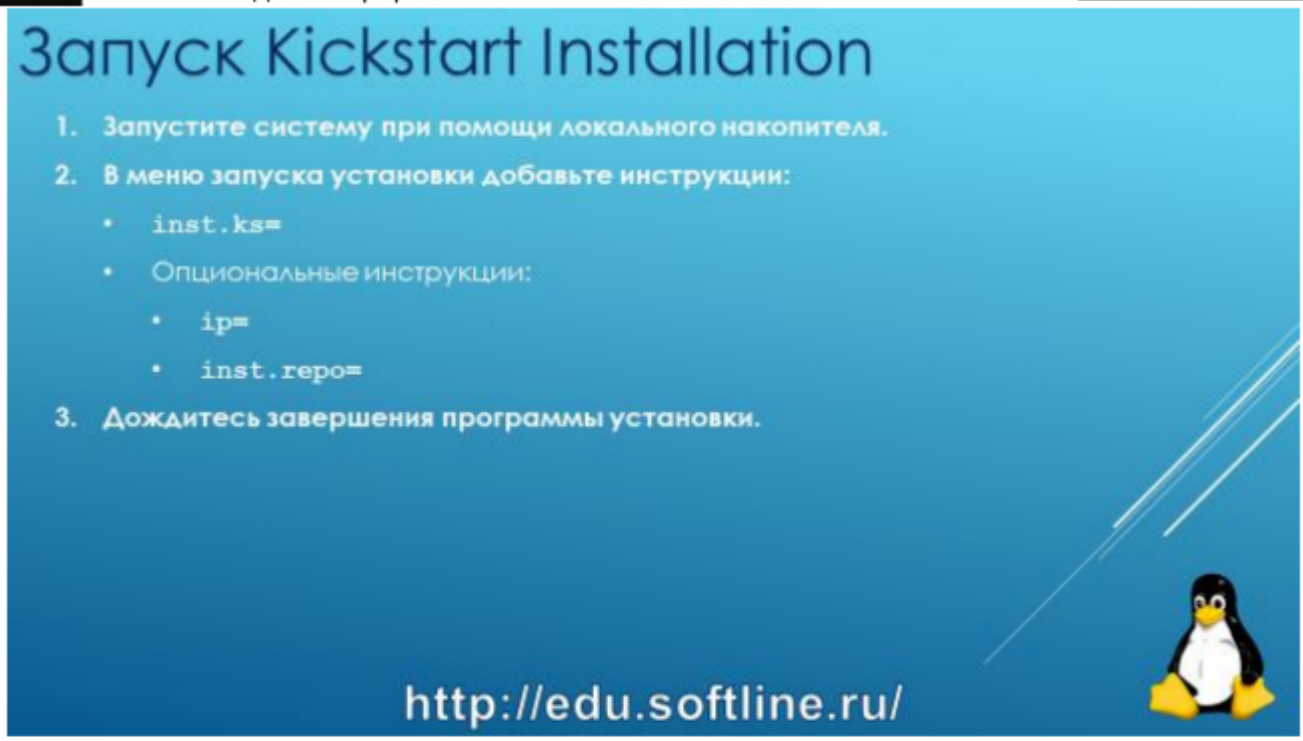
Предоставление файла ответов в доступ (Kickstart File).

Файл ответов (Kickstart File) можно разместить в одном из следующих расположений:

- На съемном накопителе, таком как DVD-диск или флэш устройство USB.
- На жестком диске, подключенном к устанавливаемой системе.
- В доступном для устанавливаемой системы сетевом расположении.

Наиболее часто файл ответов размещают в сетевом расположении, что позволяет его использовать в комплексной установке по сети с использованием PXE-загрузки.

ПРИМЕЧАНИЕ:
Для автоматической загрузки файла ответов (Kickstart File) без указания опции загрузки ***inst.ks=****, файл с именем* ***ks.cfg*** *должен располагаться на тому с меткой* ***OEMDRV****.*



Запуск автоматической установки (Kickstart File).

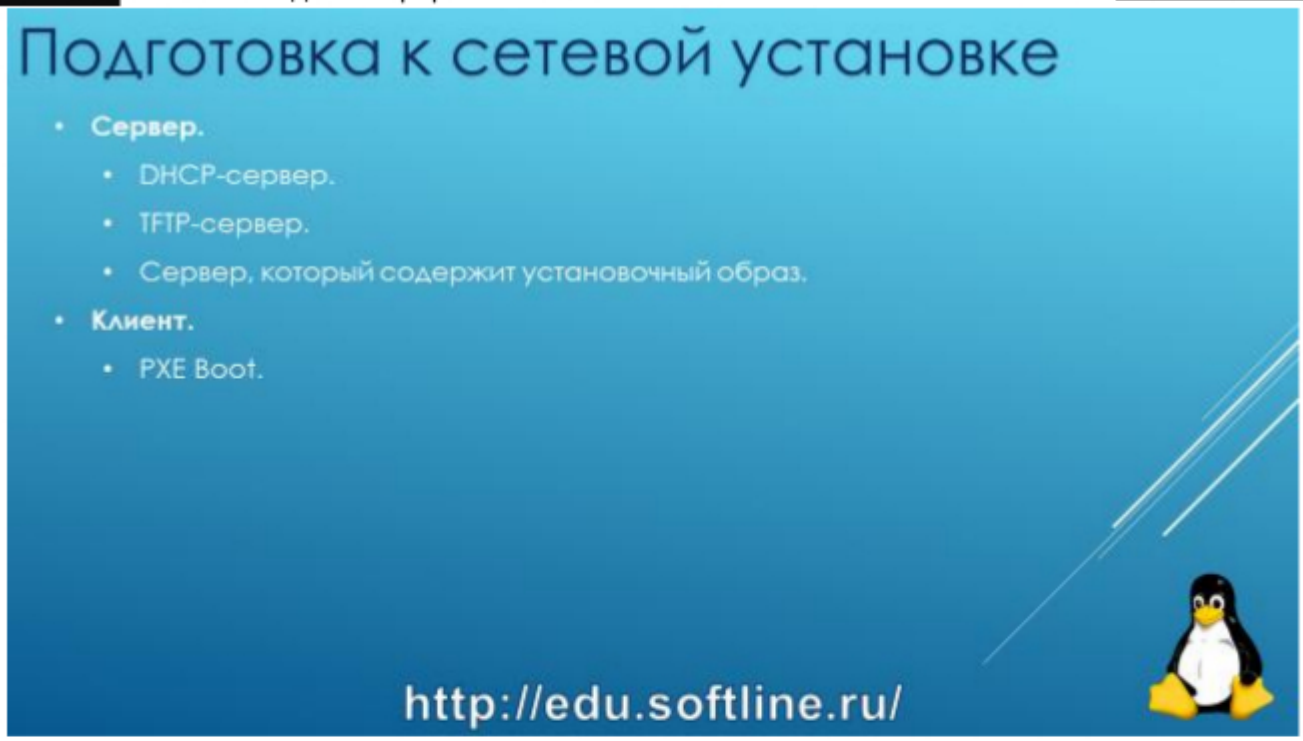
Для запуска автоматической установки, необходимо указать опцию загрузки (**`inst.ks=`**) при запуске программы установки.

Системы AMD64 и Intel 64, а также серверы IBM Power Systems обладают возможность загрузки по сети с использованием PXE-сервера. При настройке PXE-сервера, опции загрузки можно указать в конфигурационном файле, которые позволяет выполнить установку автоматически. Используя этот подход, можно полностью автоматизировать процесс установки, в том числе процесс загрузки.

Запуск процедуры автоматической установки вручную (Kickstart Installation).

Процедура запуска автоматической установки (Kickstart Installation) при помощи опций загрузки (указанных в строке **boot:**), состоит из следующих шагов:

1. Запустите систему при помощи локального накопителя (CD, DVD или USB-накопитель).
2. В строке загрузки укажите опцию загрузки **`inst.ks=`** и расположение файла ответов (Kickstart File). Если файл ответов (Kickstart File) находится в сетевом расположении, необходимо сконфигурировать сети при помощи опции **`ip=`**. В некоторых случаях также необходимо указать расположение источника установочных файлов, при помощи опции **`inst.repo=`**.
3. Запустите процесс установки вместе с добавленными опциями загрузки.



Подготовка к сетевой установке

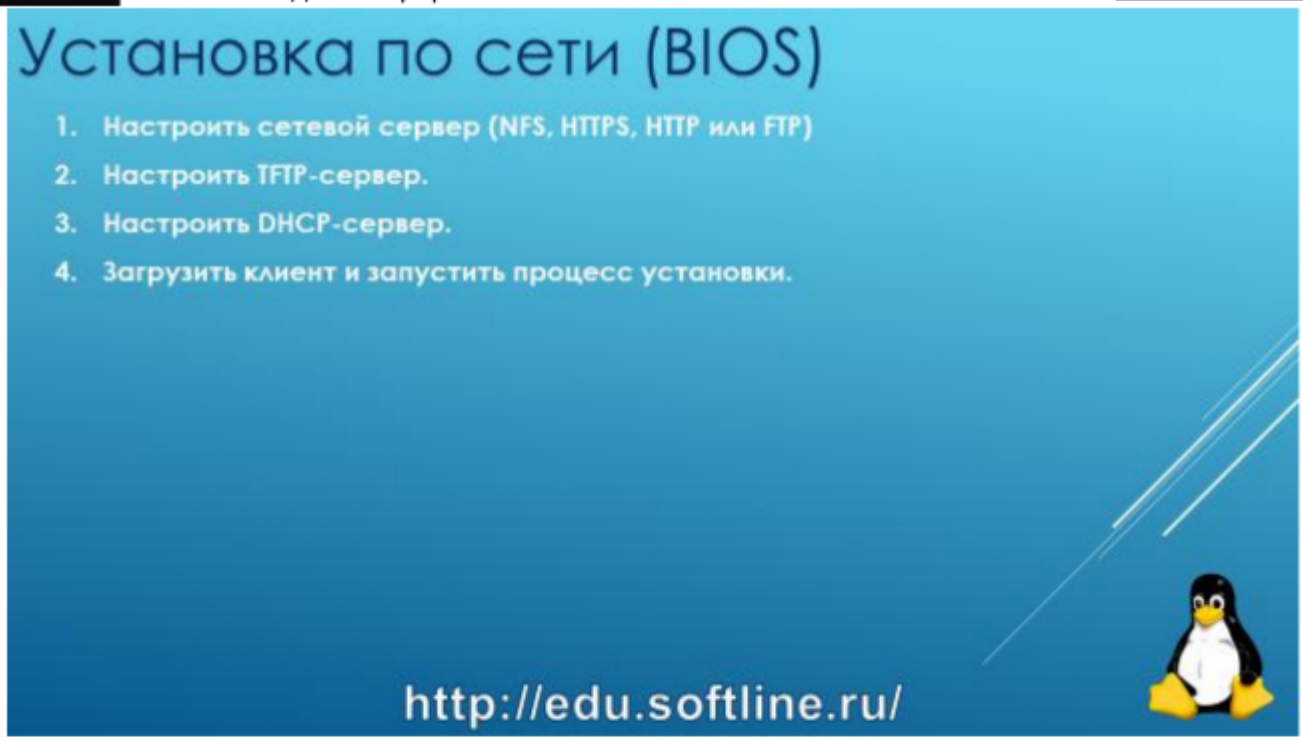
Сетевая установка при помощи сервера установки позволяет вам установить CentOS на множестве систем при помощи сетевой загрузки (Network Boot) сервера. В этом случае все настроенные системы загрузятся при помощи образа, предоставленного сервером и автоматически запустят программу установки.

Требуется минимум два сервера для установки по сети:

- *Сервер* – система, на которой запущены DHCP-сервер, TFTP-сервер для предоставления загрузочных файлов и HTTP, FTP или NFS-сервер, который содержит установочный образ. Теоретически, каждой из серверов может быть запущен на отдельной машине для распределения нагрузок.
- *Клиент* – система на которой необходимо установить CentOS. Когда установка начинается, клиент запрашивает DHCP-сервер, получает загрузочные файлы с TFTP-сервера и загружает установочный образ с HTTP, FTP или NFS-сервера.

⚠ *ПРИМЕЧАНИЕ:*
Клиентская система должна обладать, как минимум 2 Гб RAM для успешной установки по сети.

В отличии от других способов установки, для установки по сети не требуется загрузочный накопитель, подключенный к клиенту (к системе на которую производится установка), для начала процесса установки.



Для подготовки к установке по сети, требуется выполнить следующие шаги:

1. Настроить сетевой сервер (NFS, HTTPS, HTTP или FTP) для экспорта дерева установки или установочного ISO-образа.
2. Настроить файлы на сервере **`tftp`**, необходимые для загрузки по сети, настроить DHCP и запустить службу **`tftp`** на PXE-сервере.

ПРИМЕЧАНИЕ:

Менеджер загрузок GRUB2 поддерживает сетевую загрузку с HTTP в дополнение к серверу **`tftp`**. *Однако, получение загрузочных файлов (ядра и инициирующего диска оперативной памяти для установщика) по этому протоколу очень медленное и может привести к ошибкам ожидания. Рекомендуется использовать сервер* **`tftp`** *для предоставления загрузочных файлов.*

*Это предупреждение применимо только к ядру и начальному диску оперативной памяти (***`vmlinuz`*** и* **`initrd`***). Получение установочных файлов с HTTP-сервера не сопровождается подобными рисками.*

3. Загрузить клиент (систему, на которую производится установка) и запустить процесс установки.

ВНИМАНИЕ:

Процедуры, описанные далее, подходят для настройки сервера загрузки по сети на базе системы под управлением CentOS 7, они имеют ряд отличий от предыдущих версий.

Настройка загрузки по сети.

После настройки сетевого сервера, содержащего репозитории пакетов, которые будут использованы в процессе установки, следующим шагом необходимо настроить PXE-сервер. Этот сервер будет содержать файлы, необходимые для загрузки CentOS и запуска установки. Дополнительно, должен быть сконфигурирован DHCP-сервер и все необходимые службы должны быть включены для автоматического запуска и запущены.

ПРИМЕЧАНИЕ:
Процедура настройки загрузки по сети имеет различия для систем AMD64/Intel 64, в зависимости от использования BIOS или UEFI.

Существует отдельная процедура для загрузки IBP Power Systems по сети при помощи загрузчика GRUB2.

Настройка PXE-сервера для загрузки клиентов AMD64 и Intel 64 на базе BIOS.

Следующая процедура подготовит PXE-сервер к загрузке систем AMD64 и Intel 64 на базе BIOS:

1. Установите пакет **tftp**:

```
# yum install tftp-server
```

2. В конфигурационном файле **/etc/xinetd.d/tftp**, измените значение параметра **disabled** с **yes** на **no**.

Прочие опции, управляющие поведением сервера **tftp**, также доступны в этом конфигурационном файле. Для получения информации о доступных опциях, воспользуйтесь страницей руководства **xinetd.conf(5)**.

3. Разрешите входящие подключения к службе **tftp** в межсетевом экране.

```
# firewall-cmd --add-service=tftp
# firewall-cmd --add-service=tftp --permanent
```

4. Настройте DHCP-сервер на использование загрузочных образов, упакованных при помощи SYSLINUX.

ПРИМЕЧАНИЕ:
Настройка DHCP-сервера выходит за рамки данного курса.

Пример конфигурационного файла DHCP-сервера:

```
default-lease-time 600;
max-lease-time 7200;
log-facility local7;

subnet 10.0.0.160 netmask 255.255.255.224 {
  range 10.0.0.180 10.0.0.189;
  option domain-name-servers 10.0.0.34;
  option domain-name "nerv.local";
  option routers 10.0.0.161;
  option broadcast-address 10.0.0.191;
  default-lease-time 600;
  max-lease-time 7200;
}
  allow booting;
  allow bootp;
  option option-128 code 128 = string;
  option option-129 code 129 = text;
  next-server 10.0.0.179;
  filename "pxelinux.0";
```

5. Теперь необходим файл **pxelinux.0** из пакета **SYSLINUX** в образе ISO полного установочного DVD:

```
# mount -t iso9660 /path/image.iso /mnt_point -o loop,ro
# cp -pr /mnt_point/Packages/syslinux-ver-arch.rpm /directory
# umount /mnt_point
```

```
# rpm2cpio syslinux-ver-arch.rpm | cpio -dimv
```

ПРИМЕЧАНИЕ:

*Альтернативным способом получения этих файла **pxelinux.0** может быть установка пакета **syslinux**:*

 # yum install syslinux -y

*В таком случае файл **pxelinux.0** будет находиться в каталоге **/usr/share/syslinux/***

6. Создайте каталог **pxelinux/** внутри **tftpboot** и скопируйте в него файл **pxelinux.0**:

```
# mkdir /var/lib/tftpboot/pxelinux
# cp directory/usr/share/syslinux/pxelinux.0 /var/lib/tftpboot/pxelinux
# cp directory/usr/share/syslinux/menu.c32 /var/lib/tftpboot/pxelinux
```

7. Создайте каталог **pxelinux.cfg** в каталоге **pxelinux/** и добавьте конфигурационный файл с именем **default** в него:

```
# mkdir /var/lib/tftpboot/pxelinux/pxelinux.cfg
```

Пример конфигурационного файла **/var/lib/tftpboot/pxelinux/pxelinux.cfg/default**:

```
default vesamenu.c32
prompt 1
timeout 600

display boot.msg

label linux
    menu label ^Install system
    menu default
    kernel vmlinuz
    append initrd=initrd.img ip=dhcp
inst.repo=http://xx.xx.xx.xx/mnt/archive/CentOS-7/7.x/Server/x86_64/os/
label vesa
    menu label Install system with ^basic video driver
    kernel vmlinuz
    append initrd=initrd.img ip=dhcp inst.xdriver=vesa nomodeset
inst.repo=http://xx.xx.xx.xx/mnt/archive/CentOS-7/7.x/Server/x86_64/os/
label rescue
    menu label ^Rescue installed system
    kernel vmlinuz
    append initrd=initrd.img rescue
label local
    menu label Boot from ^local drive
    localboot 0xffff
```

ПРИМЕЧАНИЕ:

*Опция **inst.repo=Anaconda**, показанная в предыдущем примере, должна всегда использоваться для указания образа программы установки, также, как и источника установки. Без этой опции программа установки не сможет загрузиться.*

8. Скопируйте загрузочные образы в корневой каталог **tftp/**:

```
# cp path/os/images/pxeboot/{vmlinuz,initrd.img} /var/lib/tftpboot/pxelinux/
```

9. Наконец можно запустить службы **xinetd** и **dhcpd**, если они не были запущены ранее или перезапустить службы **tftp**, **xinetd** и **dhcp**, если они уже запущены:

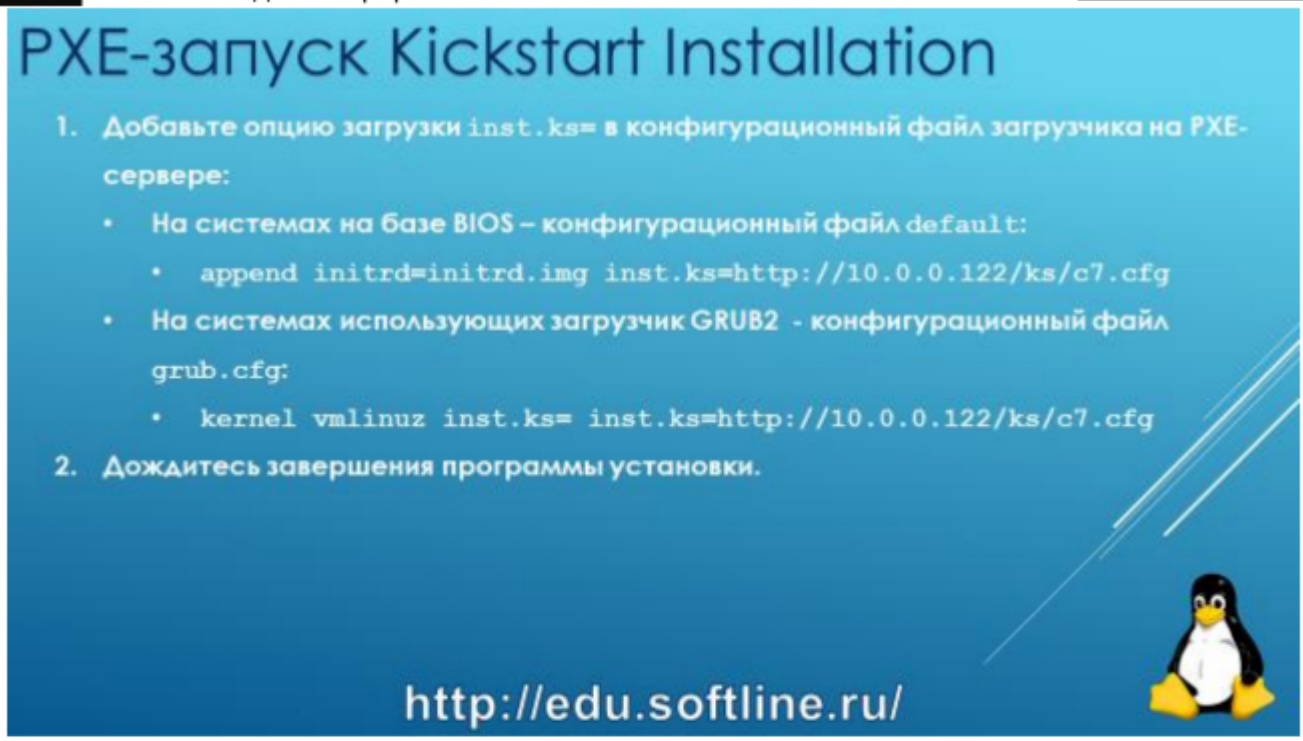
```
# systemctl start xinetd.service dhcpd.service
```

Дополнительно можно настроить автоматический запуск служб при загрузке системы:

```
# systemctl enable xinetd.service dhcpd.service
```

Для перезагрузки конфигурации уже запущенных служб, используйте команду: **`systemctl reload`**.

После завершения этой процедуры PXE-сервер готов к запуску установки по сети. Для этого необходимо запустить систему, на которую вы хотите установить CentOS, выбрать PXE Boot, когда будет запрошено указание источника загрузки и запустить установку по сети.



Автоматический запуск процедуры автоматической установки (Kickstart Installation).

Процедура запуска автоматической установки (Kickstart Installation) при помощи загрузки по сети с использованием PXE-сервера, состоит из следующих шагов:

1. Откройте конфигурационный файл загрузчика на PXE-сервере и добавьте опцию загрузки inst.ks= в необходимую строку. Имя конфигурационного файла загрузчика зависит от архитектуры и оборудования системы:
   - На системах AMD64 и Intel 64 на базе BIOS используйте конфигурационный файл **`default`**:

     `append initrd=initrd.img inst.ks=http://10.0.0.122/ks/c7.cfg`

   - На системах, использующих загрузчик GRUB2 (AMD64 и Intel 64 на базе UEFI и серверы IBM Power Systems) используйте конфигурационный файл **`grub.cfg`**:

     `kernel vmlinuz inst.ks=http://10.0.0.122/ks/c7.cfg`

2. Запустите установку по сети с измененной конфигурацией.



Настройка PXE-сервера для загрузки клиентов AMD64 и Intel 64 на базе UEFI.

Следующая процедура подготовит PXE-сервер к загрузке систем AMD64 и Intel 64 на базе UEFI:

1. Установите пакет **tftp**:

```
# yum install tftp-server
```

2. В конфигурационном файле **/etc/xinetd.d/tftp**, измените значение параметра **disabled** с **yes** на **no**.

Прочие опции, управляющие поведением сервера **tftp**, также доступны в этом конфигурационном файле. Для получения информации о доступных опциях, воспользуйтесь страницей руководства **xinetd.conf(5)**.

3. Разрешите входящие подключения к службе **tftp** в межсетевом экране.

```
# firewall-cmd --add -service=tftp
# firewall-cmd --add -service=tftp --permanent
```

4. Настройте DHCP-сервер на использование загрузочных образов, упакованных при помощи SYSLINUX.

ПРИМЕЧАНИЕ:
Настройка DHCP-сервера выходит за рамки данного курса.

Пример конфигурационного файла DHCP-сервера, для систем на базе BIOS и UEFI:

```
option space pxelinux;
option pxelinux.magic code 208 = string;
option pxelinux.configfile code 209 = text;
option pxelinux.pathprefix code 210 = text;
option pxelinux.reboottime code 211 = unsigned integer 32;
option architecture-type code 93 = unsigned integer 16;
subnet 10.0.0.0 netmask 255.255.255.0 {
   option routers 10.0.0.254;
   range 10.0.0.2 10.0.0.253;
   class "pxeclients" {
     match if substring (option vendor-class-identifier, 0, 9) = "PXEClient";
```

```
        next-server 10.0.0.1;
        if option architecture-type = 00:07 {
              filename "uefi/shim.efi";
        } else {
              filename "pxelinux/pxelinux.0";
        }
    }
}
```

5. Теперь необходим файл **shim.efi** из пакета **shim** и файл grubx64.efi из пакета grub2.efi с образа ISO:

```
# mount -t iso9660 /path/image.iso /mnt_point -o loop,ro
# cp -pr /mnt_point/Packages/shim-ver-arch.rpm /directory
# cp -pr /mnt_point/Packages/grub2-efi-ver-arch.rpm /directory
# umount /mnt_point
# rpm2cpio shim-ver-arch.rpm | cpio -dimv
# rpm2cpio grub2-efi-ver-arch.rpm | cpio -dimv
```

6. Создайте каталог **uefi/** внутри **tftpboot** для образов загрузки EFI:

```
# mkdir /var/lib/tftpboot/uefi
# cp directory/usr/share/shim/x64-ver.el7.centos/shim.efi /var/lib/tftpboot/uefi/
# cp directory/boot/efi/EFI/centos/grubx64.efi /var/lib/tftpboot/uefi/
```

7. Добавьте конфигурационный файл **grub.cfg** в каталог **uefi/**. Также на системе с CentOS 7 присутствует пример конфигурационного файла: **/var/lib/tftpboot/uefi/grub.cfg**.

Пример конфигурационного файла **grub.cfg**:

```
set timeout=60
    menuentry 'CentOS 7' {
    linuxefi uefi/vmlinuz ip=dhcp
    inst.repo=http://10.32.5.1/mnt/archive/CentOS-7/7.1/Server/x86_64/os/
    initrdefi uefi/initrd.img
}
```

ПРИМЕЧАНИЕ:

*Опция **inst.repo=Anaconda**, показанная в предыдущем примере, должна всегда использоваться для указания образа программы установки, также, как и источника установки. Без этой опции программа установки не сможет загрузиться.*

8. Скопируйте загрузочные образы в корневой каталог **tftp/**:

```
# cp path/os/images/pxeboot/{vmlinuz,initrd.img} /var/lib/tftpboot/uefi/
```

9. Наконец можно запустить службы **xinetd** и **dhcpd**, если они не были запущены ранее или перезапустить службы **tftp**, **xinetd** и **dhcp**, если они уже запущены:

```
# systemctl start xinetd.service dhcpd.service
```

Дополнительно можно настроить автоматический запуск служб при загрузке системы:

```
# systemctl enable xinetd.service dhcpd.service
```

Для перезагрузки конфигурации уже запущенных служб, используйте команду: **systemctl reload**.

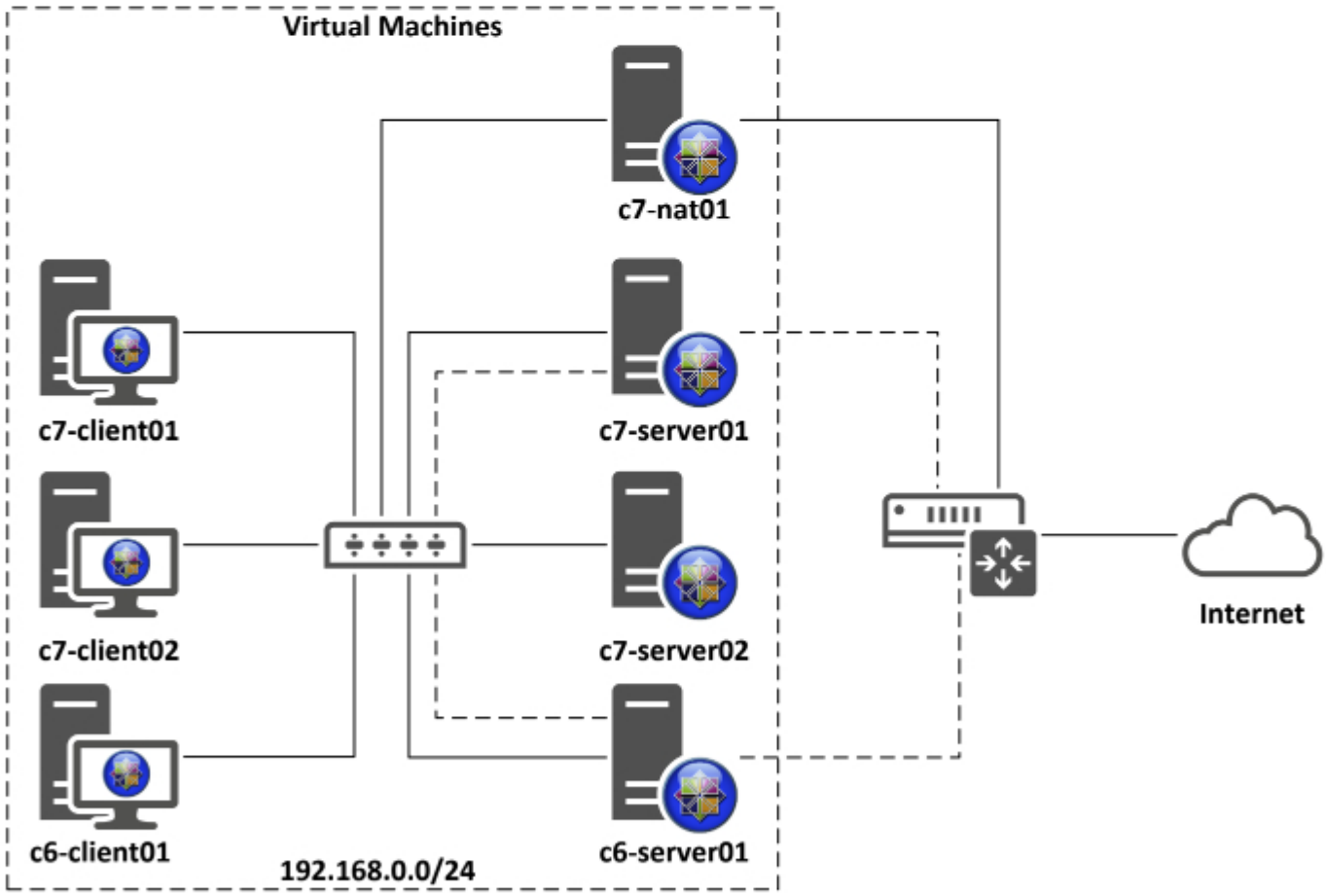
После завершения этой процедуры PXE-сервер готов к запуску установки по сети. Для этого необходимо запустить систему, на которую вы хотите установить CentOS, выбрать PXE Boot, когда будет запрошено указание источника загрузки и запустить установку по сети.

Лабораторная работа

Лабораторная работа 1

- **Упражнения:**
  - Установка CentOS 7 в графическом режиме.
  - Автоматическая установка CentOS 7.
- **Необходимые машины:**
  - `c7-nat01`.
- **Учетные данные:**
  - Пользователь: `root`.
  - Пароль: `Pa$$w0rd`.

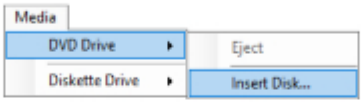
http://edu.softline.ru/





Упражнение 1 – Установка CentOS 7 в графическом режиме.

1. Запустите виртуальную машину c7-clean01.
2. При помощи меню «Media» окна подключения к виртуальной машине смонтируйте ISO-образ с CentOS 7.



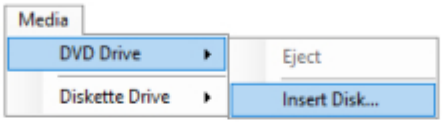
ПРИМЕЧАНИЕ:
Расположение ISO-образа с CentOS 7 уточните у инструктора.

3. Запустите установку CentOS 7 (**Install CentOS 7**) без проверки накопителя.
4. В окне выбора языка оставьте значения по умолчанию (English – United States).
5. Выполните следующую настройку:
   - В разделе DATE & TIME укажите свой часовой пояс (например Asia/Omsk).
   - В разделе INSTALLATION SOURCE выберите локальный накопитель (Local Media).
   - В разделе SOFTWARE SELECTION выберите сервер с графическим интерфейсом (Server with GUI) и дополнительно выберите базу данных MariaDB (MariaDB Database Server).
   - В разделе INSTALLAION DESTINATION выберите ручную настройку (I will configure partitioning).
   - В разделе MANUAL PARTITIONING выберите стандартную схему разметки диска (Standard Partitioning) и создайте следующие разделы:
     - **`/boot 500 MiB`**.
     - **`/home 2048 MiB`**.
     - **`/var 8192 MiB`**.
     - **`swap 4096 MiB`**.
     - **`/`** все что осталось.
   - В разделе KDUMP отключите kdump.
   - В разделе NETWORK & HOST NAME укажите имя хоста (Host name): c7-clean01, а также настройте автоматический запуск интерфейса (убедитесь, что IP-адрес получен).
6. Запустите процедуру установки и в процессе установки задайте следующие настройки:
   - Пароль суперпользователя (root): Pa$$w0rd.
   - Создайте дополнительного пользователя со следующими настройками:
     - Полное имя пользователя (Full name): Student.
     - Имя пользователя (User name): student.
     - Сделайте пользователя администратором (Make this user administrator).
     - Пароль пользователя: Pa$$w0rd.
7. Дождитесь окончания установки и перезагрузите машину.
8. Выполните шаги по начальной настройке системы.
9. По окончании начальной настройки войдите в систему.



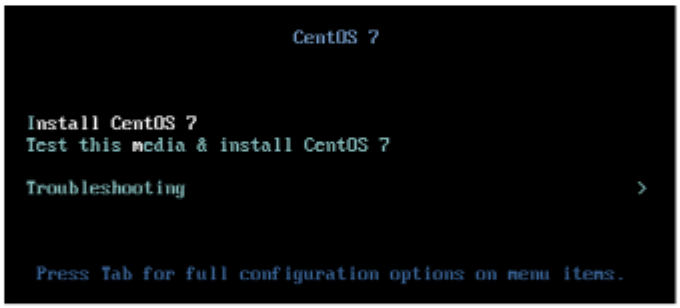
Ответы к упражнению 1 – Установка CentOS 7 в графическом режиме.

1. Запустите виртуальную машину c7-clean01.
2. При помощи меню «Media» окна подключения к виртуальной машине смонтируйте ISO-образ с CentOS 7.

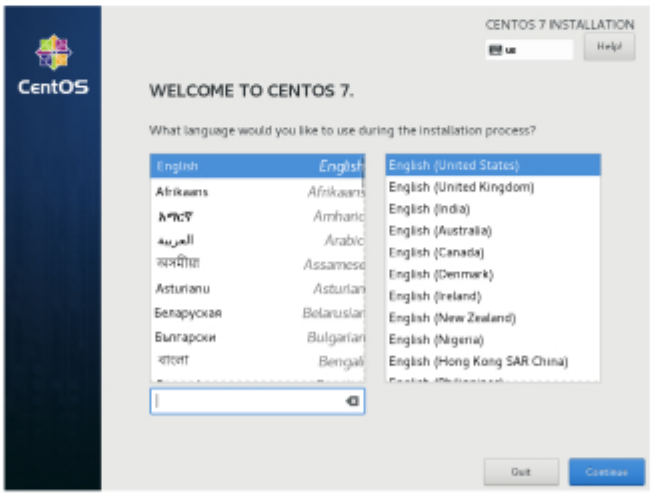


ПРИМЕЧАНИЕ:
Расположение ISO-образа с CentOS 7 уточните у инструктора.

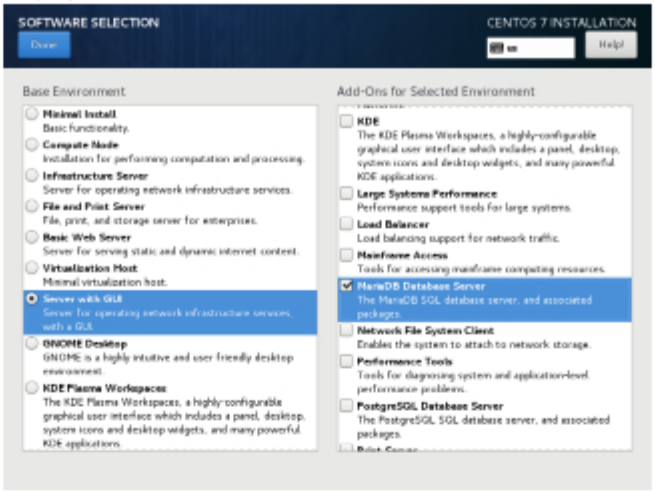
3. Запустите установку CentOS 7 (**Install CentOS 7**) без проверки накопителя.



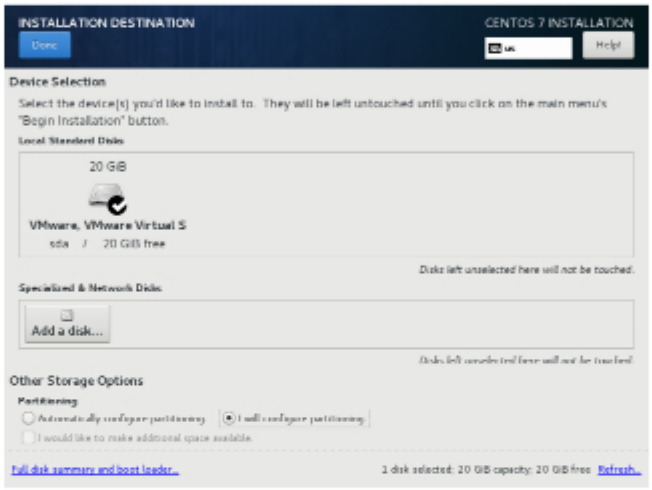
4. В окне выбора языка оставьте значения по умолчанию (English – United States).



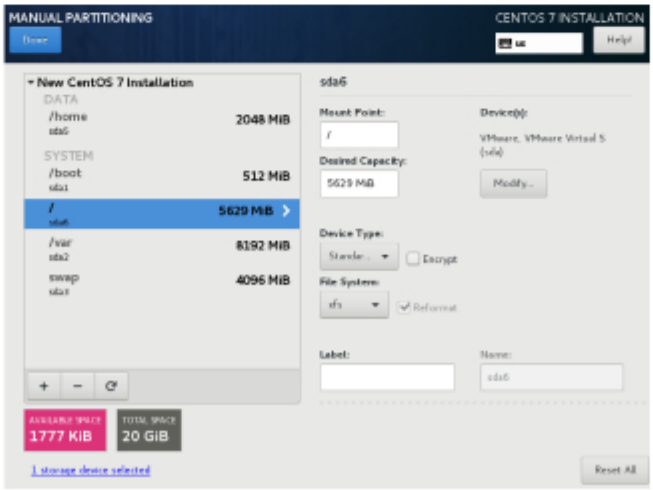
5. Выполните следующую настройку:
   - В разделе DATE & TIME укажите свой часовой пояс (например Asia/Omsk).
   - В разделе INSTALLATION SOURCE выберите локальный накопитель (Local Media).
   - В разделе SOFTWARE SELECTION выберите сервер с графическим интерфейсом (Server with GUI) и дополнительно выберите базу данных MariaDB (MariaDB Database Server).



- В разделе INSTALLAION DESTINATION выберите ручную настройку (I will configure partitioning).

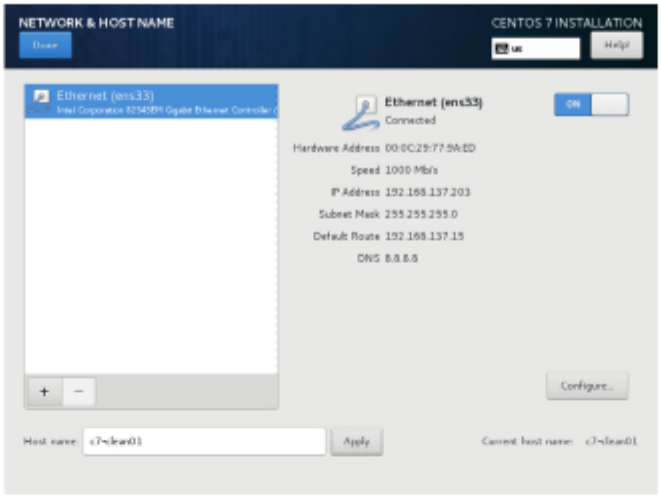


- В разделе MANUAL PARTITIONING выберите стандартную схему разметки диска (Standard Partitioning) и создайте следующие разделы:
  - **`/boot 500 MiB`**.
  - **`/home 2048 MiB`**.
  - **`/var 8192 MiB`**.
  - **`swap 4096 MiB`**.
  - **`/`** все что осталось.

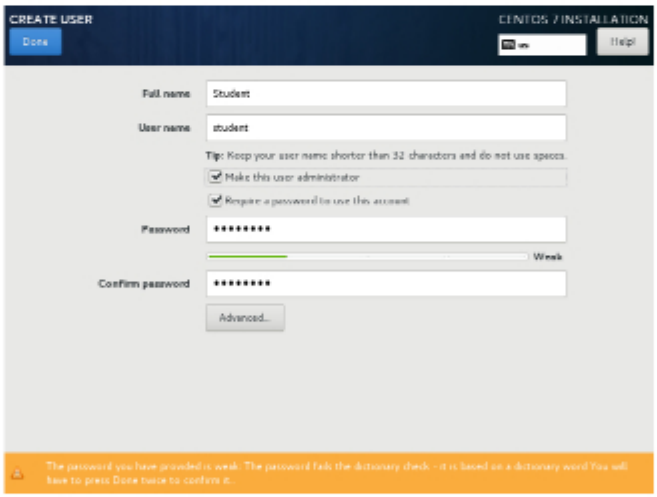


- В разделе KDUMP отключите kdump.

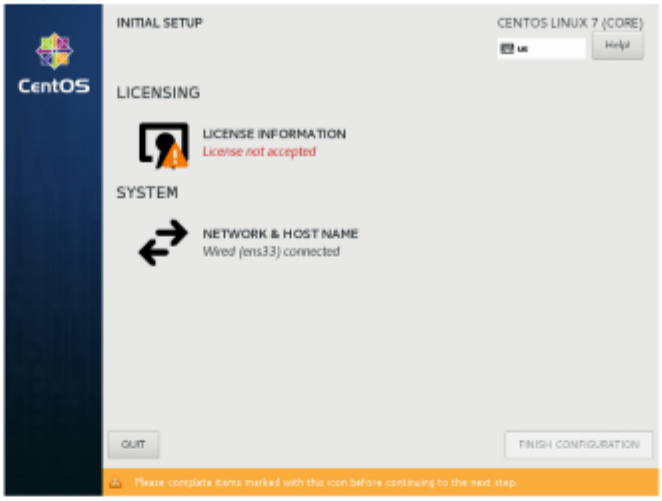
- В разделе NETWORK & HOST NAME укажите имя хоста (Host name): c7-clean01, а также настройте автоматический запуск интерфейса (убедитесь, что IP-адрес получен).



6. Запустите процедуру установки и в процессе установки задайте следующие настройки:
   - Пароль суперпользователя (root): Pa$$w0rd.
   - Создайте дополнительного пользователя со следующими настройками:
     - Полное имя пользователя (Full name): Student.
     - Имя пользователя (User name): student.
     - Сделайте пользователя администратором (Make this user administrator).
     - Пароль пользователя: Pa$$w0rd.



7. Дождитесь окончания установки и перезагрузите машину.
8. Выполните шаги по начальной настройке системы.



9. По окончании начальной настройки войдите в систему.



Упражнение 2 – Автоматическая установка CentOS 7.

Задача 1: Проверка файла ответов.

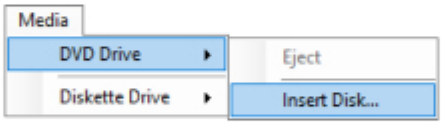
1. Войдите на систему **c7-client01** или **c7-client02**.
2. Установите библиотеку Python для управления файлами автоматической установки (**`pykickstart`**).
3. Загрузите файл **`c7-ks.cfg`**.

> *ПРИМЕЧАНИЕ:*
> *Расположение файла ответов **`c7-ks.cfg`** уточните у инструктора.*

4. Выполните проверку файла ответов.

Задача 2: Установка CentOS 7 с использованием файла ответов.

1. Запустите виртуальную машину **c7-clean02**.
2. При помощи меню «Media» окна подключения к виртуальной машине смонтируйте ISO-образ с CentOS 7.



> *ПРИМЕЧАНИЕ:*
> *Расположение ISO-образа с CentOS 7 уточните у инструктора.*

3. Отредактируйте меню загрузки, указав файл ответов в директиве (**`inst.ks`**).

   … **`inst.ks=http://`***`URL`***`/files/c7-ks.cfg`**

4. Убедитесь, что установщик принял файл ответов и запустил установку.

> *ПРИМЕЧАНИЕ:*
> *Можно не дожидаться окончания процесса установки, а просто остановить/откатить виртуальную машину.*



Ответы к упражнению 2 – Автоматическая установка CentOS 7.

Задача 1: Проверка файла ответов.

10. Войдите на систему c7-client01 или c7-client02.
11. Установите библиотеку Python для управления файлами автоматической установки (**pykickstart**).

```
# yum install pykickstart -y
```

12. Загрузите файл **c7-ks.cfg**.

```
# wget http://URL/files/c7-ks.cfg
```

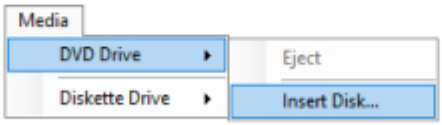
⚠ *ПРИМЕЧАНИЕ:*
*Расположение файла ответов **c7-ks.cfg** уточните у инструктора.*

13. Выполните проверку файла ответов.

```
# ksvalidator c7-ks.cfg
```

Задача 2: Установка CentOS 7 с использованием файла ответов.

1. Запустите виртуальную машину **c7-clean02**.
2. При помощи меню «Media» окна подключения к виртуальной машине смонтируйте ISO-образ с CentOS 7.



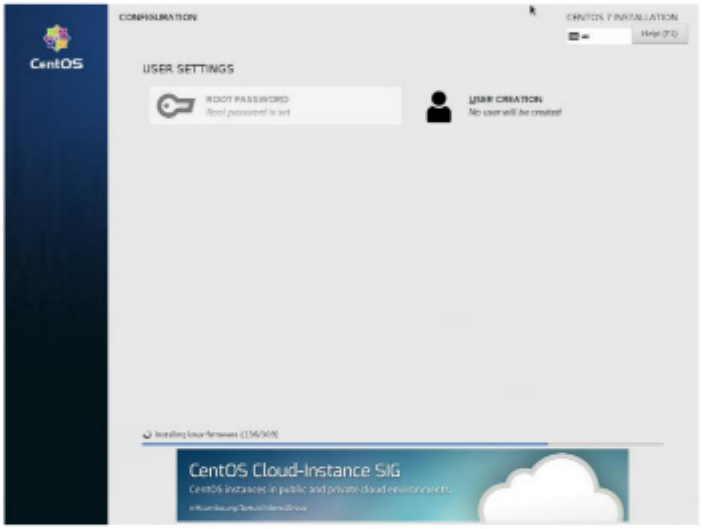
⚠ *ПРИМЕЧАНИЕ:*
Расположение ISO-образа с CentOS 7 уточните у инструктора.

3. Отредактируйте меню загрузки, указав файл ответов в директиве (**inst.ks**).

```
… inst.ks=http://URL/files/c7-ks.cfg
```



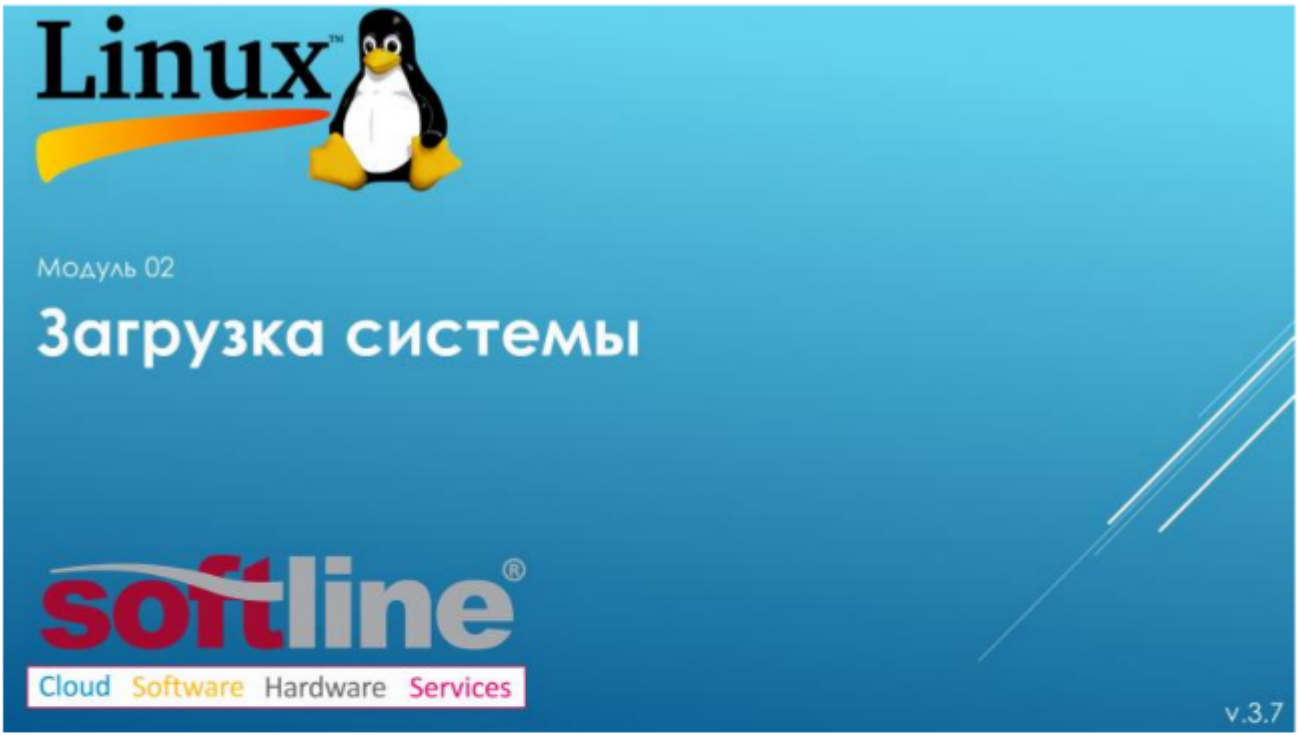
4. Убедитесь, что установщик принял файл ответов и запустил установку.



ПРИМЕЧАНИЕ:

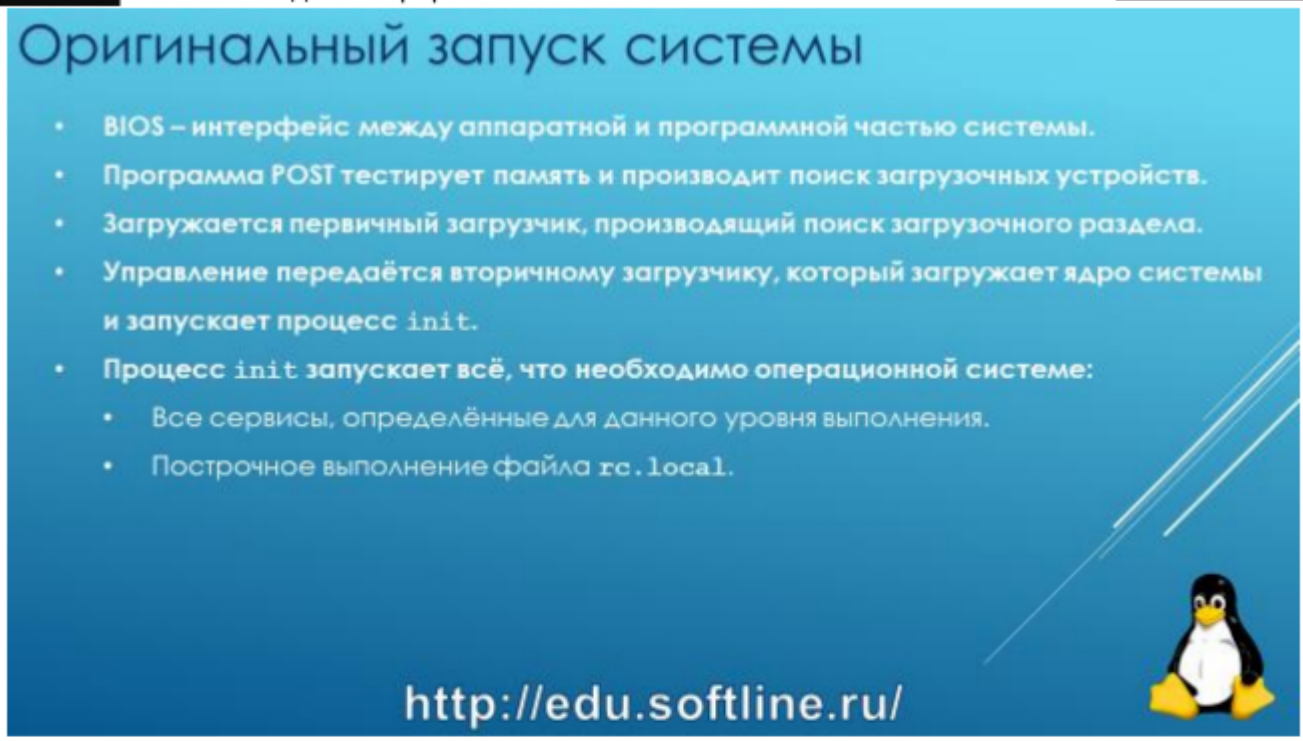
Можно не дожидаться окончания процесса установки, а просто остановить/откатить виртуальную машину.

Модуль 02: Загрузка системы



В модуле представлены:

- Загрузка, выключение, приостановка и гибернация системы.
- Управление загрузчиком GRUB.
- Управление уровнями выполнения.
- Управление загрузчиком GRUB2.
- Парольная защита загрузчика GRUB2.
- Демон инициализации **`systemd`**.
- Управление службами и таргетами **`systemd`**.



BIOS (Basic Input Output System) выполняет роль связующего между основными звеньями аппаратной вычислительной системы. BIOS предоставляет базовые инструкции нижнего уровня, используемые операционной системой.

При включении питания, BIOS запускает программу POST (Power On Self-Test), которая проверяет память, наличие различных устройств и так далее. После успешного завершения тестирования, программа POST выдаёт прерывание Int 19h и обработчик этого прерывания пытается найти загрузочные устройства. Загрузочных устройств может быть несколько, именно из них и будет выбираться устройство для загрузки, приоритеты могут быть заданы пользователям в настройках BIOS.

BIOS производит чтение первого сектора загрузочного устройства (как правило, это 512 байт), называемого Master Boot Record (MBR). MBR состоит из 3-х частей:

- 446 байт – первичный загрузчик.
- 64 байт – таблица разделов, хранящая 4 записи первичных разделов.
- 2 байта – магическое число, используемое для проверки того, что устройство является загрузочным.

Задача первичного загрузчика – найти в таблице разделов активный раздел и загрузить вторичный загрузчик, который находится на выбранном разделе, начиная с первого сектора.

Вторичный загрузчик осуществляет загрузку ядра и образа **`initrd`** (Initial Ram Disc), при необходимости.

Ядро, после загрузки, выполняет запуск программы **`init`**.



GRUB – загрузчик с очень широким функционалом, позволяющим загружать различные операционные системы, передавать управление другим загрузчикам. Одной из главных особенностей GRUB является его способность загружать системы, находящиеся на RAID-массивах, LVM-томах. Предоставляет возможность защиты меню загрузки с помощью пароля.

После проведения всех необходимых процедур и загрузки вторичного загрузчика, на экране монитора появляется графическое меню, в котором начинается отсчёт времени, перед тем как будет загружена операционная система, выбранная по умолчанию. Чтобы прервать отсчёт и получить доступ к меню, необходимо нажать клавишу **<Enter>**, либо клавишу **<Space>**. После нажатия клавиши, управление переходит к пользователю, который, в свою очередь, может выбрать для загрузки необходимую ему систему или войти в режим редактирования опций загрузки.

Для входа в режим редактирования требуется нажать клавишу **<e>**, установив курсор на требуемый пункт. В режиме редактирования пользователь может указать на ядро, либо изменить параметры загрузки ядра. Вне вносимые изменения не будут сохранены при последующих загрузках. Чтобы загрузиться с внесёнными изменениями, необходимо нажать клавишу **<b>**.

GRUB предоставляет пользователю возможность управления загрузчиком посредством командной строки, чтобы получить список команд, необходимо дать команду **`help`**.

Конфигурация загрузчика хранится в файле **`/boot/grub/grub.conf`**:

```
default=3
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title CentOS 6 (2.6.32-279.el6.i686)
        root (hd0,0)
        kernel /vmlinuz-2.6.32-279.el6.i686 ro root=/dev/tuxnote/sroot rhgb quiet
        initrd /initrd-2.6.32-279.el6.i686.img
```

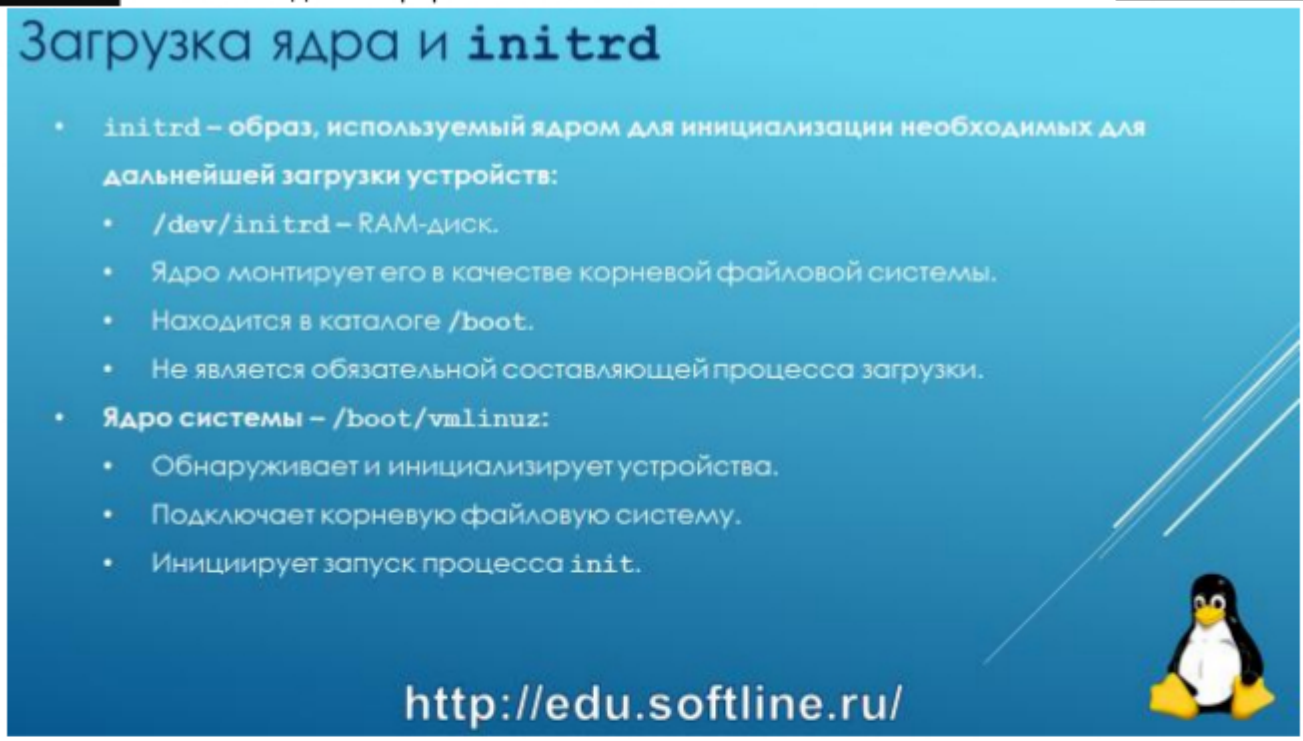
Изменения, внесённые в конфигурацию, вступают в силу незамедлительно.

Установка загрузчика может быть осуществлена с помощью скрипта **`/sbin/grub-install`**:

```
# grub-install /dev/sda
Installation finished. No error reported.
...
```

Не всегда возможна автоматическая установка загрузчика, иногда требуется ручная установка, для этого вы должны воспользоваться командным режимом, вызвав его с помощью команды ***grub***:

```
# grub
(g) root (hd0,0)     # Укажите раздел, на котором находится каталог
/boot
(g) setup (hd0)      # Установите загрузчик в MBR
(g) setup (hd0,0)    # Установите загрузчик в начало раздела
(g) quit
```

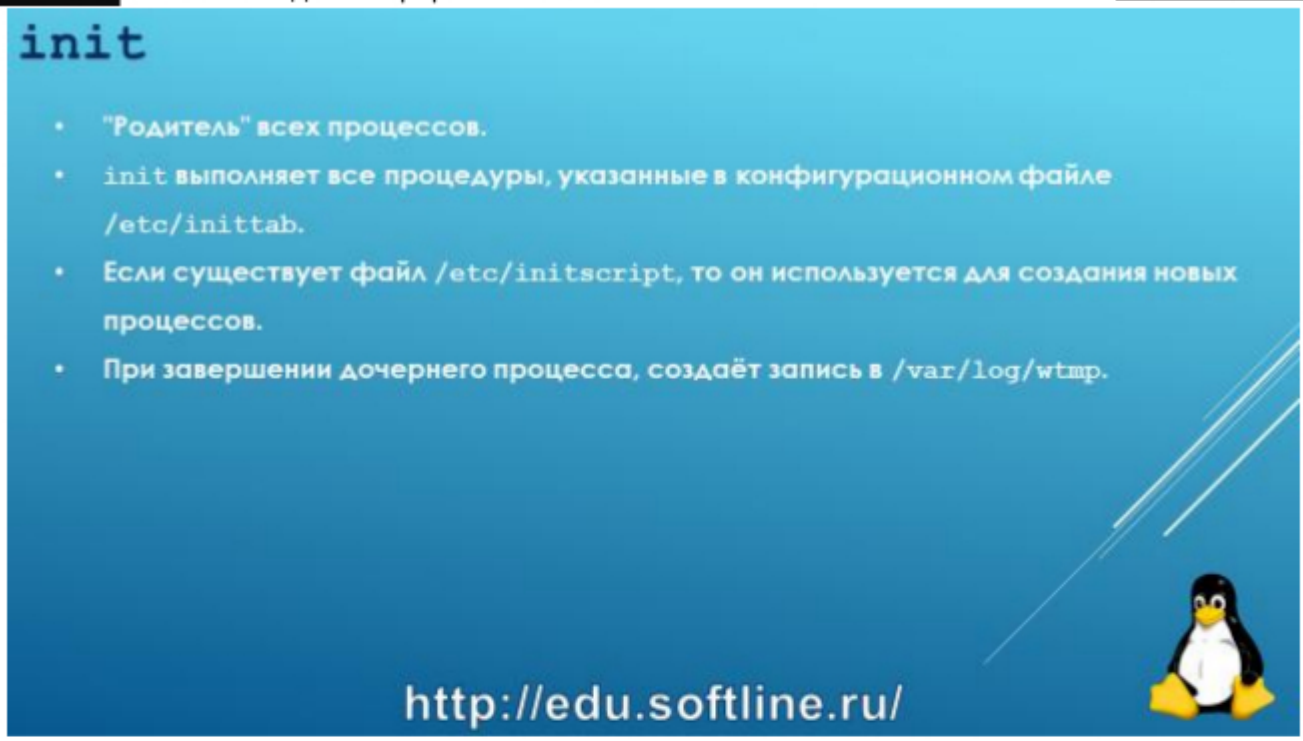


Ядро – это программа, которая обеспечивает взаимодействие с устройствами, управление памятью, управление процессами, сетевое взаимодействие и выполняет ряд других функций.

В процессе инициализации системы, вторичный загрузчик загружает ядро в оперативную память. Как только ядро оказывается загруженным, оно сразу приступает к выполнению процедур, необходимых для запуска системы:

- Распознавание и инициализация устройств;
- Подключение корневой файловой системы;
- Запуск процесса **init**.

В случае, когда для загрузки системы ядру необходимо подключить какие-либо модули, например, для работы с RAID-массивом, то эти модули должны быть включены либо в ядро, либо в образ **initrd**. Загрузчик загружает образ **initrd**, тем самым создавая временную файловую систему с необходимыми модулями. Ядро, используя эту файловую систему в качестве корневой, загружает в память необходимые для работы модули, после чего заменяет временную корневую файловую систему на ту, которая является основной.



Процесс **init** запускается ядром и всегда является первым процессом в запущенной системе, его идентификатор (**PID**) всегда равен единице.

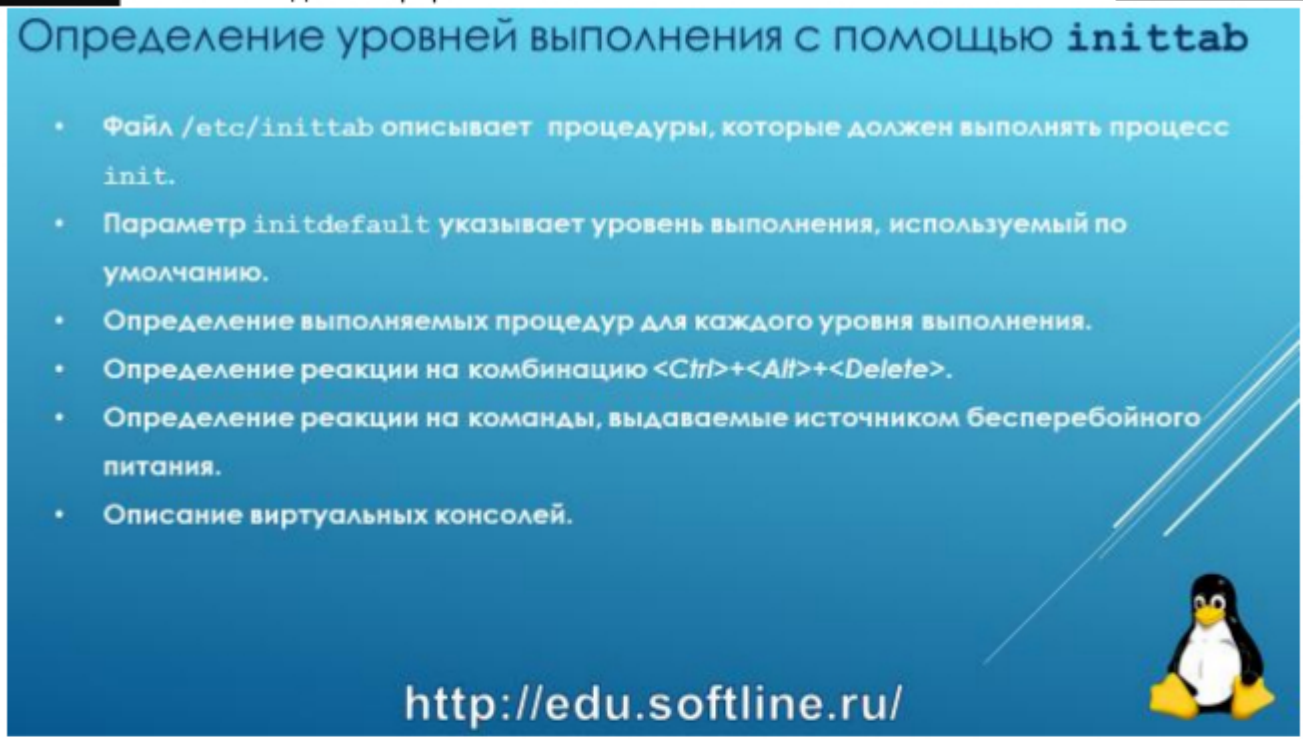
Основная задача программы **init** – выполнение процедур, указанных в файле **/etc/inittab**.

Когда на последнем этапе загрузки ядро вызывает программу **init**, она проверяет файл **/etc/inittab** на наличие записи типа **initdefault**. Эта запись содержит в себе информацию об уровне выполнения, который будет использован по умолчанию. Если такая запись не будет найдена, уровень выполнения необходимо будет указать с помощью специального параметра загрузчика GRUB.

Чтобы задать (или переопределить) уровень выполнения на этапе загрузки операционной системы, пользователь должен перейти в режим редактирования параметров загрузчика GRUB и добавить в конец номер необходимого уровня выполнения:

```
/vmlinuz-2.6.32-279.el6.i686 ro root=/dev/sda1 rhgb quiet 3
```

При запуске любого нового процесса, **init** проверяет наличие файла **/etc/initscript**, если это файл существует, то он будет использован для выполнения процедур, указанных в **/etc/inittab**.



Файл **/etc/inittab** описывает процедуры, которые должны быть выполнены во время загрузки и во время функционирования системы. Программа **init** различает несколько уровней выполнения: **0-6** и **A**,**B**,**C** для специальных записей. Каждая запись имеет следующий формат:

```
id:runlevels:action:process
```

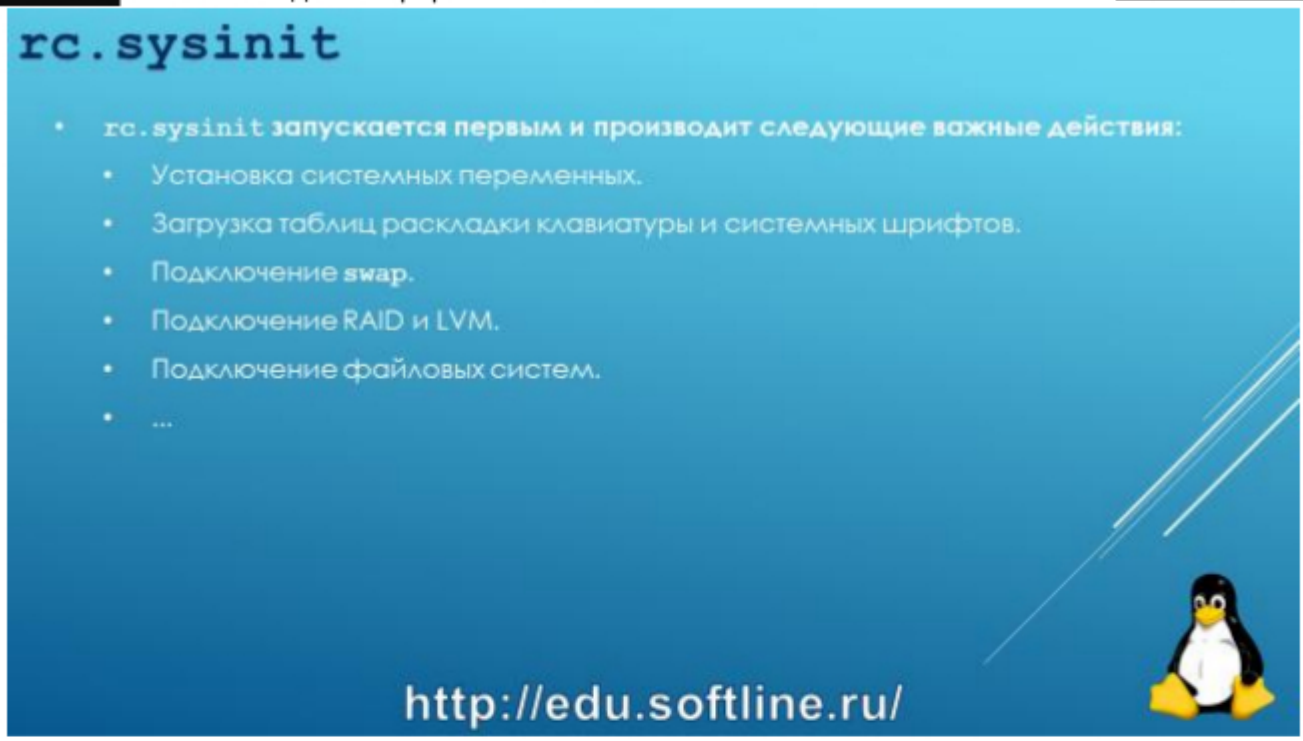
id – уникальный идентификатор, состоящий из 4-х символов. Для процессов, осуществляющих регистрацию в системе, поле id должно оставаться таким же, как и суффикс соответствующего терминала, **1** для **tty1**.

runlevels – список уровней выполнения, для которых указанное действие будет выполнено. Могут быть указаны несколько уровней, например, 345 будет означать, что выполнение указанного действия необходимо осуществлять на 3-м, 4-м и 5-м уровнях выполнения.

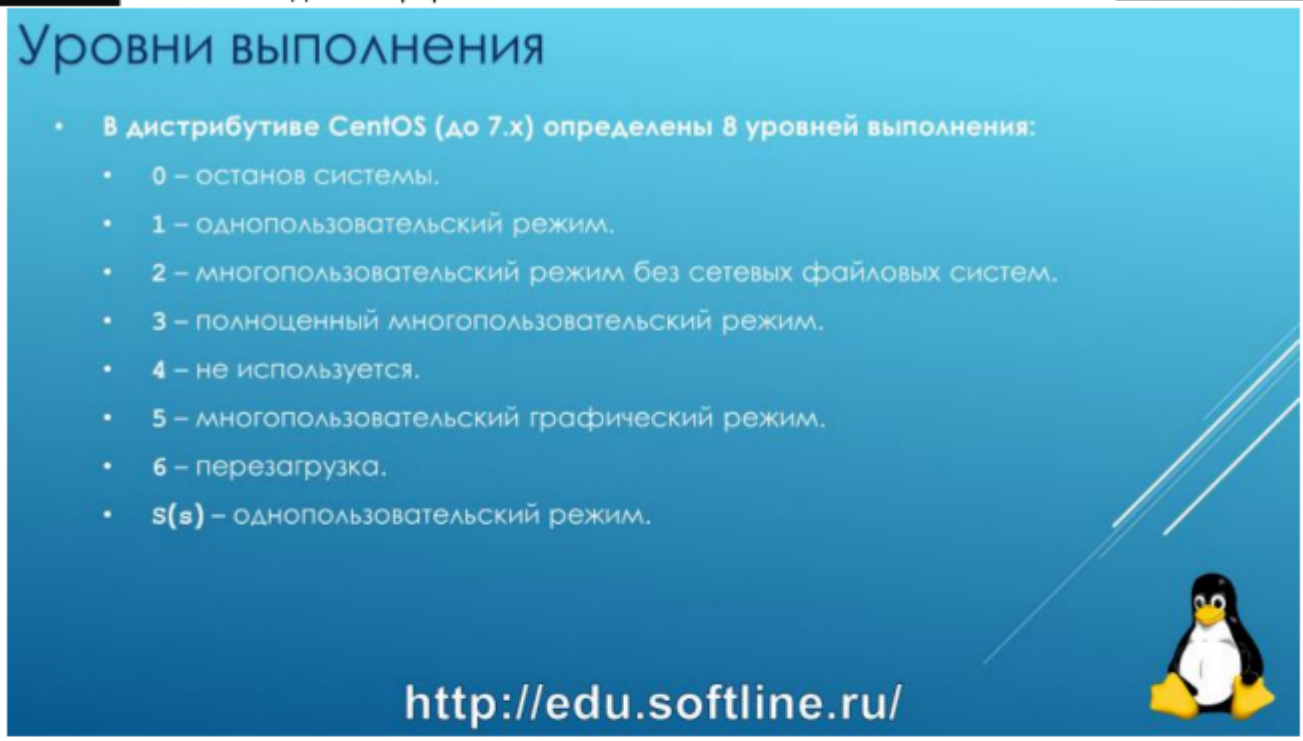
action – необходимое для выполнения действие. Существует множество допустимых действий:

- **respawn** – процесс будет всегда перезапускаться при его завершении;
- **wait** – процесс запускается один раз и *init* будет ожидать завершения его работы;
- **once** – процесс будет запущен однажды;
- **boot** – процесс будет запущен при загрузке системы, поле *runlevels* игнорируется;
- **bootwait** – процесс будет запущен при загрузке системы и *init* будет ожидать завершения его работы, поле *runlevels* игнорируется;
- **off** – ничего не будет выполнено;
- **initdefault** – указание уровня загрузки, в который необходимо перейти после загрузки системы, если не указан, то *init* предложит его ввести;
- **sysinit** – процесс будет запускаться во время загрузки системы перед boot и bootwait;
- **powerwait** – процесс будет запускаться при исчезновении питания и init будет ожидать завершения его работы;
- **powerfail** – процесс будет запускаться при исчезновении питания;
- **ctrlaltdel** – процесс будет запущен, когда *init* получит сигнал *SIGINT*. Этот сигнал может быть инициирован нажатием клавиш <Ctrl>+<Alt>+<Delete>.

- `process` – процесс, необходимый для выполнения.



Сценарий **`rc.sysinit`** запускается однажды при загрузке системы. Его задачей является определение основных параметров системы, подключение виртуальных файловых систем **`/proc`** и **`/sys`**, инициализация SELinux, подключение файловой системы **`swap`**.



Уровень загрузки определяет состав программ, необходимых для запуска процессом `init`. Программа **`init`** использует 8 уровней загрузки, значения которых могут отличаться в зависимости от используемого дистрибутива.

Уровни **0**, **1**, **6** являются зарезервированными и означают следующее:

- **0** – Останов системы.
- **1** – Однопользовательский режим.
- **6** – Перезагрузка системы.

Уровни выполнения **S** и **s** также являются однопользовательскими, но при этом они не требуют наличия файла **`/etc/inittab`** и командная оболочка предоставляется пользователю непосредственно на устройстве **`/dev/console`**.

Уровни загрузки **2-5** используются для загрузки системы в следующих режимах:

- **2** – многопользовательский режим без инициализации сетевых устройств.
- **3** – многопользовательский режим, в котором запускаются все необходимые приложения. Этот режим загрузки используется, как правило, на серверах, так как не вызывает запуск X-сервера.
- **4** – не используется.
- **5** – многопользовательский режим с запуском X-сервера, используется, как правило, на рабочих станциях пользователей.

Все уровни загрузки описаны в файле **`/etc/inittab`**:

```
l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6
```

Запуск программ на определённом уровне выполнения осуществляется с помощью скрипта **rc**, который принимает в качестве аргумента номер уровня. Скрипт производит запуск программ, относящихся к заданному уровню выполнения. Каким образом определяется, какая именно программа должна быть запущена на том или ином уровне выполнения?

Дистрибутив CentOS (RHEL) предлагает иерархию каталогов **/etc/rc.d/rc*?*.d**, где *?*-номер уровня выполнения:

```
$ ls -ld /etc/rc.d/rc?.d
drwxr-xr-x 2 root root 4096 Sep  6 17:30 /etc/rc.d/rc0.d
drwxr-xr-x 2 root root 4096 Sep  6 17:30 /etc/rc.d/rc1.d
drwxr-xr-x 2 root root 4096 Sep  6 17:30 /etc/rc.d/rc2.d
drwxr-xr-x 2 root root 4096 Sep  6 17:30 /etc/rc.d/rc3.d
drwxr-xr-x 2 root root 4096 Sep  6 17:30 /etc/rc.d/rc4.d
drwxr-xr-x 2 root root 4096 Sep  6 17:30 /etc/rc.d/rc5.d
drwxr-xr-x 2 root root 4096 Sep  6 17:30 /etc/rc.d/rc6.d
```

Каждый из каталогов содержит множество символических ссылок на стартовые скрипты из каталога **/etc/init.d**.

Стартовый скрипт предназначен для управления запуском/остановом программы. Для каждого приложения используется отдельный стартовый скрипт, который должен обрабатывать два основных аргумента – **start** и **stop**. Если стартовый скрипт вызывается с аргументом **start**, то он осуществляет запуск процесса, если **stop** – останавливает его. Например, для запуска программы **sendmail** необходимо вызвать стартовый скрипт командой:

```
# /etc/init.d/sendmail start
```

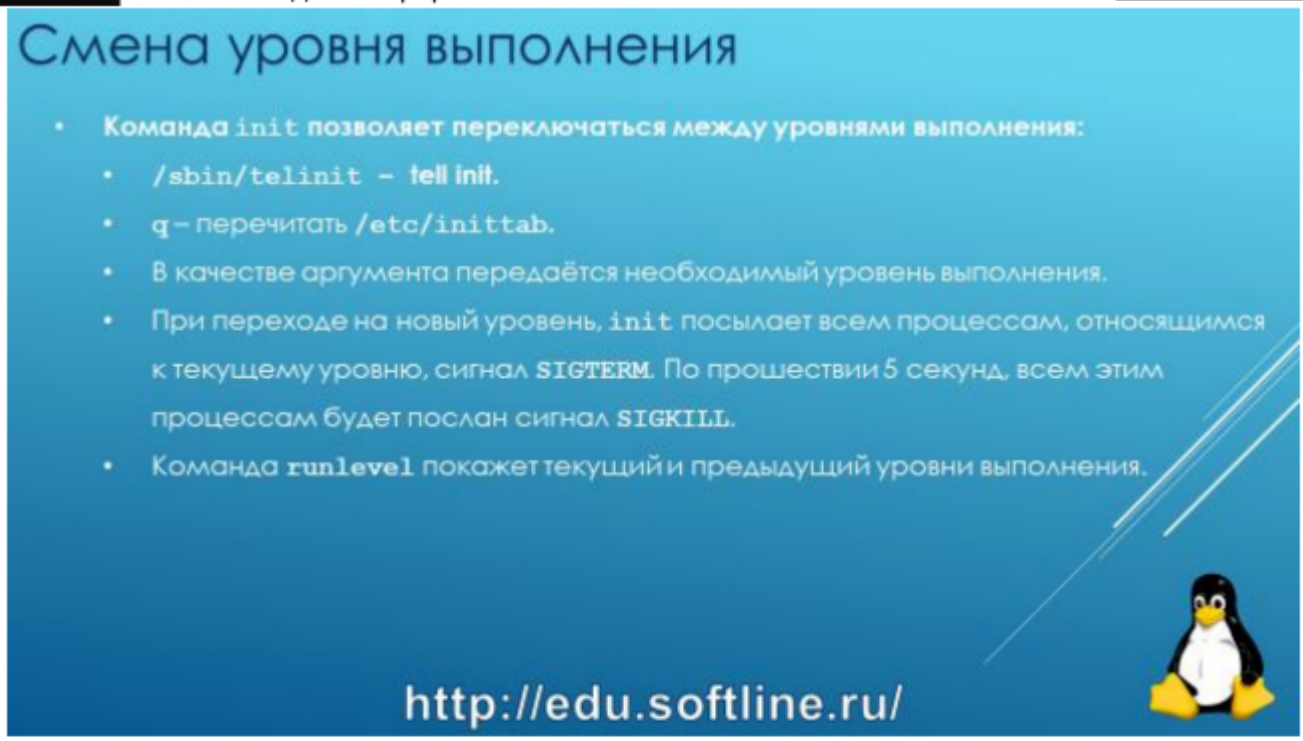
или командой:

```
# service sendmail start
```

Символические ссылки, расположенные в каталогах **/etc/rc.d/rc?.d**, содержат в имени префикс, который состомт из буквы **S** (start) или **K** (kill) и номера, указывающего на порядковый номер:

```
$ ls /etc/rc.d/rc3.d/ | grep ^K | head
K02avahi-dnsconfd
K02oddjobd
K05conman
K05saslauthd
K10psacct
K20nfs
K24irda
K30spamassassin
K35vncserver
K35winbind
…
```

Таким образом, программный сценарий **rc** будет осуществлять запуск программ по символической ссылке, при этом порядок запуска будет определяться порядковым номером в префиксе ссылки.

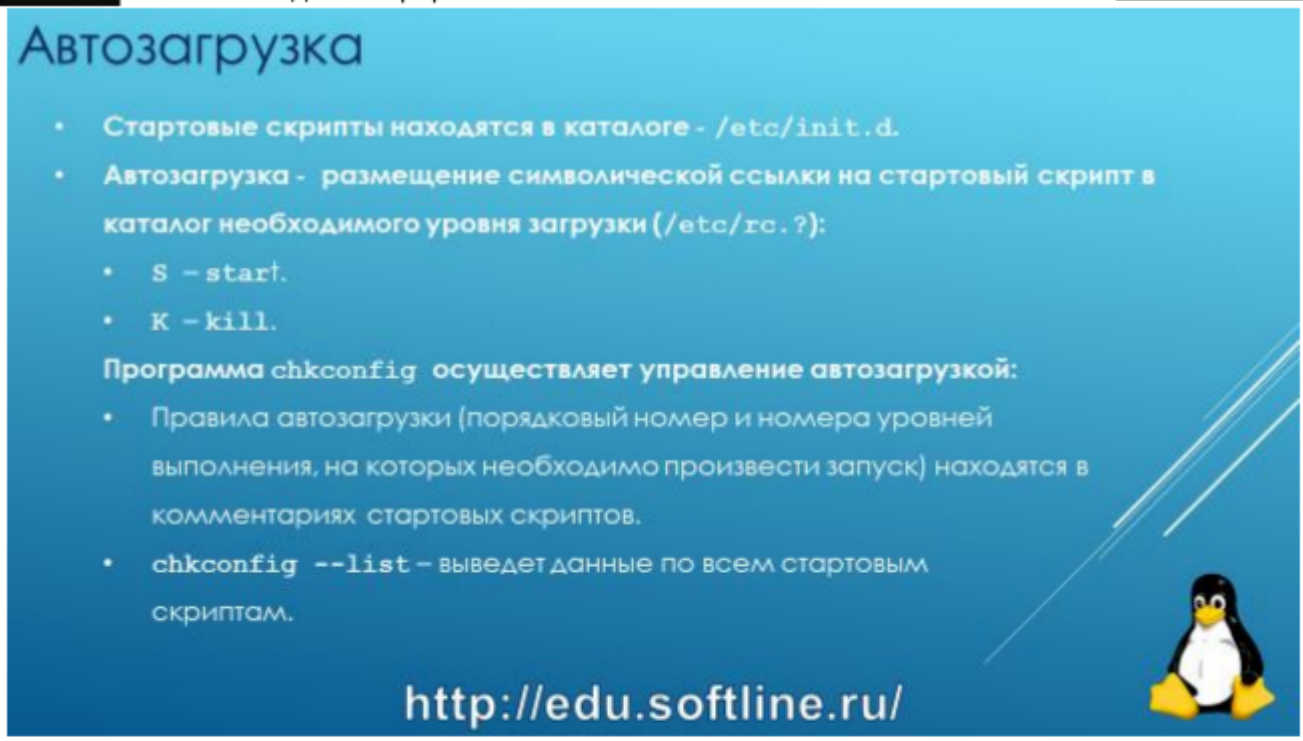


В процессе работы пользователь может перейти на другой уровень выполнения с помощью вызова команды **init** и передаче ей, в качестве аргумента, номера уровня выполнения, на который необходимо перейти:

```
$ init 3
```

Команда **runlevel** выдаст информацию о том, на каком уровне выполнения система находится на данный момент времени и на каком была:

```
# runlevel
N 5
```



С точки зрения операционной системы Linux, автозагрузка подразумевает размещение стартового скрипта в необходимом каталоге **/etc/rc.?**, откуда он будет запущен программой **init** при загрузке операционной системы. Например, если у вас есть стартовый скрипт **/etc/init.d/progstart**, то определить его запуск с аргументом **start** на третьем уровне выполнения можно следующим образом:

```
# ln -s /etc/init.d/progstart /etc/rc.d/rc3.d/S79progstart
# ln -s /etc/init.d/progstart /etc/rc.d/rc0.d/K15progstart
# ln -s /etc/init.d/progstart /etc/rc.d/rc6.d/K15progstart
```

Как видно, мы указали, что на третьем уровне скрипт **/etc/init.d/progstart** должен быть запущен под номером **79** с аргументом **start**. А на уровнях выполнения 0 и 6 наш скрипт уже будет запущен под номером **15** с аргументом **stop**. Согласитесь, что это не очень удобный способ управления автозагрузкой. Утилита **chkconfig** позволяет автоматизировать эту процедуру. Эта команда оперирует данными, находящимися в комментариях стартовых скриптов. Давайте посмотрим на примере стартового скрипта программы **syslog**:

```
$ cat /etc/init.d/syslog | head
#!/bin/bash
#
# syslog        Starts syslogd/klogd.
# chkconfig: 2345 12 88
```

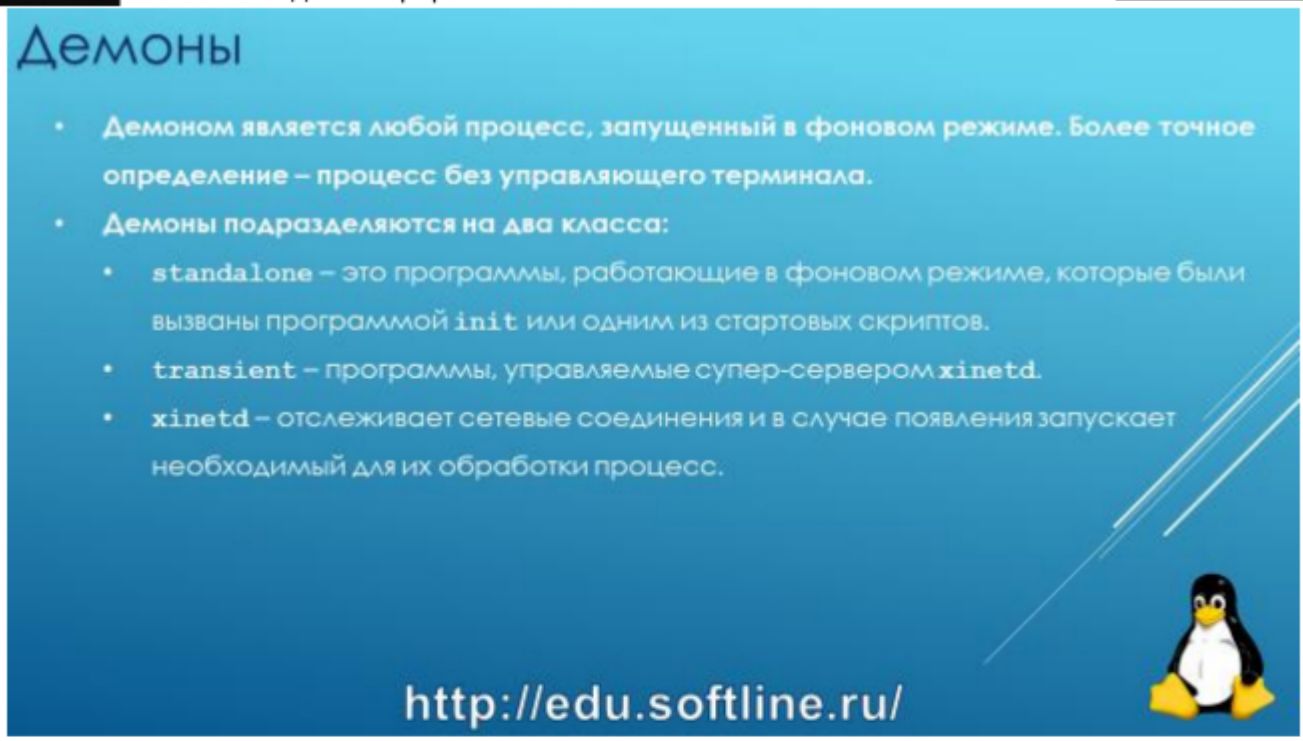
Программа **chkconfig** управляет только теми стартовыми скриптами, которые зарегистрированы в её базе данных. Чтобы получить информацию о стартовых скриптах, включённых для использования утилитой, необходимо вызвать **chkconfig** с параметром **--list**:

```
# chkconfig --list | head -n 2
acpid           0:off   1:off   2:off   3:on    4:on    5:on    6:off
anacron         0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

Чтобы удалить скрипт из базы данных программы **chkconfig**, необходимо вызвать её с параметром **--del**, а для добавления в базу использовать параметр **--add**.

Для включения/выключения автозагрузки в соответствии с указанными данными в комментариях, необходимо выполнить команду **chkconfig** и передать ей в качестве аргументов имя стартового скрипта и **on** для включения в автозагрузку, **off** для исключения:

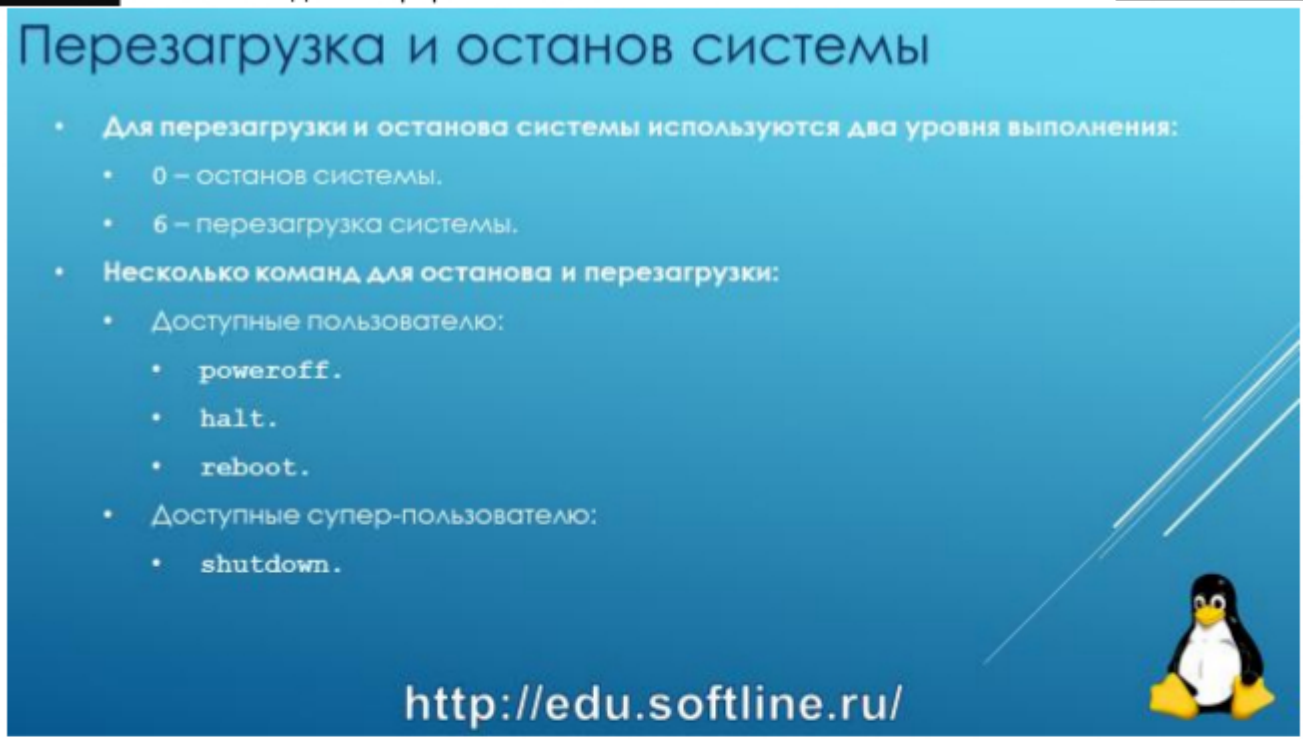
```
# chkconfig acpid on
# chkconfig acpid off
```



Любой процесс, не имеющий управляющего терминала, является демоном (**daemon**). Программы, обеспечивающие сетевые сервисы, обслуживание системы, как правило, запускаются в качестве демонов. Существует два типа демонов:

- **Standalone** - Запускаемые программой `init`, стартовым скриптом, пользователем. Проще говоря, программы, способные становиться демонами самостоятельно. Такие программы загружаются в память и выполняют свои функции до тех пор, пока не наступит какое-либо условие, принуждающее программу завершиться;
- **Transient** – процессы, запускаемые в качестве демонов при помощи воздействия внешней программы. Такой программой является демон **xinetd.**

Основная задача суперсервера **xinetd** – открытие сетевых портов тех служб, которые находятся под его управлением. Таким образом, сетевой порт открыт для внешних соединений, но этот порт обслуживается не тем процессом, который должен обрабатывать входящие соединения, а суперсервером `xinetd`. Как только поступает запрос на соединение, xinetd запускает соответствующий процесс и передаёт ему управление. Такая система позволяет экономить аппаратные ресурсы сервера, но и имеет ряд недостатков. Суперсервер `xinetd` на сегодняшний день используется достаточно редко, преимущественно для специфических задач.



Завершение работы с операционной системой необходимо производить корректно, в первую очередь для того, чтобы не потерять важные данные. Linux отличается высокой стабильностью и остановы системы требуются не часто, в основном при работе с оборудованием или при обновлении ядра.

Останов или перезагрузку системы можно произвести несколькими способами:

- Перейти в соответствующий уровень выполнения:
  - **# init 0**
  - **# init 6**
- Команда **shutdown** обеспечивает необходимый функционал для операций по перезагрузке и останову системы. В качестве аргумента может быть передано значение, указывающее на время, через которое команда должна быть выполнена. Команда **shutdown** может быть вызвана со следующими основными параметрами:
  - **-k** – не производить реальное выключение, а только послать предупреждающее сообщение всем пользователям.
  - **-r** – после выключения перезагрузить компьютер.
  - **-h** – произвести останов системы.
  - **-c** – отменить отложенный вызов команды **shutdown**, если команда **shutdown** ранее была вызвана с аргументом времени.
- Команда **halt** выполняет останов системы:
  - **-n** – не производить синхронизацию устройств хранения информации.



Работа с загрузчиком GRUB 2.

CentOS 7 (RHEL 7) распространяется с GNU Grand Unified Boot Loader (GRUB) версии 2, который позволяет выбирать операционную систему или загрузчик в процессе загрузки системы. GRUB 2 также позволяет передавать аргументы ядру.

Настройка загрузчика GRUB 2.

GRUB2 читает настройки из файла **/boot/grub2/grub.cfg** или **/boot/efi/EFI/redhat/grub.cfg** на машинах с UEFI. Этот файл содержит информацию меню, однако, он не поддерживает редактирование, так как его генерирует утилита **/usr/sbin/grub2-mkconfig** в каталоге расположения ядра **/boot/**. Скрипты, расположены в **/etc/grub.d/**, а специальные настройки в файле **/etc/default/grub**. Любые ручные настройки будут потеряны в процессе обновления. Файл **grub.cfg** автоматически обновляется каждый раз, когда устанавливается новое ядро. Для обновления конфигурационного файла вручную, введите следующую команду с правами пользователя **root**:

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

Для систем с UEFI используйте:

```
# grub2-mkconfig -o /boot/efi/EFI/redhat/grub.cfg
```

Среди различных фрагментов кода и директив файл **grub.cfg** содержит один или более блоков **menuentry**. Каждый представляет собой одну запись в загрузочном меню GRUB 2. Эти блоки всегда начинаются с ключевого слова **menuentry**, далее название, список опций и блок в фигурных скобках. Все что находится между фигурными скобками должно быть написано с отступом. Следующий пример, демонстрирует блок **menuentry** для CentOS 7.1 с ядром 3.10.0-229.el7.x86_64:

```
menuentry 'CentOS Linux 7 (Core), with Linux 3.10.0-229.el7.x86_64' --class rhel
fedora --class gnu-linux --class gnu --class os --unrestricted $menuentry_id_option
'gnulinux-3.10.0-229.el7.x86_64-advanced-903040a3-1c9d-4ea8-9be4-eef3859199c5' {
        load_video
        set gfxpayload=keep
```

```
        insmod gzio
        insmod part_msdos
        insmod xfs
        set root='hd0,msdos1'
        if [ x$feature_platform_search_hint = xy ]; then
          search --no-floppy --fs-uuid --set=root --hint-bios=hd0,msdos1 --hint-
efi=hd0,msdos1 --hint-baremetal=ahci0,msdos1 --hint='hd0,msdos1'  de092607-0d4c-
4204-8308-6a06176bbac8
        else
          search --no-floppy --fs-uuid --set=root de092607-0d4c-4204-8308-
6a06176bbac8
        fi
        linux16 /vmlinuz-3.10.0-229.el7.x86_64 root=/dev/mapper/centos-root ro
rd.lvm.lv=centos/root rd.lvm.lv=centos/swap rhgb quiet LANG=en_US.UTF-8
        initrd16 /initramfs-3.10.0-229.el7.x86_64.img
}
```

Каждый блок **menuentry**, который представляет установленное ядро Linux содержит директивы **linux** (**linuxefi** на системах с EFI) и **initrd**, за ними следуют пути к ядру и образу **initramfs**. Если отдельный раздел **/boot** был создан, пути к ядру и образу **initramfs** записываются от **/boot**. В примере выше строка **initrd /initramfs-3.10.0-229.el7.x86_64.img** обозначает, что образ **initramfs** на самом деле расположен в **/boot/initramfs-3.10.0-229.el7.x86_64.img**, когда корневая файловая система смонтирована, так же и с путем к ядру.

Версия ядра в строке **linux /vmlinuz-kernel_version** должна совпадать с версией образа **initramfs** в строке **initrd /initramfs-kernel_version.img** каждого блока **menuentry**.



ПРИМЕЧАНИЕ:

*В блоке **menuentry** директива **initrd** должна указывать путь к файлу **initramfs** той же версии что и ядро. Директива называется **initrd**, потому что ранее использовались файлы **initrd**, создаваемые при помощи утилиты **mkinitrd**. Директива **initrd** осталась в файле **grub.cfg** для поддержки совместимости со старыми инструментами. Для создания начальных RAM-дисков теперь используется утилита **dracut**.*



Настройка меню GRUB 2.

Скрипты GRUB 2 ищут на машине операционные системы и строят загрузочное меню. Для отражения новейших загрузочных опций, загрузочное меню автоматически перестраивается при каждом обновлении ядра или, когда новое ядро добавляется.

Однако, пользователи могут захотеть перестроить меню таким образом, чтобы добавить в него необходимые записи или изменить порядок записей. GRUB 2 позволяет базовую настройку загрузочного меню для предоставления управления отображаемым на экране.

GRUB 2 использует серию скриптов для построения меню. Они расположены в каталоге **`/etc/grub.d/`** и включают:

- **`00_header`**, который загружает параметры GRUB 2 из файла **`/etc/default/grub`**.
- **`10_linux`**, который находит ядра в разделе по умолчанию CentOS (RHEL).
- **`30_os-prober`**, который строит записи для операционных систем, найденных на других разделах.
- **`40_custom`**, шаблон, который используется для создания дополнительных записей меню.

Скрипты из каталога **`/etc/grub.d/`** читаются в алфавитном порядке и могут быть переименованы для изменения порядка записей загрузочного меню.



ПРИМЕЧАНИЕ:

*Если вы установите ключ **`GRUB_TIMEOUT`** в значение **`0`** в файле **`/etc/default/grub`**, GRUB 2 не будет отображать список возможных для загрузки ядер во время старта системы. Для отображения списка загрузки зажмите цифровую или символьную клавишу сразу после отображения информации о BIOS и тогда GRUB 2 отобразит меню.*

Смена загрузочной записи по умолчанию.

По умолчанию используется значение **`saved`** для ключа **`GRUB_DEFAULT`** в файле **`/etc/default/grub`**. Это инструктирует GRUB 2 загружать последнюю успешно загруженную

операционную систему. Однако GRUB 2 не поддерживает числовые значения для этого ключа, для смены порядка по умолчанию, в котором операционные системы загружаются. Для указания, какая операционная систем должна быть загружена первой, используйте команду **grub2-set-default**, например:

```
# grub2-set-default 2
```

Обратите внимание, что позиции меню начинаются с **0**, таким образом предыдущая команда установила загрузку третей записи по умолчанию.

Перед перезагрузкой системы обновите конфигурационный файл, запустив команду:

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

Или для систем с UEFI:

```
# grub2-mkconfig –o /boot/efi/EFI/redhat/grub.cfg
```

Редактирование записей.

Параметры ядра.

Для использования параметров ядра только в процессе загрузки, когда загрузочное меню GRUB 2 отображается перейдите к ядру, которые вы хотите запустить, нажмите клавишу **e** и отредактируйте строку с ядром, добавив в нее параметры ядра. Например, для запуска системы в спасательном режиме (Emergency Mode) добавьте параметр **emergency** в конец строки **linux**.

```
linux16 /vmlinuz-3.10.0-229.el7.x86_64 root=/dev/mapper/centos-root ro
rd.lvm.lv=centos/root rd.lvm.lv=centos/swap rhgb quiet LANG=en_US.UTF-8 emergency
```

Однако, этот параметр применится только один раз, для того чтобы параметр остался, отредактируйте значения в ключе **GRUB_CMDLINE_LINUX** в файле **/etc/default/grub**. Например, вы можете включить аварийный режим (Emergency Mode) при каждой загрузке, отредактировав файл следующим образом:

```
GRUB_CMDLINE_LINUX="emergency"
```

Обратите внимание, что вы можете указать несколько параметров для ключа **GRUB_CMDLINE_LINUX** также, как и при добавлении параметров, а загрузочном меню GRUB 2.

Затем запустите команду:

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

Или для систем с UEFI:

```
# grub2-mkconfig –o /boot/efi/EFI/redhat/grub.cfg
```

Добавление новой записи.

Когда выполняется команда **grub2-mkconfig**, GRUB 2 ищет ядра Linux и другие операционные системы на базе файлов в каталоге **/etc/grub.d/**. Скрипт **10_linux** ищет установленные linux ядра на том же разделе. **30_os-prober** ищет остальные операционные системы. Записи меню также автоматически добавляются в загрузочное меню при обновлении ядра.

Файл **40_custom** расположенный в каталоге **/etc/grub.d/** - это шаблон для записей создаваемых вручную и выглядит он следующим образом.

```
#!/bin/sh
exec tail -n +3 $0
# This file provides an easy way to add custom menu entries.  Simply type the
# menu entries you want to add after this comment.  Be careful not to change
# the 'exec tail' line above.
```

Этот файл может быть отредактирован или скопирован. Запомните, что минимальная, корректная запись должна включать как минимум следующее:

```
Menuentry "<Title>"{
<Data>
}
```

Использование только настроенных меню.

Если вы хотите использовать только настроенные записи меню, вы можете создать настраиваемое меню.



ВНИМАНИЕ:

Перед началом данной процедуры сделайте резервную копию каталога ***/etc/grub.d/*** *на случай, если придется вернуться назад позже.*



ПРИМЕЧАНИЕ:

Редактирование файла ***/etc/default/grub*** *не оказывает эффекта при создании настраиваемого меню.*

1. Скопируйте и вставьте содержимое файла **/boot/grub2/grub.cfg** или **/boot/efi/EFI/redhat/grub.cfg** в файл **/etc/grub.d/40_custom** после заголовка, так чтобы исполняемая часть скрипта **40_custom** осталась.
2. Удалите строки выше первой загрузочной записи до исполняемой части скрипта и ниже последней записи.

   Пример настроенного скрипта **40_custom**:

3. Удалите все файлы из каталога **/etc/grub.d**, кроме **00_header**, **40_custom** и **README**.

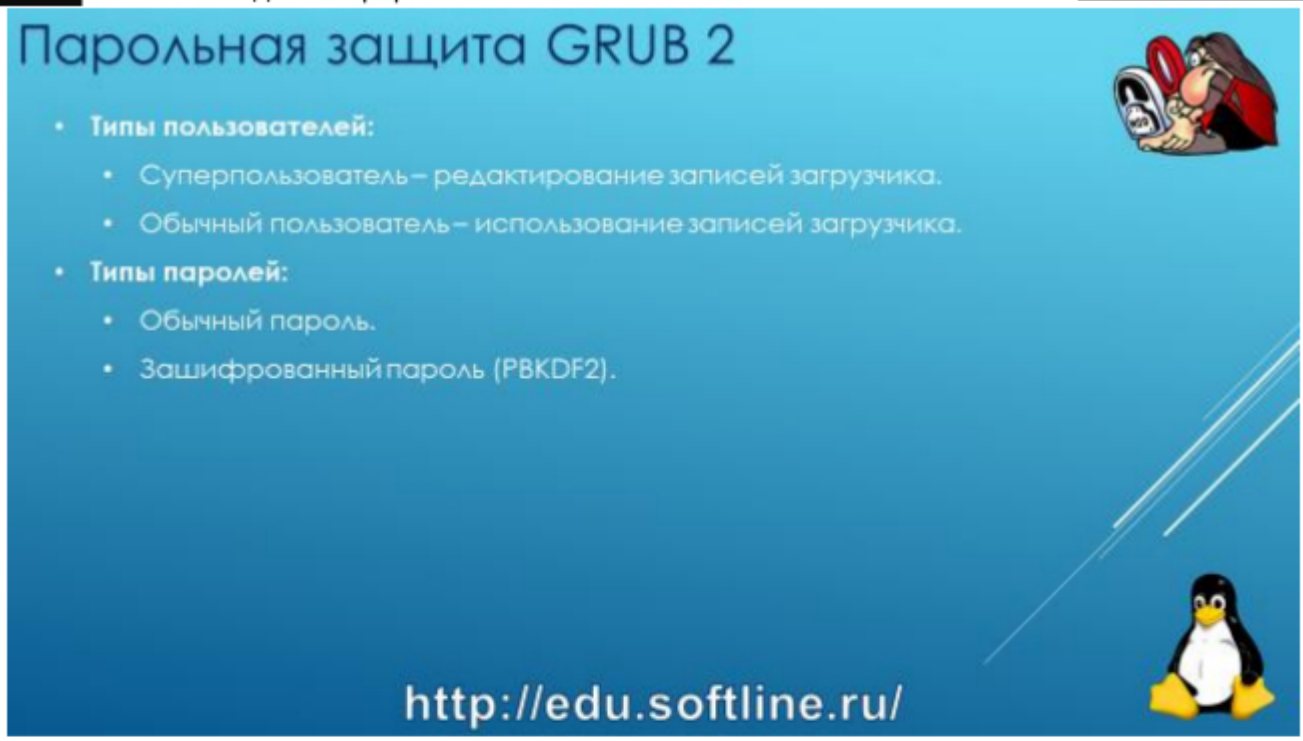
   Вместо удаления файлов можно снять право выполнения на файлах при помощи команды **chmod a-x** *имя_файла*.

4. Отредактируйте, добавьте или удалите записи в **40_custom**.
5. Обновите файл **grub.cfg**, выполнив команду:

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

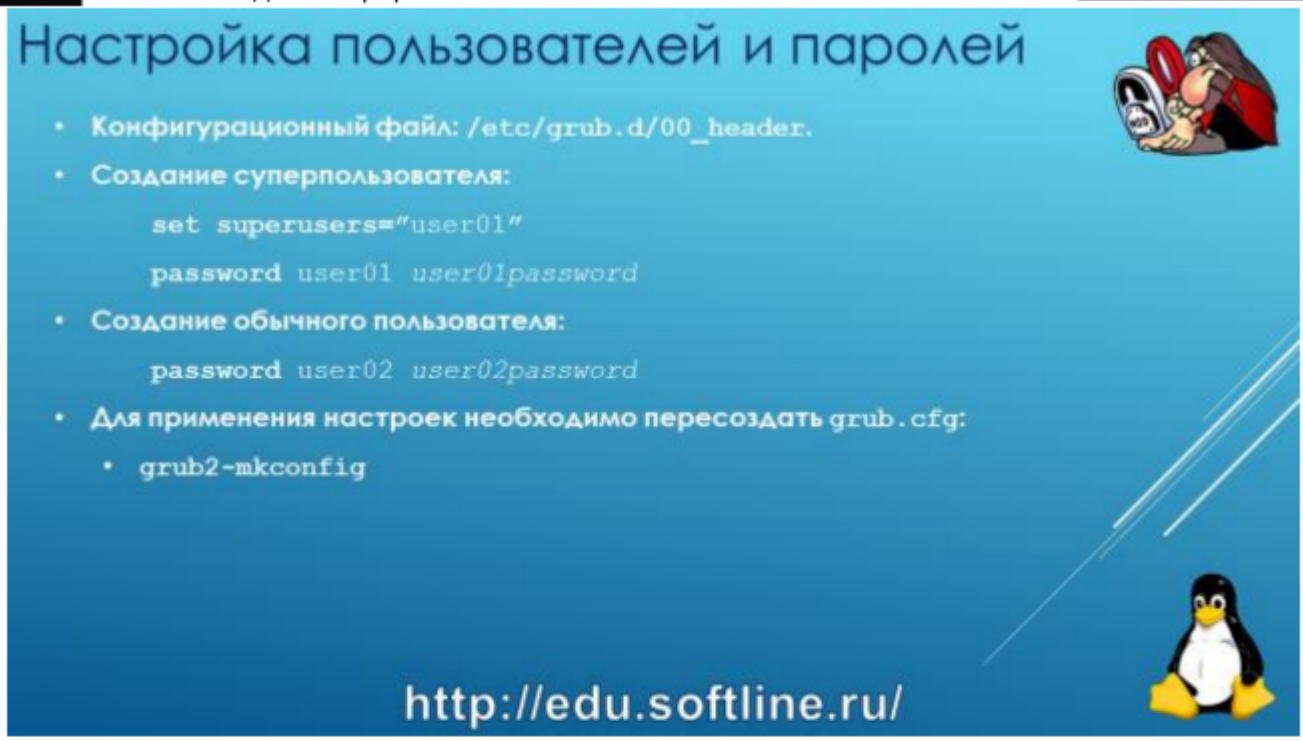
   Или для систем с UEFI:

   ```
   # grub2-mkconfig -o /boot/efi/EFI/redhat/grub.cfg
   ```



Парольная защита GRUB 2.

GRUB 2 поддерживает защиту при помощи пароля, которая использует незашифрованные пароли в шаблонах файлов GRUB 2. Для включения данного функционала, укажите супер-пользователя, который может использовать защищенные записи и пароль. Прочим пользователям, также может быть разрешен доступ к этим записям. Для указания к какие записи меню должны быть защищены паролем, отредактируйте файл **`/etc/grub.d/00_header`**. В качестве альтернативы, если вы хотите сохранить настройки после обновлений GRUB 2, измените файл **`/etc/grub.d/40_custom`**.



Настройка пользовательской и парольной защиты, определение записей меню.

1. Для указания супер-пользователя добавьте следующие строки в файл **/etc/grub.d/00_header**, где **john** это имя пользователя оформленного, как супер-пользователь а **johnpassword** – это его пароль:

```
cat << EOF
set superusers="john"
password john johnpassword
EOF
```

2. Чтобы разрешить остальным пользователям доступ к записям меню, добавьте следующие строки для каждого пользователя в конец файла **/etc/grub.d/00_header**.

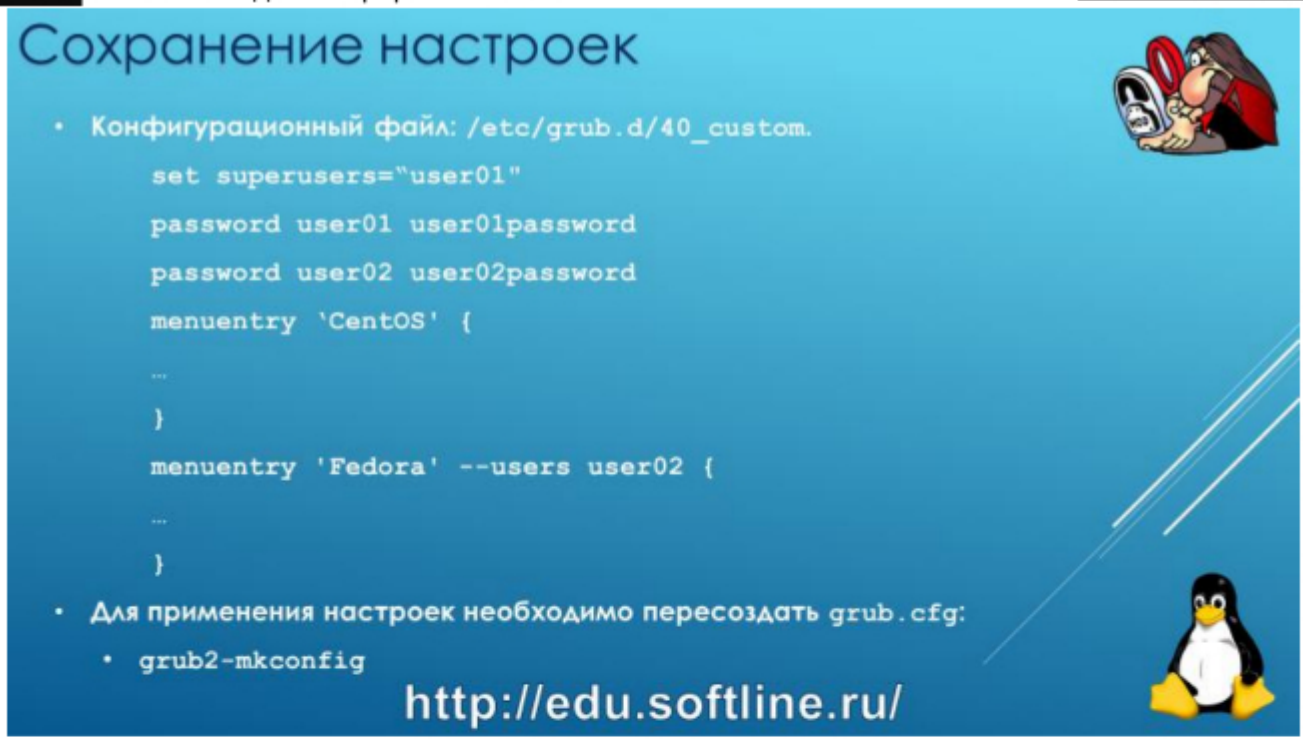
```
cat << EOF
set superusers="john"
password john johnpassword
password jane janepassword
EOF
```

3. После указания пользователей и паролей, укажите записи, которые должны быть защищены. Если вы не укажите записи, то все записи будут защищены паролем по умолчанию.
4. Обновите конфигурационный файл **grub.cfg**, выполнив команду:

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

Или для систем с UEFI:

```
# grub2-mkconfig -o /boot/efi/EFI/redhat/grub.cfg
```



Сохранение настроек после обновлений GRUB 2.

В качестве альтернативы, если вы хотите сохранить изменения после будущих обновлений GRUB2, добавьте строки в файл **/etc/grub.d/40_custom**, как показано в примере:

```
set superusers="john"
password john johnspassword
password jane janespassword
m enuentry 'Red Hat Enterprise Linux Client' {
set root=(hd0,m sdos1)
linux /vm linuz
}
m enuentry 'Fedora' --user jane {
set root=(hd0,m sdos2)
linux /vm linuz
}
```

В примере выше, супер-пользователь (superuser) **john** может загружать любую запись меню и использовать командную строку меню GRUB 2 в процессе загрузки. В данном случае **john** может получить доступ к обоим записям: CentOS и Fedora. Любой может загрузить CentOS. Пользователь **jane** может загрузить запись Fedora, так как она указана в конфигурации. Если вы не укажите записи меню, то парольная защита не будет работать.

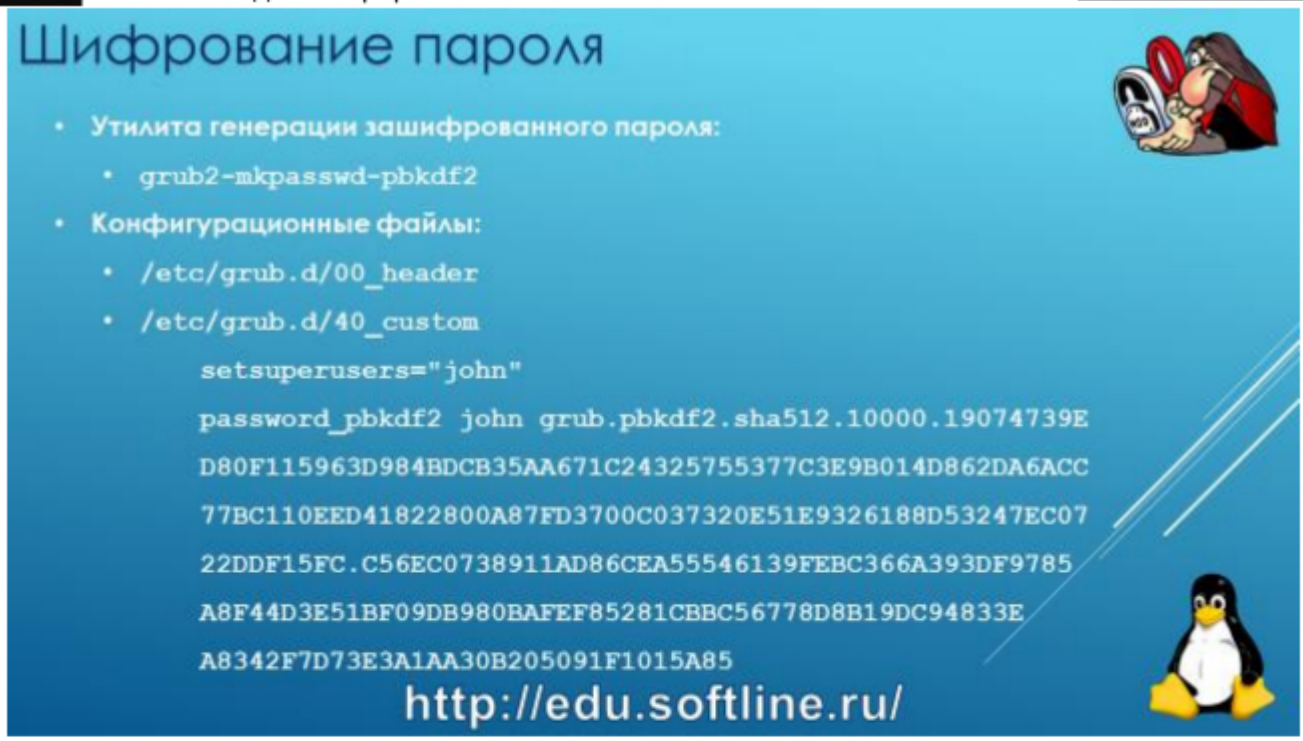
После внесения изменений, необходимо обновить конфигурационный файл **grub.cfg**, выполнив команду:

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

Или для систем с UEFI:

```
# grub2-mkconfig –o /boot/efi/EFI/redhat/grub.cfg
```

Если вы не указали пользователя и пароль для записей меню, пароль супер-пользователя (superuser) будет запрашиваться при попытке доступа к системе.



Шифрование пароля.

По умолчанию пароли сохраняются в виде плоского текста в скриптах GRUB 2. Нужно учесть, что к файлам нету доступа в процессе загрузки без указания корректного пароля, но безопасность может быть усилена за счет шифрования пароля при помощи команды **grub2-mkpasswd-pbkdf2**. Эта команда конвертирует указанный пароль в длинный хэш, который затем помещается в скрипты GRUB 2, вместо обычного текстового пароля.

1. Для генерации зашифрованного пароля используйте команду **grub2-mkpasswd-pbkdf2** от имени пользователя root.
2. Введите нужный пароль и повторите его после соответствующего запроса. Команда выдаст вам пароль в зашифрованном виде.
3. Скопируйте хэш и вставьте его в файл шаблона, где вы настраивали пользователя: **/etc/grub.d/00_header** или **/etc/grub.d/40_custom**. Следующий формат применяется для файла **00_header**:

```
cat <<EOF
setsuperusers="john"
password_pbkdf2 john
grub.pbkdf2.sha512.10000.19074739ED80F115963D984BDCB35AA671C24325755377C3E9B0
14D862DA6ACC77BC110EED41822800A87FD3700C037320E51E9326188D53247EC0722DDF15FC.
C56EC0738911AD86CEA55546139FEBC366A393DF9785A8F44D3E51BF09DB980BAFEF85281CBBC
56778D8B19DC94833EA8342F7D73E3A1AA30B205091F1015A85
EOF
```

А следующий формат для файла **40_heder**:

```
setsuperusers="john"
password_pbkdf2 john
grub.pbkdf2.sha512.10000.19074739ED80F115963D984BDCB35AA671C24325755377C3E9B0
14D862DA6ACC77BC110EED41822800A87FD3700C037320E51E9326188D53247EC0722DDF15FC.
C56EC0738911AD86CEA55546139FEBC366A393DF9785A8F44D3E51BF09DB980BAFEF85281CBBC
56778D8B19DC94833EA8342F7D73E3A1AA30B205091F1015A85
```



ВНИМАНИЕ:

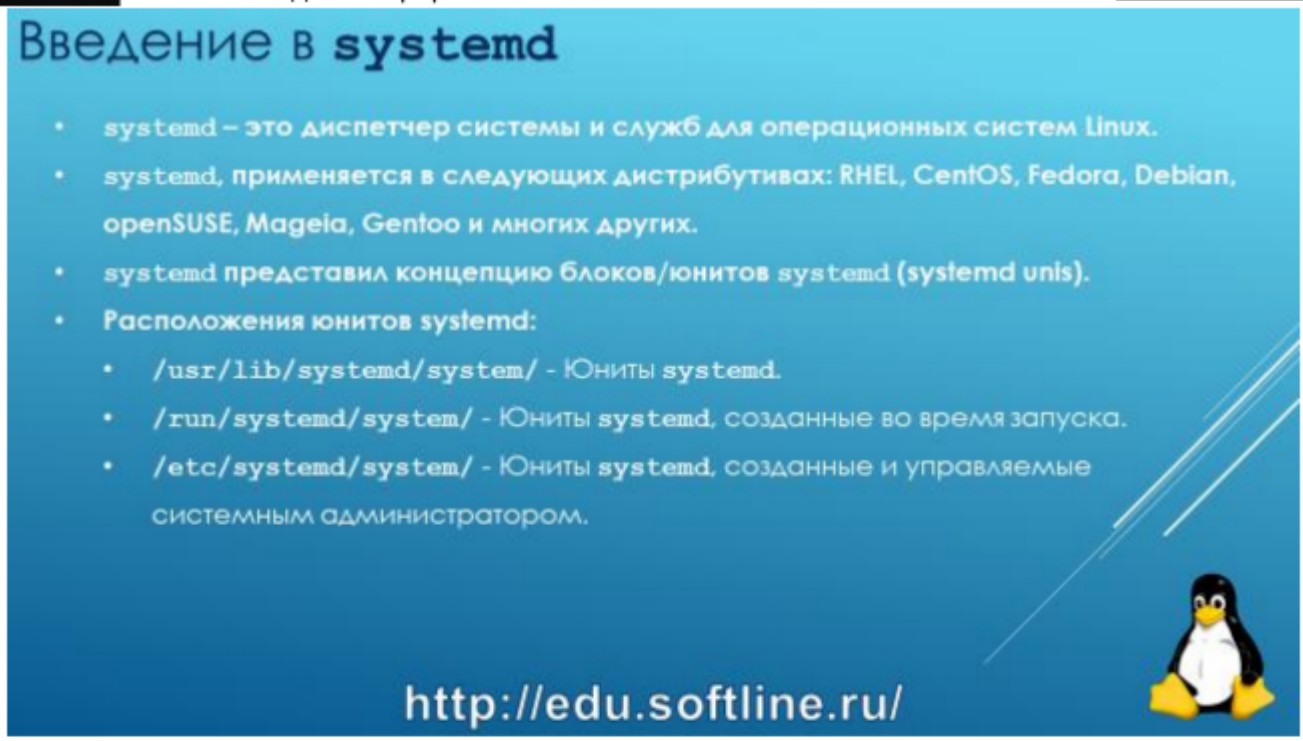
Если вы используете некорректный формат или внесете неверные данные в файл вы не сможете запустить систему.



Дополнительные ресурсы.

Для получения дополнительной информации по GRUB 2 воспользуйтесь следующими ресурсами:

- **`/usr/share/doc/grub2-tools-<`***`version-number`***`>`** - каталог содержит информацию об использовании и настройке GRUB 2.
- **`info grub2`** – Страница руководства, содержит общую информацию, руководства для пользователя и программиста, а также наиболее часто задаваемые вопросы и многое другое.



systemd – это диспетчер системы и служб для операционных систем Linux. Он разработан с возможностью обратной совместимости с инициализационными скриптами SysV и предоставляет набор возможностей, таких как параллельный запуск системных служб во время загрузки, запуск демонов по требованию, поддержка моментальных снимков состояния системы и логику управления службой, основанную на зависимостях. В CentOS 7 (RHEL 7), systemd заменил Upstart, как систему инициализации по умолчанию.

ПРИМЕЧАНИЕ:
systemd, применяется в следующих дистрибутивах: Fedora, Debian, openSUSE, Mageia, Gentoo и многих других.

systemd представил концепцию блоков/юнитов systemd (systemd units). Эти юниты представлены конфигурационными файлами юнитов, расположенных в одном из специальных каталогов (таблица расположения юнитов systemd, ниже), и инкапсулирующие информацию о системных службах, прослушиваемых сокетах, сохраненных моментальных снимках состояния системы и прочих объектах, которые имеют отношение к системе инициализации. Полный список юнитов systemd представлен в таблице ниже.

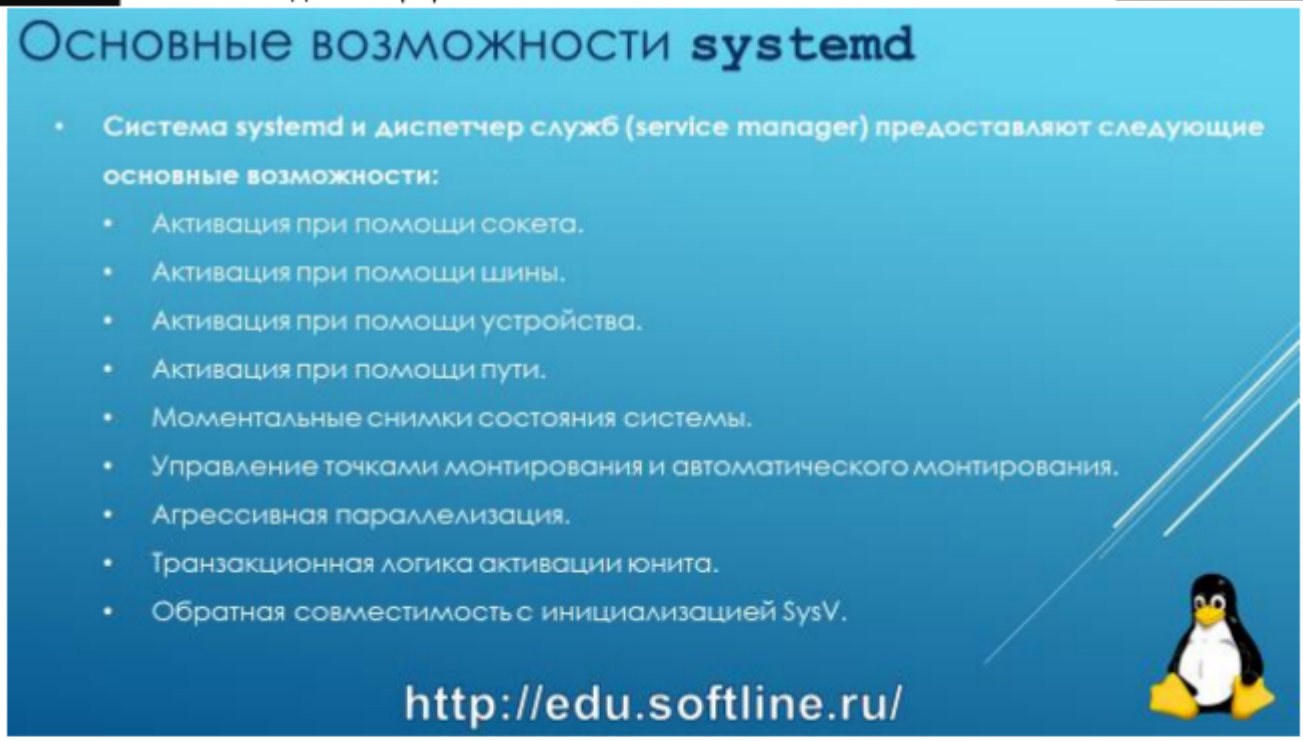
Доступные типы юнитов systemd (systemd Units):

Тип юнита (Unit Type)	Расширение файла File Extension)	Описание
Service unit	`.service`	Системная служба.
Target unit	`.target`	Группа системных служб.
Automount unit	`.automount`	Точка автоматического монтирования файловой системы.
Device unit	`.device`	Файл устройств, распознаваемых ядром.
Mount unit	`.mount`	Точка монтирования файловой системы.
Path unit	`.path`	Файл или каталог на файловой системе.
Scope unit	`.scope`	Процесс, созданный из вне.
Slice unit	`.slice`	Группа юнитов, организованные в иерархию, которые управляют системными процессами.
Snapshot unit	`.snapshot`	Сохраненное состояние диспетчера systemd.

Socket unit	`.socket`	Сокет коммуникаций внутри процесса.
Swap unit	`.swap`	Устройство или файл подкачки.
Timer unit	`.timer`	Таймер systemd.

Расположения юнитов systemd:

Каталог	Описание
`/usr/lib/systemd/system/`	Юниты **`systemd`**, распространяемые с устанавливаемыми RPM-пакетами.
`/run/systemd/system/`	Юниты **`systemd`**, созданные во время запуска. Этот каталог имеет приоритет перед каталогом с юнитами установленных служб.
`/etc/systemd/system/`	Юниты **`systemd`**, созданные и управляемые системным администратором. Этот каталог имеет приоритет перед каталогом с юнитами, созданными во время запуска.



В CentOS 7 (RHEL7) система systemd и диспетчер служб (Service Manager) предоставляют следующие основные возможности:

Socket-based activation (Активация при помощи сокета) – В процессе загрузки, systemd создает прослушиваемый сокет для всех системных служб, которые поддерживают этот тип активации, и передает этот сокет этим службам, как только они запустятся. Это не только позволяет sysytemd запускать службы параллельно, но также делает возможным перезапуск службы без потери сообщений, отправленных службе в процессе перезапуска (пока она недоступна): соответствующий сокет остается доступным и принимает в очередь все сообщения.

Systemd использует юниты сокета (Socket units) для активации при помощи сокета.

Bus-based activation (Активация при помощи шины) – Системные службы используют D-Bus для коммуникации внутри процесса, могут быть запущены по требованию, когда приложение-клиент впервые обращается для коммуникации с ним.

Systemd использует файл службы D-Bus для активации при помощи шины.

Device-based activation (Активация при помощи устройства) – Системные службы, которые поддерживают активацию при помощи устройства, могут быть запущены по требованию, когда устройство конкретного типа подключено и становится доступно.

Systemd используют юниты устройств для активации при помощи устройства.

Path-based activation (Активация при помощи пути) – Системные службы, которые поддерживают активацию при помощи пути, могут быть запущены когда конкретный файл или каталог изменяет свое состояние.

Systemd используют юниты пути для активации при помощи пути.

System state snapshots (Моментальные снимки состояния системы) – Systemd может временно сохранить текущее состояние всех юнитов или восстановить предыдущее состояние системы из динамически созданного моментального снимка юнитов.

Mount and automount point management (Управление точками монтирования и автоматического монтирования) – Systemd отслеживает и управляет точками монтирования и автоматического монтирования.

Aggressive parallelization (Агрессивная параллелизация) – За счет использования активации при помощи сокета, systemd может запускать системные службы параллельно, как только все прослушиваемые сокеты на месте. В комбинации с системными службами, которые поддерживают активацию по требованию, параллельная активация значительно сокращает время, необходимое для загрузки системы.

Transactional unit activation logic (Транзакционная логика активации юнита) – Прежде чем активировать или деактивировать юнит, systemd вычисляет его зависимости, создает временную транзакцию и проверяет ее консистенцию (целостность) транзакции. Если консистенция транзакции нарушена, systemd автоматически пробует скорректировать ее и удалить несущественные задания из нее, прежде чем сообщать об ошибке.

Backwards compatibility with SysV init (Обратная совместимость с инициализацией SysV) – Systemd полностью поддерживает скрипты инициализации SysV, как описано в Linux Standard Base Core Specification, что упрощает возможность обновления до юнитов служб systemd.

Изменения совместимости.

Система systemd и диспетчер служб (Service Manager) разработаны с учетом поддержки максимальной совместимости с инициализацией SysV и Upstart. Далее перечислены наиболее важные изменения с момента предыдущего релиза CentOS (RHEL):

Systemd имеет ограниченную поддержку уровней загрузки. Он предоставляет набор целевых юнитов/таргетов (Target units), которые могут быть напрямую связанны с уровнями загрузки для обеспечения совместимости, также он распространяется с командой **`runlevel`**. Не все таргеты (Target) могут быть напрямую связанны с уровнями загрузки, однако, как следствие команда **`runlevel`** может вернуть **`N`** для индикации неизвестного уровня загрузки. Рекомендуется по возможности отказаться от использования команды **`runlevel`**.

Утилита **`systemctl`** не поддерживает пользовательские команды. В дополнение к стандартным командам, таким как **`start`**, **`stop`** и **`status`**, авторы скриптов инициализации SysV могут использовать поддержку произвольного числа команд для обеспечения дополнительной функциональности. Например, скрипты инициализации **`iptables`** в CentOS 6 (RHEL6) может быть выполнен с командой **`panic`**, которая моментально включает «панический режим» (panic mode) и перенастраивает систему на сброс всех входящих и исходящих пакетов. Это не поддерживается в systemd и **`systemctl`** принимает только задокументированные команды.

Утилита **`systemctl`** не может взаимодействовать со службами, которые не были запущенны при помощи systemd. Когда systemd запускает системную службу, он сохраняет идентификатор основного процесса (PID), чтобы продолжить отслеживать его. Затем утилита **`systemctl`** использует PID для запросов и управления службой. Следовательно, если пользователь напрямую запускает демон из командной строки, **`systemctl`** не сможет определить его текущий статус или остановить его.

Systemd останавливает только запущенные службы. В CentOS 6 (RHEL6) и более ранних релизах, когда инициализируется последовательность выключения система использовала символьные ссылки,

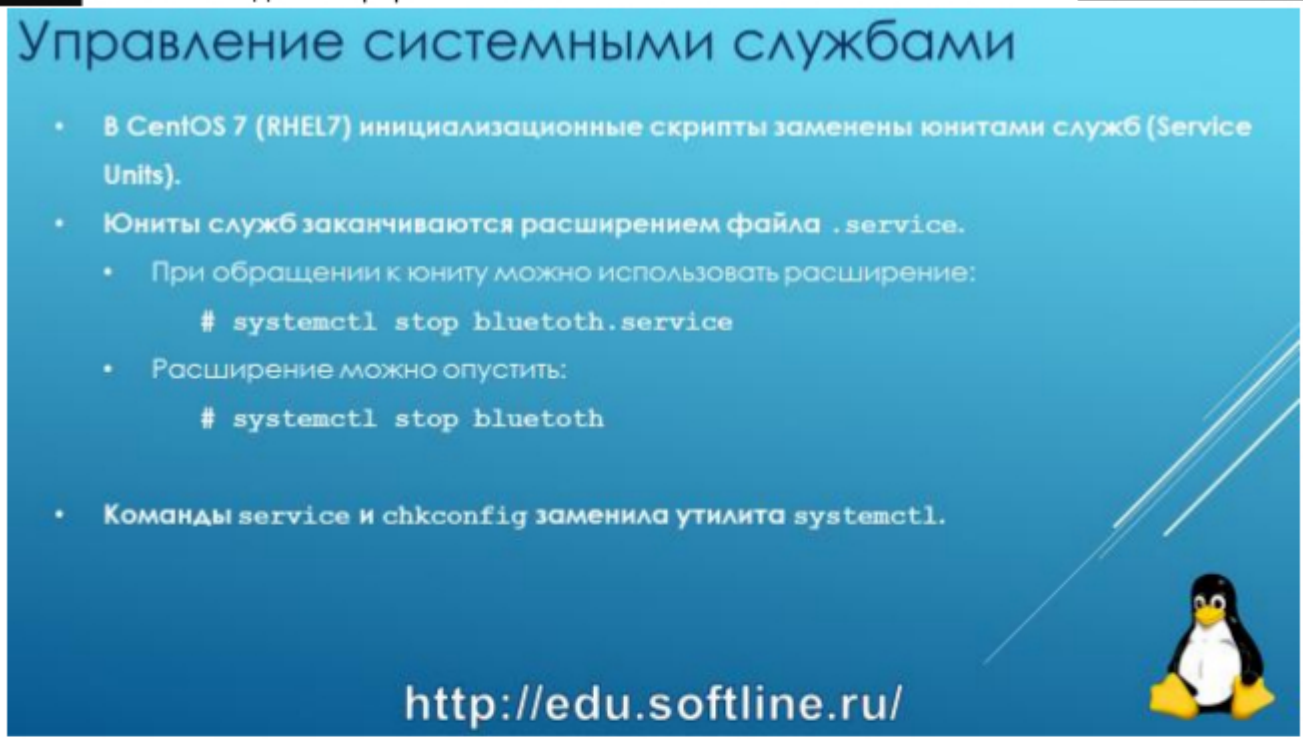
расположенные в каталоге **/etc/rc0.d/** для остановки всех доступных системных служб, вне зависимости от их состояния. С systemd только запущенные службы останавливаются в процессе выключения.

Системные службы не могут читать из стандартного входного потока. Когда systemd запускает службу, он подключает ее стандартный ввод к **/dev/null**, для предотвращения любых взаимодействий с пользователем.

Системные службы не наследуют никакого контекста (такого как переменные окружения **HOME** и **PATH**) от выполняющего их запуск пользователя и его сессии. Каждая служба запускается в чистом контексте выполнения.

Systemd читает информацию о зависимостях в заголовке LSB (Linux Standard Base) и интерпретирует ее в процессе запуска.

Все операции на юнитах служб (Service units) имеют таймаут в 5 минут, чтобы предотвратить возможность остановки системы из-за неисправной службы.



Предыдущие версии CentOS (RHEL), распространялись с SysV или Upstart, которые используют инициализационные скрипты (Init Sctipts), расположенные в каталоге **`/etc/rc.d/init.d/`**. Обычно эти инициализационные скрипты написаны на Bash и позволяют системному администратору контролировать состояние служб и демонов в его системе. В CentOS 7 (RHEL7) инициализационные скрипты заменены юнитами служб (Service Units).

Юниты служб заканчиваются расширением файла **`.service`** и служат той же цели, что и скрипты инициализации. Для просмотра, запуска, остановки, перезапуска, включения и отключения системных служб используйте команду **`systemctl`** как описано в таблице ниже. Команды **`service`** и **`chkconfig`** остались доступны в системе и работают корректно, но это сделано только для совместимости и рекомендуется отказаться от их использования.

ПРИМЕЧАНИЕ:

Для ясности, все примеры далее используют полные имена юнитов с расширением ***`.service`****, например:*

```
systemctl stop bluetoth.service
```

При работе с системными службами, можно пропустить расширение файла, для сокращения ввода. Когда утилита ***`systemctl`*** *встречает имя юнита без расширения файла, она автоматически принимает его как юнит службы (service unit). Следующая команда равнозначна, приведенной выше:*

```
systemctl stop bluetoth
```

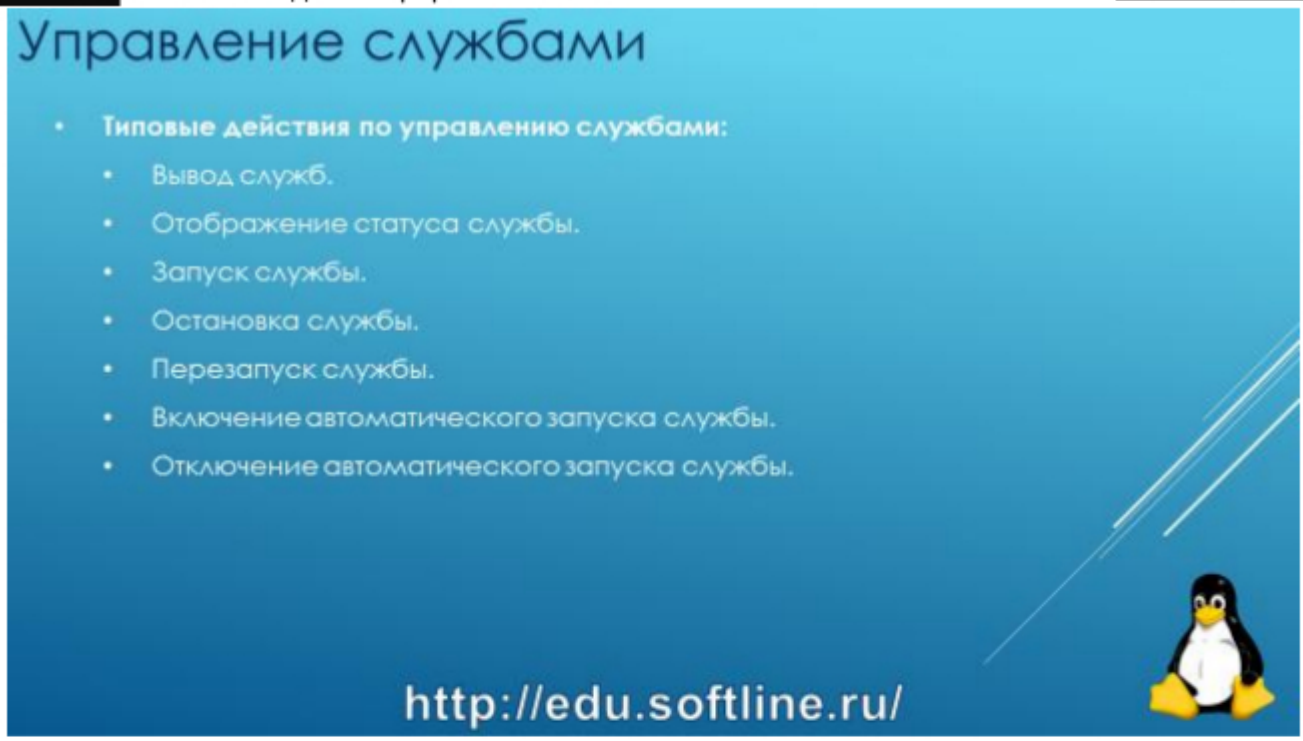
Таблица сравнения утилит **`service`** и **`systemctl`**:

service	systemctl
Запуск службы.	
`service name start`	`systemctl start name.service`
Остановка службы.	
`service name stop`	`systemctl stop name.service`
Перезапуск службы.	
`service name restart`	`systemctl restart name.service`
Перезапуск только запущенной службы.	
`service name condrestart`	`systemctl try-restart name.service`
Перезагрузка конфигурации	

`service name reload`	`systemctl reload name.service`
Проверка состояния службы.	
`service name status`	`systemctl status name.service` `systemctl is-active name.service`
Отображение статуса всех служб.	
`service --status-all`	`systemctl list-units --type service --all`

Таблица сравнения утилит **chkconfig** и **systemctl**:

chkconfig	systemctl
Включение автоматического запуска службы.	
`chkconfig name on`	`systemctl enable name.service`
Отключение автоматического запуска службы	
`chkconfig name off`	`systemctl disable name.service`
Проверка автоматического запуска службы.	
`chkconfig --list name`	`systemctl status name.service`
Вывод списка все служб с состоянием автоматического запуска.	
`chkconfig –list`	`systemctl list-unit-files --type service`



Вывод служб.

Для вывода всех загруженных юнитов служб, введите следующую команду:

```
systemctl list-units –type service
```

Для каждого юнита службы, эта команда отобразит его полное имя (**UNIT**), далее следует заметка о состоянии загрузки юнита (**LOAD**), его состояние высокого уровня (high-level) (**ACTIVE**) и низкого уровня (low-level) (**SUB**) и короткое описание (**DESCRIPTION**).

По умолчанию команда **systemctl list-units** отображает только активные юниты. Если вы хотите увидеть список всех загруженных юнитов вне зависимости от их состояния, запустите команду с опцией **–all** или **–a**.

```
# systemctl list-units --type service --all
```

Пример вывода загруженных на данный момент юнитов служб:

```
$ systemctl list-units --type service
UNIT                              LOAD     ACTIVE   SUB         DESCRIPTION
abrt-ccpp.service                 loaded   active   exited      Install ABRT coredump hook
abrt-oops.service                 loaded   active   running     ABRT kernel log watcher
abrt-vm core.service              loaded   active   exited      Harvest vmcores for ABRT
abrt-xorg.service                 loaded   active   running     ABRT Xorg log watcher
abrtd.service                     loaded   active   running     ABRT Automated Bug Reporting
Tool
...
systemd-vconsole-setup.service    loaded   active   exited      Setup Virtual Console
tog-pegasus.service               loaded   active   running     OpenPegasus CIM Server

LOAD    = Reflects whether the unit definition was properly loaded.
ACTIVE  = The high-level unit activation state, i.e. generalization of SUB.
SUB     = The low-level unit activation state, values depend on unit type.

46 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'
```

Также вы можете вывести список всех доступных юнитов служб, чтобы увидеть состояние их автозапуска. Используйте следующую команду:

```
# systemctl list-unit-files --type service
```

Пример вывода юнита служб с выводом типа автозапуска:

```
$ systemctl list-unit-files --type service
UNIT                            FILE STATE
abrt-ccpp.service               enabled
abrt-oops.service               enabled
abrt-vm core.service            enabled
abrt-xorg.service               enabled
abrtd.service                   enabled
...
wpa_supplicant.service          disabled
ypbind.service                  disabled

208 unit files listed.
```

Для каждого юнита службы эта команда отобразит его полное имя (UNIT FILE), далее будет указано состояние автоматического запуска (STATE).

Отображение статуса службы.

Для отображения подробной информации о юните службы, который связан с системной службой, введите следующую команду:

```
systemclt status name.service
```

Замените **name** на имя юнита службы информацию о которой хотите исследовать (например **gdm**). Эта команда выведет имя выбранного служебного юнита и его короткое описание, а также одно или более поле, описанных в таблице ниже. Если команда выполнена от имени пользователя **root**, также будут выведены последние записи журнала.

Доступная информация о юните службы.

Поле	Описание
Loaded	Информация о состоянии загрузки юнита (loaded), абсолютном пути к файлу юнита и отметка об автоматическом запуске службы.
Active	Информация о текущем состоянии запуска юнита службы с временной отметкой.
Main PID	PID соответствующей системной службы и ее имя.
Status	Дополнительная информация о системной службе.
Process	Дополнительная информация о связанном процессе.
CGroup	Дополнительная информация о связанной группе управления (Control Groups).

Чтобы определить только состояние запуска службы, используйте:

```
# systemctl is-active name.service
```

Чтобы определить только состояние автозапуска службы, используйте:

```
# systemctl is-enabled name.service
```

ПРИМЕЧАНИЕ:
Обе команды ***systemctl is-active*** *и* ***systemctl is-anbled****, возвращают выходной статус* ***0****.*

Пример вывода статуса службы.

Юнит службы GNOME Display Manager называется gdm.service. Для определения текущего статуса этого юнита службы, введите следующую команду:

```
# systemctl status gdm.service
gdm.service - GNOME Display Manager
   Loaded: loaded (/usr/lib/systemd/system/gdm.service; enabled)
   Active: active (running) since Thu 2013-10-17 17:31:23 CEST; 5min ago
 Main PID: 1029 (gdm)
   CGroup: /system.slice/gdm.service
           ├─1029 /usr/sbin/gdm
           ├─1037 /usr/libexec/gdm-simple-slave --display-id /org/gno...
           └─1047 /usr/bin/Xorg :0 -background none -verbose -auth /r...

Oct 17 17:31:23 localhost systemd[1]: Started GNOME Display Manager.
```

Запуск службы.

Для запуска юнита службы, который связан с системной службой, введите следующую команду:

```
# systemctl start name.service
```

Замените *name* именем юнита службы, которую вы хотите запустить (например, **gdm**). Эта команда запускает выбранный юнит службы в текущей сессии.

Пример запуска службы.

Служба Apache HTTP Server называется **httpd.service**. Для активации этого юнита службы запустите демон **httpd** в текущей сессии, выполните следующую команду, от имени пользователя root:

```
# systemctl start httpd.service
```

Остановка службы.

Для остановки юнита службы, который связан с системной службой, введите следующую команду:

```
# systemctl stop name.service
```

Замените *name* именем юнита службы, которую вы хотите остановить (например, **bluetoth**). Эта команда остановит выбранный юнит службы в текущей сессии.

Пример остановки службы.

Юнит службы для демона **bluetothd** называется **bluetoth.service**. Для активации этого юнита службы остановите демон **bluetothd** в текущей сессии, выполните следующую команду, от имени пользователя root:

```
# systemctl stop bluetoth.service
```

Перезапуск службы.

Для перезапуска юнита службы, который связан с системной службой, введите следующую команду от имени пользователя root:

```
# systemctl restart name.service
```

Замените *name* именем юнита службы, который хотите перезапустить (например, **httpd**). Эта команда остановит выбранный юнит службы в текущей сессии и немедленно запустите его снова. Нужно отметить, что, если текущий юнит службы не запущен, данная команда все равно его запустит. Для того чтобы команда сработала только в том случае, когда соответствующая служба запущена, выполните ее следующим образом от имени root:

```
# systemctl try-restart name.service
```

Некоторые службы поддерживают перезагрузку конфигурационных файлов без воздействия на выполнение службы, выполните следующую команду от имени пользователя root:

```
# systemctl reload name.service
```

Обратите внимание, что, если системна служба не поддерживает данную возможность, она проигнорирует всю команду целиком. Для удобства команда **systemctl** поддерживает команды **reload-or-restart** и **reload-or-try-restart**, которые перезапускают службы, не поддерживающие перезагрузку конфигурационных файлов.

Пример перезапуска службы.

Чтобы предотвратить получение ошибки пользователем служба Apache HTTP Server поддерживает возможность внесения изменения в конфигурацию без необходимости остановки службы. Для перезагрузки конфигурации демона **httpd** в текущей сессии, выполните следующую команду, от имени пользователя root:

```
# systemctl reload httpd.service
```

Включение автоматического запуска службы.

Для настройки юнита службы на автоматический старт в процессе загрузки, выполните следующую команду от имени пользователя root:

```
# systemctl enable name.service
```

Замените *name* именем юнита службы, который хотите настроить на автоматический запуск (например, **httpd**). Эта команда читает секцию **[Install]** выбранного юнита службы и создает необходимую символьную ссылку к файлу **/usr/lib/system/name.service** в каталоге **/etc/system/system/** и его подкаталогах. Однако эта команда не переписывает ссылку если такая уже существует. Если вы хотите убедиться, что ссылка вновь создана, используйте следующую команду от имени root:

```
# systemctl reenable name.service
```

Данная команда отключает выбранный юнит службы и вновь включает его снова.

Пример включения автоматического запуска службы.

Чтобы настроить автоматический запуск службы Apache HTTP Server в процессе загрузки, выполните следующую команду, от имени пользователя root:

```
# systemctl enable httpd.service
```

Отключение автоматического запуска службы.

Для отключения автоматического запуска юнита службы и соответствующей системной службы, введите следующую команду от имени пользователя root:

```
# systemctl disable name.service
```

Замените **name** именем юнита службы, который хотите отключить от автоматического запуска (например, **bluetooth**). Этак команда читает секцию **[Install]** выбранного юнита службы и удаляет необходимую символьную ссылку к файлу **/usr/lib/system/name.service** из каталога **/etc/system/system/** и его подкаталогах. Дополнительно вы можете замаскировать любой юнит службы, чтобы предотвратить его запуск вручную иной службой. Для этого используйте следующую команду от имени root:

```
# systemctl mask name.service
```

Данная команда заменит файл **/etc/system/system/name.service** на символьную ссылку к **/dev/null**, делая оригинальный файл недоступным для **systemd**. Чтобы отменить это действие снять маскировку с юнита службы, используйте следующую команду от имени пользователя root:

```
# systemctl unmask name.service
```

Пример отключения автоматического запуска службы.

Чтобы отключить автоматический запуск юнита службы **bluetooth.service** в текущей сессии, выполните следующую команду, от имени пользователя root:

```
# systemctl disable httpd.service
```



Предыдущие версии CentOS (RHEL), которые распространялись с SysV init или Upstart, использовали определенный заранее набор уровней загрузки (runlevel), которые представляли собой специфические режимы выполнения. Эти уровни загрузки нумеровались от 0 до 6 и определялись системными службами, которые должны быть запущены на соответствующем уровне загрузки. В CentOS 7 (RHEL7) концепция уровней загрузки была заменена таргетами systemd (*systemd targets*).

Таргеты systemd представлены в юнитах таргетов (*target units*). Юниты таргетов заканчиваются расширением файла **`.target`** и их основное назначение в группировке юнитов system в цепочку зависимостей. Например, юнит **`graphical.target`**, который используется для запуска графической сессии, запускает системные службы, таки как GNOME Display Managaer (**`gdm.service`**) или Account Service (**`account-daemon.service`**) и также активирует **`multi-user.target`**. Аналогично **`multi-user.target`** активирует прочие основные системные службы, такие как NetworkManager (**`NetworkManager.service`**) или D-Bus (**`dbus.service`**) и активирует еще один юнит таргета – **`basic.target`**.

CentOS 7 (RHEL7) распространяется с определенным заранее набором таргетов, которые отчасти похожи на стандартные уровни загрузки из предыдущего релиза системы. Для обратной совместимости таргеты распространяются я вместе с псевдонимами, напрямую связанными с уровнями загрузки SysV.

Уровень загрузки (Runlevel)	Целевые юниты (Target Units)	Описание
0	`runlevel0.target`, `poweroff.target`	Завершение работы и выключение системы.
1	`runlevel1.target`, `rescue.target`	Запуск аварийной оболочки.
2	`runlevel2.target`, `multiuser.target`	Запуск многопользовательской системы без графического окружения.
3	`runlevel3.target`, `multiuser.target`	Запуск многопользовательской системы без графического окружения.
4	`runlevel4.target`, `multiuser.target`	Запуск многопользовательской системы без графического окружения.

5	**runlevel5.target, graphical.target**	Запуск многопользовательской системы с графическим окружением
6	**runlevel6.target, reboot.target**	Завершение работы и перезагрузка системы.

Для просмотра, изменения или настройки таргетов systemd используйте утилиту **systemctl**. Команды **runlevel** и **telinit** остались доступны в системе только для совместимости, поэтому постарайтесь исключить их из работы.

Устаревшая команда	Новая команда	Описание
runlevel	**systemctl list-units --type target**	Список загруженных юнитов таргетов.
telinit runlevel	**systemctl isolate** name**.target**	Смена текущего таргета.



Просмотр таргета (Target) по умолчанию.

Чтобы определить какой юнит таргета используется по умолчанию выполните следующую команду:

```
# systemctl get-default
```

Данная команда проверяет назначение символьной ссылки **/etc/systemd/system/default.target** и отображает результат.

Пример просмотра таргета по умолчанию:

```
$ systemctl get-default
graphical.target
```

Просмотр текущего таргета (Target).

Для вывода списка всех загруженных на данный момент юнитов таргетов, введите следующую команду:

```
# systemctl list-units --type target
```

Для каждого юнита таргета эта команда отображает его полное имя [**UNIT**], указывая примечание о том, что юнит загружен [**LOAD**], его верхний уровень (high-level) [**ACTIVE**] и нижний уровень (low-level) [**SUB**] состояния активации юнита и короткое описание [**DESCRIPTION**]

По умолчанию команда **systemctl list-units** отображает только активные юниты. Если вы хотите вывести список всех загруженных юнитов вне зависимости от их состояния, выполните команду вместе с опицией **-all** или **-a**:

```
# systemctl list-units --type target --all
```

Пример просмотра текущего таргета.

```
$ systemctl list-units --type target
UNIT                    LOAD   ACTIVE SUB    DESCRIPTION
```

```
basic.target              loaded active active Basic System
cryptsetup.target         loaded active active Encrypted Volum es
getty.target              loaded active active Login Prom pts
graphical.target          loaded active active Graphical Interface
local-fs-pre.target       loaded active active Local File System s (Pre)
local-fs.target           loaded active active Local File System s
multi-user.target         loaded active active Multi-User System
network.target            loaded active active Network
paths.target              loaded active active Paths
remote-fs.target          loaded active active Rem ote File System s
sockets.target            loaded active active Sockets
sound.target              loaded active active Sound Card
spice-vdagentd.target     loaded active active Agent daem on for Spice guests
swap.target               loaded active active Swap
sysinit.target            loaded active active System Initialization
time-sync.target          loaded active active System Tim e Synchronized
timers.target             loaded active active Timers

LOAD = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB = The low-level unit activation state, values depend on unit type.

17 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'.
```

Смена таргета (Target) по умолчанию.

Для настройки системы на использование другого таргета по умолчанию, введите следующую команду от имени пользователя root:

```
# systemctl set-default name.target
```

Замените name именем юнита таргета, который вы хотите использовать по умолчанию (например, multi-user). Эта команда заменяет файл **/stc/systemd/system/default.target** символьной ссылкой к **/usr/lib/systemd/system/**name**.target**, где name это имя юнита таргета, который вы собираетесь использовать.

Пример смены таргета по умолчанию.

```
# systemctl set-default multi-user.target
rm '/etc/systemd/system/default.target'
ln -s '/usr/lib/systemd/system/multi-user.target'
'/etc/systemd/system/default.target'
```

Смена текущего таргета (Target).

Для переключения на другой таргет в рамках текущей сессии, используйте следующую команду от имени пользователя root:

```
# systemctl isolate name.target
```

Замените name на имя юнита таргета, который вы хотите использовать (например, multi-user). Эта команда запускает юнит таргета name и все зависимые юниты, все остальные останавливает.

Пример смены текущего таргета.

```
# systemctl isolate multi-user.target
```

Переход в аварийный режим (Rescue Mode).

Аварийный режим (Rescue Mode) – предоставляет удобный однопользовательское окружение, которое позволит вам восстановить вашу систему в случае невозможности загрузиться в нормальное состояние. В аварийном режиме система пытается смонтировать все локальные файловые системы и запускает только важные системные сервисы, но не запускает сетевые интерфейсы и не позволяет параллельный вход в систему. В CentOS 7 (RHEL 7) аварийный режим полностью соответствует однопользовательскому режиму (Single User Mode) и требует ввода пароля пользователя root.

Чтобы сменить текущий таргет и войти в аварийный режим в текущей сессии, введите следующую команду от имени пользователя root:

```
# systemctl rescue
```

Эта команда похожа на **systemctl isolate rescue.target**, но она также посылает информационное сообщение всем пользователям, которые на данный момент находятся в системе. Чтобы предотвратить отправку сообщения, введите команду с указанием опции **--no-wall**:

```
# systemctl --no-well rescue
```

Пример перехода в аварийное окружение.

```
~]# systemctl rescue
Broadcast m essage from root@ localhost on pts/0 (Fri 2013-10-25 18:23:15 CEST):
The system is going down to rescue m ode NOW!
```

Переход в спасательный режим (Emergency Mode).

Спасательный режим (Emergency Mode) – предоставляет наиболее минимальное окружение из возможных и позволяет вам восстанавливать вашу систему даже в ситуации, когда система не может войти в аварийное окружение (Rescue Mode). В спасательном окружении система монтирует корневую файловую систему только для чтения, не пробует монтировать остальные локальные файловые системы, не активирует сетевые интерфейсы и запускает только несколько ключевых сервисов. В CentOS 7 (RHEL 7) спасательный режим требует пароль пользователя root.

Для смены текщего таргета и входа в спасательный режим, введите следующую команду от имени пользователя root:

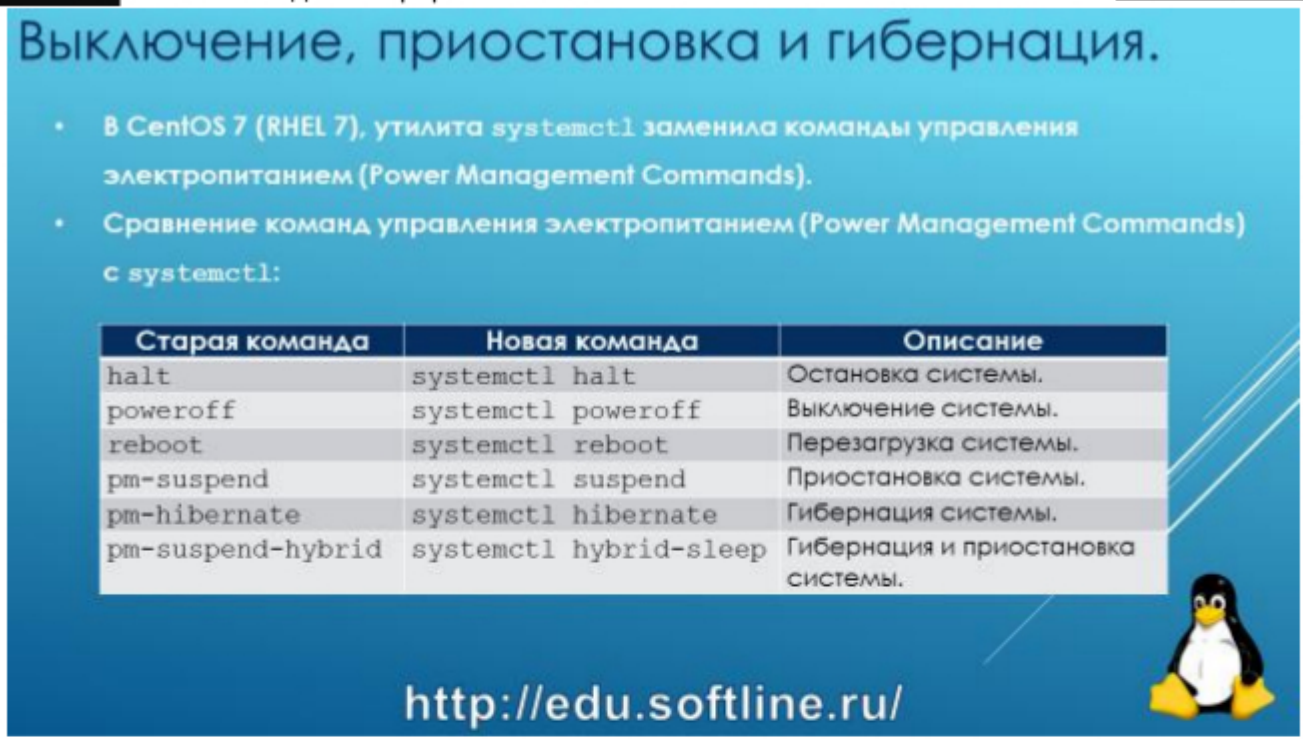
```
# systemctl emergency
```

Эта команда похожа на **systemctl isolate emergency.target**, но она также посылает информационное сообщение всем пользователям, которые на данный момент находятся в системе. Чтобы предотвратить отправку сообщения, введите команду с указанием опции **--no-wall**:

```
# systemctl --no-well emergency
```

Пример перехода в аварийное окружение.

```
# systemctl --no-wall emergency
```



В CentOS 7 (RHEL 7), утилита **systemctl** заменила команды управления электропитанием (Power Management Commands) используемыми в предыдущих версиях CentOS (RHEL). Оригинальные команды оставлены в составе системы для обеспечения совместимости, но постарайтесь от них отказаться в пользу **systemctl** если это возможно.

Сравнение команд управления электропитанием (Power Management Commands) с **systemctl**.

Старая команда	Новая команда	Описание
halt	**systemctl**	Остановка системы.
poweroff	**systemctl poweroff**	Выключение системы.
reboot	**systemctl reboot**	Перезагрузка системы.
pm-suspend	**systemctl suspend**	Приостановка системы.
pm-hibernate	**systemctl hibernate**	Гибернация системы.
pm-suspend-hybrid	**systemctl hybrid-sleep**	Гибернация и приостановка системы.

Выключение системы.

Для выключения системы введите следующую команду с правами пользователя root:

```
# systemctl poweroff
```

Для выключения и остановки системы без выключения машины введите следующую команду от имени пользователя root:

```
# systemctl halt
```

По умолчанию обе эти команды заставляют systemd отправить информационное сообщение всем пользователям, находящимся в системе. Чтобы предотвратить отправку сообщений, выполните команду с указанием опции **--no-wall**:

```
# systemctl --no-wall poweroff
```

Перезагрузка системы.

Для перезагрузки системы введите следующую команду с правами пользователя root:

```
# systemctl reboot
```

По умолчанию эта команды заставляют systemd отправить информационное сообщение всем пользователям, находящимся в системе. Чтобы предотвратить отправку сообщений, выполните команду с указанием опции **--no-wall**:

```
# systemctl --no-wall reboot
```

Приостановка системы.

Для приостановки системы, введите следующую команду от имен пользователя root:

```
# systemctl suspend
```

Эта команда сохраняет состояние системы в оперативной памяти и за исключением модуля оперативной памяти выключает устройства машины. Когда вы вновь включаете машину, система восстанавливает состояние без необходимости выполнять процесс загрузки. Потому как состояние системы сохранено в оперативной памяти, а не на жестком диске, восстановление системы из происходит значительно быстрее, чем при гибернации. С другой стороны, приостановка системы чувствительна к потри электропитания.

Гибернация системы.

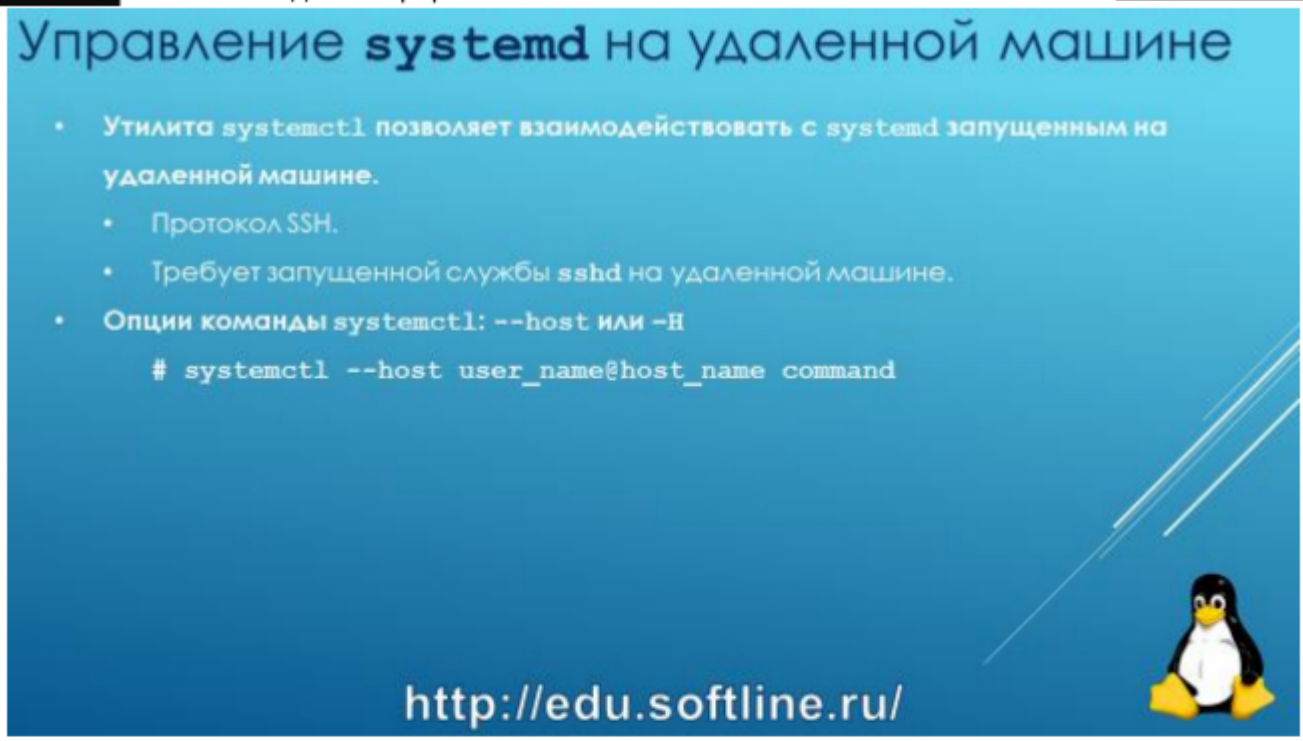
Для гибернации системы, введите следующую команду от имен пользователя root:

```
# systemctl hibernate
```

Эта команда сохраняет состояние системы на жестком диске и выключает машину. Когда вы вновь включаете машину, система восстанавливает состояние из данных, сохраненных на диске, без необходимости загрузки. Так как данные сохранены на жестком диске, машина не нуждается в энергоснабжении, но этот процесс занимает больше времени чем восстановление из состояния приостановки.

Для одновременной гибернации и приостановки системы введите следующую команду от имени пользователя root:

```
# systemctl hybrid-sleep
```



В дополнение к управлению системой systemd и службами локально, утилита **systemctl** также позволяет вам взаимодействовать с systemd запущенным на удаленной машине по протоколу SSH. При условии, что служба **sshd** на удаленной машине запущена, вы можете подключиться к этой машине при помощи команды **systemctl** с указанием опций **--host** или **-H**:

```
# systemctl --host user_name@host_name command
```

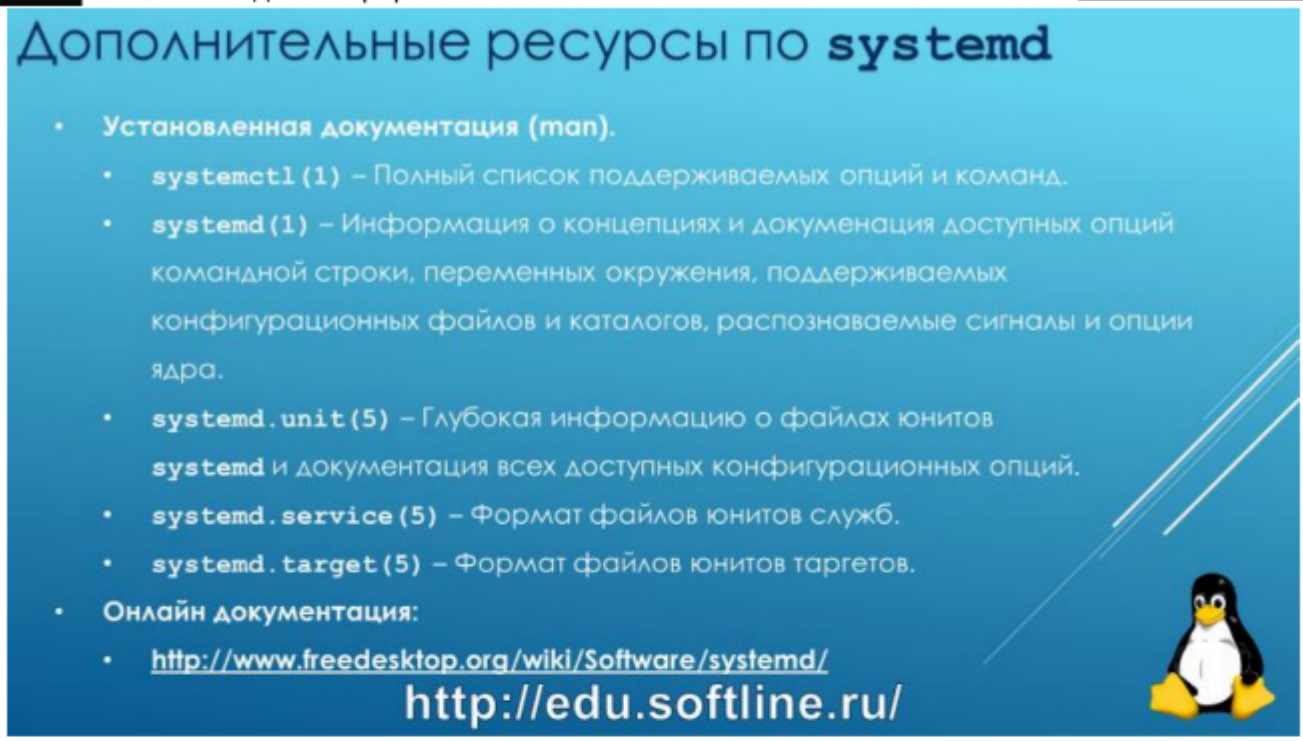
Замените *user_name* именем удаленного пользователя, *host_name* именем хоста и *command* любой командой **systemctl**, описанной выше.

ПРИМЕЧАНИЕ:
Удаленная машина должна быть сконфигурирована для разрешения пользователю подключаться удаленно при помощи сервера SSH.

Пример удаленного управления.

Для входа на удаленную машину **c7-server01.ll-100.local**, от имени пользователя root и определения текущего статуса юнита **httpd.service**, введите следующую команду:

```
$ systemctl -H root@c7-server01.ll-100.local status httpd.service
root@c7-server01.ll-103.local's password:
httpd.service - The Apache HTTP Server
Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled)
Active: active (running) since Fri 2013-11-01 13:58:56 CET; 2h 48min ago
Main PID: 649
Status: "Total requests: 0; Current requests/sec: 0; Current traffic: 0 B/sec"
CGroup: /system.slice/httpd.service
```



Для получения дополнительной информации о systemnd и его применении в CentOS 7 (RHEL 7), воспользуйтесь следующими ресурсами:

Установленная документация.

- **`systemctl`**(1) – Страница руководства для команды **`systemctl`** содержит полный список поддерживаемых опций и команд.
- **`systemd`**(1) – Страница руководства для системы systemd и диспетчера служб содержит больше информации о концепциях и документирует доступные опции командной строки, переменные окружения, поддерживаемые конфигурационные файлы и каталоги, распознаваемые сигналы и опции ядра.
- **`systemd.unit`**(5) – Страница руководства содержит глубокую информацию о файлах юнитов systemd и документирует все доступные конфигурационные опции.
- **`systemd.service`**(5) – Страница руководства содержит формат файлов юнитов служб (Service Units).
- **`systemd.target`**(5) – Страница руководства содержит формат файлов юнитов таргетов (Target Units).

Онлайн документация:

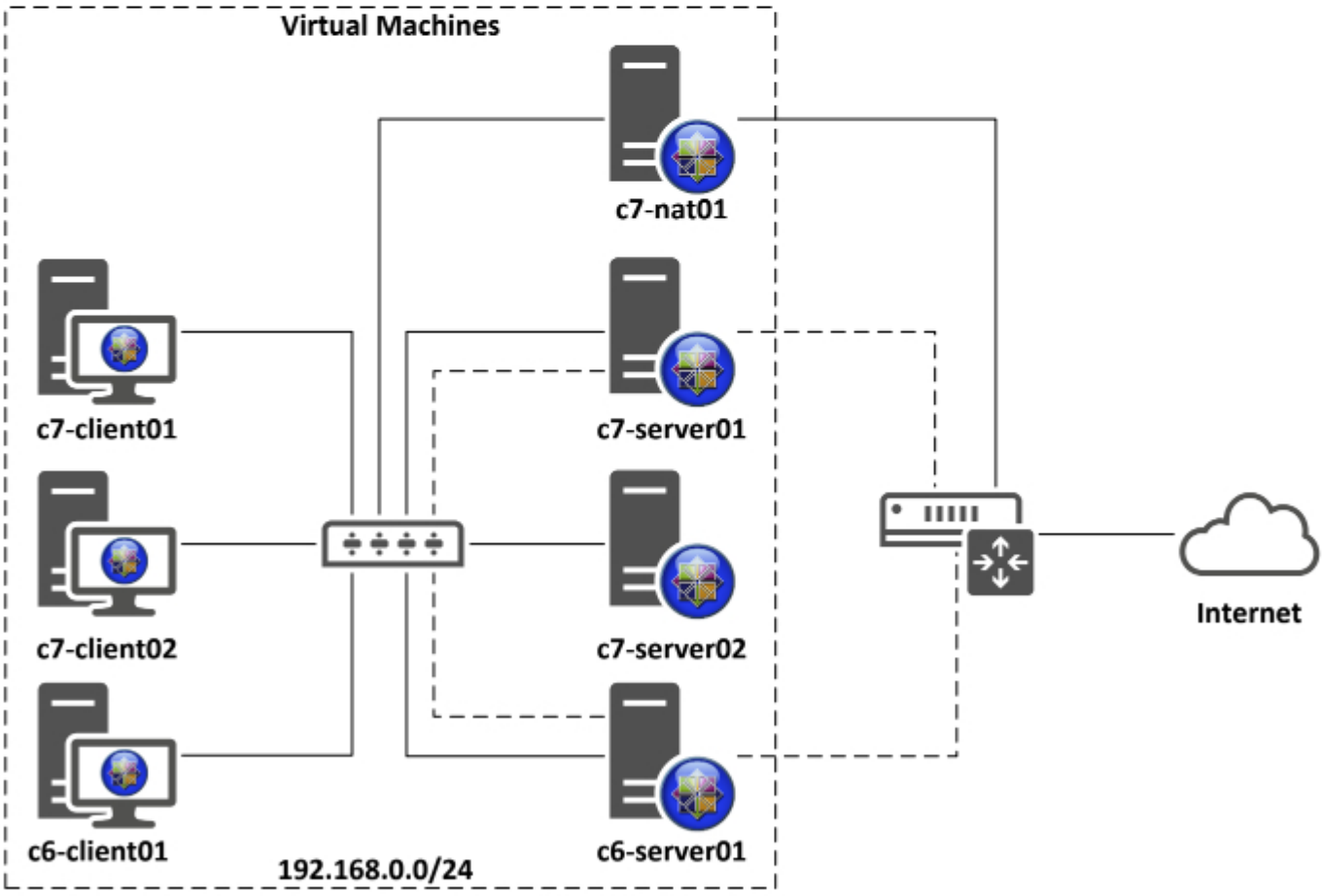
http://www.freedesktop.org/wiki/Software/systemd/ - домашняя страница проекта, содержит больше информации о systemd.

Лабораторная работа

Лабораторная работа 2

- **Упражнения:**
  - Управление загрузчиком GRUB2 в CentOS 7.
  - Управление автоматическим запуском и таргетами в CentOS 7.
- **Необходимые машины:**
  - `c7-nat01`.
- **Учетные данные:**
  - Пользователь: `root`.
  - Пароль: **`Pa$$w0rd`**.

http://edu.softline.ru/





Упражнение 1 – Управление загрузчиком GRUB2 в CentOS 7.

ПРИМЕЧАНИЕ:
Перед началом выполнения заданий упражнения 3 не забудьте сделать резервную копию всех файлов из каталога ***/etc/grub.d/****.*

1. Подключитесь к виртуальной машине c7-server01.
2. Увеличьте время ожидания на экране выбора операционной системы.
3. Добавьте настраиваемую загрузочную запись в конфигурацию GRUB2.

ПРИМЕЧАНИЕ:
Убедитесь в наличии соответствующего ядра и начального RAM-диска в каталоге ***/boot/****.*

4. Пересоздайте конфигурацию загрузчика при помощи команды **`grub2-mkconfig`**.
5. Перезагрузите систему, используйте для загрузки вновь созданную загрузочную запись.
6. Обновите ядро системы.
7. Убедитесь, что новая загрузочная запись для нового ядра системы успешно добавлена, при этом настраиваемая загрузочная запись осталась в системе.



Ответы к упражнению 1 – Управление загрузчиком GRUB2 в CentOS 7.

ПРИМЕЧАНИЕ:
Перед началом выполнения заданий упражнения 3 не забудьте сделать резервную копию всех файлов из каталога ***/etc/grub.d/****.*

1. Подключитесь к виртуальной машине c7-server01.
2. Увеличьте время ожидания на экране выбора операционной системы.

```
# vi /etc/default/grub
GRUB_TIMEOUT=15
```

3. Добавьте настраиваемую загрузочную запись в конфигурацию GRUB2.

```
# vi /etc/grub.d/40_custom
menuentry 'Custom Linux Boot' {
        load_video
        set gfxpayload=keep
        insmod gzio
        insmod part_msdos
        insmod xfs
        set root='hd0,msdos1'
        linux16 /vmlinuz-3.10.0-862.el7.x86_64 root=/dev/mapper/centos-root
ro rd.lvm.lv=centos/root rd.lvm.lv=centos/swap
        initrd16 /initramfs-3.10.0-862.el7.x86_64.img
}
```

ПРИМЕЧАНИЕ:
Убедитесь в наличии соответствующего ядра и начального RAM-диска в каталоге ***/boot/****.*

4. Пересоздайте конфигурацию загрузчика при помощи команды **grub2-mkconfig**.

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

5. Перезагрузите систему, используйте для загрузки вновь созданную загрузочную запись.

```
# reboot
```

6. Обновите ядро системы.

```
# yum update kernel -y
```

7. Убедитесь, что новая загрузочная запись для нового ядра системы успешно добавлена, при этом настраиваемая загрузочная запись осталась в системе.



Упражнение 2. Управление автоматическим запуском и таргетами в CentOS 7.

Упражнение выполняйте на машине **c7-client01** или **c7-client02**.

1. С помощью команды **`systemctl`** проверьте статус демона **`cups`**.
2. С помощью команды **`systemctl`** проверьте настройку автоматического запуска демона **`cups`**.
3. С помощью команды **`systemctl`** остановите демон **`cups`**.
4. С помощью команды **`systemctl`** отключите автоматический запуск демона **`cups`**.
5. Перезапустите систему и убедитесь, что демон **`cups`** не запустился.
6. Смените таргет системы на многопользовательский.
7. Перезапустите систему и убедитесь, что загружен многопользовательский таргет.
8. Верните все настройки в исходное состояние.



Ответы к упражнению 2. Управление автоматическим запуском и таргетами в CentOS 7.

Упражнение выполняйте на машине **c7-client01**.

1. С помощью команды **systemctl** проверьте статус демона **cups**.

```
# systemctl status cups
```

2. С помощью команды **systemctl** проверьте настройку автоматического запуска демона **cups**.

```
# systemctl is-enabled cups
```

3. С помощью команды **systemctl** остановите демон **cups**.

```
# systemctl stop cups
```

4. С помощью команды **systemctl** отключите автоматический запуск демона **cups**.

```
# systemctl disable cups
```

5. Перезапустите систему и убедитесь, что демон **cups** не запустился.

```
# systemctl status cups
…
# systemctl status cups
```

6. Смените таргет системы на многопользовательский.

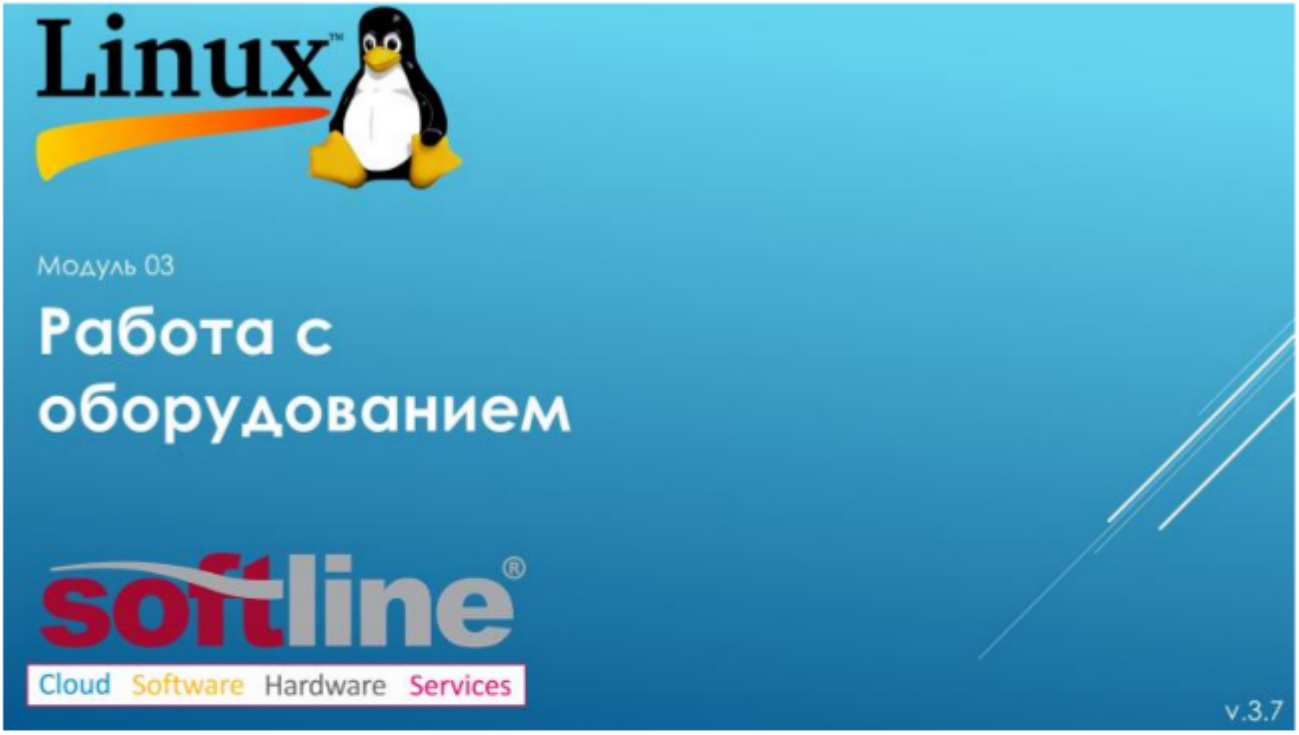
```
# systemctl set-default multi-user.target
```

7. Перезапустите систему и убедитесь, что загружен многопользовательский таргет.

```
# systemctl status cups
…
# systemctl get-default
```

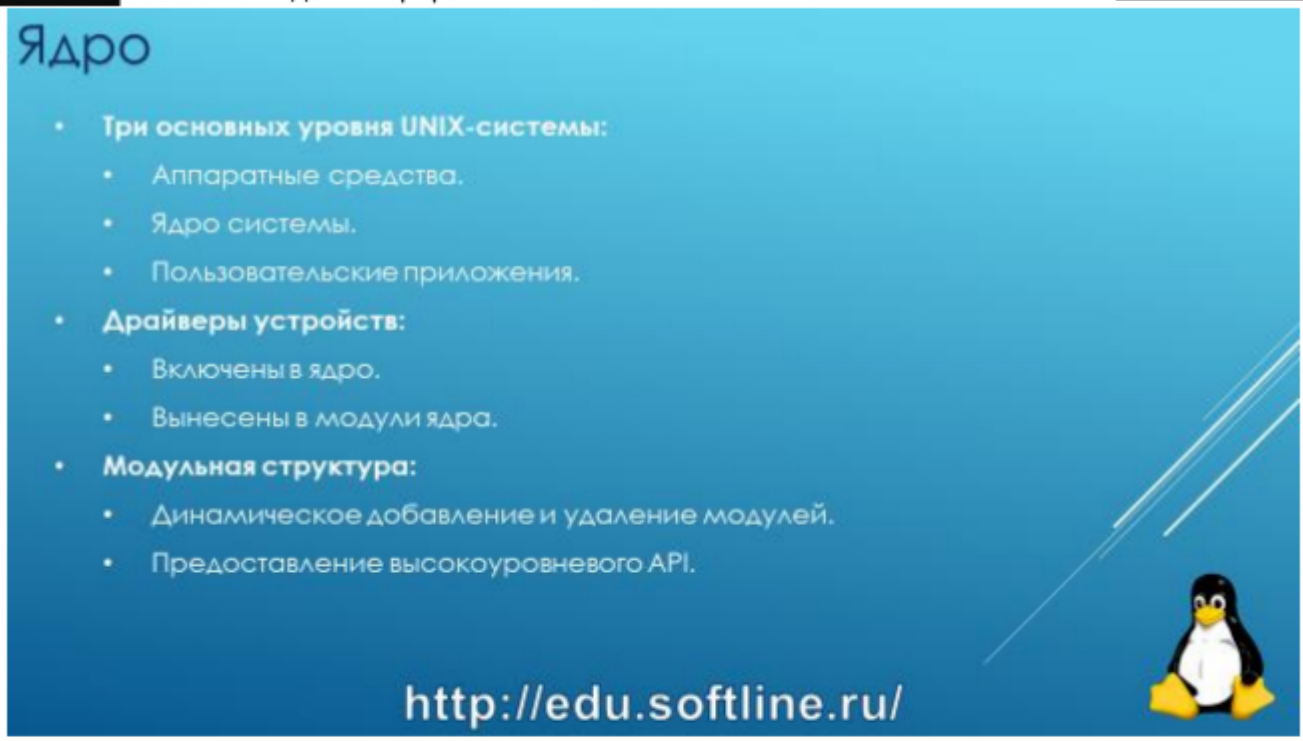
8. Верните все настройки в исходное состояние.

Модуль 03: Работа с оборудованием



В модуле представлены:

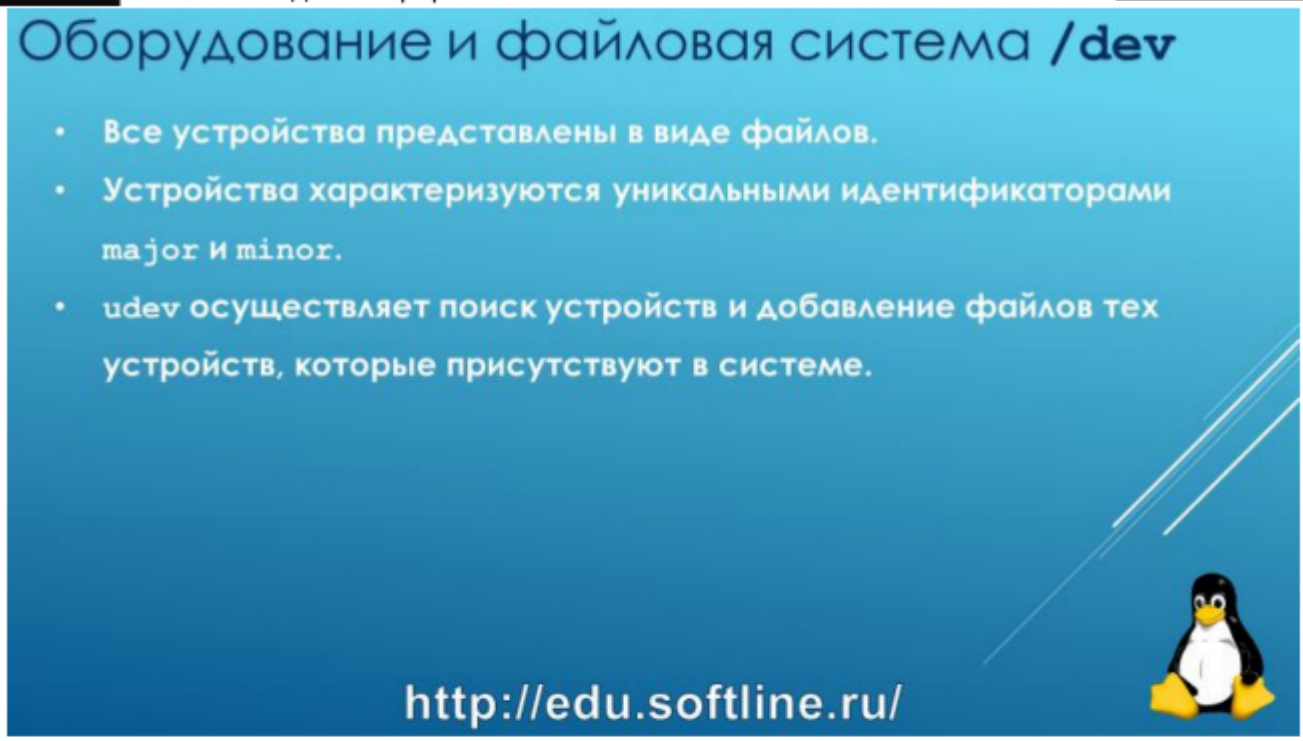
- Поиск оборудования.
- Подключение и настройка.
- Файловая система **`/dev`**.
- Файловая система **`/proc`**.
- Управление модулями ядра.



Большая часть ядра Linux – это драйверы, с помощью которых происходит процесс общения с аппаратными устройствами. Ядро скрывает этот процесс, предоставляя высокий уровень абстракции с помощью API, который предоставляет программистам и пользователям некоторые общие концепции, присущие определённому типу устройств, тем самым освобождая программы от задачи общения с разнообразными устройствами.

Ядро содержит драйверы устройств, которые управляют отдельными элементами аппаратного уровня. Учитывая, что Linux поддерживает огромное множество самого разнообразного оборудования, встаёт вопрос о размере ядра и его быстродействии. Ответом на этот вопрос служит модульная структура ядра. Драйверы могут быть вынесены в отдельные модули, что позволяет оставить в ядре только самый необходимый функционал, а, следовательно, уменьшить размер ядра и увеличить его быстродействие.

Модули могут быть загружены и выгружены динамически, не требуя останова системы.



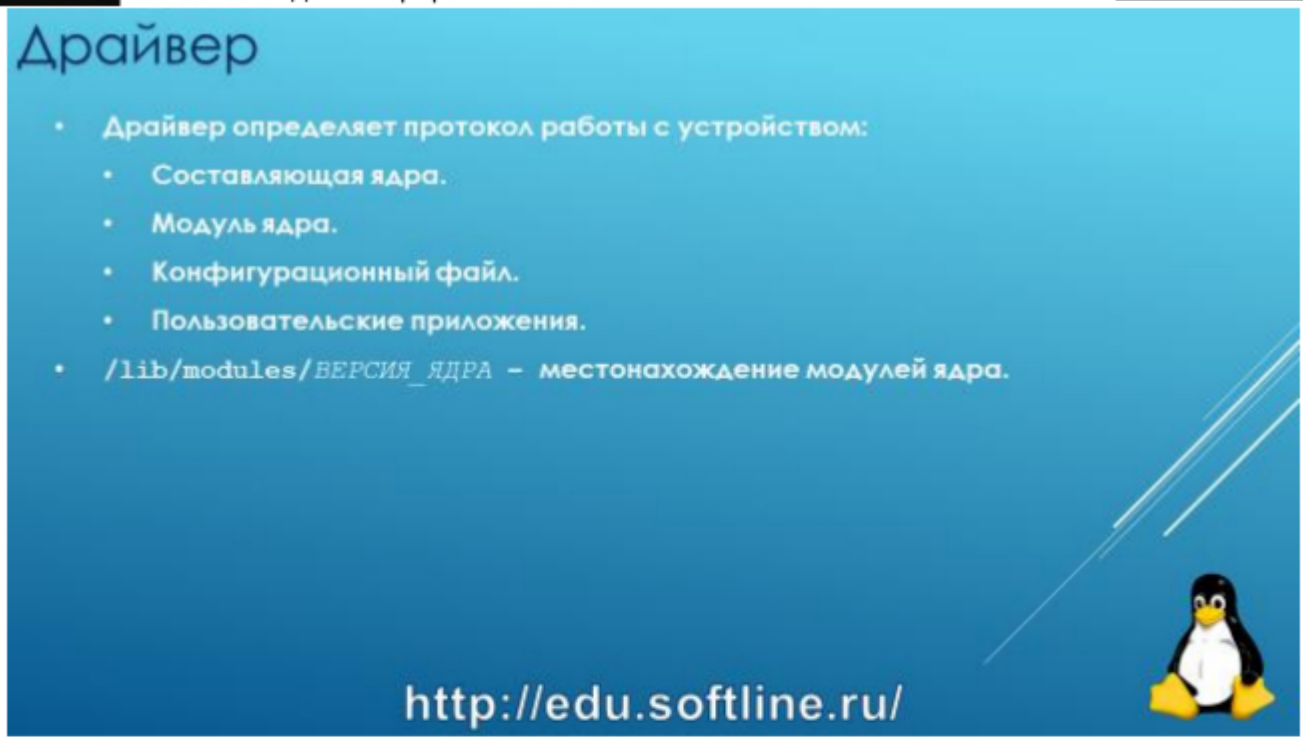
Драйвер устройства – это программа, которая обеспечивает взаимодействие системы с определённым компонентом аппаратной среды. Драйвер выполняет роль переводчика команд конкретного устройства на "язык" API-функций ядра.

Драйверы являются компонентами ядра, а не пользовательскими процессами. Несмотря на это, ядро предоставляет доступ к устройствам программам пользовательского уровня. Для этого существует специальная виртуальная файловая система **/dev**, в которой создаются специальные файлы устройств, ядро преобразует все операции над этими файлами в обращения к драйверу.

Каждый файл устройства в каталоге **/dev** имеет старший и младший номер устройства (**major** и **minor**). Старший номер устройства обозначает драйвер, к которому относится этот файл, а младший обозначает конкретный тип устройства. Посмотреть номера устройств можно с помощью команды ls, в поле, в котором для обычных файлов указывается размер, будут указаны номера устройства:

```
$ ls -l /dev/sda /dev/sda1
brw-r----- 1 root disk 8, 0 Sep 16 10:45 /dev/sda
brw-r----- 1 root disk 8, 1 Sep 16 10:46 /dev/sda1
```

Существует два типа специальных файлов: блок-ориентированные и байт-ориентированные. Чтение из блок-ориентированного устройства производится по одному блоку (как правило, кратному 512) за раз, тогда как чтение из байт-ориентированного устройства производится побайтно



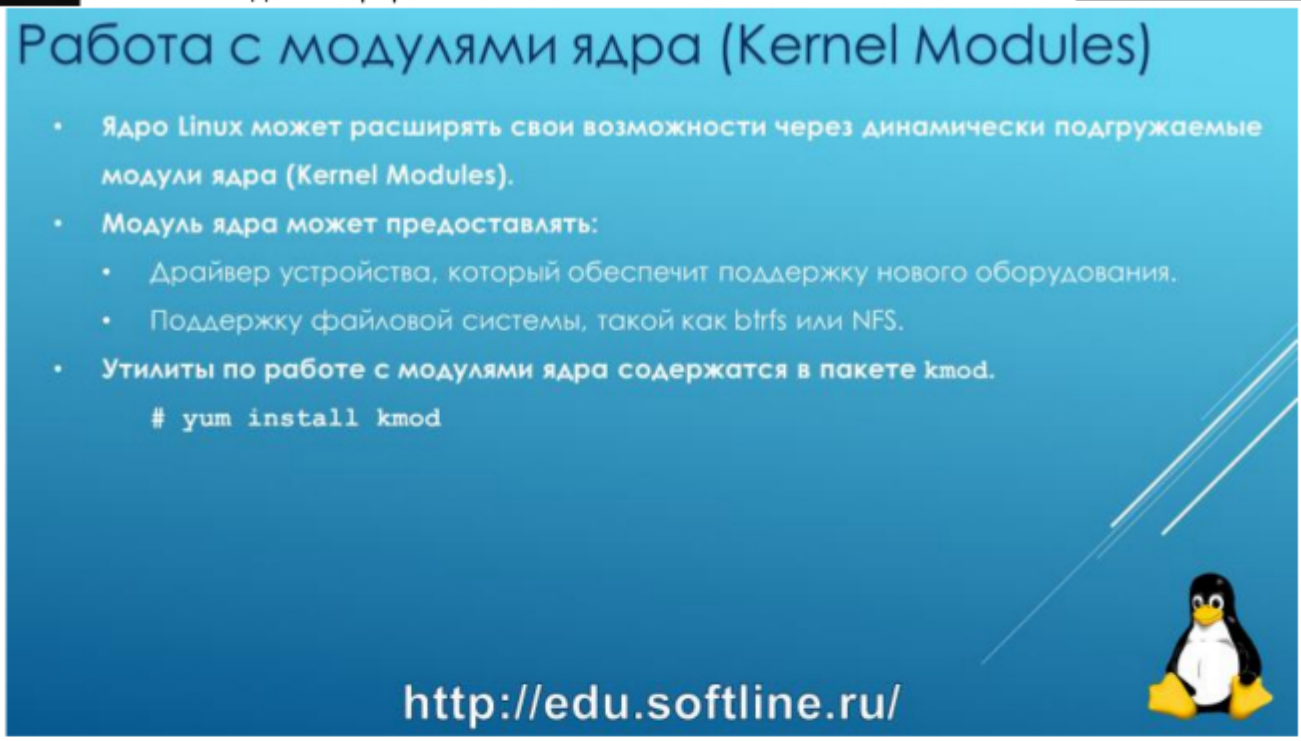
Драйвер – неотъемлемая часть операционной системы, благодаря драйверам ядро может "разговаривать" с устройствами на одном языке.

Ядро Linux включает в себя огромное множество драйверов, позволяющих работать с самым разнообразным оборудованием. Учитывая модульную структуру ядра, совсем не обязательно расходовать ресурсы на всевозможные драйверы, из которых необходимы единицы. Драйверы могут быть вынесены в отдельные модули, загружаемые ядром по мере необходимости.

Не всегда драйверы являются составляющей частью ядра. Протоколом, определяющим работу с устройством, может быть и простой текстовый файл, например, PPD-файлы, определяющие протокол взаимодействия с принтерами.

Все драйверы (модули ядра) находятся в каталоге **/lib/kernel/***ВЕРСИЯ_ЯДРА*:

```
$ ls -l /lib/modules/2.6.32-279.el6.i686/kernel
drwxr-xr-x  3 root root 4096 Aug  9 10:34 arch
drwxr-xr-x  2 root root 4096 Aug  9 10:34 crypto
drwxr-xr-x 40 root root 4096 Aug  9 10:34 drivers
drwxr-xr-x 45 root root 4096 Aug  9 10:34 fs
drwxr-xr-x  4 root root 4096 Aug  9 10:34 lib
drwxr-xr-x 25 root root 4096 Aug  9 10:34 net
drwxr-xr-x  9 root root 4096 Aug  9 10:34 sound
```



Ядро Linux является модульным, это значит, оно может расширять свои возможности через динамически подгружаемые модули ядра (Kernel Modules).

Модуль ядра может предоставлять:

- Драйвер устройства, который обеспечит поддержку нового оборудования.
- Поддержку файловой системы, такой как btrfs или NFS.

Как и само ядро, модули могут принимать параметры, которые настраивают их поведение. В большинстве случаев параметры по умолчанию работают достаточно хорошо, чтобы не пришлось их менять. Инструменты пользовательского пространства могут выводить список модулей, загруженных в запущенное ядро; запрашивать все доступные модули, доступные параметры модулей и информацию о модуле; динамически подгружать и выгружать (удалять) модули в или из запущенного ядра. Многие утилиты, которые предоставляются пакетом **kmod**, обладают способностью обрабатывать зависимости модулей при выполнении операций, что не требует ручного отслеживания зависимостей.

На современных системах модули ядра автоматически загружаются при помощи различных механизмов при появлении потребности в их вызове. Однако, возможны ситуации, при которых необходимо загружать и выгружать модули вручную, например, когда один модуль предпочтительнее другого, хотя оба предоставляют одинаковую функциональность, или, когда модуль функционирует некорректно.

Далее будет рассмотрено следующее:

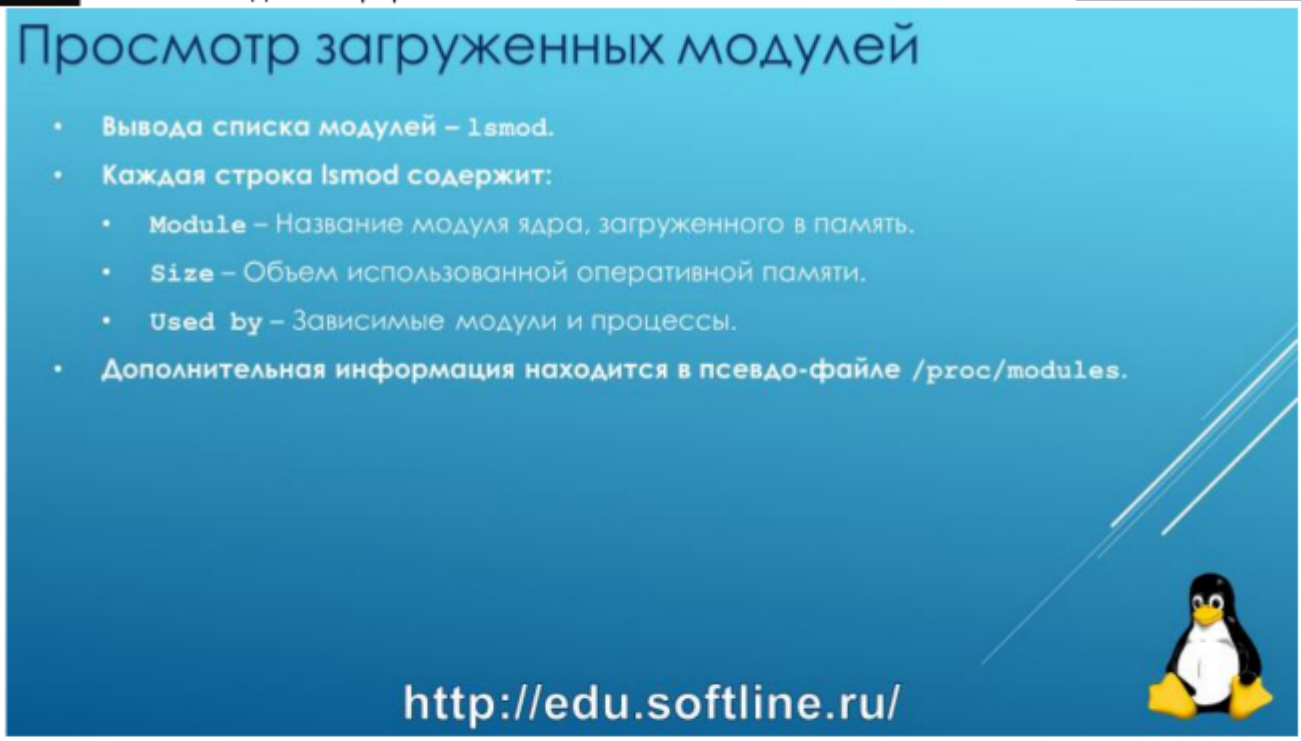
- Использование утилит пользовательского пространства **kmod** для отображения, запросов, загрузки и выгрузки модулей ядра и их зависимостей.
- Динамическая установка параметров при помощи командной строки и на постоянной основе, для настройки поведения модулей.
- Загрузка модулей ядра в процессе загрузки системы.



ПРИМЕЧАНИЕ:

Для того чтобы использовать утилиты по работе с модулями ядра, описанные далее, в первую очередь необходимо убедиться, что пакет **`kmod`** *установлен на системе:*

```
# yum install kmod
```



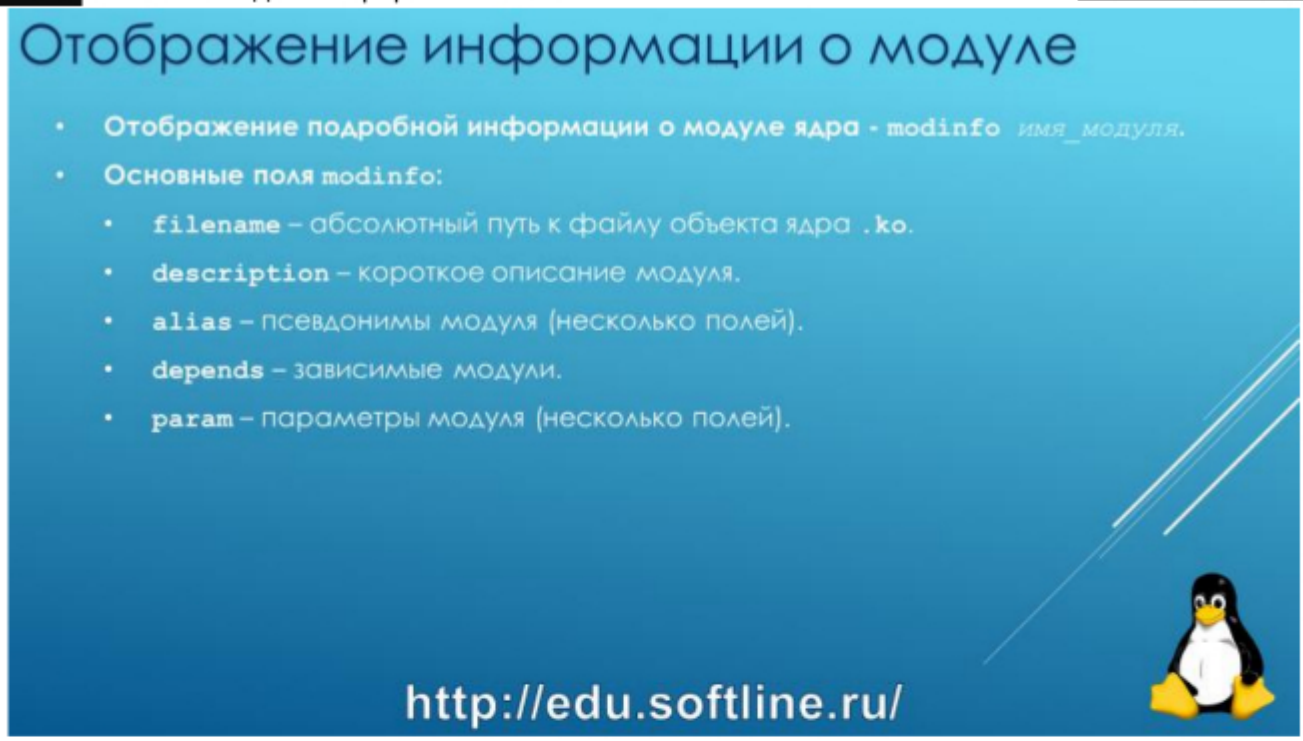
Для вывода списка модулей, которые загружены в ядро, воспользуйтесь командой **lsmod**:

```
# lsmod
Module                  Size  Used by
ip6t_rpfilter          12546  1
ip6t_REJECT            12939  2
ipt_REJECT             12541  2
xt_conntrack           12760  23
ebtable_nat            12807  0
ebtable_broute         12731  0
bridge                115385  1 ebtable_broute
stp                    12976  1 bridge
llc                    14552  2 stp,bridge
ebtable_filter         12827  0
ebtables               30913  3 ebtable_broute,ebtable_nat,ebtable_filter
ip6table_nat           12864  1
nf_conntrack_ipv6      18738  13
nf_defrag_ipv6         34651  1 nf_conntrack_ipv6
nf_nat_ipv6            14131  1 ip6table_nat
ip6table_mangle        12700  1
ip6table_security      12710  1
ip6table_raw           12683  1
ip6table_filter        12815  1
…
```

Каждая строка **lsmod** содержит:

- **Module** – Название модуля ядра, загруженного в память.
- **Size** – Объем использованной оперативной памяти.
- **Used by** – Количество процессов, которые используют модуль и прочие модули, зависящие от этого, далее следует список имен этих модулей. При помощи этого списка, вы можете в первую очередь выгрузить все модули, зависящие от модуля, который вы хотите выгрузить.

Вывод **lsmod** менее подробный и более простой для восприятия, чем содержимое псевдо-файла **/proc/modules**.



Вы можете отобразить подробную информацию о модуле ядра при помощи команды **modinfo** *имя_модуля*.

ПРИМЕЧАНИЕ:

*При вводе имени модуля в качестве аргумента утилиты **kmod**, не нужно добавлять расширение **.ko** в конец имени. Имена модулей ядра не имеют расширений, в отличии от их файлов.*

Для отображения информации о модуле **e1000e**, который является сетевым драйвером Intel PRO/1000, введите следующую команду, от имени пользователя root:

```
# modinfo e1000e
filename:       /lib/modules/3.10.0-
229.1.2.el7.x86_64/kernel/drivers/net/ethernet/intel/e1000e/e1000e.ko
version:        2.3.2-k
license:        GPL
description:    Intel(R) PRO/1000 Network Driver
author:         Intel Corporation, <linux.nics@intel.com>
rhelversion:    7.1
srcversion:     0EED9FB230041A33B6B9BA9
alias:          pci:v00008086d000015A3sv*sd*bc*sc*i*
…
alias:          pci:v00008086d000010A4sv*sd*bc*sc*i*
alias:          pci:v00008086d0000105Fsv*sd*bc*sc*i*
alias:          pci:v00008086d0000105Esv*sd*bc*sc*i*
depends:        ptp
intree:         Y
vermagic:       3.10.0-229.1.2.el7.x86_64 SMP mod_unload modversions
signer:         CentOS Linux kernel signing key
sig_key:        34:B5:BC:A2:B7:06:D8:2E:72:A5:BE:3E:E4:09:BE:C7:19:5E:A5:08
sig_hashalgo:   sha256
parm:           debug:Debug level (0=none,...,16=all) (int)
parm:           copybreak:Maximum size of packet that is copied to a new buffer on
receive (uint)
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:           IntMode:Interrupt Mode (array of int)
parm:           SmartPowerDownEnable:Enable PHY smart power down (array of int)
parm:           KumeranLockLoss:Enable Kumeran lock loss workaround (array of int)
parm:           WriteProtectNVM:Write-protect NVM [WARNING: disabling this can lead
to corrupted NVM] (array of int)
parm:           CrcStripping:Enable CRC Stripping, disable if your BMC needs the
CRC (array of int)
```

Описание основных полей вывода **modinfo**:

- **filename** – абсолютный путь к файлу объекта ядра **.ko**. Для вывода только этой информации, воспользуйтесь командой **modinfo -n**.
- **description** – короткое описание модуля. Для вывода только этой информации, воспользуйтесь командой **modinfo -d**.
- **alias** – поле alias представлено столько раз, сколько псевдонимов есть у модуля, если у модуля нет псевдонимов, после будет отсутствовать.
- **depends** – содержит разделенный запятыми список всех модулей, зависящих от текущего.



ПРИМЕЧАНИЕ:

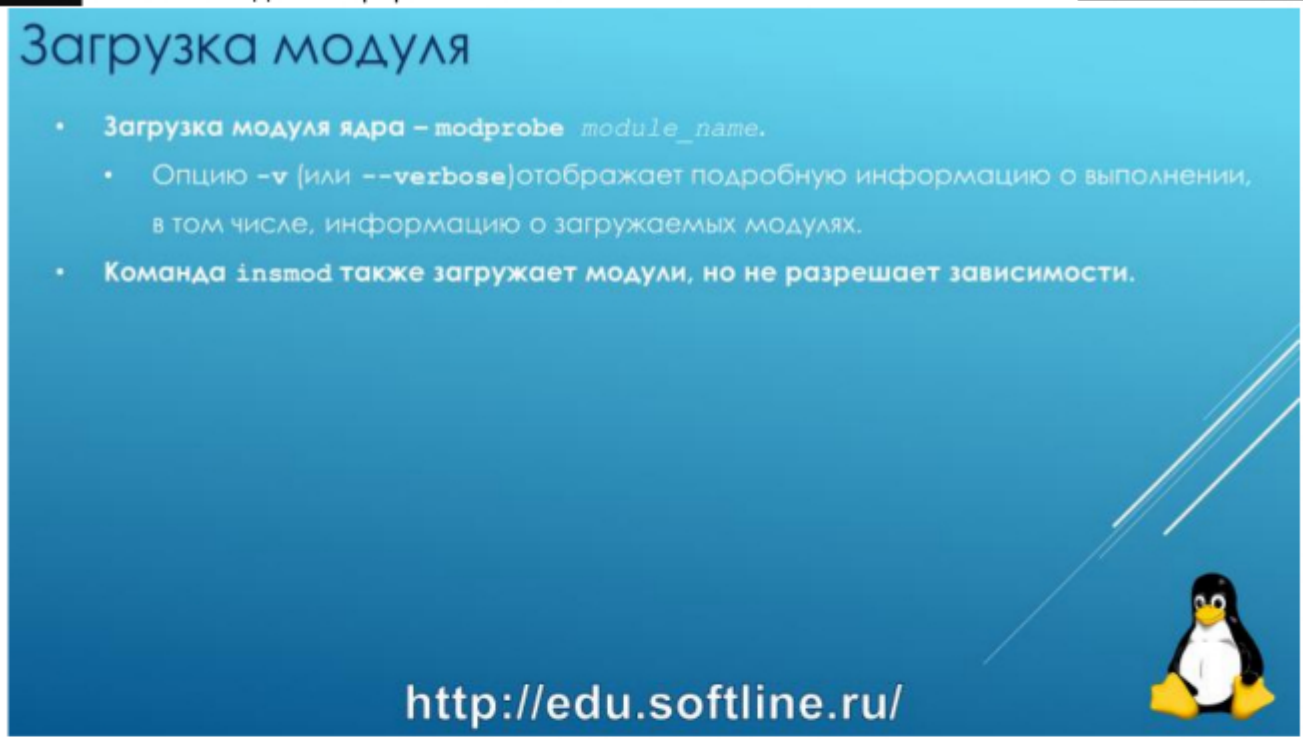
Если у модуля не зависимостей, поле depends может отсутствовать в выводе.

- **param** – каждое поле **param** представляет один параметр в следующей форме:

  имя_параметра:*описание*

где:

  - *имя_параметра* – это точный синтаксис, который можно использовать как параметр в командной строке или в строке опций в файле **.conf** в каталоге **/etc/modprobe.d/**.
  - *описание* – краткое объяснение назначения параметра вместе с типом значения, которое может принимать параметр (такое как **int**, **unit** или **array of int**) в круглых скобках.



Для загрузки модуля ядра, выполните **modprobe** *имя_модуля*, от имени пользователя root. Например, для загрузки модуля **wacom**, воспользуйтесь следующей командой:

```
# modprobe wacom
```

По умолчанию, **modprobe** пытается загрузить модуль из **/lib/modules/***версия_ядра***/kernel/drivers/**. В этом каталоге, каждый тип модуля имеет собственный подкаталог, такой как **net/** и **scsi/**, для сети и интерфейсного драйвера SCSI соответственно.

Некоторые модули имеют зависимости – другие модули ядра, которые должны быть загружены до загрузки зависимого модуля. Команда **modprobe** всегда учитывает зависимости при выполнении операций. Когда вы запрашиваете **modprobe** загрузить указанный модуль ядра, в первую очередь она проверяет зависимости модуля и загружает их, в случае их наличия и, если они не загружены. **modprobe** проверяет зависимости рекурсивно: она загружает все зависимости зависимостей и так далее, чтобы убедится, что все зависимости загружены.

Вы можете использовать опцию **-v** (или **--verbose**), чтобы заставить **modprobe** отображать подробную информацию о выполнении, в том числе, информацию о загружаемых модулях.

Пример подробного вывода подгружаемых зависимостей для модуля Fibre Channel over Ethernet (**fcoe**):

```
# modprobe -v fcoe
insmod /lib/modules/3.10.0-9.el7.x86_64/kernel/drivers/scsi/scsi_tgt.ko
insmod /lib/modules/3.10.0-9.el7.x86_64/kernel/drivers/scsi/scsi_transport_fc.ko
insmod /lib/modules/3.10.0-9.el7.x86_64/kernel/drivers/scsi/libfc/libfc.ko
insmod /lib/modules/3.10.0-9.el7.x86_64/kernel/drivers/scsi/fcoe/libfcoe.ko
insmod /lib/modules/3.10.0-9.el7.x86_64/kernel/drivers/scsi/fcoe/fcoe.ko
```

В данном примере, вы можете увидеть, что **modprobe** загрузила модули **scsi_tgt**, **scsi_transport_fc**, **libfc** и **libfcoe** как зависимости перед загрузкой модуля **fcoe**. Также, обратите внимание, что **modprobe** использует более примитивную команду **insmod** для вставки модулей в запущенное ядро.



ПРИМЕЧАНИЕ:

Для загрузки модулей в запущенное ядро можно использовать команду **`insmod`**, *но из-за того, что она не разрешает зависимости, рекомендуется использовать команду* **`modprobe`**.



Вы можете выгрузить модуль ядра, при помощи команды **modprobe -r** *имя_модуля*, запущенной от имени пользователя root. Например, для выгрузки модуля **wacom**, воспользуйтесь следующей командой.

```
# modprobe -r wacom
```

Однако эта команда может завершится неудачей, если какой-либо процесс использует модуль **wacom** или в ядро загружен зависящий от него модуль.

Например, чтобы выгрузить модуль **firewire_ohci**, необходимо проследить всю цепочку его зависимостей:

```
# modinfo  -F depends firewire_ohci
firewire-core
# modinfo  -F depends firewire-core
crc-itu-t
# modinfo -F depends crc-itu-t
```

Как видно из примера кода выше, модуль **firewire_ohci** зависит от **firewire-core**, а тот в свою очередь зависит от **crc-itu-t**.

Вы можете выгрузить модуль при помощи команды **modprobe -v -r** *имя_модуля*, где **-r** сокращение **--remove**, а **-v** сокращение **--verbose**:

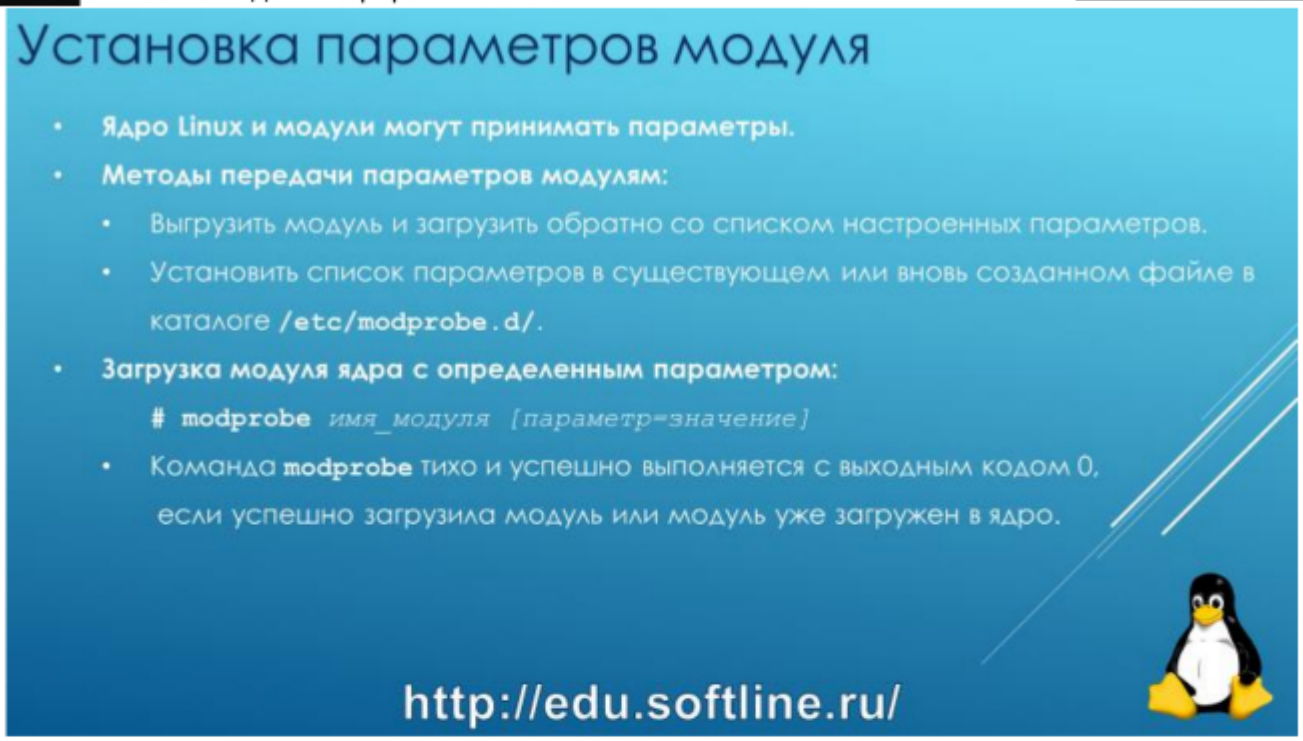
```
# modprobe -r -v firewire_ohci
rmmod firewire_ohci
rmmod firewire_core
rmmod crc_itu_t
```

Вывод показывает, что модули выгружены в обратном порядке, от того, как они были загружены, данный вывод говорит о том, что никакие процессы, зависящие от этих модулей, не были выгружены.



ПРИМЕЧАНИЕ:

Для выгрузки модулей из запущенного ядра можно использовать команду ***rmmod****, но из-за того, что она выгружает только указанный модуль, рекомендуется использовать команду* ***modprobe****.*



Как и само ядро, модули могут принимать параметры, которые меняют их поведение. В большинстве случае, параметры по умолчанию отлично подходят, но иногда необходимо или желательно установить параметр модуля, запущенного в ядро в иное значение. Существует два метода сделать это:

1. Вы можете выгрузить модуль вместе с зависимостями, при помощи команды **modprobe -r** и затем загрузить обратно при помощи **modprobe** вместе со списком настроенных параметров. Этот метод часто используется, когда у модуля не много зависимостей или тестируются различные комбинации параметров без установки их на постоянно основе.
2. В качестве альтернативы, можно установить список параметров в существующем или вновь созданном файле в каталоге **/etc/modprobe.d/**. Этот метод делает параметры постоянными, они будут использоваться при загрузке системы и вызове команды **modprobe**.

Для загрузки модуля ядра с определенным параметром, используйте команду modprobe в следующем формате:

 # modprobe *имя_модуля* [*параметр*=*значение*]

Когда вы загружаете модуль с определенными параметрами в командной строке:

- Вы можете ввести несколько параметров и значений, разделив их пробелами.
- Некоторые параметры модулей используют список, разделенный запятыми, в качестве значения. При вводе такого списка нельзя использовать пробелы, иначе **modprobe** некорректно интерпретирует значение после пробела, как дополнительный параметр.
- Команда **modprobe** тихо и успешно выполняется с выходным кодом **0**, если успешно загрузила модуль или модуль уже загружен в ядро.

Таким образом, вы должны предварительно убедиться, что модуль еще не загружен, прежде чем загружать его с настроенными параметрами. Команда **modprobe** не перезагружает модуль автоматически и не выдает предупреждения о том, что он уже загружен.

Рассмотрим рекомендованные шаги для установки настраиваемого параметра и загрузки модуля ядра. В качестве примера, возьмем модуль **e1000e**, который является драйвером сетевого адаптера Intel PRO/1000.

1. Убедитесь, что модуль еще не загружен в ядро:

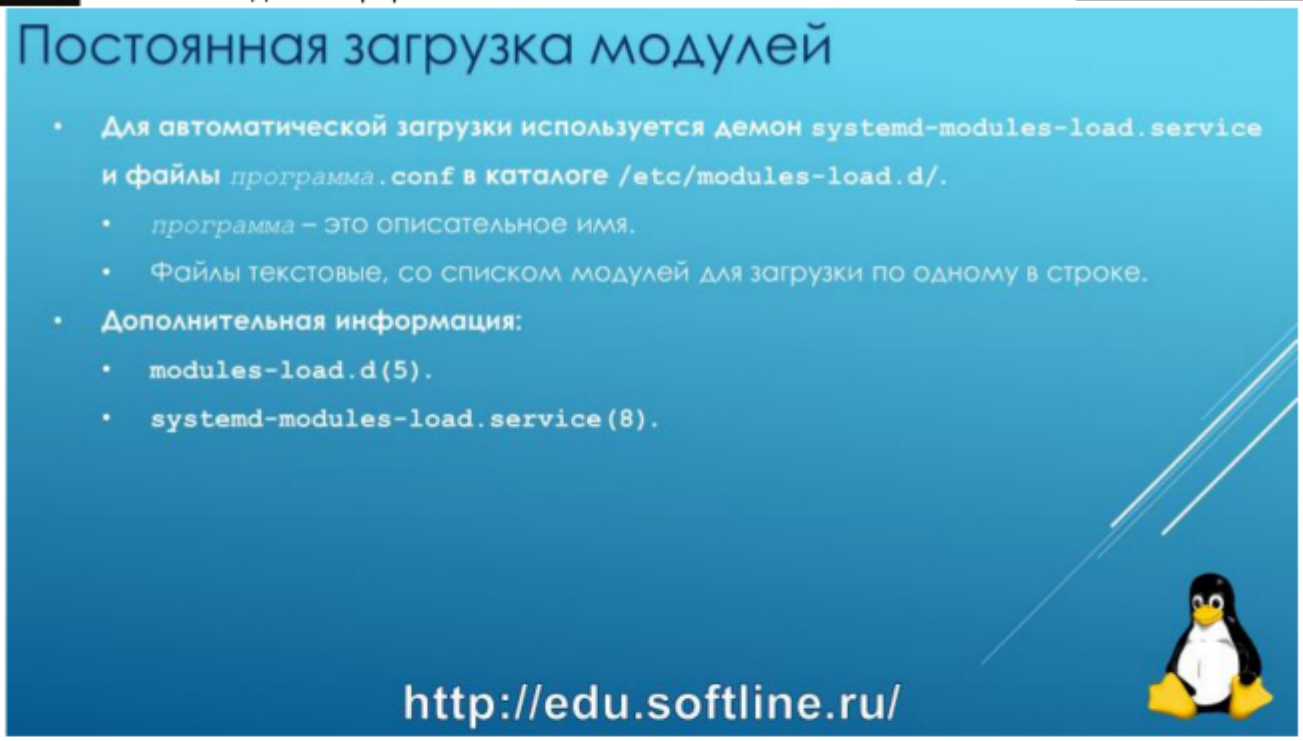
   ```
   # lsmod | grep e1000e
   ```

   Если вывод команды покажет загруженный модуль, его необходимо выгрузить.

2. Загрузите модуль, перечислив все необходимые настраиваемые параметры после имени модуля. Например, если вы хотите загрузить драйвер сетевого адаптера Intel PRO/1000 с уровнем частоты прерывания, установленным в **3000** прерываний в секунду, для первого, второго и третьего интерфейсов драйвера и включить отладку, введите следующую команду от имени пользователя root:

   ```
   # modprobe e1000e InterruptThrottleRate=3000,3000,3000 debug=1
   ```

Данный пример иллюстрирует передачу нескольких значений, разделенных запятыми, в одном параметре без использования пробелов.



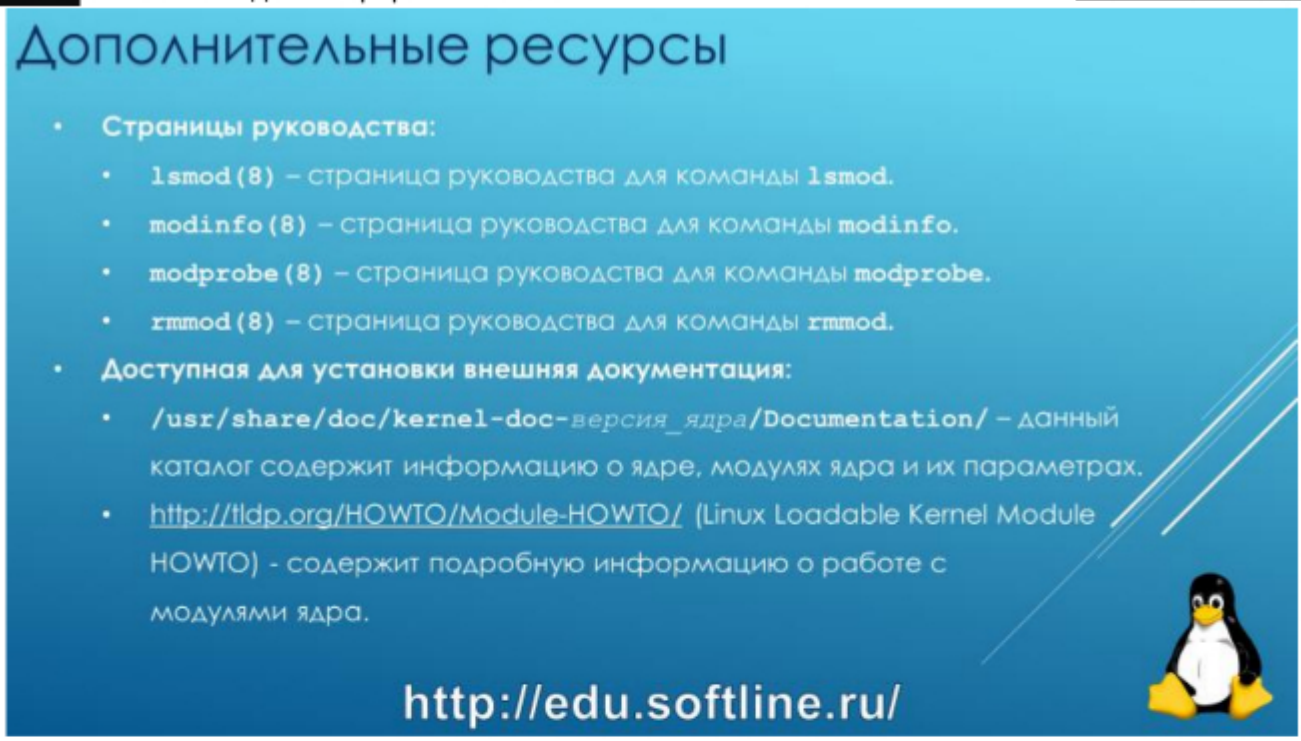
Как можно было заметить ранее, множество модулей ядра загружаются автоматически в процессе загрузки системы. Вы можете указать дополнительные модули для автоматической загрузки демоном **systemd-modules-load.service** за счет создания файла *программа*.**conf** в каталоге **/etc/modules-load.d/**, где *программа* – это любое описательное имя на ваш выбор. Файлы в каталоге **/etc/modules-load.d/** текстовые, со списком модулей для загрузки по одному в строке.

Пример текстового файла для загрузки модуля **virtio-net.ko** (Сетевой драйвер Virtio).

Чтобы создать файл для загрузки модуля **virtio-net.ko**, создайте файл **/etc/modules-load.d/virtio-net.conf** со следующем содержимым:

```
# Load virtio-net.ko at boot
virtio-net
```

Для получения дополнительной информации обратитесь к страницам руководства **modules-load.d(5)** и **systemd-modules-load.service(8)**.



Для получения дополнительной информации по модулям ядра и их утилитам, смотрите следующие ресурсы:

- Страницы руководства:
  - **lsmod(8)** – страница руководства для команды **lsmod**.
  - **modinfo(8)** – страница руководства для команды **modinfo**.
  - **modprobe(8)** – страница руководства для команды **modprobe**.
  - **rmmod(8)** – страница руководства для команды **rmmod**.
- Доступная для установки внешняя документация:
  - **/usr/share/doc/kernel-doc-***версия_ядра***/Documentation/** - данный каталог, который предоставляется пакетом **kernel-doc**, содержит информацию о ядре, модулях ядра и их параметрах. Для установки документации, воспользуйтесь следующей командой от имени пользователя root:

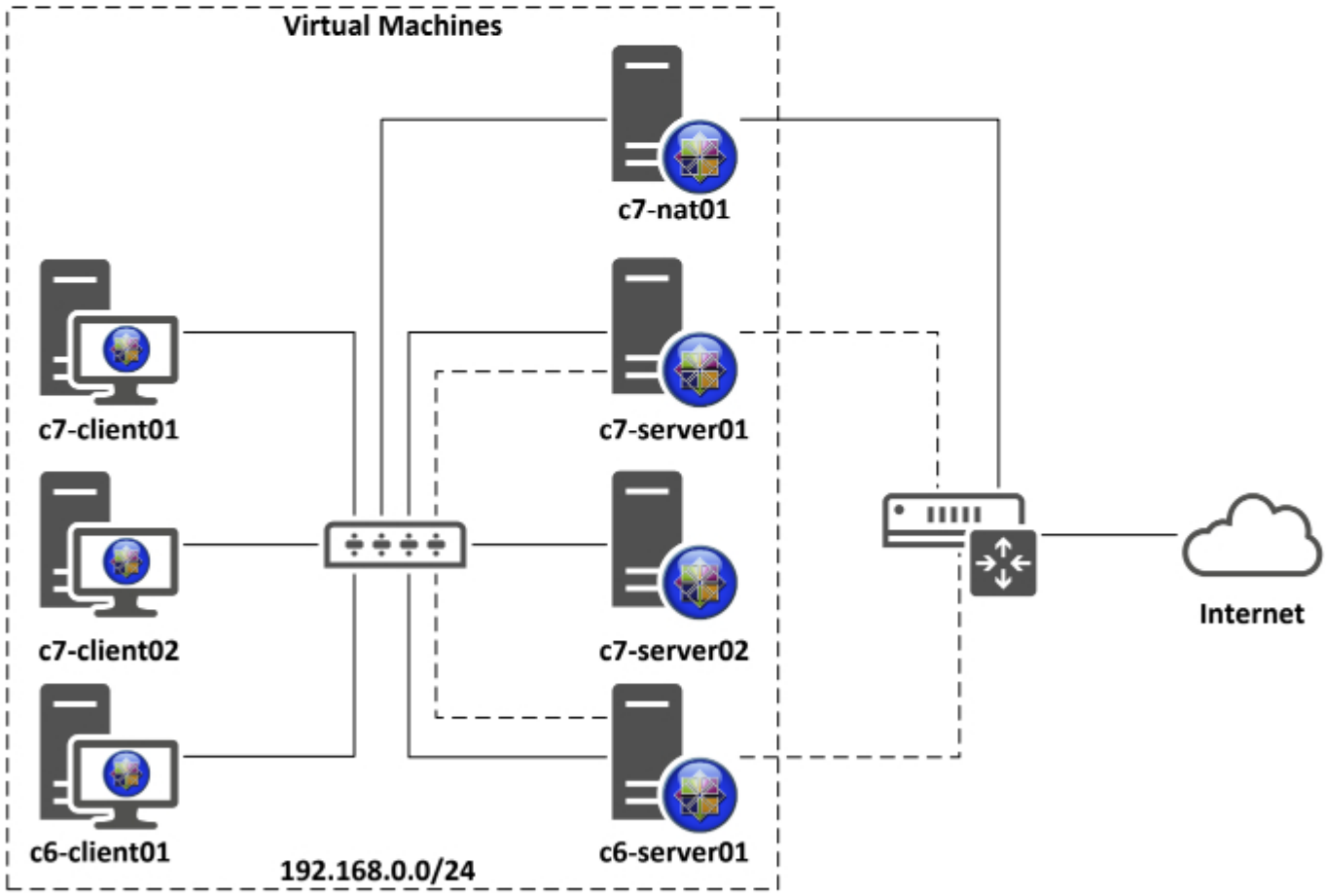
    ```
    # yum install kernel-doc
    ```

  - Linux Loadable Kernel Module HOWTO (http://tldp.org/HOWTO/Module-HOWTO/) – содержит подробную информацию о работе с модулями ядра.

Лабораторная работа 3

- **Упражнения:**
  - Управление модулями ядра.
- **Необходимые машины:**
  - `c7-nat01`.
- **Учетные данные:**
  - Пользователь: `root`.
  - Пароль: **`Pa$$w0rd`**.

http://edu.softline.ru/





Упражнение 1 – Управление модулями ядра.

1. Запустите машину **c7-server01**.
2. Получите список загруженных на системе модулей ядра.

```
# lsmod
```

3. Получите информацию о модуле ядра **e1000**.

```
# modinfo e1000
```

4. Выясните от каких модулей ядра зависит модуль **nfs**?

```
# modinfo nfs -F depends
```

5. Импортируйте модуль ядра **e1000**.

```
# modprobe -v e1000
```

6. Настройте автоматическую загрузку модуля ядра **e1000**

```
# vi /etc/modules-load.d/e1000.conf
e1000
```

7. Перезагрузите систему и убедитесь, что модуль **e1000** успешно загружен.

```
# reboot
# lsmod  | grep e1000
```

8. Выгрузите модуль ядра **e1000**.

```
# modprobe -v -r e1000
```

9. Отключите автоматическую загрузку модуля ядра **e1000**.

```
# rm -f /etc/modules-load.d/e1000.conf
```



Ответы к упражнению 1 – Управление модулями ядра.

1. Запустите машину **c7-server01**.
2. Получите список загруженных на системе модулей ядра.

```
# lsmod
```

3. Получите информацию о модуле ядра e1000.

```
# modinfo e1000
```

4. Выясните от каких модулей ядра зависит модуль **nfs**?

```
# modinfo nfs -F depends
```

5. Импортируйте модуль ядра **e1000**.

```
# modprobe -v e1000
```

6. Настройте автоматическую загрузку модуля ядра **e1000**

```
# vi /etc/modules-load.d/e1000.conf
e1000
```

7. Перезагрузите систему и убедитесь, что модуль **e1000** успешно загружен.

```
# reboot
# lsmod  | grep e1000
```

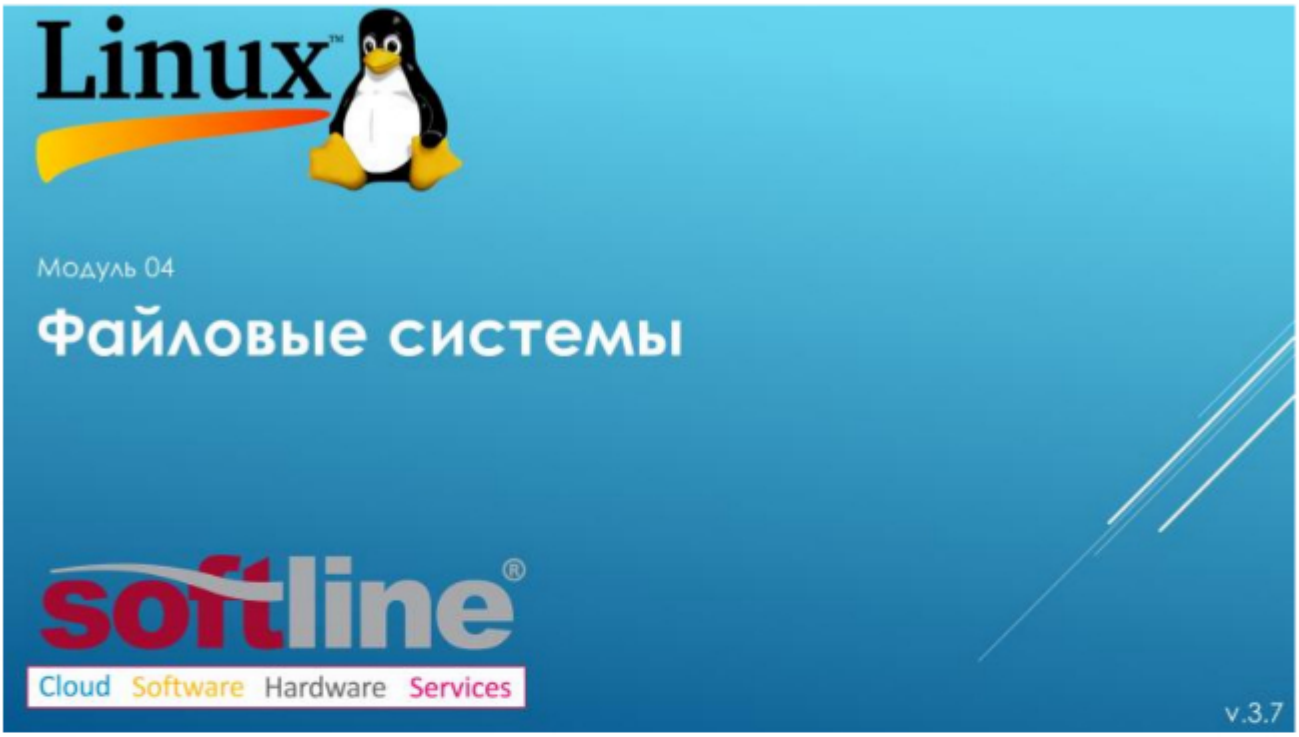
8. Выгрузите модуль ядра **e1000**.

```
# modprobe -v -r e1000
```

9. Отключите автоматическую загрузку модуля ядра **e1000**.

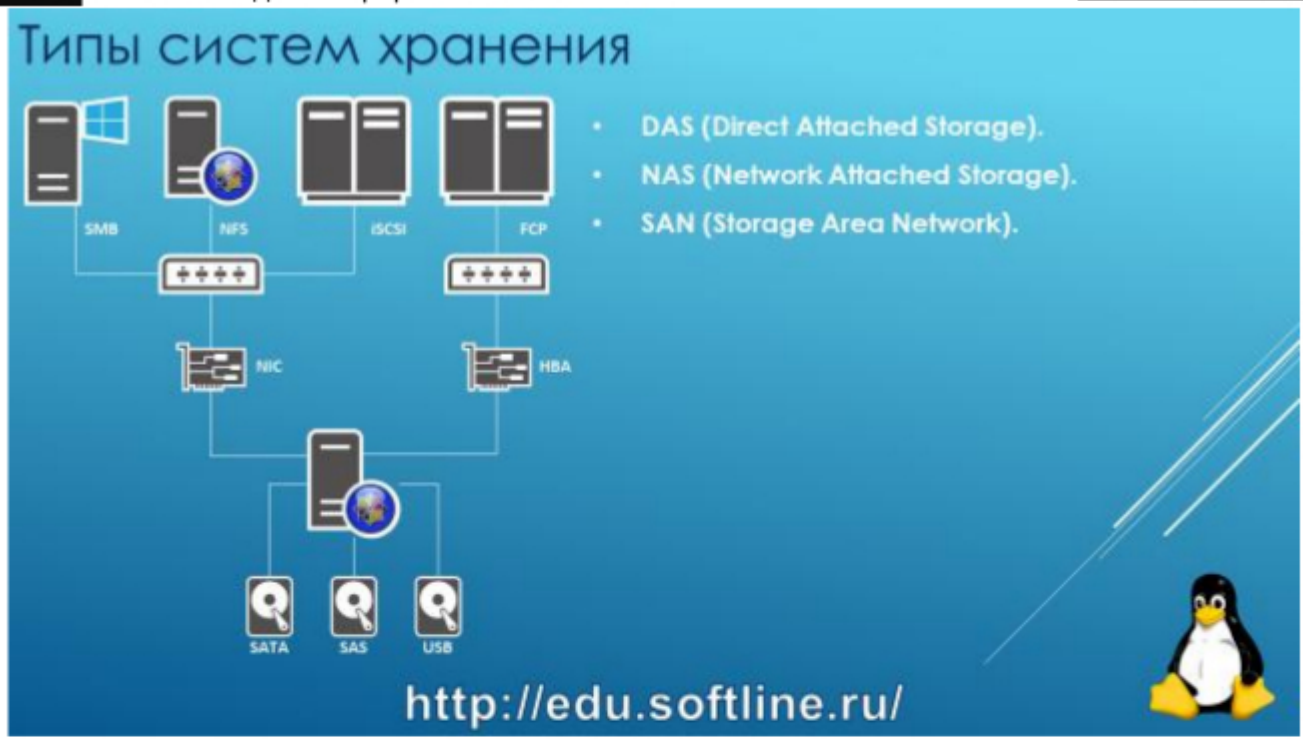
```
# rm -f /etc/modules-load.d/e1000.conf
```

Модуль 04: Файловые системы.



В модуле представлены:

- Типы систем хранения.
- Управление разделами дисков.
- Управление программным RAID.
- Управление менеджером логических томов (LVM).
- Проверка целостности файловых систем.
- Управление областью подкачки (SWAP).



DAS (Direct Attached Storage) – система хранения данных с прямым подключением.

Запоминающее устройство, непосредственно подключенное к серверу или рабочей станции без помощи сети хранения данных.

NAS (Network Attached Storage) – сетевая система хранения данных, сетевое хранилище.

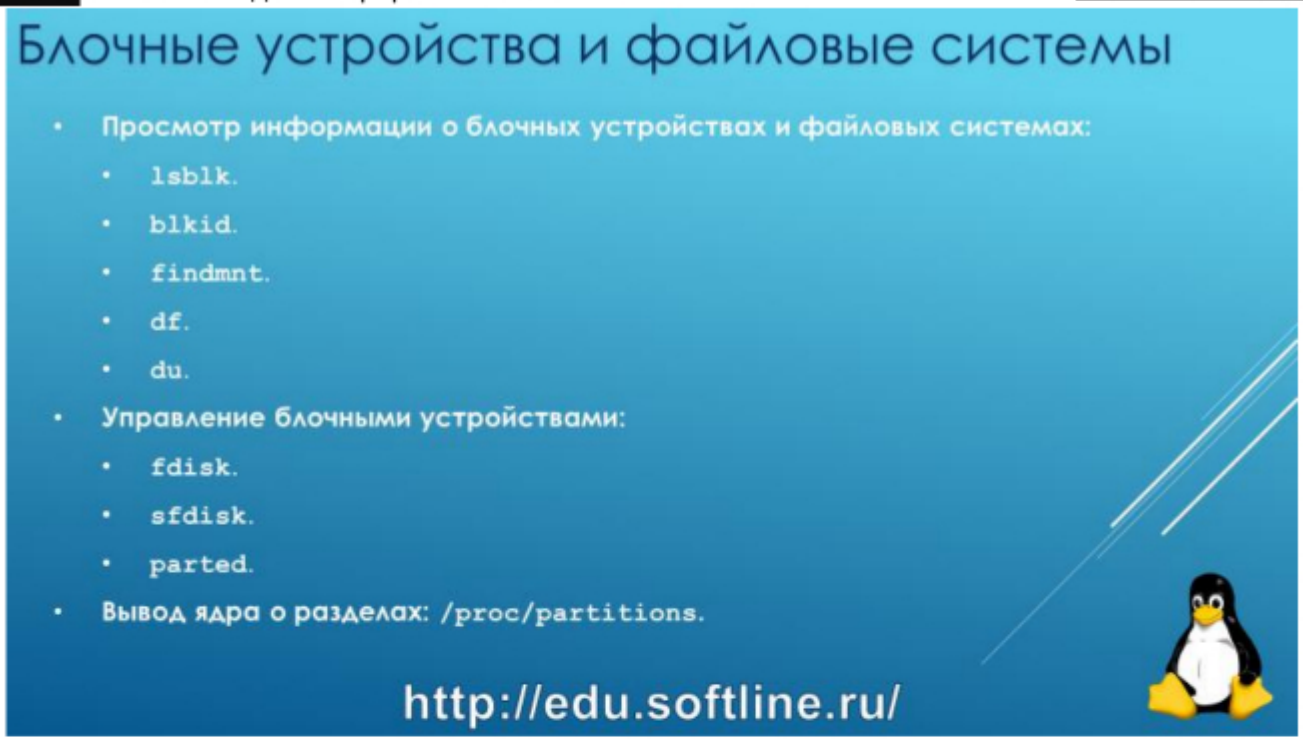
Представляет собой компьютер с некоторым дисковым массивом, подключенный к сети (обычно локальной) и поддерживающий работу по принятым в ней протоколам.

Наиболее популярные протоколы NAS: NFS и SMB/CIFS.

SAN (Storage Area Network) – сеть хранения данных (СХД).

Представляет собой архитектурное решение для подключения внешних устройств хранения данных, таких как дисковые массивы, ленточные библиотеки, оптические приводы к серверам таким образом, чтобы операционная система распознала подключённые ресурсы как локальные.

Наиболее популярные протоколы SAN: Fibre Channel (FC) и iSCSI.



Получить информацию о текущей структуре жёсткого диска можно самыми разнообразными способами. Самый простой из них, обращение к ядру через файловую систему **/proc**:

```
# cat /proc/partitions
major minor  #blocks  name
   8     0  117220824 sda
   8     1  104391 sda1
   8     2  115708162 sda2
   8     3  1012095 sda3
```

Использование команды lsblk.

Команда **lsblk** позволяет вам отобразить список доступных блочных устройств.

```
# lsblk
```

Для каждого отображаемого блочного устройства команда **lsblk** отображает имя устройства (**NAME**), мажорный [major] и минорный [minor] номера устройства (**MAJ:MIN**), пометку для съемных [removable] устройств (**RM**), размер устройства (**SIZE**), пометку для устройств только для чтения (**RO**), тип (**TYPE**) и точку монтирования (**MOUNTPOINT**).

```
# lsblk
NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda               8:0    0   20G  0 disk
├─sda1            8:1    0  500M  0 part /boot
└─sda2            8:2    0 19.5G  0 part
  ├─centos-root 253:0    0 18.5G  0 lvm  /
  └─centos-swap 253:1    0    1G  0 lvm  [SWAP]
sr0              11:0    1 1024M  0 rom
```

По умолчанию **lsblk** оторажает блочные устройства в древовидном формате. Для отображения информации в отсортированном списке, используйте опцию **-l**:

```
# lsblk -l
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda           8:0    0   20G  0 disk
```

```
sda1            8:1    0  500M  0 part /boot
sda2            8:2    0 19.5G  0 part
centos-root 253:0    0 18.5G  0 lvm  /
centos-swap 253:1    0    1G  0 lvm  [SWAP]
sr0            11:0    1 1024M  0 rom
```

Для получения информации по всем доступным опциям воспользуйтесь страницей руководства **lsblk**(8).

Использование команды blkid.

Команда **blkid** позволяет вам отобразить информацию о доступных блочных устройствах. Для этого потребуются права пользователя root.

 # blkid

Для каждого отображаемого устройства команда **blkid** отображает доступные атрибуты, такие как универсальный уникальный номер [Universal Unique Identifier] (**UUID**), тип файловой системы (**TYPE**), метка тома (**LABEL**).

```
# blkid
/dev/sda1: UUID="323df270-83ce-44f5-9676-ab1f0de709c5" TYPE="xfs"
/dev/sda2: UUID="R3sOR0-15P6-0VBL-xtTZ-P6lG-M2t8-Edy8Bw" TYPE="LVM2_member"
/dev/mapper/centos-root: UUID="ea1912a8-19e5-49c9-8b5b-b890eef29cb8" TYPE="xfs"
/dev/mapper/centos-swap: UUID="2d29ba08-38d8-4a7b-b1ec-8dbb9d9d70be" TYPE="swap"
```

По умолчанию команда **blkid** отображает список всех доступных блочных устройств. Для отображения информации о выбранном устройстве, укажите имя устройства в командной строке.

 # blkid *имя_устройстве*

Например, для отображения информации о **/dev/sda1**, используйте:

```
# blkid /dev/sda1
/dev/sda1: UUID="323df270-83ce-44f5-9676-ab1f0de709c5" TYPE="xfs"
```

Также, вы можете использовать опции **-p** и **-o udev**, для отображения более подробной информации. Для выполнения команды требуются привилегии пользователя root.

```
# blkid -po udev /dev/sda1
ID_FS_UUID=323df270-83ce-44f5-9676-ab1f0de709c5
ID_FS_UUID_ENC=323df270-83ce-44f5-9676-ab1f0de709c5
ID_FS_TYPE=xfs
ID_FS_USAGE=filesystem
ID_PART_ENTRY_SCHEME=dos
ID_PART_ENTRY_TYPE=0x83
ID_PART_ENTRY_FLAGS=0x80
ID_PART_ENTRY_NUMBER=1
ID_PART_ENTRY_OFFSET=2048
ID_PART_ENTRY_SIZE=1024000
ID_PART_ENTRY_DISK=8:0
```

Для получения информации по всем доступным опциям, воспользуйтесь страницей руководства **blkid**(8).

Использование команды findmnt.

Команда **findmnt** позволяет вам отобразить список смонтированных на данный момент файловых систем.

```
# findmnt
```

Для каждой файловой системы, команда **findmnt** отображает целевую точку монтирования (**TARGET**), исходное устройство (**SOURCE**), тип файловой системы (**FSTYPE**) и связанные опции монтирования (**OPTIONS**).

```
# findmnt
TARGET                                SOURCE     FSTYPE     OPTIONS
/                                     /dev/mapper/centos-root
                                                 xfs        rw,relatime,seclabel,attr2,
├─/proc                               proc       proc       rw,nosuid,nodev,noexec,rela
│ └─/proc/sys/fs/binfmt_misc          systemd-1  autofs     rw,relatime,fd=31,pgrp=1,ti
├─/sys                                sysfs      sysfs      rw,nosuid,nodev,noexec,rela
│ ├─/sys/kernel/security              securityfs security   rw,nosuid,nodev,noexec,rela
│ ├─/sys/fs/cgroup                    tmpfs      tmpfs      rw,nosuid,nodev,noexec,secl
│ │ ├─/sys/fs/cgroup/systemd          cgroup     cgroup     rw,nosuid,nodev,noexec,rela
│ │ ├─/sys/fs/cgroup/cpuset           cgroup     cgroup     rw,nosuid,nodev,noexec,rela
│ │ ├─/sys/fs/cgroup/cpu,cpuacct      cgroup     cgroup     rw,nosuid,nodev,noexec,rela
│ │ ├─/sys/fs/cgroup/memory           cgroup     cgroup     rw,nosuid,nodev,noexec,rela
│ │ ├─/sys/fs/cgroup/devices          cgroup     cgroup     rw,nosuid,nodev,noexec,rela
│ │ ├─/sys/fs/cgroup/freezer          cgroup     cgroup     rw,nosuid,nodev,noexec,rela
│ │ ├─/sys/fs/cgroup/net_cls          cgroup     cgroup     rw,nosuid,nodev,noexec,rela
│ │ ├─/sys/fs/cgroup/blkio            cgroup     cgroup     rw,nosuid,nodev,noexec,rela
│ │ ├─/sys/fs/cgroup/perf_event       cgroup     cgroup     rw,nosuid,nodev,noexec,rela
│ │ └─/sys/fs/cgroup/hugetlb          cgroup     cgroup     rw,nosuid,nodev,noexec,rela
│ ├─/sys/fs/pstore                    pstore     pstore     rw,nosuid,nodev,noexec,rela
│ ├─/sys/kernel/config                configfs   configfs   rw,relatime
│ ├─/sys/fs/selinux                   selinuxfs  selinuxf   rw,relatime
│ └─/sys/kernel/debug                 debugfs    debugfs    rw,relatime
├─/dev                                devtmpfs   devtmpfs   rw,nosuid,seclabel,size=237
│ ├─/dev/shm                          tmpfs      tmpfs      rw,nosuid,nodev,seclabel
│ ├─/dev/pts                          devpts     devpts     rw,nosuid,noexec,relatime,s
│ ├─/dev/hugepages                    hugetlbfs  hugetlbf   rw,relatime,seclabel
│ └─/dev/mqueue                       mqueue     mqueue     rw,relatime,seclabel
├─/run                                tmpfs      tmpfs      rw,nosuid,nodev,seclabel,mo
└─/boot                               /dev/sda1  xfs        rw,relatime,seclabel,attr2,
```

По умолчанию **findmnt** отображает файловые системы в древовидном формате. Для отображения отсортированного списка, добавьте в команду опцию **-l**.

```
# findmnt -l
TARGET                     SOURCE     FSTYPE     OPTIONS
/proc                      proc       proc       rw,nosuid,nodev,noexec,relatime
/sys                       sysfs      sysfs      rw,nosuid,nodev,noexec,relatime,s
/dev                       devtmpfs   devtmpfs   rw,nosuid,seclabel,size=237720k,n
/sys/kernel/security       securityfs security   rw,nosuid,nodev,noexec,relatime
/dev/shm                   tmpfs      tmpfs      rw,nosuid,nodev,seclabel
/dev/pts                   devpts     devpts     rw,nosuid,noexec,relatime,seclabe
/run                       tmpfs      tmpfs      rw,nosuid,nodev,seclabel,mode=755
/sys/fs/cgroup             tmpfs      tmpfs      rw,nosuid,nodev,noexec,seclabel,m
/sys/fs/cgroup/systemd     cgroup     cgroup     rw,nosuid,nodev,noexec,relatime,x
/sys/fs/pstore             pstore     pstore     rw,nosuid,nodev,noexec,relatime
/sys/fs/cgroup/cpuset      cgroup     cgroup     rw,nosuid,nodev,noexec,relatime,c
/sys/fs/cgroup/cpu,cpuacct cgroup     cgroup     rw,nosuid,nodev,noexec,relatime,c
/sys/fs/cgroup/memory      cgroup     cgroup     rw,nosuid,nodev,noexec,relatime,m
/sys/fs/cgroup/devices     cgroup     cgroup     rw,nosuid,nodev,noexec,relatime,d
/sys/fs/cgroup/freezer     cgroup     cgroup     rw,nosuid,nodev,noexec,relatime,f
/sys/fs/cgroup/net_cls     cgroup     cgroup     rw,nosuid,nodev,noexec,relatime,n
/sys/fs/cgroup/blkio       cgroup     cgroup     rw,nosuid,nodev,noexec,relatime,b
/sys/fs/cgroup/perf_event  cgroup     cgroup     rw,nosuid,nodev,noexec,relatime,p
/sys/fs/cgroup/hugetlb     cgroup     cgroup     rw,nosuid,nodev,noexec,relatime,h
/sys/kernel/config         configfs   configfs   rw,relatime
/                          /dev/mapper/centos-root
                                      xfs        rw,relatime,seclabel,attr2,inode6
```

```
/sys/fs/selinux             selinuxfs  selinuxf rw,relatime
/proc/sys/fs/binfmt_misc    systemd-1  autofs   rw,relatime,fd=31,pgrp=1,timeout=
/dev/hugepages              hugetlbfs  hugetlbf rw,relatime,seclabel
/dev/mqueue                 mqueue     mqueue   rw,relatime,seclabel
/sys/kernel/debug           debugfs    debugfs  rw,relatime
/boot                       /dev/sda1  xfs      rw,relatime,seclabel,attr2,inode6
```

Также, вы можете отобразить список файловых систем, только выбранного типа. Для этого используйте опцию **-t** с указанием необходимой файловой системы.

```
# findmnt -t тип
```

Например, для отображения списка всех смонтированных файловых систем **xfs**, используйте:

```
# findmnt -t xfs
TARGET  SOURCE                  FSTYPE OPTIONS
/       /dev/mapper/centos-root xfs    rw,relatime,seclabel,attr2,inode64,noquot
└─/boot /dev/sda1               xfs    rw,relatime,seclabel,attr2,inode64,noquot
```

Для получения информации о всех доступных опциях, используйте страницу руководства **findmnt**(8).

Использование команды df.

Команда **df** позволяет вам отобразить подробный отчет об использовании пространства системных дисков.

```
# df
```

Для каждой отображаемой файловой системы, команда **df** выводит ее имя (**Filesystem**), размер (**1K-blocks** или **Size**), объем использованного места (**Used**),объем доступного места (**Available**), процент использованного места (**Use%**) и точку монтирования (**Mounted on**).

```
# df
Filesystem              1K-blocks    Used Available Use% Mounted on
/dev/mapper/centos-root  19351552 1066996  18284556   6% /
devtmpfs                   237720       0    237720   0% /dev
tmpfs                      247160       0    247160   0% /dev/shm
tmpfs                      247160   16708    230452   7% /run
tmpfs                      247160       0    247160   0% /sys/fs/cgroup
/dev/sda1                  508588  105604    402984  21% /boot
```

По умолчанию команда **df** отображает размеры разделов ы блоках по 1 килобайту (**K**) и объем использованного места в килобайтах. Для просмотра информации в мегабайтах (**M**) и гигабайтах (**G**) используйте опцию **-h**.

```
# df -h
Filesystem              Size  Used Avail Use% Mounted on
/dev/mapper/centos-root  19G  1.1G   18G   6% /
devtmpfs                233M     0  233M   0% /dev
tmpfs                   242M     0  242M   0% /dev/shm
tmpfs                   242M   17M  226M   7% /run
tmpfs                   242M     0  242M   0% /sys/fs/cgroup
/dev/sda1               497M  104M  394M  21% /boot
```

Для получения информации о всех доступных опциях, используйте страницу руководства **df**(1).

Использование команды du.

Команда **du** позволяет вам отобразить объем использованного файлами места в каталоге. Для отображения места использованного каждым подкаталогом в текущем каталоге, используйте команду без дополнительных параметров.

```
# du
4       ./.cache/abrt
4       ./.cache
0       ./.config/abrt
0       ./.config
4       ./.ssh
44      .
```

По умолчанию команда du отображает использованное место на диске в килобайтах (**K**). Для просмотра информации в мегабайтах (**M**) и гигабайтах (**G**) используйте опцию **-h**.

```
# du -h
4.0K    ./.cache/abrt
4.0K    ./.cache
0       ./.config/abrt
0       ./.config
4.0K    ./.ssh
44K     .
```

В конце вывода, команда всегда выводит суммарный объем текущего каталога. Для отображения только суммарного объема текущего каталога, используйте опцию **-s**.

```
# du -sh
44K     .
```

Для получения информации о всех доступных опциях, используйте страницу руководства **du**(1).



Зачем нужны разделы? Деление диска на логические части давно уже стало стандартом любой операционной системы. Дело в том, что разделяя диск, вы реализуете ряд преимуществ:

- Изоляция данных друг от друга. Вы можете отделить пользовательские данные от системных, тем самым обезопасив их, в случае возникновения уязвимости в системе, позволяющей повредить данные на диске. Более того, вы можете легко переносить данные от одной системы к другой, в случае апгрейда.
- Быстродействие. Грамотное создание логических разделов позволяет экономить время на поиске информации и позиционировании считывающих головок.
- Квотирование. Если вы желаете включить квотирование для отдельного каталога, используйте дополнительный раздел.

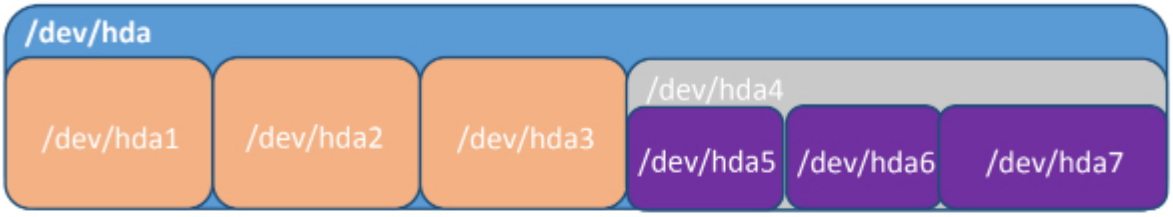
Диск может быть поделён на 4 основных раздела. В случае, когда необходимо большее количество, создаётся специальный «Extended» раздел, в котором размещаются логические разделы. Поэтому, если один из основных разделов помечен как «Extended», это означает, что есть ещё один блок, описывающий разделы. Суммарное количество разделов зависит от типа устройства:

- IDE – 63 раздела;
- SCSI – 15 разделов.

Использование основных разделов:



Использование расширенных разделов:



Для управления разделами во всех дистриубутивах существует несколько программ.

Программа **fdisk** позволяет создавать и удалять разделы. Её основной минус в большом количестве ошибок, поэтому выполняйте этой программой только те операции, в результате которых вы уверены. Например, разметить новое устройство быстрее и проще всего с помощью **fdisk**.

Программа **sfdisk** является более гибким и надёжным аналогом **fdisk**. Но её использование более сложно. Одним из достоинств – возможность использования из командной строки.

Утилита **parted** предлагает широчайший функционал, позволяющий работать с файловыми системами различных типов, а также изменять размеры существующих разделов

Основная функция команды **partprobe** – заставить ядро перечитать информацию о разделах на диске, тем самым исключая необходимость перезагрузки компьютера.



Разделы в CentOS 7

Утилита **parted** позволяет:

- Просматривать существующую таблицу разделов.
- Изменять размер существующих разделов.
- Добавлять разделы из свободного пространства или дополнительных жестких дисков.

По умолчанию пакет **parted** устанавливается вместе с CentOS (RHEL). Для запуска parted, войдите в систему от имени пользователя root и введите команду **parted /dev/sda** в командной оболочке (где **/dev/sda** – это имя накопителя, которое вы хотите настраивать).

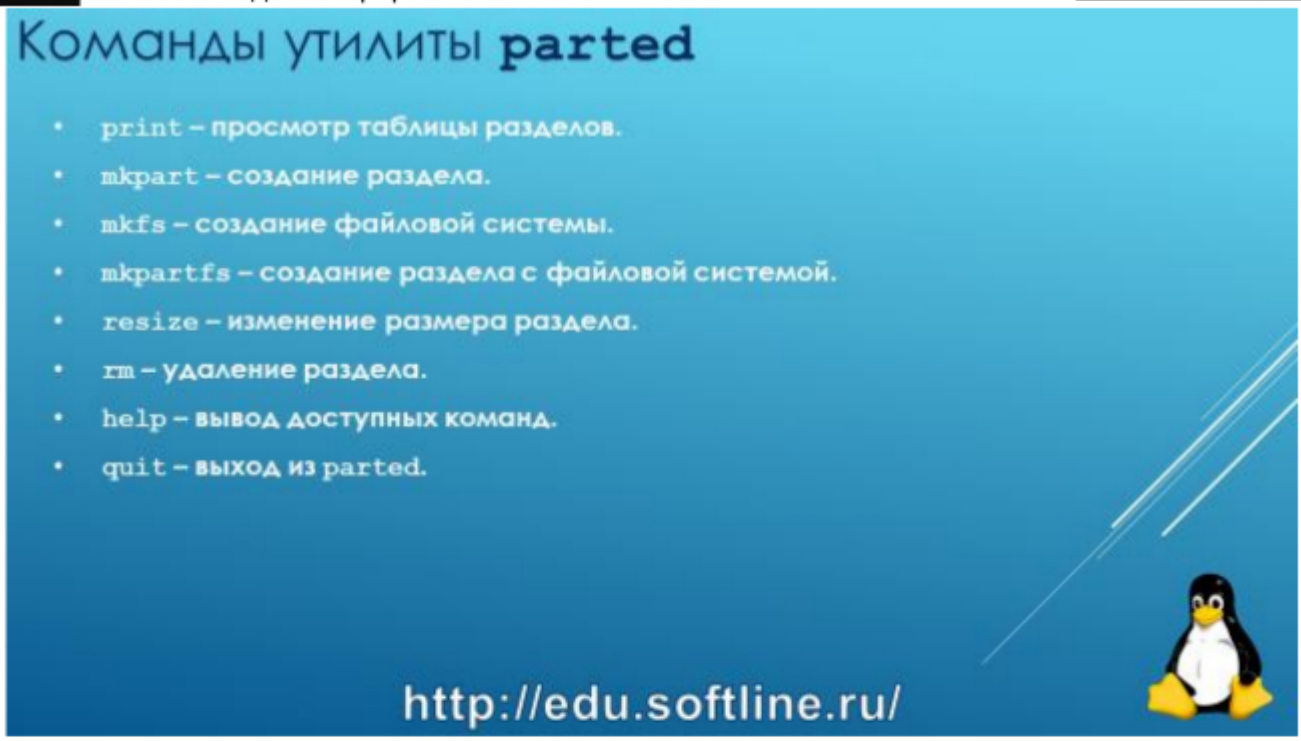
Если вы хотите удалить или изменить в размерах раздел, устройство не должно использоваться. Создание новых разделов на устройстве поддерживается, но не рекомендуется.

Чтобы устройство не использовалось, никакие разделы не должны быть смонтированы и никакая область подкачки (SWAP) не должна быть включена.

Также, таблица разделов не должна меняться, пока она используется, так как ядро может неправильно распознать изменения. Если таблица разделов не соответствует актуальному смонтированному состоянию, это может привести к потере или перезаписи данных.

Самый простой путь добиться этого – это загрузить систему в аварийном режиме (Rescue Mode). Когда будет предложено смонтировать файловые системы, выбрать **Skip** (пропустить).

В качестве альтернативы, если на диске не содержится разделов, которые необходимы для работы системы (на которых стоит системная блокировка), можно отмонтировать раздел при помощи команды **unmount** и выключить подкачку (SWAP) при помощи команды **swapoff**.



Следующая таблица содержит список наиболее часто используемых команд утилиты **parted**.

Команда	Описание
check *minor-num*	Выполнение простой проверки файловой системы.
cp *from to*	Копирование файловой системы с одного раздела на другой. *from* и *to* – это младшие номера (Minor Numbers) разделов.
help	Вывод списка доступных команд.
mklabel *label*	Создание метки диска для таблицы разделов.
mkfs *minor-num file-system-type*	Создание файловой системы *file-system-type*.
mkpart *part-type fs-type start-mb end-mb*	Создание раздела, без создания файловой системы.
mkpartfs *part-type fs-type start-mb end-mb*	Создание раздела с указанной файловой системой.
move *minor-num start-mb end-mb*	Перемещение раздела.
name *minor-num name*	Именование раздела, только для дисковых меток Mac и PC98.
print	Вывод таблицы разделов.
quit	Выход из **parted**.
rescue *start-mb end-mb*	Аварийное восстановление потерянного раздела.
resize *minor-num start-mb end-mb*	Изменение размера раздела.
rm *minor-num*	Удаление раздела.
select *device*	Выбор иного устройства для настройки.
set *minor-num flag state*	Установка флага на разделе, *state* может быть либо **on**, либо **off**.
toggle *[NUMBER [FLAG]*	Переключение флага (*FLAG*) на разделе (*NUMBER*).
unit *UNIT*	Установка юнита по умолчанию в *UNIT*.

Просмотр таблицы разделов.

После запуска **parted**, используйте команду **print** для вывода таблицы разделов.

```
# parted
GNU Parted 3.1
Using /dev/sda
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) print
Model: VMware, VMware Virtual S (scsi)
Disk /dev/sda: 21.5GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start   End     Size    Type     File system  Flags
 1      1049kB  525MB   524MB   primary  xfs          boot
 2      525MB   21.5GB  20.9GB  primary               lvm
```

В начале вывода команды **print** содержится, тип диска, производитель, номер модели и интерфейс. Оставшийся вывод, после первых 5 строк, содержит таблицу разделов.

В таблице разделов младший номер (Minor **Number**) – это номер раздела, например раздел с младшим номером 1 соответствует разделу **/dev/sda1**. Поддерживаемые типы (**Type**) это: **metadata**, **free**, **primary**, **extended** или **logical**. Тип файловой системы (**Filesystem**) может быть любым из следующего списка:

- **ext2**
- **ext3**
- **fat16**
- **fat32**
- **hfs**
- **jfs**
- **linux-swap**
- **ntfs**
- **reiserfs**
- **hp-ufs**
- **sun-ufs**
- **xfs**

Если тип файловой системы (**Filesystem**) не отображает значения, значит он не известен (**unknown**).

Колонка **Flags** содержит список флагов, установленных на разделах. Доступные флаги: **boot**, **root**, **swap**, **hidden**, **raid**, **lvm** или **lba**.



ПРИМЕЧАНИЕ:

*Для выбора другого устройства без перезапуска **parted**, используйте команду **select** с указанием имени устройства (например, **/dev/sdb**). Это позволит просматривать и настраивать таблицу разделов на другом накопителе.*

Создание раздела.



ВНИМАНИЕ:

Не пытайтесь создать таблицу разделов на используемом устройстве.

Процедура создания раздела:

1. Перед созданием раздела загрузитесь в аварийный режим (Rescue Mode) или отмонтируйте все разделы и выключите область подкачки (Swap) на устройстве.
2. Запустите утилиту **parted**, где **/dev/sda** – это устройство на котором необходимо создать разделы.

```
# parted /dev/sda
```

3. Просмотрите текущую таблицу разделов, для определения наличия достаточного количества свободного места.

```
(parted) print
```

Если места недостаточно, можно изменить размер существующих разделов.

Из таблицы разделов, необходимо определить начальную и конечную точки нового раздела, а также выбрать тип раздела. Может быть только 4 основных (Primary) раздела или три и дополнительный (Extended) с множеством логических (Logical) на нем.

Например, для создания основного раздела с файловой системой ext3 с 1024 Mb до 2048 Mb, на жестком диске, введите следующую команду:

```
(parted) mkpart primary ext3 1024 2048
```



ПРИМЕЧАНИЕ:

*Если вы используете **mkpartfs**, файловая система создается после создания раздела. Но **parted** не поддерживает создание файловой системы **ext3**. Таким образом, если вы хотите создать файловую систему ext3, используйте **mkpart** и создайте файловую систему при помощи команды **mkfs**.*

Изменения вступают в силу сразу после нажатия клавиши **<Enter>**, поэтому проверяйте команды перед их выполнением.

После создания раздела, используйте команду **print**, для проверки создания раздела, с корректным типом файловой системы и размером. Также запомните младший номер раздела (Minor Number), с его помощью вы сможете пометить любую файловую систему на нем. Вы также можете посмотреть вывод команды **cat /proc/partitions**, после выхода из **parted**, чтобы убедиться, что ядро распознало новый раздел.

Максимальное число разделов, которое parted может создать – 128. Хотя спецификация GUID Partition Table (GPT), позволяет создавать больше разделов, за счет увеличения области, зарезервированной для таблицы разделов. Рекомендуется не превышать 128 разделов.

Удаление раздела.



ВНИМАНИЕ:

Не пытайтесь удалить раздел на устройстве, которое используется.

Процедура удаления раздела.

1. Перед удалением раздела загрузитесь в аварийный режим (Rescue Mode) (или отмонтируйте все разделы и выключите область подкачки (Swap) на устройстве).
2. Запустите **parted**, где **/dev/sda** – имя устройства, на котором необходимо удалить раздел:

```
# parted /dev/sda
```

3. Просмотрите текущую таблицу разделов, чтобы определить младший номер (Minor Number) раздела, который необходимо удалить.

```
(parted) print
```

4. Удалите раздел при помощи команды **rm**. Например, для удаления раздела с младшим номером (Minor Number) **3**, воспользуйтесь следующей комндой:

```
(parted) rm 3
```

Изменения вступают в силу сразу после нажатия клавиши **<Enter>**, поэтому проверяйте команды перед их выполнением.

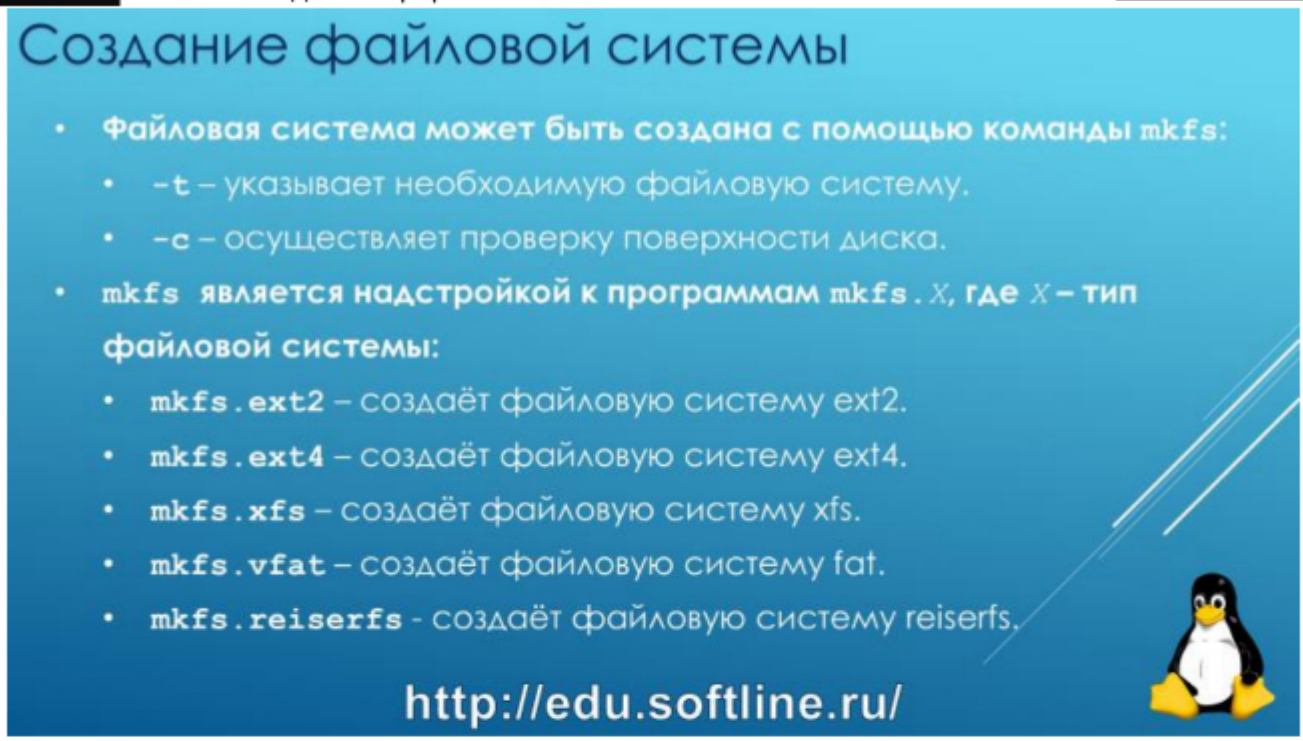
5. После создания раздела, используйте команду **print**, для проверки удаления раздела. Вы также можете посмотреть вывод команды **cat /proc/partitions**, после выхода из **parted**, чтобы убедиться, что ядро распознало удаление раздела.

```
# cat /proc/partitions
```

6. Последним шагом удалите запись из файла **/etc/fstab**.

Изменение размера раздела.

CentOS (RHEL) поддерживает изменение раздела только на LVM. Поэтому во всех остальных случаях сделайте резервную копию информации, хранящейся на файловой системе, и выполните процедуры удаления и создания нового раздела.



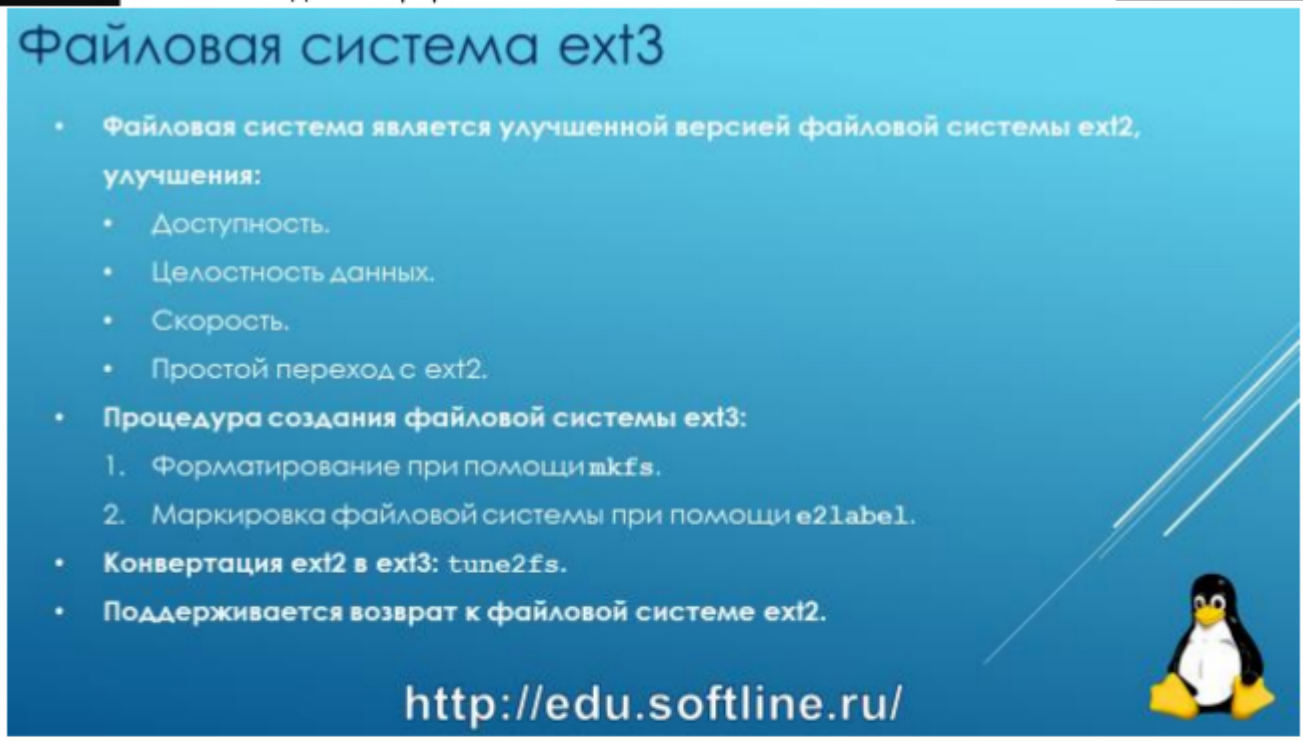
Для создания файловой системы предназначена утилита **mkfs**, которая является лишь надстройкой для вызова команд, соответствующих создаваемой файловой системы: **mkfs.ext2**, **mkfs.ext4**, **mkfs.reiserfs** и т.д. Тип создаваемой файловой системы определяется параметром **-t**.

Для создания файловых систем ext2/ext3/ext4 существует отдельная утилита **mke2fs**:

- **-b** – указание размера блока. Если вы планируете хранить множество маленьких файлов, то лучше использовать блоки меньшего размера, иначе – большего.
- **-c** – проверить поверхность диска на наличие повреждённых блоков.
- **-i** – соотношение byte/inode, определяющее сколько байт будет занимать один индексный дескриптор. Чем больше это значение, тем меньше индексных дескрипторов будет создано.
- **-N** – отменяет вычисление максимального количества индексных дескрипторов, основанное на предыдущем параметре, и задаёт определённое число, определяющее максимальное число индексных дескрипторов.
- **-m** – определяет процент блоков, зарезервированных для суперпользователя. По умолчанию – 5%.
- **-L** – установить метку тома.

Метка тома используется для удобства администратора, позволяя обращаться к устройству не по пути к файлу-устройства, а по имени, заданному для раздела диска. Метку тома можно установить командой **e2label**:

```
# e2label /dev/sda1 root
```



Файловая система является улучшенной версией файловой системы ext2. Эти улучшения заключаются в следующем:

- Доступность.
- Целостность данных.
- Скорость.
- Простой переход с ext2.

Доступность.

После сбоя электропитания или некорректного выключения системы («грязное выключение»), каждая смонтированная файловая система. На машине должна быть проверена на целостность утилитой **`e2fsck`**. Это достаточно продолжительный по времени процесс, может существенно задержать запуск операционной системы, особенно при наличии больших томов с большим числом файлов. В процессе проверки данные на томах недоступны.

Имеется возможность запустить **`fsck`** `-n` на живой файловой системе. Однако, процедура не внесет никаких изменений и в случае нахождения частично записанных метаданных может выдать ошибочные результаты.

Также, в случае использования LVM, можно создать моментальный снимок (LVM Snapshot) файловой системы и запустить **`fsck`** на нем.

Наконец, можно перемонтировать файловую систему в режим только для чтения (Read Only). Все ожидающие обновления метаданных и записи будут применены к диску, прежде чем он перемонтируется. Это позволит убедится в целостности состояния файловой системы, если она не была повреждена ранее. Теперь можно запускать **`fsck -n`**.

Журналирование, которое появилось в файловой системе ext3, позволяет отказаться от подобных проверок файловой системы после некорректного выключения системы. Единственный случай, когда потребуется проверка на целостность – это серьезные аппаратные сбои, такие как выход из строя

жесткого диска. Время восстановления файловой системы ext3 после некорректного выключения не зависит от размеров файловой системы и числа файлов на ней, оно зависит от размера журнала (journal) используемого для поддержки целостности. При использовании размера журнала по умолчанию – восстановление занимает несколько секунд, также время восстановления зависит от скорости оборудования.



ПРИМЕЧАНИЕ:
Единственный режим журналирования, поддерживаемый Red Hat (CentOS): **`data=ordered`** *(по умолчанию).*

Целостность данных.

Файловая система ext3 предотвращает потерю целостности данных в случае некорректного выключения системы. Файловая система ext3 позволяет выбирать тип и уровень защиты ваших данных. Тома с файловой системой ext3 по умолчанию сконфигурированы для поддержания высокого уровня целостности данных.

Скорость.

Несмотря на запись некоторых данных больше одного раза, в большинстве случаев ext3 имеет большую пропускную способность, чем ext2, так как журналирование оптимизирует движение головки жесткого диска. В ext3 заложена возможность выбора режима журналирования с целью оптимизации скорости, но это снижает защиту целостности данных.



ПРИМЕЧАНИЕ:
Единственный режим журналирования, поддерживаемый Red Hat (CentOS): **`data=ordered`** *(по умолчанию).*

Простой переход.

Конвертация файловой системы ext2 в ext3 проходит очень легко, и вы получаете все преимущества журналируемой файловой системы, при этом не требуется форматирование томов.



ПРИМЕЧАНИЕ:
RHEL 7 (CentOS 7) использует универсальный драйвер extN. Это отключает параметры ext2 и ext3 и вместо них использует **`ext4.co`** для этих форматов диска. В результате ядро использует сообщения ext4 вне зависимости от используемой файлвой системы.

Создание файловой системы Ext3.

После установки, иногда необходимо создать новую файловую систему ext3. Например, после добавления нового жесткого диска, может потребовать разбить его с использование файловой системы ext3.

Процедура создания файловой системы ext3:

1. Форматирование раздела или тома LVM с файловой системой ext3 при помощи `mkfs`.
2. Маркировка файловой системы при помощи **`e2label`**.

Присутствует возможность добавить специфичный UUID к файловой системе. Например, чтобы добавить UUID **`7cd65de3-e0be-41d9-b66d-96d749c02da7`** к файловой системе на разделе **`/dev/sda8`**, используйте следующие команды:

```
# mkfs -t ext3 -U 7cd65de3-e0be-41d9-b66d-96d749c02da7 /dev/sda8
# tune2fs -U 7cd65de3-e0be-41d9-b66d-96d749c02da7 /dev/sda8
```

При использовании другой файловой системы (например: ext4) процедура будет аналогичной.

Конвертация файловой системы Ext2 в Ext3.

Команда **`tune2fs`** позволяет конвертировать файловую систему ext2 в ext3.



ПРИМЕЧАНИЕ:

При конвертировании ext2 в ext3, всегда используйте утилиту **`e2fsck`** для проверки вашей файловой системы до и после использования tune2fs. Прежде чем приступать к конвертации файловой системы ext2 в ext3, выполните полное резервное копирование файловой системы ext2 на случай возникновения ошибок.
В дополнение рекомендуется создать новую файловую систему ext3, если это возможно, и переместить данные на нее, вместо конвертации из ext2 в ext3.

Для конвертации файловой системы ext2 в ext3 потребуется войти в систему, от имени пользователя root и выполнить следующую команду:

```
# tune2fs -j block_device
```

`block_device` содержит файловую систему ext2 и будет конвертирован.

Выполните команду **`df`** для отображения смонтированных файловых систем.

Возврат к файловой системе ext2.

Для возвращения к файловой системе ext2 воспользуйтесь следующей процедурой.

В данном примере в качестве блочного устройства используется: **`/dev/mapper/VolGroup00-LogVol02`**

1. Войдите в систему под пользователем root и отмонтируйте раздел:

```
# umount /dev/mapper/VolGroup00-LogVol02
```

2. Измените тип файловой системы на ext2:

```
# tune2fs -O ^has_journal /dev/mapper/VolGroup00-LogVol02
```

3. Проверьте раздел на ошибки:

```
# e2fsck -y /dev/mapper/VolGroup00-LogVol02
```

4. Смонтируйте раздел с использованием файловой системы ext2:

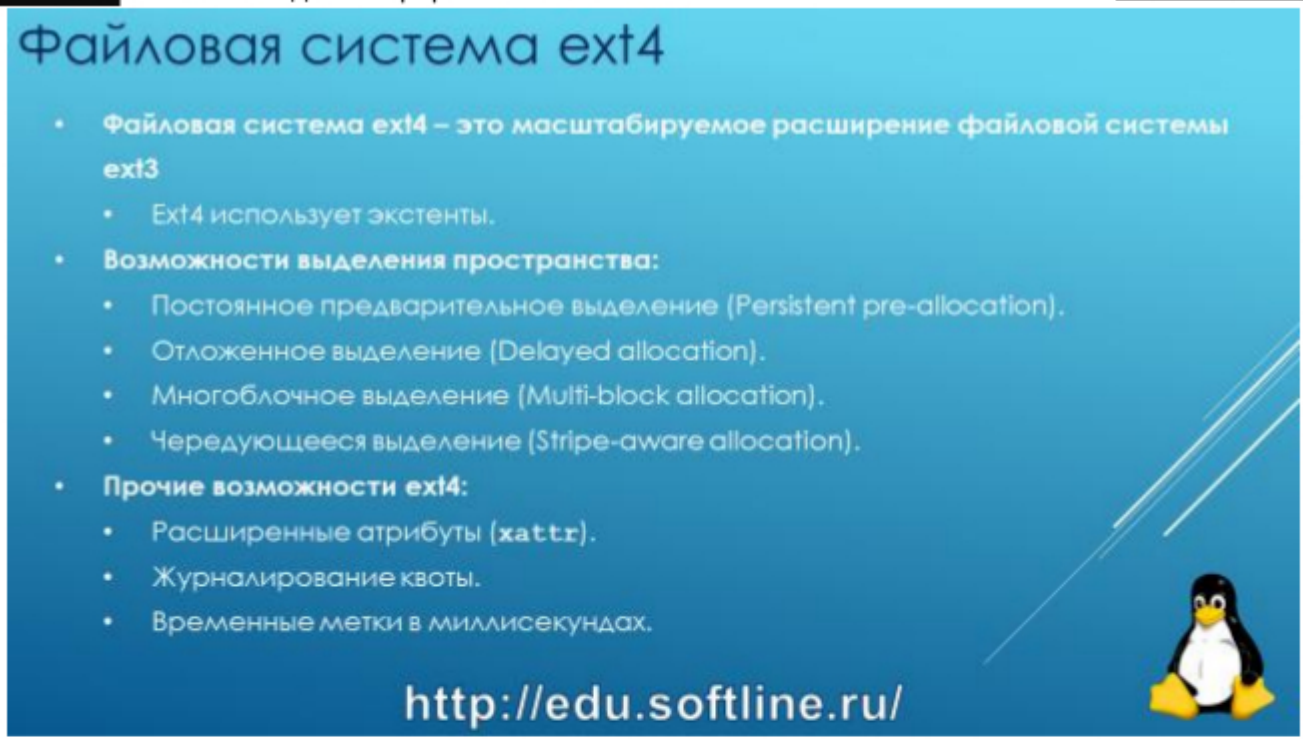
```
# mount -t ext2 /dev/mapper/VolGroup00-LogVol02 /mount/point
```

В последней команде замените **/mount/point** на точку монтирования раздела.



ПРИМЕЧАНИЕ:
Если файл **.journal** существует в корне раздела – удалите его.

Не забудьте изменить файловую систему раздела на ext2 в файле **/etc/fstab**, иначе все вернется назад после перезагрузки.



Файловая система ext4 – это масштабируемое расширение файловой системы ext3. В CentOS 7 (RHEL 7) она поддерживает максимальный размер отдельного файла в 16 терабайт (ТБ) и том с файловой системой объемом в 50 ТБ, в отличии от CentOS 6 (RHEL 6), лимит файловой системы которого составлял 16 ТБ. Также файловая система не имеет ограничений на количество подкаталогов (У ext3 – до 32,000), хотя, как только количество ссылок превышает 65000, она сбрасывается на 1 и более не увеличивается. Компонент **`bigalloc`** более не поддерживается.

ПРИМЕЧАНИЕ:
Как и файловая система ext3, том с ext4 должен быть отмонтирован прежде, чем выполнять ***`fsck`****.*

Основные возможности.

Ext4 использует экстенты (в отличии от традиционной системы назначения блока, используемой в ext2 и ext3), которые увеличивают производительность, когда используются большие файлы и сокращает переизбыток метаданных для больших файлов. В дополнение, ext4 маркирует неразмеченные блоки в группы и секции таблицы inode соответственно, это позволяет их пропускать во время проверки файловой системы. Это позволяет осуществлять быструю проверку файловой системы, которая становится все больше.

Возможности выделения пространства.

Возможности файловой системы ext4 заключаются в следующих схемах выделения пространства:

- Постоянное предварительное выделение (Persistent pre-allocation).
- Отложенное выделение (Delayed allocation).
- Многоблочное выделение (Multi-block allocation).
- Чередующееся выделение (Stripe-aware allocation).

Поведение ext4 при записи файлов на диск отличается от ext3, так как присутствует отложенное выделение и прочие возможности оптимизации производительности. В ext4, когда программа пишет на файловую систему, это не гарантирует что ее данные находятся на диске, до тех пор пока программа не вызовет **`fsync()`** после записи.

По умолчанию ext3 автоматически применяет вновь созданные файлы на диск незамедлительно и без **`fsync()`**. Это позволяет избежать ошибок в программах, которые не используют **`fsync()`**, чтобы убедиться, что записываемые данные находятся на диске. С другой стороны, файловая система ext4, ожидает несколько секунд прежде чем записать изменения на диск, это позволяет комбинировать и выстраивать в нужном порядке операции записи, что дает большую производительность диска, чем у ext3.



ПРИМЕЧАНИЕ:

*В отличии от ext3, файловая система ext4 не применяет данные к диску по завершении транзакции. Она собирает их в буфер, прежде чем выгрузить на диск. Как и в других файловых системах, используйте вызовы проверки целостности данных, такие как **`fsync()`**, чтобы убедиться в нахождении данных на диске.*

Прочие возможности ext4.

Файловая система Ext4 также поддерживает следующее:

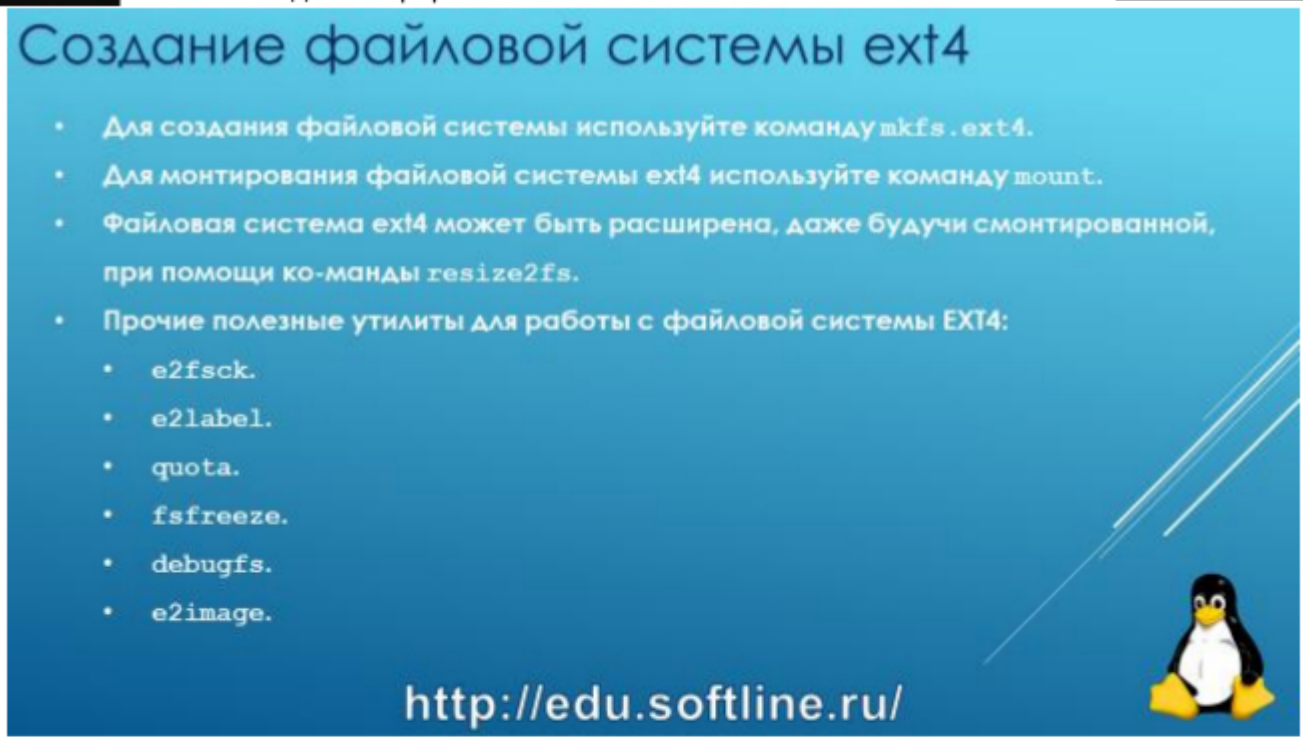
- Расширенные атрибуты (**`xattr`**) – это позволяет связывать с фалом несколько дополнительных пар имен и значений с фалом.
- Журналирование квоты – это позволяет избежать длительных проверок на целостность квоты после аварий.



ПРИМЕЧАНИЕ:

*Единственный поддерживаемый тип журналирования в ext4 – это **`data=ordered`** (по умолчанию).*

- Временные метки в миллисекундах – создание временных меток с точностью до миллисекунды.



Для создания файловой системы используйте команды **mkfs.ext4**. Опции по умолчанию в большинстве случаев оптимальны:

```
# mkfs.ext4 /dev/device
```

Вывод команды содержит информацию о геометрии файловой системы и возможностях.

```
~]# mkfs.ext4 /dev/sdb1
mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
245280 inodes, 979456 blocks
48972 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=1006632960
30 block groups
32768 blocks per group, 32768 fragments per group
8176 inodes per group
Superblock backups stored on blocks:
32768, 98304, 163840, 229376, 294912, 819200, 884736
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done
```

Для чередующихся блочных устройств, (например, массивов RAID5), чередующаяся геометрия может быть указана в момент создания файловой системы. Использование точного указания чередующейся геометрии существенно увеличивает производительность файловой системы ext4.

При создании файловой системы на LVM или MD томах, **mkfs.ext4** выбирает оптимальную геометрию. То же самое происходит с некоторыми аппаратными RAID, которые передают информацию о геометрии в операционную систему.

Для принудительного указания чередующейся геометрии используйте опцию **-E** команды **mkfs.ext4** (это расширенные опции файловой системы) ей нужно передать дополнительные параметры:

stride=*значение*

Указывает размер блока RAID.

stride-with=*значение*

Указывает число дисков с данными в RAID устройстве или число чередующихся элементов в наборе.

Для обоих параметров значение может быть указано в элементах блока файловой системы. Например, для создания файловой системы с чередованием 64 Кб (16 x 4096) на файловой системе с блоком 4 Кб, используйте следующую команду:

```
# mkfs.ext4 -E stride=16, stride-with=64 /dev/device
```



ПРИМЕЧАНИЕ:

*Имеется возможность, при помощи **tune2fs**, включить некоторые возможности ext4 на файловой системе ext3 и использовать драйвер ext4 для монтирования файловой системы ext3. Red Hat не рекомендует эти действия к применению, так как не были до конца протестированы в RHEL7 (CentOS 7) и никто не гарантирует производительность и целостность файловой системы.*

Также имеется возможность добавить UUID к файловой системе и использовать его для монтирования.

Монтирование файловой системы EXT4.

Файловая система может быть смонтирована без добавления особых опций. Например:

```
# mount /dev/device /mount/point
```

Файловая система поддерживает несколько опций монтирования, которые влияют на ее работу. Например, параметр **acl** включает поддержку списков контроля доступа, в то время, как параметр **user_xattr** включает расширенные атрибуты для пользователей. Для включения обоих параметров передайте их с помощью ключа **-o** команды **mount**:

```
# mount -o acl,user_xattr /dev/device /mount/point
```

Как и в случае с ext3 опция **data_err=abort**, может быть использована для прерывания ведения журнала в случае возникновения ошибки в файле данных.

```
# mount -o data_err=abort /dev/device /mount/point
```

Утилита tune2fs, также позволяет администраторам установить опции монтирования по умолчанию в суперблоке файловой системы. За дополнительной информацией обратитесь к man tune2fs.

Запись барьеров

По умолчанию, ext4 использует барьеры для того, чтобы убедиться в целостности файловой системы, когда происходит обрыв электроэнергии на устройстве с включенной записью в кэш. Для устройств без записи кэша или использующих дополнительные средства защиты от перебоев электроэнергии, отключите барьеры при помощи опции **nobarrier**:

```
# mount -o nobarrier /dev/device /mount/point
```

Изменение размера файловой системы EXT4.

Прежде чем расширять файловую систему ext4, убедитесь в том, что конечное блочное устройство обладает необходимым объемом. Используйте подходящий метод изменения размера для вашего блочного устройства:

Файловая система ext4 может быть расширена, даже будучи смонтированной, при помощи команды **resize2fs**:

```
# resize2fs /mount/device размер
```

Аналогично команда **resize2fs** работает с не смонтированными устройствами:

```
# resize2fs /dev/device размер
```

В процессе изменения файловой системы утилита **resize2fs** читает размер в единицах размера блоков файловой системы, для удобства можно использовать соответствующие суффиксы:

- **s** - сектора (512 байт).
- **K** – Килобайты (Кб).
- **M** – Мегабайты (Мб).
- **G** – Гигабайты (Гб).



ПРИМЕЧАНИЕ:

*Указание размера является опциональным и в процессе увеличения, если не указан размер **resize2fs** использует максимум свободного места на логическом томе или разделе.*

За дополнительной информацией обратитесь к **man resize2fs**.

Прочие полезные утилиты для работы с файловой системы EXT4.

e2fsck

Используется для восстановления файловой системы. Эта утилита проверяет и исправляет файловую систему ext4, работает гораздо эффективней с ext4 по сравнению с ext3, благодаря изменениям в дисковой структуре ext4.

e2label

Изменяет лейбл на файловой системе ext4. Аналогично работает с ext2 и ext3.

quota

Контролирует и сообщает об использованном месте (блоки) и файлах (иноды) пользователей и групп на файловой системе.

fsfreeze

Для приостановки доступа к файловой системе используйте команду:

```
# fsfreeze -f mount-point
```

Для возобновления доступа используйте ключ **-u**:

```
# fsfreeze -u mount-point
```

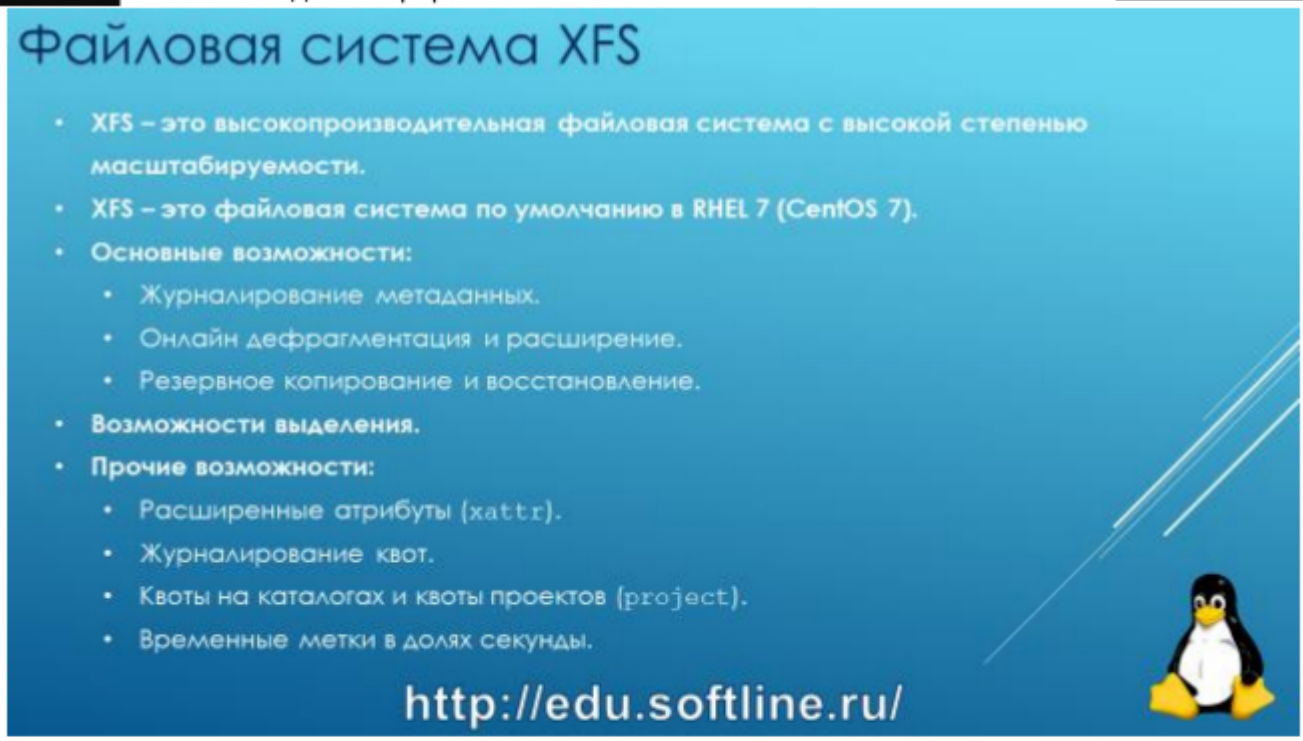
fsfreeze останавливает доступ к файловой системе и создает стабильный образ на диске. За дополнительной информацией обратитесь к странице **man fsfreeze(8)**.

debugfs

Отлаживает файловые системы ext2, ext3, ext4.

e2image

Сохраняет критические метаданные файловых систем ext2, ext3, ext4 в файл.



Файловая система XFS.

XFS это высокопроизводительная файловая система с высокой степенью масштабируемости изначально разработанная в 1993 году (представлена в 1994) в Silicon Graphics (SGI). XFS – это файловая система по умолчанию в CentOS 7 (RHEL 7).

Основные возможности.

XFS поддерживает журналирование метаданных, которое обеспечивает быстрое восстановление при авариях. Файловая система, будучи смонтированной и находясь в использовании, поддерживает возможность дефрагментации и расширения. В дополнение CentOS 7 (RHEL 7) содержит специальные утилиты для резервного копирования и восстановления XFS.

Возможности выделения.

XFS поддерживает следующие схемы выделения:

- Выделение на основе экстентов (Extent-based allocation).
- Политики выделения с чередованием (Stripe-aware allocation policies).
- Отложенное выделение (Delayed allocation).
- Предварительное выделение пространства (Space pre-allocation).

Отложенное выделение и прочие функции оптимизации влияют на XFS также, как и на ext4. Фактически, когда программа производит запись на диск, данные на диске гарантированно появляются только после вызова **`fsync()`**.



ПРИМЕЧАНИЕ:

Создание или расширение файла может закончиться неудачей с ошибкой записи (ENOSPC), даже если места на диске хватает. Это происходит из-за ориентации файловой системы на производительность. На практике такое происходит крайне редко, только в том случае, когда свободное место осталось лишь в нескольких блоках.

Прочие возможности.

Файловая система XFS, также поддерживает следующее:

- *Расширенные атрибуты (`xattr`)* – это позволяет системе прикреплять к файлам несколько дополнительных атрибутов (пар имен и значений). Включено по умолчанию.
- *Журналирование квот* – позволяет избежать необходимости в продолжительных проверках квот после аварий.
- *Квоты на каталогах и квоты проектов (Project)* – позволяет выставлять ограничения квотами на деревьях каталогов.
- *Временные метки в долях секунды* – позволяет использовать временные метки с точностью до миллисекунд.



Миграция с файловой системы ext4 на XFS.

Основное изменение в CentOS 7 (RHEL 7) – это переход с файловой системы ext4 на файловую систему XFS по умолчанию. В данном разделе мы разберем отличия, возникающие при использовании и администрировании файловой системы XFS.



ПРИМЕЧАНИЕ:

Файловая система ext4 полностью поддерживается в CentOS 7 (RHEL 7) и при необходимости может быть выбрана на этапе установки операционной системы. Не смотря на возможность миграции с ext4 на XFS, это вовсе не обязательно.

Команды и утилиты для работы с ext3/ext4 и их аналогия для XFS.

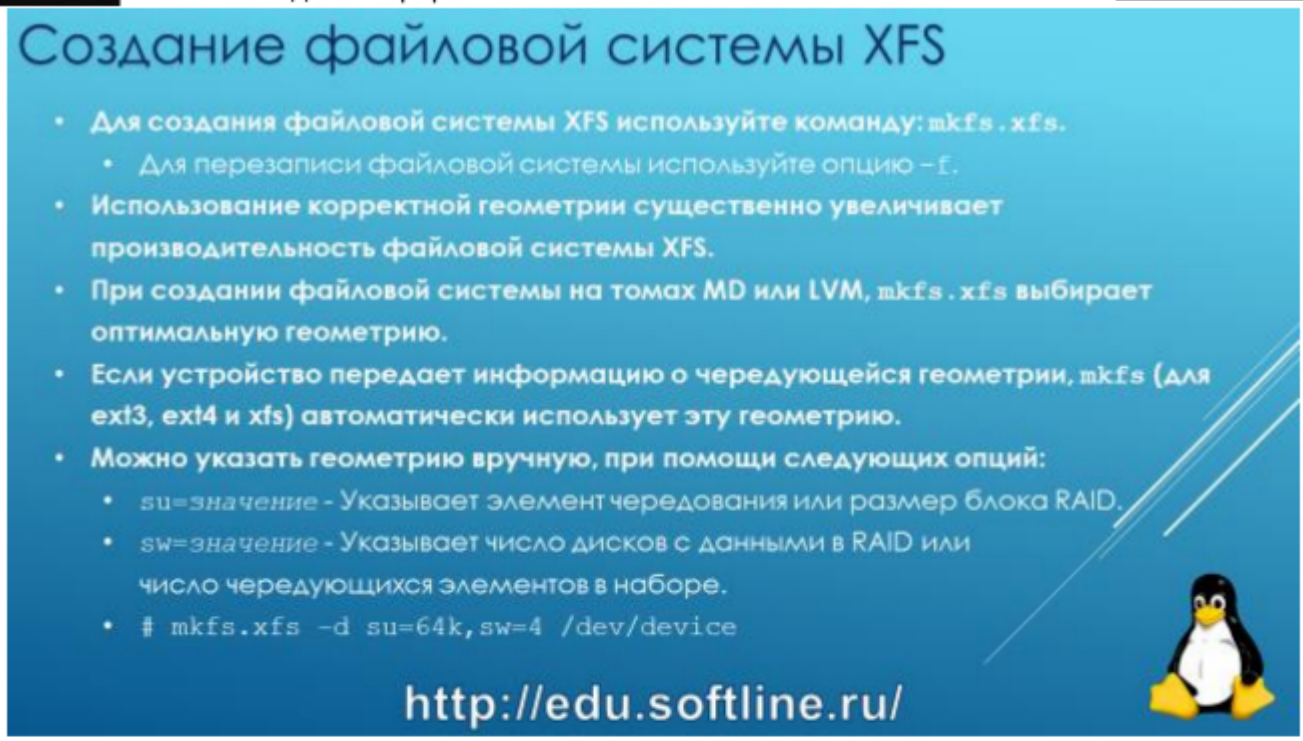
Основные команды для ext3/ext4 и XFS.

Задача	ext3/ext4	XFS
Создание	**mkfs.ex3** и **mkfs.ext4**	**mkfs.xfs**
Проверка	**e2fsck**	**xfs_repair**
Изменение размера	**resize2fs**	**xfs_growfs**
Сохранение образа	**e2image**	**xfs_metadump** и **xfs_mdrestore**
Маркировка и настройка	**tune2fs**	**xfs_admin**
Резервное копирование и восстановление	**dump** и **restore**	**xfsdump** и **xfsrestore**

Основные инструменты для ext4 и XFS.

Задача	ext4	XFS
Квоты	**quota**	**xfs_quota**
Разметка файлов	**filefrag**	**xfs_bmap**

У каждого инструмента имеется собственное комплексное описание на страницах **man**.



Для создания файловой системы XFS используйте команду: **`mkfs.xfs /dev/device`**. Опции по умолчанию как правило оптимальны.

Когда вы используете **`mkfs.xfs`** на блочном устройстве, содержащем файловую систему, необходимо использовать опцию **`–f`** для принудительной перезаписи файловой системы.

Пример вывода команды **`mkfs.xfs`**:

```
meta-data =/dev/device     isize=256 agcount=4,     agsize=3277258 blks
         =                 sectsz=512               attr=2
Data     =                 bsize=4096               blocks=13109032, imaxpct=25
         =                 sunit=0                  swidth=0 blks
Naming   =version 2        bsize=4096               ascii-ci=0
Log      =internal log     bsize=4096               blocks=6400, version=2
         =                 sectsz=512               sunit=0 blks, lazy-count=1
Realtime =none             extsz=4096               blocks=0, rtextents=0
```

ПРИМЕЧАНИЕ:
После создания файловой системы XFS, ее размер не может быть уменьшен. Но может быть увеличен при помощи команды ***`xfs_growfs`***.

Для чередующихся блочных устройств (например, массивов RAID5), чередующаяся геометрия может быть указана на этапе создания файловой системы. Использование корректной чередующейся геометрии существенно увеличивает производительность файловой системы XFS.

При создании файловой системы на томах MD или LVM, **`mkfs.xfs`** выбирает оптимальную геометрию. То же самое происходит на некоторых аппаратных RAID-контроллерах, колторые передают геометрию операционной системе.

Если устройство передает информацию о чередующейся геометрии, **`mkfs`** (для ext3, ext4 и xfs) автоматически использует эту геометрию, если это не возможно можно указать геометрию вручную, при помощи следующих опций:

su=*значение*

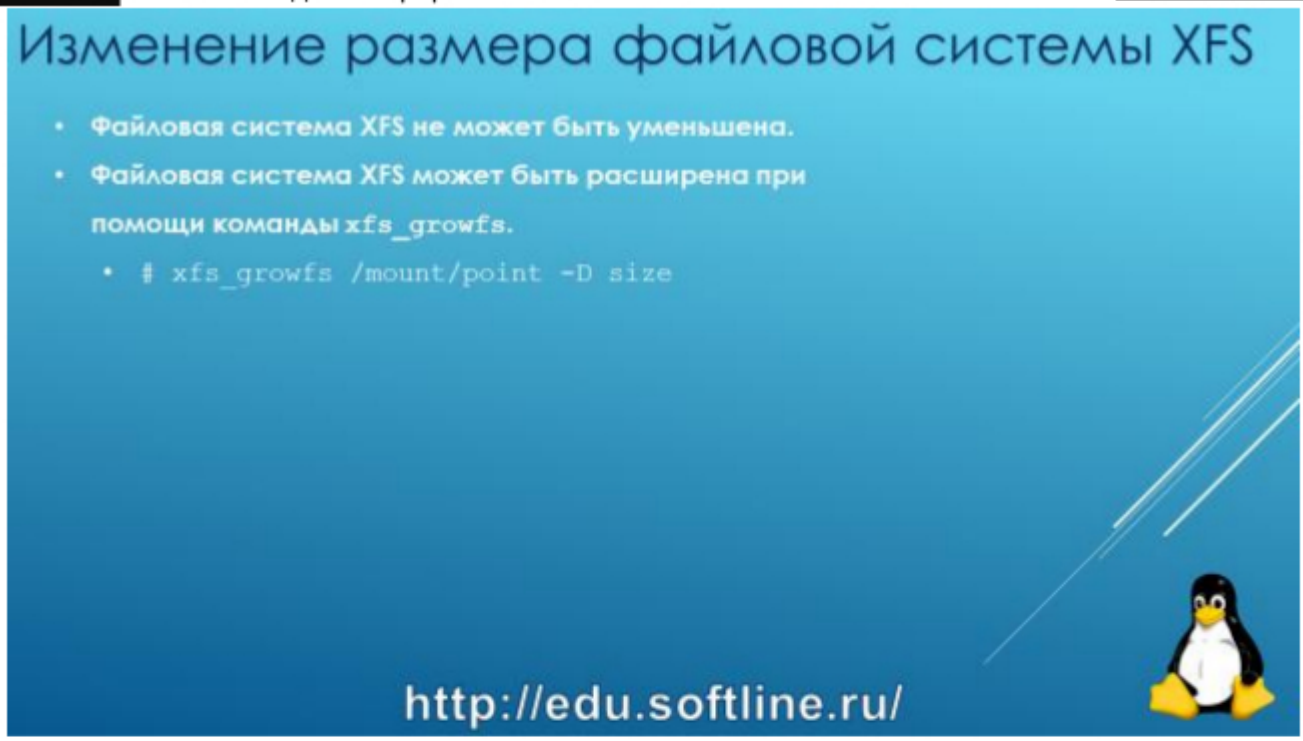
Указывает элемент чередования или размер блока RAID. Значение должно быть указано в байтах или с опциональным суффиксом: **k**, **m**, **g**.

sw=*значение*

Указывает число дисков с данными в RAID или число чередующихся элементов в наборе.

В следующем примере указан размер блока 64k на RAID устройстве, содержащем 4 чередующихся элемента.

```
# mkfs.xfs -d su=64k,sw=4 /dev/device
```



Файловая система XFS может быть расширена, даже если она смонтирована, при помощи команды **xfs_growfs**:

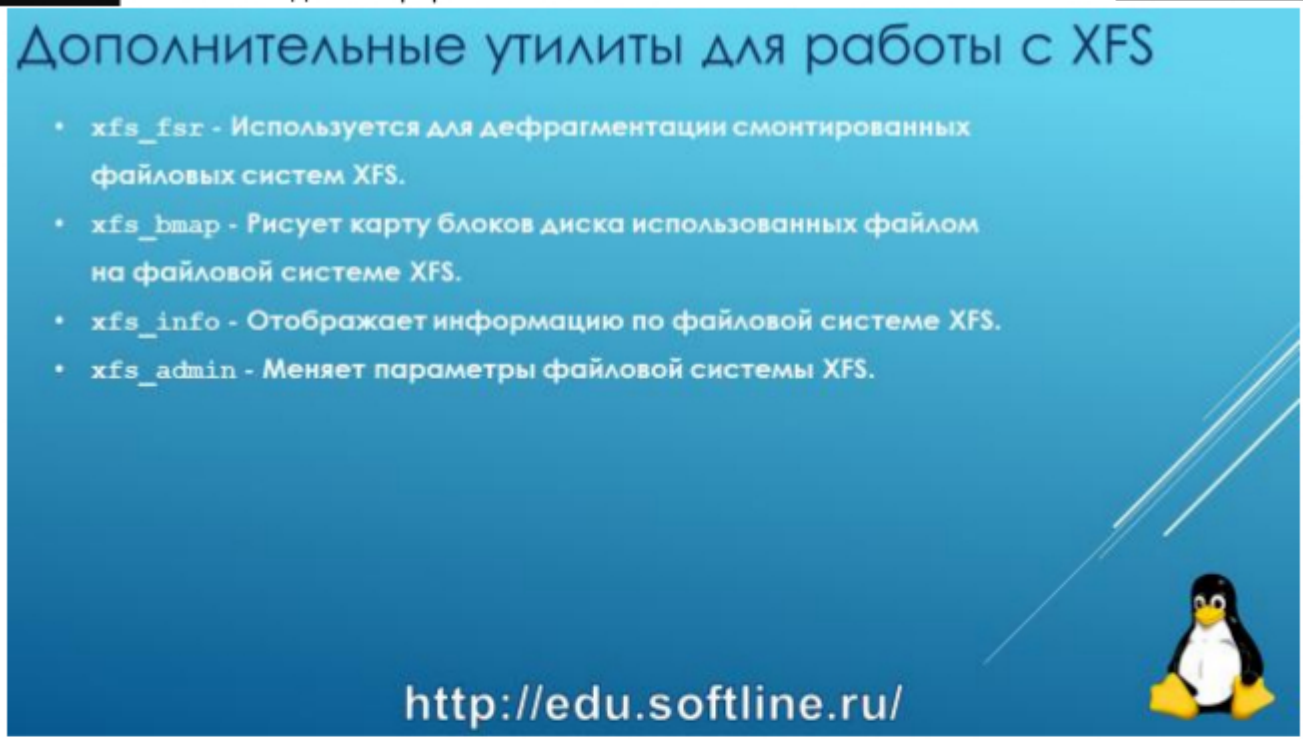
```
# xfs_growfs /mount/point -D size
```

Ключ **-D** увеличивает размер файловой системы до указанного (***size***, задается в блоках файловой системы). Без указания ключа **-D** size, **xfs_growfs** увеличит размер файловой системы до максимально доступного на носителе.

Прежде чем увеличивать файловую систему XFS при помощи **-D *size***, убедитесь, что на блочном устройстве есть необходимый объем свободного места. Для расширения файловой системы используйте метод, советующий вашему блочному устройству.

ПРИМЕЧАНИЕ:
Файловая система XFS не может быть уменьшена не в смонтированном не в отмонтированном состоянии.

За дополнительной информацией обращайтесь к **man xfs_growfs**.



Прочие полезные утилиты для работы с файловой системой XFS.

CentOS 7 (RHEL 7) также содержит дополнительные утилиты для управления файловой системой XFS.

xfs_fsr

Используется для дефрагментации смонтированных файловых систем XFS. Если вызвать без аргументов, **xfs_fsr** дефрагментирует все обычные файлы на всех смонтированных файловых системах XFS. Эта утилита также позволяет приостановить дефрагментацию в нужное время и затем возобновить ее выполнение.

В дополнение **xfs_fsr** позволяет дефрагментировать отдельный файл: **xfs_fsr /path/to/file**. Red Hat не рекомендует периодически дефрагментировать всю файловую систему. Это связано с тем, что XFS как правило избегает фрагментации файловой системы и полная дефрагментация может привести к фрагментации свободного места на диске.

xfs_bmap

Рисует карту блоков диска использованных файлом на файловой системе XFS. Карта выводит каждый экстент, использованный указанным файлом, а также области в файле не связанные с блоками (дыры).

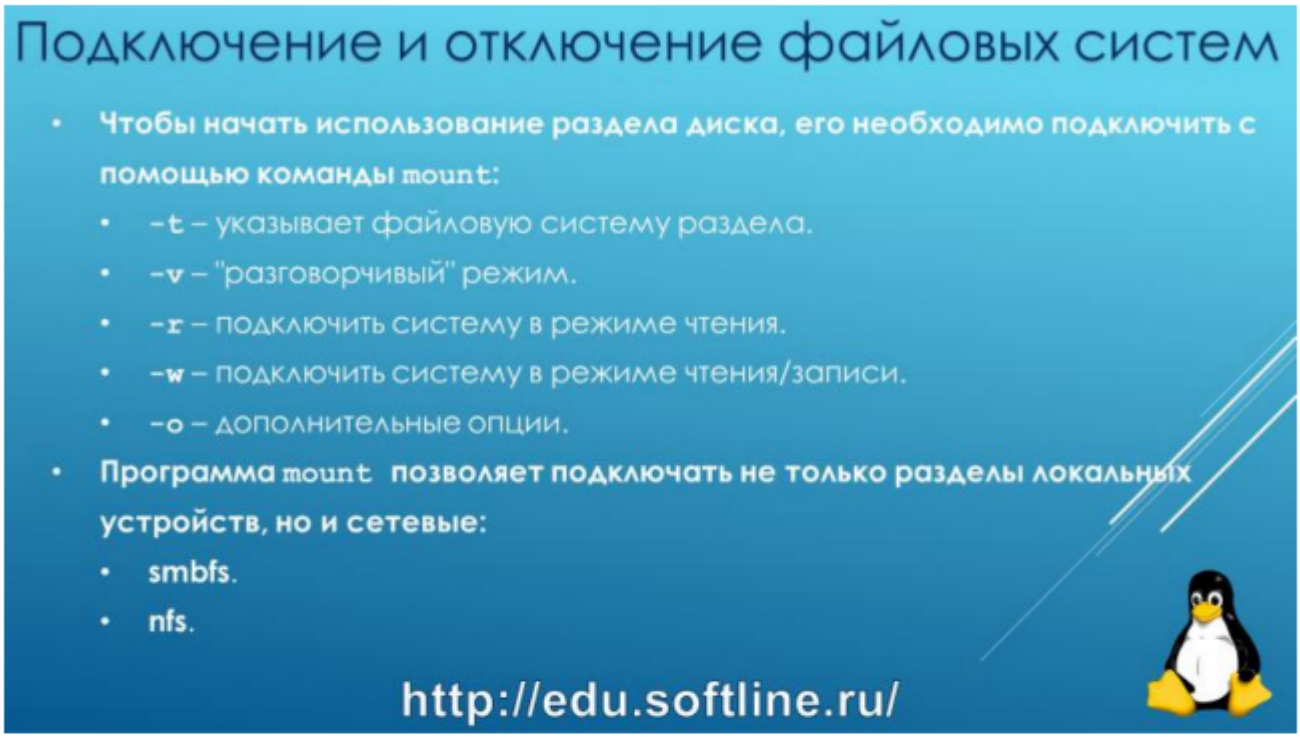
xfs_info

Отображает информацию по файловой системе XFS.

xfs_admin

Меняет параметры файловой системы XFS. Утилита xfs_admin может менять параметры только не смонтированных устройств или файловых систем.

За дополнительной информацией по данным утилитам обратитесь к страницам **man**.



Программа **mount** используется для «*монтирования*» (подключения) файловых систем, находящихся на различных устройствах или доступных посредством сети.

Запущенная без аргументов, mount прочитает файл `/etc/mtab` и выведет информацию обо всех подключенных файловых системах:

```
# mount
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=620)
/dev/sda1 on /boot type ext3 (rw)
/dev/sda2 on / type ext3 (rw)
tmpfs on /dev/shm type tmpfs (rw)
```

Для подключения файловой системы необходимо указать её тип, устройство, на котором находится эта файловая система, и точку монтирования – каталог, в котором будет доступна подключаемая ФС:

```
# mount -t iso9660 /dev/cdrom /mnt/cdrom
```

Для отключения файловой системы используется команда **umount**, принимающая в качестве аргумента либо точку монтирования, либо имя устройства:

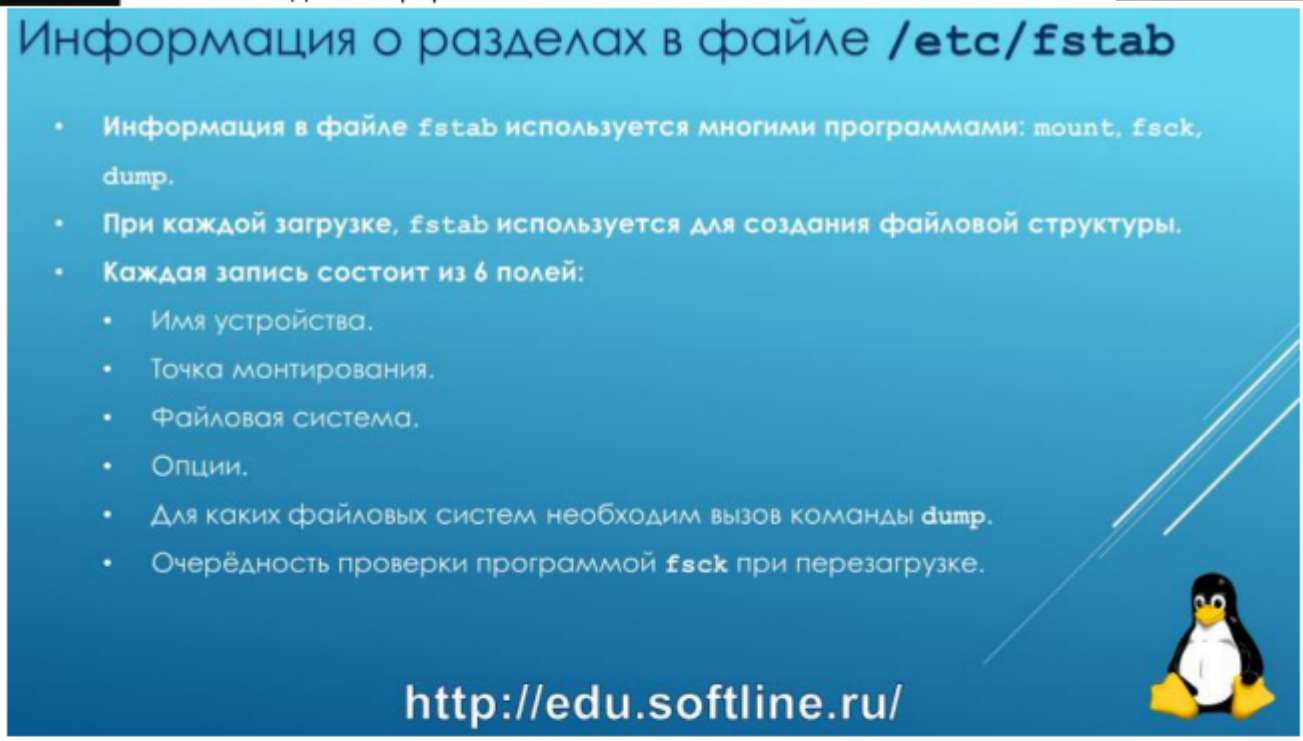
```
# umount /dev/cdrom
```

Если файловая система используется, то есть пользователями открыты файлы, находящиеся на ней, то отключить такую систему не удастся. Для решения этой проблемы используется команда **fuser**, определяющая процессы, использующие файлы, чтобы просмотреть информацию о процессах, использующих файлы, дайте команду:

```
# fuser -v /mnt/cdrom
```

Чтобы завершить все процессы, дайте команду:

```
# fuser -km /mnt/cdrom
```



Файл **/etc/fstab** является очень важным для системы, информацию, хранящуюся в нём, используют программы **mount**, **fsck** и многие другие. На основании этого файла производится подключение файловых систем в процесса загрузки операционной системы.

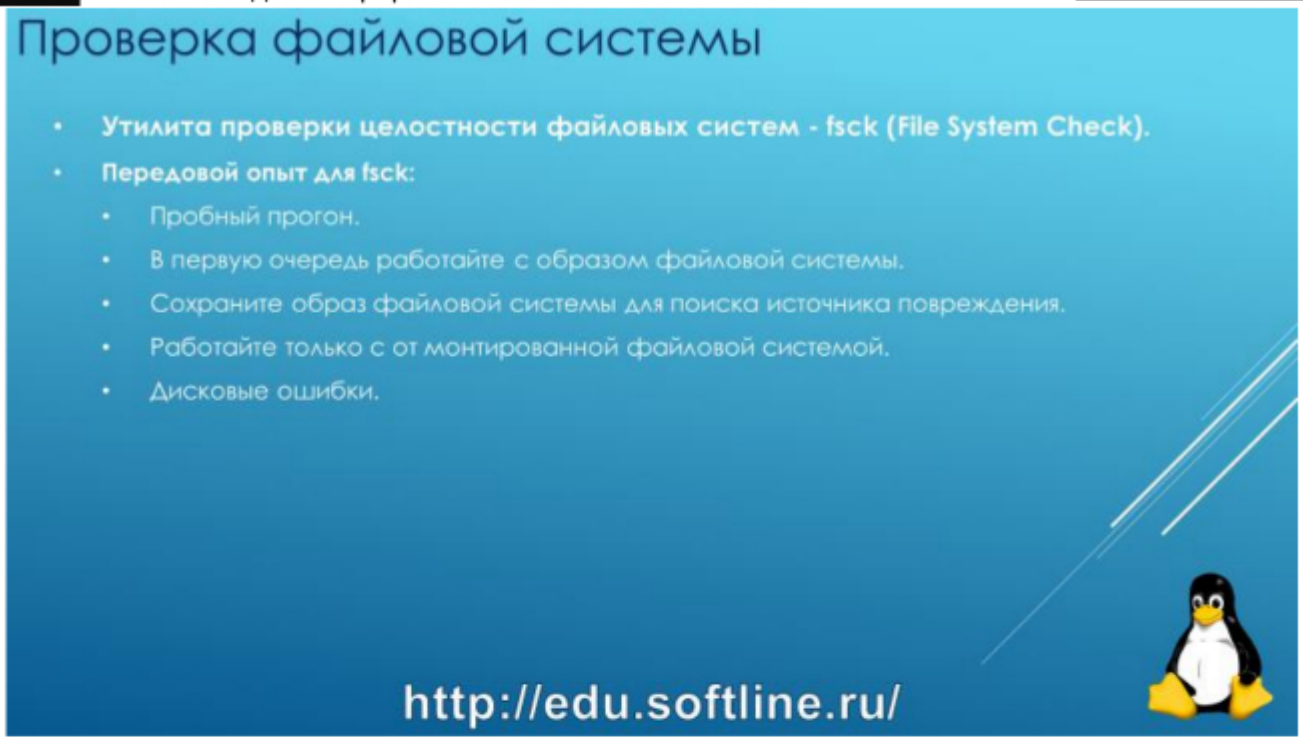
В файле находится множество записей, по одной на каждую файловую систему. Каждая запись состоит из 6-ти полей:

- **Имя устройства** – в этом поле необходимо указать устройство, для которого предназначена эта запись. В качестве устройства можно указать либо имя в файловой системе **/dev**, либо метку тома.
- **Точка монтирования** – указание каталога в структуре файловой системы, в который данная файловая система должна быть подключена.
- **Тип файловой системы** – в качестве типа файловой системы указывается любой из типов, поддерживаемых программой **mount**.
- **Опции** – различные опции из тех, что предлагает программа **mount**. Опции разделяются запятой.
- **Частота вызова программы dump** – 1-ежедневно, 0-никогда.
- **Очерёдность выполнения проверки командой fsck** – 0-не проверять, 1-первый, 2-второй и так далее до 9-ти.

Во время запуска файловой системы и выполнения скрипта **rc.sysinit**, используется файл **/etc/fstab**.

Также, **fstab** используется программой **mount**. Если администратор пытается подключить файловую систему, имеющую описание, то достаточно передать команде **mount** один аргумент – устройство:

```
# mount /dev/sda1
```



Файловая система может быть проверена на целостность и опционально исправлена, при помощи специфичных утилит для файловой системы. Как правило эти утилиты имеют отношение к **`fsck`** (сокращение от File System Check).

ПРИМЕЧАНИЕ:
Эти утилиты для проверки файловых систем, гарантируют только целостность метаданных, они не умеют работать с реальными данными, содержащимися на файловой системе и не являются инструментами восстановления данных.

Нарушение целостности файловой системы может случить по различным причинам, в том числе из-за аппаратных ошибок, ошибок администратора хранилища и программных неисправностей.

До того, как современные файловые системы с ведением журнала метаданных стали популярными, каждый сбой электропитания и некорректная перезагрузка требовали проверки файловой системы. Это происходило из-за того, что каждое прерванное обновление файловой системы, приводило ее в состояние нарушения целостности. В результате, проверка каждой файловой системы, указанной в **`/etc/fstab`** автоматически запускалась во время загрузки. Для файловых систем с журналированием метаданных, эта операция проходит очень быстро, так как при помощи журнала метаданных проверить целостность значительно проще, даже после аварии.

Однако, возможны ситуации нарушения целостности файловой системы и повреждения данных, даже для журналируемых файловых систем. В таких случаях необходимо использовать проверщик для починки файловой системы.

ПРИМЕЧАНИЕ:
Для CentOS (RHEL) не рекомендуется отключать проверку при загрузке за исключением, машин, которые не используют загрузку с диска, очень больших файловых систем и файловых систем на удаленном хранилище.

Передовой опыт для `fsck`.

Обычно, при запуске утилиты проверки и починки файловой системы, в случае нахождения нарушений целостности, происходит автоматическая починка файловой системы. В некоторых случаях, серьезно поврежденные иноды (inodes) или каталогу могут быть удалены, если их невозможно починить. Чтобы убедить что неожиданные и нежелательные изменения не произойдут, воспользуйтесь следующими шагами:

Пробный прогон.

Большинство проверщиков файловых систем поддерживают режим, который проверяет, но не исправляет файловую систему. В этом режиме проверщик выведет любые найденные ошибки и действия, которые он может предпринять, не внося никаких изменений на файловую систему.

В первую очередь работайте с образом файловой системы.

Большинство файловых систем поддерживают создание образа метаданных (Metadata Image) - выборочной копии файловой системы, которая содержит только метаданные. Так как проверщик файловой системы работает только с метаданными, можно выполнить пробный прогон на образе метаданных, чтобы оценить какие изменения будут внесены. Если изменения будут приемлемыми, то затем можно выполнить операцию почини на файловой системе.



ПРИМЕЧАНИЕ:

Серьезно поврежденная файловая система может вызвать проблему при создании образа метаданных.

Сохраните образ файловой системы для поиска источника повреждения.

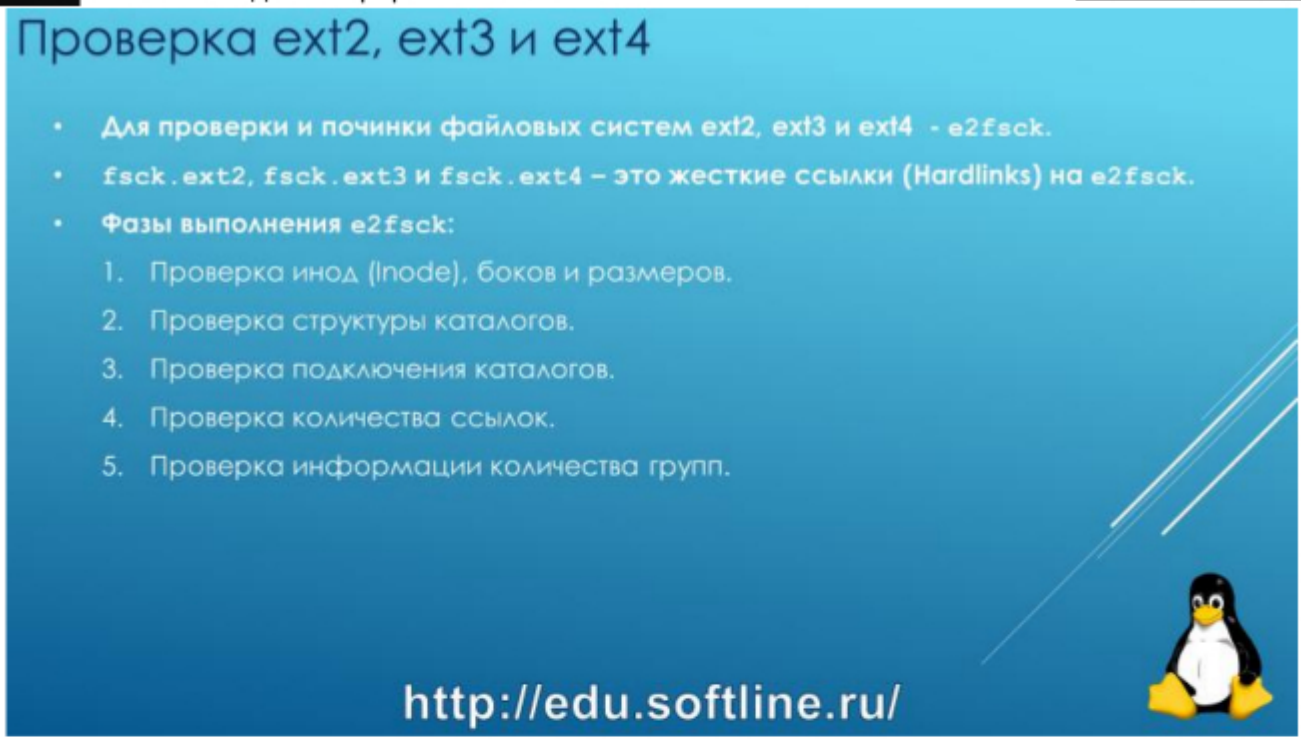
Образ мета данных, созданный до операции починки файловой системы, может быть полезен при поиске причин повреждения файловой системы, вызванной ошибкой программного обеспечения.

Работайте только с от монтированной файловой системой.

Починка файловой системы может быть выполнена только на отмонтированной файловой системе. Инструмент починки должен иметь индивидуальный однопользовательский доступ к файловой системе, чтобы избежать дальнейшего повреждения файловой системы. Некоторые инструменты позволяют выполнять проверку без починки на смонтированной файловой системе, такая проверка может показать не существующие ошибки, если файлы будут заняты сторонними процессами.

Дисковые ошибки.

Инструменты проверки файловой системы не могут исправить аппаратные проблемы. Файловая система должна быть полностью доступна для чтения и записи, чтобы операция починки завершилась успешно. Если файловая система повреждена в следствии аппаратных ошибок, ее можно переместить на другой диск при помощи утилиты **`dd(8)`**.



Файловые системы **ext2**, **ext3** и **ext4** используют исполняемый файл **e2fsck** для проверки и починки файловой системы. Имена файлов **fsck.ext2**, **fsck.ext3** и **fsck.ext4** – это жесткие ссылки (Hardlinks) на исполняемый файл **e2fsck**. Эти исполняемые файлы запускаются автоматически во время загрузки и их поведение зависит от состояния файловой системы и момента запуска проверки.

Полная проверка вызывается на файловой системе **ext2**, которая не поддерживает ведение журнала метаданных файловой системы и для **ext4**, если отключено журналирование.

Для файловых систем **ext3** и **ext4** с журналированием метаданных, журнал повторно выполняет в пространстве пользователя и исполняемый файл завершает свою работу. Это является действием по умолчанию, так как повторной выполнение журнала обеспечивает целостность файловой системы после аварии.

Если обнаруживается нарушение целостности файловой системы, когда она смонтирована, эта информация записывается в суперблок. Если **e2fsck** находит эту отметку в суперблоке, то она выполняет полную проверку после повторного применения журнала (если журнал ведется).

e2fsck может запросить пользовательский ввод в процессе выполнения, если не указана опция **-p**. Опция **-p** сообщает **e2fsck** автоматически выполнять все исправления, если они могут выполниться безопасно. Если требуется участие пользователя **e2fsck** отображает этот статус в выходном коде.

ПРИМЕЧАНИЕ:

*Выходные коды (Exit Code) возвращаемые **e2fsck**, это суммы следующих статусов:*

- ***0** – ошибки отсутствуют.*
- ***1** – ошибки файловой системы исправлены.*
- ***2** – ошибки файловой системы исправлены, система должна быть перезагружена.*
- ***4** – ошибки файловой системы остались неисправленными.*
- ***8** – ошибка выполнения.*
- ***16** – ошибка использования или синтаксиса.*
- ***32** – **e2fsck** была отменена пользователем.*
- ***128** – ошибка общей библиотеки.*

Популярные опции **e2fsck**:

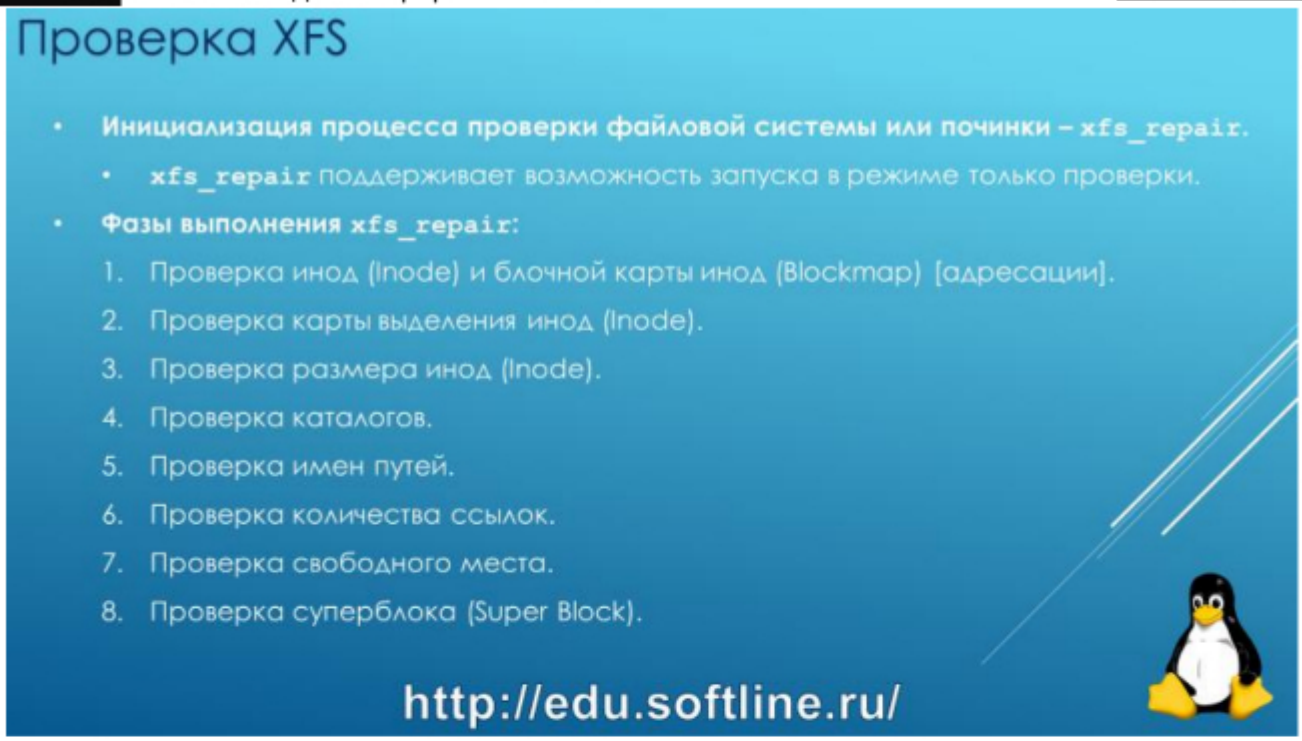
-n	Режим без изменений, только проверка.
-b *суперблок*	Указывает номер блока, в качестве альтернативы суперблоку, если первый поврежден.
-f	Принудительный запуск полной проверки, даже если в суперблоке не содержится информации об ошибках.
-j *журнал*	Указывает путь к журналу.
-p	Автоматически чинит или «приводит в порядок» файловую систему, без необходимости пользовательского ввода.
-y	Принимает ответ «да» на все выдаваемые вопросы.

Дополнительную информацию по всем опциям можно найти на странице руководства **e2fsck(8)**.

Следующие пять фаз выполняются, когда запущена **e2fsck**:

1. Проверка инод, боков и размеров.
2. Проверка структуры каталогов.
3. Проверка подключения каталогов.
4. Проверка количества ссылок.
5. Проверка информации количества групп.

Утилита **e2image(8)** может быть использована для создания образа метаданных до операции починки или в целях тестирования и диагностики. Опция **-r** должна быть использована в целях тестирования для создания выборочного файла размером с файловую систему. **e2fsck** может напрямую работать с результирующим файлом. Опция **-Q** должна быть использована для указания на образ, который должен быть заархивирован или использован для диагностики. Это создает более компактный формат файла, подходящий для передачи.



Никакие операции восстановления не выполняются автоматически в процессе загрузки. Для инициализации процесса проверки файловой системы или починки, должна быть использована утилита **xfs_repair**.

ПРИМЕЧАНИЕ:

*В пакете **xfsprogs** содержится утилита **fsck.xfs**, она предоставлена только для соответствия инициализирующим скрипатом, которые выглядят следующим образом **fsck.**фс. Утилита **fsck.xfs** немедленно завершает свою работу с выходным кодом 0.*

*Более старая версия пакета **xfsprogs** содержит утилиту **xfs_check**. Это очень медленная утилита, не поддерживающая большие файловые системы. В результате она устарела и более не используется, так как появилась возможность запускать **xfs_repair -n**.*

Чистый журнал на файловой системе требуется для работы **xfs_reapir**. Если файловая система не была корректно отмонтирована, она должна быть смонтирована и отмонтированна заново перед использованием **xfs_repair**. Если журнал поврежден и не может быть воспроизведен, можно использовать опцию **-L** для обнуления журнала.

ПРИМЕЧАНИЕ:

*Опция **-L** должна быть использована только в том случае, когда журнал не может быть воспроизведен. Опция отменяет все обновления метаданных в журнале, что может привести к нарушению. Целостности в будущем.*

xfs_reapir поддерживает возможность запуска в режиме только проверки при помощи опции **-n**. В этом случае никакие изменения не будут внесены на файловую систему.

Наиболее часто используемые опции **xfs_repair**:

- **-n** – Режим проверки без изменений.
- **-L** – Обнуление журнала метаданных.

- **-m** *maxmem* – Ограничение памяти во время запуска до значения *maxmem* MB. **0** может быть указан для запуска с минимально возможным объемом памяти.
- **-l** *logdev* – Указывает внешнее устройство для ведения журнала.

Подробное описание опций и их использования представлены на странице руководства **xfs_reapir(8)**.

Следующие 8 основных фаз выполняет **xfs_repair** после запуска:

1. Проверка инод (Inode) и блочной карты инод (Blockmap) [адресации].
2. Проверка карты выделения инод (Inode).
3. Проверка размера инод (Inode).
4. Проверка каталогов.
5. Проверка имен путей.
6. Проверка количества ссылок.
7. Проверка свободного места.
8. Проверка суперблока (Super Block).

Эти фазы и сообщения, которые выводятся в процессе работы подробно описаны на странице руководства **xfs_reapir(8)**.

xfs_reapir не интерактивна. Все операции выполняются автоматически без участия пользователя.

Рекомендуется создать образ метаданных до начала починки для последующей диагностики и тестирования, для этого можно использовать утилиты **xfs_metadump(8)** и **xfs_mdrestore(8)**.



Избыточный массив независимых дисков (Redundant Array of Independent Disks, RAID).

Основной идеей, находящейся в основе RAID, является объединение нескольких, недорогих дисковых накопителей в массив для достижения производительности или отказоустойчивости, что недостижимо при помощи одного большого и дорогого накопителя.

RAID.

RAID позволяет получить доступ к нескольким дискам. RAID использует такие технологии, как чередование дисков (Disk Striping, RAID0), зеркалирование дисков (Disk Mirroring, RAID1) и чередование дисков с контролем четности (Disk Striping with Parity, RAID5) для достижения отказоустойчивости, меньшей задержки, увеличения пропускной способности и максимизации восстановления данных при выходе дисков из строя.

RAID целостно распространяет данные между каждым диском в массиве. RAID разбивает данные на равные части (Обычно 32Kb или 64 Kb, хотя можно использовать и другие размеры). Каждая часть записывается на диск в RAID массиве в соответствии с выбранным уровнем RAID. Когда данные считываются, процесс обращается, давая представление, что множеств дисков в массиве являются одним большим накопителем.

Преимущества от использования RAID.

Системные администраторы и прочие, кто управляет большими объемами данных получат преимущества от использования технологии RAID. Основные преимущества от использования RAID:

- Увеличенная скорость.
- Увеличенный объем хранения при помощи одного виртуального диска.
- Минимизация ошибок диска.

Типы RAID.

Существует три подхода к реализации RAID: встроенный микропрограммный RAID (Firmware RAID), аппаратный RAID (Hardware RAID) и программный RAID (Software RAID).

Встроенный микропрограммный RAID (Firmware RAID).

Встроенный микропрограммный RAID (также известный как ATARAID) – это тип программного RAID, при котором набор RAID настраивается при помощи микропрограммного обеспечение, которое обычно встроено в BIOS, что позволяет грузиться с этого RAID массива. Различные производители используют различные мета-данные диска для маркировки членов RAID-набора. Intel Matrix RAID – это хороший пример системы встроенного микропрограммного RAID.

Аппаратный RAID (Hardware RAID).

Массив на основе аппаратного RAID контроллера управляется независимо от хоста. Это предоставляет один диск на RAID массив хосту.

Аппаратные RAID устройства подключаются к SCSI контроллеру и предоставляют RAID массив в качестве одиночного SCSI накопителя. Внешняя RAID система перемещает всю обработку на отдельный контроллер, расположенный в отдельной дисковой подсистеме. Вся подсистема подключается к хосту через обычный SCSI контроллер и представляется хосту как один диск.

Карта RAID-контроллера функционирует как SCSI контроллер в операционной системе и обрабатывает все текущие дисковые коммуникации. К RAID контроллеру подключаются накопители (так же как к обычному SCSI-контроллеру) и затем добавляются в конфигурацию RAID контроллера и предоставляются операционной системе, которая не видит разницы между SCSI-контроллером и RAID-контроллером.

Программный RAID (Software RAID).

Программный RAID применяет различные уровни RAID в коде ядра диска (блочного устройства). Это предоставляет максимально дешёвое решением по сравнению с аппаратным, если не требуется горячая замена. Программный RAID работает и с дешёвыми дисками (IDE) и со SCSI-дисками. С современными быстрыми процессорами, программный RAID имеет преимущество над аппаратным RAID.

Ядро Linux содержит драйвер multi-disk (MD), который позволяет решению RAID быть полностью аппаратно-независимым. Производительность программного массива зависит от производительности и загруженности процессора.

Основные возможности программного RAID в Linux:

- Многопоточная структура.
- Возможность перемещение массивов между Linux-машинами без перенастройки.
- Фоновая реконструкция при помощи простаивающих системных ресурсов.
- Поддержка подключения накопителей на горячую.
- Автоматическое определение CPU для получения преимуществ от определенной оптимизации CPU.
- Регулярные проверки целостности данный RAID, для проверки здоровья массива.
- Интерактивный мониторинг массива с возможностью уведомления по почте о важных событиях.

- Битовые карты записи, которые увеличивают скорость событий ресинхронизации, указывая ядру какие части массива должны быть синхронизированы, вместо синхронизации всего массива.
- Моментальные снимки ресинхронизации, замораживающие синхронизацию на время перезагрузки системы.
- Возможность изменения массива после создания, например, расширения массива без остановки.



Уровни RAID и линейная поддержка.

RAID поддерживает различные конфигурации, в том числе уровни 0, 1, 4, 5, 6, 10 и линейный.

Уровень 0.

Уровень RAID 0, также называемый чередование (Striping) –это ориентированная на производительность техника чередования данных. Эта техника позволяет разбивать данные на части и записывать их на все диски массива, предоставляя высокую производительность ввода/вывода, низкую стоимость и не предоставляя отказоустойчивости. Объем хранилища в массиве с уровнем 0 равен сумме объемов дисков в аппаратном RAID или сумме разделов в программном RAID.

Уровень 1.

Уровень RAID 1 или зеркалирование используется дольше, чем любой другой уровень RAID. Уровень 1 предоставляет отказоустойчивость за счет записи данных на каждый диск в массиве, оставляя зеркальную копию на каждом диске. Зеркалирование остается популярным за счет простоты и высокого уровня доступности данных. Уровень 1 оперирует двумя или более дисками, что позволяет использовать параллельный доступ при чтении, но как правило оперируют независимо для предоставления высоких показателей ввода/вывода. Уровень 1 предоставляет очень хорошую надежность и увеличивает производительность приложений с высокими потребностями в чтении данных, при этом с достаточно высокой стоимостью. Объем массива с уровнем 1 равен объему одного диска в аппаратном RAID или одного массива в программном RAID.



ПРИМЕЧАНИЕ:

Для записи 10 Gb данных в RAID1 состоящем из двух дисков потребуется записать 20 Gb, по 10 Gb на каждый из накопителей.

Уровень 4.

Уровень RAID 4 использует четность, содержащуюся на одном диске для защиты данных. Это лучше подходит для операций ввода/вывода а не передачи больших файлов. Так как выделенный для четности диск выступает узким местом решения, поэтому уровень 4 редко используется без сопутствующих технологий, таких как кэширование. Также RAID уровня 4 не поддерживается в CentOS. Объем дискового пространства в аппаратных RAID уровня 4 равен сумме объемов дисков минус объем одного диска, выделенного для четности.



ПРИМЕЧАНИЕ:

Объем массива RAID4 равен объему массива RAID5, но уровень 5 имеет ряд преимуществ, поэтому уровень 4 не поддерживается в CentOS (RHEL).

Уровень 5.

Уровень RAID 5 наиболее популярный тип RAID. За счет распространения четности между несколькими или всеми дисками в массиве, уровень 5 не наследует узкое место при записи от уровня 4. Узким местом остается только вычисление четности, что не является проблемой для программных массивов RAID с современными процессорами. Также, как и в уровне 4 чтение значительно снижает скорость записи. Уровень 5 часто используется вместе с кэшированием, что существенно снижает асимметрию в производительности. Объем дискового пространства в аппаратных RAID уровня 5 равен сумме объемов дисков минус объем одного диска, а для программных равен сумме объемов разделов минус объем одного раздела.

Уровень 6.

Уровень 6 – типичный уровень RAID, при котором данные избыточны и сохранны, но менее производительны. RAID 6 использует комплексную схему четности, которая позволяет восстановиться при потере любых двух дисков в наборе. Эта комплексная схема четности создает гораздо больше нагрузок на процессоре в процессе записи, что существенно увеличивает асимметрию при записи по сравнению с уровнями 4 и 5.

Линейный RAID.

Линейный RAID – это простая группировка накопителей для создания одного большого виртуального накопителя. В линейном RAID части выделяются последовательно, заполняя накопители один за другим, переход к заполнению следующего накопителя происходит после полного заполнения предыдущего. Такая группировка не предоставляет никаких преимуществ, кроме одного большого виртуального накопителя, также снижается стабильность из-за потери одного накопителя весь массив будет непригоден. Объем линейного RAID равен сумме объемов дисков в массиве.

RAID-подсистема Linux

RAID в Linux состоит из следующих подсистем.

Драйвера аппаратного RAID-контроллера.

Аппаратный RAID-контроллер не имеет специально подсистемы в Linux, так как он использует специальный RAID-чипсет. Аппаратный RAID поставляется в комплекте с драйверами, которые позволяют системе определить наборы RAID, как обычные диски.

mdraid.

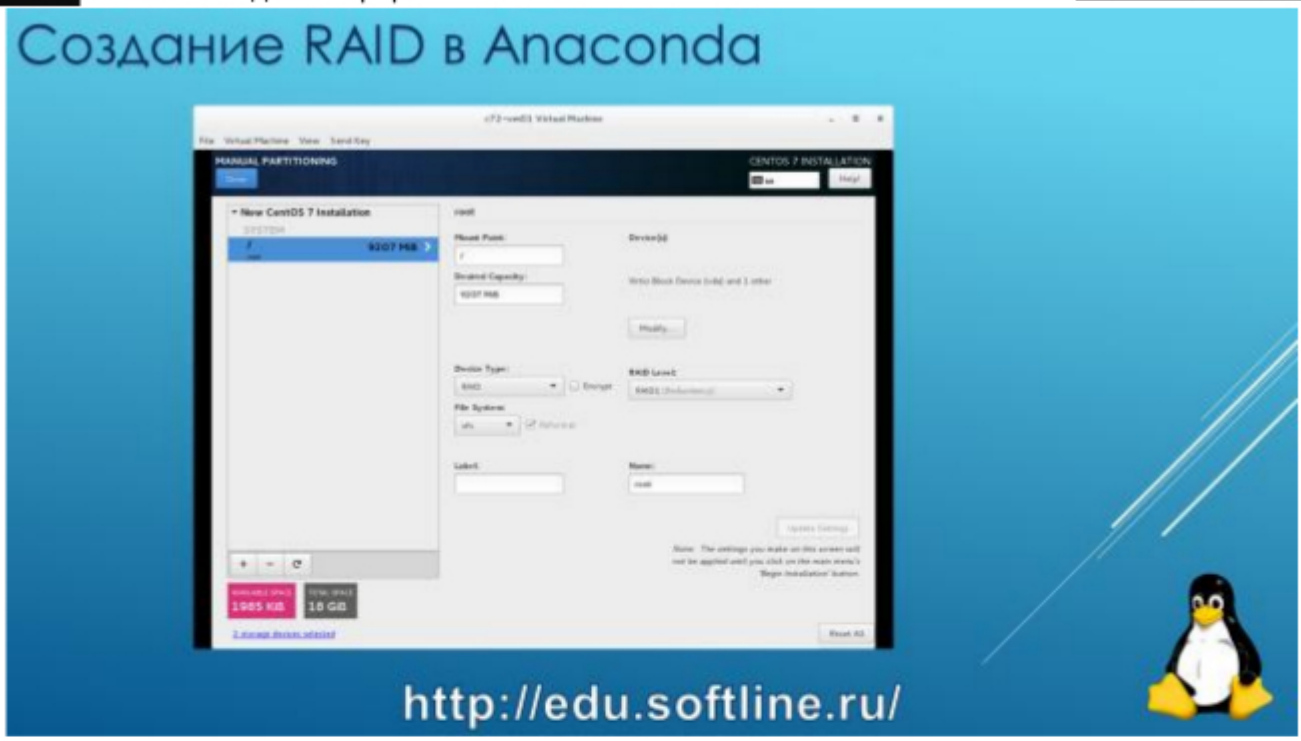
Подсистема **mdraid** разработана как программное RAID-решение для Linux, является предпочтительным программным решением RAID для Linux. эта подсистема использует собственный формат метаданных (**mdraid**).

Также **mdraid** поддерживает другие форматы метаданных, известных как внешние метаданные. CentOS 7 (RHEL7) использует **mdraid** с внешними метаданными для доступа к наборам ISW / IMSM (Intel firmware RAID). Наборы **mdraid** настраиваются и управляются через утилиту **mdadm**.

dmraid.

Device-mapper RAID или **dmraid** принадлежит к коду ядра device-mapper, который предлагает механизм объединения дисков в наборы RAID. Этот код ядра не предоставляет механизмов настройки RAID.

dmraid настраивается в пользовательском пространстве, упрощая поддержку различных метаданных диска. Как таковой **dmraid** используется для широкого разнообразия микропрограммных реализаций RAID.



Поддержка RAID в установщике.

Установщик **Anaconda** автоматически определяет любые аппаратные и встроенные микропрограммные RAID-наборы на системе, делая из доступными для установки. **Anaconda** также поддерживает программные RAID при помощи **`mdraid`** и может распознать существующие наборы **`mdraid`**.

Anaconda предоставляет утилиты для создания наборов RAID в процессе установки. Однако эти утилиты позволяют использовать только разделы в качестве членов нового набора. Чтобы использовать весь диск для RAID, придется создать 1 раздел, содержащий все свободное место диска и объединить его в RAID-набор.

Когда корневая файловая система использует RAID-набор, **Anaconda** добавляет специальную опцию ядра в конфигурацию загрузчика, для указания **`initrd`** какой RAID-набор активировать для поиска корневой файловой системы.



Создание нового RAID устройства.

Для создания нового RAID устройства, выполните следующую команду, от имени пользователя root:

```
# mdadm --create raid_device --level=level --raid-devices=number device…
```

Это самый простой путь для создания RAID массива. В команде присутствует множество опций, которые позволяют указывать число подключаемых устройств, размер блока чередующегося массива и так далее. Все эти опции будут иметь влияние на производительность, для получения детальной информации по ним ознакомьтесь с секцией **CREATE MODE** на странице руководства **mdadm**(8).

Пример создания нового RAID устройства:

Представим, что у нас есть два неиспользованных SCSI-диска и на каждом из них есть раздел одинакового размера.

```
[root@c7-server01 ~]# ls /dev/sd*
/dev/sda  /dev/sda1  /dev/sda2  /dev/sdb  /dev/sdb1  /dev/sdc  /dev/sdc1  /dev/sdd
```

Для создания **/dev/md0**, как нового RAID1 массива из **/dev/sdb1** и **/dev/sdc1**, запустите следующую команду:

```
[root@c7-server01 ~]# mdadm --create /dev/md0 --level=1 --raid-devices=2 /dev/sdb1
/dev/sdc1
mdadm: Note: this array has metadata at the start and
    may not be suitable as a boot device.  If you plan to
    store '/boot' on this device please ensure that
    your boot-loader understands md/v1.x metadata, or use
    --metadata=0.90
Continue creating array? y
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md0 started.
```

Просмотр текущей конфигурации RAID.

Когда используется программный RAID, базовая информация обо всех активных RAID-устройствах записывается в специальном файле **/proc/mdstat**. Для просмотра файла можно воспользоваться командой **cat**:

```
# cat /proc/mdstat
```

Для определения, когда устройство является RAID устройством, а когда компонентным, выполните следующую команду от имени пользователя root:

```
# mdadm --query device…
```

Для получения детальной информации о RAID-устройстве, используйте следующую команду:

```
# mdadm --detail raid_device…
```

Аналогично, проверки компонентного устройства, введите:

```
# mdadm --examine component_device…
```

Команда **mdadm --detail** отображает информацию о RAID-устройстве, **mdadm --examine** только информацию, относящуюся к компонентному устройству. Это различие имеет значение, когда RAID устройство является компонентом другого RAID-устройства.

Команда **mdadm --query**, также, как и команды **mdadm --detail** и **mdadm --examine** позволяют вам указывать несколько устройств за раз.

Пример просмотра конфигурации RAID.

Проверки, что устройство **/dev/md0** является RAID устройством:

```
[root@c7-server01 ~]# mdadm --query /dev/md0
/dev/md0: 99.94GiB raid1 2 devices, 0 spares. Use mdadm --detail for more detail.
```

Как видно из вывода, эта команда предоставляет краткий обзор RAID-устройства и его конфигурации. Для отображения подробной информации воспользуемся следующей командой:

```
[root@c7-server01 ~]# mdadm --detail /dev/md0
/dev/md0:
        Version : 1.2
  Creation Time : Sat Jan  2 01:27:29 2016
     Raid Level : raid1
     Array Size : 104791040 (99.94 GiB 107.31 GB)
  Used Dev Size : 104791040 (99.94 GiB 107.31 GB)
   Raid Devices : 2
  Total Devices : 2
    Persistence : Superblock is persistent

    Update Time : Sat Jan  2 01:36:11 2016
          State : clean
 Active Devices : 2
Working Devices : 2
 Failed Devices : 0
  Spare Devices : 0

           Name : c7-server01.nerv.local:0  (local to host c7-server01.nerv.local)
           UUID : 8119886b:63181d89:9d29e86f:f2ab9565
         Events : 17

    Number   Major   Minor   RaidDevice State
```

```
       0       8       17        0      active sync   /dev/sdb1
       1       8       33        1      active sync   /dev/sdc1
```

Для вывода всех активных RAID устройств, введите:

```
[root@c7-server01 ~]# cat /proc/mdstat
Personalities : [raid1]
md0 : active raid1 sdc1[1] sdb1[0]
      104791040 blocks super 1.2 [2/2] [UU]

unused devices: <none>
```

Замена вышедшего из строя устройства в массиве.

Для замены необходимого устройства в программном RAID, в первую очередь убедитесь, что устройство помечено как неисправное (Faulty), при помощи следующей команды:

```
# mdadm raid_device --fail component_device
```

Затем удалите неисправное устройство:

```
# mdadm raid_device --remove component_device
```

После починки устройства, вы можете повторно добавить его в массив:

```
# mdadm raid_device --add component_device
```

Пример замены неисправного устройства:

У нас в системе присутствует следующее RAID-устройство **/dev/md0**:

```
[root@c7-server01 ~]# mdadm --detail /dev/md0 | tail -n 3
    Number   Major   Minor   RaidDevice State
       0       8       17        0      active sync   /dev/sdb1
       1       8       33        1      active sync   /dev/sdc1
```

Предположим первый диск вышел из строя и должен быть заменен, для этого пометим **/dev/sdc1** как неисправный:

```
[root@c7-server01 ~]# mdadm /dev/md0 --fail /dev/sdc1
mdadm: set /dev/sdc1 faulty in /dev/md0
```

Затем удалим его из RAID-устройства:

```
[root@c7-server01 ~]# mdadm /dev/md0 --remove /dev/sdc1
mdadm: hot removed /dev/sdc1 from /dev/md0
```

После замены оборудования, можно вернуть диск **/dev/sdc1** на место в массив:

```
[root@c7-server01 ~]# mdadm /dev/md0 --add /dev/sdc1
mdadm: added /dev/sdc1
```

Расширение существующего массива.

Для добавления нового устройства в существующий массив, используйте следующую команду, с правами пользователя root:

```
# mdadm raid_device --add component_device
```

Это добавит устройство в качестве запасного устройства. Для расширения массива и активного использования устройства введите следующую команду:

```
# mdadm --grow raid_device --raid-devices=number
```

Пример расширения RAID-устройства.

Предположим в системе имеется активное RAID-устройство, **/dev/md0**:

```
[root@c7-server01 ~]# mdadm --detail /dev/md0 | tail -n 3
    Number   Major   Minor   RaidDevice State
       0       8       17        0      active sync   /dev/sdb1
       2       8       33        1      active sync   /dev/sdc1
```

Также имеется дисковый накопитель SCSI **/dev/sdd** с одним разделом. Для добавления раздела в массив **/dev/md0**, воспользуемся следующей командой:

```
[root@c7-server01 ~]# mdadm /dev/md0 --add /dev/sdd1
mdadm: added /dev/sdd1
[root@c7-server01 ~]# mdadm --detail /dev/md0 | tail -n 5
    Number   Major   Minor   RaidDevice State
       0       8       17        0      active sync   /dev/sdb1
       2       8       33        1      active sync   /dev/sdc1

       3       8       49        -      spare   /dev/sdd1
```

Это добавить запасное устройство, для того чтобы задействовать устройство в массиве, необходимо использовать следующую команду:

```
[root@c7-server01 ~]# mdadm --grow /dev/md0 --raid-devices=3
raid_disks for /dev/md0 set to 3
unfreeze
[root@c7-server01 ~]# mdadm --detail /dev/md0 | tail -n 4
    Number   Major   Minor   RaidDevice State
       0       8       17        0      active sync   /dev/sdb1
       2       8       33        1      active sync   /dev/sdc1
       3       8       49        2      spare rebuilding   /dev/sdd1
```

Удаление устройства RAID.

Для удаления RAID-устройства, в первую очередь необходим его деактивировать:

```
# mdadm --stop raid_device
```

После деактивации можно удалить RAID-устройство:

```
# mdadm --remove raid_device
```

В конце нужно обнулить супеблок всех устройств, которые участвовали в массиве:

```
# mdadm --zero -superblock component_device…
```

Пример удаления RAID-устройства.

Предположим в системе имеется активное RAID-устройство, **/dev/md0**:

```
[root@c7-server01 ~]# mdadm --detail /dev/md0 | tail -n 4
    Number   Major   Minor   RaidDevice State
       0       8       17        0      active sync   /dev/sdb1
       2       8       33        1      active sync   /dev/sdc1
       3       8       49        2      active sync   /dev/sdd1
```

Для удаления устройства, в первую очередь деактивируем его:

```
[root@c7-server01 ~]# mdadm --stop /dev/md0
mdadm: stopped /dev/md0
```

После остановки устройства удалим его (Если вы не делали сохранение конфигурации, этот шаг не потребуется):

```
[root@c7-server01 ~]# mdadm --remove /dev/md0
mdadm: error opening /dev/md0: No such file or directory
```

Для завершения процесса удаления устройства, обнулим суперблок всех компонентных устройств:

```
[root@c7-server01 ~]# mdadm --zero-superblock /dev/sdb1 /dev/sdc1 /dev/sdd1
```

Сохранение конфигурации.

По умолчанию, изменения сделанные при помощи команды **mdadm** применяются только в текущей сессии и не сохраняются после перезапуска системы. В процессе загрузки служба **mdmonitor** читает содержимое конфигурационного файла **/etc/mdadm.conf**, чтобы узнать какие RAID-устройства запускать. Если программный RAID был сконфигурирован в процессе графической установки, этот файл содержит следующие директивы.

Опция	Описание
ARRAY	Позволяет определить конкретный саммив.
DEVICE	Позволяет указать список устройств для сканирования компонентов RAID (например, **/dev/sdb1**). Также поддерживаются ключевые слова **partitions**, для использования всех разделов из файла **/proc/partitions** или **containers**, для указания контейнера массива.
MAILADDR	Позволяет указать почтовый адрес для отправки предупреждений.

Для отображения какие строки **ARRAY** в настоящее время используются вне зависимости от конфигурации, используйте следующую команду от имени пользователя root.

```
# mdadm --detail --scan
```

Используйте вывод команды для определения, какие строки добавить в файл **/etc/mdadm.conf**. Вы также можете отобразить строку **ARRAY** для указанного устройства:

```
# mdadm --detail --brief raid_device
```

При помощи перенаправления вывода вы можете добавить необходимые строки в конфигурационный файл:

```
# mdadm --detail --brief raid_device >> /etc/mdadm.conf
```

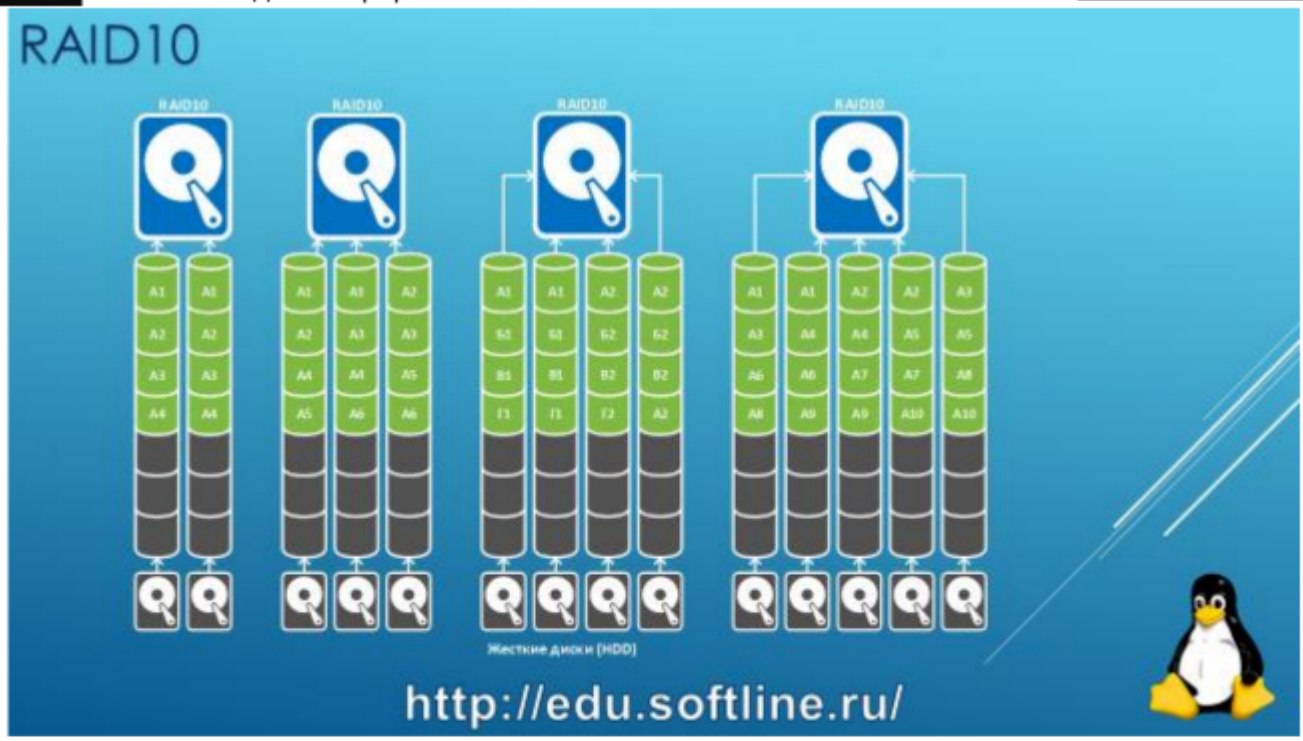
Пример сохранения конфигурации.

По умолчанию **/etc/mdadm.conf** содержит конфигурации программного RAID, созданного в процессе установки системы:

```
[root@c72-vm01 ~]# cat /etc/mdadm.conf
# mdadm.conf written out by anaconda
MAILADDR root
AUTO +imsm +1.x -all
ARRAY /dev/md/root level=raid1 num-devices=2 UUID=99151385:62f0e12f:e289e334:21d4e8c3
```

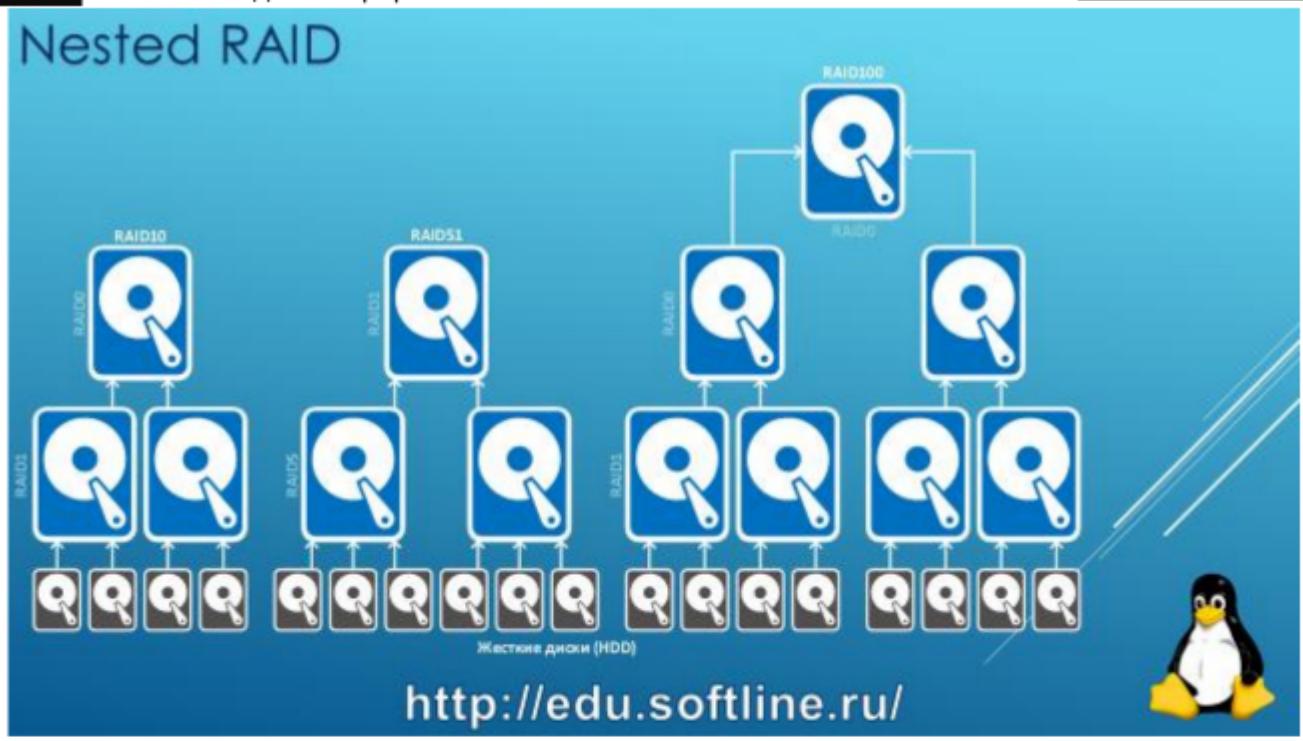
Для добавления конфигурации собственного RAID-устройства **/dev/md0** в конфигурационный файл, воспользуемся командой:

```
[root@c7-server01 ~]# mdadm  --detail --brief /dev/md0 >> /etc/mdadm.conf
[root@c7-server01 ~]# cat /etc/mdadm.conf
ARRAY /dev/md0 metadata=1.2 name=c7-server01.nerv.local:0 UUID=c47d3b98:03a224d0:2c3e7da7:f773999b
```



RAID 10.

Уровень 10 позволяет комбинировать преимущества производительности уровня 0 и избыточности уровня 1. Также этот уровень позволяет сократить накладные расходы объема диска, присущие RAID1. При помощи уровня 10 возможно создать массив из 3 дисков, где данные будут храниться только дважды, что позволит создать массив размером в полтора раза больше самого маленького устройства в массиве.



Nested RAID, также известный, как гибридный RAID (Hybrid RAID) объединяет в себе два или более стандартных уровня RAYD для получения производительности, дополнительной отказоустойчивости или всего сразу. Уровни Nested RAID включают RAID 01, RAID 10, RAID 100, RAID 50 and RAID 60 и многие другие.



Дополнительные ресурсы.

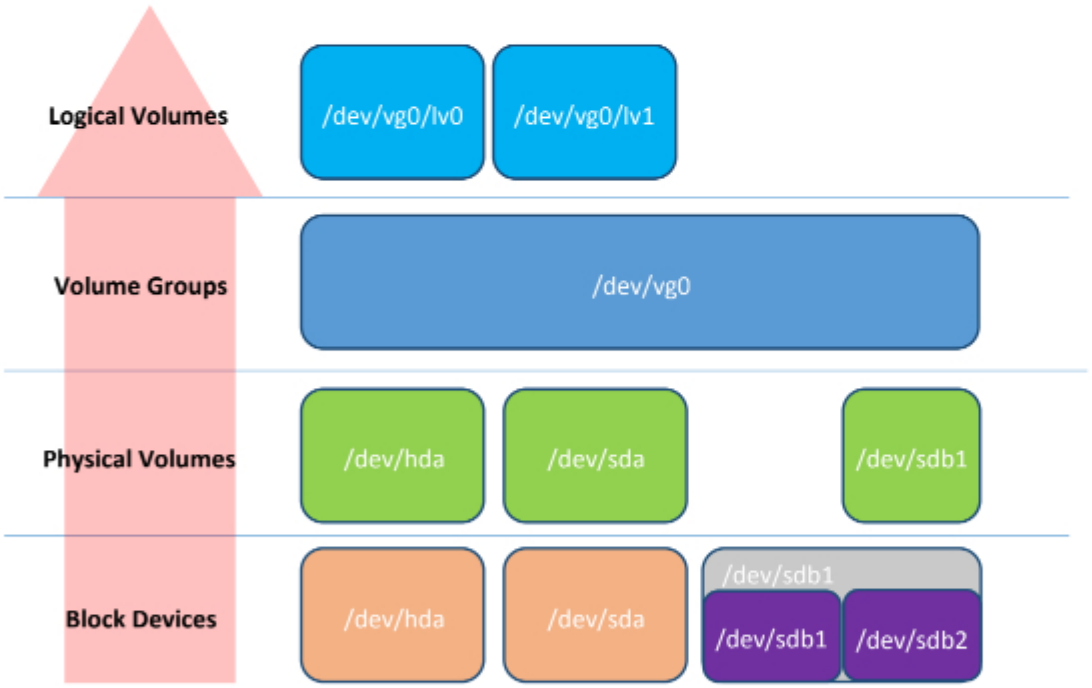
За дополнительной информацией о RAID обратитесь к следующим ресурсам.

Установленная документация:

- Страница руководства **mdadm** – описание утилиты **mdadm**.
- Страница руководства **mdadm.conf** – описание полного списка конфигурационных опций, доступных в **/etc/mdadm.conf**.



LVM (Logical Volume Manager), менеджер логических томов – это система управления дисковым пространством, абстрагирующаяся от физических устройств. Эта система позволяет легко управлять дисковым пространством, добавлять тома, изменять их размер и т.д.



Менеджер логических томов оперирует несколькими основными понятиями:

- **PV** (Physical Volume) – это раздел диска, либо раздел RAID-массива. Физический том входит в состав группы томов;
- **VG** (Volume Group) – группа, объединяющая несколько физических томов, является верхним уровнем абстрактной модели LVM;
- **LV** (Logical Volume) – раздел группы томов. Для более простого понимания, можно представить группу томов в качестве жёсткого диска, а логические тома в качестве разделов этого диска;

- **PE** (Physical Extent) – каждый физический том состоит из блоков определённого размера, называемых экстентами. Количество и размер физических экстентов определяют размер физического тома;
- **LE** (Logical Extent) – каждый логический том состоит из блоков определённого размера, кратного размеру физического экстента.

Процедура создания логических томов состоит из создания физических томов и их объединения в группы. Чтобы получить информацию о созданных физических томах, воспользуйтесь командой **pvscan**, команды **vgscan** и **lvscan** выводят информацию о группах томов и логических томах, соответственно.

Чтобы создать на диске/разделе диска структуру физического тома, воспользуйтесь командой **pvcreate**, в случае, когда вы работаете с разделом, сначала установите тип раздела в значение *0x8e* (командой `fdisk`):

```
# pvcreate /dev/sdb1
Physical volume "/dev/sdb1" successfully created
# pvcreate /dev/sdb2
Physical volume "/dev/sdb2" successfully created
```

Теперь, когда есть два физических тома, их необходимо объединить в группу с помощью команды **vgcreate**:

```
# vgcreate vg00 /dev/sdb1 /dev/sdb2
Volume group "vg00" successfully created
```

Теперь перейдём к созданию логических томов, объектов, с которыми и будет проводиться дальнейшая работа операционной системы. Воспользуемся командой **lvcreate**:

```
# lvcreate -L1000 -nlv1 vg00
Logical volume "lv1" created
```

Следующим шагом является активация группы томов командой `vgchange`:

```
# vgchange -ay
```

На этом этапе мы уже имеем один логический том и можем получить о нём информацию командой `lvdisplay`. Но этот том не приспособлен для работы, так как не имеет никакой файловой системы, давайте создадим её:

```
# mkfs -t ext3 /dev/vg00/lv1
```

Случаются ситуации, когда свободное место логического тома заканчивается и его необходимо расширить, для этой операции существует команда `lvextend`:

```
# lvextend -L+500M /dev/vg00/lv1
Logical volume lv1 successfully resized
```

Но этой командой мы увеличили только размер тома, но не размер файловой системы, для изменения размера файловой системы воспользуемся командой `resize2fs`, перед эти выполнив проверку файловой системы:

```
# e2fsck -f /dev/vg00/lv1

# resize2fs /dev/vg00/lv1
```

Для уменьшения размера тома предназначена команда **lvreduce**.



Пространство подкачки (Swap Space) в Linux используется, когда весь объем физической памяти (RAM) использован. Если системе требуется дополнительная память, а физическая память (RAM) уже исчерпана, неактивные страницы перемещаются в область подкачки. Несмотря на то что подкачка может помочь машинам с небольшим объемом RAM, она не является заменой физической памяти. Пространство подкачки располагается на жестких дисках, которые обладают более медленной скоростью доступа чем физическая память. Пространство подкачки может быть выделено в раздел подкачки (рекомендуется), файл подкачки мом комбинацию из разделов и файлов подкачки.

ПРИМЕЧАНИЕ:
Btrfs не поддерживает пространство подкачки.

Последние годы рекомендованный объем пространства подкачки увеличивался линейно, вместе с объемом RAM в системе. Однако, современные системы часто содержат сотни гигабайтов RAM. В результате рекомендованное пространство подкачки рассматривается как функция нагрузки на системную память, а не как системная память.

В таблице ниже предоставлены рекомендованные размеры разделов подкачки в зависимости от объема RAM в системе и возможности использования гибернации на системе. Рекомендованный размер раздела подкачки вычисляется автоматически в процессе установки. Однако, чтобы разрешить гибернацию на системе, необходимо вручную отредактировать размер раздела подкачки на этапе разметки диска.

Рекомендации в таблице ниже особенно важны для систем с с объемом памяти 1 Gb и меньше. Некорректное выделение пространства подкачки на системе, может стать причиной нестабильной работы или невозможности запуска установленной системы.

Рекомендованные пространства подкачки систем:

Объем RAM в системе	Рекомендованная область подкачки	Рекомендованная область подкачки для использования гибернации.
≤ 2 Gb	Двойной объем RAM.	Тройной объем RAM.
> 2 Gb – 8Gb	Равная объему RAM.	Двойной объем RAM.
> 8 Gb – 64 Gb	Как минимум 4 Gb.	1,5 объема RAM.
> 64 Gb	Как минимум 4 Gb.	Гибернация не рекомендуется.

На границе между каждым диапазоном, указанным в таблице выше, основной выбор заключается в использовании или не использовании режима гибернации на системе. Если системные ресурсы позволяют, то увеличение объема пространства подкачки, может увеличить производительность системы. Область подкачки, как минимум в 100 Gb рекомендуется для систем со 140 и более логическими процессорами или 3 и более Tb физической памяти.

Распределение области подкачки по нескольким устройствам хранения также увеличивают производительность пространства подкачки, особенно на системах с быстрыми дисками, котроллерами и интерфейсами.



ПРИМЕЧАНИЕ:

Файловые системы и тома LVM2 назначенные пространству подкачки не должны использоваться в процессе модификации. Любая попытка изменить подкачку приведет к ошибке процесса или ядра, использующую пространство подкачки. Используйте команду **`free`** *и* **`cat /proc/swaps`** *для проверки того, сколько и где подкачки используется.*

Вы должны изменять область подкачки, когда система загружена в аварийном режиме (Rescue Mode).



Добавление пространства подкачки (SWAP).

В некоторых случаях необходимо увеличить объем пространства подкачки после установки системы. Например, в случае увеличения объема RAM на вашей системе с 1 Gb до 2 Gb, при этом объем подкачки сохраняется в размере 2 Gb. Вы можете получить преимущества от увеличения области подкачки до 4 Gb на системах с активным использованием памяти или приложениями, требующими большего объема памяти.

У вас есть три опции: создать новый раздел подкачки, создать файл подкачки или увеличить область подкачки на существующем логическом томе LVM2. В случае наличия последнего рекомендуется увеличивать логический том.

Увеличение подкачки на логическом томе LVM2.

По умолчанию CentOS 7 (RHEL 7) использует все доступное пространство в процессе установки. В таком случае вам необходимо добавить дополнительный физический том (Physical Volume) в группу томов (Volume Group) на которой размещается область подкачки. После добавления дополнительного хранилища в группу томов, можно расширять область подкачки.

Процедура расширения подкачки на логическом томе LVM2:

1. Отключение подкачки на логическом томе:

```
# swapoff -v /dev/VolGroup00/LogVol01
```

2. Увеличение размера логического тома на 2 Gb:

```
# lvresize /dev/VolGroup00/LogVol01 -L +2G
```

3. Форматирование нового пространства подкачки:

```
# mkswap /dev/VolGroup00/LogVol01
```

4. Включение подкачки на увеличенном логическом томе:

```
# swapon -v /dev/VolGroup00/LogVol01
```

Чтобы проверить, что логический том и область подкачки увеличились используйте команды **free** или **cat /proc/swaps**.

Создание логического тома LVM2 для подкачки.

Для добавления логического тома подкачки в группе томов, используйте следующую процедуру:

1. Создание логического тома, размером 2

```
# lvcreate VolGroup00 -n LogVol02 -L 2G
```

2. Форматирование нового пространства подкачки:

```
# mkswap /dev/VolGroup00/LogVol02
```

3. Добавление новой записи в **/etc/fstab**:

```
/dev/VolGroup00/LogVol02   swap   swap   defaults   0 0
```

4. Включение подкачки на новом логическом томе LVM2:

```
# swapon -v /dev/VolGroup00/LogVol02
```

Чтобы проверить, что логический том и область подкачки увеличились используйте команды **free** или **cat /proc/swaps**.

Создание файла подкачки.

Для добавления файла подкачки, используйте следующую процедуру:

1. Определите необходимый размер файла подкачки в мегабайтах и умножьте на 1024 для определения необходимого числа блоков. Например, для файла подкачки, размером 64 Mb, потребуется 65536 блоков по 1 Kb каждый.
2. При помощи команды **dd** создайте необходимый файл:

```
# dd if=/dev/zero of=/swapfile bs=1024 count=65536
```

3. Отформатируйте файл подкачки:

```
# mkswap /swapfile
```

4. Настройте безопасность файла подкачки:

```
# chmod 0600 /swapfile
```

5. Включите использование файла подкачки:

```
# swapon /swapfile
```

6. Настройте автоматический запуск файла подкачки в процессе загрузки системы:

```
/swapfile   swap   swap   defaults   0 0
```

Чтобы проверить, что файл подкачки используется используйте команды **free** или **cat /proc/swaps**.

Удаление пространства подкачки (SWAP).

Иногда бывает необходимо сократить область подкачки после установки. Например, если вы уменьшили объем RAM на вашей системе с 1 Gb до 512 Mb, в таком случае 2 GB области подкачки остается назначено системе. Вы можете получить преимущества от сокращения области подкачки до 1 Gb на системе, вместо простаивающего пространства подкачки размером в 2 Gb.

У вас есть три опции: удалить весь логический том LVM, используемый для подкачки, удалить файл подкачки или уменьшить область подкачки на существующем логическом томе LVM2.

Уменьшение подкачки на логическом томе LVM2.

Для уменьшения логического тома LVM2, используемого для подкачки, воспользуйтесь следующей процедурой:

1. Отключите подкачку на указанном логическом томе LVM2:

```
# swapoff -v /dev/VolGroup00/LogVol01
```

2. Уменьшите логический том на 1 Gb:

```
# lvreduce /dev/VolGroup00/LogVol01 -L -512M
```

3. Отформатируйте новое пространство подкачки:

```
# mkswap /dev/VolGroup00/LogVol01
```

4. Включите подкачку на уменьшенном логическом томе LVM2:

```
# swapon -v /dev/VolGroup00/LogVol01
```

Чтобы проверить, что логический том и область подкачки уменьшились используйте команды **free** или **cat /proc/swaps**.

Удаление логического тома LVM2, используемого для подкачки.

Для удаления логического тома LVM2, используемого для подкачки, воспользуйтесь следующей процедурой:

1. Отключите подкачку на связанном логическом томе:

```
# swapoff -v /dev/VolGroup00/LOgVol02
```

2. Удалите логический том LVM2:

```
# lvremove /dev/VolGroup00/LogVol02
```

3. Удалите или закоментируйте запись из файла **/etc/fstab**:

```
/dev/VolGroup00/LogVol02   swap   swap   defaults   0 0
```

Чтобы проверить, что логический том удален и область подкачки уменьшилась используйте команды **free** или **cat /proc/swaps**.

Удаление файла подкачки.

Для удаления файла подкачки, воспользуйтесь следующей процедурой:

1. Отключите использование файла для подкачки:

```
# swapoff -v /swapfile
```

2. Удалите строку из файла **/etc/fstab**:

```
/swapfile   swap   swap   defaults   0 0
```

3. Удалите файл подкачки:

```
# rm /swapfile
```

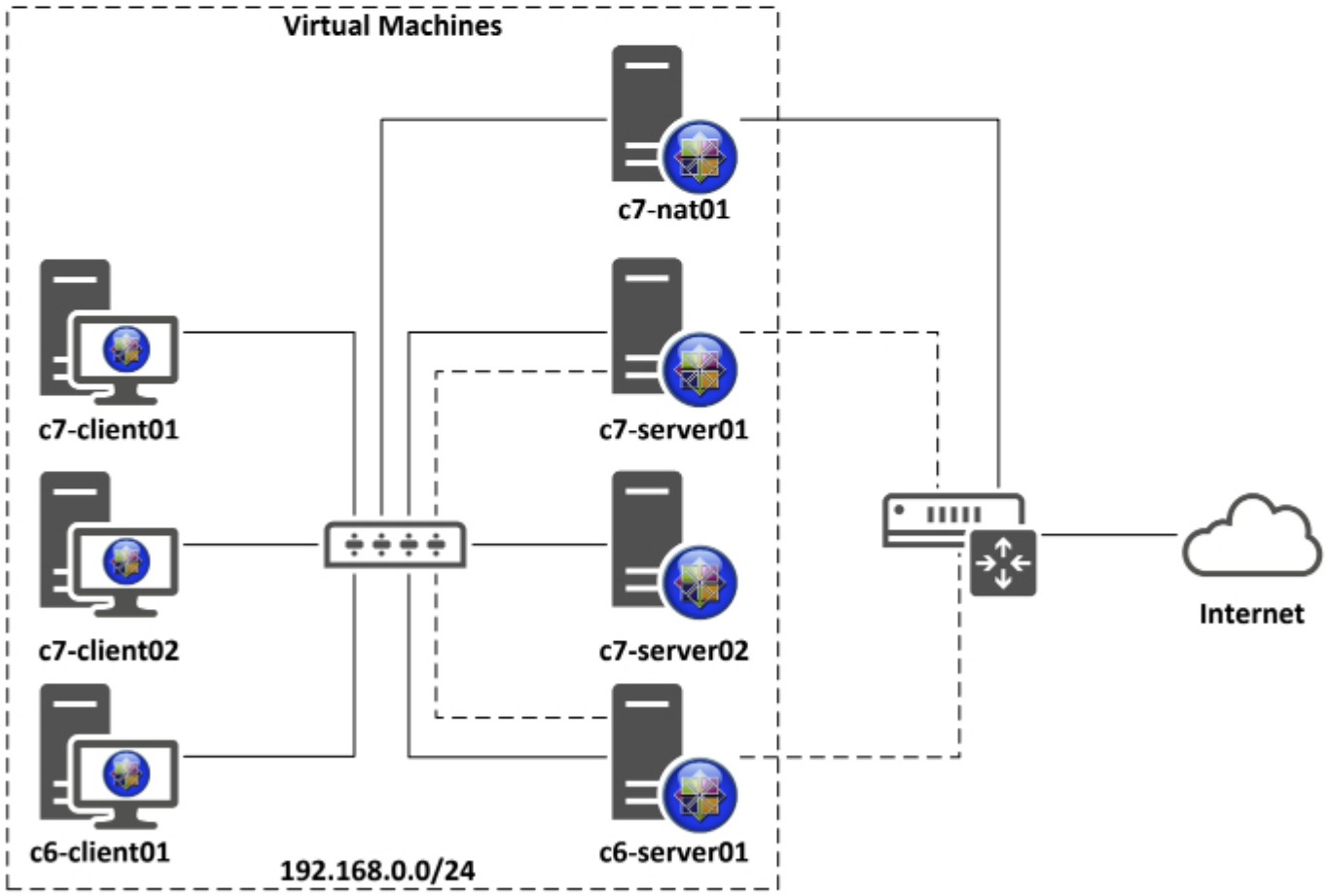
Перемещение пространства подкачки (SWAP).

Для перемещения пространства подкачки используйте шаги по добавлению нового пространства подкачки и удалению старого.

Лабораторная работа 4

- **Упражнения:**
  - Создание и управление разделами и файловыми системами.
  - Управление файловой системой XFS.
  - Управление областью подкачки.
  - Управление программным RAID.
  - Управление LVM.
- **Необходимые машины:**
  - `c7-nat01`.
- **Учетные данные:**
  - Пользователь: `root`.
  - Пароль: **`Pa$$w0rd`**.

http://edu.softline.ru/





Упражнение 1. Создание и управление разделами и файловыми системами.

1. Запустите виртуальную машину **c7-server01**.
2. При помощи команды **`fdisk`** просмотрите конфигурацию диска **`/dev/sdb`**.
3. На диске **`/dev/sdb1`** основной раздел размером 2 Gb.
4. Создайте на разделе **`/dev/sdb1`** файловую систему **`ext2`**.
5. Создайте каталог **`/mnt/ext`** для монтирования файловой системы.
6. Смонтируйте раздел с файловой системой ext2 в каталог **`/mnt/ext`**.
7. Скопируйте каталог **`/etc`** в **`/mnt/ext`**.
8. Отмонтируйте раздел **`/dev/sdb1`**.
9. Создайте метку тома с именем **`ext_fs`**.
10. Сделайте так, чтобы новый раздел подключался при старте системы в каталог **`/mnt/ext2`** по метке тома.
11. Перезагрузите систему и убедитесь, что файловая система смонтирована корректно.
12. Конвертируйте файловую систему ext2 в файловую систему ext4.
13. После конвертации необходимо проверить файловую систему на наличие ошибок.
14. Замените в конфигурационном файле **`/etc/fstab`** файловую систему у раздела **`/dev/sdb1`** на **`ext4`**.
15. Смонтируйте все файловые системы, определенные в файле **`/etc/fstab`**.
16. Убедитесь, что файловая система смонтирована корректно и скопированный каталог **`etc`** остался на разделе.



Ответы к упражнению 1: Создание и управление разделами и файловыми системами.

1. Запустите виртуальную машину **c7-server01**.
2. При помощи команды **fdisk** просмотрите конфигурацию диска **/dev/sdb**.

```
# fdisk -l /dev/sdb
```

3. На диске **/dev/sdb1** основной раздел размером 1 Gb.

```
# fdisk /dev/sdb
Command (m for help): p
Command (m for help): n
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-20971519, default 2048): 2048
Last sector, +sectors or +size{K,M,G} (2048-20971519, default 20971519): +2G
Command (m for help): p
Command (m for help): w
```

4. Создайте на разделе **/dev/sdb1** файловую систему ext2.

```
# mkfs.ext2 /dev/sdb1
```

5. Создайте каталог **/mnt/ext** для монтирования файловой системы.

```
# mkdir /mnt/ext
```

6. Смонтируйте раздел с файловой системой ext2 в каталог **/mnt/ext**.

```
# mount /dev/sdb1 /mnt/ext/
```

7. Скопируйте каталог **/etc** в **/mnt/ext**.

```
# cp -R /etc /mnt/ext/
```

8. Отмонтируйте раздел **/dev/sdb1**.

```
# umount /dev/sdb1
```

9. Создайте метку тома с именем **ext_fs**.

```
# e2label /dev/sdb1 ext_fs
```

10. Сделайте так, чтобы новый раздел подключался при старте системы в каталог **/mnt/ext** по метке тома.

```
# vi /etc/fstab
…
LABEL=ext_fs    /mnt/ext                ext2    defaults        0 0
```

11. Перезагрузите систему и убедитесь, что файловая система смонтирована корректно.

```
# reboot
...
# df -h
```

12. Конвертируйте файловую систему ext2 в файловую систему ext4.

```
# tune2fs -O extents,uninit_bg,dir_index,has_journal /dev/sdb1
```

13. После конвертации необходимо проверить файловую систему на наличие ошибок.

```
# e2fsck -pf /dev/sdb1
```

14. Замените в конфигурационном файле **/etc/fstab** файловую систему у раздела **/dev/sdb1** на **ext4**.

```
# vi /etc/fstab
...
LABEL=ext_fs    /mnt/ext                ext4    defaults        0 0
```

15. Смонтируйте все файловые системы, определенные в файле **/etc/fstab**.

```
# mount -a
```

16. Убедитесь, что файловая система смонтирована корректно и скопированный каталог **etc** остался на разделе.

```
# df -h
# ls /mnt/ext/
```



Упражнение 2: Управление файловой системой XFS.

1. Запустите виртуальную машину **c7-server01**.
2. С помощью программы **`fdisk`** создайте новый раздел на диске **`/dev/sdc`** размером 5 Gb.
3. Отформатируйте вновь созданный раздел в файловую систему XFS.
4. Смонтируйте вновь созданный раздел в каталог **`/mnt/XFS`**
5. Получите информацию о смонтированной в каталоге **`/mnt/XFS`** файловой системе XFS.
6. С помощью программы **`parted`** увеличьте размер раздела **`/dev/sdc1`** до 7518 Mb.
7. При помощи **`lsblk`** убедитесь, что раздел увеличен.
8. Увеличьте файловую систему XFS на устройстве **`/dev/sdc1`** до максимально доступного размера.
9. Получите информацию о смонтированной в каталоге **`/mnt/XFS`** файловой системе XFS.



Ответы к упражнению 2: Управление XFS.

1. Запустите виртуальную машину **c7-server01**.
2. С помощью программы **fdisk** создайте новый раздел на диске **/dev/sdc** размером 5 Gb.

```
# fdisk /dev/sdc.
```

3. Отформатируйте вновь созданный раздел в файловую систему XFS.

```
# mkfs.xfs /dev/sdc1
```

4. Смонтируйте вновь созданный раздел в каталог **/mnt/XFS**

```
# mkdir /mnt/XFS && mount /dev/logic/vol4 /mnt/XFS
```

5. Получите информацию о смонтированной в каталоге **/mnt/XFS** файловой системе XFS.

```
# xfs_info /mnt/XFS/
```

6. С помощью программы **parted** увеличьте размер раздела до 7518 Mb.

```
(parted) select /dev/sdc
(parted) print
(parted) resizepart
Partition number? 1
End?  [5370MB]? 7518
```

7. При помощи **lsblk** убедитесь, что раздел увеличен.

```
# lsblk
```

8. Увеличьте файловую систему XFS на устройстве **/dev/sdc1** до максимально доступного размера.

```
# xfs_growfs /mnt/XFS
```

9. Получите информацию о смонтированной в каталоге **/mnt/XFS** файловой системе XFS.

```
# df -h
```



Упражнение 3. Управление областью подкачки.

1. Запустите виртуальную машину **c7-server02**.
2. Создайте файл подкачки (**`/swapfile`**) размером 4 Gb.
3. Отформатируйте файл подкачки:
4. Настройте безопасность файла подкачки (0600):
5. Включите использование файла подкачки:
6. Убедитесь, что область подкачки, доступная системе увеличилась.
7. Настройте автоматический запуск файла подкачки в процессе загрузки системы и отключите автоматический запуск раздела подкачки при старте системы:
8. Перезагрузите систему и убедитесь, что область подкачки, доступная системе 3.9 Gb.



Ответы к упражнению 3. Управление областью подкачки.

1. Запустите виртуальную машину **c7-server02**.
2. Создайте файл подкачки (**/swapfile**) размером 4 Gb.

```
# dd if=/dev/zero of=/swapfile bs=1024 count=4096000
```

3. Отформатируйте файл подкачки:

```
# mkswap /swapfile
```

4. Настройте безопасность файла подкачки (0600):

```
# chmod 0600 /swapfile
```

5. Включите использование файла подкачки:

```
# swapon /swapfile
```

6. Убедитесь, что область подкачки, доступная системе увеличилась.

```
# free -h
```

7. Настройте автоматический запуск файла подкачки в процессе загрузки системы и отключите автоматический запуск раздела подкачки при старте системы:

```
# vi /etc/fstab
…
#/dev/mapper/centos-swap swap    swap    defaults    0 0
/swapfile    swap    swap    defaults    0 0
```

8. Перезагрузите систему и убедитесь, что область подкачки, доступная системе 3.9 Gb.

```
# free -h
```



Упражнение 4 – Управление программным RAID.

1. Запустите виртуальную машину **c7-server01**.
2. Создайте RAID-массив **/dev/md1** с уровнем RAID0 (Stripe) из двух дисков **/dev/sdd** и **/dev/sde**.
3. Создайте RAID-массив **/dev/md2** с уровнем RAID1 (Mirror) из двух дисков **/dev/sdf** и **/dev/sdg**.
4. Дождитесь окончания синхронизации накопителей.
5. Отформатируйте RAID-массив **/dev/md0** в файловую систему ext4.
6. Отформатируйте RAID-массив **/dev/md0** в файловую систему XFS.
7. Создайте два каталога **/mnt/raid0** и **/mnt/raid1**.
8. Смонтируйте **/dev/md0** в каталог **/mnt/raid0**, а **/dev/md1** в каталог **/mnt/raid1**.
9. Убедитесь, что RAID-массивы успешно смонтированы.
10. Симулируйте повреждение устройства **/dev/sdf** в массиве **/dev/md1**.
11. Убедитесь, что RAID-массив поврежден.
12. Удалите поврежденное устройство из массива.
13. Убедитесь, что устройство удалено из массива.
14. Добавьте в поврежденный массив новое устройство **/dev/sdh**.
15. Дождитесь окончания синхронизации накопителей.
16. Убедитесь, что массив восстановлен и остался смонтирован и доступен.



Ответы к упражнению 4 – Управление программным RAID.

1. Запустите виртуальную машину **c7-server01**.
2. Создайте RAID-массив **/dev/md1** с уровнем RAID0 (Stripe) из двух дисков **/dev/sdd** и **/dev/sde**.

```
# mdadm -C /dev/md0 -l 0 -n 2 /dev/sdd /dev/sde
```

3. Создайте RAID-массив **/dev/md2** с уровнем RAID1 (Mirror) из двух дисков **/dev/sdf** и **/dev/sdg**.

```
# mdadm -C /dev/md1 -l 1 -n 2 /dev/sdf /dev/sdg
```

4. Дождитесь окончания синхронизации накопителей.

```
# cat /proc/mdstat
```

5. Отформатируйте RAID-массив **/dev/md0** в файловую систему ext4.

```
# mkfs.ext4 /dev/md0
```

6. Отформатируйте RAID-массив **/dev/md0** в файловую систему XFS.

```
# mkfs.xfs /dev/md1
```

7. Создайте два каталога **/mnt/raid0** и **/mnt/raid1**.

```
# mkdir /mnt/raid{0,1}
```

8. Смонтируйте **/dev/md0** в каталог **/mnt/raid0**, а **/dev/md1** в каталог **/mnt/raid1**.

```
# mount /dev/md0 /mnt/raid0/
# mount /dev/md1 /mnt/raid1/
```

9. Убедитесь, что RAID-массивы успешно смонтированы.

```
# df -h
Filesystem               Size  Used Avail Use% Mounted on
…
/dev/md0                  20G   45M   19G   1% /mnt/raid0
/dev/md1                  10G   33M   10G   1% /mnt/raid1
```

10. Симулируйте повреждение устройства **/dev/sdf** в массиве **/dev/md1**.

```
# mdadm /dev/md1 -f /dev/sdf
```

11. Убедитесь, что RAID-массив поврежден.

```
# mdadm -D /dev/md1
```

12. Удалите поврежденное устройство из массива.

```
# mdadm /dev/md1 -r /dev/sdf
```

13. Убедитесь, что устройство удалено из массива.

```
# mdadm -D /dev/md1
```

14. Добавьте в поврежденный массив новое устройство **/dev/sdh**.

```
# mdadm /dev/md1 -a /dev/sdh
```

15. Дождитесь окончания синхронизации накопителей.

```
# cat /proc/mdstat
```

16. Убедитесь, что массив восстановлен и остался смонтирован и доступен.

```
# mdadm -D /dev/md1
# df -h
```



Упражнение 5 – Управление LVM.

Задание 1: Настройка гибкого дискового хранилища при помощи LVM.

1. Запустите виртуальную машину **c7-server01**.
2. Просмотрите текущие физические тома (PV), группы томов (VG) и логические тома (LV).

ПРИМЕЧАНИЕ:

На этом этапе вы должны получить следующие результаты:

```
# pvs
  PV         VG     Fmt  Attr PSize   PFree
  /dev/sda2  centos lvm2 a--  <19.00g    0
# vgs
  VG     #PV #LV #SN Attr   VSize   VFree
  centos   1   2   0 wz--n- <19.00g    0
# lvs
  LV   VG     Attr       LSize   Pool Origin Data%
  root centos -wi-ao---- <17.00g
  swap centos -wi-ao----   2.00g
```

3. Просмотрите текущую конфигурацию дисков **/dev/sdi**, **/dev/sdj** и **/dev/sdk**.
4. Ссоздайте на дисках **/dev/sdi**, **/dev/sdj** и **/dev/sdk** один основной раздел максимального размера, измените тип раздел на **Linux LVM** (**8e**) и запишите изменения на диск.

ПРИМЕЧАНИЕ:

Результат предыдущего шага должен выглядеть следующим образом:

```
# fdisk -l /dev/sdi

Disk /dev/sdi: 10.7 GB, 10737418240 bytes, 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disk label type: dos
Disk identifier: 0x034d76ea

   Device Boot      Start         End      Blocks   Id  System
/dev/sdi1            2048    20971519    10484736   8e  Linux LVM
```

5. Убедитесь, что разделы успешно созданы.
6. Из разделов **/dev/sdi1**, **/dev/sdj1** и **/dev/sdk1** создайте физические тома (PV).
7. Убедитесь, что физические тома (PV) успешно созданы.
8. Создайте группу томов (VG) **lvm2_vg01** из физических томов (PV) **/dev/sdi1**, **/dev/sdj1** и **/dev/sdk1**.
9. Убедитесь, что группа томов (VG) успешно создана:

ПРИМЕЧАНИЕ:

Результат предыдущего шага должен выглядеть следующим образом:

```
# vgdisplay lvm2_vg01
  --- Volume group ---
  VG Name               lvm2_vg01
  System ID
  Format                lvm2
  Metadata Areas        3
  Metadata Sequence No  1
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                0
  Open LV               0
  Max PV                0
  Cur PV                3
  Act PV                3
  VG Size               <29.99 GiB
  PE Size               4.00 MiB
  Total PE              7677
  Alloc PE / Size       0 / 0
  Free  PE / Size       7677 / <29.99 GiB
  VG UUID               4sFUaC-fCVP-vxuO-7ox0-UcdU-qF8P-LL1sFx
```

10. Создайте 3 логических тома (LV):
    a. Том **lvm2_vg01_lv01** с размером 5 Gb.
    b. Том **lvm2_vg01_lv02** с размером 8 Gb.
    c. Том **lvm2_vg01_lv03** с размером 10 Gb.
11. Убедитесь, что логические тома (LV) успешно созданы.
12. На логическом томе (LV) **lvm2_vg01_lv01** создайте файловую систему XFS, а на томах **lvm2_vg01_lv01** и **lvm2_vg01_lv01** файловую систему ext4.
13. Создайте каталоги **/mnt/lv01**, **/mnt/lv02** и **/mnt/lv03**.
14. Добавьте в конфигурационный файл **/etc/fstab** записи, для автоматического монтирования логических томов (LV) в соответствующие каталоги:
    a. Том **lvm2_vg01_lv01** в каталог **/mnt/lv01**.
    b. Том **lvm2_vg01_lv02** в каталог **/mnt/lv02**.
    c. Том **lvm2_vg01_lv03** в каталог **/mnt/lv03**.
15. Смонтируйте все определенные в файле **/etc/fstab** файловые системы.
16. Убедитесь, что логические тома (LV) успешно смонтированы.

Задание 2: Изменение размеров LVM.

1. Просмотрите текущие физические тома (PV), группы томов (VG) и логические тома (LV).

ПРИМЕЧАНИЕ:

На этом этапе вы должны получить следующие результаты:

```
# pvs
  PV         VG         Fmt  Attr PSize   PFree
  /dev/sda2  centos     lvm2 a--  <19.00g     0
  /dev/sdi1  lvm2_vg01  lvm2 a--  <10.00g  4.99g
  /dev/sdj1  lvm2_vg01  lvm2 a--  <10.00g <2.00g
  /dev/sdk1  lvm2_vg01  lvm2 a--  <10.00g     0
# vgs
  VG         #PV #LV #SN Attr   VSize   VFree
  centos       1   2   0 wz--n- <19.00g     0
  lvm2_vg01    3   3   0 wz--n- <29.99g <6.99g
# lvs
  LV              VG         Attr       LSize   Pool Origin Data%
  root            centos     -wi-ao---- <17.00g
  swap            centos     -wi-ao----   2.00g
  lvm2_vg01_lv01 lvm2_vg01 -wi-ao----   5.00g
  lvm2_vg01_lv02 lvm2_vg01 -wi-ao----   8.00g
  lvm2_vg01_lv03 lvm2_vg01 -wi-ao----  10.00g
```

2. Добавьте по 2 Gb к логическим томам (LV) **lvm2_vg01_lv01** и **lvm2_vg01_lv02**.
3. Убедитесь, что логические тома (LV) успешно увеличены.
4. Увеличьте файловые системы на логических томах (LV) **lvm2_vg01_lv01** и **lvm2_vg01_lv02**.
5. Убедитесь, что файловые системы успешно увеличены.
6. Отмонтируйте логический том (LV) **lvm2_vg01_lv03**.
7. Проверьте логический том логический том (LV) **lvm2_vg01_lv03** на наличие ошибок.
8. Уменьшите файловую систему на логическом томе (LV) **lvm2_vg01_lv03** до 5 Gb.
9. Уменьшите логическом том (LV) **lvm2_vg01_lv03** на 5 Gb.
10. При помощи **resize2fs** убедитесь, что файловая система не была нарушена во время уменьшения логического тома (LV).
11. Убедитесь, что логический тома (LV) успешно уменьшен.
12. Смонтируйте логический том **lvm2_vg01_lv03**.
13. Убедитесь, что файловая система успешно уменьшена.



Ответы к упражнению Упражнение 5 – Управление LVM.

Задание 1: Настройка гибкого дискового хранилища при помощи LVM.

1. Запустите виртуальную машину **c7-server01**.
2. Просмотрите текущие физические тома (PV), группы томов (VG) и логические тома (LV).

```
# pvs
# vgs
# lvs
```



ПРИМЕЧАНИЕ:

На этом этапе вы должны получить следующие результаты:

```
# pvs
  PV         VG     Fmt  Attr PSize   PFree
  /dev/sda2  centos lvm2 a--  <19.00g    0
# vgs
  VG     #PV #LV #SN Attr   VSize   VFree
  centos   1   2   0 wz--n- <19.00g    0
# lvs
  LV   VG     Attr       LSize   Pool Origin Data%
  root centos -wi-ao---- <17.00g
  swap centos -wi-ao----   2.00g
```

3. Просмотрите текущую конфигурацию дисков **/dev/sdi**, **/dev/sdj** и **/dev/sdk**.

```
# fdisk -l /dev/sd{i,j,k}
```

4. Ссоздайте на дисках **/dev/sdi**, **/dev/sdj** и **/dev/sdk** один основной раздел максимального размера, измените тип раздел на **Linux LVM** (**8e**) и запишите изменения на диск.

```
# fdisk /dev/sdi
Command (m for help): n
Select (default p):
Partition number (1-4, default 1):
First sector (2048-20971519, default 2048):
Last sector, +sectors or +size{K,M,G} (2048-20971519, default 20971519):
Command (m for help): t
Hex code (type L to list all codes): 8e
Command (m for help): w
```

ПРИМЕЧАНИЕ:

Результат предыдущего шага должен выглядеть следующим образом:

```
# fdisk -l /dev/sdi

Disk /dev/sdi: 10.7 GB, 10737418240 bytes, 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disk label type: dos
Disk identifier: 0x034d76ea

   Device Boot      Start         End      Blocks   Id  System
/dev/sdi1            2048    20971519    10484736   8e  Linux LVM
```

5. Убедитесь, что разделы успешно созданы.

```
# lsblk /dev/sd{i,j,k}
```

6. Из разделов **/dev/sdi1**, **/dev/sdj1** и **/dev/sdk1** создайте физические тома (PV).

```
# pvcreate /dev/sdi1 /dev/sdj1 /dev/sdk1
```

7. Убедитесь, что физические тома (PV) успешно созданы.

```
# pvs
```

8. Создайте группу томов (VG) **lvm2_vg01** из физических томов (PV) **/dev/sdi1**, **/dev/sdj1** и **/dev/sdk1**.

```
# vgcreate lvm2_vg01 /dev/sdi1 /dev/sdj1 /dev/sdk1
```

9. Убедитесь, что группа томов (VG) успешно создана:

```
# vgs
```

ПРИМЕЧАНИЕ:

Результат предыдущего шага должен выглядеть следующим образом:

```
# vgdisplay lvm2_vg01
  --- Volume group ---
  VG Name               lvm2_vg01
  System ID
  Format                lvm2
  Metadata Areas        3
  Metadata Sequence No  1
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                0
  Open LV               0
  Max PV                0
  Cur PV                3
  Act PV                3
  VG Size               <29.99 GiB
  PE Size               4.00 MiB
  Total PE              7677
  Alloc PE / Size       0 / 0
  Free  PE / Size       7677 / <29.99 GiB
  VG UUID               4sFUaC-fCVP-vxuO-7ox0-UcdU-qF8P-LL1sFx
```

10. Создайте 3 логических тома (LV):
    a. Том **lvm2_vg01_lv01** с размером 5 Gb.
    b. Том **lvm2_vg01_lv02** с размером 8 Gb.
    c. Том **lvm2_vg01_lv03** с размером 10 Gb.

```
# lvcreate -L 5G -n lvm2_vg01_lv01 lvm2_vg01
# lvcreate -L 8G -n lvm2_vg01_lv02 lvm2_vg01
# lvcreate -L 10G -n lvm2_vg01_lv03 lvm2_vg01
```

11. Убедитесь, что логические тома (LV) успешно созданы.

```
# lvs
```

12. На логическом томе (LV) **lvm2_vg01_lv01** создайте файловую систему XFS, а на томах **lvm2_vg01_lv01** и **lvm2_vg01_lv01** файловую систему ext4.

```
# mkfs.xfs /dev/lvm2_vg01/lvm2_vg01_lv01
# mkfs.ext4 /dev/lvm2_vg01/lvm2_vg01_lv02
# mkfs.ext4 /dev/lvm2_vg01/lvm2_vg01_lv03
```

13. Создайте каталоги **/mnt/lv01**, **/mnt/lv02** и **/mnt/lv03**.

```
# mkdir /mnt/lv0{1..3}
```

14. Добавьте в конфигурационный файл **/etc/fstab** записи, для автоматического монтирования логических томов (LV) в соответствующие каталоги:
    a. Том **lvm2_vg01_lv01** в каталог **/mnt/lv01**.
    b. Том **lvm2_vg01_lv02** в каталог **/mnt/lv02**.
    c. Том **lvm2_vg01_lv03** в каталог **/mnt/lv03**.

```
# vi /etc/fstab
```

```
...
/dev/lvm2_vg01/lvm2_vg01_lv01   /mnt/lv01   xfs    defaults   0 0
/dev/lvm2_vg01/lvm2_vg01_lv02   /mnt/lv02   ext4   defaults   0 0
/dev/lvm2_vg01/lvm2_vg01_lv03   /mnt/lv03   ext4   defaults   0 0
```

15. Смонтируйте все определенные в файле **/etc/fstab** файловые системы.

```
# mount -a
```

16. Убедитесь, что логические тома (LV) успешно смонтированы.

```
# df -h /mnt/lv0?
```

Задание 2: Изменение размеров LVM.

1. Просмотрите текущие физические тома (PV), группы томов (VG) и логические тома (LV).

```
# pvs
# vgs
# lvs
```

ПРИМЕЧАНИЕ:

На этом этапе вы должны получить следующие результаты:

```
# pvs
  PV         VG         Fmt  Attr PSize   PFree
  /dev/sda2  centos     lvm2 a--  <19.00g     0
  /dev/sdi1  lvm2_vg01  lvm2 a--  <10.00g  4.99g
  /dev/sdj1  lvm2_vg01  lvm2 a--  <10.00g <2.00g
  /dev/sdk1  lvm2_vg01  lvm2 a--  <10.00g     0
# vgs
  VG         #PV #LV #SN Attr   VSize   VFree
  centos       1   2   0 wz--n- <19.00g     0
  lvm2_vg01    3   3   0 wz--n- <29.99g <6.99g
# lvs
  LV              VG         Attr       LSize   Pool Origin Data%
  root            centos     -wi-ao---- <17.00g
  swap            centos     -wi-ao----   2.00g
  lvm2_vg01_lv01 lvm2_vg01 -wi-ao----   5.00g
  lvm2_vg01_lv02 lvm2_vg01 -wi-ao----   8.00g
  lvm2_vg01_lv03 lvm2_vg01 -wi-ao----  10.00g
```

2. Добавьте по 2 Gb к логическим томам (LV) **lvm2_vg01_lv01** и **lvm2_vg01_lv02**.

```
# lvextend -L +2G /dev/lvm2_vg01/lvm2_vg01_lv01
# lvextend -L +2G /dev/lvm2_vg01/lvm2_vg01_lv02
```

3. Убедитесь, что логические тома (LV) успешно увеличены.

```
# lvs
```

4. Увеличьте файловые системы на логических томах (LV) **lvm2_vg01_lv01** и **lvm2_vg01_lv02**.

```
# resize2fs /dev/lvm2_vg01/lvm2_vg01_lv02
# xfs_growfs /dev/lvm2_vg01/lvm2_vg01_lv01
```

5. Убедитесь, что файловые системы успешно увеличены.

```
# df -h /mnt/lv0?
```

6. Отмонтируйте логический том (LV) **lvm2_vg01_lv03**.

```
# umount /mnt/lv03/
```

7. Проверьте логический том логический том (LV) **lvm2_vg01_lv03** на наличие ошибок.

```
# e2fsck /dev/lvm2_vg01/lvm2_vg01_lv03
# e2fsck -f /dev/lvm2_vg01/lvm2_vg01_lv03
```

8. Уменьшите файловую систему на логическом томе (LV) **lvm2_vg01_lv03** до 5 Gb.

```
# resize2fs /dev/lvm2_vg01/lvm2_vg01_lv03 5G
```

9. Уменьшите логическом том (LV) **lvm2_vg01_lv03** на 5 Gb.

```
# lvreduce -L -5G /dev/lvm2_vg01/lvm2_vg01_lv03
```

10. При помощи **resize2fs** убедитесь, что файловая система не была нарушена во время уменьшения логического тома (LV).

```
# resize2fs /dev/lvm2_vg01/lvm2_vg01_lv03
```

11. Убедитесь, что логический тома (LV) успешно уменьшен.

```
# lvs
```

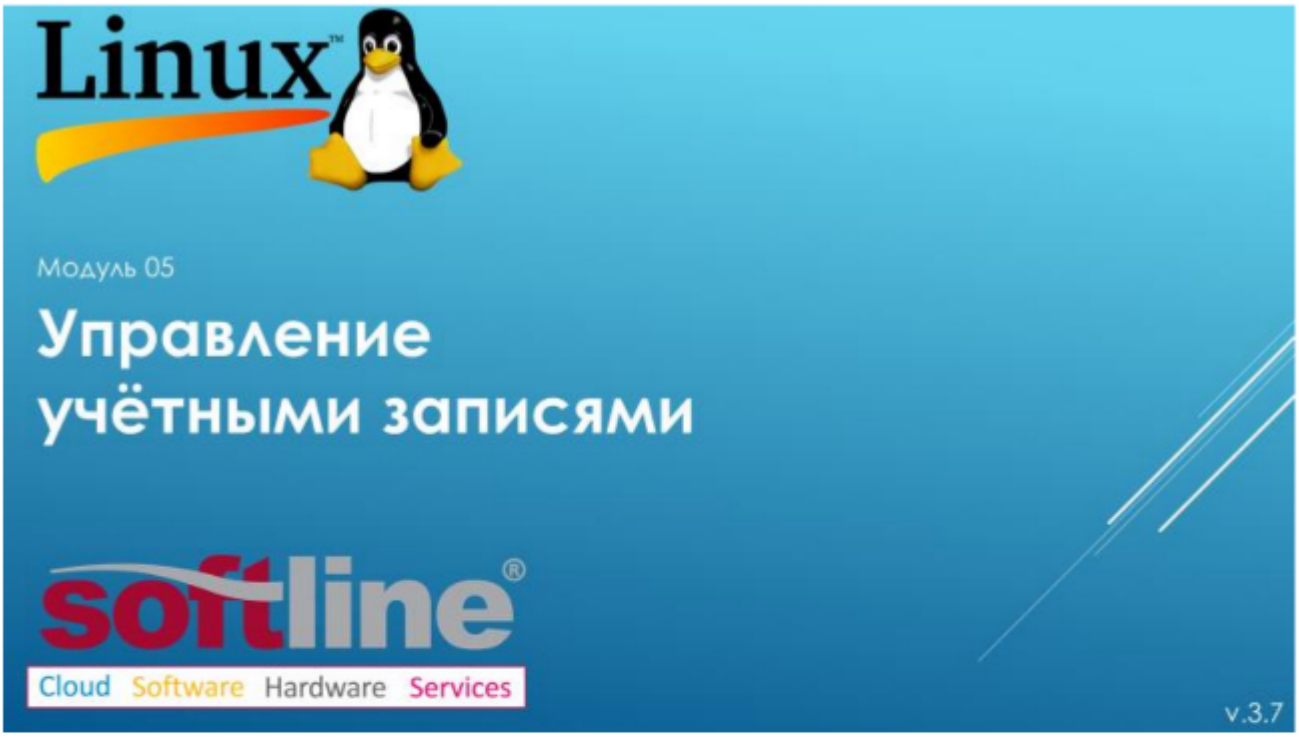
12. Смонтируйте логический том **lvm2_vg01_lv03**.

```
# mount /dev/lvm2_vg01/lvm2_vg01_lv03
```

13. Убедитесь, что файловая система успешно уменьшена.

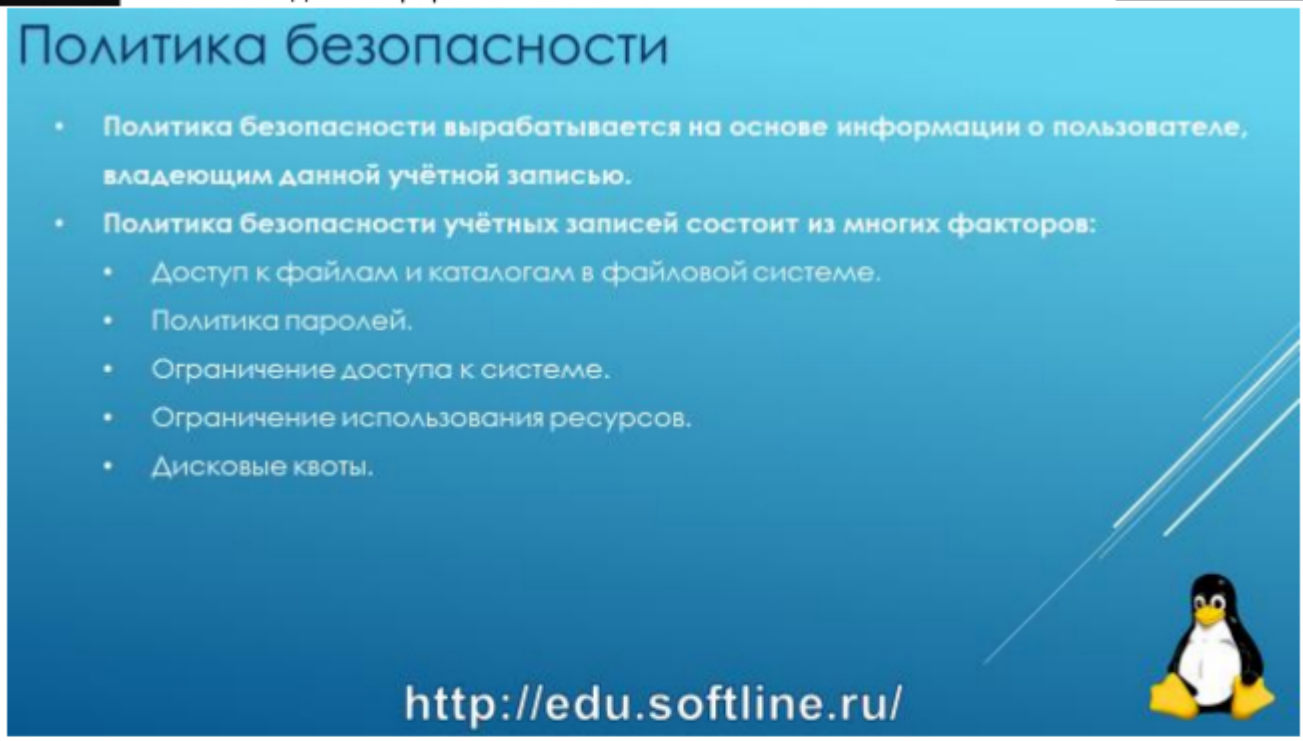
```
# df -h /mnt/lv0?
```

Модуль 5: Управление учетными записями.



В модуле представлены:

- Управление учетными записями пользователей.
- Управление учетными записями групп.
- Управление паролями.
- Конфигурационные файлы **`/etc/passwd`**, **`/etc/group`**, **`/etc/shadow`**, **`/etc/gshadow`**.
- Управление дисковыми квотами.
- Управление квотами XFS.



При управлении учётными записями, необходимо помнить о том, что у всех пользователей полномочия должны иметь четкие границы. Тщательное планирование политики учётных записей – один из китов, составляющих безопасность вычислительной системы.

Основные аспекты, на которые следует обратить внимание при планировании политики безопасности:

- Ограничение доступа к файлам и каталогам файловой системы. Как вы уже заметили, в операционной системе Linux пользователи имеют доступ не ко всем файлам. Более того, каждый файл может быть доступен учётной записи, за которой не закреплён пользователь (системной записи). Файл – основа операционной системы. Если простой пользователь получит доступ к файлу, в котором хранятся пароли, то такую систему нельзя назвать безопасной.
- Парольная политика очень важна для стабильной и безопасной системы. Использование коротких паролей, являющихся простыми словами небезопасно. Хороший пароль состоит из набора буков (в разном регистре) и цифр, чем длиннее такой пароль, тем он "прочнее", что затруднит процесс его взлома.
- Ограничение доступа к системе. Любой пользователь должен иметь доступ к той системе, которая ему необходима, никогда не давайте пользователю излишних привилегий. Иногда необходимо ограничить доступ по времени входа, согласитесь, странно, если секретарь захочет войти в систему в 3 часа ночи.
- Ограничения, касающиеся системных ресурсов, позволят не допустить большого расходования ресурсов пользователем.
- Дисковые квоты позволят ограничить максимальный объём дискового пространства, доступного пользователю.



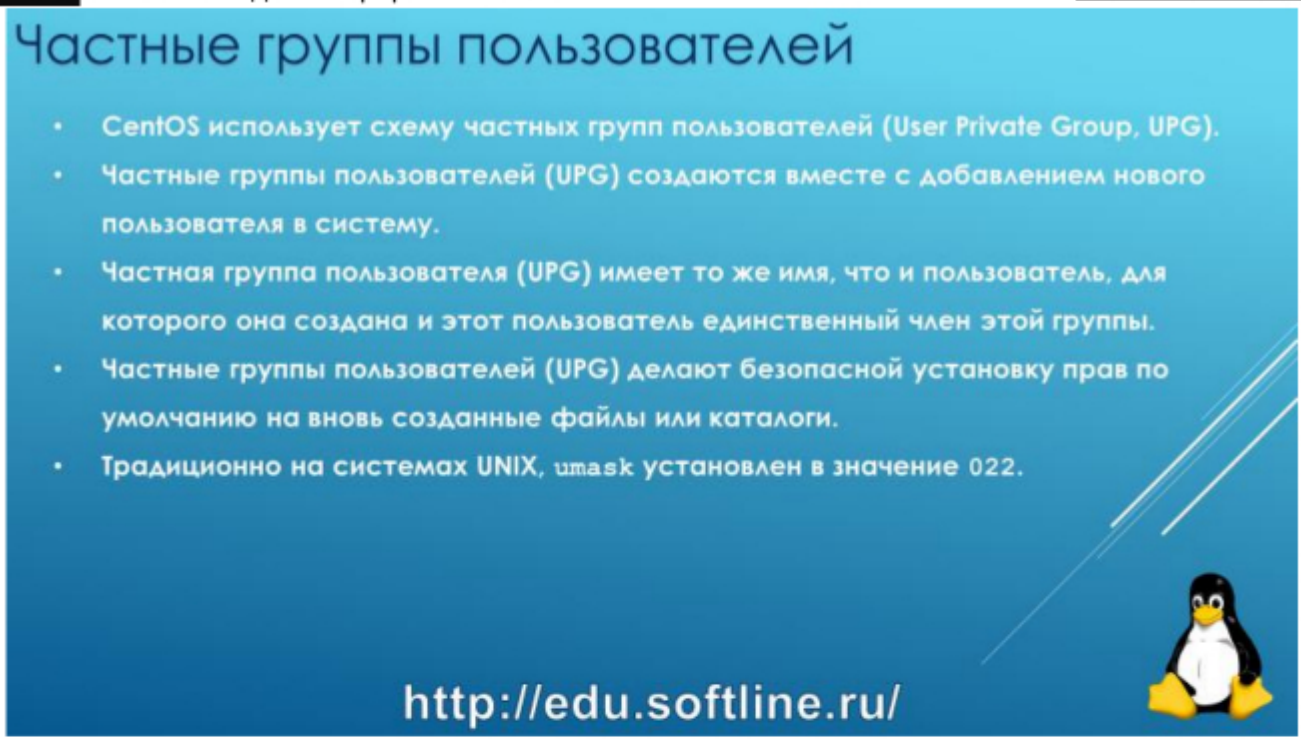
Управление пользователями и группами – это ключевой элемент системного администрирования CentOS (RHEL).

Описание пользователей и групп.

Пользователи одновременно могут быть и реальным людьми (аккаунтами, привязанными к физическим людям) и аккаунтами, которые необходимы для использования определенными приложениями. Группы – это логические выражения организации, объединяющие пользователей вместе для общих задач. Пользователи, состоящие в группе, могут одновременно читать, писать или выполнять файлы, принадлежащие группе.

Каждый пользователь связан с уникальным числовым номером, называемым User ID (UID). По аналогии каждая группа связанна с Group ID (GID). Пользователь, который создает файл, одновременно владелец (Owner) и член группы владельца (Owner Group) этого файла. Файлу назначаются разные комбинации прав чтение (Read), запись (Write) и выполнение (Execute) для владельца, группы владельца и всех остальных. Владелец файла может быть изменен только пользователем root, права доступа могут быть изменены только пользователем root и владельцем файла.

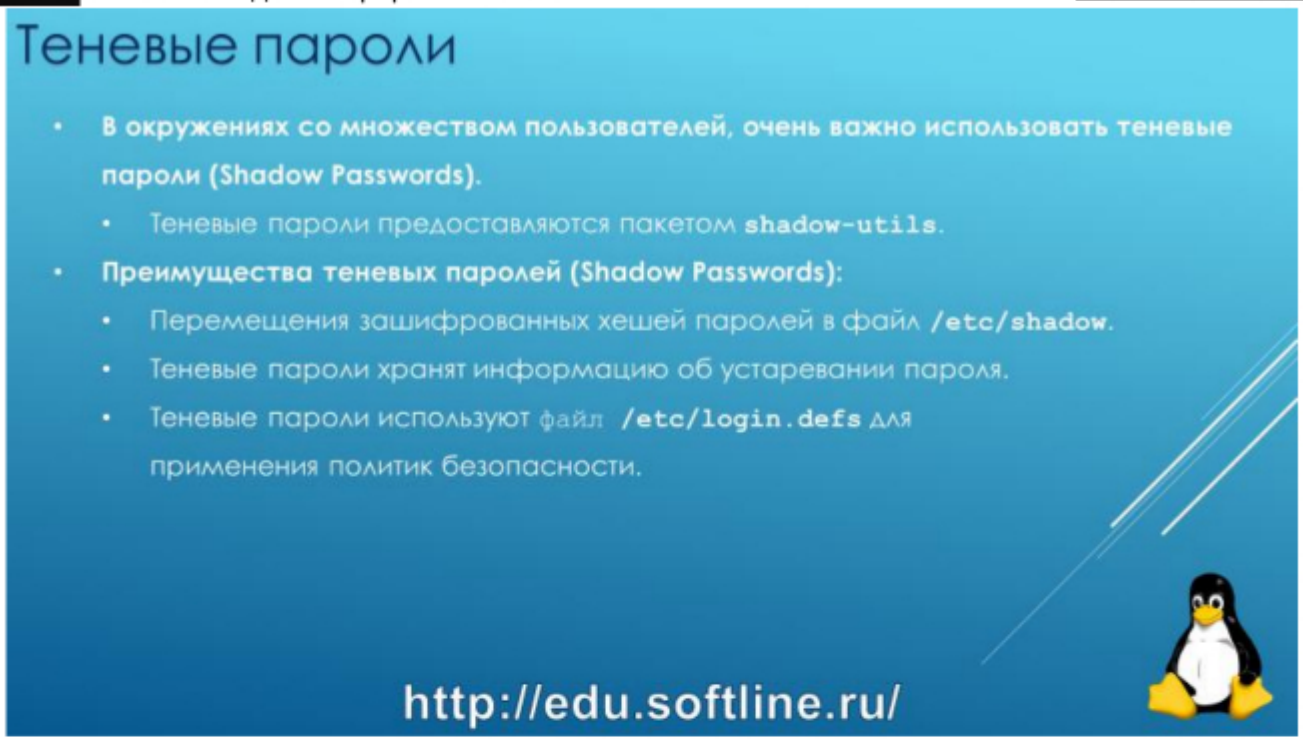
Дополнительно CentOS (RHEL) поддерживает списки контроля доступа (Access Control Lists, ACL) для файлов и каталогов, которые позволяют назначать права за пределами стандартных пользователя, группы и всех остальных.



CentOS (RHEL) использует схему частных групп пользователей (User Private Group, UPG), которая упрощает управление группами UNIX. Частные группы пользователей (UPG) создаются вместе с добавлением нового пользователя в систему. Частная группа пользователя (UPG) имеет то же имя, что и пользователь, для которого она создана и этот пользователь единственный член этой группы.

Частные группы пользователей (UPG) делают безопасной установку прав по умолчанию на вновь созданные файлы или каталоги, позволяя пользователю и его частной группе вносить изменения в файл или каталог.

Параметры, которые определяют права к вновь созданным файлам или каталогам называются **`umask`** и настраиваются в файле **`/etc/bashrc`**. Традиционно на системах UNIX, **`umask`** установлен в значение **`022`**, что позволяет только пользователям, которые создали файл или каталог вносить изменения. В рамках этой схемы, все остальные пользователи, включая членов группы создателя, не имеют права вносить какие-либо изменения. Однако, в рамках схемы UPG, данная защита группы (Group Protection) не обязательна, так как каждый пользователь имеет собственную частную группу.



В окружениях со множеством пользователей, очень важно использовать теневые пароли (Shadow Passwords), предоставляемые пакетом **shadow-utils** для расширения безопасности файлов системной аутентификации. Для этих целей, программа установщик включает теневые пароли по умолчанию.

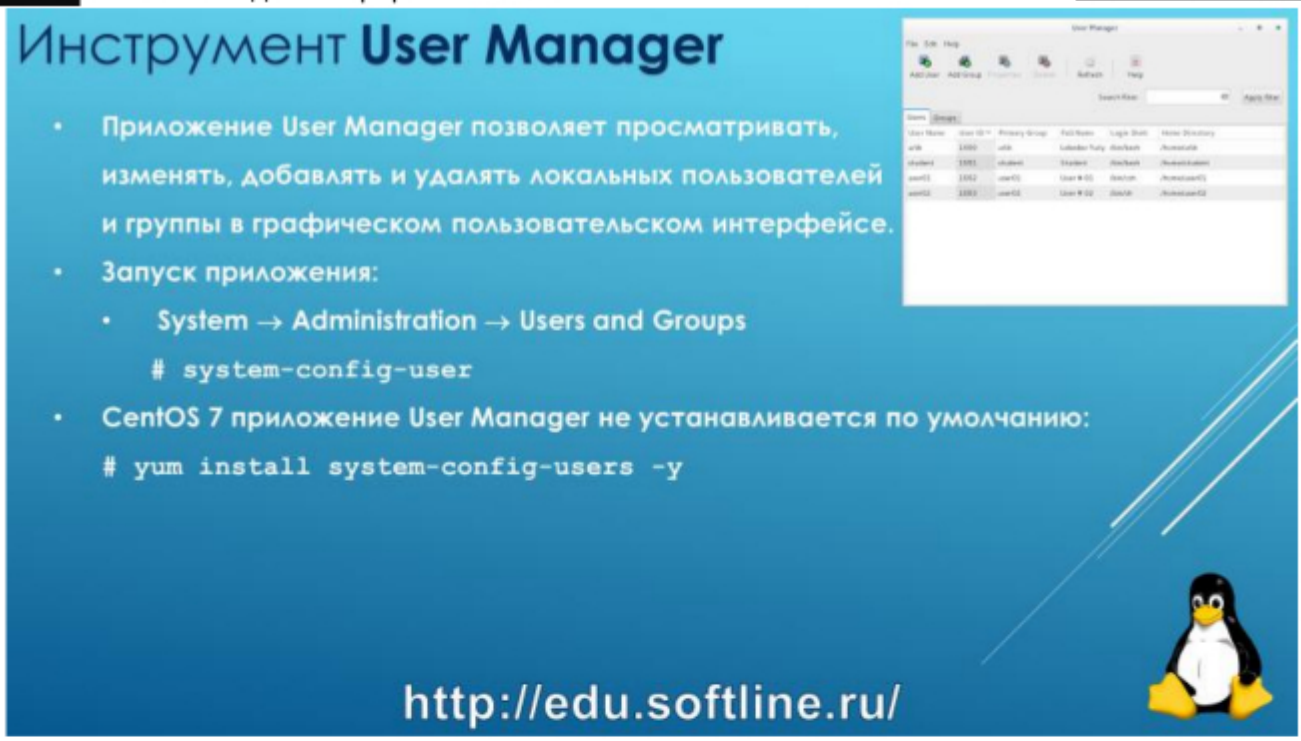
Список преимуществ теневых паролей над традиционным хранением паролей в системах UNIX:

- Теневые пароли улучшают безопасность системы, за счет перемещения зашифрованных хешей паролей в файл **/etc/shadow** (доступен для чтения только для пользователя root) из незашифрованного вида в файле **/etc/passwd**.
- Теневые пароли хранят информацию об устаревании пароля.
- Теневые пароли позволяют файлу **/etc/login.defs** для применения политик безопасности.

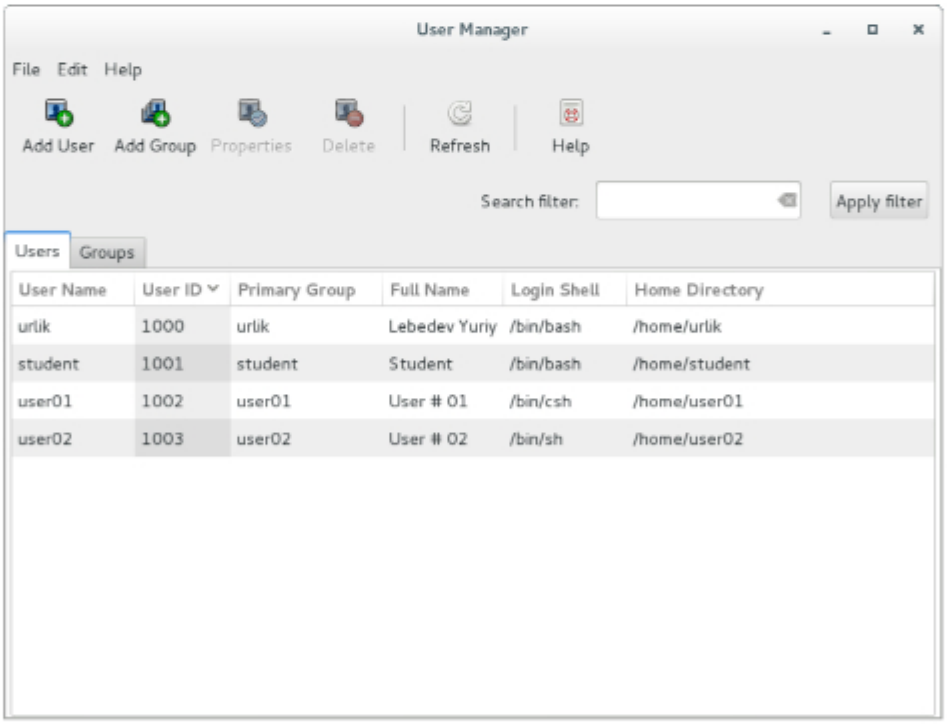
Большинство утилит, предоставляемых пакетом **shadow-utils**, работают корректно вне зависимости от того включено или отключено использование теневых паролей. Однако, с тех пор как информация об устаревании пароля эксклюзивно хранится в файле **/etc/shadow**, любые команды, которые создают или изменяют информацию об устаревании пароля не работают.

Список утилит, которые не работают, если не включены теневые пароли:

- Утилита **chage**.
- Утилита **gpasswd**.
- Команда **usermod** с опциями **-e** или **-f**.
- Команда **useradd** с опциями **-e** или **-f**.



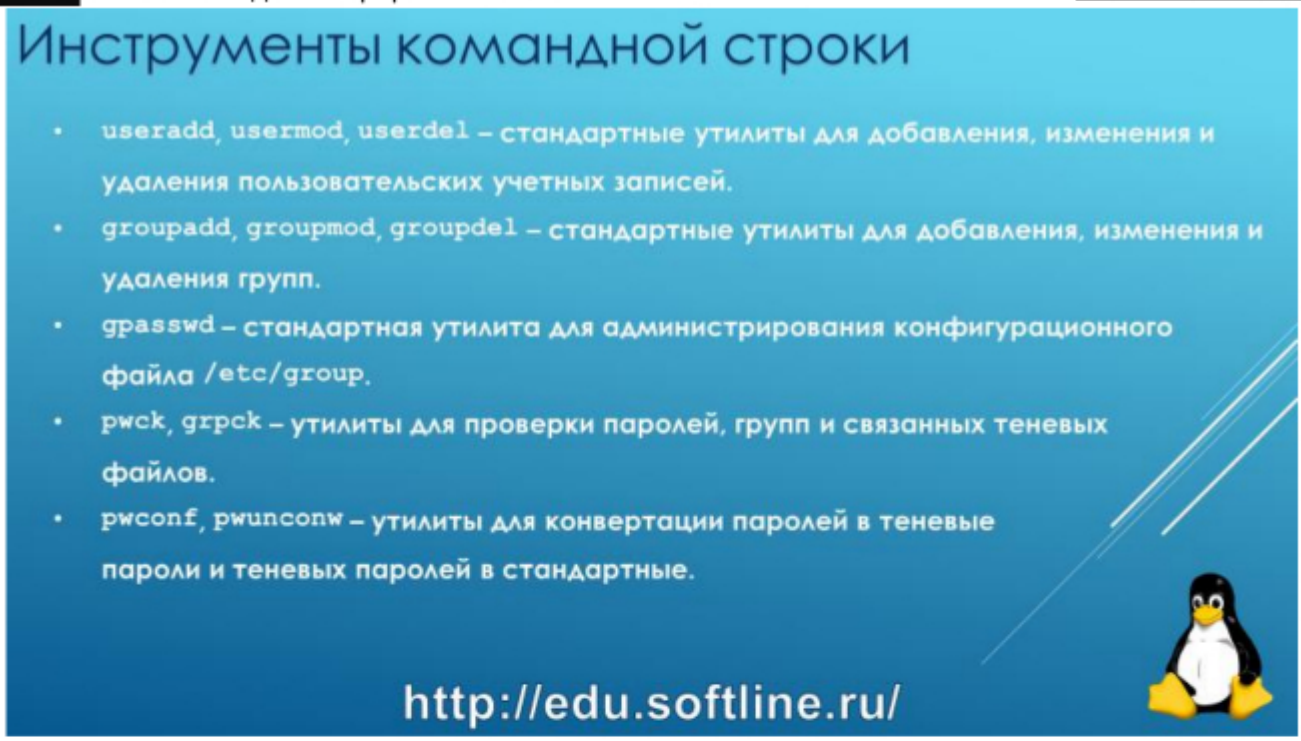
Приложение User Manager позволяет вам просматривать, изменять, добавлять и удалять локальных пользователей и группы в графическом пользовательском интерфейсе. Для запуска приложения перейдите в **System → Administration → Users and Groups** из панели или введите **`system-config-user`** в командной оболочке. Если вы не имеете привилегий супер-пользователя, приложение запросит вас пройти аутентификацию от имени пользователя root.



ПРИМЕЧАНИЕ:

В CentOS 7 приложение User Manager не устанавливается по умолчанию в режиме установки Gnome Desktop, для его установки воспользуйтесь следующей командой:

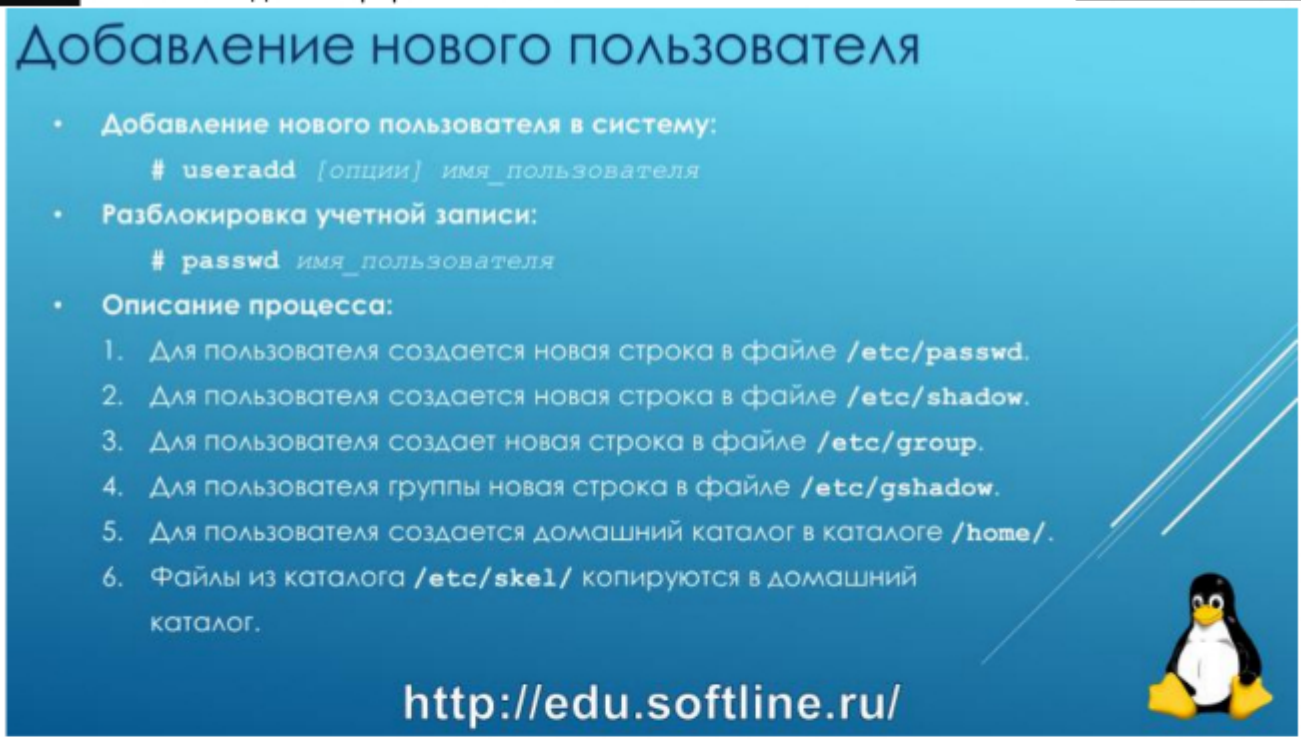
```
# yum install system-config-users -y
```



Наиболее простой путь управления пользователями и группами в CentOS – это использования приложения User Manager. Однако, если вы предпочитаете инструменты командной строки или на системе не установлена X Window System, вы можете использовать утилиты командной строки.

Утилиты командной строки для управления пользователями и группами.

Утилиты	Описание
useradd, usermod, userdel	Стандартные утилиты для добавления, изменения и удаления пользовательских учетных записей.
groupadd, groupmod, groupdel	Стандартные утилиты для добавления, изменения и удаления групп.
gpasswd	Стандартная утилита для администрирования конфигурационного файла /etc/group.
pwck, grpck	Утилиты для проверки паролей, групп и связанных теневых файлов.
pwconf, pwunconw	Утилиты для конвертации паролей в теневые пароли и теневых паролей в стандартные.



Для добавления нового пользователя в систему, ведите следующую команду, от имени пользователя root:

```
# useradd [опции] имя_пользователя
```

По умолчанию команда **useradd** создает заблокированную учетную запись пользователя. Для разблокировки учетной записи, выполните следующую команду от имени пользователя root:

```
# passwd имя_пользователя
```

Опционально можно установить политику устаревания пароля.

Таблица популярных опций командной строки команды **useradd**.

Опция	Описание
-c *'комментарий'*	Комментарий, как правило используют для указания полного имени.
-d *домашний_каталог*	Домашний каталог, который будет использоваться вместо **/home/***имя_пользователя*.
-e *дата*	Дата для учетной записи, в которую она будет отключена в формате *ГГГГ-ММ-ДД*.
-f *дни*	Число дней, после истечения срока действия пароля, когда учетная запись будет отключена. Если указано значение **0**, учетная запись отключается сразу после истечения срока действия пароля. Если указано значение **-1**, учетная запись не будет отключаться после истечения срока действия пароля.
-g *имя_группы*	Имя группы или GID, указывает группу пользователя по умолчанию. Группа должна быть создана предварительно.
-G *список_групп*	Список дополнительных групп или числовых значений GID (помимо группы по умолчанию), в которые необходимо добавить пользователя. Группы должны быть созданы предварительно.
-m	Создает домашний каталог, если он не существует.
-M	Не создает домашний каталог.
-N	Не создает частной группы для пользователя.

-p *пароль*	Пароль, зашифрованный при помощи **crypt**.
-r	Создает системную учетную запись с UID меньше 500 и без домашнего каталога.
-s	Оболочка пользователя, по умолчанию: **/bin/bash**.
-u *UID*	Указывает UID пользователя, должен быть уникальным и больше чем 499.

Описание процесса.

Следующие шаги иллюстрируют действия, которые происходят на системе с включенными теневыми паролями, когда вызывается команда **useradd student**.

1. Для пользователя **student** создается новая строка в файле **/etc/passwd**:

   student:x:501:501::/home/juan:/bin/bash

В строке содержится 7 полей:

1	**student**	Имя пользователя (username).
2	**x**	Обозначает, что система использует теневые пароли.
3	**501**	UID, в CentOS (RHEL) значения UID до 500 зарезервированы для системного использования и не должны назначаться пользователям.
4	**501**	GID, в CentOS (RHEL) значения GID до 500 зарезервированы для системного использования и не должны назначаться пользователям.
5	*пусто*	опциональная информация (GECOS), как правило используется для указания полного имени или какой-либо дополнительной информации.
6	**/home/student**	домашний каталог.
7	**/bin/bash**	оболочка по умолчанию.

2. Для пользователя **student** создается новая строка в файле **/etc/shadow**:

   student:!!:14798:0:99999:7:::

В строке содержится 9 полей:

1	**student**	Имя пользователя
2	**!!**	Зашифрованный пароль, два восклицательных знака (**!!**) – блокируют учетную запись.
3	**14798**	Дата последнего изменения пароля.
4	**0**	Минимальный возраст пароля.
5	**99999**	Максимальный возраст пароля.
6	**7**	Период предупреждения об истечении пароля.
7	*пусто*	Период неактивного пароля.
8	*пусто*	Дата устаревания учетной записи.
9	*пусто*	Зарезервированное поле.

3. Для пользователя student создает новая строка в файле **/etc/group**, содержащая информацию о частной группе пользователя:

   student:x:501:

В строке содержится 4 поля:

1	**student**	Имя группы.
2	**x**	Обозначает, что система использует теневые пароли
3	**501**	GID, в CentOS (RHEL) значения GID до 500 зарезервированы для системного использования и не должны назначаться пользователям.
4	*пусто*	Список членов группы (указываются username), за исключением владельца частной группы.

4. Для пользователя группы новая строка в файле **/etc/gshadow**:

```
student:!::
```

В строке содержится 4 поля:

1	**student**	Имя группы.
2	**!**	Зашифрованный пароль, восклицательный знак обозначает что группа заблокирована.
3	*пусто*	Администраторы группы.
4	*пусто*	Члены группы.

5. Каталог для пользователя **student** создается в каталоге **/home/**

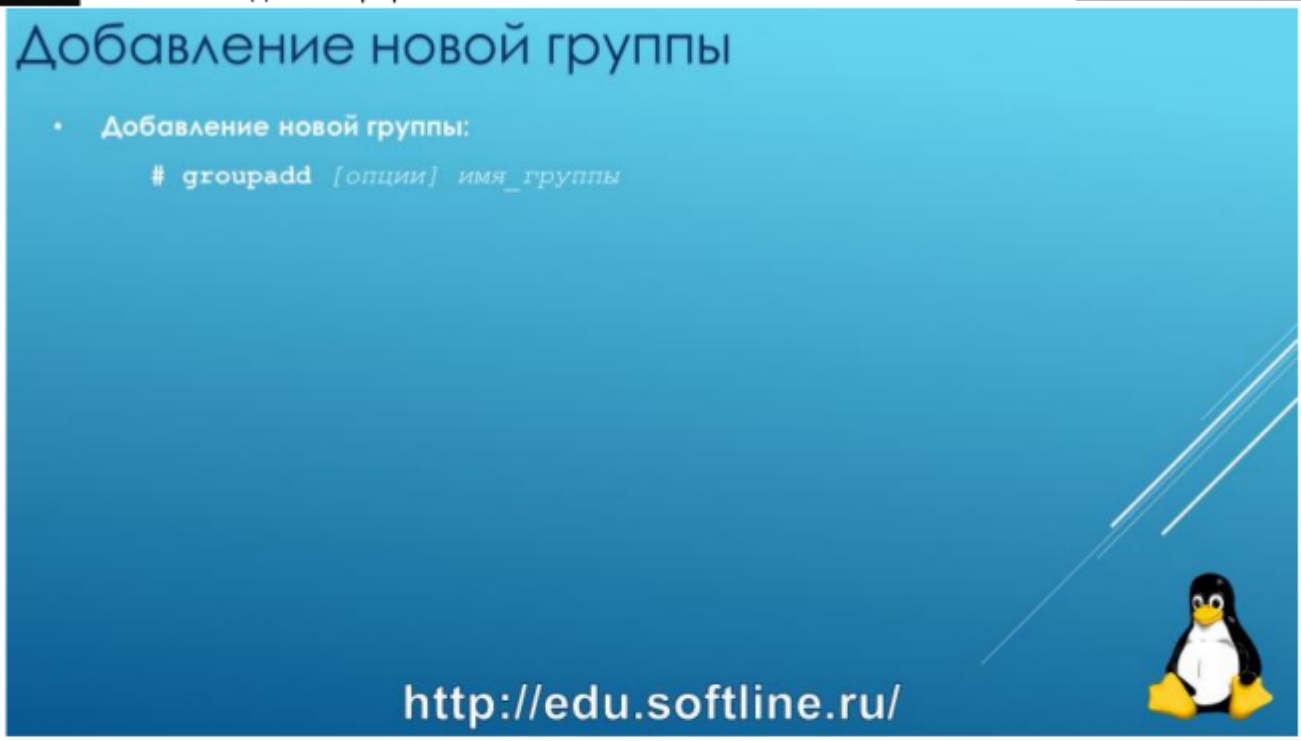
```
drwx------. 4 juan juan 4096 Mar 3 18:23 student
```

Этот каталог принадлежит пользователю **student** и группе **student**. Права на чтение, запись и выполнение предоставлены только пользователю student. Все остальные права запрещены.

6. Файлы из каталога **/etc/skel/** (который содержит настройки пользователя по умолчанию) копируются в созданный на предыдущем шаге домашний каталог пользователя.

```
drwx------. 4 student student 4096 Mar 3 18:23 .
drwxr-xr-x. 5 root root 4096 Mar 3 18:23 ..
-rw-r--r--. 1 student student 18 Jun 22 2010 .bash_logout
-rw-r--r--. 1 student student 176 Jun 22 2010 .bash_profile
-rw-r--r--. 1 student student 124 Jun 22 2010 .bashrc
drwxr-xr-x. 2 student student 4096 Jul 14 2010 .gnom e2
drwxr-xr-x. 4 student student 4096 Nov 23 15:09 .mozilla
```

На этом этапе заблокированная учетная запись **student** существует на системе. Для ее активации необходимо назначить пароль, при помощи команды **passwd**, опционально можно задать дополнительные параметры пароля.

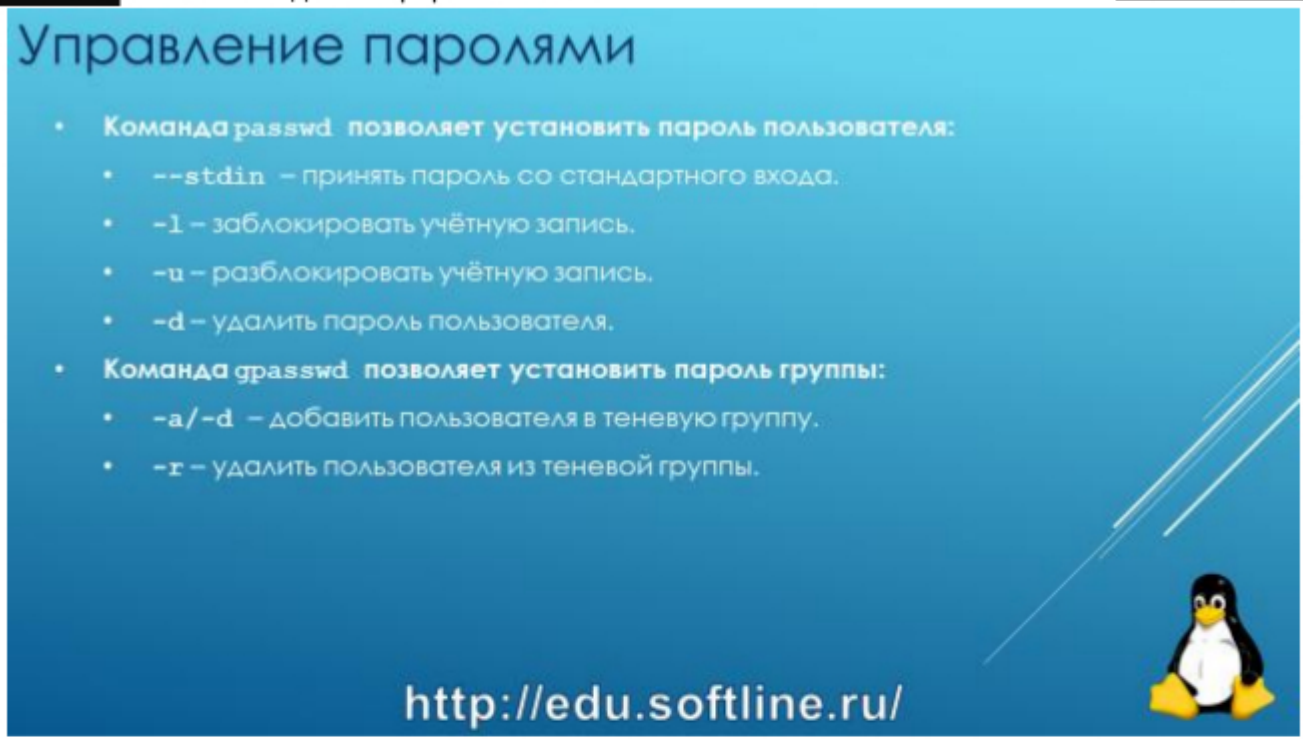


Для добавления новой группы на систему, введите следующую команду от имени пользователя root:

```
# groupadd [опции] имя_группы
```

Таблица популярных опций командной строки команды **groupadd**.

Опция	Описание
-f, **--force**	Комментарий, как правило используют для указания полного имени.
-g *GID*	GID, в CentOS (RHEL) значения GID до 500 зарезервированы для системного использования и не должны назначаться пользователям.
-K, **--key** *ключ=значение*	Переписывает значения по умолчанию, определенные в файле **/etc/login.defs**.
-o, **--non-unique**	Позволяет создавать дубликаты групп с неуникальным GID.
-p, --password	Указывает зашифрованный пароль для новой группы.
-r	Создает системную группу с GID меньше 500.



Пароли – одно из самых уязвимых мест в системе безопасности IT-инфраструктуры. Администратор может организовать очень защищённую структуру сети, но пользователь, имеющий высокие привилегии и использующий "слабый" пароль, может сохранить огромную уязвимость всей системы. Именно по этой причине, хороший администратор обязан сделать всё для того, чтобы пользователи не могли использовать "слабые" пароли.

Какие действия необходимо принять, чтобы пароли пользователей не ослабляли систему безопасности:

- Планирование и создание парольной политики, запрещающей установление "слабых" паролей;
- Создание паролей при помощи генераторов паролей;
- Периодический аудит паролей всех учётных записей.

Пароли учётных записей пользователей задаются с помощью программы **passwd**:

```
# passwd ivan
```

Программа **passwd** используется не только для задания пароля, но и для блокирования/разблокирования учётных записей пользователей:

```
# passwd -l ivan
# passwd -u ivan
```

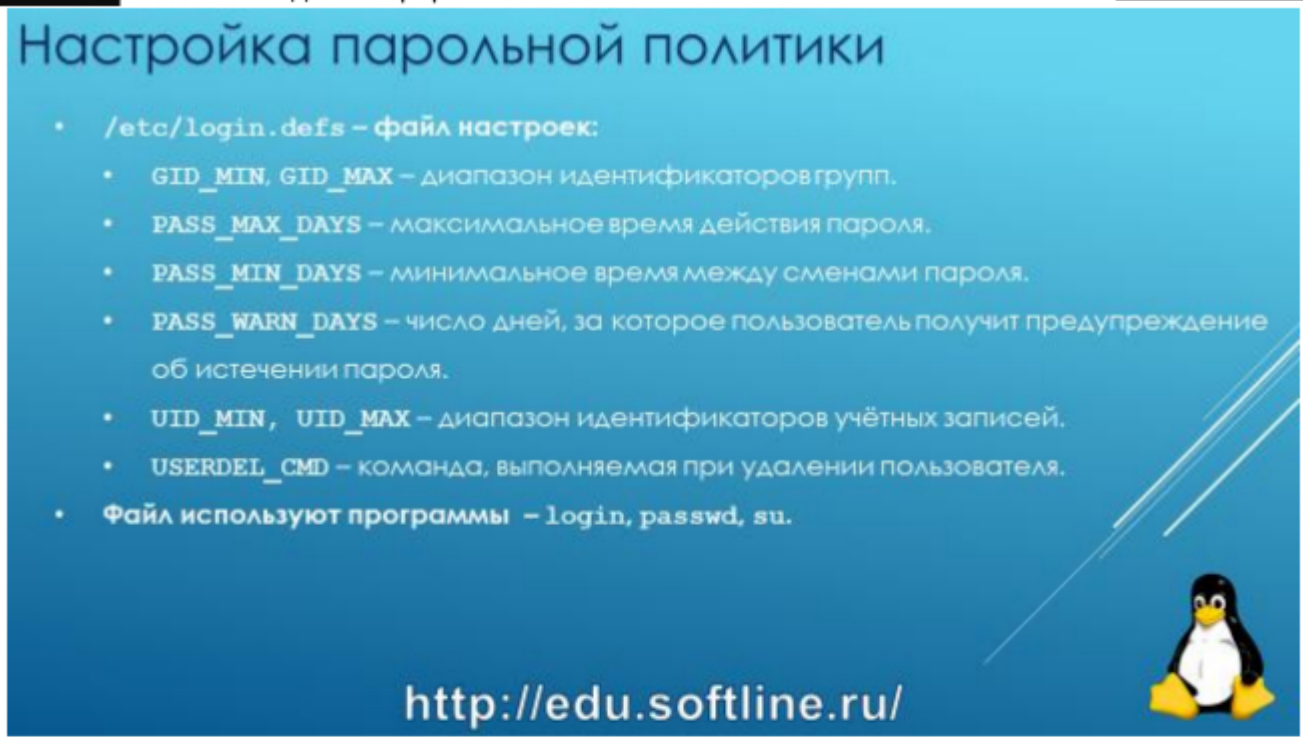
Команду passwd можно использовать в сценариях командного процессора, используя параметр **-stdin**, который указывает, что пароль необходимо принять на стандартный канал входа:

```
# echo password > passwd –stdin ivan
```

Как говорилось ранее, пароли могут быть заданы не только для учётных записей пользователей, но и для учётных записей групп. Задать пароль учётной записи группы можно с помощью команды **gpasswd**:

```
# gpasswd marketing
```

Также, **gpasswd** используется для добавления учётных записей пользователей в теневую группу.

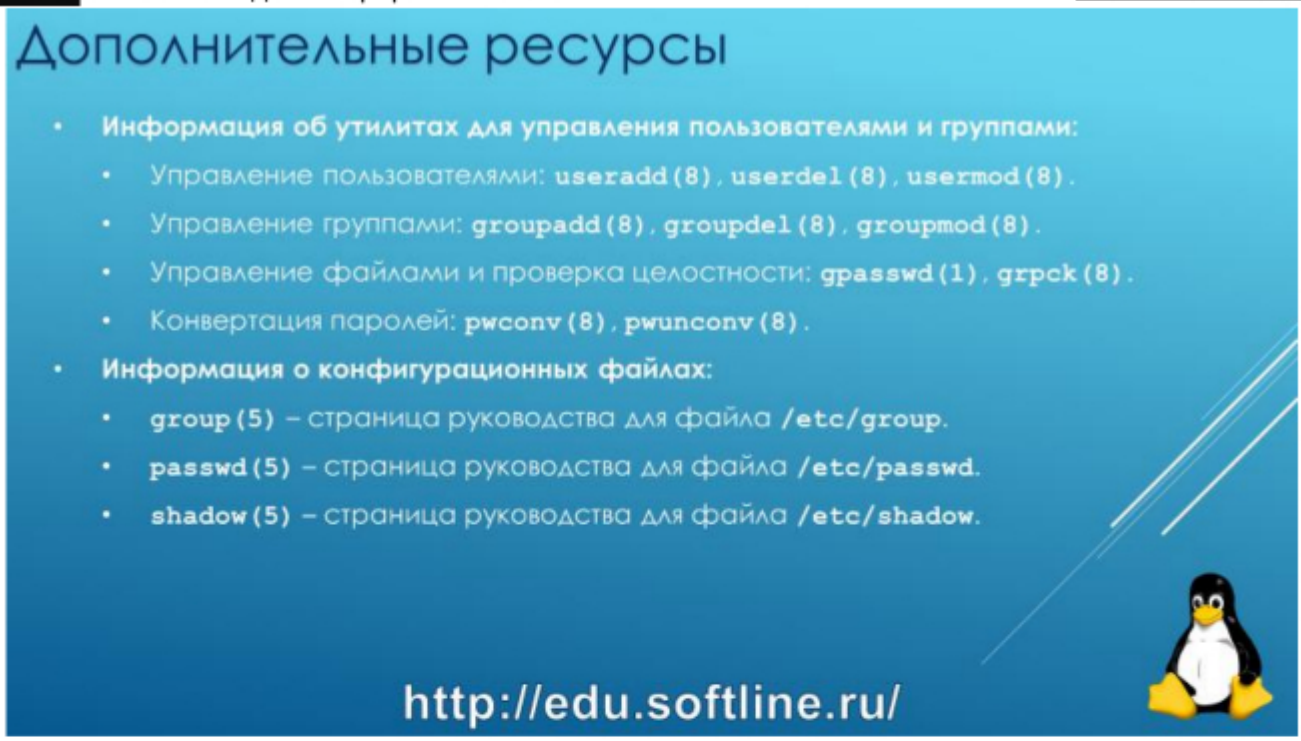


Файл **/etc/login.defs** используется для настройки служб, использующих систему теневых паролей. Это текстовый файл, каждая строка описывает один параметр.

С помощью этого файла задаётся парольная политика, политика учётных записей и многое другое:

- **GID_MIN/GID_MAX** – определение диапазона идентификаторов, из которого будут браться номера для создания новых учётных записей.
- **UID_MIN/UID_MAX** – определение диапазона идентификаторов, из которого будут браться номера для создания новых учётных записей.
- **PASS_MAX_DAYS** – определяет максимальное время актуальности пароля.
- **PASS_WARN_DAYS** – определяет число дней, в течение которых пользователю будет выдаваться предупреждение о скором блокировании учётной записи.
- **PASS_MIN_LEN** – минимальная длина пароля.
- **USERDEL_CMD** – указание команды, которую необходимо выполнить при удалении пользователя. Это очень полезный параметр, позволяющий удалять все "хвосты" автоматически.

Дополнительная безопасность обеспечивается библиотекой **cracklib**, которая позволяет проверить устанавливаемый пароль с помощью словарей.



Для получения дополнительной информации об управлении пользователями и группами, воспользуйтесь следующими ресурсами:

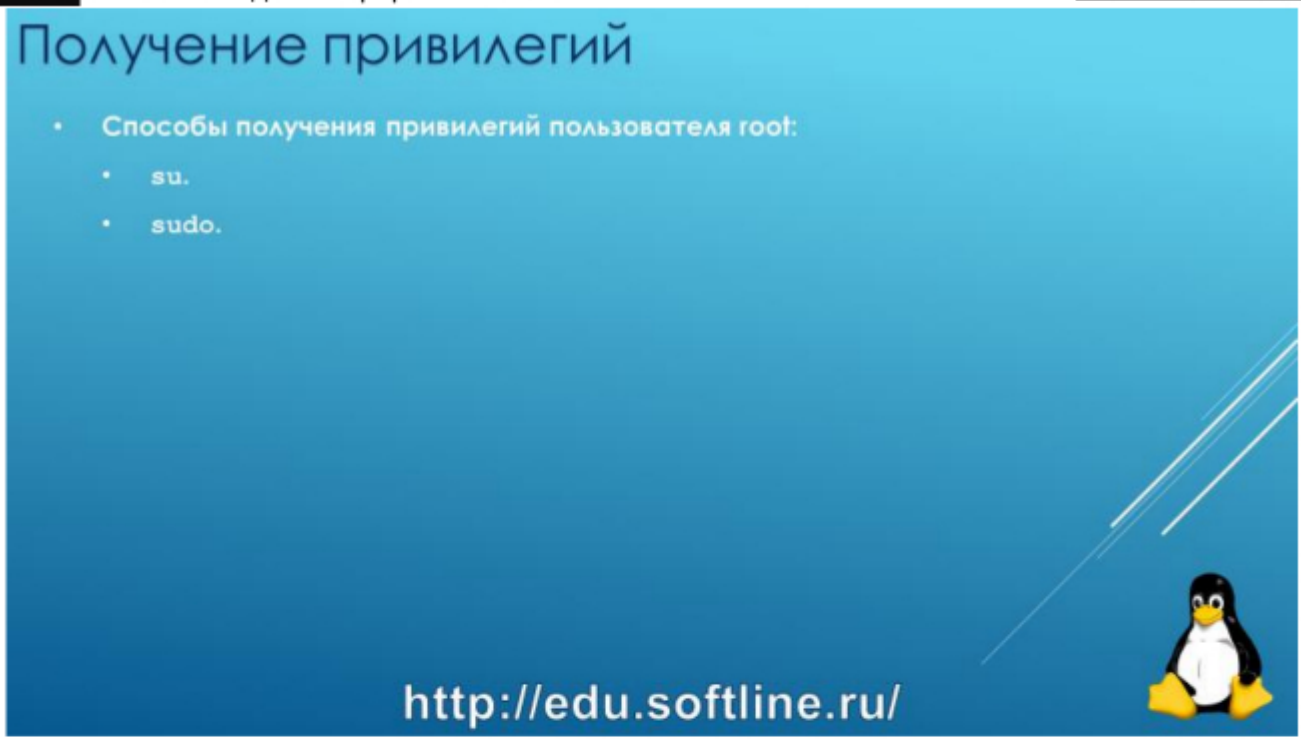
Установленная документация.

Для получения информации об утилитах для управления пользователями и группами в CentOS (RHEL), воспользуйтесь следующими страницами руководства:

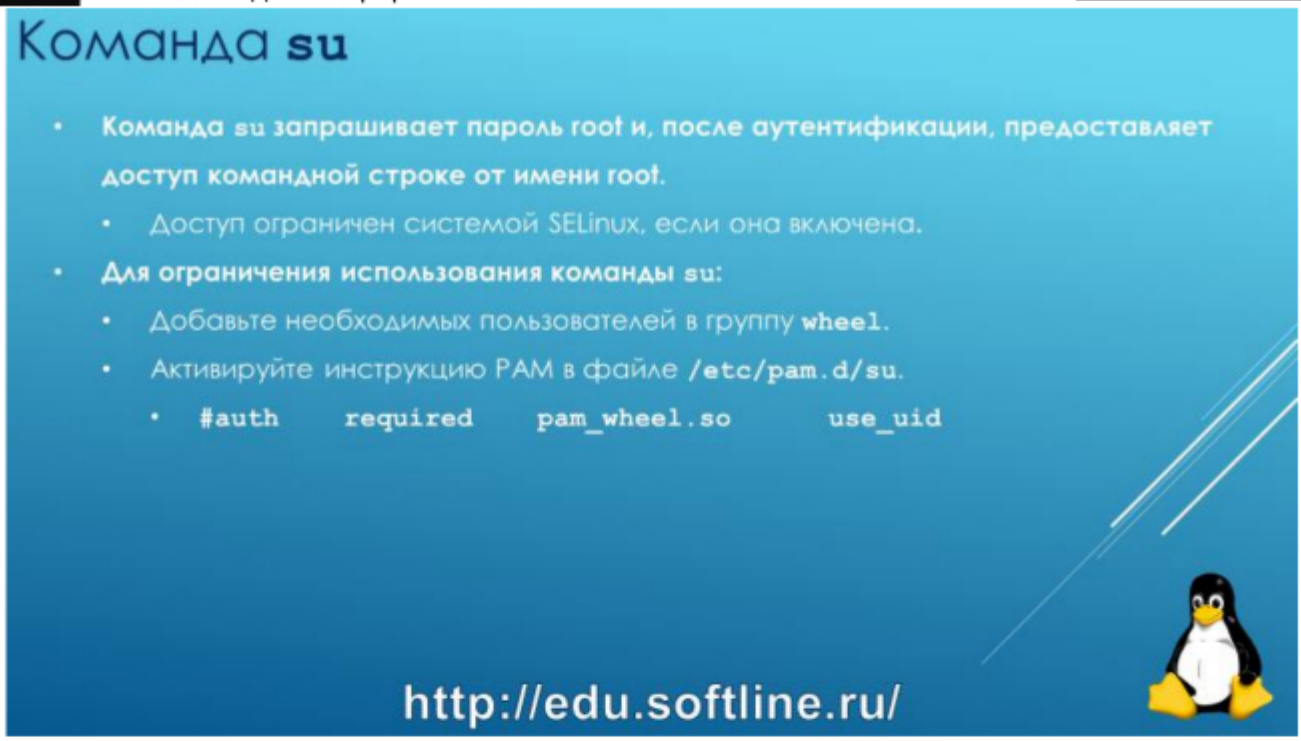
- **useradd(8)** – страница руководства для команды **useradd**, содержит информацию о создании новых пользователей.
- **userdel(8)** – страница руководства для команды **userdel**, содержит информацию об удалении пользователей.
- **usermod(8)** – страница руководства для команды **usermod**, содержит информацию об изменении пользователей.
- **groupadd(8)** – страница руководства для команды **groupadd**, содержит информацию о создании новых групп.
- **groupdel(8)** – страница руководства для команды **groupdel**, содержит информацию об удалении групп.
- **groupmod(8)** – страница руководства для команды **groupmod**, содержит информацию об изменении групп.
- **gpasswd(1)** – страница руководства для команды **gpasswd**, содержит информацию об управлении файлом **/etc/group**.
- **grpck(8)** – страница руководства для команды **grpck**, содержит информацию о проверке целостности файла **/etc/group**.
- **pwck(8)** – страница руководства для команды **gwck**, содержит информацию о проверке целостности файлов **/etc/passwd** и **/etc/shadow**.
- **pwconv(8)** – страница руководства для команды **pwconw**, содержит информацию о конвертации стандартных паролей в теневые.
- **pwunconv(8)** – страница руководства для команды **pwunconv**, содержит информацию о конвертации теневых паролей в стандартные.

Для получения информации о связанных конфигурационных файлах, воспользуйтесь следующими страницами руководства:

- **group(5)** – страница руководства для файла **/etc/group**, содержит информацию об определении групп в системе.
- **passwd(5)** – страница руководства для файла **/etc/passwd**, содержит информацию об определении пользователей в системе.
- **shadow(5)** – страница руководства для файла **/etc/shadow**, содержит информацию об определении пароля и его параметров.



Системные администраторы и, в некоторых случаях, пользователи, должны выполнять определенные задачи с применением административного доступа. Получение доступа к системе с правами пользователя root несет потенциальную опасность и может привести к причинению непреднамеренного вреда системе и данным. Данный раздел описывает пути получения административных привилегий при помощи программ смены идентификатора пользователя, таких как **su** и **sudo**. Данные программы позволяют указанным пользователям выполнят задачи, которые обычно доступны только пользователю root, при этом поддерживают высокий уровень управления и безопасности системы.



Когда пользователь выполняет команду **su**, у него запрашивается пароль root и, после аутентификации, предоставляет доступ командной строке от имени пользователя root.

После входа с использованием команды **su**, пользователь становиться пользователем root и получает абсолютный административный доступ к системе. Примечательно, что доступ ограничен системой SELinux, если она включена. Дополнительно, после того как пользователь повысил свои привилегии до root, он может использовать команду **su** для переключения в любого пользователя на системе без запроса пароля.

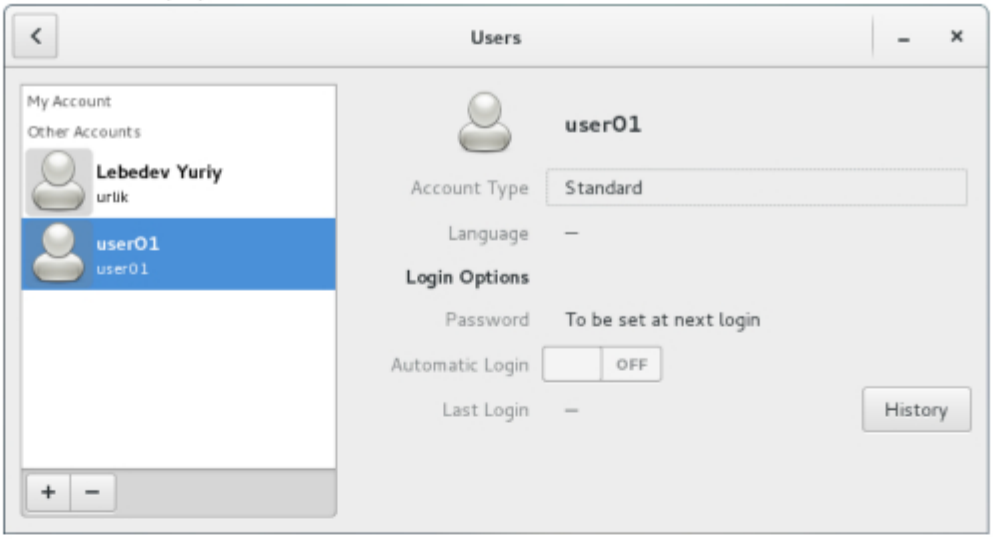
Так как данная программа настолько сильна, администраторы компании могут быть заинтересованы в ограничении доступа к этой команде.

Один из простейших путей сделать это, это добавить пользователя в специальную административную группу wheel. Для этого выполните следующую команду от имени пользователя root:

```
# usermod -a -G wheel username
```

В предыдущей команде замените *username* на имя пользователя, которого хотите добавить в группу wheel.

Вы можете также использовать утилиту настройки **Users**. Сменив тип учетной записи (Account Type) со стандартного (Standard) на административный (Administrator) вы добавите пользователя в группу wheel.

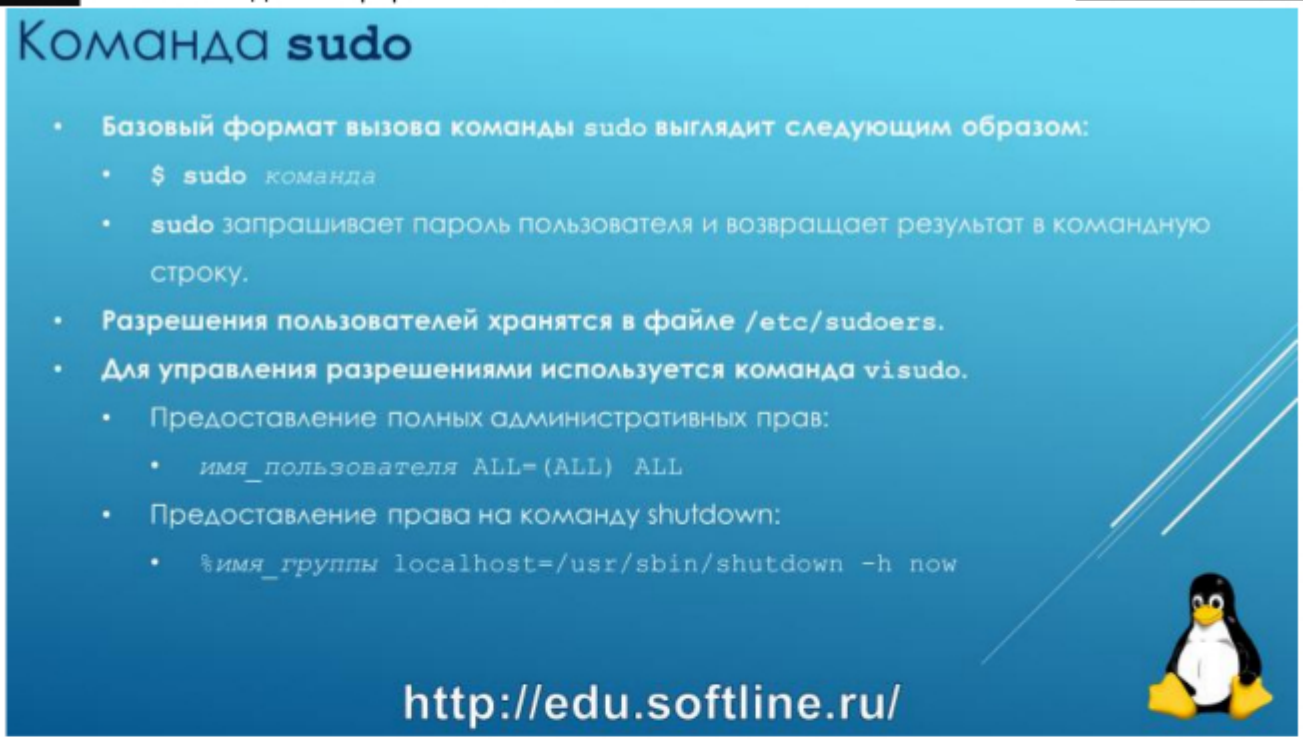


После добавление необходимых пользователей в группу wheel, можно сделать так, чтобы только ее члены смогут использовать команду **su**. Для этого необходимо отредактировать конфигурационный файл Pluggable Authentication Module (PAM) для **su – /etc/pam.d/su**. Откройте файл в текстовом редакторе и раскомментируйте следующую строку, удалив символ **#**:

```
#auth     required  pam_wheel.so   use_uid
```

Это изменение обозначает, что только члены административной группы wheel могут переключать пользователя при помощи команды **su**.

ПРИМЕЧАНИЕ:
Пользователь root является частью группы wheel по умолчанию.



Команда **sudo** предлагает иной подход к предоставлению пользователю административного доступа. Когда доверенный пользователь вызывает административную команду при помощи **sudo**, у него запрашивается его собственный пароль. Затем, после успешной аутентификации и проверки разрешения на выполнение команды, административная команд выполняется с привилегиями пользователя root.

Базовый формат вызова команды **sudo** выглядит следующим образом:

```
$ sudo команда
```

В примере выше, *команда* может быть заменена на команду, зарезервированную для пользователя root, такую как **mount**.

Команда **sudo** обеспечивает высокий уровень гибкости. Для примера, только пользователи, перечисленные в конфигурационном файле **/etc/sudoers** могут пользоваться командой **sudo** и команды выполняются в пользовательской оболочке, не в оболочке пользователя root. Это значит, что оболочка пользователя root может быть полностью отключена.

Каждая успешная аутентификация при помощи команды **sudo** записывается в файл **/var/log/messages**, а вызванная команда вместе с именем вызывавшего записывается в файл **/var/log/secure**. Если требуется расширенное ведение журнала, можно воспользоваться модулем **pam_tty_audit** для включения аудита TTY для указанных пользователей при помощи добавления следующих строк в ваш файл **/etc/pam.d/system-auth**:

```
session required pam_tty_audit.so disable=pattern enable=pattern
```

Где *pattern* заменяется разделенным запятыми списком пользователей с опциональным использование подстановочных символов. Например, следующая конфигурация включает аудит TTY для пользователя root и отключает его для всех остальных пользователей:

```
session required pam_tty_audit.so disable=* enable=root
```

ПРИМЕЧАНИЕ:

Настройка модуля PAM **pam_tty_audit** *для аудита TTY записывает только ввод TTY. Это значит, когда пользователь, на котором ведется аудит, входит в систему,* **pam_tty_audit** *записывает вводимые пользователем строки в файл* **/var/log/audit/audit.log**. *Для получения дополнительной информации, обратитесь к странице руководства* **pam_tty_audit(8)**.

Другим преимуществом команды **sudo** является возможность администратора разрешить разным пользователям доступ к указанным командам в соответствии с их нуждами.

Для редактирования конфигурационного файла **/etc/sudoers** администраторы должны использовать команду **visudo**.

Чтобы дать кому-либо полные административные права, введите **visudo** и добавьте строку указания привилегий:

```
student ALL=(ALL) ALL
```

В данном примере пользователь `student` может использовать **sudo** с любого хоста и выполнять любую команду.

Пример ниже иллюстрирует гранулированные возможности настройки **sudo**:

```
%users localhost=/usr/sbin/shutdown -h now
```

Данный пример указывает, что любой член системной группы `users` может вызвать команду **/sbin/shutdown -h now** из локальной консоли машины.

Детальное описание опций файла содержатся на странице **sudoers** руководства man.

ПРИМЕЧАНИЕ:

Есть несколько потенциальных опасностей о которых нужно знать при использовании команды **sudo**. *Вы можете избежать их за счет редактирования конфигурационного файла* **/etc/sudoers** *при помощи команды* **visudo**, *как описано выше. Если оставить файл* **/etc/sudoers** *в его состоянии по умолчанию, то это дает каждому пользователю из группы wheel неограниченные права от имени root.*

- *По умолчанию,* **sudo** *хранит пароль пользователя в течении пяти минут. Любое последующее использование команды в течении этого периода не запрашивает пароль пользователя. Это может быть использовано злоумышленником, при условии, что пользователь оставит свою рабочую станцию не заблокированной без присмотра. Это поведение может быть изменено за счет добавления следующей строки в файл* **/etc/sudoers**:

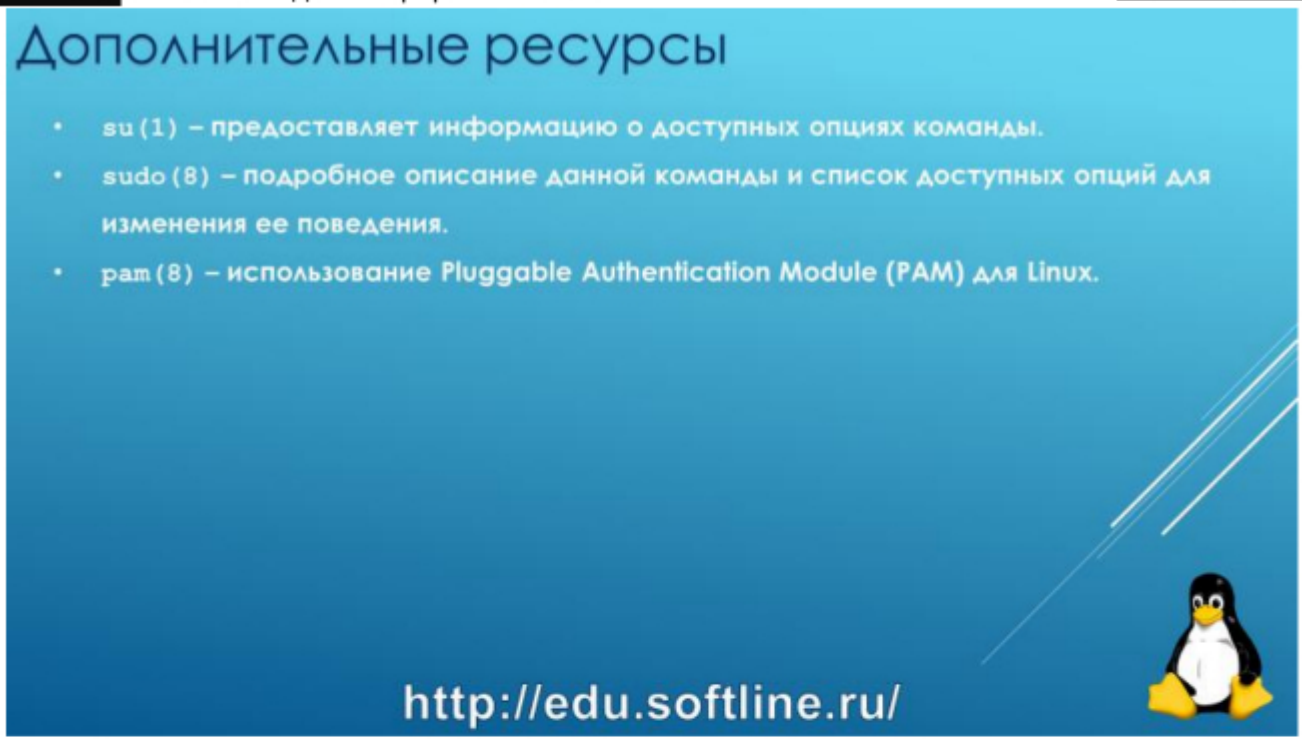
  Defaults timestamp_timeout=*значение*

  Где `значение` *– это разрешенный таймаут в минутах. Установив* `значение` *в* **0** *приведет к запросу пароля при каждом обращении к* **sudo**.

- *Если учетная запись пользователя* **sudo** *(sudoers) скомпрометирована, злоумышленник может использовать* **sudo** *для открытия новой оболочки с административными правами:*

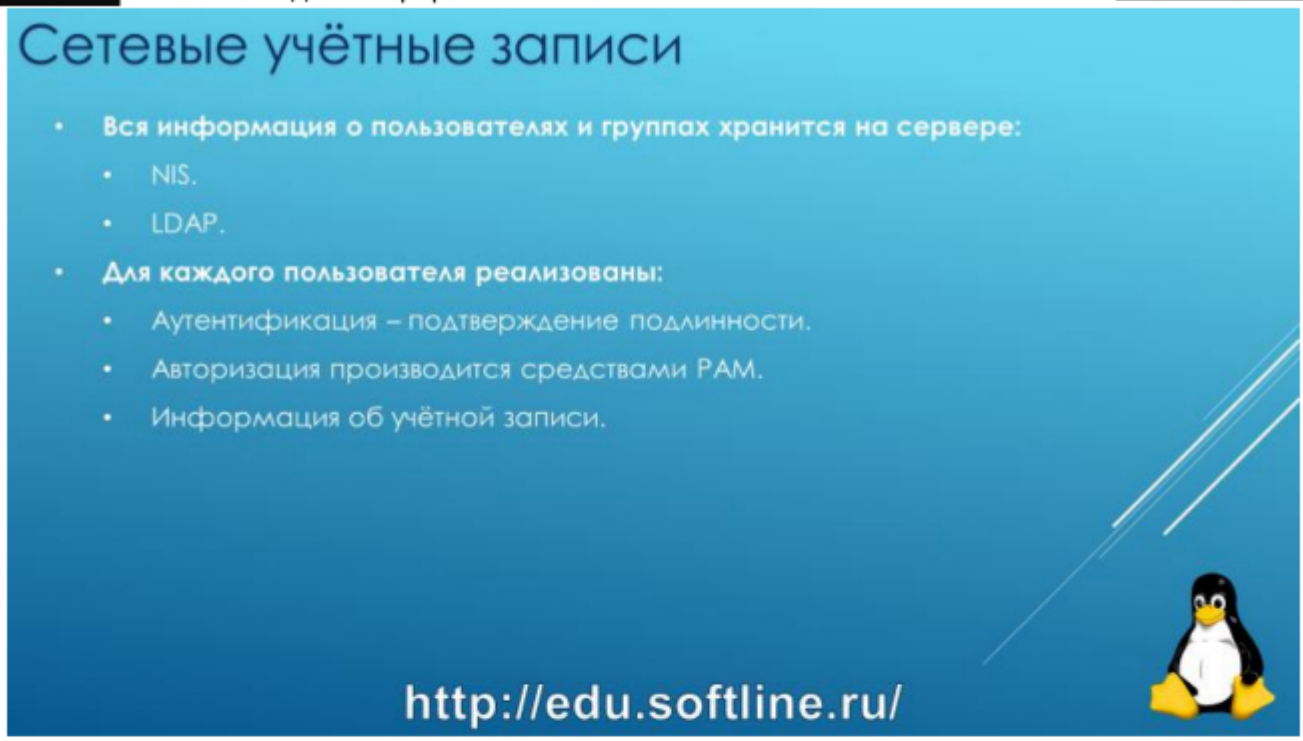
  $ sudo /bin/bash

Открытие новой оболочки от имени пользователя root или аналогичные действия дадут злоумышленнику административный доступ на неограниченное время, превышающее лимиты, прописанные в файле **/etc/sudoers**, *а также не будет запрашивать пароль для* **sudo** *до тех пор, пока открытая сессия не будет закрыта.*



Дополнительные ресурсы:

- **su(1)** – страница руководства для **su** предоставляет информацию о доступных опциях команды.
- **sudo(8)** – страница руководства для **sudo** включает подробное описание данной команды и список доступных опций для изменения ее поведения.
- **pam(8)** – страница руководства описывающая использование Pluggable Authentication Module (PAM) для Linux.

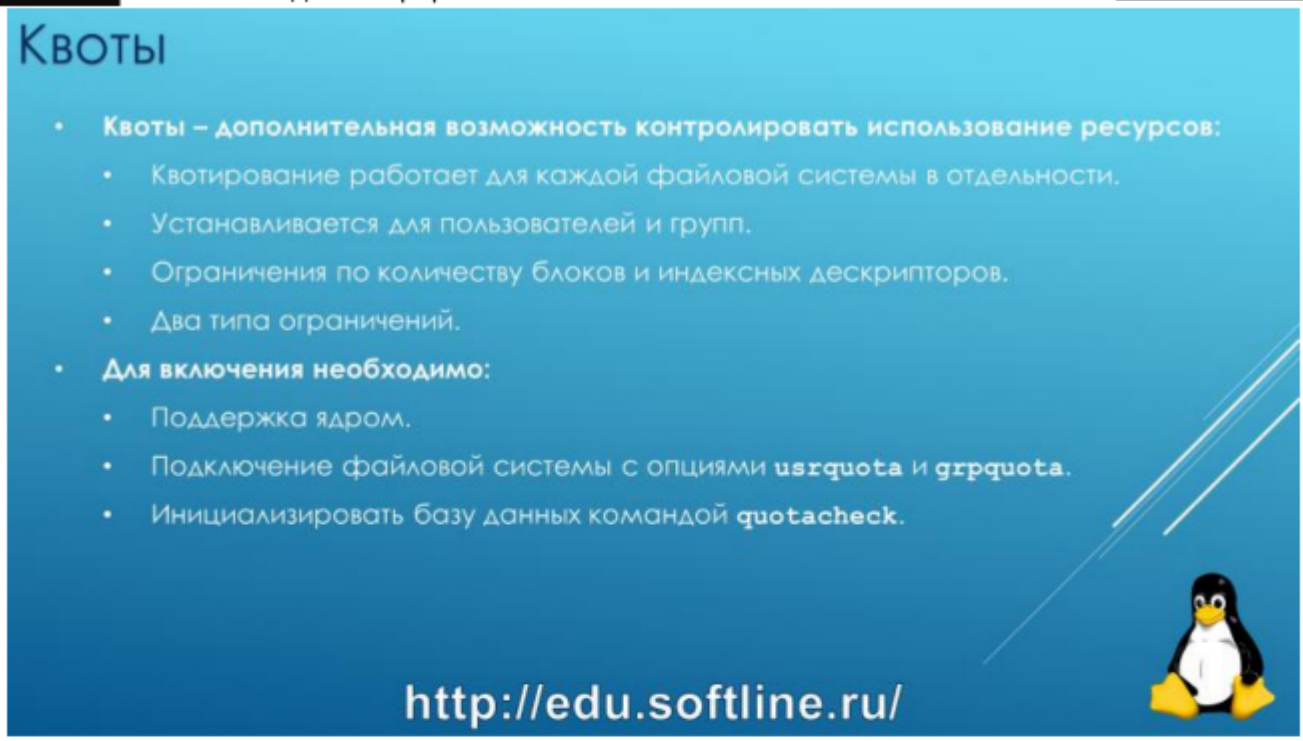


В условиях сложной топологии сети и множества серверов очень сложно управлять учётными записями на каждой системе, более того, это снижает общий уровень безопасности IT-системы. Для решения этой проблемы существуют решения, представляющие собой централизованные хранилища учётных записей пользователей и групп, все остальные серверы будут являться клиентами такого хранилища.

Сетевая учётная запись, по содержанию, ничем не отличается от локальных записей в файлах **`/etc/passwd`** и **`/etc/group`**. На сервере хранятся идентификаторы пользователей и групп, их имена, командные процессоры и домашние каталоги, информация о членстве в группах. Каждый сервер, являющийся клиентом, должен получать доступ к двум типам информации.

Аутентификация – основная процедура для подтверждения того, что связка, состоящая из имени учётной записи и пароля, является валидной. Задача хранилища дать заключение, но решение о том, предоставлять доступ к ресурсам сервера или нет, остаётся за самим сервером.

Информация о пользователях – эта информация необходима системе во многих случаях, для получения информации о членстве в группах, для получения информации о том, какой командный процессор необходимо предоставить пользователю и т.д.



Система квотирования позволяет задавать ограничение на использование дискового пространства. Для больших систем, расходование дискового пространства – головная боль, установка квот призвана решить эту проблему.

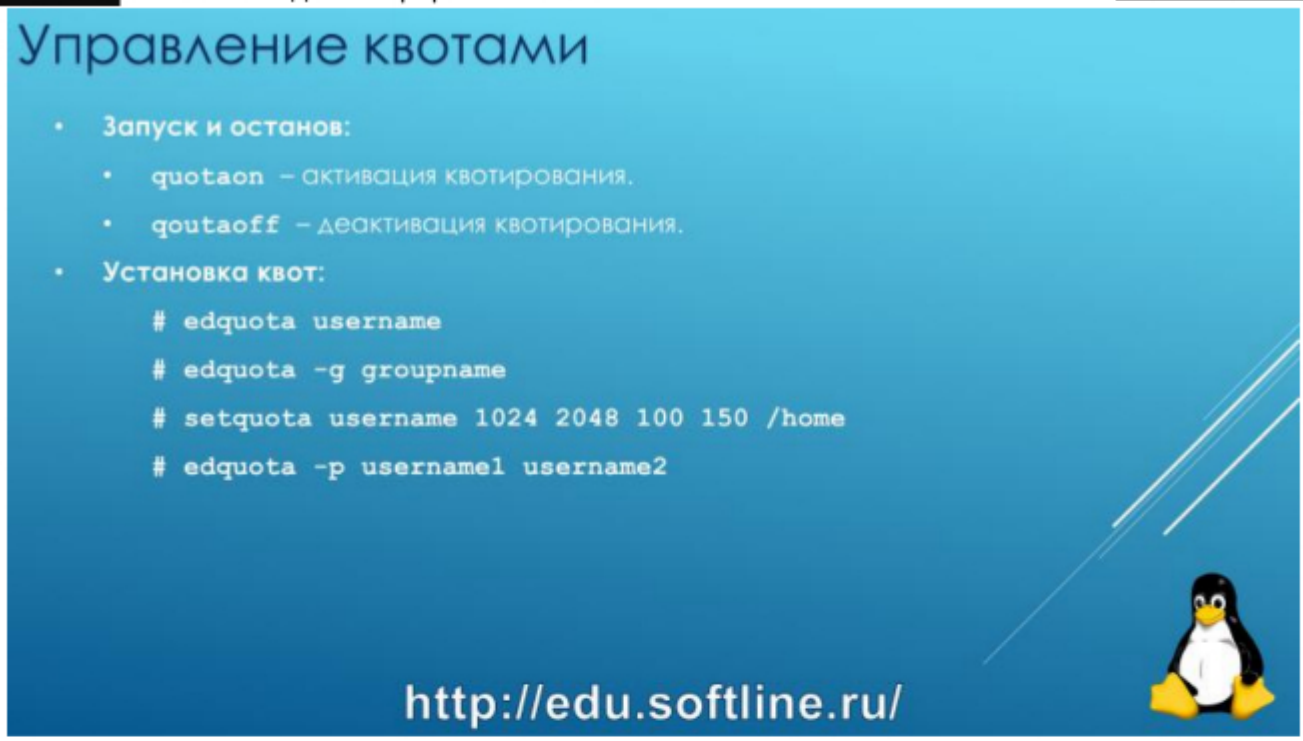
Чтобы включить квотирование необходимо подключить файловую систему с параметрами **usrquota** – квоты для пользователей, **grpquota** – квоты для групп:

```
# cat /etc/fstab | grep home
/dev/sda2      /home         ext3     defaults,usrquota,grpquota      1 2
```

После включения квот, инициализируйте базу данных командой ***quotacheck***:

```
# quotacheck -ug /dev/sda2
```

После создания базу данных, в разделе, для которого было включено квотирование, появятся файлы aquota.user и aquota.group.



Допускается активация и деактивация системы квотирования без перезагрузки операционной системы, достаточно использование всего двух команд – **quotaon** и **qoutaoff**, которые включают и выключают систему квотирования:

```
# quotaoff /dev/sdaX
# quotaon /dev/sdaX
```

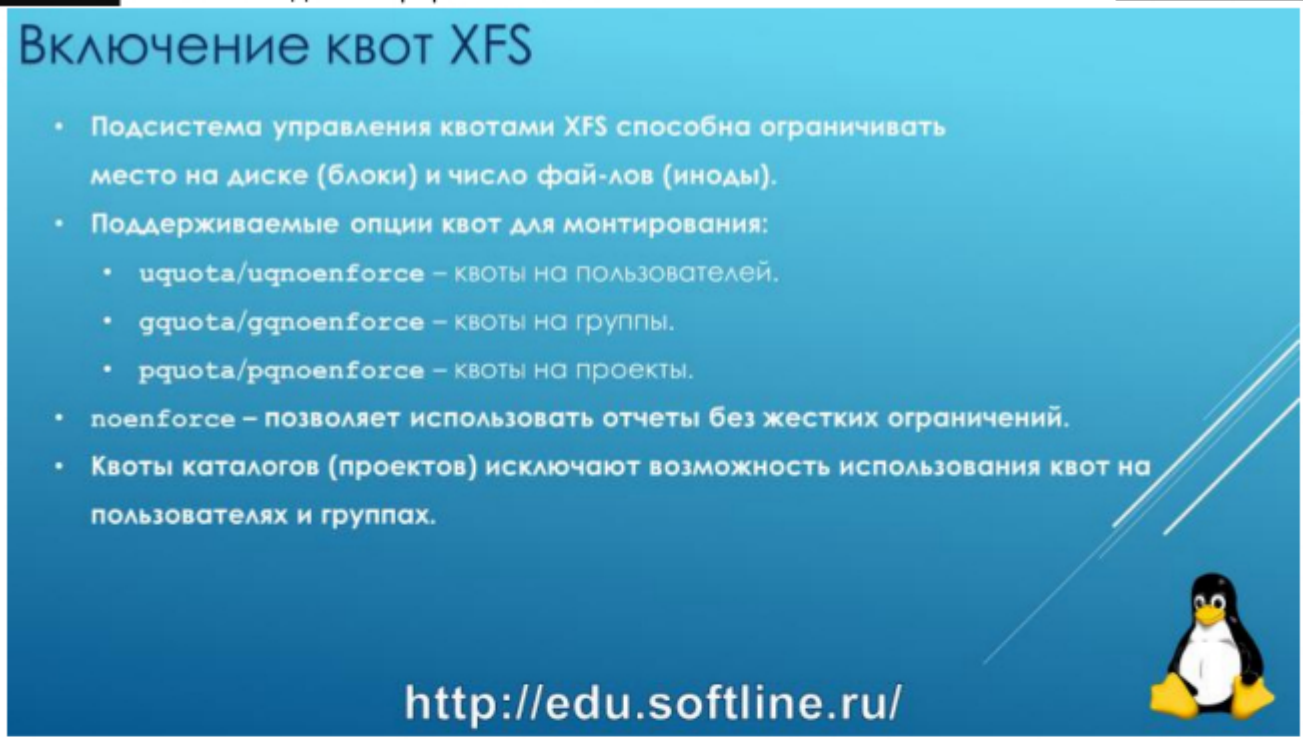
Когда система квотирования инициализирована, остаётся только назначить квоты пользователям для определённых файловых систем, это можно сделать с помощью двух команд, edquota предлагает задать квоты с помощью текстового редактора, указанного в переменной окружения `$EDITOR`, а setquota принимает значения из командной строки, что удобно, при использовании и командных сценариях:

```
# edquota username
# setquota username 1024 2048 100 150 /home
```

Обе команды принимают имя пользователя, в качестве аргумента. Программа edquota откроет текстовый редактор, указанный в переменной окружения `$EDITOR` (**vi**, при её отсутствии) с несколькими колонками. Команда **setquota** сразу принимает необходимые параметры, давайте их разберём:

- **1024** – "мягкая" квота, указывается в блоках. Квоты такого типа обозначают количество блоков, при достижении которого, пользователю будет выдано предупреждение о превышении квоты, а по истечении определённого периода пользователю будет запрещена запись данных на раздел;
- **2048** – "жёсткая" квота, указывается в блоках. При достижении количество блоков, заданного в качестве жёсткой квоты, пользователь больше не сможет записывать данные на раздел;
- **100** – "мягкая" квота применительно к индексным дескрипторам;
- **150** – "жёсткая" квота применительно к индексным дескрипторам.

Чтобы установить срок действия "мягкой квоты", используйте команду: **edquota -t**.



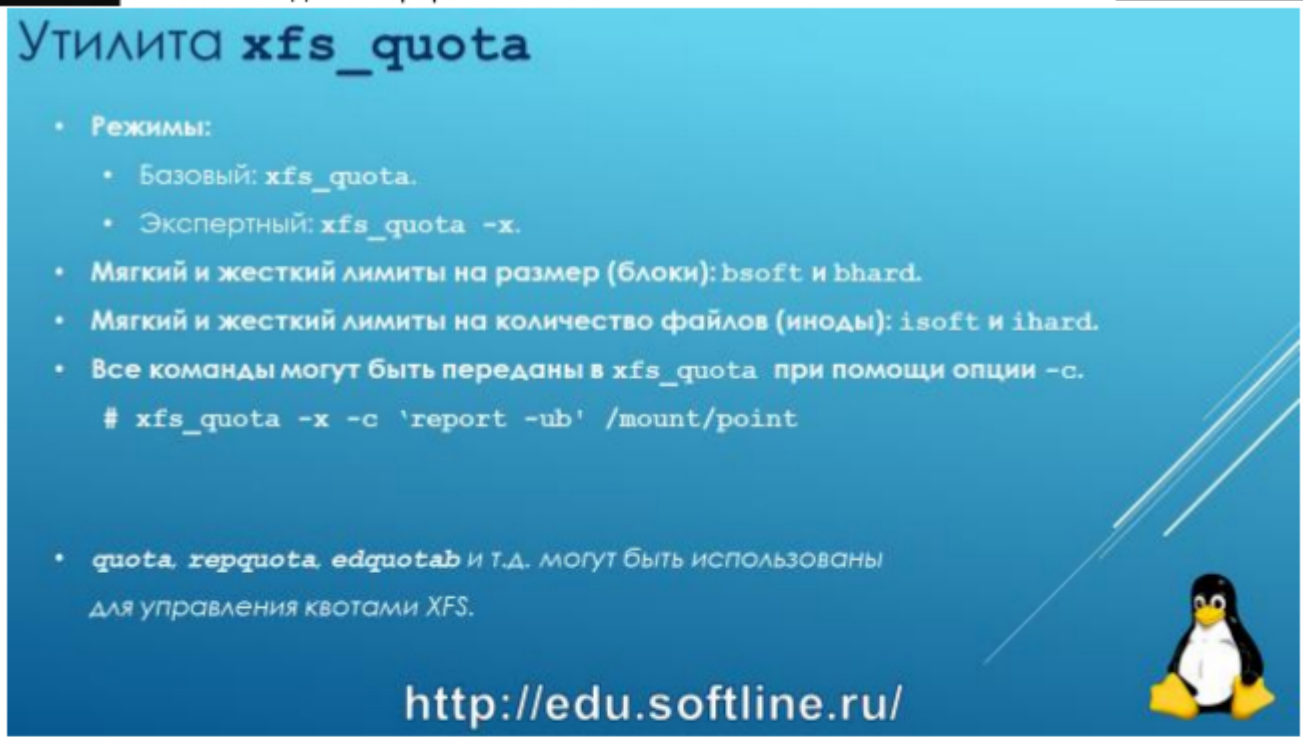
Подсистема управления квотами XFS способна ограничивать место на диске (блоки) и число файлов (иноды). Квоты XFS позволяют управлять и информировать об использовании лимитов пользователями, группами, каталогами или на уровне проектов. Учтите, что квоты на пользователей, группы, каталоги и проекты включаются отдельно друг от друга и квоты каталогов и проектов взаимоисключающие.

При управлении квотами с использованием проектов или каталогов, XFS управляет использованием диска иерархией каталогов, ассоциированной с отдельным проектом. При этом, XFS позволяет кросс-организационную группу связей между проектами. Это принципиально новый уровень, нежели управление квотами на уровне пользователей и групп.

Квоты XFS включаются во время монтирования, при помощи специальных опций. Каждая опция может быть указана как **noenforce**; это позволяет использовать отчеты без жестких ограничений.

Поддерживаемые опции квот для монтирования:

- **uquota/uqnoenforce** –Квоты на пользователей.
- **gquota/gqnoenforce** – Квоты на группы.
- **pquota/pqnoenforce** – Квоты на проекты.



После включения квот, утилита **xfs_quotas** может быть использована для установки ограничений и создания отчетов об использовании диска. По умолчанию **xfs_quotas** запускается интерактивно в базовом режиме (Basic Mode). Базовый режим использует команды для упрощения создания отчетов об использовании и доступен всем пользователям.

Базовый режим **xfs_quotas** поддерживает следующие команды:

 # quota username/userID

Отображает использование и лимиты для указанного пользователя (**username**) или идентификатора пользователя (**userID**).

 # df

Отображает свободное и использованное количество блоков и инод.

В дополнение, **xfs_quotas** поддерживает экспертный режим (Expert Mode). Команды этого режима позволяют менять ограничения и строить более детальные отчеты. Экспертный режим доступен только пользователям с повышенными привилегиями. Для интерактивного использования экспертного режима, запустите: **xfs_quotas -x**.

Экспертный режим **xfs_quotas** поддерживает следующие команды:

report /path

Отчет по квотам для указанной файловой системы.

limit

Изменение лимитов квот.

Для получения полного списка команд и справки по ним в обоих режимах используйте команду **help**.

Все команды могут быть переданы в **xfs_quota** при помощи опции **-c**, для команд из экспертного режима в паре с **-x**.

Для демонстрации отчета по квотам на каталоге **/home** (устройство: **/dev/blockdevice**), используйте следующую команду:

```
# xfs_quota -x -c 'report -h' /home.
```

Пример вывода:

```
User quota on /home (/dev/blockdevice)
                               Blocks
User ID          Used     Soft      Hard   Warn/Grace
---------- ---------------------------------
root                0      0         0     00 [------]
testuser        103.4G     0         0     00 [------]
...
```

Для установки мягкого и жесткого лимита на инод (файлов) для пользователя john (Домашний каталог: /home/john), используйте следующую команду:

```
# xfs_quota -x -c 'limit isoft=500 ihard=700 john' /home/
```

/home – точка монтирования.

По умолчанию команда **limit** задает квоты для пользователя, для задания квоты для группы используйте ключ **-g**, аналогично для проектов необходимо использовать ключ **-p**.

Мягкий и жесткий лимиты для блоков могут быть заданы при помощи параметров **bsoft** и **bhard** вместо **isoft** и **ihard**.

Для установки мягкой и жесткой квоты на блоки в размере 1000 Мб и 1200 Мб для группы **accounting** на файловую систему **/target/path** – используйте следующую команду:

```
# xfs_quota -x -c 'limit -g bsoft=1000m bhard=1200m accounting' /target/path
```



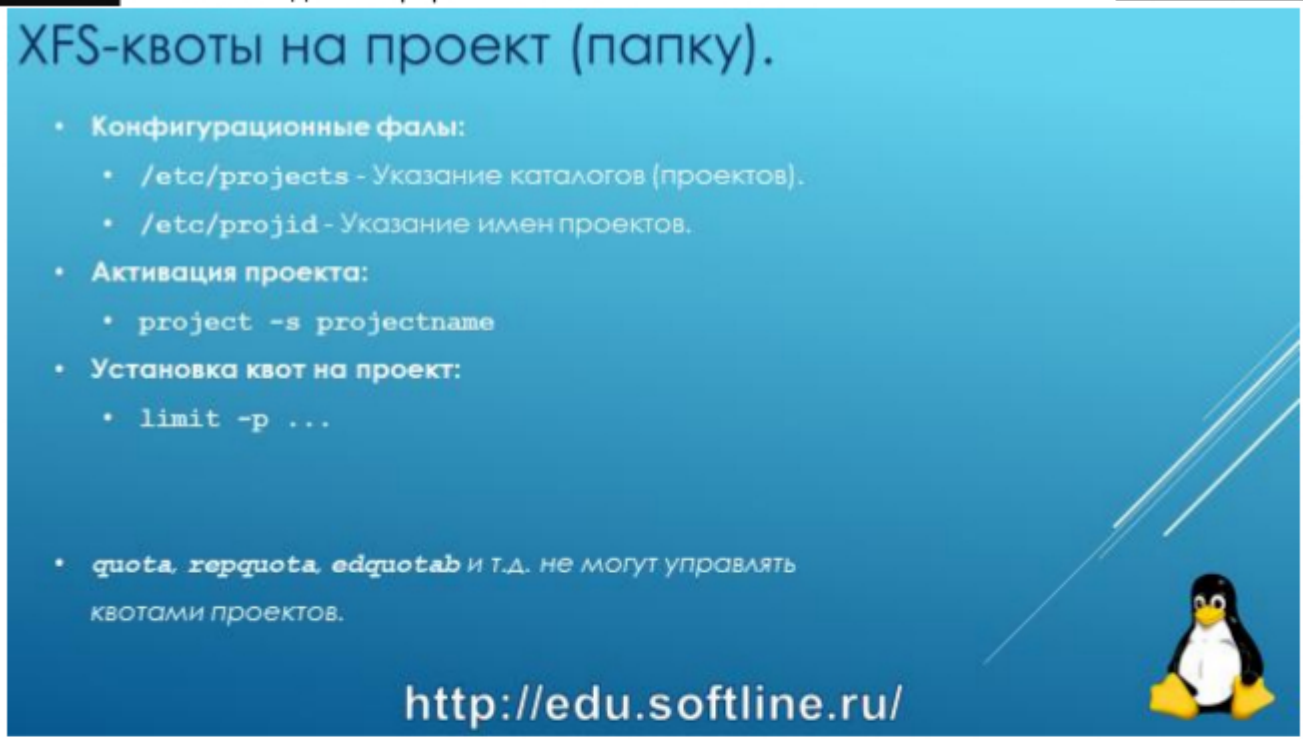
ПРИМЕЧАНИЕ:

*Параметры **bosft** и **bhard** по умолчанию задаются в байтах.*



ПРИМЕЧАНИЕ:

*Блоки реального времени (**rtbhard/rtbsoft**) описаны в **man xfs_quota** как поддерживаемые элементы для задания квоты Суб-тома реального времени не включены в релиз RHEL7 (CentOS 7), поэтому параметры **rtbhard** и **rtbsoft** не применимы.*



Установка ограничений на проект.

Перед настройкой ограничений на каталоге при помощи проекта, добавьте их сначала в **/etc/projects**. Имена проектов можно занести в файл **/etc/projected** для связи идентификаторов проектов с именами. После добавления проекта в **/etc/projects**, можно приступать к его инициализации каталога проекта:

```
# xfs_quota -x -c 'project -s projectname' project_path
```

Квоты для проекта с инициализированным каталогом могут быть сконфигурированы при помощи:

```
# xfs_quota -x -c 'limit -p bsoft=1000m bhard=1200m projectname'
```

Классические инструменты управления квотами (**quota**, **repquota**, **edquota**, ...) также могут быть использованы для управления квотами XFS, но они не могут быть использованы для управления квотами на проектах.



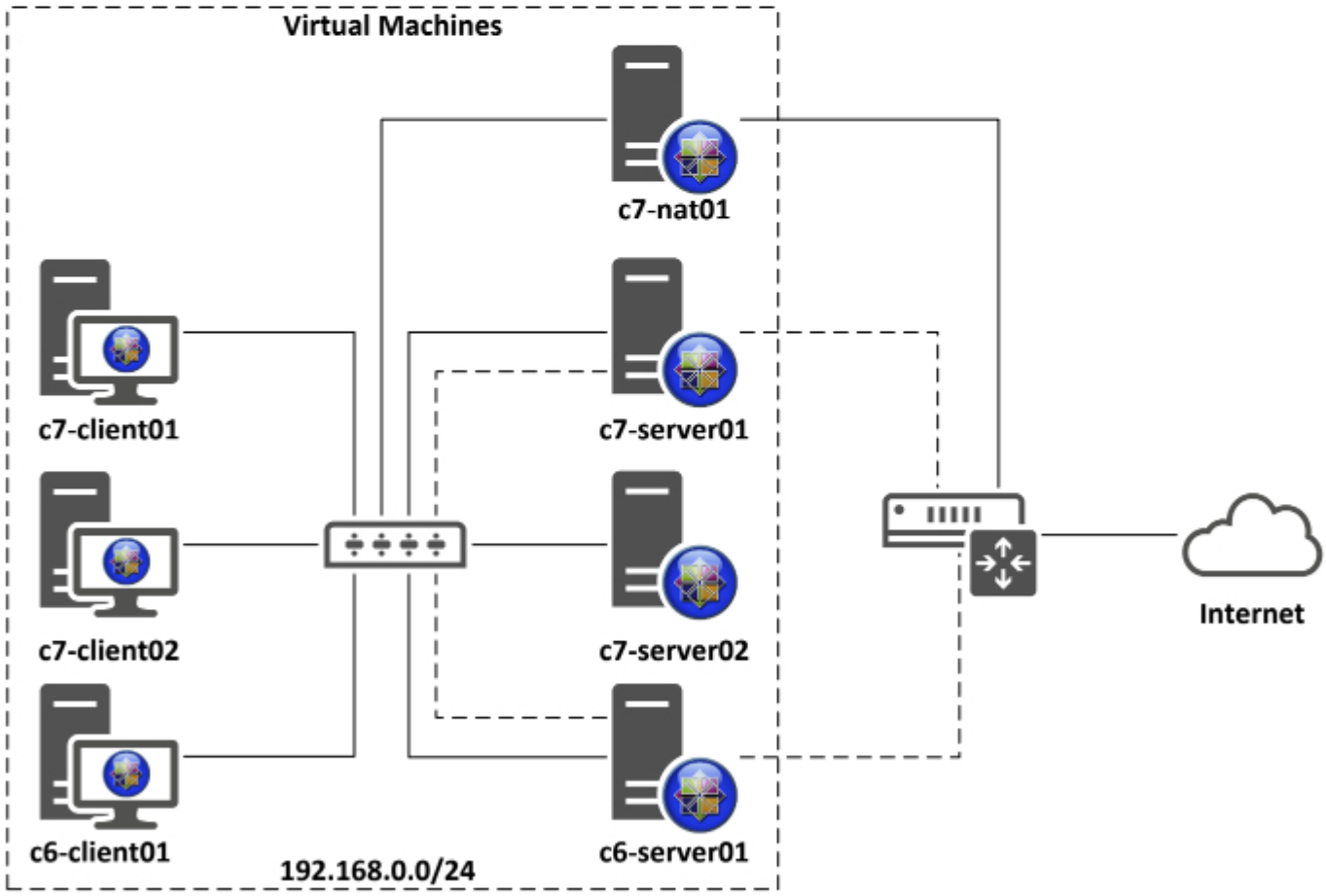
ПРИМЕЧАНИЕ:
*Red Hat рекомендует использовать **xfs_quotas**.*

За дополнительной информацией обратитесь к **man xfs_quota**, **man projid(5)** и **man projects(5)**.

Лабораторная работа 5

- **Упражнения:**
  - Управление пользователями и группами.
  - Управление квотами пользователей и групп.
  - Делегирование полномочий.
- **Необходимые машины:**
  - `c7-nat01`.
- **Учетные данные:**
  - Пользователь: `root`.
  - Пароль: `Pa$$w0rd`.

http://edu.softline.ru/





Упражнение 1 – Управление пользователями и группами.

1. Запустите виртуальную машину **c7-server01**.
2. Добавьте учетные записи пользователей **user01**, **user02** и **admin01**.
3. Задайте пользователям **user01**, **user02** и **admin01** пароль **Pa$$w0rd**.
4. Добавьте группу **sales**.
5. Добавьте пользователей **user01** и **user02** в группу **sales**.
6. Убедитесь, что пользователи **user01** и **user02** успешно добавлены в группу **sales**.
7. Добавьте пользователя **admin01** в группу **wheel**.
8. Убедитесь, что пользователь **admin01** успешно добавлен в группу **wheel**.
9. Заблокируйте пользователя **user02** и замените ему оболочку на **/dev/null**.
10. Установите срок действия учетной записи **user01** на 7 дней от текущей даты.



Ответы к упражнению 1 – Управление пользователями и группами.

1. Запустите виртуальную машину **c7-server01**.
2. Добавьте учетные записи пользователей **user01**, **user02** и **admin01**.

```
# useradd user01
# useradd user02
# useradd admin01
```

3. Задайте пользователям **user01**, **user02** и **admin01** пароль **Pa$$w0rd**.

```
# echo Pa\$\$w0rd | passwd --stdin user01
# echo Pa\$\$w0rd | passwd --stdin user02
# echo Pa\$\$w0rd | passwd --stdin admin01
```

4. Добавьте группу **sales**.

```
# groupadd sales
```

5. Добавьте пользователей **user01** и **user02** в группу **sales**.

```
# usermod -G sales user01
# usermod -G sales user02
```

6. Убедитесь, что пользователи **user01** и **user02** успешно добавлены в группу **sales**.

```
# id user01
# id user02
```

7. Добавьте пользователя **admin01** в группу **wheel**.

```
# usermod -G wheel admin01
```

8. Убедитесь, что пользователь **admin01** успешно добавлен в группу **wheel**.

```
# id admin01
```

9. Заблокируйте пользователя **user02** и замените ему оболочку на **/dev/null**.

```
# usermod -L user02
# usermod -s /dev/null user02
```

10. Установите срок действия учетной записи **user01** на 7 дней от текущей даты.

```
# chage -E $(date +%Y-%m-%d -d "+7 days") user01
```



Упражнение 2 – Управление квотами пользователей и групп.

ПРИМЕЧАНИЕ:
*Перед выполнением лабораторной работы убедитесь, что все задания упражнения «**Управление пользователями и группами**» успешно выполнены.*

11. Запустите виртуальную машину **c7-server01**.

ПРИМЕЧАНИЕ:
Если у вас есть раздел с файловой системой XFS, можно работать с ним, иначе создайте раздел с файловой системой XFS:

```
# fdisk /dev/sdb
Command (m for help): n
Partition number (1-4, default 1): 1
First sector (2048-20971519, default 2048): 2048
Last sector, +sectors or +size{K,M,G} (2048-20971519, default 20971519):
Command (m for help): w
# mkfs.xfs /dev/sdb1
# mkdir /mnt/XFS
# mount /dev/sdb1 /mnt/XFS/
```

12. Просмотрите текущие опции монтирования файловой системы XFS в каталоге **/mnt/XFS**.
13. Отмонтируйте раздел с файловой системой XFS от каталога **/mnt/XFS**.
14. Вновь смонтируйте раздел с файловой системой XFS в каталог **/mnt/XFS** с указанием опций **uquota** и **gquota**.
15. Убедитесь, что опции монтирования **uquota** и **gquota** применились.
16. Запустите расширенный интерфейс управления квотами XFS.
17. В интерфейсе управления квотами XFS выведите список доступных путей XFS.
18. Подключитесь к **/mnt/XFS**.
19. Назначьте мягкий лимит в 5 файлов и жесткий лимит в 8 для пользователя **user01**
20. Убедитесь, что лимит задан корректно.
21. Назначьте мягкий лимит в 100 Мб и жесткий лимит в 1g для группы **sales**.
22. Убедитесь, что лимит задан корректно.
23. Выйдите из расширенного интерфейса управления квотами XFS.
24. Предосьавте все права всем пользователям системы (**777**) на каталоге **/mnt/XFS**.
25. Переключитесь в контекст пользователя **user01**.
26. Попробуйте создать 10 файлов в каталоге **/mnt/XFS**.



Ответы к упражнению 2 –Управление квотами пользователей и групп.

ПРИМЕЧАНИЕ:
*Перед выполнением лабораторной работы убедитесь, что все задания упражнения «**Управление пользователями и группами**» успешно выполнены.*

1. Запустите виртуальную машину **c7-server01**.

ПРИМЕЧАНИЕ:
Если у вас есть раздел с файловой системой XFS, можно работать с ним, иначе создайте раздел с файловой системой XFS:

```
# fdisk /dev/sdb
Command (m for help): n
Partition number (1-4, default 1): 1
First sector (2048-20971519, default 2048): 2048
Last sector, +sectors or +size{K,M,G} (2048-20971519, default 20971519):
Command (m for help): w
# mkfs.xfs /dev/sdb1
# mkdir /mnt/XFS
# mount /dev/sdb1 /mnt/XFS/
```

2. Просмотрите текущие опции монтирования файловой системы XFS в каталоге **/mnt/XFS**.

```
# findmnt /mnt/XFS
```

3. Отмонтируйте раздел с файловой системой XFS от каталога **/mnt/XFS**.

```
# umount /mnt/XFS/
```

4. Вновь смонтируйте раздел с файловой системой XFS в каталог **/mnt/XFS** с указанием опций **uquota** и **gquota**.

```
# mount -o uquota,gquota /dev/sdb1 /mnt/XFS/
```

5. Убедитесь, что опции монтирования **uquota** и **gquota** применились.

```
# findmnt /mnt/XFS
```

6. Запустите расширенный интерфейс управления квотами XFS.

```
# xfs_quota -x
```

7. В интерфейсе управления квотами XFS выведите список доступных путей XFS.

```
xfs_quota> path
```

8. Подключитесь к **/mnt/XFS**.

```
xfs_quota> path 002
```

9. Назначьте мягкий лимит в 5 файлов и жесткий лимит в 8 для пользователя **user01**

```
xfs_quota> limit isoft=5 ihard=8 user01
```

10. Убедитесь, что лимит задан корректно.

```
xfs_quota> report -iu
```

11. Назначьте мягкий лимит в 100 Мб и жесткий лимит в 1g для группы **sales**.

```
xfs_quota> limit -g bsoft=500m bhard=1g sales
```

12. Убедитесь, что лимит задан корректно.

```
xfs_quota> report -bg
```

13. Выйдите из расширенного интерфейса управления квотами XFS.

```
xfs_quota> quit
```

14. Предосьавте все права всем пользователям системы (**777**) на каталоге **/mnt/XFS**.

```
# chmod 777 /mnt/XFS/
```

15. Переключитесь в контекст пользователя **user01**.

```
# su - user01
```

16. Попробуйте создать 10 файлов в каталоге **/mnt/XFS**.

```
$ cd /mnt/XFS/
$ touch file{01..10}
touch: cannot touch ‘file09’: Disk quota exceeded
touch: cannot touch ‘file10’: Disk quota exceeded
```



Упражнение 3: Делегирование полномочий.

1. Добавьте на систему **c7-server02** пользователя **admin** с паролем **Pa$$w0rd**.
2. Предоставьте пользователю **admin** право просматривать содержимое файла **/etc/shadow** при помощи команд **sudo** и **cat**.



ПРИМЕЧАНИЕ:
*Чтобы команда **visudo** запускало редактор **nano**, выполните следующую команду:*

```
# export EDITOR=nano
```

3. Убедитесь, что право предоставлено:
4. Добавьте на систему **c7-server01** группу **remote_admins**.
5. Добавьте пользователя **admin** в группу **remote_admins**.
6. Предоставьте группе **renmote_admin** право перезапускать сервис **network** при помощи команд **sudo** и **systemctl**.
7. Убедитесь, что право предоставлено:
8. Отключите предоставленные права.



Ответы к упражнению 3: Делегирование полномочий.

1. Добавьте на систему **c7-server02** пользователя **admin** с паролем **Pa$$w0rd**.

```
# useradd admin
# passwd admin
```

2. Предоставьте пользователю **admin** право просматривать содержимое файла **/etc/shadow** при помощи команд **sudo** и **cat**.

```
# visudo
…
admin ALL=/usr/bin/cat /etc/shadow
```



ПРИМЕЧАНИЕ:
*Чтобы команда **visudo** запускало редактор **nano**, выполните следующую команду:*
 # export EDITOR=nano

3. Убедитесь, что право предоставлено:

```
# su - admin
$ sudo cat /etc/shadow
```

4. Добавьте на систему **c7-server01** группу **remote_admins**.

```
# groupadd remote_dmins
```

5. Добавьте пользователя **admin** в группу **remote_admins**.

```
# groupmems -a admin -g remote_admins
```

6. Предоставьте группе **renmote_admin** право перезапускать сервис **network** при помощи команд **sudo** и **systemctl**.

```
# visudo
…
%remote_admins ALL=/usr/bin/systemctl restart network
```

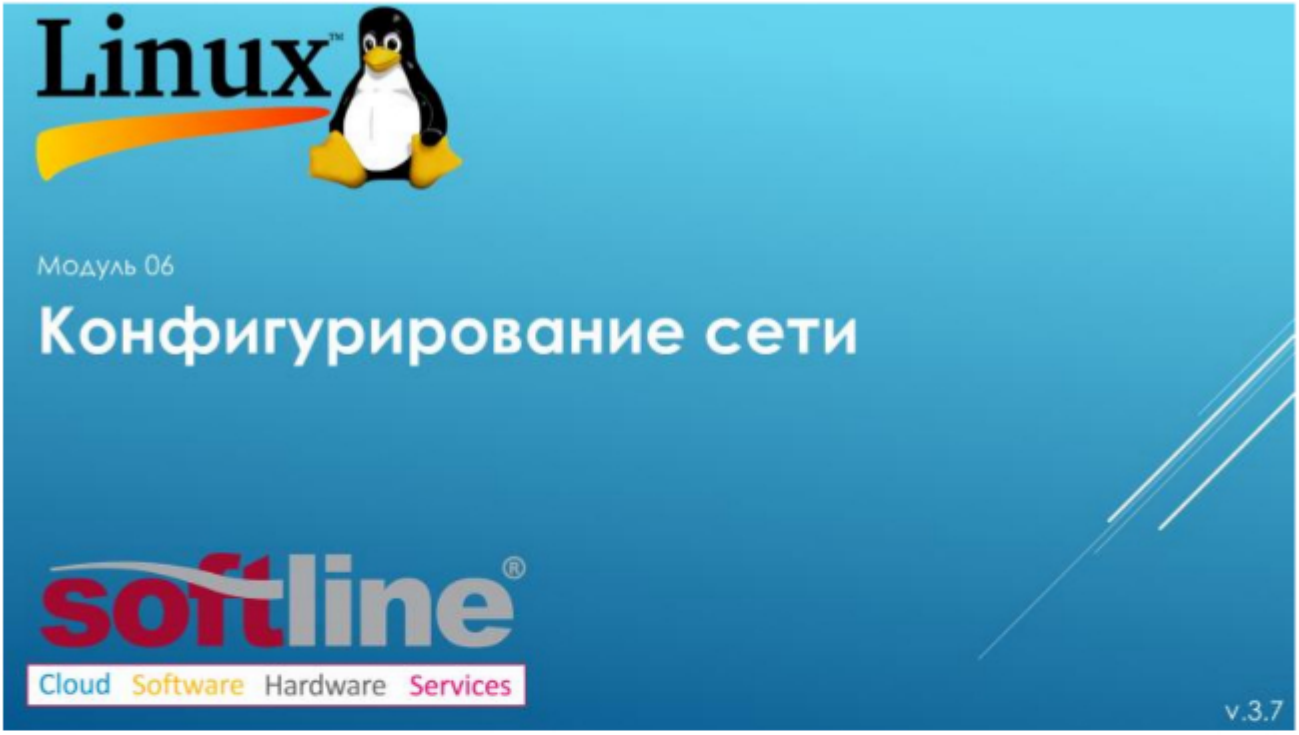
7. Убедитесь, что право предоставлено:

```
# su - admin
$ sudo systemctl restart network
```

8. Отключите предоставленные права.

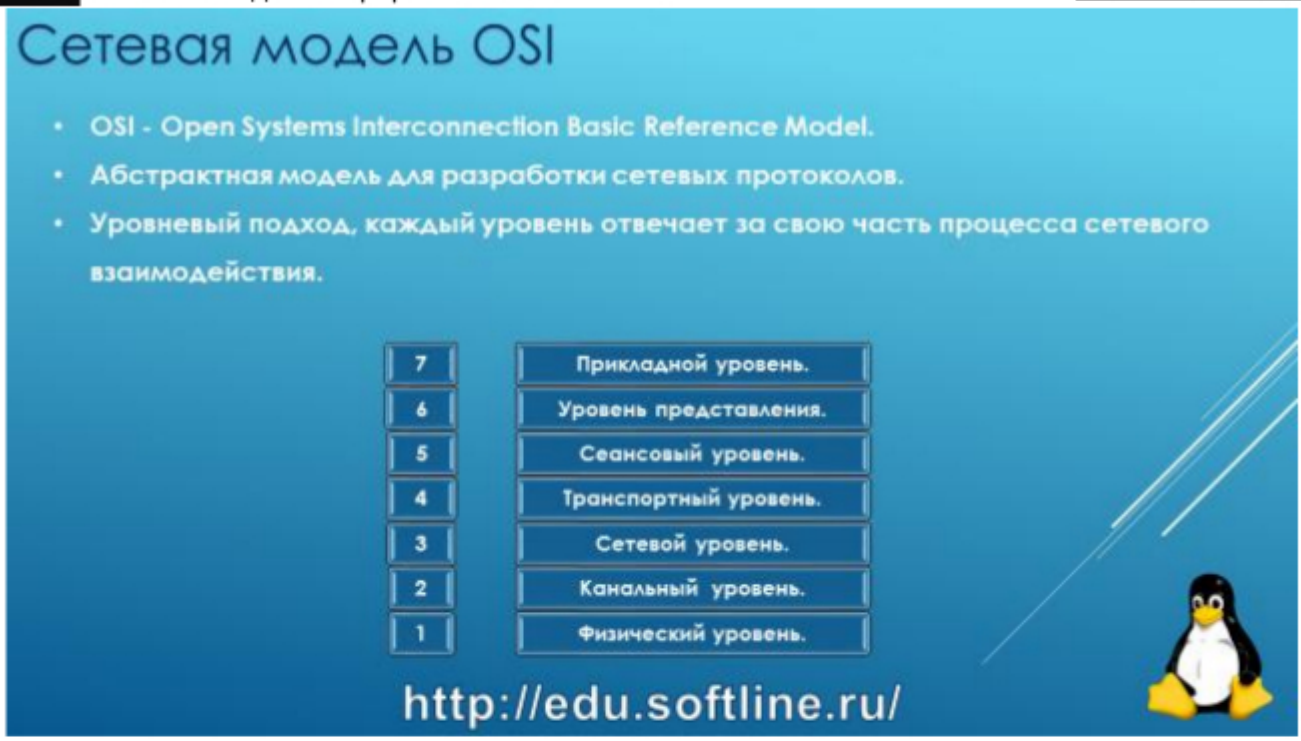
```
# visudo
…
# admin ALL=/usr/bin/cat /etc/shadow
# %remote_admins ALL=/usr/bin/systemctl restart network
```

Модуль 6: Конфигурирование сети



В модуле представлены:

- Сетевая модель OSI.
- Стек протоколов TCP/IP.
- Управление сетевыми интерфейсами.
- Таблица IP маршрутизации.
- Локальное разрешение имён.



Сетевая модель OSI (Open Systems Interconnection Reference Model-OSI – модель взаимосвязи открытых систем). Эта модель является абстрактной, предлагая уровневую модель построения сети. Такая модель позволяет более структурно подходить к разработке сетевого программного обеспечения. Несмотря на это существует мнение, что модель очень сложна в реализации и является неудачной.

Прикладной уровень (Application Layer) – уровень, который обеспечивает взаимодействие пользователя с сетевой системой. именно на этом уровне работают конечные приложения, предлагающие высокий уровень абстракции. Типичный пример приложения, работающего на 7-м уровне модели OSI – web-браузер. Обеспечивает формирование запросов к 6-му уровню.

Уровень представления (Presentation Layer) – уровень, обеспечивающий кодирование данных. Данные, полученные от приложений, преобразуются в формат для передачи по сети. Данные, полученные из сети, преобразуются в формат для передачи приложениям 7-го уровня.

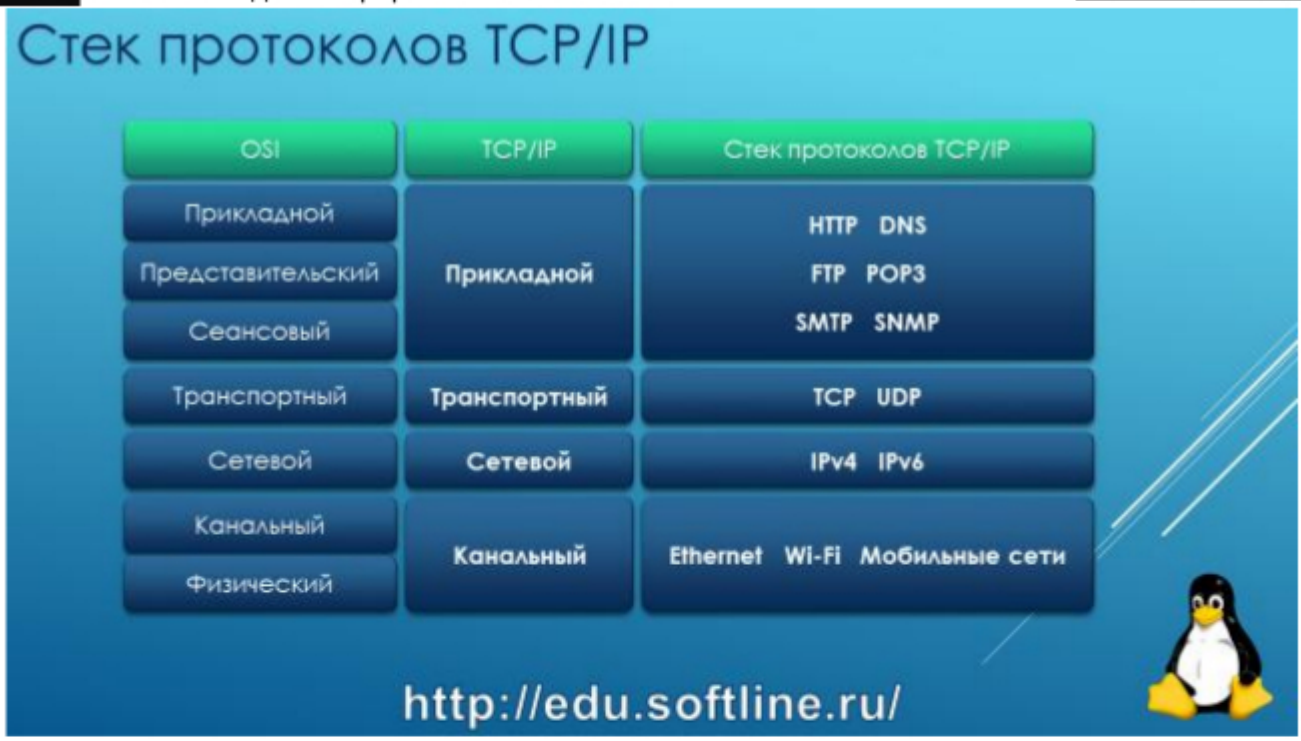
Сеансовый уровень (Session Layer) – 5-й уровень модели OSI обеспечивает поддержание сеанса связи, что позволяет устанавливать соединения на длительное время. Синхронизация обеспечивается с помощью контрольных точек.

Транспортный уровень (Transport Layer) – данный уровень обеспечивает непротиворечивость передаваемых данных. На этом уровне происходит фрагментация данных, определение последовательности и т.д.

Сетевой уровень (Network Layer) – на сетевом уровне модели OSI определяются средства передачи данных. Определение маршрута доставки пакетов, преобразование имён и логических адресов в физические адреса и многое другое.

Канальный уровень (Data Link Layer) – этот уровень является последним программным. Предназначен для обеспечения взаимодействия сетей на физическом уровне. На 2-м уровне обеспечивается определение кадров, проверка их целостности. С точки зрения программного обеспечения, канальный уровень представлен в виде драйвера сетевого устройства.

Физический уровень (Physical Layer) – нижний уровень модели OSI предназначен для непосредственной передачи данных с помощью электрических/оптических сигналов и их приёма.



Стек протоколов TCP/IP – набор сетевых протоколов передачи данных, используемых в сетях, включая сеть Интернет. Название TCP/IP происходит из двух наиважнейших протоколов семейства – Transmission Control Protocol (TCP) и Internet Protocol (IP), которые были разработаны и описаны первыми в данном стандарте.

Протоколы работают друг с другом в стеке (Stack) – это означает, что протокол, располагающийся на уровне выше, работает «поверх» нижнего, используя механизмы инкапсуляции. Например, протокол TCP работает поверх протокола IP.

Стек протоколов TCP/IP включает в себя четыре уровня:

- Прикладной уровень (Application Layer).
- Транспортный уровень (Transport Layer).
- Сетевой уровень (Internet Layer).
- Канальный уровень (Link Layer).

Протоколы этих уровней полностью реализуют функциональные возможности модели OSI. На стеке протоколов TCP/IP построено всё взаимодействие пользователей в IP-сетях. Стек является независимым от физической среды передачи данных.

Протокол IP.

Протокол IP находится не сетевом уровне сетевой модели OSI и является основой семейства протоколов TCP/IP и предназначен для обеспечения межсетевого маршрутизируемого взаимодействия. Основной особенностью является то, что протокол IP не даёт гарантии доставки данных.

На сегодняшний день основное развитие получила четвёртая версия протокола IP (IPV4). Уже в ближайшие годы стоит ожидать повсеместного внедрения протокола версии IPV6, это связано с тем, что при проектировании IPV4 не учли бурного роста сети Internet и адресное пространство стремительно заканчивается.

Протокол TCP.

TCP – один из основных протоколов сети Internet, созданный для управления передачей данных.

Основная особенность протокола TCP – работа с предварительной установкой соединения, что даёт гарантию целостности передаваемых данных. Протокол TCP обеспечивает контроль над процессом передачи информации, осуществление повторов передачи, в случае, когда часть данных была утеряна, исключение дублирования пакетов и т.д. Гарантия целостности данных ведёт за собой повышенное потребление ресурсов, а потому, в случае, когда требуется передавать данные быстро, например, голосовые, используют другие протоколы.

Протокол UDP.

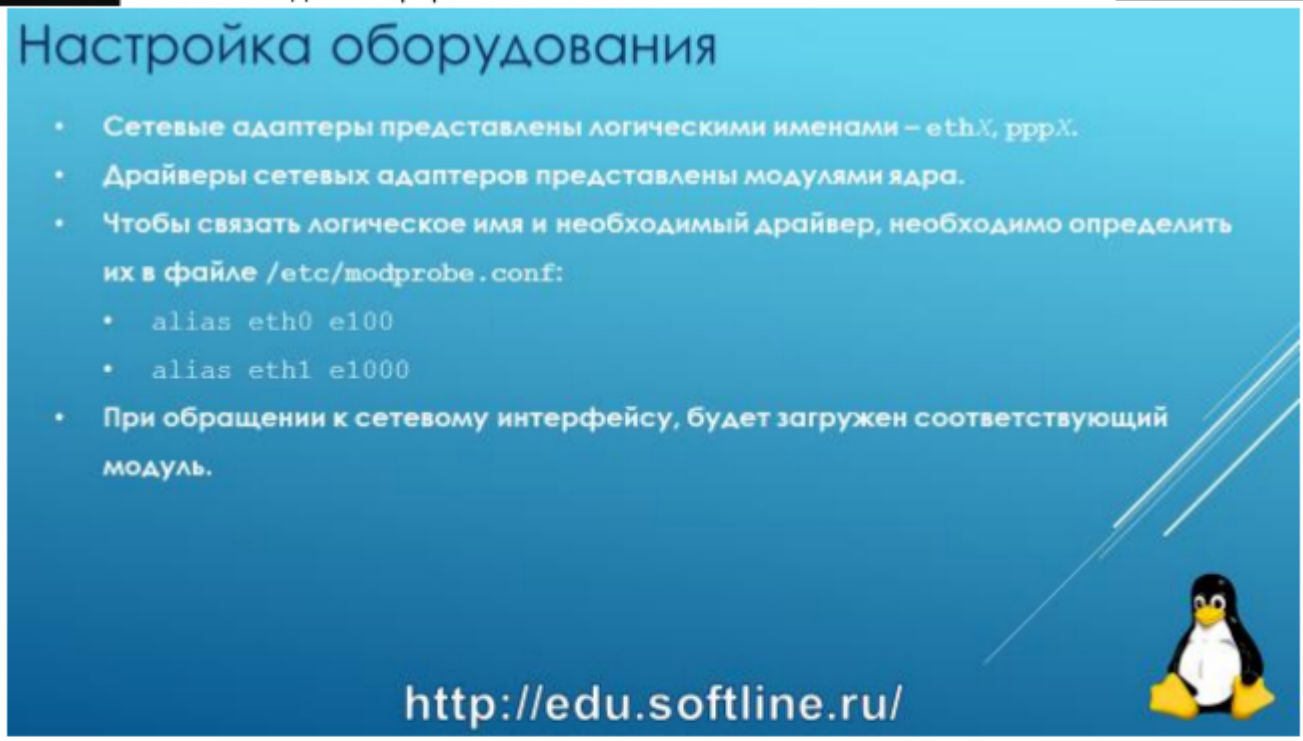
В отличие от протокола TCP, требующего установления соединения, протокол UDP просто отправляет пакеты данных по одному без предварительных договорённостей, более того, удалённый узел не должен подтверждать принятие пакета. В протоколе UDP отсутствует всяческий контроль за правильностью доставки пакета.

Такая особенность делает передачу по протоколу UDP наиболее «дешевой», но и этой дешевизне есть своя цена. В случае, когда всё же нужен контроль за доставкой пакетов, его должно обеспечить само приложение.

Протокол ICMP.

Основная функция протокола ICMP-передача сообщений об ошибках и других исключительных ситуациях, возникающих при сетевом взаимодействии. Дополнительно, ICMP может использоваться для выполнения сервисных функций.

Содержимое ICMP-пакета составляют тип сообщения, код сообщения и само сообщение, зависящее от кода и типа.



Настройка сети начинается с добавления сетевого адаптера. В современных системных платах имеются встроенные адаптеры. Лучший вариант, когда вы знаете, с каким устройством работаете, тем не менее, если вы не знаете модель сетевой карты, то на помощь вам придут команды **`lcpci`** и **`dmesg`**, вот их примерный вывод:

```
# dmesg | grep -i ethernet
eth0: Broadcom 4400 10/100BaseT Ethernet 00:17:08:44:6b:ca
#  lspci | grep -i ethernet
02:0e.0 Ethernet controller: Broadcom Corporation BCM4401-B0 100Base-TX (rev 02)
```

Стоит отметить, что сетевые устройства являются виртуальными и не имеют представления в виртуальной файловой системе **`/dev`**. Каждый тип сетевого устройства имеет своё виртуальное имя:

- Ethernet адаптеры — `eth[0-9]`;
- PPP — `ppp[0-9]`;
- Token Ring - `tr[0-9]`.

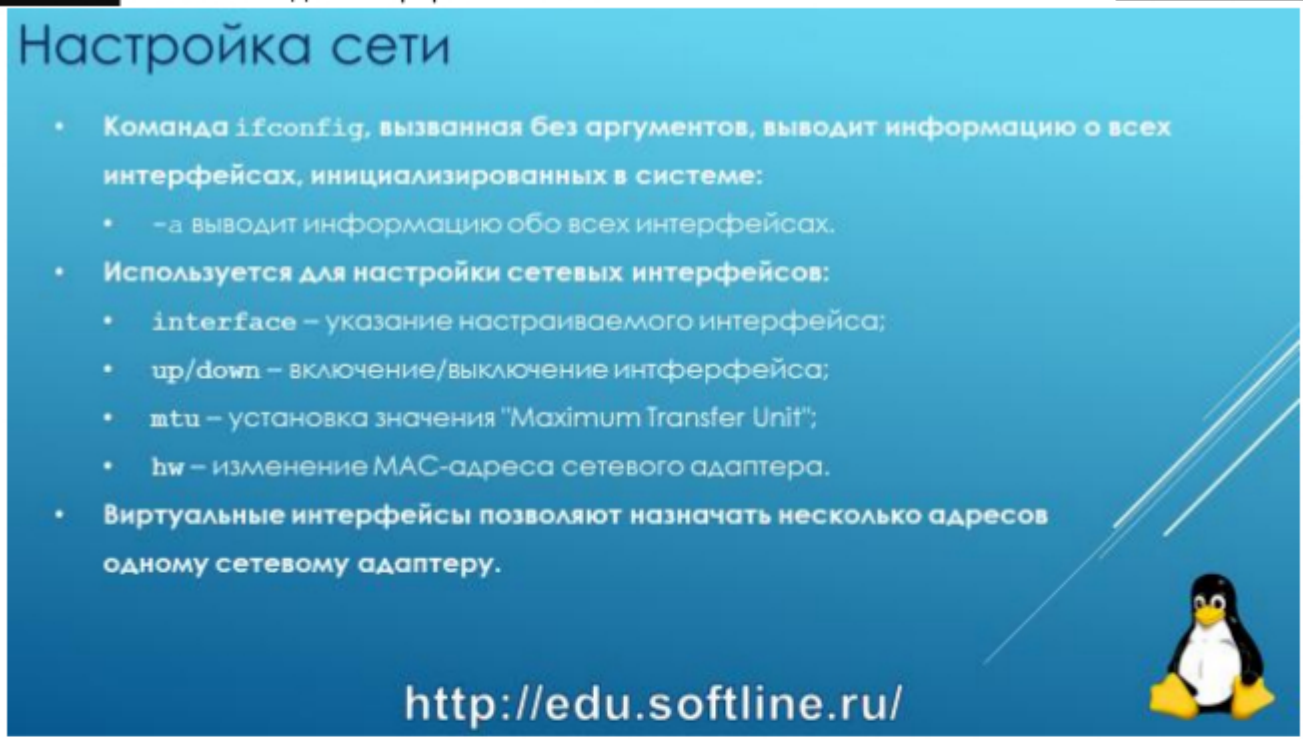
После того как вы определили модель сетевого адаптера, необходимо найти соответствующий модуль ядра (драйвер). Это можно сделать с помощью команды `modinfo`:

```
# modinfo /lib/modules/2.6.32-279.el6.i686/kernel/drivers/net/b44.ko
filename:       /lib/modules/2.6.32-
279.el6.i686/kernel/drivers/net/b44.ko
version:        2.0
license:        GPL
description:    Broadcom 4400 10/100 PCI ethernet driver
author:         Florian Schirmer, Pekka Pietikainen, David S. Miller
```

Подведём итоги — вам известна модель сетевой карты, известно месторасположение драйвера. Вы знаете, необходимые сетевые реквизиты. Остаётся только сопоставить драйвер устройства сетевому адаптеру и инициировать сетевой интерфейс. Это достигается с помощью редактирования файла **`/etc/modprobe.conf`**.

```
alias      eth0 b44
```

Эта строка говорит ядру о том, что при обращении к устройству **eth0** необходимо загрузить модуль **b44**.



Программа **ifconfig** — основное и очень мощное средство для настройки сетевых интерфейсов. Простой вызов позволяет получить информацию обо всех запущенных сетевых интерфейсов:

```
# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:17:08:44:6B:CA
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:169
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:2365 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2365 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:3662954 (3.4 MiB)  TX bytes:3662954 (3.4 MiB)
```

Основной аргумент, принимаемый командой **ifconfig**, имя сетевого интерфейса. Если это единственный аргумент, то будет выведена информация по указанному интерфейсу. Чтобы определить сетевые реквизиты для интерфейса, необходимо использовать следующую конструкцию:

```
# ifconfig eth0 10.10.10.1 netmask 255.255.255.0
# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:17:08:44:6B:CA
          inet addr:10.10.10.1  Bcast:10.10.10.255  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:169
```

Отключение и включение интерфейса, необходимо использовать параметры команды **ifconfig** — *up* и *down*:

```
# ifconfig eth0 down
# ifconfig eth0 up
```

Также, **ifconfig** предоставляет администратору возможность изменять MAC-адрес сетевого адаптера, для этого необходимо указать с помощью параметра **hw** тип сетевого интерфейса и новый адрес:

```
# ifconfig eth0 down
# ifconfig eth0 hw ether 00:01:02:03:04:05
# ifconfig eth0 up
# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:01:02:03:04:05
          inet addr:10.10.10.1  Bcast:10.10.10.255  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:169
```

Создание дополнительных виртуальных интерфейсов необходимо в случае, когда для одного сетевого адаптера требуется назначить несколько ip-адресов:

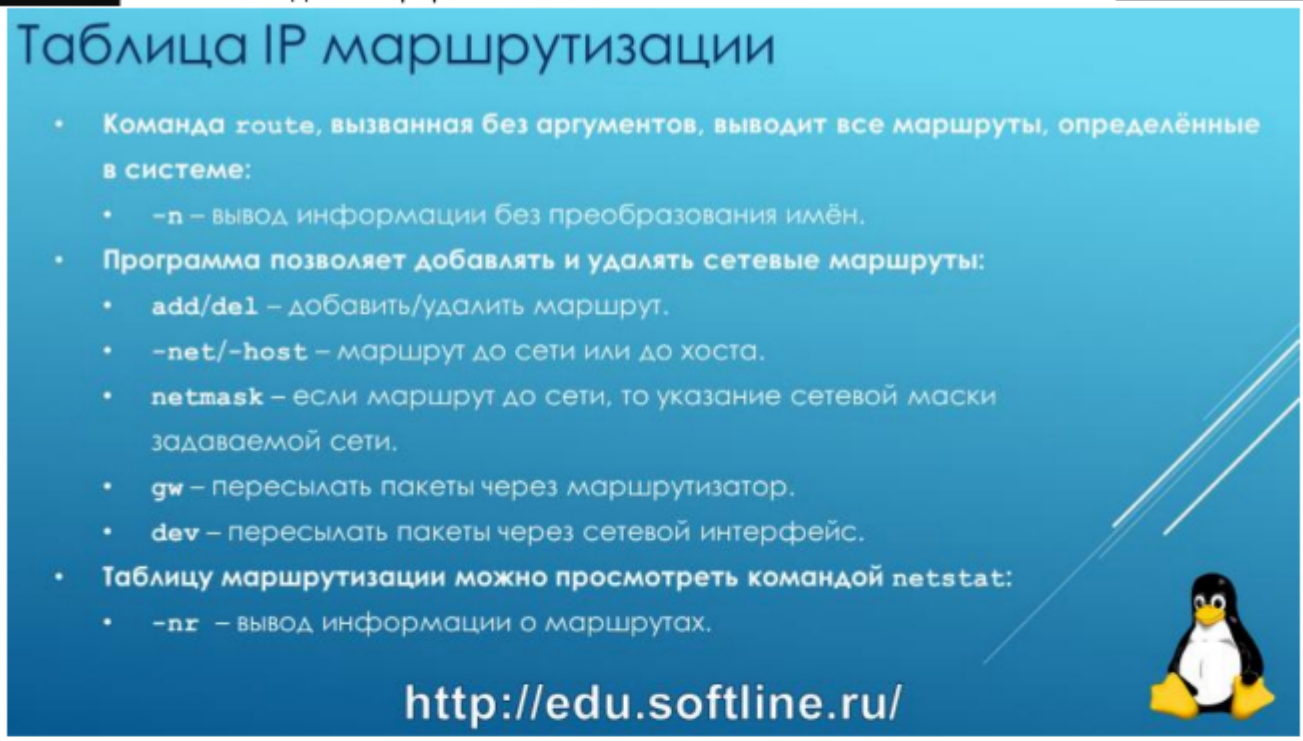
```
# ifconfig eth0:0 10.10.10.1 up
```

Каждый дистрибутив предлагает средства сохранения настроек для последующего применения при перезагрузках. В рассматриваемом дистрибутиве CentOS конфигурационные файлы, определяющие сетевые настройки интерфейсов, находятся в каталоге **/etc/sysconfig/networking-scripts** и называются **ifcfg-ethX**, где **X**-номер интерфейса:

```
# ls /etc/sysconfig/network-scripts/ifcfg-*
/etc/sysconfig/network-scripts/ifcfg-eth0
/etc/sysconfig/network-scripts/ifcfg-lo
```

Каждый файл содержит несколько ключевых параметров:

- `DEVICE=eth0` — имя интерфейса.
- `BOOTPROTO=static` — способ получения реквизитов `dhcp/static`.
- `HWADDR=00:17:08:44:6B:CA` — MAC-адрес устройства.
- `IPADDR=10.10.10.6` — IP-адрес устройства.
- `ONBOOT=yes` — необходимость инициализации при загрузке системы.



Маршрутизация — основа работы сети, именно благодаря маршрутизаторам множество изолированных сетей могут общаться между собой. Изначально, сетевой узел "видит" только те узлы, которые находятся в одной сети.

Статической маршрутизацией управляет программа **route**, позволяющая просматривать и добавлять новые маршруты. Вызов команды без аргументов приводит к выводу информации о текущих маршрутах:

```
# route -n
10.10.10.0      0.0.0.0         255.255.255.0   U     0      0        0 eth1
0.0.0.0         10.10.10.1      0.0.0.0         UG    0      0        0 eth1
```

В стандартном случае каждый узел сети имеет маршрут «по умолчанию». Этот маршрут определяет маршрутизатор, на который необходимо посылать все пакеты, для которых не заданы иные правила. Добавление такого маршрута производится командой:

```
# route add default gw 10.10.10.1
```

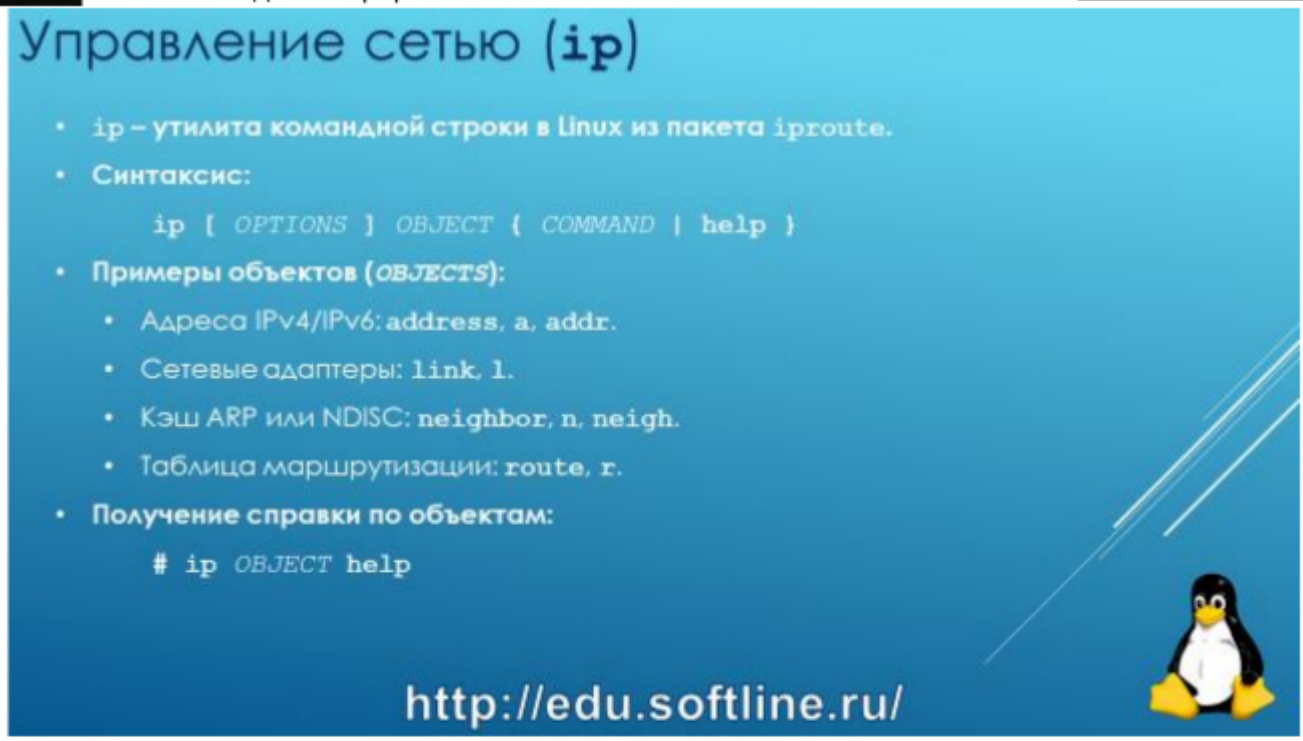
В более-менее крупных сетях часто возникает необходимость добавления дополнительных маршрутов до каких-либо сетей/узлов, находящихся вне сети, которой принадлежит данный узел. Существует два вида маршрутов: до сети и до узла. Обратите внимание на то, что маршрутизация оперирует понятием "ДО". Чтобы добавить маршрут до сети/узла необходимо воспользоваться командой:

```
# route add -net 10.10.20.0 netmask 255.255.255.0 gw 10.10.10.5
# route add -host 10.10.30.5 dev eth3
```

Программа ***netstat*** также показывает информацию о текущих маршрутах, что бывает полезно, например в ОС FreeBSD, где команда route не предназначена для вывода таблицы маршрутизации:

```
# netstat -nr
10.10.10.0      0.0.0.0         255.255.255.0   U     0      0        0 eth1
```

```
0.0.0.0         10.10.10.1      0.0.0.0         UG    0      0        0 eth1
```



ip – утилита командной строки в Linux из пакета **iproute**. Позволяет выполнять настройку сетевой подсистемы и является заменой таких утилит, как **ifconfig**, **route**, **arp**.

ПРИМЕЧАНИЕ:
В состав CentOS 7 (RHEL 7) входит ***iproute-3*** *(CentOS 7.2.1511 –* ***iproute-3.10.0-54.el7.x86_64****).*
В состав CentOS 6 (RHEL 6) входит ***iproute-2*** *(CentOS 6.8 –* ***iproute-2.6.32-54.el6.x86_64****).*

Синтаксис:

```
ip [ OPTIONS ] OBJECT { COMMAND | help }
```

Список объектов команды **ip**, за полным списком обратитесь к документации:

Объект	Аббревиатура	Команды	Описание
address	**a, addr**	add, del, show, flush	Адреса IPv4/IPv6.
link	**l**	set, show	Сетевые адаптеры.
neighbor	**n, neigh**	add, del, change, replace, show, flush	Кэш ARP или NDISC
route	**r**	list, add, del, flush	Таблица маршрутизации

Получение справки по объектам:

```
# ip address help
# ip r help
```

Примеры использования команды ip.

Отображение информации по всем сетевым интерфейсам:

```
# ip a
```

Отображение только IPv4:

```
# ip -4 a
```

Отображение только IPv6:

```
# ip -6 a
```

Отображение информации по выбранным интерфесам:

```
# ip a show eth0
# ip a list eth0
# ip a show dev eth0
```

Назначение IP-адреса на интерфейс:

```
# ip a add 192.168.0.1/255.255.255.0 dev eth0
# ip a add 192.168.0.1/24 dev eth0
```

Удаление IP-адреса с интерфеса:

```
# ip a del 192.168.1.200/255.255.255.0 dev eth0
# ip a del 192.168.1.200/24 dev eth0
```

Включение и отключение интерфейса:

```
# ip link set dev eth0 down
# ip link set dev eth0 up
```

Отображение ARP-кэша:

```
# ip n show
192.168.1.1 dev eth0 lladr 00:15:5d:46:8b:0a STALE
…
```

Последнее поле отображает состояние записи, возможные состояния:

- **STALE** – запись корректная, но будет проверена перед первой передачей.
- **DELAY** – пакет был отправлен и ожидается подтверждение.
- **REACHABLE** – запись корректна.



ПРИМЕЧАНИЕ:

В некоторых сценариях устранения неисправностей может пригодиться возможность внесения изменений в записи ARP-кэша:

Добавление записи:

```
ip neigh add 192.168.1.5 lladdr 00:1a:30:38:a8:00 dev eth0 nud perm
```

Удаление записи:

```
ip neigh del 192.168.1.5 dev eth1
```

Изменение состояния записис:

```
ip neigh chg 192.168.1.100 dev eth1 nud reachable
```

Очистка ARP-кэша:

```
ip -s -s n f 192.168.1.5
ip -s -s n flush 192.168.1.5
```

Отображение таблицы маршрутизации:

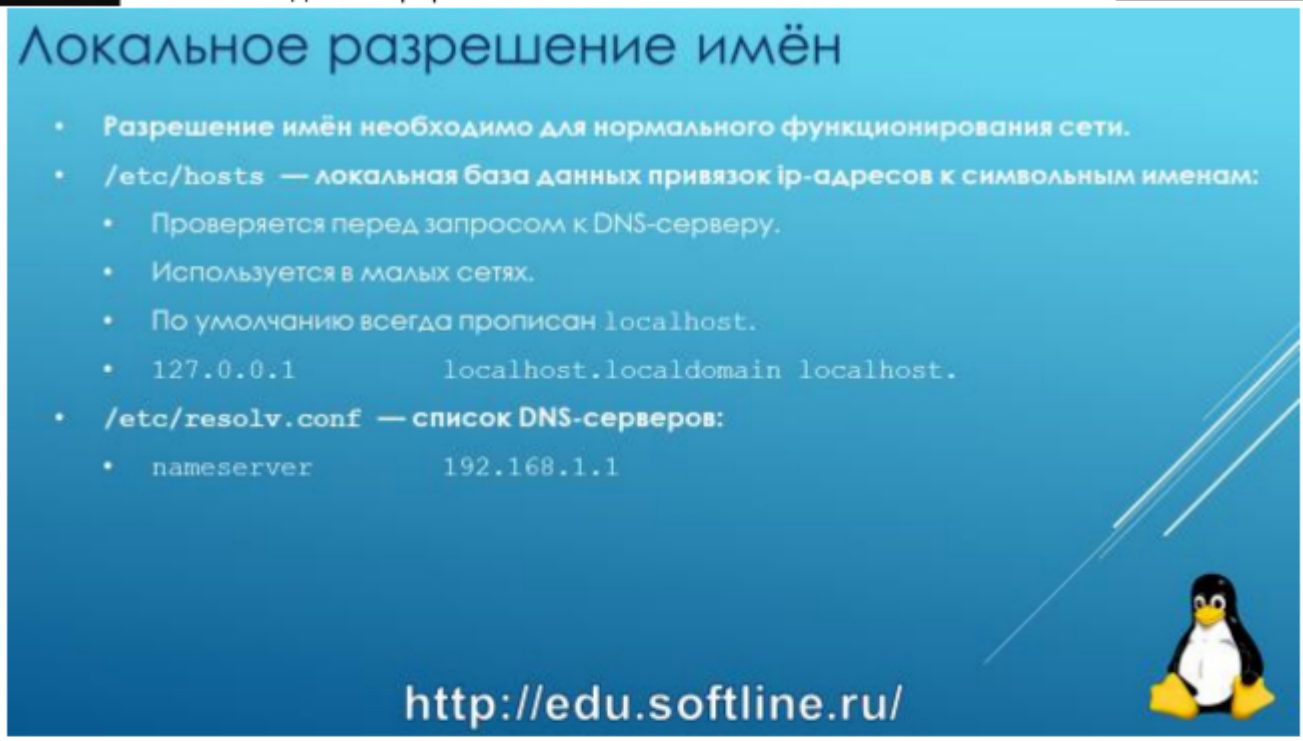
```
# ip r
# ip r list
# ip r list 192.168.10.0/24
```

Добавление нового маршрута:

```
# ip route add 192.168.10.0/24 via 192.168.1.2
```

Удаление маршрута:

```
# ip route del 192.168.10.0/24 dev eth0
```



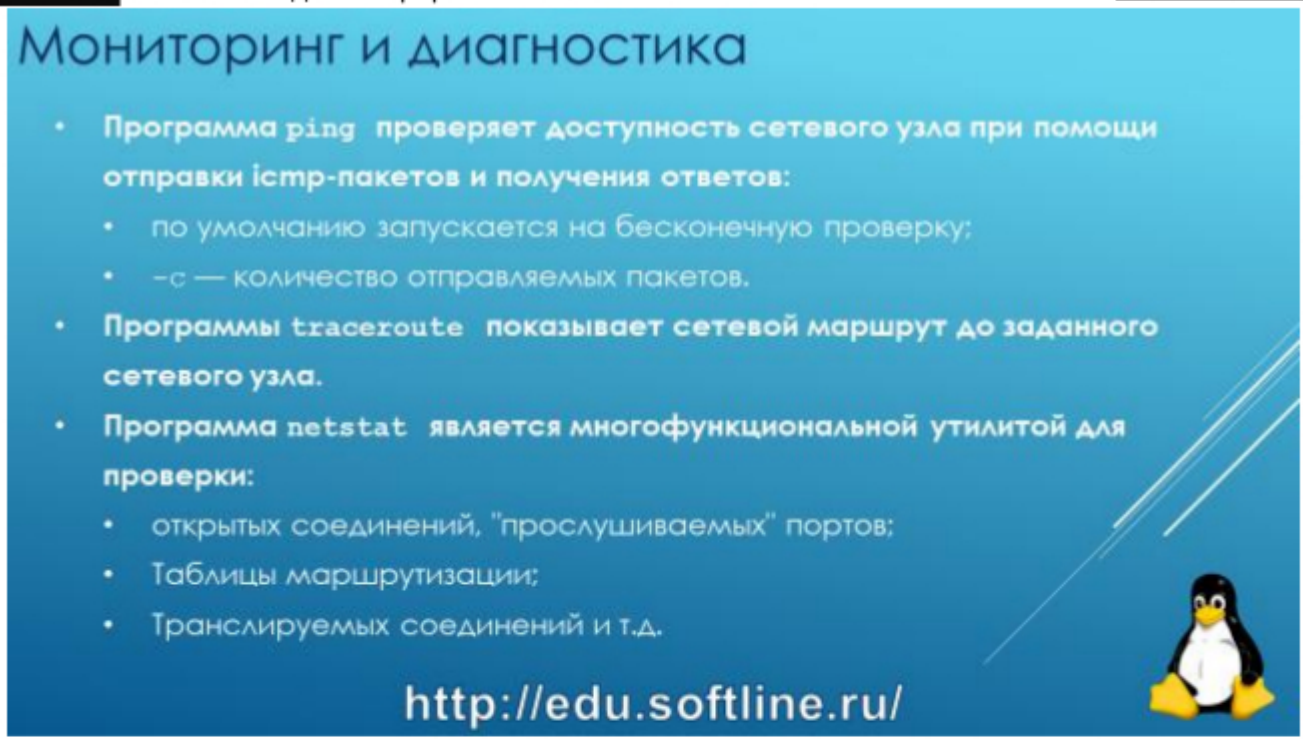
Система разрешения имён сделала работу в сети удобной и массовой. Без разрешения имён не обходится практически ни один запрос.

Как и в операционной системе MS Windows, Linux имеет файл **/etc/hosts**, в котором указывается соответствие имён ip-адресам. Естественно, что использование такого файла в качестве основной системы разрешения имён вызывает огромные сложности, именно поэтому файл **/etc/hosts** используется только для разрешения имени localhost:

```
# cat /etc/hosts
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1       tuxnote localhost.localdomain   localhost
```

Учитывая, что ОС Linux является системой, очень сильно ориентированной на сетевую работу, то можно смело предложить, что она умеет работать и с сервером имён, называемым DNS (Domain Name System), чтобы указать системе адрес сервера имён, необходимо внести изменения в файл **/etc/resolv.conf**:

```
# cat /etc/resolv.conf
nameserver 10.10.10.10
nameserver 192.168.1.2
```



Стандартные средства диагностики работоспособности сети используются администраторами очень часто. К стандартным средствам можно отнести не один десяток программ, но мы остановимся на 3-х основных, с помощью которых вы всегда сможете установить факт сбоя в сети, чтобы в дальнейшем прибегнуть к соответствующим способам восстановления работоспособности.

Команда **ping** используется для проверки доступности сетевого узла. Программа формирует специальный icmp-пакет, называемый **ECHO_REQUEST** и отправляет его удалённому узлу, ожидая ответ, называемый **ECHO_RESPONSE**. В случае, если ответ не пришёл, то можно сделать вывод о недоступности сетевого узла, НО администратор всегда должен помнить, что неполучение ответа говорит лишь о том, что какой-то из пакетов не дошёл до адресата, а это может быть вызвано правилами брандмауэра в любой точке на пути следования пакета.

По умолчанию, команда ping работает бесконечно (ожидая сигнала к завершению, данного пользователем):

```
# ping 10.10.10.1
PING 10.10.10.1 (10.10.10.1) 56(84) bytes of data.
64 bytes from 10.10.10.1: icmp_seq=1 ttl=64 time=1.81 ms
64 bytes from 10.10.10.1: icmp_seq=2 ttl=64 time=1.03 ms
64 bytes from 10.10.10.1: icmp_seq=3 ttl=64 time=1.04 ms
--- 10.10.10.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 1.036/1.298/1.813/0.366 ms
```

Программа принимает разнообразные аргументы:

- **-b** — позволяет "пинговать" широковещательные адреса;
- **-c** — сколько пакетов необходимо отправить;
- **-q** — тихий режим, будет выведена только итоговая информация.

Программа **traceroute** демонстрирует администратору маршрут до удалённого узла, это необходимо для проверки таблицы маршрутизации и определения проблем. Используются различные сетевые протоколы, но принцип действия один — посылается tcp-пакет с установленным значением **TTL=1**. Первый же маршрутизатор, получивший такой пакет, отправит ответ, что время жизни пакета истекло, после получения такого ответа, traceroute формирует новый пакет со значением **TTL=2** и так далее, вплоть до узла назначения.

Программа **netstat** — многофункциональный инструмент для просмотра состояния сетевого стека операционной системы, позволяя отслеживать информацию о сетевых интерфейсах, таблице маршрутизации, статистике интерфейса, информацию о транслируемых адресах и т.д.

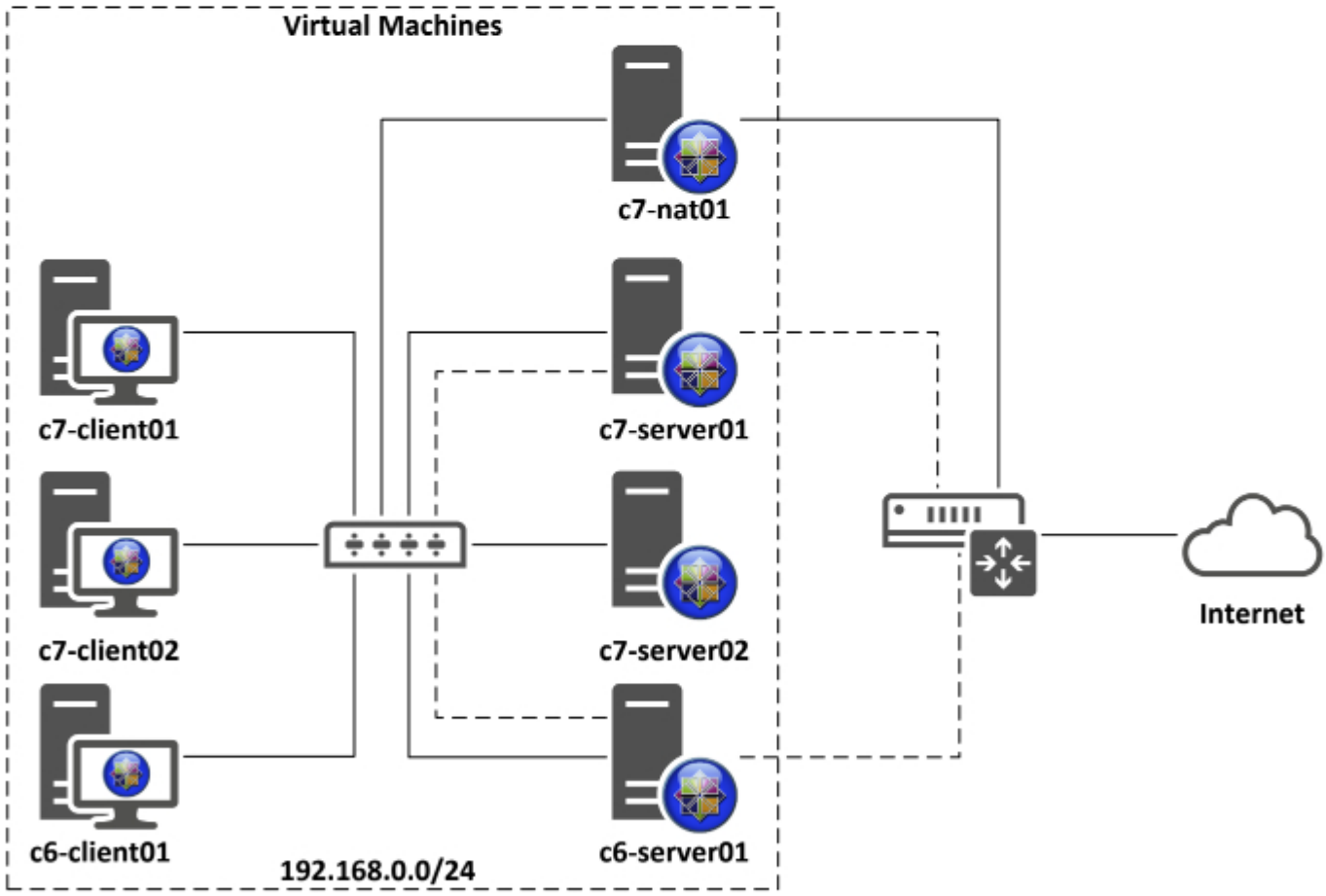
Чтобы посмотреть открытые в системе сетевые порты, дайте команду:

```
# netstat -na | head -n 10
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address               Foreign Address             State
tcp        0      0 0.0.0.0:802                 0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:139                 0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:5900                0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:111                 0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:20465               0.0.0.0:*                   LISTEN
tcp        0      0 127.0.0.1:631               0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:445                 0.0.0.0:*                   LISTEN
tcp        0      0 10.10.10.44:42136           66.102.9.99:80              ESTABLISHED
```

Лабораторная работа 6

- **Упражнения:**
  - Динамическое управление сетевыми интерфейсами в CentOS 7.
  - Управление сетевыми интерфейсами при помощи конфигурационных файлов в CentOS 7.
  - Объединение нескольких сетевых интерфейсов в режиме ACTIVE-BACKUP
- **Необходимые машины:**
  - `c7-nat01`.
- **Учетные данные:**
  - Пользователь: `root`.
  - Пароль: `Pa$$w0rd`.

http://edu.softline.ru/





Упражнение 1 – Динамическое управление сетевыми интерфейсами в CentOS 7.

Задача 1: Подготовка к лабораторной работе.

1. Войдите на виртуальную машину **c7-nat01**.
2. Остановите сервис **dhcpd**.
3. Убедитесь, что сервис успешно остановлен.
4. Запустите виртуальную машину **c7-server01**.

Задача 2: Управление IP-адресами при помощи пакета **iproute**.

1. Выведите список доступных сетевых интерфейсов.
2. Выведите информацию о сетевом интерфейсе **eth0**.
3. Остановите сетевой интерфейс **eth0**.
4. Убедитесь, что состояние интерфейса **eth0** – остановлен (**DOWN**).
5. Запустите сетевой интерфейс **eth0**.
6. Убедитесь, что состояние интерфейса **eth0** – запущен (**UP**).
7. Посмотрите текущую конфигурацию IPv4 адресов.
8. Посмотрите текущую конфигурацию IPv6 адресов.
9. Посмотрите всю текущую конфигурацию интерфейса **eth0**.
10. Назначьте IP-адрес **192.168.0.200** с маской **255.255.255.0** на интерфейс **eth0**.
11. Убедитесь, что IP-адрес успешно назначен на интерфейс **eth0**.
12. Убедитесь, что можете выполнить обмен ICMP-пакетами (**ping**) с хостом **192.168.0.1**.
13. Добавьте еще один IP-адрес **192.168.0.201** с маской **255.255.255.0** на интерфейс **eth0**.
14. Убедитесь, что IP-адрес успешно добавлен на интерфейс **eth0**.
15. Удалите IP-адрес **192.168.0.200** с интерфейса **eth0**.
16. Убедитесь, что IP-адрес успешно удален с интерфейса **eth0**.
17. Убедитесь, что можете выполнить обмен ICMP-пакетами (ping) с хостом **192.168.0.1**.

Задача 3: Управление маршрутизацией при помощи пакета **iproute**.

1. Отобразите таблицу маршрутизации.
2. Добавьте маршрут в сеть учебного класса через **c7-nat01** (**192.168.0.1**).

ПРИМЕЧАНИЕ:
Уточните IP-адрес сети учебного класса у инструктора.

3. Убедитесь, что маршрут успешно добавлен.
4. Убедитесь, что можете выполнить обмен ICMP-пакетами (**ping**) с хостом в сети учебного класса.



ПРИМЕЧАНИЕ:

Уточните IP-адрес в сети учебного класса у инструктора.

5. Удалите маршрут в сеть учебного класса через **c7-nat01** (**192.168.0.1**).
6. Убедитесь, что маршрут успешно удален.
7. Добавьте в таблицу маршрутизации шлюз по умолчанию: **192.168.0.1**.
8. Убедитесь, что можете выполнить обмен ICMP-пакетами (**ping**) с хостом **8.8.8.8**.

Задача 4: завершение лабораторной работы.

1. Войдите на виртуальную машину **c7-nat01**.
2. Запустите сервис **dhcpd**.
3. Убедитесь, что сервис успешно запущен.
4. Войдите на виртуальную машину **c7-server01**.
5. Перезапустите сервис **network.service**.



Ответы к упражнению 1 – Динамическое управление сетевыми интерфейсами в CentOS 7.

Задача 1: Подготовка к лабораторной работе.

1. Войдите на виртуальную машину **c7-nat01**.
2. Остановите сервис **dhcpd**.

```
# systemctl stop dhcpd.service
```

3. Убедитесь, что сервис успешно остановлен.

```
# systemctl status dhcpd.service
```

4. Запустите виртуальную машину **c7-server01**.

Задача 2: Управление IP-адресами при помощи пакета **iproute**.

1. Выведите список доступных сетевых интерфейсов.

```
# ip link
```

2. Выведите информацию о сетевом интерфейсе **eth0**.

```
# ip link show eth0
```

3. Остановите сетевой интерфейс **eth0**.

```
# ip link set dev eth0 down
```

4. Убедитесь, что состояние интерфейса **eth0** – остановлен (**DOWN**).

```
# ip link show eth0
```

5. Запустите сетевой интерфейс **eth0**.

```
# ip link set dev eth0 up
```

6. Убедитесь, что состояние интерфейса **eth0** – запущен (**UP**).

```
# ip link show eth0
```

7. Посмотрите текущую конфигурацию IPv4 адресов.

```
# ip -4 address
```

8. Посмотрите текущую конфигурацию IPv6 адресов.

```
# ip -6 address
```

9. Посмотрите всю текущую конфигурацию интерфейса **eth0**.

```
# ip address show eth0
```

10. Назначьте IP-адрес **192.168.0.200** с маской **255.255.255.0** на интерфейс **eth0**.

```
# ip a add 192.168.0.200/255.255.255.0 dev eth0
```

11. Убедитесь, что IP-адрес успешно назначен на интерфейс **eth0**.

```
# ip address show eth0
```

12. Убедитесь, что можете выполнить обмен ICMP-пакетами (**ping**) с хостом **192.168.0.1**.

```
# ping -c 4 192.168.0.1
```

13. Добавьте еще один IP-адрес **192.168.0.201** с маской **255.255.255.0** на интерфейс **eth0**.

```
# ip a add 192.168.0.201/255.255.255.0 dev eth0
```

14. Убедитесь, что IP-адрес успешно добавлен на интерфейс **eth0**.

```
# ip address show eth0
```

15. Удалите IP-адрес **192.168.0.200** с интерфейса **eth0**.

```
# ip address del 192.168.0.200/24 dev eth0
```

16. Убедитесь, что IP-адрес успешно удален с интерфейса **eth0**.

```
# ip address show eth0
```

17. Убедитесь, что можете выполнить обмен ICMP-пакетами (ping) с хостом **192.168.0.1**.

```
# ping -c 4 192.168.0.1
```

Задача 3: Управление маршрутизацией при помощи пакета **iproute**.

1. Отобразите таблицу маршрутизации.

```
# ip route list
```

2. Добавьте маршрут в сеть учебного класса через **c7-nat01** (**192.168.0.1**).

```
# ip route add 172.20.70.128/25 via 192.168.0.1
```



ПРИМЕЧАНИЕ:
Уточните IP-адрес сети учебного класса у инструктора.

3. Убедитесь, что маршрут успешно добавлен.

```
# ip route list
```

4. Убедитесь, что можете выполнить обмен ICMP-пакетами (**ping**) с хостом в сети учебного класса.

```
# ping -c 4 172.20.70.129
```



ПРИМЕЧАНИЕ:
Уточните IP-адрес в сети учебного класса у инструктора.

5. Удалите маршрут в сеть учебного класса через **c7-nat01** (**192.168.0.1**).

```
# ip route del 172.20.70.128/25 via 192.168.0.1
```

6. Убедитесь, что маршрут успешно удален.

```
# ip route list
```

7. Добавьте в таблицу маршрутизации шлюз по умолчанию: **192.168.0.1**.

```
# ip route add default via 192.168.0.1
```

8. Убедитесь, что можете выполнить обмен ICMP-пакетами (**ping**) с хостом **8.8.8.8**.

```
# ping -c 4 8.8.8.8
```

Задача 4: Завершение лабораторной работы.

1. Войдите на виртуальную машину **c7-nat01**.
2. Запустите сервис **dhcpd**.

```
# systemctl start dhcpd.service
```

3. Убедитесь, что сервис успешно запущен.

```
# systemctl status dhcpd.service
```

4. Войдите на виртуальную машину **c7-server01**.
5. Перезапустите сервис **network.service**.

```
# systemctl restart network.service
```



Упражнение 2 –Управление сетевыми интерфейсами при помощи конфигурационных файлов в CentOS 7.

Задача 1: Настройка сетевого интерфейса при помощи конфигурационного файла.

5. Войдите на виртуальную машину **c7-server01**.
6. При помощи конфигурационного файла, задайте следующие настройки сетевому интерфейсу eth0:
   - IP-адрес: 192.168.0.101.
   - Маска: 255.255.255.0.
   - Шлюз по умолчанию: 192.168.0.1
   - DNS-сервер: 172.20.70.253.
   - Домен поиска: ll-100.local.

ПРИМЕЧАНИЕ:
Уточните IP-адрес DNS-сервера у инструктора.

7. Перезапустит сетевую подсистему.
8. Убедитесь, что интерфейсу **eth0** назначен IP-адрес **192.168.0.101**.
9. Проверьте возможность доступа во внешнюю сеть.

Задача 2: Назначение нескольких IP адресов одному сетевому интерфейсу.

1. Измените конфигурацию на интерфейсе **eth0** назначив ему два IP-адреса: 192.168.0.101/24 и 10.0.0.25/16:
2. Перезапустите сетевую подсистему.
3. Убедитесь, что интерфейсу **eth0** назначено два IP-адреса.
4. Проверьте возможность доступа во внешнюю сеть.
5. Самостоятельно сконфигурируйте интерфейс **eth0** на машине **c7-client01** на использование двух IP-адресов 192.168.0.102/24 и 10.0.0.26/16.
6. Убедитесь в возможности сетевого соединения между машинами **c7-server01** и **c7-client01** при помощи команды **ping**.
7. Верните конфигурацию интерфейса **eth0** на машинах **c7-server01** и **c7-client01** в исходное состояние.



Ответы к упражнению 2 –Управление сетевыми интерфейсами при помощи конфигурационных файлов в CentOS 7.

Задача 1: Настройка сетевого интерфейса при помощи конфигурационного файла.

1. Войдите на виртуальную машину **c7-server01**.
2. При помощи конфигурационного файла, задайте следующие настройки сетевому интерфейсу eth0:
   - IP-адрес: 192.168.0.101.
   - Маска: 255.255.255.0.
   - Шлюз по умолчанию: 192.168.0.1
   - DNS-сервер: 172.20.70.253.
   - Домен поиска: ll-100.local.

```
# vi /etc/sysconfig/network-scripts/ifcfg-eth0
BOOTPROTO=static
ONBOOT=yes
IPADDR=192.168.0.101
NETMASK=255.255.255.0
GATEWAY=192.168.0.1
DNS1=172.20.70.253
DOMAIN=ll-100.local
```

ПРИМЕЧАНИЕ:
Уточните IP-адрес DNS-сервера у инструктора.

3. Перезапустит сетевую подсистему.

```
# systemctl restart network
```

4. Убедитесь, что интерфейсу **eth0** назначен IP-адрес **192.168.0.101**.

```
# ip a
```

5. Проверьте возможность доступа во внешнюю сеть.

```
# ping -c 4 www.yandex.ru
```

Задача 2: Назначение нескольких IP адресов одному сетевому интерфейсу.

1. Измените конфигурацию на интерфейсе **eth0** назначив ему два IP-адреса: 192.168.0.101/24 и 10.0.0.25/16:

```
# vi /etc/sysconfig/network-scripts/ifcfg-eth0
BOOTPROTO=static
```

```
ONBOOT=yes
IPADDR0=192.168.0.101
NETMASK0=255.255.255.0
GATEWAY0=192.168.0.1
DNS1=172.20.70.251
IPADDR1=10.0.0.25
NETMASK1=255.255.0.0
```

2. Перезапустите сетевую подсистему.

```
# systemctl restart network
```

3. Убедитесь, что интерфейсу **eth0** назначено два IP-адреса.

```
# ip a
```

4. Проверьте возможность доступа во внешнюю сеть.

```
# ping -c 3 www.yandex.ru
```

5. Самостоятельно сконфигурируйте интерфейс **eth0** на машине **c7-client01** на использование двух IP-адресов 192.168.0.102/24 и 10.0.0.26/16.
6. Убедитесь в возможности сетевого соединения между машинами **c7-server01** и **c7-client01** при помощи команды **ping**.
7. Верните конфигурацию интерфейса **eth0** на машинах **c7-server01** и **c7-client01** в исходное состояние.



Упражнение 3 – Объединение нескольких сетевых интерфейсов в режиме ACTIVE-BACKUP.

6. Войдите на виртуальную машину **c7-server01**.
7. Убедитесь, что модуль **hv_netvsc** импортирован:
8. Создайте конфигурационный файл для объединенного интерфейса **ifcfg-bond0** со следующим содержимым:

```
DEVICE=bond0
NAME=bond0
TYPE=bond
BONDING_MASTER=yes
ONBOOT=yes
BOOTPROTO=none
BONDING_OPTS="mode=1 miimon=100"
IPADDR=192.168.0.201
NETMASK=255.255.255.0
GATEWAY=192.168.0.1
DNS1=172.20.70.253
DOMAIN=ll-100.local
```



ПРИМЕЧАНИЕ:
Уточните IP-адрес DNS-сервера у инструктора.

9. Измените конфигурацию интерфейсов **eth0** и **eth1**, указав им участие в **bond0**:

```
BOOTPROTO=none
…
ONBOOT=yes
…
MASTER=bond0
SLAVE=yes
```

10. Перезапустите сетевую подсистему.
11. Проверьте, как ядро распознало новую конфигурацию объединенного интерфейса:
12. Проверьте возможность доступа во внешнюю сеть.
13. С машины **c7-client01** проверьте доступность машины **c7-server01** (192.168.0.201) при помощи команды **ping**.



Ответы к упражнению 3 – Объединение нескольких сетевых интерфейсов в режиме ACTIVE-BACKUP.

1. Войдите на виртуальную машину **c7-server01**.
2. Убедитесь, что модуль **hv_netvsc** импортирован:
3. Создайте конфигурационный файл для объединенного интерфейса **ifcfg-bond0** со следующим содержимым:

```
# vi /etc/sysconfig/network-scripts/ifcfg-bond0
DEVICE=bond0
NAME=bond0
TYPE=bond
BONDING_MASTER=yes
ONBOOT=yes
BOOTPROTO=none
BONDING_OPTS="mode=1 miimon=100"
IPADDR=192.168.0.201
NETMASK=255.255.255.0
GATEWAY=192.168.0.1
DNS1=172.20.70.253
DOMAIN=ll-100.local
```

ПРИМЕЧАНИЕ:
Уточните IP-адрес DNS-сервера у инструктора.

4. Измените конфигурацию интерфейсов **eth0** и **eth1**, указав им участие в **bond0**:

```
# vi /etc/sysconfig/network-scripts/ifcfg-eth0
BOOTPROTO=none
…
ONBOOT=yes
…
MASTER=bond0
SLAVE=yes
```

5. Перезапустите сетевую подсистему.

```
# systemctl restart network
```

6. Проверьте, как ядро распознало новую конфигурацию объединенного интерфейса:

```
# cat /proc/net/bonding/bond0
```

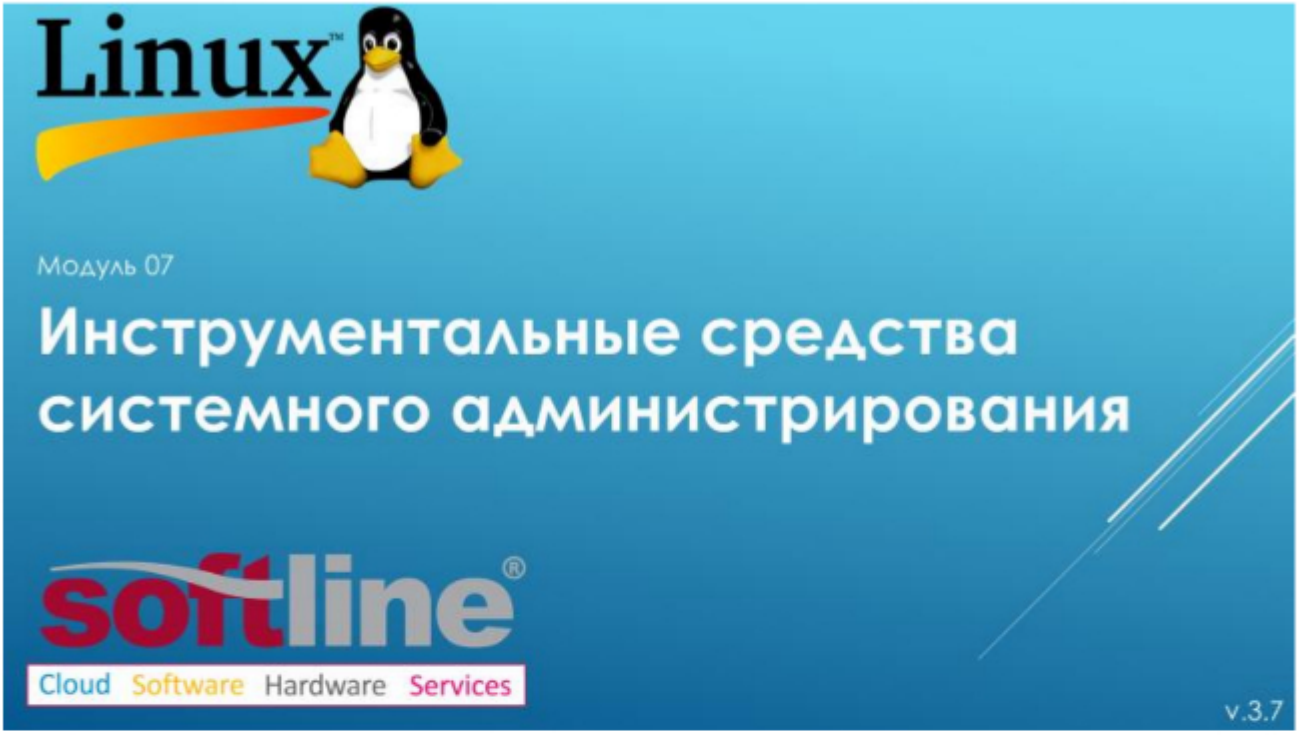
7. Проверьте возможность доступа во внешнюю сеть.

```
# ping -c 4 www.yandex.ru
```

8. С машины **c7-client01** проверьте доступность машины **c7-server01** (192.168.0.201) при помощи команды **ping**.

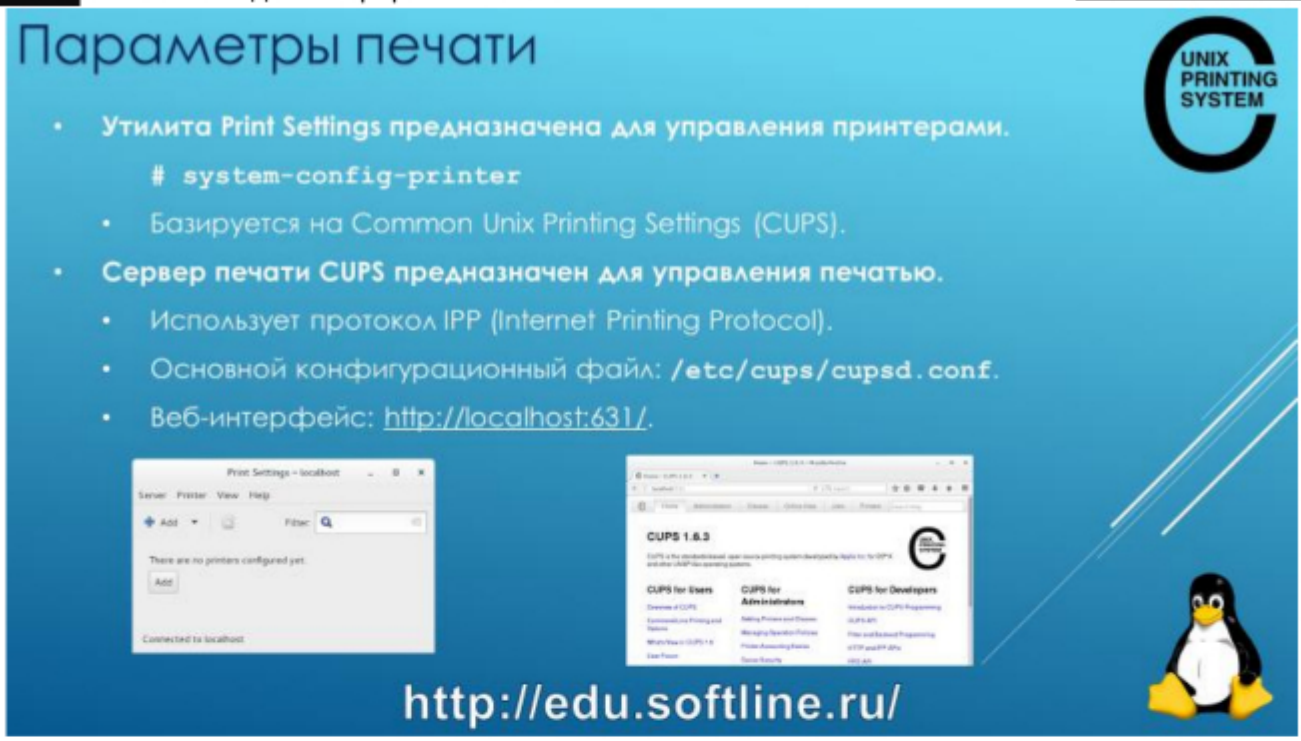
```
# ping c7-server01
```

Модуль 07: Инструментальные средства системного администрирования



В модуле представлены:

- Сервер печати – CUPS.
- Журнальные файлы (**`syslog`**/**`rsyslog`**.).
- Резервное копирование (**`dump`**/**`restore`**).
- Резервное копирование XFS.
- OpenSSH.
- TigerVNC.



Утилита **Print Settings** предназначена для настройки принтера, обслуживания конфигурационных файлов принтеров, буферных каталогов печати и фильтров печати, а также для управления классами принтеров.

Утилита базируется на Common Unix Printing Settings (CUPS). Если вы обновили систему с предыдущей версии CentOS (RHEL), которая использовала CUPS, процесс обновления сохранит настроенные принтеры.

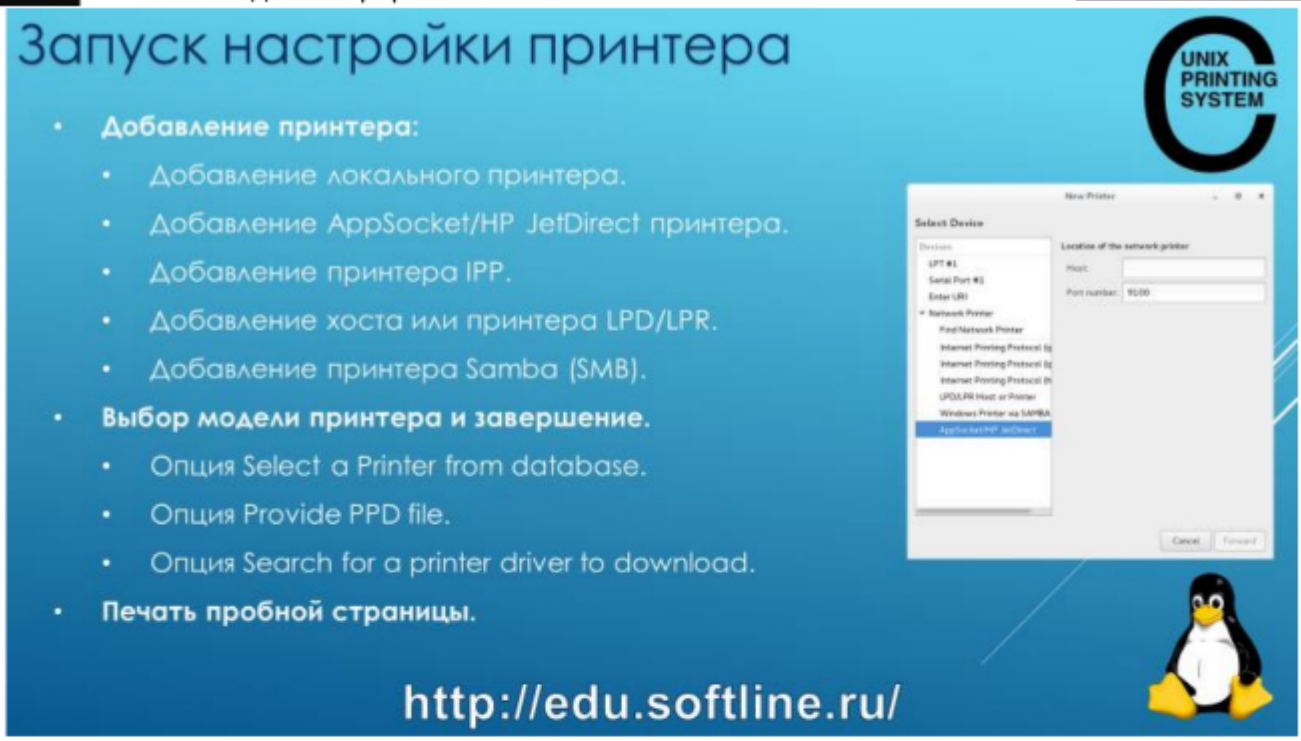
ПРИМЕЧАНИЕ:
Страница руководства ***cupsd.conf*** *описывает настройку сервера CUPS. Она содержит директивы для включения поддержки SSL. Однако, CUPS не позволяет управлять версией используемого протокола. В связи с уязвимостью POODLE SSLv3.0 рекомендуется не использовать встроенные возможности SSL. Надежнее использовать* ***stunnel*** *для предоставления безопасного канала и отключить SSLv3.*

ПРИМЕЧАНИЕ:
Можно выполнить аналогичные и дополнительные операции напрямую через веб-приложение CUPS или командную строку. Для доступа к приложению обратитесь к ссылке http://localhost:631/ при помощи веб-браузера. Для получения доступа к руководствам CUPS воспользуйтесь ссылками на вкладке ***Home****.*

Запуск утилиты настройки параметров печати (Print Settings).

При помощи утилиты настройки **Print Settings** можно выполнять различные действия на существующих принтерах, устанавливать новые принтеры. Также можно использовать CUPS напрямую (при помощи веб-приложения доступного по ссылке http://localhost:631/).

Для запуска утилиты Print Settings из командной строки, введите **system-config-printer** в строку оболочки.



Запуск настройки принтера.

Процесс установки принтера изменяется в зависимости от типа очереди принтера.

Если вы настраиваете локальный принтер, подключенный через USB, принтер обнаруживается и добавляется автоматически. Возможно, будет запрошено подтверждение установки пакета и пароль администратора или root. Локальные принтеры, подключаемые через другие типы портов, и сетевые принтеры нуждаются в ручной настройке.

Процедура запуска ручной настройки принтера:

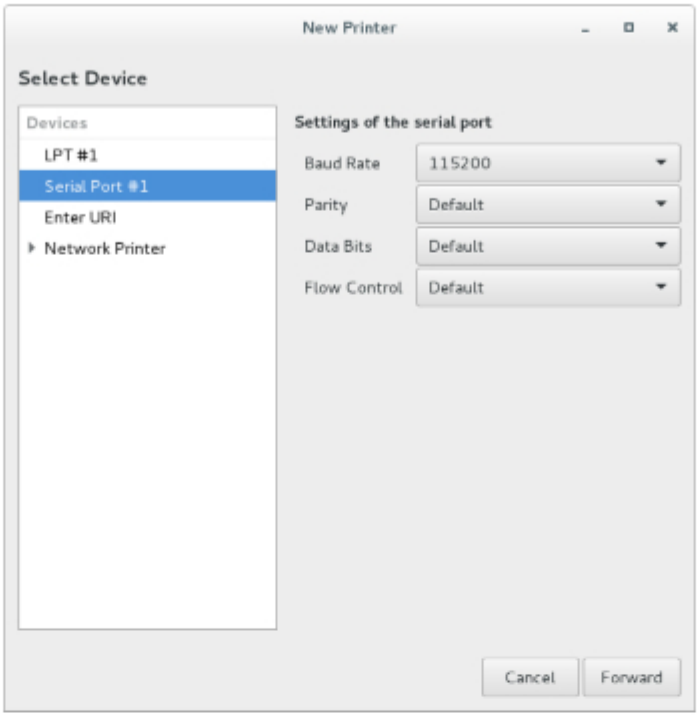
1. Запустите утилиту **Printer Settings**.
2. Выберите **Server** > **New** > **Printer**.
3. В диалоговом окне **Authenticate** введите пароль администратора или пользователя root. Если в первые настраивается удаленный принтер, будет запрошено разрешение на изменение в межсетевом экране (Firewall).
4. Выберите тип подключения принтера и укажите детали в области справа.

Добавление локального принтера.

Используйте следующую процедуру для добавления локального принтера, подключенного через последовательный порт (Serial/LPT):

1. Откройте диалоговое окно добавления принтера (**Add**).
2. Если устройство не отобразилось автоматически, выберите порт к которому принтер подключен в списке слева (Например, **Serial Port #1** или **LPT #1**).
3. Справа укажите опции подключения:
   - Для Other:
     - URI (например, file:/dev/lp0).
   - Для Serial Port:
     - Baud Rate.

- Parity.
- Data Bits.
- Flow Control.

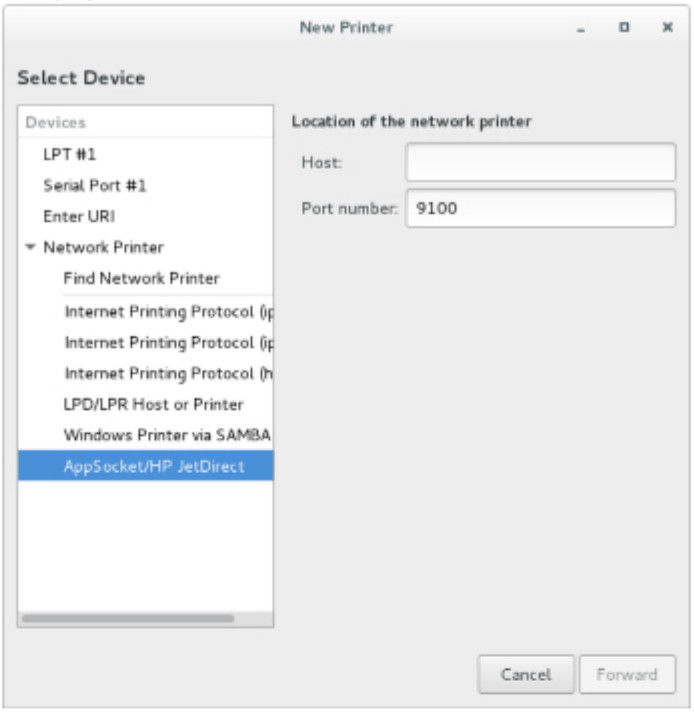


4. Нажмите **Forward**.
5. Выберите модель принтера.

Добавление AppSocket/HP JetDirect принтера.

Используйте следующую процедуру для добавления AppSocket/HP JetDirect:

1. Откройте диалоговое окно добавления принтера (**Add**).
2. В списке слева выберите **Network Printer** > **AppSocket/HP JetDirect**.
3. Справа укажите опции подключения:
   - **Host** – имя хоста принтера или IP-адрес.
   - **Port number** – Порт, прослушиваемый принтером для заданий печати (по умолчанию 9100).



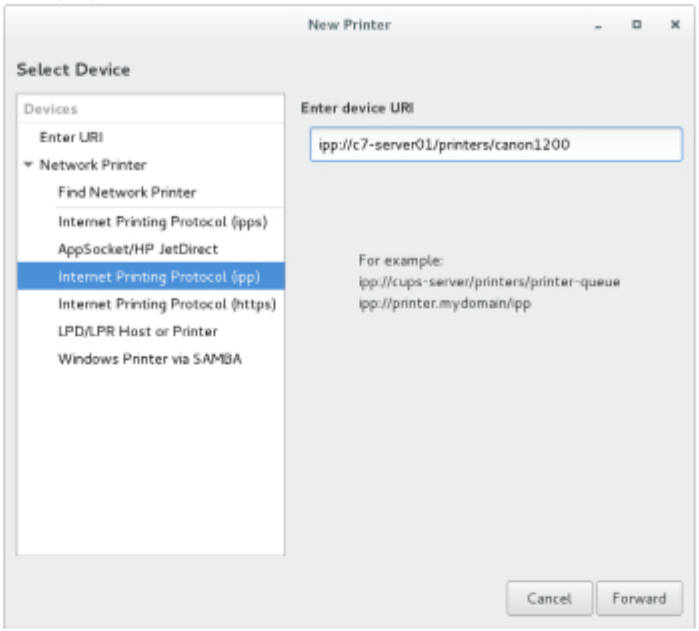
4. Нажмите **Forward**.
5. Выберите модель принтера.

Добавление принтера IPP.

Принтер IPP – это принтер, подключенный к другой системе в сети TCP/IP. Система, к которой подключен принтер, может использовать CUPS или аналогично сконфигурирована для IPP. Если на сервере печати включен межсетевой экран, то он должен быть сконфигурирован для разрешения входящих подключений TCP на порту 631. Протокол просмотра CUPS разрешает клиентским машинам обнаруживать предоставленные в общий доступ очереди CUPS. Чтобы обеспечить это, межсетевой экран на клиентской машине должен разрешить входящие пакеты UDP на порту 631.

Используйте следующую процедуру для добавления принтера IPP:

1. Откройте диалоговое окно добавления принтера (**Add**).
2. В списке слева выберите **Network Printer** > **Internet Printing Protocol (ipp)**, **Internet Printing Protocol (ipps)** или **Internet Printing Protocol (https)**.
3. Справа укажите опции подключения:
   - **Host** – имя хоста принтера или IP-адрес.
   - **Queue** – имя очереди, которое будет дано новой очереди (если поле оставить пустым, будет использовано имя основанное на узле устройства).

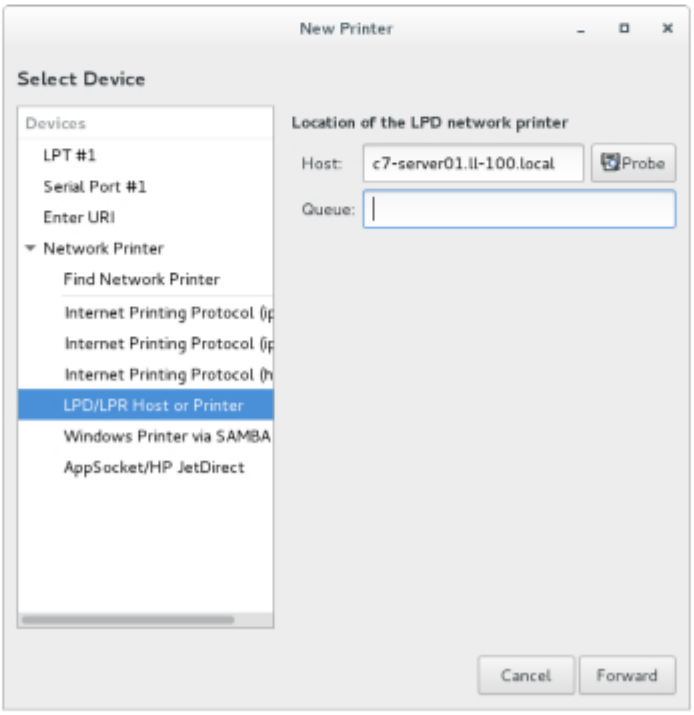


4. Нажмите **Forward**.
5. Выберите модель принтера.

Добавление хоста или принтера LPD/LPR.

Используйте следующую процедуру для добавления хоста или принтера LPD/LPR:

1. Откройте диалоговое окно добавления принтера (**Add**).
2. В списке слева выберите **Network Printer** > **LPD/LPR Host or Printer**.
3. Справа укажите опции подключения:
   - **Host** – имя хоста принтера или хоста LPD/LPR.
     Опционально можно нажать **Probe**, чтобы найти очереди на хосте LPD.
   - **Queue** – имя очереди, которое будет дано новой очереди (если поле оставить пустым, будет использовано имя основанное на узле устройства).



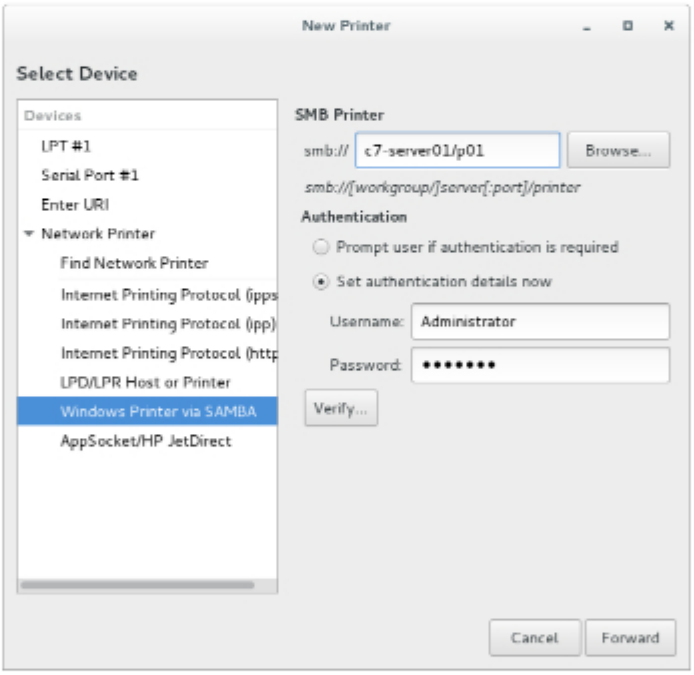
4. Нажмите **Forward**.
5. Выберите модель принтера.

Добавление принтера Samba (SMB).

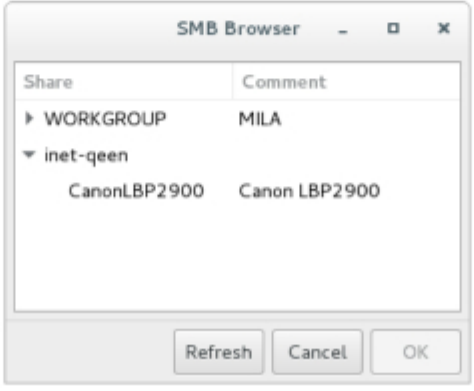
> *ПРИМЕЧАНИЕ:*
> *Перед добавлением принтера Samba, необходимо установить пакет* ***samba-client****:*
> ***# yum install samba-client***

Используйте следующую процедуру для добавления принтера Samba:

1. Откройте диалоговое окно добавления принтера (**Add**).
2. В списке слева выберите **Network Printer** > **Windows Printer via SAMBA**.
3. Введите адрес SMB в поле **smb://**. Используйте формат *имя компьютера/общий принтер*.



4. Нажмите **Browse**, чтобы увидеть доступные рабочие группы/домены. Для отображения только указанного хоста, введите имя хоста (имя NetBios) и нажмите **Browse**.



5. Выберите одну из следующих опций:
   a. **Prompt user if authentication is required** – имя пользователя и пароль будут запрашиваться у пользователя во время печати документа.
   b. **Set authentication details now** – предоставить информацию по аутентификации сейчас, чтобы она не требовалась позже. В поле **Username**, введите имя пользователя для доступа к принтеру. Этот пользователь должен существовать на системе SMB и должен обладать правами на доступ к принтеру. Имя пользователя по умолчанию обычно **guest** для серверов Windows или **nobody** для серверов Samba.
6. Введите пароль (если требуется) для указанного пользователя в поле Password.



ВНИМАНИЕ:

Имя пользователя и пароль от принтера Samba сохраняются на сервере печати в виде незашифрованных файлов, доступных для чтения пользователю root и Linux Printing Daemon (`lpd`).

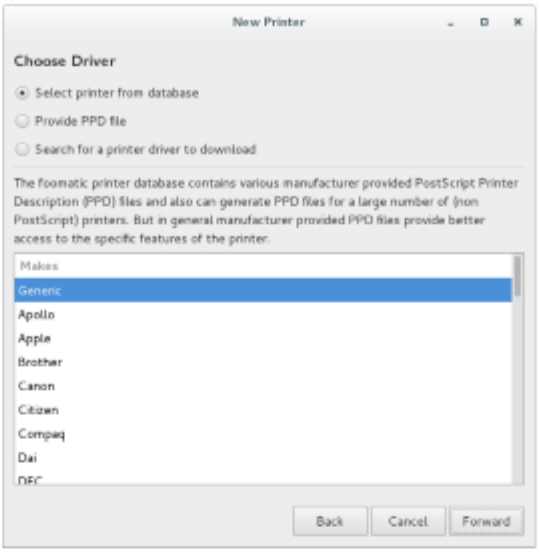
7. Нажмите **Verify**, чтобы проверить подключение.
8. Нажмите **Forward**.
9. Выберите модель принтера.

Выбор модели принтера и завершение.

После выбора необходимого типа подключения принтера, система пробует получить драйвер. Если процесс завершается неуспешно, можно указать или найти ресурсы драйвера вручную.

Используйте следующую процедуру для предоставления драйвера принтера и завершения установки:

1. В окне, отображенном после неудачного автоматического определения драйвера, выберите одну из следующих опций:
   a. **Select a Printer from database** – система выберет драйвер основываясь на выбранном производителе принтера из списка **Makes**. Если модель принтера не представлена, можно выбрать **Generic**.
   b. **Provide PPD file** – система использует предоставленный файл PostScript Printer Description (PPD) для установки. Файл PPD может поставляться вместе с принтером, или он может быть получен от производителя принтера. Если файл PPD доступен, всегда нужно выбирать данную опцию и использовать панель браузера ниже описания опций для выбора файла PPD.
   c. **Search for a printer driver to download** – укажите производителя и модель принтера в поле Make and model для поиска соответствующих пакетов на OpenPrinting.org.

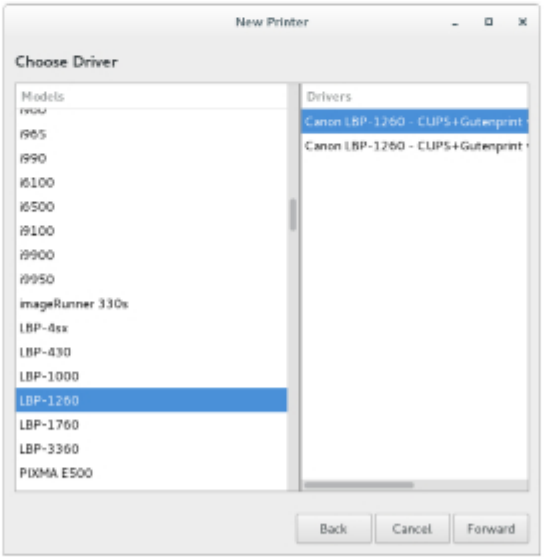


2. В зависимости от сделанного выбора предоставьте детали в области, отображенной ниже:
   - Производителя принтера для опции **Select a Printer from database**.
   - Расположение файла PPD для опции **Provide PPD file**.
   - Производителя и модель принтера для опции **Search for a printer driver to download**.
3. Нажмите **Forward** для продолжения.
4. Если опции подойдут, отобразится окно выбора соответствующей модели.

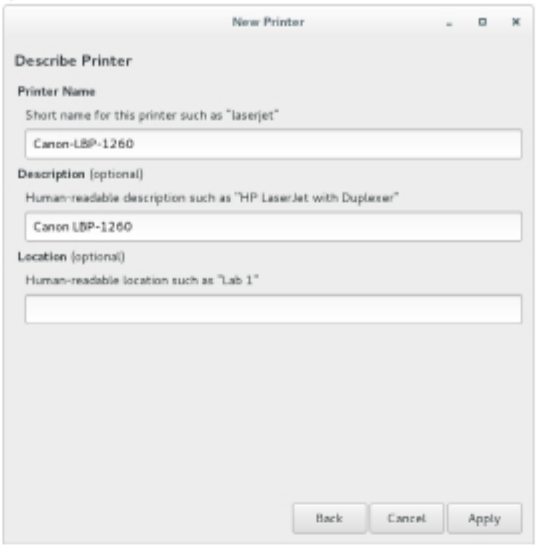


ПРИМЕЧАНИЕ:

После выбора модели в колонке Models слева, справа будет выбран рекомендованный драйвер, однако, вы можете выбрать другой доступный драйвер. Драйвер принтера обрабатывает данные, которые приходят на печать, в формат, который принтер может понять. После подключения локального принтера к компьютеру, необходим драйвер принтера для обработки данных перед отправкой их на устройство.



5. Нажмите **Forward**.
6. Под **Describe Printer** введите уникальное имя для принтера в поле **Printer Name**. Имя принтера может содержать буквы, числа, тире (-) и подчеркивания (_); оно не должно содержать пробелов. Также можно использовать поля Description и Location для добавления информации о принтере. Оба поля опциональны и могут содержать пробелы.

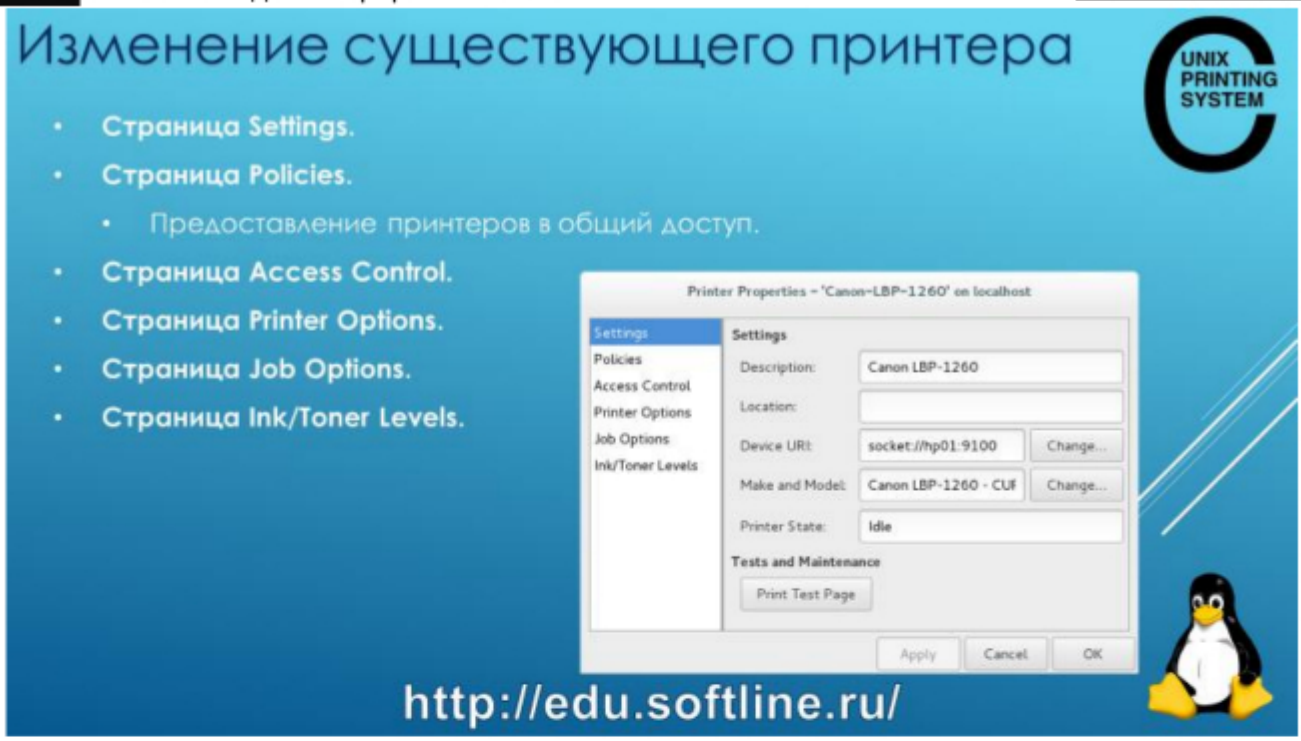


7. Нажмите **Apply** для подтверждения конфигурации принтера и добавьте очередь печати, если параметры корректные. Если необходимо изменить конфигурацию принтера, можно воспользоваться кнопкой **Back**.
8. После применения изменений отобразится диалоговое окно, позволяющее напечатать пробную страницу. Нажмите Yes для печати пробной страницы. В качестве альтернативы, можно распечатать пробную страницу позже.

Печать пробной страницы.

После настройки принтера или изменения его конфигурации, печать пробной страницы подтвердит корректность его настройки:

1. Вызовите **Properties** на принтере, отображаемом в окне Printing.
2. В окне **Properties** нажмите Settings слева.
3. На вкладке **Settings** нажмите кнопку **Print Test Page**.



Для удаления существующего принтера в окне **Print Settings**, выберите принтера и перейдите **Printer** > **Delete**. Подтвердите удаления принтера. Также можно воспользоваться клавишей **<Delete>**.

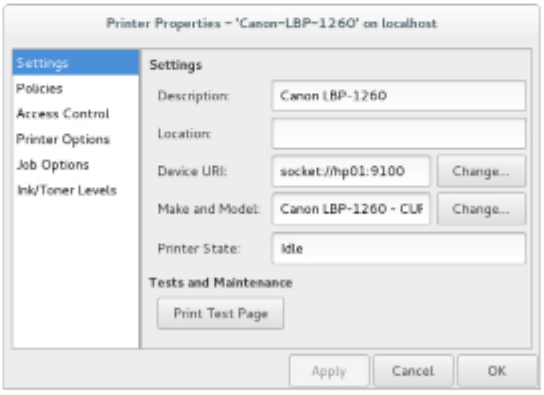
Чтобы установить принтер по умолчанию, вызовите меню на принтере и выберите кнопку **Set as Default** в контекстном меню.

Для изменения параметров принтера, воспользуйтесь следующими страницами:

Страница Settings.

Чтобы изменить конфигурацию драйвера принтера, откройте страницу **Settings** необходимого принтера.

На ней можно изменять параметры принтера, такие как производитель и модель, печатать тестовую страницу, изменять расположение устройства (URL) и так далее.



Страница Policies.

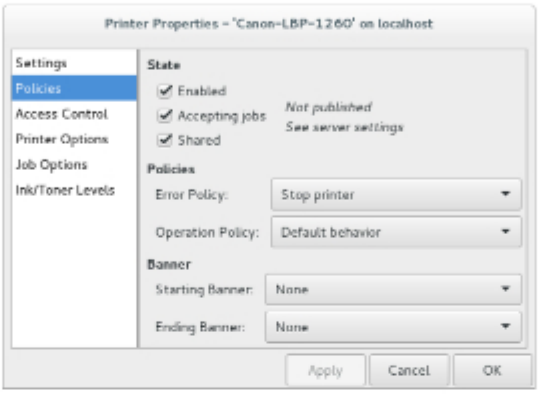
Нажмите кнопку Policies слева чтобы изменить состояние и вывод принтера.

Вы можете выбрать состояния принтера, настроить **Error Policy** принтера (доступны на выбор прервать задание печати, повторить или остановить при возникновении ошибки).

Также можно создать Banner Page (страницу которая описывает аспекты задания печати, такие как исходный принтер, имя пользователя, создавшего задание и статус печатаемого документа): нажмите Starting Banner или Ending Banner в ниспадающем меню и выберите опции, которые лучше всего описывают природу задания печати (например, конфиденциальность).

Предоставление принтеров в общий доступ.

На странице **Policies**, вы можете отметить принтер как общий: если принтер общий, пользователи, имеющие доступ к сети, могут его использовать. Чтобы разрешить общий доступ к принтеру, перейдите **Server** > **Settings** и выберите **Publish shared printers connected to this system**.



Убедитесь, что межсетевой экран разрешает входящие соединения TCP на порту 631, порт для протокола Network Printing Server (IPP).

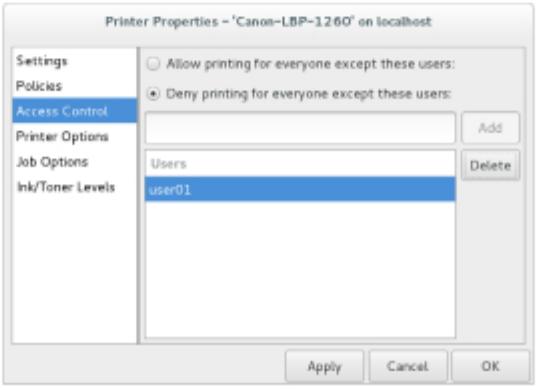
Чтобы разрешить службу IPP в firewalld на CentOS 7 (RHEL 7) воспользуйтесь следующей командой:

```
# firewall-cmd --add-service=ipp --permanent
# firewall-cmd --reload
```

В качестве альтернативы, можно воспользоваться графическим инструментом **`firewall-config`**.

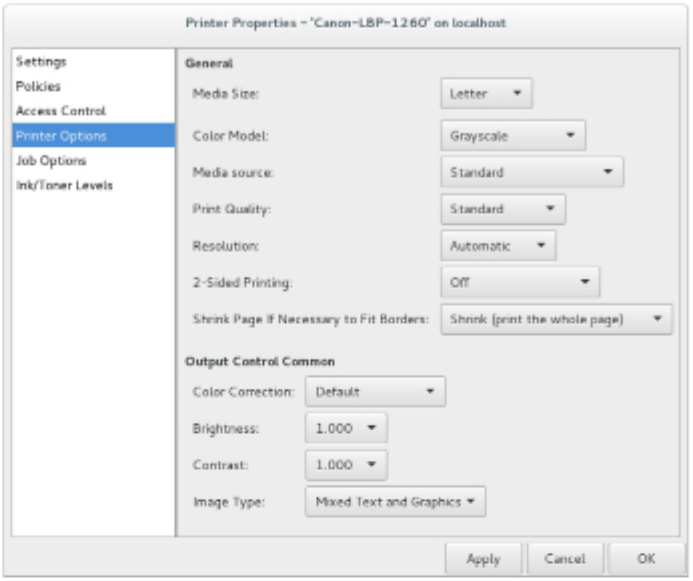
Страница Access Control.

На странице **Access Control** можно изменить уровень доступа пользователя к настроенному принтеру. Выберите один из двух вариантов **Allow printing for everyone except these users** или **Deny printing for everyone except these users**, затем определите список пользователей ниже, вводя имена пользователей в поле и нажимая кнопку **Add**.



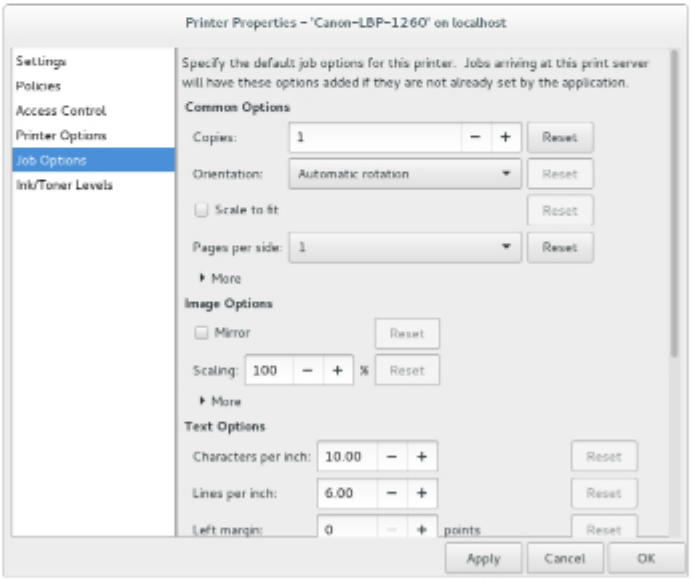
Страница Printer Options.

Страница **Printer Options** содержит различные опции настройки печати, контент этой страницы меняется в зависимости от выбранного принтера. Она содержит общие настройки, параметры печати, бумаги, качество и размеров печати.



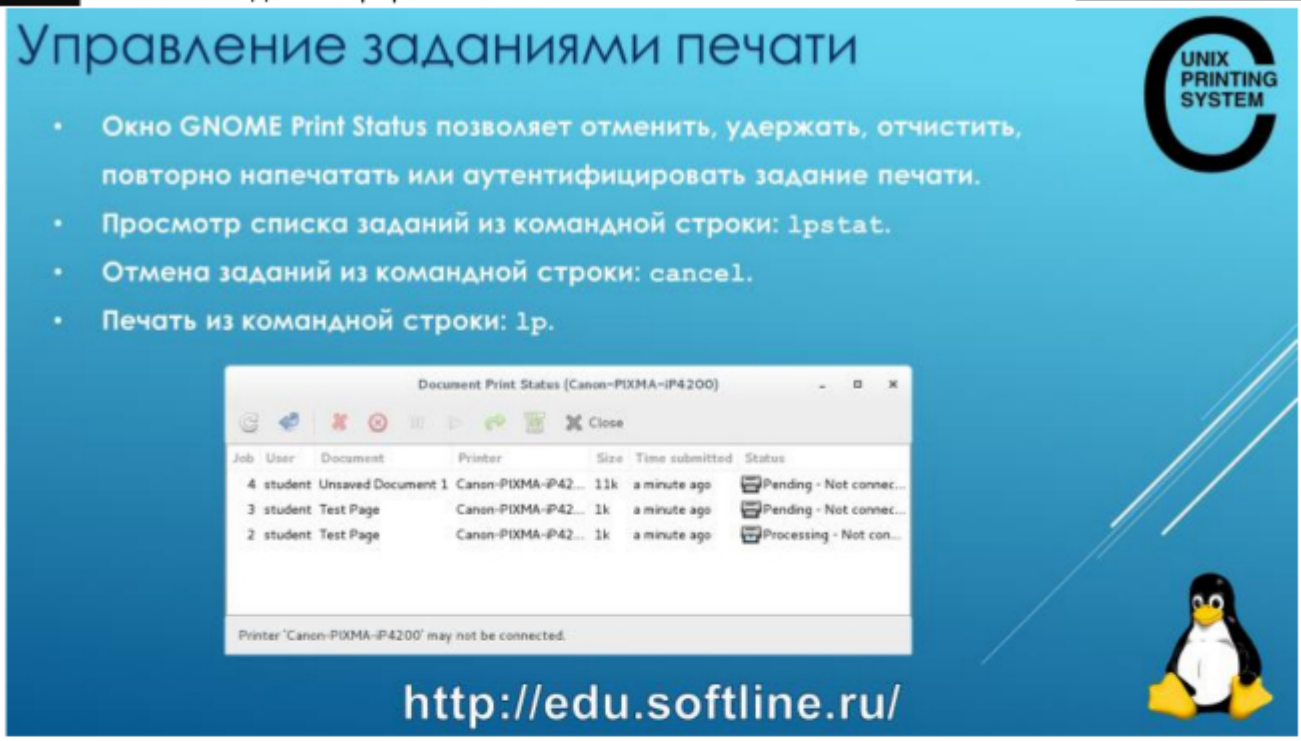
Страница Job Options.

Страница **Job Options** содержит подробности заданий печати. Измените параметры по умолчанию, чтобы применить настраиваемые опции заданий, такие как: число копий, ориентация, количество страниц на листе, масштабирование (увеличение или уменьшение области печати на листе), подробные опции текста и настраиваемые опции заданий.



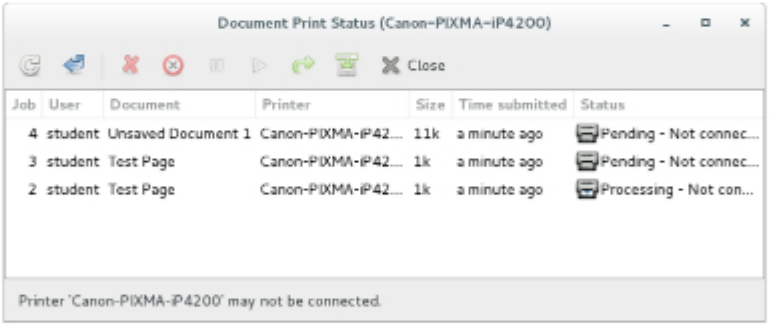
Страница Ink/Toner Levels.

Страница **Ink/Toner Levels** содержит подробности о статусе тонера, если доступны статусные сообщения принтера.



Когда вы отправляет задание печати на принтер, такое как печать текстового файла из редактора Emacs или изображения из GIMP, задание печати добавляется в очередь буфера печати. Буфер очереди печати – это список заданий печати, которые были отправлены на принтер и информация о каждом запросе печати, такая как статус запроса, номер задания и так далее.

В окружении рабочего стола GNOME, в процессе печати, иконка **Printer Status** отобразится в области уведомлений (Notification Area) на панели. Для проверки статуса задания печати, щелкните иконку Printer Status и отобразится окно **GNOME Print Status**.



Чтобы отменить, удержать, отчистить, повторно напечатать или аутентифицировать задание печати в GNOME Print Status выберите меню Job и щелкните необходимую команду.

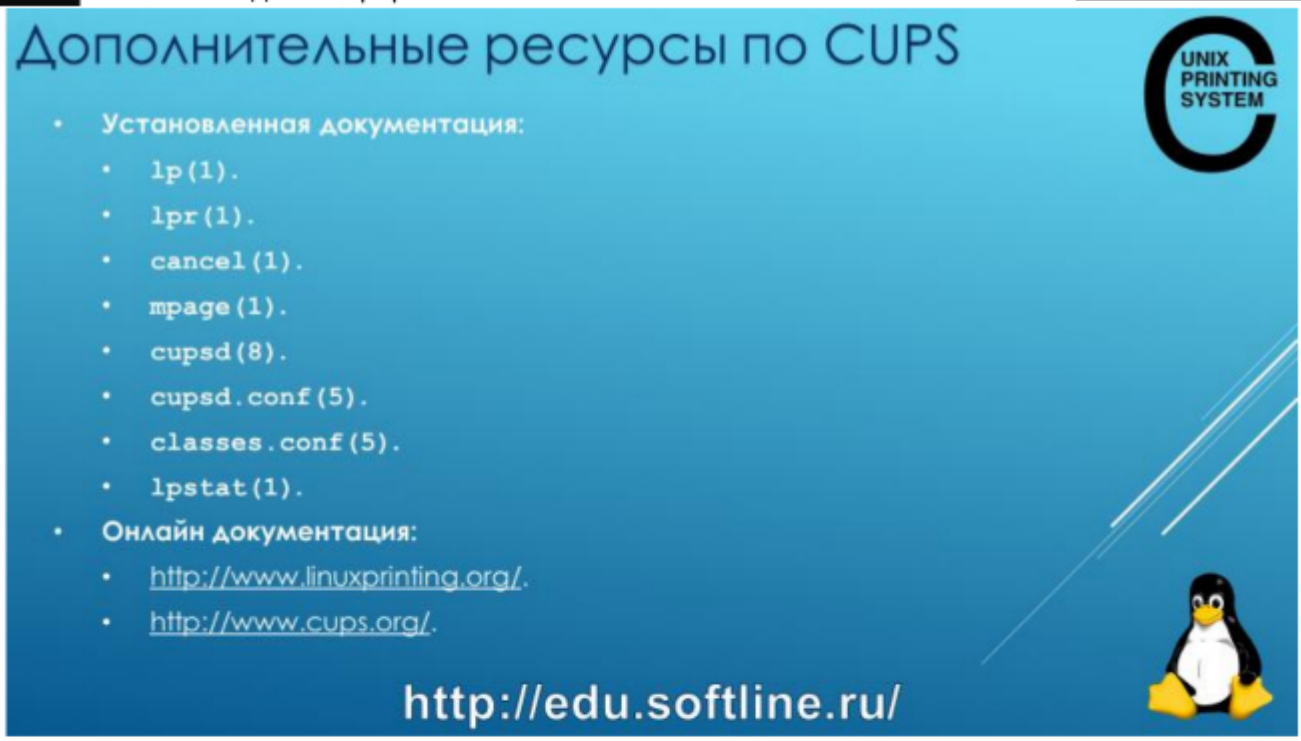
Для просмотра списка заданий печати в буфере печати из командной строки, введите команду **lpstat -o**.

```
# lpstat -o
Canon-LBP-1260-1        student            1024   Mon 13 Feb 2017 12:45:42 AM +06
Canon-PIXMA-iP4200-2    student            1024   Mon 13 Feb 2017 12:45:48 AM +06
Canon-PIXMA-iP4200-3    student            1024   Mon 13 Feb 2017 12:46:22 AM +06
Canon-PIXMA-iP4200-4    student           11264   Mon 13 Feb 2017 12:46:44 AM +06
```

Если необходимо отменить задание печати, найдите номер запроса при помощи команды lpstat – и затем используйте команду **cancel** *номер_задания*. Например, **cancel 60** отменит первое задание печати из вывода выше. Невозможно отменить задания печати, запущенные другими пользователями при помощи команды **cancel**. Однако, можно принудительно выполнить удаление

любого задания при помощи команды **`cancel -U root`** *`номер_задания`*. Чтобы предотвратить такую отмену, необходимо перевести операционную политику принтера в **Authenticated** для принудительной аутентификации root.

Также можно печатать файлы напрямую из строки оболочки. Например, команда **`lp sample.txt`** распечатывает текстовый файл **`sample.txt`**. Фильтр печать определяет какой тип пришедшего файла и конвертирует в формат понятный принтеру.



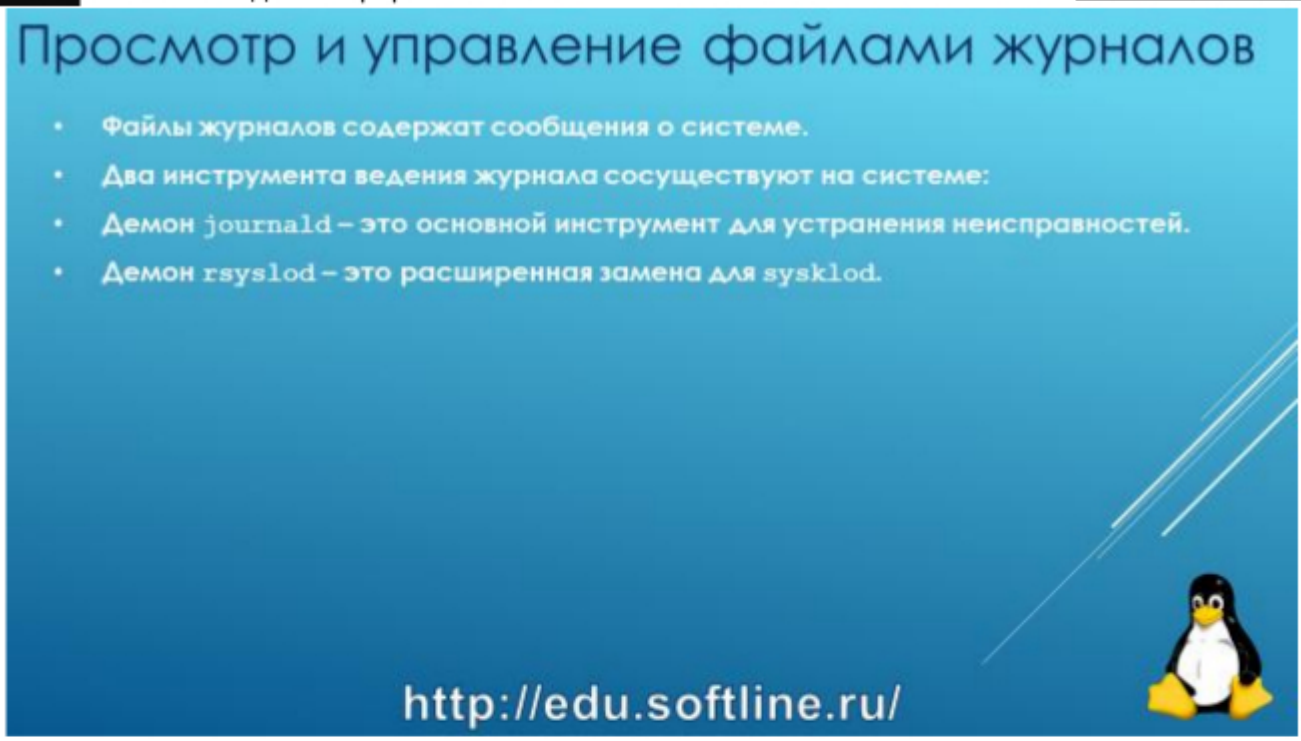
Воспользуйтесь следующими ресурсами для изучения печати в CentOS (RHEL).

Установленная документация:

- **`lp(1)`** – страница руководства для команды **`lp`**, которая позволяет печатать файлы из командной строки.
- **`lpr(1)`** – страница руководства для команды **`lpr`**, которая позволяет вам печатать файлы из командной строки.
- **`cancel(1)`** – страница руководства для утилиты командной строки для удаления заданий из очереди печати.
- **`mpage(1)`** – страница руководства для утилиты командной строки для печати нескольких страниц на одном листе.
- **`cupsd(8)`** – страница руководства для демона печати CUPS.
- **`cupsd.conf(5)`** – страница руководства для конфигурационного файла демона печати CUPS.
- **`classes.conf(5)`** – страница руководства для конфигурационного файла классов CUPS.
- **`lpstat(1)`** – страница руководства для команды **`lpstat`**, которая отображает статусную информацию о классах заданиях и принтерах.

Онлайн документация:

- http://www.linuxprinting.org/ – группа OpenPrinting на веб-сайте Linux Foundation содержит большое количество информации о печати в Linux.
- http://www.cups.org/ – вебсайт CUPS предоставляет документацию, часто задаваемые вопросы и новостные группы о CUPS.



Файлы журналов – это файлы, которые содержат сообщения о системе, в том числе ядре, службах и приложениях к запущенных на ней. Существуют различные файлы журналов для различной информации. Например, файл журнала по умолчанию, файл журнала только для сообщений безопасности и файл журнала для задач планировщика **cron**.

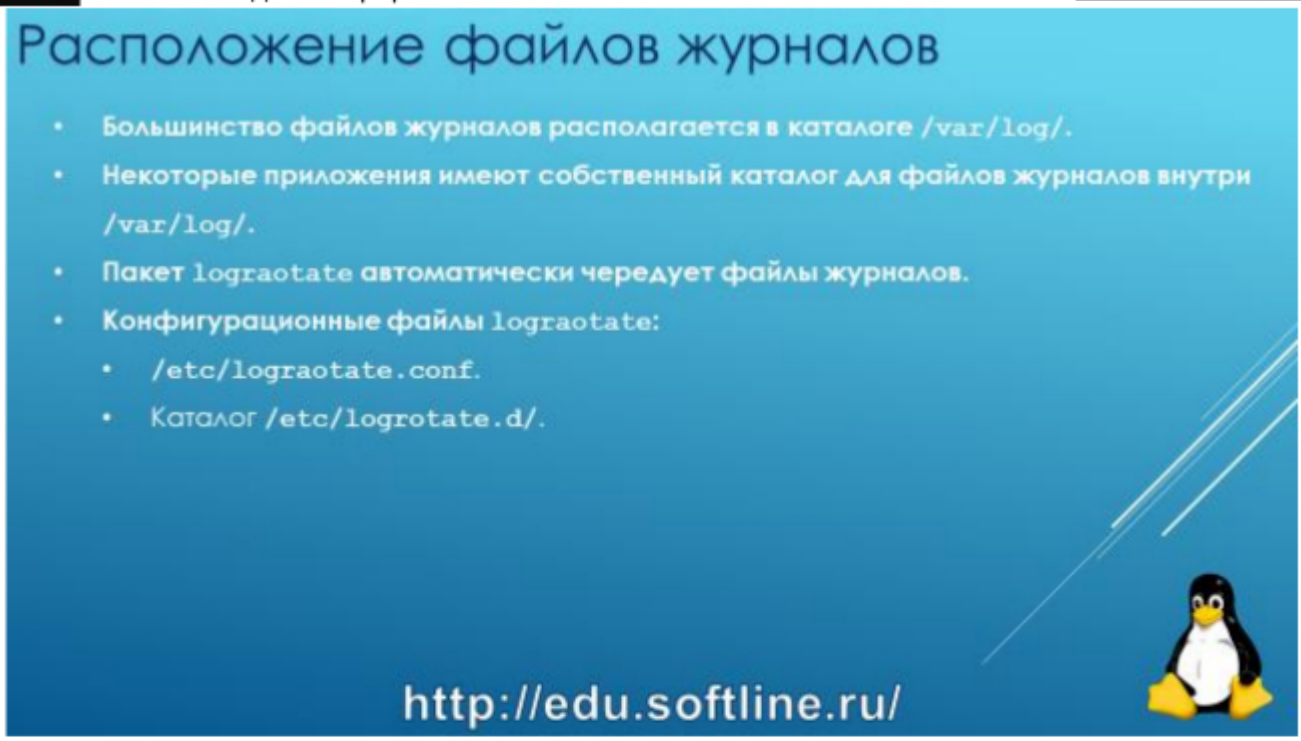
Файлы журналов могут быть очень полезными при устранении проблем с системой, таких как попытка загрузки драйвера ядра или при поиске неавторизованных попыток входа в систему.

Некоторые файлы журналов при помощи демона **rsyslod**. Демон **rsyslod** – это расширенная замена для **sysklod**, предоставляющая расширенную фильтрацию, защиту передачи сообщений при помощи шифрования, различные опции настройки, модули ввода и вывода, поддержку передачи при помощи протоколов TCP или UDP. Обратите внимание, что **rsyslod** совместим с **syslogkd**.

Также файлами журналов может управлять демон **journald** – компонент **systemd**. Демон **journald** захватывает сообщения **syslog**, сообщения журнала ядра, сообщение начального RAM-диска и начальной загрузки, а также сообщения, выводимые на стандартный вывод и стандартный вывод ошибок, от всех сервисов, индексирует их и делает доступными для пользователя. Собственный формат файла журнала, который является структурированным и индуцированным бинарным файлом, улучшает поиск и быстродействие операций, а также хранит метаданные, такие как штамп времени и идентификаторы пользователей. Файлы журналов, создаваемые **journald**, по умолчанию не постоянны, файлы журналов хранятся только в памяти или маленьком циклическом буфере в каталоге **/run/log/journal**. Объем данных журнала зависит от свободного места в памяти, при достижении лимита объема наиболее старые записи удаляются.

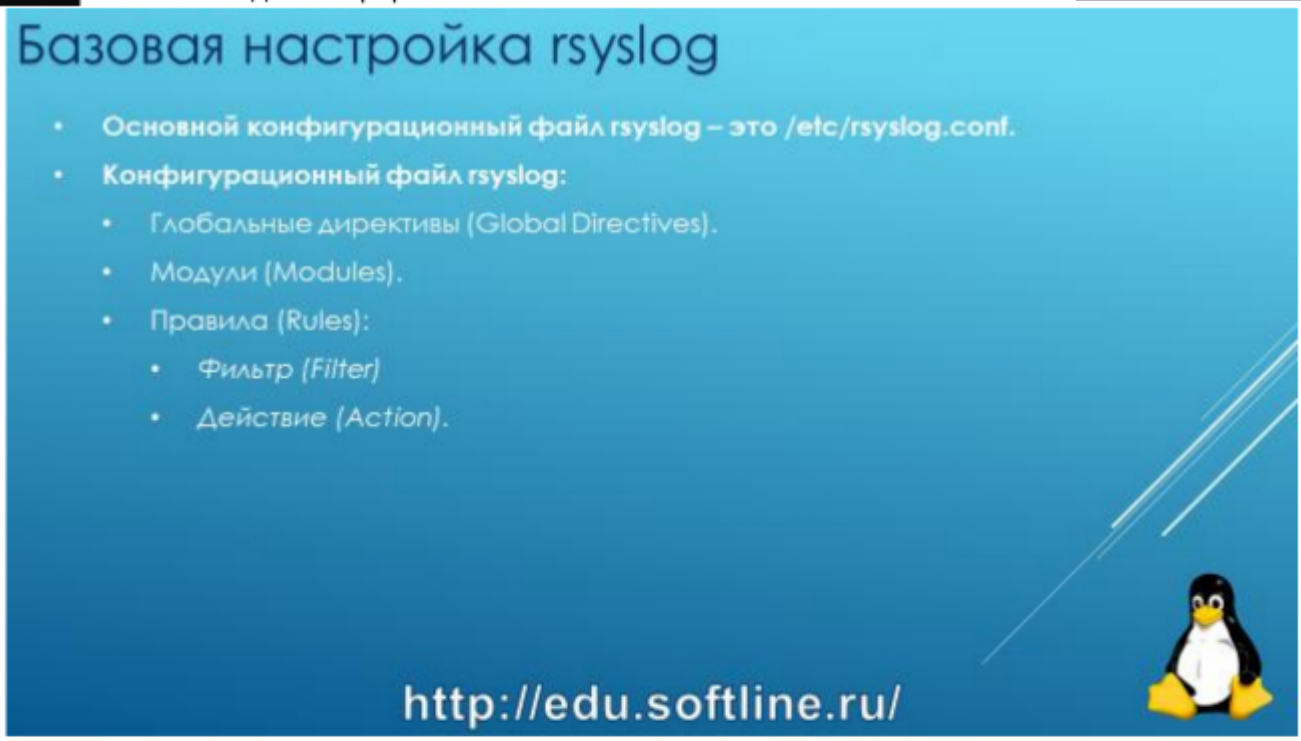
По умолчанию два инструмента ведения журнала сосуществуют на системе. Демон **journald** – это основной инструмент для устранения неисправностей. Он также предоставляет данные, необходимые для создания сообщений журнала. Полученные при помощи **journald** данные перенаправляются в сокет **/run/systemd/journal/syslog**, который может быть использован **rsyslod** для дальнейшей обработки данных. Однако, **rsyslod** по умолчанию, имеет настоящую интеграцию через модуль ввода **imjournal**, таким образом избегая использование вышеупомянутого сокета. Можно передавать данные в обратном направлении из **rsyslod** в **journald** при помощи

модуля **omjournal**. Интеграция позволяет поддерживать текстовые файлы журналов в целостном формате и убедиться в совместимости с возможными приложениями и конфигурациями, зависящими от **rsyslod**.

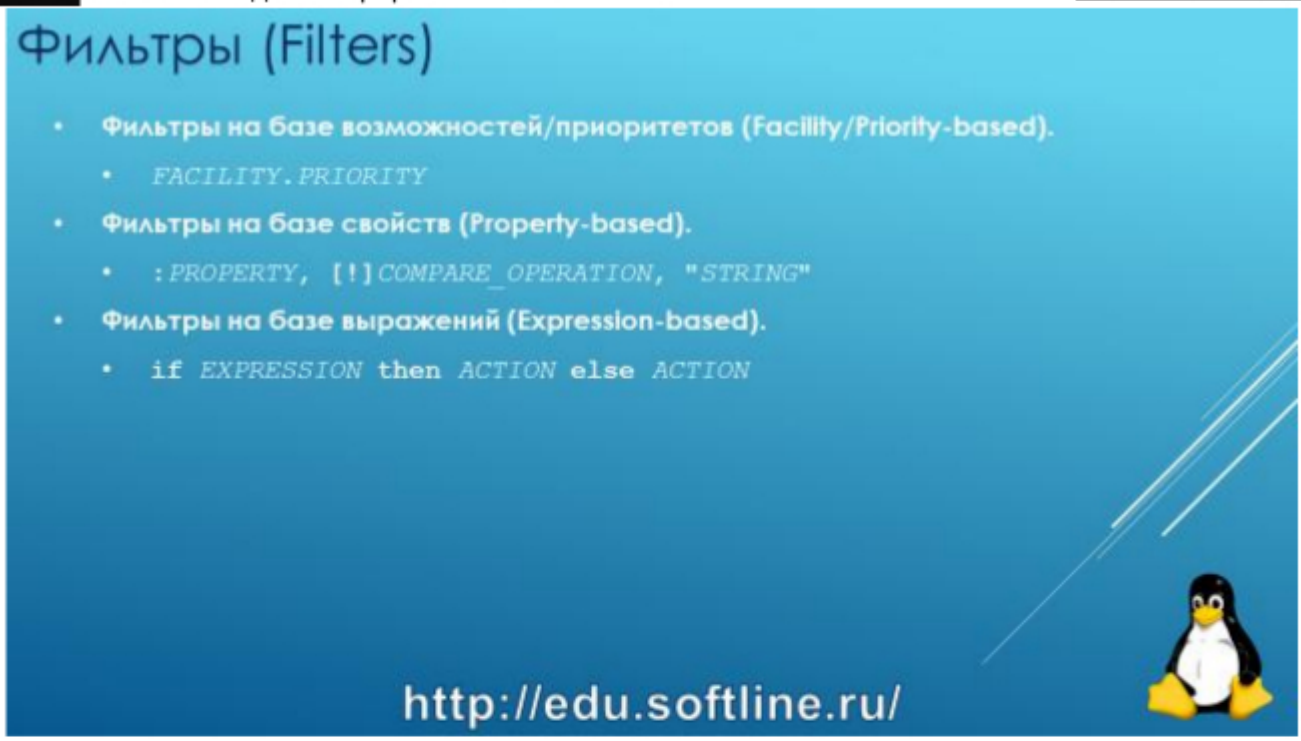


Список файлов журналов, поддерживаемых **rsyslogd** можно найти конфигурационном файле **/etc/rsyslogd.conf**. Большинство файлов журналов располагается в каталоге **/var/log/**. Некоторые приложения, такие как **httpd** и **samba** имеют собственный каталог для файлов журналов внутри **/var/log/**.

В каталоге **/var/log/** многие файлы имеют числа после имен (например **cron-20180911**). Эти числа представляют собой штамп времени, который добавляется к чередующимся файлам журналов. Файлы журналов чередуются для того чтобы размер файлов не был слишком большим. Пакет **lograotate** содержит задачу **cron**, которая автоматически чередует файлы журналов в соответствии с конфигурационным файлом **/etc/lograotate.conf** и конфигурационными файлами в каталоге **/etc/logrotate.d/**.



Основной конфигурационный файл **`rsyslog`** – это **`/etc/rsyslog.conf`**. В нём указываются глобальные директивы (Global Directives), модули (Modules) и правила (Rules), которые состоят из фильтра (Filter) и действия (Action). Дополнительно в конфигурационный файл можно добавить комментарий после знака решетка (**#**).



Правило (Rule) указывается при помощи двух частей. Первая часть – это фильтр (Filter) выбирает подмножество сообщений **syslog**, вторая часть – действие (Action), показывает, что делать с выбранными сообщениями. Для определения правила в конфигурационном файле **/etc/rsyslog.conf** необходимо определить обе части на одной строке и разделить их при помощи пробела или табуляции.

rsyslog предлагает различные пути для фильтрации сообщений syslog в соответствии с выбранными свойствами. Доступные методы фильтрации можно разделить на фильтрацию на базе возможностей/приоритетов (Facility/Priority-based), фильтрацию на базе свойств (Property-based) и фильтрацию на базе выражений (Expression-based).

Фильтры на базе возможностей/приоритетов (Facility/Priority-based).

Наиболее часто используемый и известный путь для фильтрации сообщений **syslog** – это использование фильтра на базе возможностей/приоритетов, который фильтрует сообщения syslog на базе двух условий: возможности (Facility) и приоритета (Priority), разделенных точкой. Для создания выборщиков используйте следующий синтаксис:

```
FACILITY.PRIORITY
```

Где:

FACILITY указывает подсистему, которая генерирует указанные сообщения **syslog**. Например, подсистема **mail** обрабатывает все сообщения **syslog**, связанные с почтой. *FACILITY* может быть представлена одним из следующих ключевых слов (или числовым кодом): **kern**(0), **user**(1), **mail**(2), **daemon**(3), **auth**(4), **syslog**(5), **lpr**(6), **news**(7), **uucp**(8), **cron**(9), **authpriv**(10), **ftp**(11) и от **local0** до **local7** (16 - 23).

PRIORITY указывает приоритет сообщения **syslog**. Приоритет может быть указан одним из следующих ключевых слов (или при помощи числа): **debug**(7), **info**(6), **notice**(5), **warning**(4), **err**(3), **crit**(2), **alert**(1) и **emerg**(0).

Вышеуказанный синтаксис выбирает сообщения **syslog** с указанным или более высоким приоритетом. Предшествующее знаку равно, любое ключевое слово приоритета указывает сообщения **syslog** с указанным приоритетом для выборки. Все остальные приоритеты будут проигнорированы. И наоборот, приоритет с предшествующим восклицательным знаком (**!**) выбирает все сообщения **syslog**, кроме сообщения с указанным приоритетом.

В дополнение к ключевым словам, указанным выше, можно использовать звездочку (*****) для определения возможностей или приоритетов (в зависимости от расположения звёздочки до или после точки). Ключевое слово приоритета **none** обозначает возможности без указания приоритета. Определения возможностей и приоритета чувствительны к регистру.

Для определения множества возможностей и приоритетов, необходимо разделить их при помощи запятой (**,**) для определения множества выборщиков на одной строке необходимо разделять их при помощи точки с запятой (**;**). Обратите внимание, что каждый выборщик в поле выборщика может переписать предшествующий, который мог исключить некоторые приоритеты из шаблона.

Примеры фильтров на базе возможностей/приоритетов.

Для выбора всех сообщений **syslog** от ядра с любым приоритетом добавьте следующий текст в конфигурационный файл:

```
kern.*
```

Для выбора всех почтовых сообщений **syslog** с приоритетом **crit** и выше используйте следующую форму записи:

```
mail.crit
```

Для выбора всех сообщений **cron** за исключением приоритетов **info** и **debug**, запишите конфигурацию в следующей форме:

```
cron.!info,!debug
```

Фильтры на базе свойств (Property-based).

Фильтры на базе свойств позволяют фильтровать сообщения **syslog** по любому свойству, такому как время генерации (**timegenerated**) или тегу **syslog** (**syslogtag**). Можно сравнивать каждое из указанных свойств с необходимым значением при помощи одной из операций сравнения. Имена свойств и операции сравнения чувствительны к регистру.

Фильтры на базе свойств должны начинаться с двоеточия (**:**). Для определения фильтра используется следующий синтаксис:

```
:PROPERTY, [!]COMPARE_OPERATION, "STRING"
```

где:

- Атрибут *PROPERTY* указывает необходимое свойство.
- Опциональный восклицательный знак (*!*) отрицает вывод операции сравнения. Прочие Булевы операторы на данный момент не поддерживаются в фильтрах на базе свойств.
- Атрибут *COMPARE_OPERATION* указывает одну или несколько операции сравнения, перечисленных в таблице ниже.

- Атрибут *STRING* указывает значение для сравнения с текстом предоставленным свойством. Данное значение может быть заключено в кавычки ("). Для исключения символов внутри строки, таких как кавычки (") используется символ обратного слеша (\).

Операции сравнения на базе свойств:

Операции сравнения	Описание
contains	Проверяет соответствие предоставленной строки любой части строки предоставленной свойством. Для выполнения не чувствительного к регистру сравнения используется **contains_i**.
isequal	Сравнивает предоставленную строку со всем текстом, предоставленным свойством. Данные два значения должны быть полностью одинаковыми.
startswith	Проверяет соответствие предоставленной строки с началом текста свойства. Для выполнения не чувствительного к регистру сравнения используется **startswith_i**.
regex	Сравнивает предоставленное базовое регулярное выражение POSIX (POSIX BRE) с текстом предоставленным свойством.
ereregex	Сравнивает предоставленное расширенное регулярное выражения POSIX (POSIX ERE) с текстом предоставленным свойством.
isempty	Проверяет, является ли свойство пустым (Empty). Это особенно полезно для нормализованных данных, где некоторые поля могут содержать результаты нормализации.

Примеры фильтров на базе свойств.

Для выбора сообщений **syslog**, которые содержат **Error** в тексте сообщения используется следующая директива:

```
:msg, contains, "error"
```

Следующий фильтр выбирает сообщения **syslog**, полученные от хаста с именем **c7-server02**:

```
:hostname, isequal, "c7-server02"
```

Для выбора сообщений **syslog**, которые не содержат никаких упоминаний о словах **Fatal** и **Error** с любым текстом или без текста между ними используется следующие директива:

```
:msg, !regex, "fatal .* error"
```

Фильтры на базе выражений (Expression-based).

Фильтры на базе выражений выбирают сообщения **syslog**, соответствующие определённой арифметической, сравнительной или строчной операции. Фильтры на базе выражений используют собственный язык написания скриптов **rsyslog**, называемый RainerScript, для построения комплексных фильтров.

Базовый синтаксис фильтра на базе выражения выглядит следующим образом:

```
if EXPRESSION then ACTION else ACTION
```

Где:

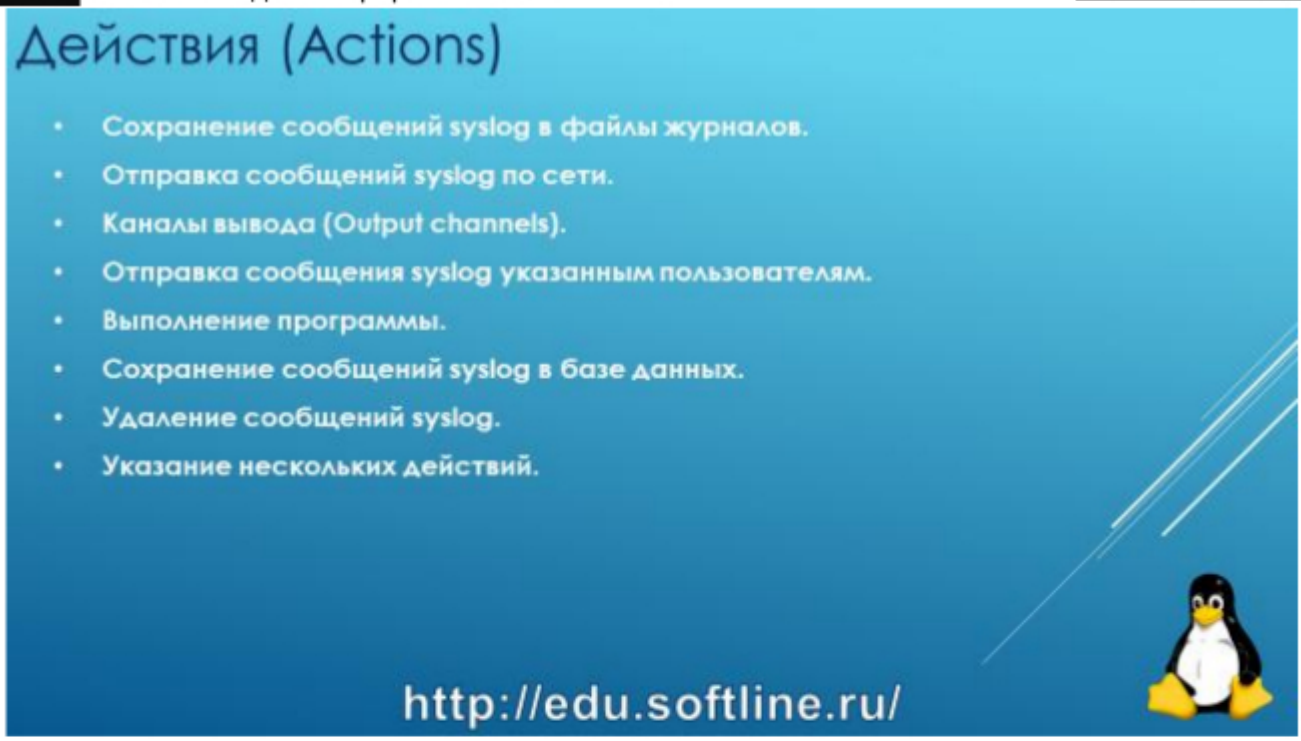
- Атрибут *EXPRESSION* представляет выражение, которое будет оценено, например: **$msg startswith 'DEVNAME' or $syslogfacility-text == 'local0'**. Можно указать более одного выражения в одном фильтре при помощи операторов **and** и **or**.
- Атрибут *ACTION* представляет действие, которое должно быть выполнено, если выражение возвращает значение **true**. Это может быть отдельное действие или произвольный комплексный скрипт, заключённый в фигурные скобки.
- Фильтры на базе выражений определяются при помощи ключевого слова **if** в начале новой строки. Ключевое слово **then** отделяет выражение от действия. Опционально можно применить ключевое слово **else** для указания действия на случай, когда условие не выполняется.

При помощи фильтров на базе выражений можно вложить множество условий при помощи скрипта, заключённого в фигурные скобки. Скрипт позволяет использовать фильтры на базе возможностей/приоритетов внутри выражения. С другой стороны, использовать фильтры на базе свойства не рекомендуется. RainerScript поддерживает регулярные выражения со специализированными функциями **re_match()** и **re_extract()**.

Примеры фильтров на базе выражений.

Следующее выражение содержит два вложенных условия. Файлы журналов, созданные при помощи программы, называемой **prog1**, разделяются на основе нахождения строки «**test**» в сообщении.

```
if $programname == 'prog1' then {
action(type="omfile" file="/var/log/prog1.log")
if $msg contains 'test' then
action(type="omfile" file="/var/log/prog1test.log")
else
action(type="omfile" file="/var/log/prog1notest.log")
}
```



Действия указывают, что необходимо сделать с сообщением, отобранным при помощи определённого выборщика.

Сохранение сообщений syslog в файлы журналов.

В большинстве случаев действие указывает в какой файл журнала будут сохранены сообщения **syslog**. Это выполняется при помощи указания пути к файлу после определенного выборщика.

```
FILTER PATH
```

Где *FILTER* используется для указания выборщика, а *PATH* – это путь к целевому файлу.

Например, следующее правило состоит из выборщика, который выбирает все сообщения **syslog** от **cron** и действия, которое сохраняет их в файл журнала **/var/log/cron.log**.

```
cron.* /var/log/cron.log
```

По умолчанию файл журнала синхронизируется каждый раз при генерации сообщения **syslog**. При помощи символа дефиса (**-**) перед именем файла указывается пропуск синхронизации.

```
FILTER -PATH
```

Обратите внимание, если система будет некорректно выключена сразу после попытки записи, это может привести к потере информации. Однако данный параметр может улучшить производительность, особенно если вы запускаете программы, которые генерируют очень подробные сообщения журнала.

Указанный путь может быть статическим (Static) или динамическим (Dynamic). Статические пути к файлам предоставляют фиксированный путь к файлу, как показано в примере выше. Динамические пути к файлам могут отличаться в зависимости от полученного сообщения. Динамические пути предоставляются при помощи шаблона и знака вопроса (**?**):

```
FILTER ?DYNAMIC-FILE
```

Где *DYNAMIC-FILE* – это имя заранее определенного шаблона, который модифицирует пути вывода. Можно использовать префикс дефис (**-**) для отключения синхронизации. Также можно использовать множество шаблонов, разделенных точкой с запятой (**;**).

Если указанный файл – это существующий терминал (Terminal) или устройство **/dev/console**, сообщения **syslog** отправляются на стандартный вывод (при помощи специального обработчика терминала) или на консоль (при помощи специального обработчика для **/dev/console**), когда используется X Windows System, соответственно.

Отправка сообщений syslog по сети.

rsyslog позволяет отправлять и получать сообщения **syslog** по сети. Даная возможность позволяет администрировать сообщения от множества хостов на одной машине. Для пересылки сообщений **syslog** на удалённую машину воспользуйтесь следующим синтаксисом:

@[(z*NUMBER***)]***HOST***:[***PORT***]**

Где:

- Знак собака (**@**) определяет, что сообщение **syslog** пересылаются на хост при помощи протокола UDP. Чтобы использовать протокол TCP, необходимо использовать два знака собака (**@@**).
- Опциональный параметр **z***NUMBER* включает сжатие **zlib** для сообщения **syslog**. Атрибут *NUMBER* указывает уровень компрессии (от минимального **0** до максимального **9**). Целесообразность сжатия автоматически проверяется **rsyslogd,** если сообщение после сжатия не становится меньше, все сообщения, чей размер меньше 60 байт, не сжимаются.
- Атрибут *HOST* указывает хост, который будет получать выбранные сообщения **syslog**.
- Атрибут *PORT* указывает порт на холсте.

При указании IPv6 адреса в качестве холста, адрес заключается в квадратные скобки (**[]**).

Примеры отправки сообщения **syslog** по сети.

Для отправки сообщений на хост с IP-адресом **192.168.0.1** по протоколу UDP, используйте:

```
*.* @192.168.0.1
```

Для отправки сообщений на хост example.com при помощи порта **6514** по протоколу TCP, используйте:

```
.* @@example.com:6514
```

Для отправки из сжатых сообщений при помощи **zlib** (с уровнем компрессии **9**) и пересылки их на хост **2001: db8::1** при помощи протокола UDP, используйте:

```
*.* @(z9)[2001:db8::1]
```

Каналы вывода (Output channels).

В основном, каналы вывода используются для указания максимального размера файла журналов, до которого можно расти. Это очень удобно для ротации файлов журналов. По больше части канал

вывода – это коллекция информации о выходных действиях. Каналы вывода определяются при помощи директивы **$outchannel**. Для определения канала вывода в **/etc/rsyslog.conf** воспользуйтесь следующим синтаксисом:

```
$outchannel NAME, FILE_NAME, MAX_SIZE, ACTION
```

где:

- Атрибут *NAME* указывает имя канала вывода.
- Атрибут *FILE_NAME* указывает имя файла вывода. Каналы вывода могут записывать только в файлы и не могут записывать в терминалы и другие виды вывода.
- Атрибут *MAX_SIZE* представляет максимальный размер указанного файла (*FILE_NAME*), до которого можно расти. Данное значение указывается в байтах.
- Атрибут *ACTION* указывает действие, которое выполняется при достижении максимального размера, определённого в *MAX_SIZE*.

Для использования определенного канала вывода в качестве действия внутри правила используется следующий синтаксис:

```
FILTER :omfile:$NAME
```

Примеры ротации журналов при помощи канала вывода.

В первую очередь канал вывода определяется при помощи директивы **$outchannel**:

```
$outchannel log_rotation, /var/log/test_log.log, 104857600,
/home/joe/log_rotation_script
```

Зачем необходимо использовать его в правиле:

```
.* :omfile:$log_rotation
```

При достижении лимита (в примере это 100 MB), выполняется скрипт **/home/joe/log_rotation_script**. Данный скрипт может содержать всё что угодно: перемещение файла в другой каталог, редактирование определённого контента или простого ударение.

Отправка сообщения syslog указанным пользователям.

rsyslog может отправлять сообщения **syslog** определённым пользователям, которым необходимо отправить сообщение, при помощи указания имени пользователя. Для указания нескольких пользователей используется разделитель – запятая (**,**). Для отправки сообщения всем пользователям, которые вошли в систему используется звездочка (*****).

Выполнение программы.

rsyslog позволяет выполнить программу для выбранных сообщений **syslog** и использовать вызов **system()** для выполнения программы в оболочке. Для указания программы, которая должна выполняться, используется префикс в виде символа крышки (**^**), далее необходимо указать шаблон, который форматирует полученное сообщение и передаёт его указанному исполняемому файлу в качестве параметра командной строки.

FILTER ^EXECUTABLE; TEMPLATE

Вывод условия *FILTER* обрабатывается программой *EXECUTABLE*. Данная программа может быть любым исполняемым файлом. *TEMPLATE*, заменяется именем шаблона форматирования.

Пример выполнения программы.

В примере ниже выбирается любое сообщение **syslog** с любым приоритетом, форматируется при помощи шаблона **template** и передается в качестве параметра в программу **test-program**, которая затем выполняется с предоставленным параметром:

```
*.* ^test-program;template
```



ПРИМЕЧАНИЕ:

Когда сообщение принимается от любого хоста, и используются действия выполнения в оболочке – система подвергается уязвимости инъекции команды. Атакующий может попробовать включить исполняемые команды в программу, указанную для выполнения в действии. Чтобы избежать подобных угроз безопасности настоятельно рекомендуется детально рассмотреть необходимость использования действия выполнения в оболочке.

Сохранение сообщений syslog в базе данных.

Выбранные сообщения syslog могут быть напрямую записаны в таблицу базы данных при помощи действия средства записи базы данных (Database Writer). Средство записи базы данных используют следующий синтаксис:

```
:PLUGIN:DB_HOST,DB_NAME,DB_USER,DB_PASSWORD;[TEMPLATE]
```

Где:

- *PLUGIN* вызывает указанную надстройку, которая обрабатывает запись в базу данных (например, настройка **ommysql**).
- Атрибут *DB_HOST* указывает имя хоста базы данных.
- Атрибут *DB_NAME* указывает имя базы данных.
- Атрибут *DB_USER* указывает имя пользователя базы данных.
- Атрибут *DB_PASSWORD* указывает пароль для пользователя базы данных.
- Атрибут *TEMPLATE* указывает опционально используемый шаблон, который изменяет сообщения **syslog**.



ПРИМЕЧАНИЕ:

*На данный момент **rsyslog** предоставляет поддержку только для MySQL (MariaDB) и PostgreSQL. Для использования средств записи баз данных MySQL и PostgreSQL необходимо установить пакеты **rsyslog-mysql** и **rsyslog-pgsql** соответственно. Дополнительно, необходимо указать загрузку соответствующих модулей в конфигурационном файле **/etc/rsyslog.conf**:*

```
module(load="ommysql") # Output module for MySQL support
module(load="ompgsql") # Output module for PostgreSQL support
```

*В качестве альтернативы можно использовать общий интерфейс баз данных предоставляемый модулем **omlibdb**. Данный модуль поддерживает Firebird, Interbase, MS SQL, Sybase, SQLite, Ingres, Oracle, mSQL.*

Удаление сообщений `syslog`.

Для удаления выбранных сообщений используется **`stop`**.

Действие удаления как правило используется для фильтрации сообщений перед выполнением последующей обработки. Это может быть эффективным если необходимо пропустить некоторые повторяющиеся сообщения, которые в противном случае заполнят файлы журналов. Результат действия удаления зависит от того, где оно указано в конфигурационном файле, для наилучшего результата необходимо разместить действие в начале списка действий. Обратите внимание, что если сообщение удалено, то оно не может быть получено в последующих строках конфигурационного файла.

Следующее правило удаляет все сообщения, которые соответствуют фильтру **`local5.*`**:

```
local5.* stop
```

В следующем примере удаляются все сообщения **`syslog`**, полученные от **`cron`**:

```
cron.* stop
```



ПРИМЕЧАНИЕ:
*В версиях, предшествующих **`rsyslog`** 7 символ Тильда (~) использовалась вместо **`stop`** для удаления сообщений **`syslog`**.*

Указание нескольких действий.

Для каждого выборщика, можно указать несколько действий. Для указания нескольких действий они записываются на отдельных строках, в начале строки указывается символ амперсанд (**`&`**):

```
FILTER ACTION
& ACTION
& ACTION
```

Указание нескольких действий улучшает общую производительность, так как исключает повторную проверку одного и того же фильтра выборщика.

Пример указания нескольких действий.

В следующем примере сообщения **`syslog`** от ядра с критическим (**crit**) приоритетом отправляются пользователю **`user1`**, обрабатываются шаблоном **`temp`** и передаются в исполняемый файл **`test-program`**, а также пересылаются на хост **`192.168.0.1`** по протоколу UDP.

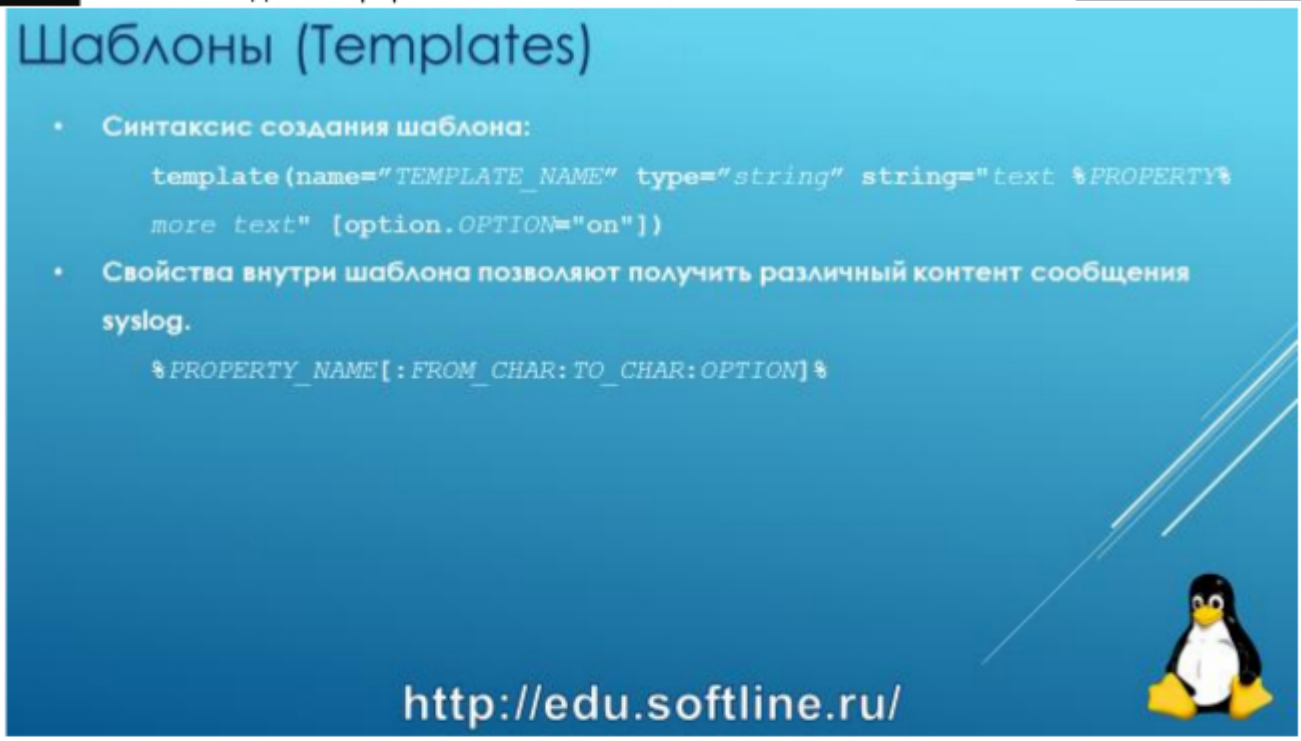
```
kern.=crit user1
& ^test-program;temp
& @192.168.0.1
```

Любое действие может сопровождаться шаблоном, который форматирует сообщение. Для указания шаблона, после действия устанавливается точка с запятой (**`;`**) и указывается имя шаблона.



ПРИМЕЧАНИЕ:

Шаблон должен быть определён перед его использованием в действии, иначе он игнорируется. Определения шаблонов всегда должны предшествовать определениям правил в **`/etc/ rsyslog.conf`**.



Любой вывод, который генерируется при помощи **rsyslog**, может быть изменён и отформатирован в соответствии с потребностями при помощи шаблонов (Templates). Для создания шаблона используется следующий синтаксис в конфигурационном файле **/etc/rsyslog.conf**:

```
template(name="TEMPLATE_NAME" type="string" string="text
%PROPERTY% more text" [option.OPTION="on"])
```

Где:

- **template()** – это директива представляющая блок определения шаблона.
- *TEMPLATE_NAME* – это обязательный аргумент, используемый для ссылки на шаблон. Все *TEMPLATE_NAME* должны быть уникальными.
- Обязательный аргумент **type** может принимать одно из следующих значений "**list**", "**subtree**", "**string**" или "**plugin**".
- Аргумент **string** - это актуальный текст шаблона. Вместе с этим текстом могут использоваться специальные символы, такие как **\n** для перехода на новую строку или **\r** для возврата каретки. Прочие символы такие как **%** или **"**, должны быть исключены или экранированы для буквального использования.
- Текст, указанный между двумя знаками процента (**%**), указывает свойство (Property), которое должно разрешить доступ к указанному контенту сообщения syslog.
- Атрибут *OPTION* указывают любую опцию, которая изменяет в функционал шаблона (Template). На данный момент поддерживаются следующие опции шаблона **sql** и **stdsql**, которые используются для форматирования текста запроса SQL или **json**, который форматирует текст таким образом, чтобы он подходил для обработки JSON и **casesensitive**, который указывает чувствительность к регистру имён свойств.



ПРИМЕЧАНИЕ:

*Обратите внимание, что Database Writer проверяет указание опций **sql** или **stdsql** в шаблоне. Если они не указаны, писатель базы данных (Database Writer) не выполняет никаких действий. Это позволяет предотвратить любые возможные угрозы безопасности, такие как инъекции SQL.*

Генерация динамических имен файлов.

Шаблоны могут быть использованы для генерации динамических имен файлов. При помощи указания свойства, как части пути файла, новый файл будет создан для каждого уникального свойства, что является удобным способом классификации сообщений syslog.

Пример использования свойства **timegenerated**, которое извлекает штамп времени из сообщения для генерации уникальных имен файлов каждого сообщения syslog:

```
template(name="DynamicFile" type="list") {
  constant(value="/var/log/test_logs/")
  property(name="timegenerated")
  constant(value"-test.log")
}
```

Учтите, что директива **$template** только указывает шаблон. Она должна быть использована внутри правила, чтобы от нее был эффект. В **/etc/rsyslog.conf** используется знак вопроса (**?**) в определении действия для указания шаблона динамических имён файлов:

```
*.* ?DynamicFile
```

Свойства.

Свойства внутри шаблона (между двумя знаками процента (**%**)) позволяют получить различный контент сообщения syslog через заменитель свойств (Property Replacer). Для определения свойства внутри шаблона (между кавычек ("…")) используется следующий синтаксис:

```
%PROPERTY_NAME[:FROM_CHAR:TO_CHAR:OPTION]%
```

Где:

- Атрибут *PROPERTY_NAME* указывает имя свойства. Список всех доступных свойств и их подробное описание можно найти на странице руководства **rsyslog.conf(5)** в разделе Available Properties.
- Атрибуты *FROM_CHAR* и *TO_CHAR* обозначают диапазон символов, которые указывают свойство, на которое нужно реагировать. В качестве альтернативы можно использовать регулярные выражения для указания диапазона символов. Чтобы это сделать, необходимо указать букву **R** как атрибут *FROM_CHAR* и указать необходимое регулярное выражение как атрибут *TO_CHAR*.
- Атрибут *OPTION* указывает любую опцию свойства, такую как опция **lowercase** для конвертации ввода в нижний регистр. Список всех доступных опций свойств и их детальное описание можно на странице руководства **rsyslog.conf(5)** в секции Property Options.

Примеры простых свойств:

- Следующее свойство принимает весь текст сообщения syslog:

  %msg%

- Следующее свойство принимает первые два символа текста сообщения syslog:

  %msg:1:2%

- Следующее свойство принимает весь текст сообщения syslog и удаляет символ перехода на новую строку (Line Feed Character):

  %msg:::drop-last-lf%

- Следующее сообщение принимает первые 10 символов штампа времени, который генерируется при получении сообщения syslog, и форматирует его в соответствии с стандартом даты RFC3999:

  %timegenerated:1:10:date-rfc3339%

Примеры шаблонов rsyslog.

Следующий шаблон форматирует сообщения syslog таким образом, чтобы выводить серьезность сообщения, средство (Facility), штамп времени, имя хоста, текст сообщения и заканчивать переходом на новую строку.

```
template(name="verbose" type="list") {
   property(name="syslogseverity")
   property(name="syslogfacility")
   property(name="timegenerated")
   property(name="HOSTNAME")
   property(name="syslogtag")
   property(name="msg")
   constant(value="\n")
}
```

Следующий шаблон напоминает традиционное настенное сообщение (сообщение, которое отправляется каждому пользователю, который вошел в систему и имеет право **mesg(1)**, установленное в значение **yes**). Данный шаблон выводит текст сообщения вместе с именем хоста, тегом сообщения и штампом времени на новой строке (при помощи **\r** и **\n**), а также звонит в колокол (при помощи **\7**).

```
template(name="wallmsg" type="list") {
   constant(value="\r\n\7Message from syslogd@")
   property(name="HOSTNAME")
   constant(value=" at ")
   property(name="timegenerated")
   constant(value=" ...\r\n ")
   property(name="syslogtag")
   constant(value=" ")
   property(name="msg")
   constant(value="\r\n")
}
```

Следующий шаблон форматирует сообщения syslog таким образом, чтобы оно могло использоваться как запрос базы данных. Обратите внимание на использование опции **sql** указанной в конце шаблона в качестве опции шаблона. Это сообщает писателю базы данных (Database Writer) форматировать сообщение как запрос SQL для MySQL.

```
template(name="dbFormat" type="list" option.sql="on") {
```

```
    constant(value="insert into SystemEvents (Message, Facility, FromHost,
    Priority, DeviceReportedTime, ReceivedAt, InfoUnitID, SysLogTag)")
    constant(value=" values ('")
    property(name="msg")
    constant(value="', ")
    property(name="syslogfacility")
    constant(value=", '")
    property(name="hostname")
    constant(value="', ")
    property(name="syslogpriority")
    constant(value=", '")
    property(name="timereported" dateFormat="mysql")
    constant(value="', '")
    property(name="timegenerated" dateFormat="mysql")
    constant(value="', ")
    property(name="iut")
    constant(value=", '")
    property(name="syslogtag")
    constant(value="')")
}
```

rsyslog также содержит набор заранее определенных шаблонов, определяемых при помощи префикса **RSYSLOG_**. Эти шаблоны зарезервированы для используя syslog, рекомендуется не создавать шаблонов с таким префиксом чтобы избежать конфликтов.

Следующий список отображает заранее определённые шаблоны вместе с их определениями:

RSYSLOG_DebugFormat – специальный формат, используемый для устранения проблем со свойствами.

```
template(name="RSYSLOG_DebugFormat" type="string" string="Debug line with all
properties:\nFROMHOST: '%FROMHOST%', fromhost-ip: '%fromhostip%', HOSTNAME:
'%HOSTNAME%', PRI: %PRI%,\nsyslogtag '%syslogtag%', programname:
'%programname%', APP-NAME: '%APP-NAME%', PROCID: '%PROCID%', MSGID:
'%MSGID%',\nTIMESTAMP: '%TIMESTAMP%', STRUCTUREDDATA: '%STRUCTURED-DATA%',\nmsg:
'%msg%'\nescaped msg: '%msg:::dropcc%'\nrawmsg: '%rawmsg%'\n\n")
```

RSYSLOG_SyslogProtocol23Format – формат, указанный в интернет проекте IETF ietf-syslog-protocol-23, который принят, чтобы стать новым стандартам RFC для syslog.

```
template(name="RSYSLOG_SyslogProtocol23Format" type="string" string="%PRI%1
%TIMESTAMP:::date-rfc3339% %HOSTNAME% %APP-NAME% %PROCID% %MSGID% %STRUCTURED-
DATA% %msg%\n ")
```

RSYSLOG_FileFormat – формат современного стиля файлов журналов, похожий на традиционный формат файлов журнала, но с высокой точностью штампа времени и информацией о временной зоне.

```
template(name="RSYSLOG_FileFormat" type="list") {
   property(name="timestamp" dateFormat="rfc3339")
   constant(value=" ")
   property(name="hostname")
   constant(value=" ")
   property(name="syslogtag")
   property(name="msg" spifno1stsp="on" )
   property(name="msg" droplastlf="on" )
   constant(value="\n")
}
```

RSYSLOG_TraditionalFileFormat – формат более старых журнала по умолчанию с низкой точностью штампов времени.

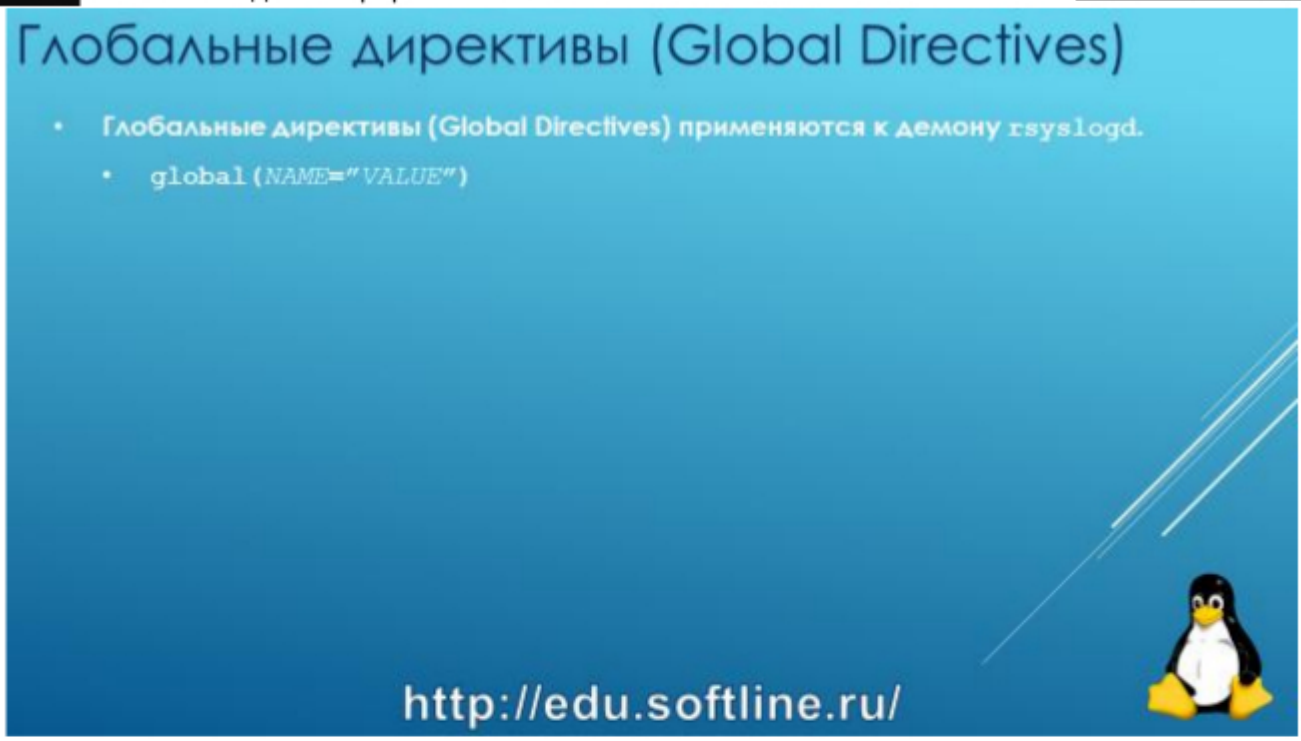
```
template(name="RSYSLOG_TraditionalFileFormat" type="list") {
   property(name="timestamp")
   constant(value=" ")
   property(name="hostname")
   constant(value=" ")
   property(name="syslogtag")
   property(name="msg" spifno1stsp="on" )
   property(name="msg" droplastlf="on" )
   constant(value="\n")
}
```

RSYSLOG_ForwardFormat – формат пересылки с высокой точностью штампов времени и информацией о временной зоне.

```
template(name="ForwardFormat" type="list") {
   constant(value="<")
   property(name="pri")
   constant(value=">")
   property(name="timestamp" dateFormat="rfc3339")
   constant(value=" ")
   property(name="hostname")
   constant(value=" ")
   property(name="syslogtag" position.from="1" position.to="32")
   property(name="msg" spifno1stsp="on" )
   property(name="msg")
}
```

RSYSLOG_TraditionalForwardFormat – традиционный формат пересылки с низкой точностью штампов времени.

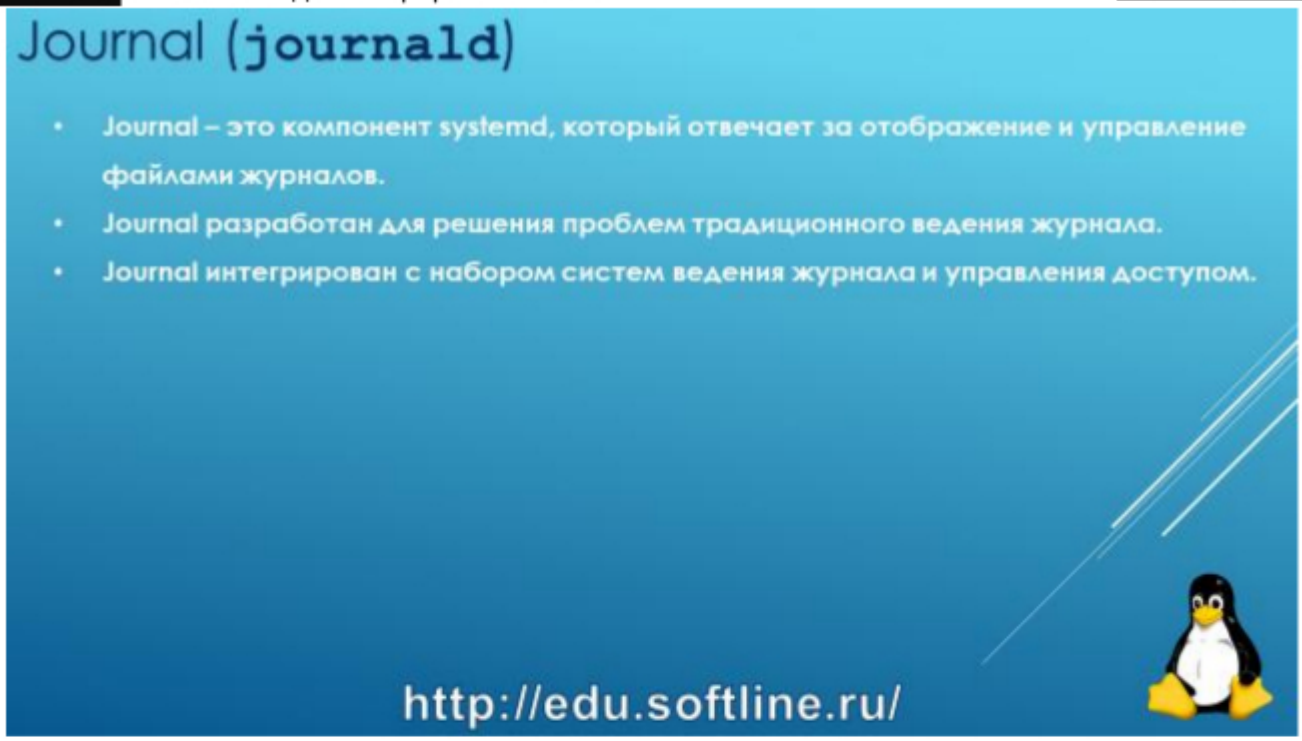
```
template(name="TraditionalForwardFormat" type="list") {
   constant(value="<")
   property(name="pri")
   constant(value=">")
   property(name="timestamp")
   constant(value=" ")
   property(name="hostname")
   constant(value=" ")
   property(name="syslogtag" position.from="1" position.to="32")
   property(name="msg" spifno1stsp="on" )
   property(name="msg")
}
```



Глобальные директивы (Global Directives) – это опции настройки, которые применяются к демону **rsyslogd**. Они обычно показывают для специфических, определённых заранее переменных, которые влияют на поведение демона **rsyslogd** или последующие правила. Все глобальные директивы заключены в блок настройки **global**. Пример глобальной директивы, которая указывает перезапись имени локального хоста для сообщений журнала:

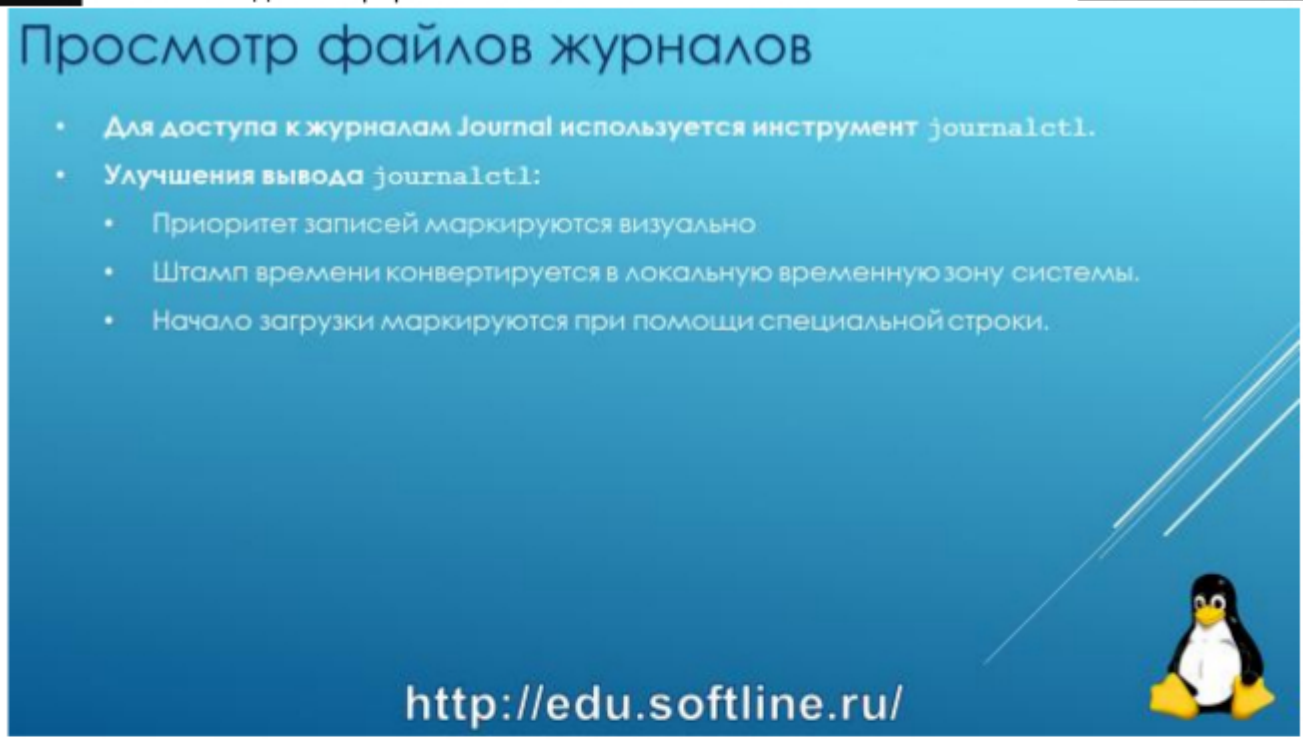
```
global(localHostname="machineXY")
```

Можно определить множество директив в конфигурационном файле **/etc/rsyslog.conf**. Директивы влияют на поведение всех опций настройки до тех пор, пока данная опция не указана еще раз с другим значением. Глобальный директивы могут быть использован для настройки действий, очередей и для отладки.



Journal – это компонент **systemd**, который отвечает за отображение и управление файлами журналов. Он может быть использован совместно или вместо традиционного демона **syslog**, такого как **rsyslogd**. Journal разработан для решения проблем, связанных традиционным ведением журнала. Он плотно интегрирован с набором систем, поддерживающих различные технологии ведения журнала и управления доступами к файлам журнала.

Данные ведения журнала накапливаются, хранятся и обрабатываются при помощи сервиса Journal – **jounrald**. Он создает и поддерживает бинарные файлы называемые журналы (Journals) на базе информации ведения журнала, которую получает от ядра, пользовательских процессов, со стандартного вывода и стандартного вывода ошибок системных сервисов или через стандартный API. Данные журналы структурированы и индексированы, что обеспечивает относительно высокую скорость поиска. Записи Journal могут обладать уникальным идентификатором. Сервис **journald** собирает набор полей метаданных для каждого сообщения журнала. Актуальные файлы журнала защищены и поэтому не могут быть отредактированы вручную.



Для доступа к журналам Journal используется инструмент **journalctl**. Для базового просмотра журналов необходимо ввести следующую команду от имени root:

```
# journalctl
```

Вывод данной команды – список всех сгенерированных файлов журналов на системе в том числе сообщений сгенерированных системными компонентами и пользователями. Структура вывода похожа на ту, которая используется в **/var/log/messages** но с важными улучшениями:

- Приоритет записей маркируются визуально. строки с приоритетом ошибка (Error) и выше выделяются красным цветом, а строки с приоритетом предупреждение (Warning) выделяются жирным шрифтом.
- Штамп времени конвертируется в локальную временную зону системы.
- Все данные, попавшие в журнал, отображаются, в том числе из журналов, прошедших ротацию.
- Начало загрузки маркируются при помощи специальной строки.

Пример вывода **journalctl**:

```
# journalctl
-- Logs begin at Sat 2018-09-22 10:32:35 +06, end at Mon 2018-09-24 15:34:40 +06
Sep 22 10:32:35 mainserver01 systemd-journal[85]: Runtime journal is using 4.0M
Sep 22 10:32:35 mainserver01 kernel: Initializing cgroup subsys cpuset
Sep 22 10:32:35 mainserver01 kernel: Initializing cgroup subsys cpu
Sep 22 10:32:35 mainserver01 kernel: Initializing cgroup subsys cpuacct
Sep 22 10:32:35 mainserver01 kernel: Linux version 3.10.0-862.11.6.el7.x86_64 (b
Sep 22 10:32:35 mainserver01 kernel: Command line: BOOT_IMAGE=/vmlinuz-3.10.0-86
lines 1-7
```

Во многих случаях, необходимо только несколько последних записей из журналов. Простейший способ сократить вывод **journalctl** это использовать опцию **-n** которая отображает только указанное число из самых последних записей:

```
# journalctl -n 10
```

Команда **journalctl** позволяет контролировать форму вывода при помощи следующего синтаксиса:

```
# journalctl -o FORM
```

Замените *FORM* на ключевое слово, указывающее необходимую форму вывода. Существует несколько опций, таких как **verbose**, который возвращает полностью структурированные элементы записей со всеми полями; **export**, который создается бинарный поточный файл, подходящий для резервного копирования и передачи по сети; а также **json**, который форматирует записи формате структуры данных JSON. Полный список поддерживаемых ключевых слов доступен на странице руководства **journalctl(1)**.

Пример подробного вывода **journalctl**:

```
# journalctl -o verbose -n 1
-- Logs begin at Sat 2018-09-22 10:32:35 +06, end at Wed 2018-09-26 08:15:00 +06
Wed 2018-09-26 08:15:00.087333 +06 [s=32f351cbd8634635bb020d0cb7f5337d;i=2bab;b=
    _TRANSPORT=syslog
    PRIORITY=6
    _SYSTEMD_SLICE=system.slice
    _BOOT_ID=2cfe7201072942ce954f1b055142f78b
    _MACHINE_ID=880be116fd0547f5b1b46e01c0c8e4c2
    _HOSTNAME=mainserver01
    SYSLOG_FACILITY=3
    SYSLOG_IDENTIFIER=named
    SYSLOG_PID=947
    _PID=947
    _UID=25
    _GID=25
    _COMM=named
    _EXE=/usr/sbin/named
    _CMDLINE=/usr/sbin/named -u named -c /etc/named.conf
    _CAP_EFFECTIVE=1000400
    _SYSTEMD_CGROUP=/system.slice/named.service
    _SYSTEMD_UNIT=named.service
    MESSAGE=error (network unreachable) resolving 'wd-prod-ss-eu-west-2-fe.weste
    _SOURCE_REALTIME_TIMESTAMP=1537928100087333
lines 1-22/22 (END)
```

Полное описание всех возможных полей доступно на странице руководства **systemd.journal-fields(7)**.

Управление доступом.

По умолчанию пользователи Journal без привилегий root могут просматривать только сгенерированные самим пользователем файлы журналов. Для того чтобы предоставить выбранным права доступа ко всем файлам журналов необходимо их добавить в группу **adm**.

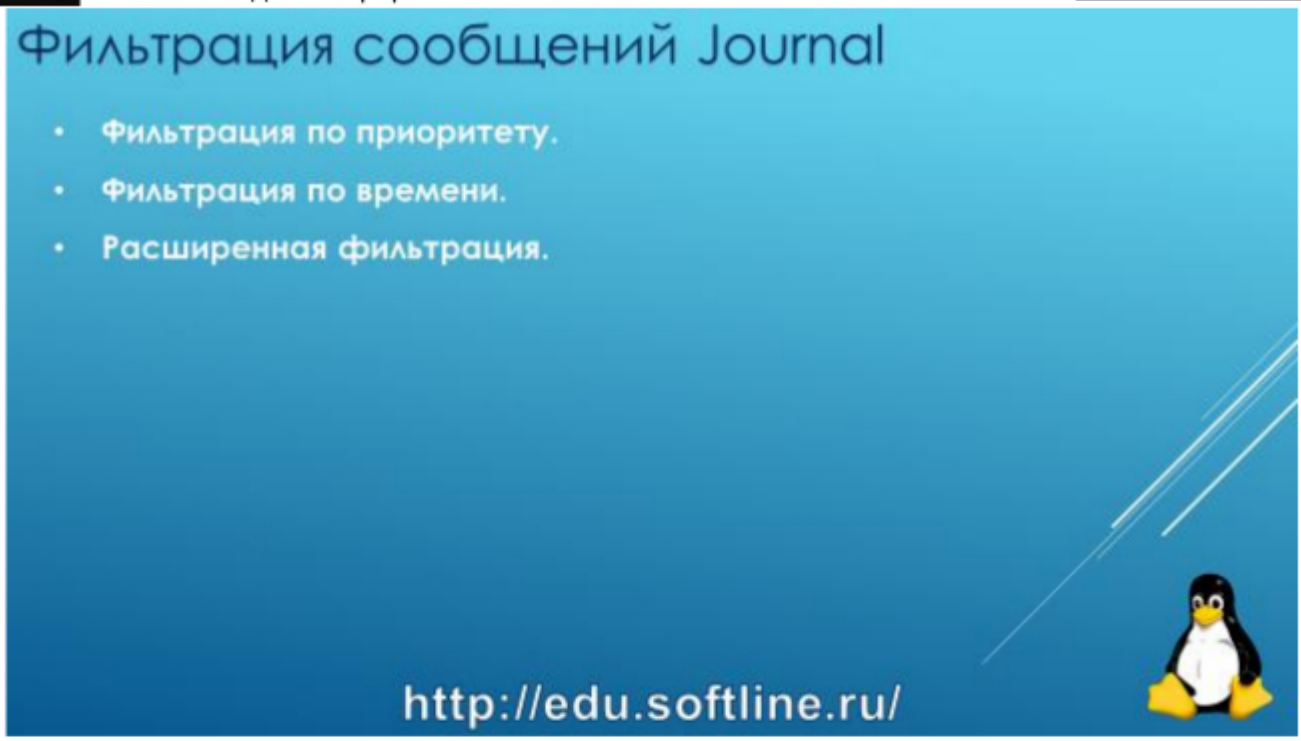
```
# usermod -a -G adm ИМЯ_ПОЛЬЗОВАТЕЛЯ
```

Управление доступом к Journal работает только в том случае если включена постоянное хранение (Persistent Storage).

Использование живого просмотра (Live View).

При вызове **journalctl** без параметров, отображается полный список сообщений журнала, начинающийся с самых старых собранных записей. При помощи живого просмотра (Live View) можно отслеживать сообщения журнала в реальном времени по мере их добавление в журнал. Для запуска **journalctl** в режиме живого просмотра (Live View) введите следующую команду:

```
# journalctl -f
```



Вывод команды **journalclt** без параметров очень большой, поэтому присутствует возможность использовать различные методы фильтрации для извлечения необходимой информации.

Фильтрация по приоритету.

Сообщения журнала часто используются для отслеживания ошибочного поведение на системе. Для просмотра записей с указанным приоритетом и выше используется следующий синтаксис:

```
# journalctl -p PRIORITY
```

необходимо заменить *PRIORITY* при помощи одного из ключевых слов (или при помощи цифры): **debug** (7), **info** (6), **notice** (5), **warning** (4), **err** (3), **crit** (2), **alert** (1), **emerg** (0).

Фильтрация по времени.

Для просмотра записей журнала, начиная с текущей загрузки, используется ключ **-b**:

```
# journalctl -b
```

Если перезагрузка прошла случайно, то **journalctl** не сократит вывод. В таком случае используется фильтрация на базе времени:

```
# journalctl --since=value --until=value
```

При помощи ключей **--since** и **--until** можно просмотреть сообщение журнала только в рамках указанного диапазона. Можно передавать значения в формате даты, времени или обоих, как показано в примере ниже:

```
# journalctl -p warning --since="2018-9-16 23:59:59"
```

Расширенная фильтрация.

Пример подробного вывода **journalctl**:

```
# journalctl -o verbose -n 1
-- Logs begin at Sat 2018-09-22 10:32:35 +06, end at Wed 2018-09-26 08:15:00 +06
Wed 2018-09-26 08:15:00.087333 +06 [s=32f351cbd8634635bb020d0cb7f5337d;i=2bab;b=
    _TRANSPORT=syslog
    PRIORITY=6
    _SYSTEMD_SLICE=system.slice
    _BOOT_ID=2cfe7201072942ce954f1b055142f78b
    _MACHINE_ID=880be116fd0547f5b1b46e01c0c8e4c2
    _HOSTNAME=mainserver01
    SYSLOG_FACILITY=3
    SYSLOG_IDENTIFIER=named
    SYSLOG_PID=947
    _PID=947
    _UID=25
    _GID=25
    _COMM=named
    _EXE=/usr/sbin/named
    _CMDLINE=/usr/sbin/named -u named -c /etc/named.conf
    _CAP_EFFECTIVE=1000400
    _SYSTEMD_CGROUP=/system.slice/named.service
    _SYSTEMD_UNIT=named.service
    MESSAGE=error (network unreachable) resolving 'wd-prod-ss-eu-west-2-fe.weste
    _SOURCE_REALTIME_TIMESTAMP=1537928100087333
lines 1-22/22 (END)
```

Как показано в примере подробного вывода, он выводит набор полей указанной записи журнала, по этим полям можно выполнить фильтрацию. Для получения полного описания метаданных который может хранить **systemd** Обратитесь к странице руководства **systemd.journal-fields(7)**. Эти метаданные собираются для каждого сообщения журнала без участия пользователя. Значения, как правило текстовые, но также могут быть бинарными и достаточно большими, также поля могут иметь несколько значений, хотя это не очень распространено.

Для просмотра списка уникальных значений, которые хранится в указанном поле, используется следующий синтаксис:

```
# journalctl -F FIELDNAME
```

Для отображения записей журнала, которые содержат определенное значение в указанном поле, используется следующий синтаксис:

```
# journalctl FIELDNAME=VALUE
```



ПРИМЕЧАНИЕ:

*Так как число полей метаданных, хранимых в **systemd**, достаточно большое, можно попросту забыть точное имя необходимого поля. В таком случае необходимо ввести команду **journalctl** и дважды нажать клавишу **<Tab>**:*

```
# journalctl
```

*Аналогично, для получения списка уникальных значений для поля, можно ввести имя поля, поставить знак равно (=) и дважды нажать **<Tab>**:*

```
# journalctl _HOSTNAME=
```

Можно указать множество значений для одного поля:

```
# journalctl FIELDNAME=VALUE1 FIELDNAME=VALUE2 ...
```

Указание двух значений для одного поля обрабатывается как логическое комбинация ИЛИ и записи с полем, соответствующим первому или второму значению отображаются.

Также есть возможность указать несколько полей для сокращения в выводимого набора:

```
# journalctl FIELDNAME1=VALUE FIELDNAME2=VALUE ...
```

Если указаны два разных поля они будут объединяться при помощи логического И, в результате будут отображаться только те записи у которых оба указанных поля соответствуют указанным значениям.

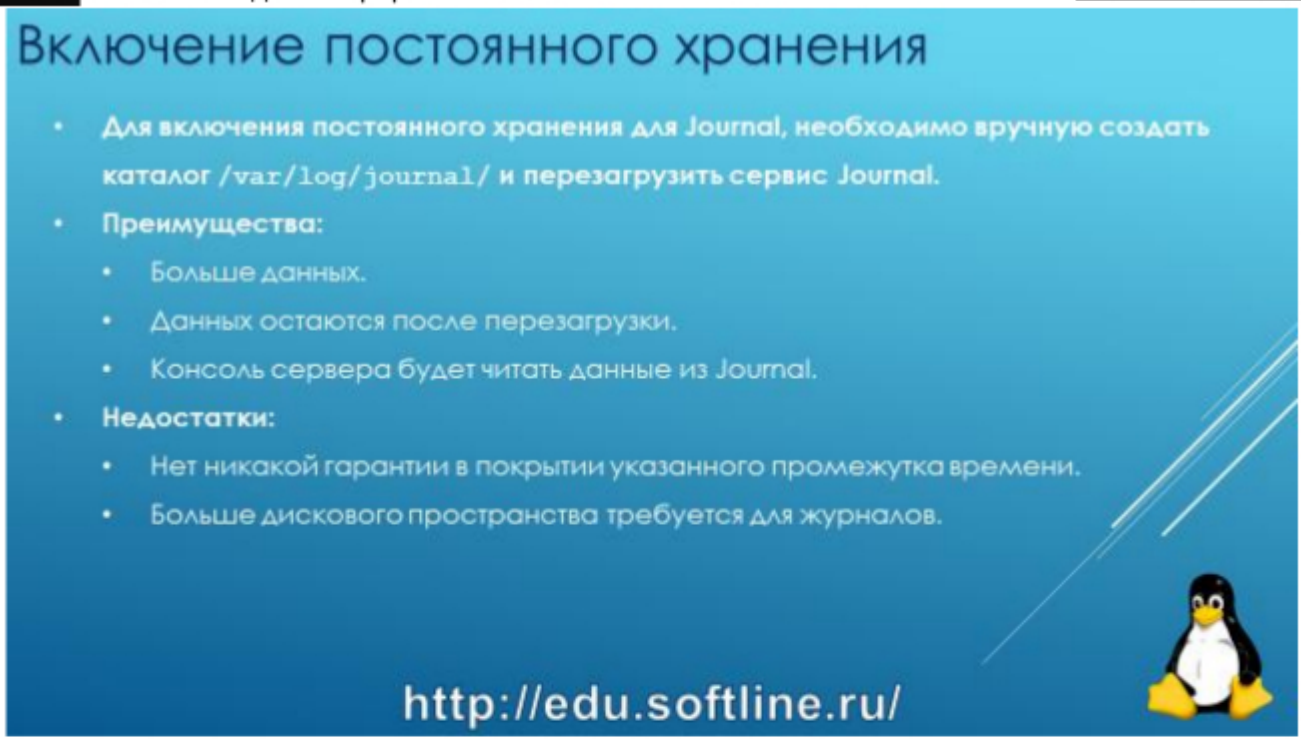
Для того чтобы заменить логическое и на логическое или используется знак плюс (+):

```
# journalctl FIELDNAME1=VALUE + FIELDNAME2=VALUE ...
```

Данная команда возвращает записи, которые соответствуют хотя бы одному из указанных условий.

Любые фильтры можно применять к режиму живого просмотра (Live View), для отображения только в выбранной группы записи журнала:

```
# journalctl -f FIELDNAME=VALUE ...
```



По умолчанию Journal хранит файлы журналов только в памяти или маленьком циклическом буфере в каталоге **/run/log/journal/**. Данный виртуальный каталог является энергозависимым и данные журналов не сохраняются после перезагрузки. При помощи конфигурация по умолчанию, syslog читает журналы Journal и сохраняет их в каталоге **/var/log/**. При помощи включения постоянного хранения, файлы Journal сохраняются в каталоге **/var/log/journal**, это значит, что они останутся после перезагрузки.

Включение постоянного хранения имеет следующие преимущества:

- Больше данных записывается для устранения неисправностей на более продолжительный период времени.
- Для немедленного устранения неисправностей, больше данных остаются доступными после перезагрузки.
- Консоль сервера будет читать данные из Journal, а не из файлов журнала.

Постоянное хранение также имеет некоторые недостатки:

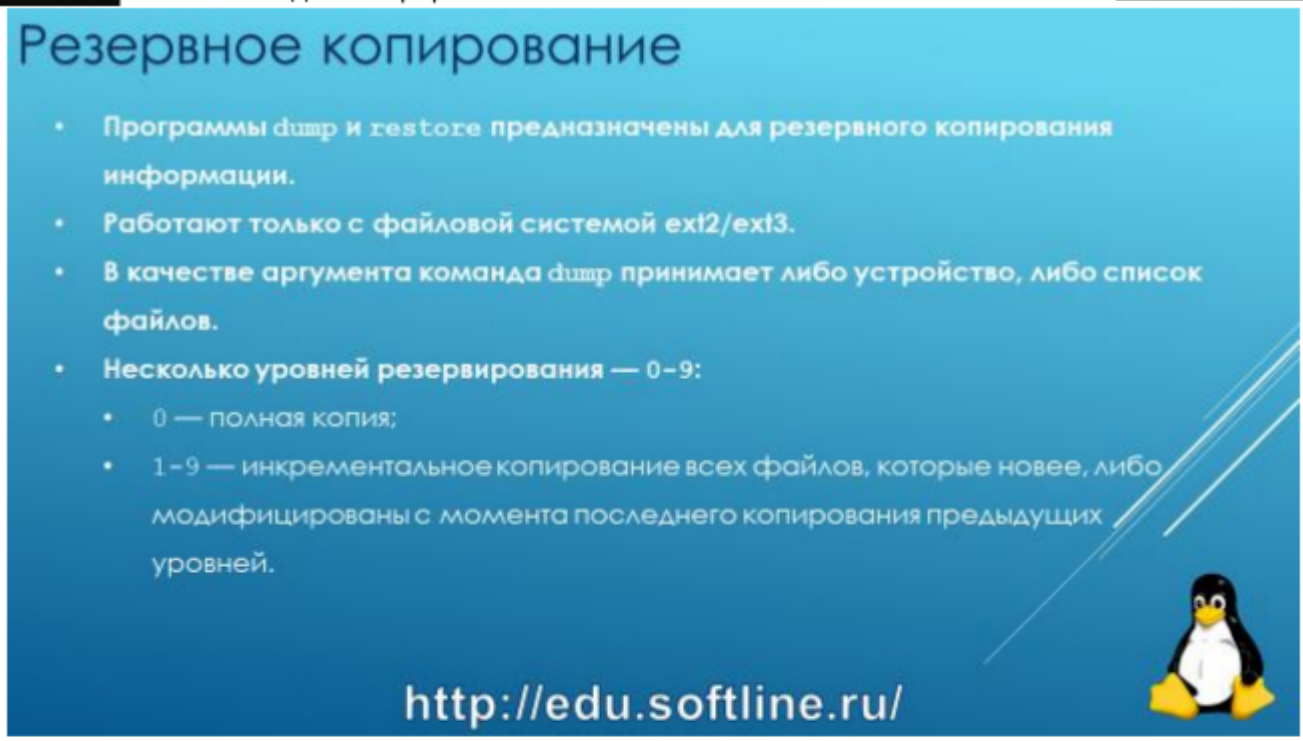
- Даже при включенном постоянном хранении, объём хранимых данных зависит от свободной памяти, нет никакой гарантии в покрытии указанного промежутка времени.
- Больше дискового пространства требуется для журналов.

Для включения постоянного хранения для Journal, необходимо вручную создать каталог **/var/log/journal/**:

```
# mkdir -p /var/log/journal/
```

После этого необходимо перезагрузить сервис Journal для применения изменений:

```
# systemctl restart systemd-journald
```



В настоящее время существует огромное множество различного программного обеспечения, выполняющего функцию резервирования данных. Программы **dump** и **restore** являются одними из представителей таких программных продуктов.

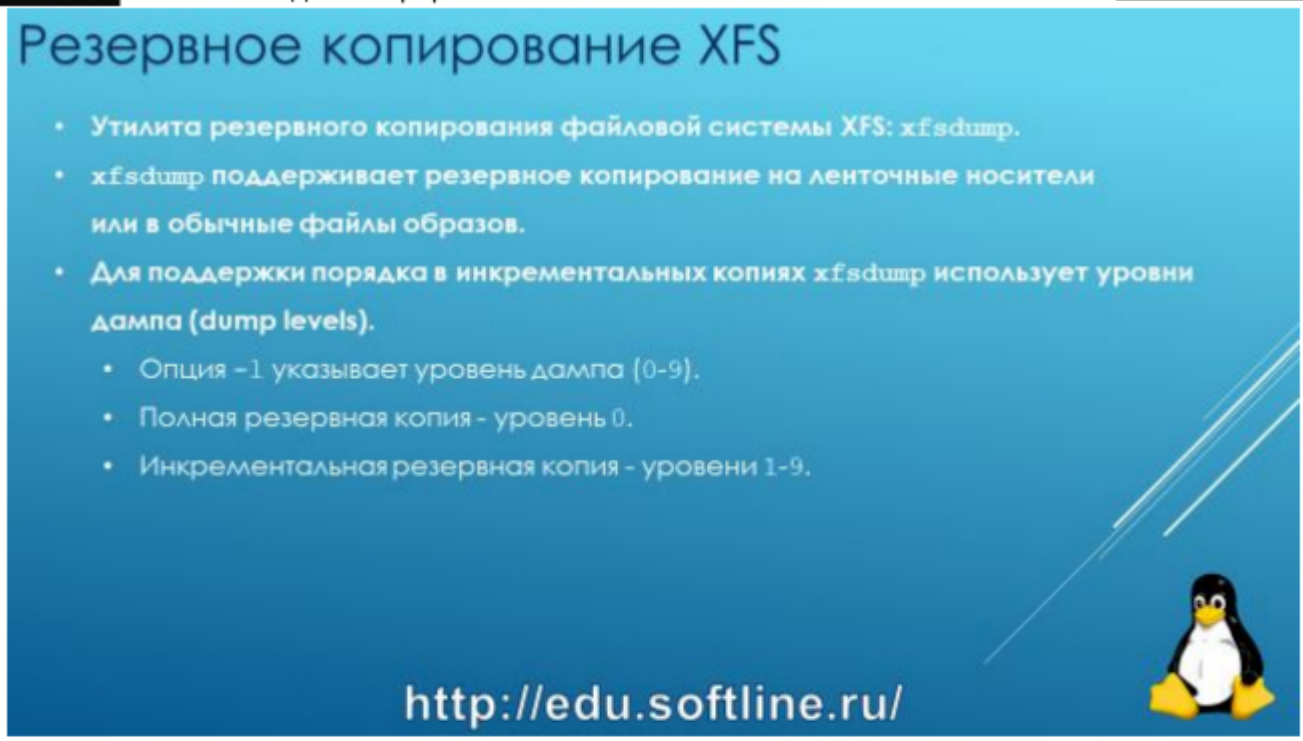
Программа dump предназначена для проверки файловых систем ext2/3/4 и определения файлов, которые необходимо сохранить. Найдены файлы могут быть на любое устройство для хранения информации, жёсткий диск, магнитная лента и так далее. Если размер архивной копии окажется больше отводимого свободного места, то архив будет поделён на части. Основной параметр, принимаемый программой — уровень резервирования, уровень 0 означает полное инкрементальное резервирование. По умолчанию используется уровень 9.

Создать резервную копию раздела **/** можно следующей командой:

```
# dump -0u -f /dev/sdb1 /
```

Программа restore восстанавливает данные, используя дамп:

```
# restore -i -d /dev/sdb1
```



Резервное копирование и восстановление файловой системы XFS происходит при помощи двух утилит: **xfsdump** и **xfsrestore**.

Для резервного копирования или дампа файловой системы XFS, используется утилита **xfsdump**. RHEL 7/CentOS 7 поддерживает резервное копирование на ленточные носители или обычные файлы образов, также поддерживается несколько дампов на одной ленте. Утилита **xfsdump** также позволяет разделить дамп на несколько лент, а в обычный файл может быть записан только один дамп. Дополнительно **xfsdump** поддерживает инкрементальные резервные копии и может исключать файлы из резервной копии на базе фильтра по размеру, пути или флагу иноды.

Для поддержки порядка в инкрементальных копиях **xfsdump** использует уровни дампа (dump levels) для определения базового дампа к которому относится указанный дамп. Опция **-l** указывает уровень дампа (**0**-**9**). Для выполнения полной резервной копии на файловой системе (**/path/to/filesystem**) используйте уровень дампа **0**:

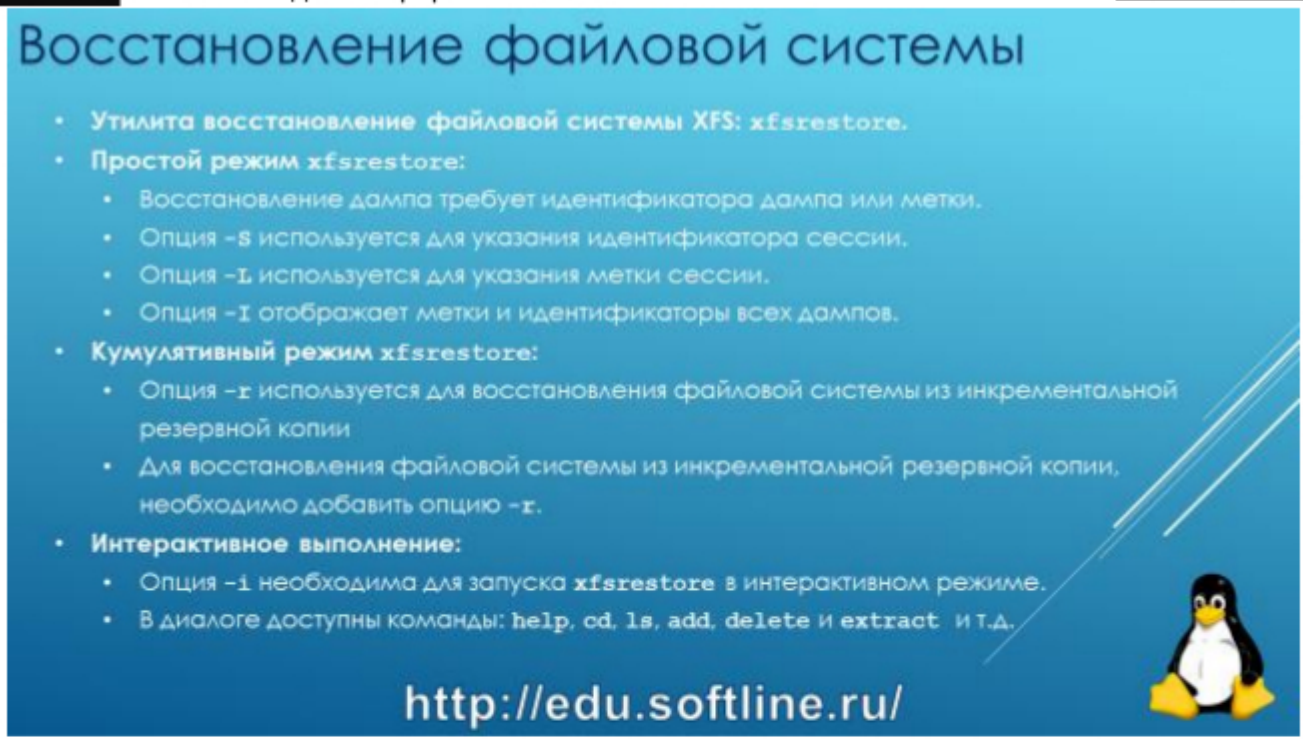
```
# xfsdump -l 0 -f /dev/device /path/to/filesystem
```

ПРИМЕЧАНИЕ:
Опция **-f** указывает назначение для резервной копии. Например, назначение **/dev/st0** обычно используется для ленточных накопителей. Назначением **xfsdump** может быть ленточный накопитель, обычный файл или удаленное ленточное устройство.

Инкрементальная резервная копия содержит дамп только тех файлов, которые изменились с момента последнего уровня дампа. Уровень дампа 1 является первой инкрементальной копией после полного резервного копирования. Следующий инкрементальный дамп получит уровень 2 и так далее. Максимальный уровень 9.

Выполнение дампа 1-го уровня на ленточный накопитель:

```
# xfsdump -l 1 -f /dev/st0 /path/to/filesystem
```



С другой стороны, утилита **xfsrestore** восстанавливает файловую систему из дампа, созданного при помощи **xfsdump**. Утилита **xfsrestore** имеет два режима: по умолчанию простой (simple) режим и кумулятивный (cumulative). Дампы идентифицируются по идентификатору сессии (session ID) или по метке сессии (session label). Восстановление дампа требует идентификатора дампа или метки. Для отображения идентификаторов и меток всех дампов (полных и инкрементальных), используйте опцию **-I**.

```
# xfsrestore -I
```

Пример вывода:

```
file system 0:
fs id: 45e9af35-efd2-4244-87bc-4762e476cbab
session 0:
mount point: bear-05:/m nt/test
device: bear-05:/dev/sdb2
time: Fri Feb 26 16:55:21 2010
session label: "my_dump_session_label"
session id: b74a3586-e52e-4a4a-8775-c3334fa8ea2c
level: 0
resum ed: NO
subtree: NO
stream s: 1
stream 0:
      pathnam e: /mnt/test2/backup
      start: ino 0 offset 0
      end: ino 1 offset 0
      interrupted: NO
      media files: 1
      media file 0:
            mfile index: 0
            mfile type: data
            mfile size: 21016
            mfile start: ino 0 offset 0
            mfile end: ino 1 offset 0
            media label: "my_dump_media_label"
```

```
                media id: 4a518062-2a8f-4f17-81fd-bb1eb2e3cb4f
xfsrestore: Restore Status: SUCCESS
```

Простой режим `xfsrestore`.

Простой режим **`xfsrestore`** позволяет пользователям восстанавливать файловую систему целиком из дампа с уровнем 0. После идентификации уровня дампа 0, указывается идентификатор сессии (в примере: ***`session-ID`***) и путь для полного восстановления на **`/path/to/destination`** при помощи:

```
# xfsresto re -f /dev/st0 -S session-ID /path/to/destination
```



ПРИМЕЧАНИЕ:

Опция **`-f`** указывает расположение дампа, опции **`-S`** или **`-L`** указывают какой именно дамп восстанавливать. Опция **`-S`** используется для указания идентификатора сессии, опция **`-L`** используется для указания метки сессии. Опция **`-I`** отображает метки и идентификаторы всех дампов.

Кумулятивный режим `xfsrestore`.

Кумулятивный режим **`xfsrestore`** позволяет восстановление файловой системы из указанной инкрементальной резервной копии, с уровня 1 по 9. Для восстановления файловой системы из инкрементальной резервной копии, необходимо добавить опцию **`-r`**.

```
# xfsrestore -f /dev/st0 -S session-ID -r /path/to/destination
```

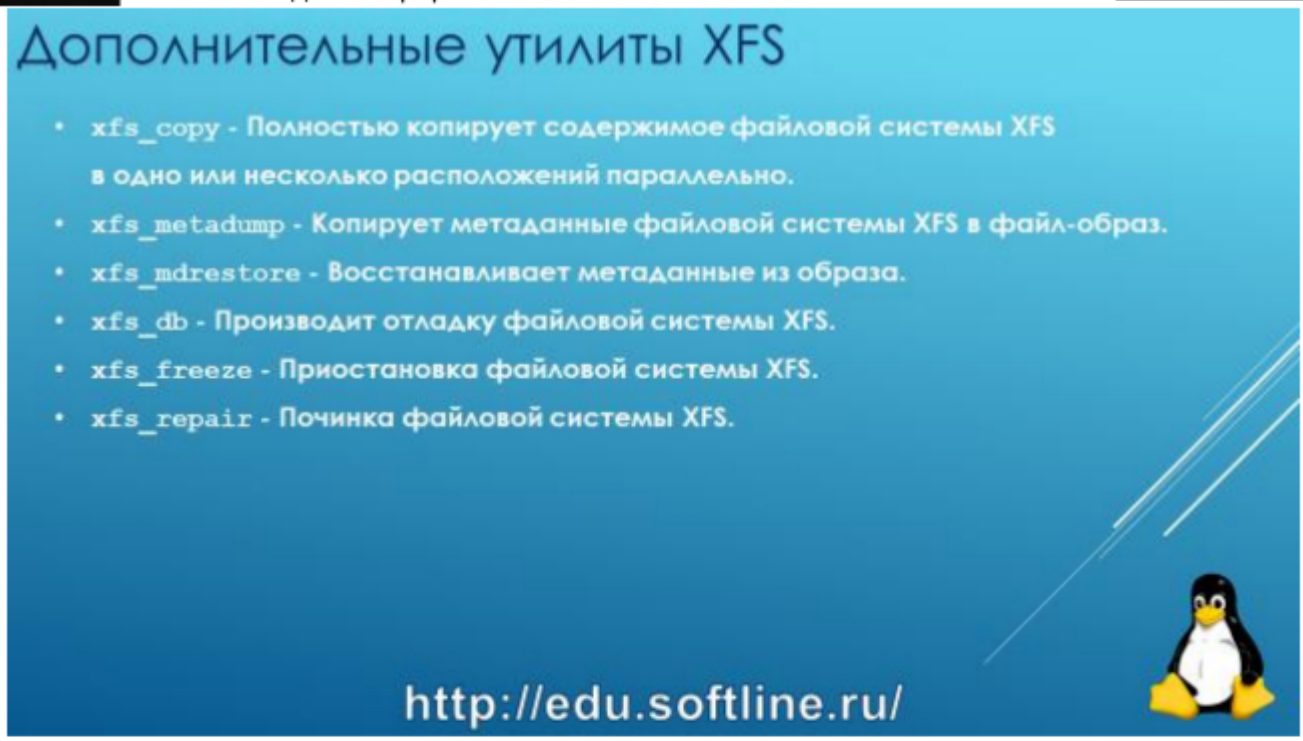
Интерактивное выполнение.

Утилита **`xfsrestore`** позволяет извлечение, добавление и удаление отдельных файлов из дампа. Для использования **`xfsrestore`** в интерактивном режиме используйте опцию **`-i`**.

```
xfsrestore -f /dev/st0 -i /destination/directory
```

Интерактивный диалог начнется сразу после того, как **`xfsrestore`** закончит чтение указанного устройства. В диалоге доступны следующие команды: **`cd`**, **`ls`**, **`add`**, **`delete`** и **`extract`**, для получения полного списка используйте **`help`**.

За дополнительной информацией о резервном копировании и восстановлении обратитесь к **`man`**: **`man xfsdump`** и **`man xfsrestore`**.



Прочие полезные утилиты для работы с файловой системой XFS.

RHEL 7/CentOS 7 также содержит дополнительные утилиты для управления файловой системой XFS.

`xfs_copy`

Полностью копирует содержимое файловой системы XFS в одно или несколько расположений параллельно.

Следующие утилиты полезны для отладки и анализа файловой системы XFS:

`xfs_metadump`

Копирует метаданные файловой системы XFS в файл. **`xfs_metadump`** поддерживает копирование метаданных только с отмонтированных файловых систем или файловых систем в режиме только для чтения, в противном случае данные могут быть повреждены или лишены целостности.

`xfs_mdrestore`

Восстанавливает метаданные из образа, сделанного при помощи **`xfs_metadump`**.

`xfs_db`

Производит отладку файловой системы XFS.

За дополнительной информацией по данным утилитам обратитесь к страницам **`man`**.

Приостановка файловой системы XFS.

Для приостановки и возобновления операций записи на файловой системе, используйте **`xfs_freeze`**. Приостановка операций записи, позволяет аппаратным устройствам создавать моментальные снимки, для целостного захвата файловой системы.



ПРИМЕЧАНИЕ:

Утилита **`xfs_freeze`** *предоставляется в пакете* **`xfsprogs`**, *который доступен только для архитектуры x86_64.*

Для приостановки (заморозки) файловой системы используйте:

```
# xfs_freeze -f /mount/point
```

Для возобновления (разморозки) файловой системы используйте:

```
# xfs_freeze -u /mount/point
```

Во время создания моментального снимка LVM, нет необходимости использовать **`xfs_freeze`** для приостановки файловой системы. Инструменты управления LVM самостоятельно приостанавливают файловую систему XFS.

Починка файловой системы XFS.

Для починки файловой системы XFS используйте **`xfs_repair`**:

```
# xfs_repair /dev/device
```

Утилита **`xfs_repair`** обладает высокой степенью масштабирования и разработана для починки даже очень больших файловых систем с большим числом поврежденных инод. В отличии от остальных файловых систем **`Linux xfs_repair`** не запускается в момент загрузки, даже тогда, когда файловая система не была корректно отмонтированна. В случае некорректного отмонтирования **`xfs_repair`** просто воспроизводит журнал в момент монтирования, убеждаясь в целостности файловой системы.



ПРИМЕЧАНИЕ:

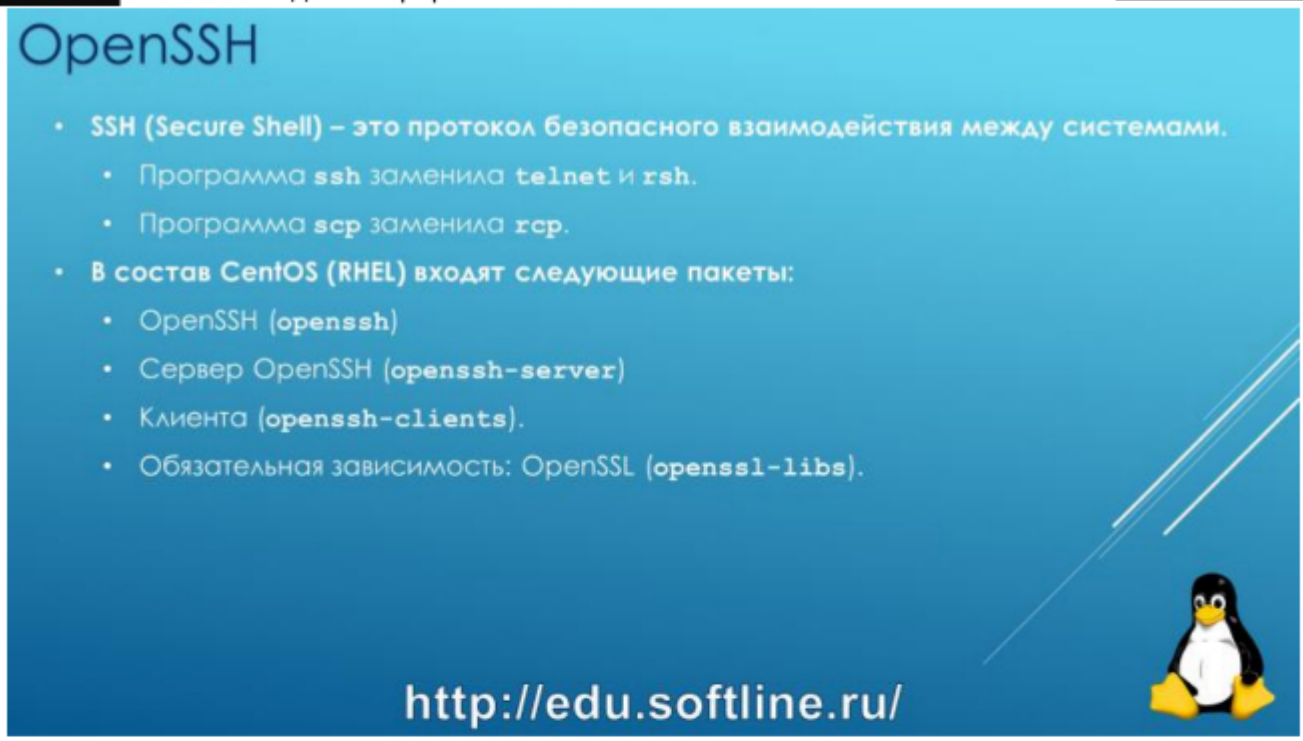
Утилита **`xfs_repair`** *не может починить файловую систему XFS при наличии грязного журнала. Для отчистки журнала смонтируйте и отмонтируйте файловую систему XFS.*

В случае повреждения журнала и его невозможности применения используйте ключ **`-L`** *(принудительное обнуление журнала) для отчистки журнала:*

```
# xfs_repair -L /dev/device
```

Будьте осторожны, в будущем это может привести к потере данных.

За дополнительной информацией обращайтесь к **`man xfs_repair`**.



SSH (Secure Shell) – это протокол, который обеспечивает безопасные коммуникации между двумя системами при помощи клиент-серверной архитектуры, а также позволяет пользователям удаленно входить на хостовые системы. В отличии от прочих протоколов удаленного взаимодействия, таких как FTP или telnet, SSH шифрует сессию входа, тем самым делая затруднительным перехват незашифрованного пароля.

Программа **ssh** разработана для замены более старых и менее безопасных терминальных приложений, используемых для входа на удаленную машину, таких как **telnet** и **rsh**. Связанная программа **scp** заменяет более старую программу **rcp**, разработанную для копирования файлов между хостами. В связи с тем, что эти старые программы не шифруют передачу пароля между клиентом и сервером, постарайтесь, по возможности, не использовать их. Использование безопасных методов входа на удаленные системы, снижает риски для клиентской и удаленной систем.

CentOS (RHEL) включает основной пакет OpenSSH (**openssh**), а также пакеты сервера OpenSSH (**openssh-server**) и клиента (**openssh-clients**). Обратите внимание, что пакеты OpenSSH требуют пакет OpenSSL (**openssl-libs**), который устанавливает несколько важных криптографических библиотек, позволяя OpenSSH предоставлять зашифрованные коммуникации.

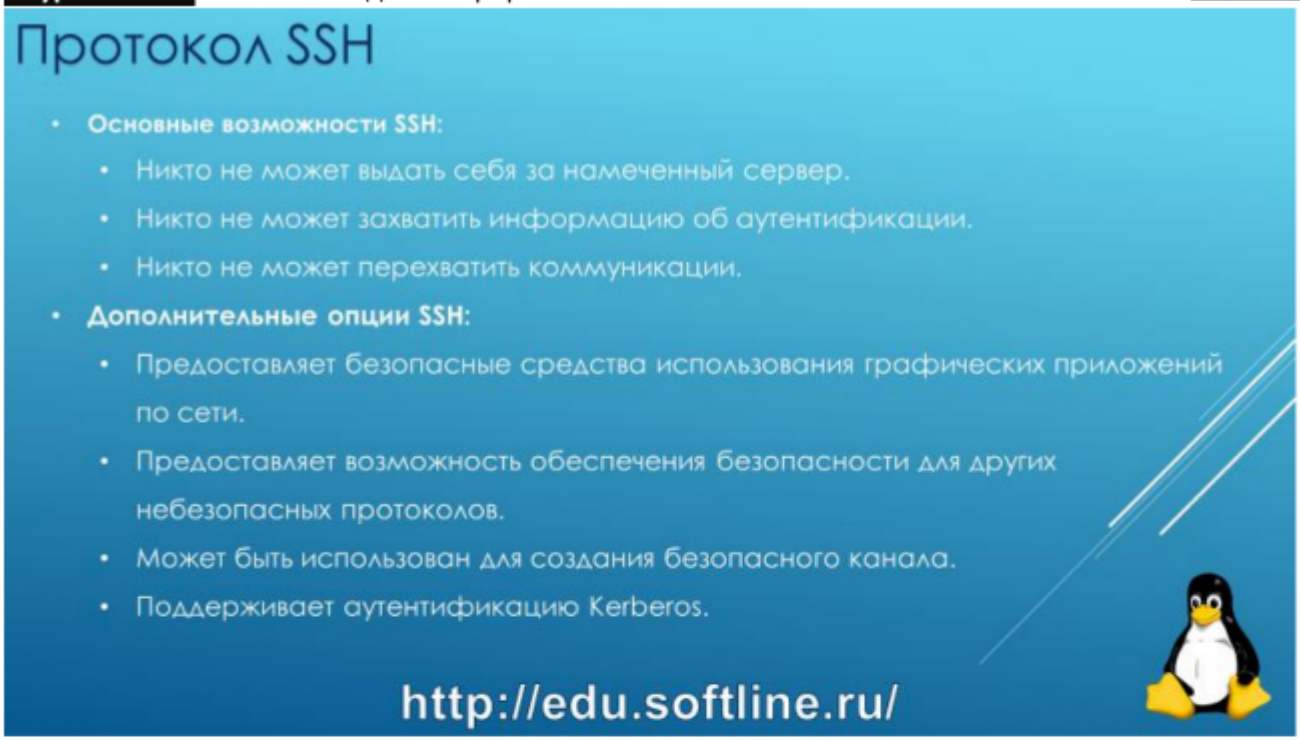
Версии протокола.

Две версии протокола SSH существуют на данный момент: версия 1 и более новая версия 2. Комплект OpenSSH, входящий в состав CentOS (RHEL), использует SSH версии 2, который использует улучшенный алгоритм обмена ключами, не подверженный известным уязвимостям первой версии. Однако, в целях обеспечения совместимости комплект OpenSSH также обеспечивает поддержку для соединений первой версии.



ПРИМЕЧАНИЕ:

Для обеспечения максимальной безопасности ваших подключений, рекомендуется использовать клиенты и серверы совместимые только со второй версией протокола SSH.



Причины использования SSH.

Потенциальные злоумышленники обладают различными инструментами для перехвата, подмены и перенаправления траффика в целях получения доступа к системе. В общем виде эти угрозы могут быть категорированы следующим образом:

Перехват коммуникаций между двумя хостами.

Злоумышленник может располагаться где угодно в сети, между взаимодействующими партнерами и копировать любую информацию, передаваемую между ними. Он может перехватывать и хранить ее или изменять и отправлять ее ожидающему получателю.

Подобная атака обычно выполняется при помощи сниффера пакетов (Packet Sniffer), это может быть практически любая сетевая утилита, способная захватывать каждый пакет, идущий через сеть и анализировать его содержимое.

Подмена одного из хостов.

Система злоумышленника может быть сконфигурирована таким образом, что будет представляться как получатель передачи. Если это сработает, то пользовательская система не узнает, что она взаимодействует с подставным хостом.

Подобная атака может быть выполнена через отравление DNS (DNS Poisoning) или через имитацию IP (IP Spoofing). В первом случае злоумышленник использует взломанный сервер DNS для перенаправления клиентских систем на дублирующий вредоносный хост. Во втором случае злоумышленник посылает фальшивые сетевые пакеты, как будто они отправлены с доверенного хоста.

Обе техники направлены на перехват чувствительной информацию и, в случае успеха злоумышленника, последствия могут быть катастрофическими. Если используется SSH для удаленного входа в оболочку и копирования файлов эти угрозы безопасности будут существенно снижены. Это связано с тем, что клиент и сервер SSH используют цифровые подписи для проверки идентичности. Дополнительно, все коммуникации меду клиентом и сервером шифруются. Попытка подмены

идентификатора на любой стороне не сработает, так как каждый пакет шифруется ключами, известными только на локальной и удаленной системе.

Основные возможности.

Протокол SSH обеспечивает следующее:

- Никто не может выдать себя за намеченный сервер.

  После начального подключения клиент может удостовериться, что подключается к тому же серверу, с которым соединялся в прошлый раз.

- Никто не может захватить информацию об аутентификации.

  Клиент передает информацию по аутентификации на сервер с использованием 128-битного шифрования.

- Никто не может перехватить коммуникации.

  Все данные отправленные и полученные в рамках сессии, передаются с использованием 128-битного шифрования, делая крайне сложным расшифровку и чтение перехваченной информации.

Дополнительно SSH предлагает следующие опции:

- Предоставляет безопасные средства использования графических приложений по сети.

  При помощи технологии перенаправления X11 (X11 Forwarding) приложений с сервера.

- Предоставляет возможность обеспечения безопасности для других небезопасных протоколов.

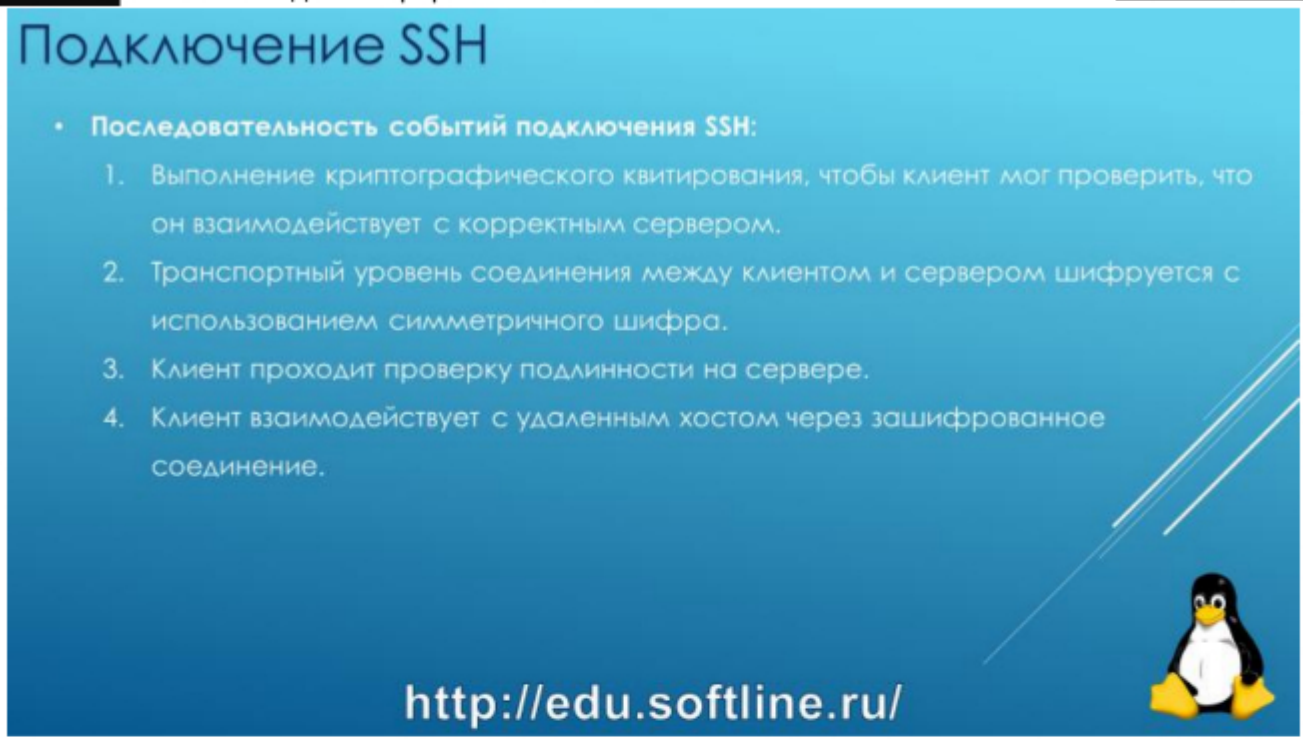
  Протокол SSH шифрует все что он отправляет и получает. При помощи техники перенаправления порта (Port Forwarding) сервер SSH может обеспечить безопасный канал для небезопасных протоколов, таких как POP, и увеличить безопасности системы и данных.

- Может быть использован для создания безопасного канала.

  Сервер и клиент OpenSSH могут быт настроены на создание тоннеля, подобному виртуальной частной сети (VPN), для траффика между серверной и клиентской машинами.

- Поддерживает аутентификацию Kerberos.

  Серверы и клиенты OpenSSH могут быть настроены на аутентификацию с использованием GSSAPI (Generic Security Services Application Program Interface) внедрения сетевого протокола проверки подлинности Kerberos.



Следующая серия событий помогает защитить целостность коммуникаций SSH между двумя хостами:

1. Выполнение криптографического квитирования, чтобы клиент мог проверить, что он взаимодействует с корректным сервером.
2. Транспортный уровень соединения между клиентом и сервером шифруется с использованием симметричного шифра.
3. Клиент проходит проверку подлинности на сервере.
4. Клиент взаимодействует с удаленным хостом через зашифрованное соединение.

Транспортный уровень.

Основная роль транспортного уровня – это упрощение безопасного и защищенного взаимодействия между двумя хостами в процессе аутентификации и последующего взаимодействия. Транспортный уровень выполняет это за счет шифрования и расшифровывания данных и обеспечения защиты целостности пакетов данных в процессе отправки и получения. Транспортный уровень также предоставляет сжатие, ускоряю передачу информации.

Когда клиент SSH контактирует с сервером, происходит обмен информацией ключей, таким образом, чтобы две системы могли корректно построить транспортный уровень. Следующие шаги происходят во время обмена:

- Обмен ключами.
- Определение алгоритма шифрования публичны ключом.
- Определение алгоритма симметричного шифрования.
- Определение алгоритма аутентификации сообщений.
- Определение алгоритма хеширования.

В процессе обмена ключами, сервер определяет себя для клиента при помощи уникального ключа хоста (Host Key). Если клиент ранее не взаимодействовал с данным сервером, ключ хоста сервера не известен клиенту, и он не может соединиться. OpenSSH решает эту проблему принимая ключ хоста сервера. Это выполняется после уведомления пользователя с последующем принятием и проверки

нового ключа хоста. При последующих подключениях, ключ хоста сервера будет проверятся с версией, сохраненной на клиенте, тем самым гарантируя, что клиент соединяется с тем же самым сервером. Если в будущем ключ хоста не совпадет, пользователь должен будет удалить сохраненную версию ключа, чтобы установить соединение.



ВНИМАНИЕ:

Это возможно для злоумышленника, замаскировать сервер SSH во время первой сессии, когда локальная система не знает различий между настоящим сервером и подставным. Чтобы избежать этого, убедитесь в корректности подключения к хосту, взаимодействуя с администратором сервера, при первом подключении к серверу.

SSH разработан с возможностью работы с поддержкой любых алгоритмов с использованием крытого ключа и форматов шифрования. После того как начальный обмен ключами создает значение хеш-функции, используемое для обмена значением общего секрета, две системы немедленно приступают к вычислению новых ключей и алгоритмов для защиты аутентификации и будущей передачи данных через подключение.

После передачи определенного объема данных с использованием определенного ключа и алгоритма (точный объем данных зависит от реализации SSH), еще один обмен ключами происходит, генерируется новый набор значений хеш-функций и значения общего секрета, данная информация будет применима на ограниченный период времени.

Аутентификация.

После построения транспортного уровня, безопасный туннель передает информацию между двумя системами, сервер сообщает клиенту различные поддерживаемые методы аутентификации, такие как использование шифрования подписи частным ключом или ввод пароля. Клиент пытается аутентифицировать на сервере с использованием одного из поддерживаемых методов.

SSH серверы и клиенты могут быть настроены для разрешения различных типов аутентификации, которые позволяют каждой стороне оптимальный объем контроля. Сервер может выбрать какой метод шифрования он поддерживает на базе собственной модели безопасности и клиент может выбрать порядок методов аутентификации исходя из доступных опций.

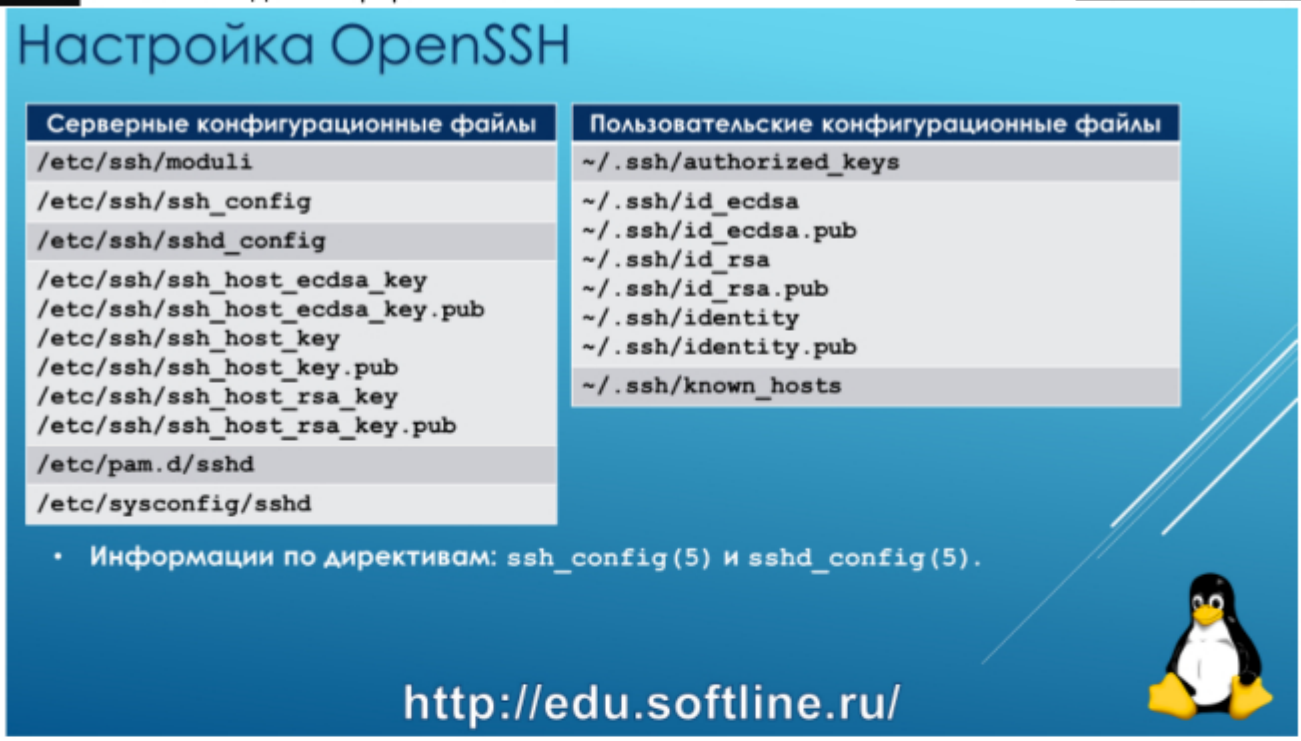
Каналы.

После успешной аутентификации через транспортный уровень SSH, несколько каналов открывается через технику, называемую мультиплексирование (Multiplexing). Каждый из этих каналов обрабатывает коммуникации для различных терминальных сессий и для перенаправления сессий X11.

Клиенты и серверы могут создавать новые каналы. Затем каждый канал назначает разный номер на каждый конец соединения. Когда клиент пытается открыть новый канал. Клиенты отправляют номер канала вместе с запросом. Это информация сохраняется на сервере и используется для прямого взаимодействия с каналом. Это делается для того, чтобы различные типы сессии не влияли друг на друга и чтобы, когда одна из сессий завершается, ее канал был закрыт без уничтожения основного соединения SSH.

Также каналы поддерживают управление потоком (Flow-Control), которое позволяет ему отправлять и принимать данные в организованном виде. В этом случае, данные не отправляются через канал до тех пор, пока клиент не получит сообщение, что этот канал открыт.

Клиент и сервер обсуждают характеристики каждого канала автоматически, в зависимости от типа сервиса, запрошенного клиентом и пути, по которому пользователь подключается к сети. Это обеспечивает потрясающую гибкость в обработке различных типов удаленных подключений без необходимости изменять базовую инфраструктуру протокола.



В OpenSSH существует два различных набора конфигурационных файлов: набор для клиентских программ (таких как **ssh**, **scp** и **sftp**) и набор для сервера (демона **sshd**).

Общая системная информация о конфигурации SSH хранится в каталоге **/etc/ssh/**, как описано в таблице ниже.

/etc/ssh/moduli – Содержит группы Диффи-Хеллмана (Diffie-Hellman), используемые для обмена ключами Диффи-Хеллмана, что является критичным для построения безопасного транспортного уровня. Когда обмен ключами совершен в начале сессии SSH, создается значение общего секрета. Это значение в дальнейшем используется для обеспечения аутентификации хоста.

/etc/ssh/ssh_config – Конфигурационный файл клиента SSH по умолчанию. Обратите внимание, что он переписывается файлом **~/.ssh/config**, если последний существует.

/etc/ssh/sshd_config – Конфигурационный файл демона **sshd**.

/etc/ssh/ssh_host_ecdsa_key – Частный ключ ECDSA, используемый демоном **sshd**.

/etc/ssh/ssh_host_ecdsa_key.pub – Открытый ключ ECDSA, используемый демоном **sshd**.

/etc/ssh/ssh_host_key – Частный ключ RSA, используемый демоном **sshd** для протокола SSH версии 1.

/etc/ssh/ssh_host_key.pub – Открытый ключ RSA, используемый демоном **sshd** для протокола SSH версии 1.

/etc/ssh/ssh_host_rsa_key - Частный ключ RSA, используемый демоном **sshd** для протокола SSH версии 2.

/etc/ssh/ssh_host_rsa_key.pub - Открытый ключ RSA, используемый демоном **sshd** для протокола SSH версии 2.

/etc/pam.d/sshd – Конфигурационный файл PAM, для демона **sshd**.

/etc/sysconfig/sshd – Конфигурационный файл для службы **sshd**.

Пользовательская информация о конфигурации SSH хранится в каталоге **~/.ssh/** домашнего каталога пользователя, как описано в ниже:

~/.ssh/authorized_keys – Удерживает список авторизованных публичных ключей для серверов. Когда клиент подключается к серверу, сервер аутентифицирует клиент, за счет проверки публичного ключа, хранящегося в данном файле.

~/.ssh/id_ecdsa – Содержит частный ключ ECDSA для пользователя.

~/.ssh/id_ecdsa.pub – Содержит открытый ключ ECDSA для пользователя.

~/.ssh/id_rsa – Содержит частный ключ RSA, используемый **ssh** для протокола SSH версии 2.

~/.ssh/id_rsa.pub – Содержит открытый ключ RSA, используемый **ssh** для протокола SSH версии 2.

~/.ssh/identity – Содержит частный ключ RSA, используемый **ssh** для протокола SSH версии 1.

~/.ssh/identity.pub - Содержит открытый ключ RSA, используемый **ssh** для протокола SSH версии 1.

~/.ssh/known_hosts – Содержит ключи хостов от серверов SSH к которым подключался пользователь. Этот файл очень важен для подтверждения того, что клиент SSH подключается к корректному серверу SSH.

Для получения информации по различным директивам, которые могут быть использованы в конфигурационных файлах SSH, обратитесь к страницам руководства: **ssh_config(5)** и **sshd_config(5)**.

Запуск сервера OpenSSH.

Перед запуском сервера OpenSSH вы должны установить пакет **openssh-server**.



ПРИМЕЧАНИЕ:
*Во всех вариантах установки CentOS 7, пакет **openssh-server** устанавливается по умолчанию, но пакет **openssh-clients** устанавливается не всегда.*

Для запуска демона **sshd** в текущей сессии, введите следующую команду от имени пользователя root:

```
# systemctl start sshd.service
```

Для остановки демона **sshd** в текущей сессии, введите следующую команду от имени пользователя root:

```
# systemctl stop sshd.service
```

Если вы хотите настроить демон **sshd** на автоматический запуск в процессе загрузки системы, выполните следующую команду от имени пользователя root:

```
# systemctl enable sshd.service
```

Демон **sshd** зависит от таргета **network.target**, которого достаточно для статически настроенных сетевых интерфейсов и опции по умолчанию **ListenAddress 0.0.0.0**. Для указание другого адреса и использования динамической конфигурации сети, добавьте в зависимости юниту **sshd.service** таргет **network-online.target**. Для этого необходимо в файле **/etc/systemd.system/sshd.service.d/local.conf** установить опции Wants и After следующим образом:

```
[Unit]
Wants=network-online.target
After=network-online.target
```

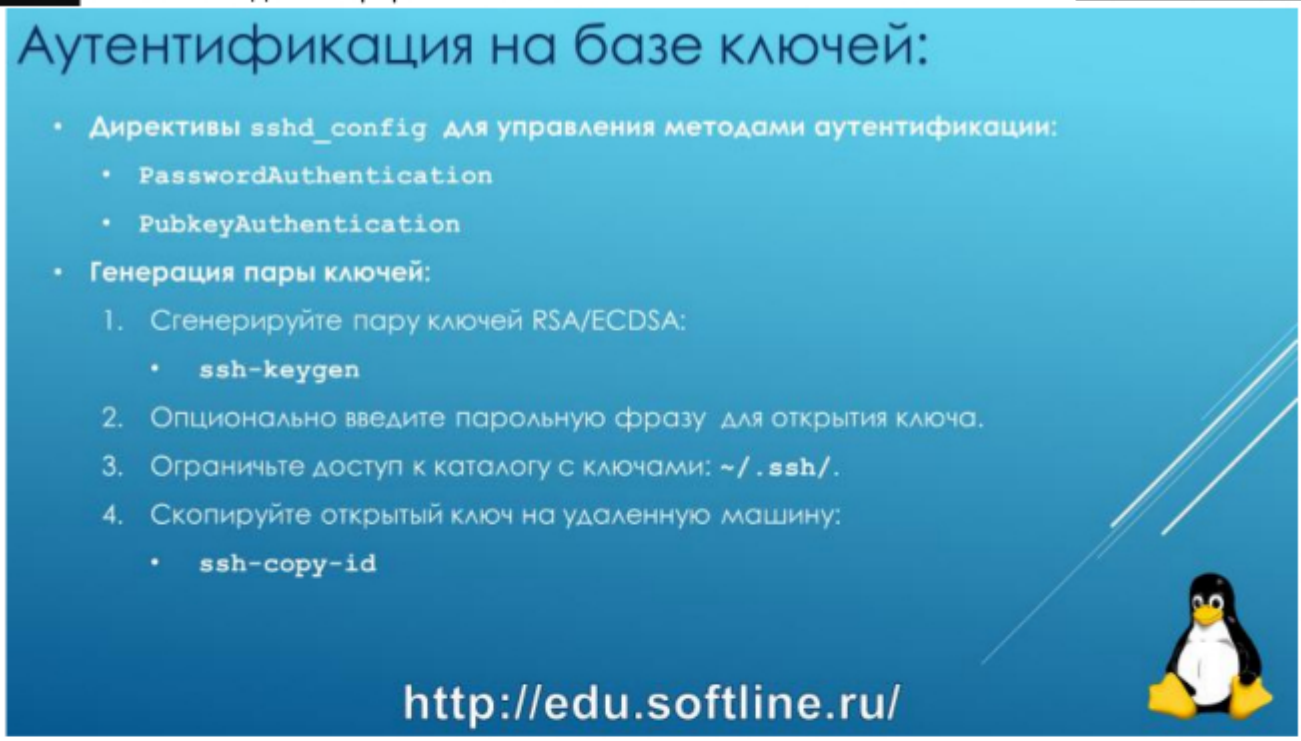
Обратите внимание что, если вы переустановите систему, будет создан новый набор ключей идентификации. В результате клиент, который подключался к системе ранее при помощи любой утилиты OpenSSH, получит следующее сообщение:

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@ WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED! @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that the RSA host key has just been changed.
```

Чтобы этого не произошло, необходимо сделать резервную копию файлов из каталога **/etc/ssh/** и в последствии восстановить их после переустановки системы.

Требования SSH для удаленных подключений.

Чтобы SSH был по настоящему эффективным, использование небезопасных подключений должно быть ограничено. В противном случае пароль, защищаемый при помощи SSH, может быть перехвачен во время сессии Telnet. Рекомендуется отключить следующие службы: **telnet**, **rsh**, **rlogin** и **vsftpd**.



Для увеличения безопасности в будущем, сгенерируйте пару ключей SSH и примените аутентификацию на базе ключей, отключив аутентификацию на базе паролей. Чтобы это сделать, откройте конфигурационный файл **/etc/ssh/sshd_config** в текстовом редакторе и измените опцию **PasswordAuthentication** следующим образом:

```
PasswordAuthentication no
```

Если вы работаете на системе, которая не была только что установлена, проверьте, что не установлена опция **PubkeyAuthentication no**. Если вы подключаетесь удаленно, перед отключением аутентификации по паролю, протестируйте возможность использования аутентификации на базе ключей.

Чтобы иметь возможность использовать **ssh**, **scp** или **sftp** для подключения к серверу с клиентской машины, сгенерируйте пару ключей авторизации, как описано в процедуре ниже. Обратите внимание, что ключи должны быть сгенерированы для каждого пользователя отдельно.

CentOS 7 (RHEL 7) использует протокол SSH версии 2 по умолчанию.



ПРИМЕЧАНИЕ:

Если вы выполните процедуру от имени пользователя root, только root, только root будет иметь возможность использовать ключи.



ПРИМЕЧАНИЕ:

*Если вы переустанавливаете вашу систему и хотите сохранить сгенерированную ранее пару ключей, сделайте резервную копию каталога **~/.ssh/**. После переустановки, скопируйте каталог обратно в домашнюю папку. Это процедура может быть выполнена для всех пользователей системы, включая пользователя root.*

Генерация пары ключей.

Для генерации пары ключей RSA для протокола SSH версии 2, воспользуйтесь следующими шагами:

1. Сгенерируйте пару ключей RSA, при помощи следующей команды:

```
$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/student/.ssh/id_rsa):
```

2. Нажмите **<Enter>** для подтверждения расположения по умолчанию (**~/.ssh/id_rsa**), для вновь созданного ключа.
3. Введите парольную фразу и подтвердите ее, когда она будет запрошена вновь. Для обеспечения безопасности не используйте пароль от учетной записи.

   После этого вам будет выдано следующее сообщение:

```
Your identification has been saved in /home/Student/.ssh/id_rsa.
Your public key has been saved in /home/Student/.ssh/id_rsa.pub.
The key fingerprint is:
e7:97:c7:e2:0e:f9:0e:fc:c4:d7:cb:e5:31:11:92:14
student@c7-client01.ll-100.local
The key's randomart image is:
+--[ RSA 2048]----+
|              E.  |
|             . .  |
|              o . |
|               . .|
|        S .     . |
|         + o o ..|
|          * * +oo|
|           O +..=|
|           o*  o.|
+-----------------+
```

4. По умолчанию права на каталог **~/.ssh/** установлены в **rwx------** или в восьмеричной системе счисления **700**. Это делается для того, чтобы только пользователь **student** мог просматривать содержимое. Чтобы в этом убедиться выполните следующую команду:

```
$ ls -ld ~/.ssh
drwx------. 2 student student 54 Nov 25 16:56 /home/Student/.ssh/
```

5. Чтобы скопировать открытый ключ на удаленную машину, выполните следующую команду:

   ssh-copy-id *user@hostname*

   Эта команда копирует последний измененный открытый ключ **~/.ssh/id*.pub**, если он еще не установлен. В качестве альтернативы можно указать имя файла следующим образом:

   ssh-copy-id -i ~/.ssh/id_rsa.pub *user@hostname*

   Эта команда скопирует содержимое файла **~/.ssh/id_rsa.pub** в файл **~/.ssh/authorized_keys** на машине к которой вы хотите подключиться. Если файл уже существует, ключ дописывается в конец.

Для генерации пары ключей ECDSA для протокола SSH версии 2, воспользуйтесь следующими шагами:

1. Сгенерируйте пару ключей ECDSA при помощи следующей команды:

```
$ ssh-keygen -t ecdsa
Generating public/private ecdsa key pair.
Enter file in which to save the key (/home/student/.ssh/id_ecdsa):
```

2. Нажмите **<Enter>** для подтверждения расположения по умолчанию (**~/.ssh/id_ecdsa**), для вновь созданного ключа.
3. Введите парольную фразу и подтвердите ее, когда она будет запрошена вновь. Для обеспечения безопасности не используйте пароль от учетной записи.

   После этого вам будет выдано следующее сообщение:

```
Your identification has been saved in /home/Student/.ssh/id_ecdsa.
Your public key has been saved in /home/student/.ssh/id_ecdsa.pub.
The key fingerprint is:
fd:1d:ca:10:52:96:21:43:7e:bd:4c:fc:5b:35:6b:63
Student@c7-client01.ll-100.local
The key's randomart image is:
+--[ECDSA 256]---+
|        .+ +o    |
|        . =.o    |
|         o o + ..|
|          + + o +|
|        S o o oE.|
|           + oo+.|
|            + o  |
|                 |
|                 |
+-----------------+
```

6. По умолчанию права на каталог **~/.ssh/** установлены в **rwx------** или в восьмеричной системе счисления **700**. Это делается для того, чтобы только пользователь **student** мог просматривать содержимое. Чтобы в этом убедиться выполните следующую команду:

```
$ ls -ld ~/.ssh
drwx------. 2 Student Student 54 Nov 25 16:56 /home/student/.ssh/
```

7. Чтобы скопировать открытый ключ на удаленную машину, выполните следующую команду:

   ssh-copy-id *user@hostname*

   Эта команда копирует последний измененный открытый ключ **~/.ssh/id*.pub**, если он еще не установлен. В качестве альтернативы можно указать имя файла следующим образом:

   ssh-copy-id -i ~/.ssh/id_ecdsa.pub *user@hostname*

   Эта команда скопирует содержимое файла **~/.ssh/id_ecdsa.pub** в файл **~/.ssh/authorized_keys** на машине к которой вы хотите подключиться. Если файл уже существует, ключ дописывается в конец.



ПРИМЕЧАНИЕ:

Частный ключ предназначен только для персонального использования пользователем, который его сгенерировал, поэтому очень важно никогда не передавать его кому-либо.

Настройка ssh-agent.

Для того чтобы сохранить парольную фразу и не вводить ее каждый раз при инициализации подключения с удаленной машиной, вы можете использовать аутентификацию агента **ssh-agent**. В рамках графической сессии в окружении рабочего стола GNOME это можно сделать при помощи инструмента **Stareup Applications Preferences** (потребуется установленный пакет **openssh-askpath**).

Для сохранения парольной фразы для оболочки командной строки, используйте следующую команду:

```
$ ssh-add
Enter passphrase for /home/USER/.ssh/id_rsa:
```

После выхода из системы парольная фраза будет забыта. Команду необходимо вводить каждый раз при подключении к виртуальной консоли или окну терминала.



Для подключения к серверу OpenSSH с клиентской машины, необходимо установить пакет **openssh-clients**.

Использование утилиты ssh.

Утилита **ssh** позволяет входить на удаленную машину и выполнять на ней команды. Это безопасная замена для таких программ как **rlogin**, **rsh** и **telnet**.

Подобно команде telnet, для входа на удаленную машину используется команда следующего вида:

```
$ ssh hostname
```

Например, для входа на удаленную машину **c7-server01.ll-100.local**, введите следующую команду:

```
$ ssh c7-server.ll-100.local
```

При таком подключении вы используете для входа то же имя пользователя, под которым работаете на локально системе. Если необходимо указать другое имя, используется команда следующего вида:

```
$ ssh username@hostname
```

Например, для входа на удаленную машину **c7-server01.ll-100.local** от имени пользователя **student** введите следующую команду:

```
$ ssh student@c7-server.ll-100.local
```

При первой инициализации подключения, выдается сообщение, подобное следующему:

```
The authenticity of host 'penguin.example.com' can't be established.
ECDSA key fingerprint is 256
da:24:43:0b:2e:c1:3f:a1:84:13:92:01:52:b4:84:ff.
Are you sure you want to continue connecting (yes/no)?
```

Пользователи должны проверить корректность цифрового отпечатка перед ответом на запрос. Пользователь может запросить у администратора сервера подтверждение корректности ключа. Если пользователь может получить прямой доступ к хосту, то для получения цифрового отпечатка, можно воспользоваться командой **ssh-keygen** следующим образом:

```
# ssh-keygen -l -f /etc/ssh/ssh_host_ecdsa_key.pub
256 da:24:43:0b:2e:c1:3f:a1:84:13:92:01:52:b4:84:ff (ECDSA)
```

Введите yes для принятия ключа и подтверждения подключения. Вы увидите уведомление, что сервер был добавлен в список известных хостов и запрос на ввод пароля:

```
Warning: Permanently added 'penguin.example.com' (ECDSA) to the list of
known hosts.
USER@ penguin.example.com's password:
```



ПРИМЕЧАНИЕ:

*Если ключ хоста сервера SSH изменился, клиент уведомляет пользователя, что подключение не может быть установлено до тех пор пока старый ключ хоста не будет удален из файла **~/.ssh/known_hosts**. Прежде чем сделать это, уточните у администратора сервера, не произошла ли подмена.*
*Для удаления ключа из файла **~/.ssh/known_hosts**, воспользуйтесь следующей командой:*

```
$ ssh-keygen -R c7-server01.ll-100.local
# Host c7-server01.ll-100.local found: line 15 type ECDSA
/home/student/.ssh/known_hosts updated.
Original contents retained as /home/USER/.ssh/known_hosts.old
```

После ввода пароля, будет предоставленастрока оболочки удаленной машины.

В качестве альтернативы, программа ssh может выполнить команду на удаленной машине без входа в строку оболочки:

```
$ ssh [username@]hostname command
```

Например, файл **/etc/redhat-release** предоставляет информацию о версии CentOS. Для просмотра содержимого этого файла на удаленной машине **c7-server01.ll-100.local**, воспользуйтесь следующей командой:

```
$ ssh student@c7-server01.ll-100.local cat /etc/redhat-release
student@c7-server01.ll-100.local's password:
CentOS 7.2
```

После ввода корректного пароля и вывода результатов выполнения команды на удаленной машине вы вернетесь в локальную строку оболочки.

Использование утилиты scp.

Утилита **scp** может быть использована для передачи файлов между машинами по безопасному, зашифрованному соединению. В общем виде она очень похожа на **rcp**.

Для передачи файла на удаленную систему, используйте команду в следующей форме:

```
$scp localfile username@hostname:remotefile
```

Например, если вы хотите передать **taglist.vom** на удаленную машину с именем **c7-server01.ll-100.local**, воспользуйтесь следующей командой:

```
[root@c7-client01 ~]# scp c7-server01:/root/file.big /root/
The authenticity of host 'c7-server01 (192.168.0.21)' can't be established.
ECDSA key fingerprint is ab:9a:7f:eb:25:3b:3d:f2:c8:63:b5:03:02:2f:0d:62.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'c7-server01' (ECDSA) to the list of known hosts.
root@c7-server01's password:
file.big                                      100%   43MB  43.1MB/s   00:01
```

Несколько файлов может быть указано за раз.

Для перемещения удаленного файла на локальную систему, используйте следующий синтаксис:

scp *username*@*hostname*:*remotefile localfile*

Использование утилиты sftp.

Утилита **sftp** может быть использована для открытия безопасной, интерактивной сессии FTP. По своей структуре, она похожа на **ftp** за исключением того, что использует безопасное, зашифрованное подключение.

Для подключения к удаленной системе используйте команду в следующей форме:

$ sftp *username*@*hostname*

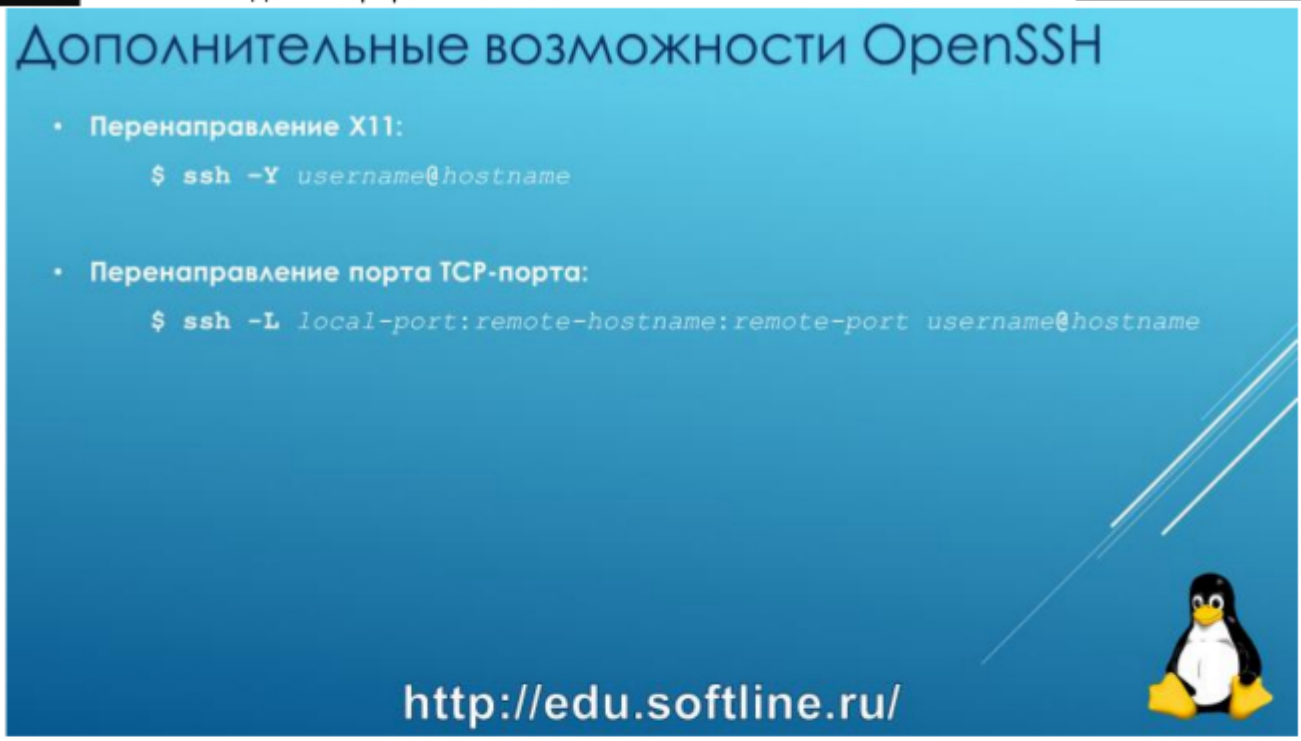
Например, для входа на удаленную машину (**c7-server01.ll-100.local**) от имени учетной записи **student**, введите:

```
[root@c7-client01 ~]# sftp root@c7-server01
root@c7-server01's password:
Connected to c7-server01.
sftp>
```

После ввода корректного пароля, будет представлена командная строка. Утилита **sftp** принимает набор команд, похожий на используемый **ftp**, некоторые из них представлены в таблице ниже:

Команда	Описание
ls *[каталог]*	Вывод содержимого удаленного *каталога*, если каталог не указан, то по умолчанию используется текущий каталог.
cd *каталог*	Сменить удаленный рабочий каталог на *каталог*.
mkdir *каталог*	Создать удаленный каталог (*каталог*).
rmdir *путь*	Удалить удаленный каталог (*каталог*).
put *localfile [remotefile]*	Переместить локальный файл на удаленную машину.
get *[remotefile] localfile*	Переместить удаленный файл на локальную машину.

Для получения полного списка доступных команд, обратитесь к странице руководства **sftp(1)**.



Безопасный интерфейс командной строки – это только начало для множества возможностей использования SSH. При наличии достаточного объема пропускной способности, сессии X11 могут быть перенаправлены через канал SSH. Или, при помощи перенаправления TCP/IP, изначально небезопасные соединения между системами могут быть привязаны к указанным каналам SSH.

Перенаправление X11.

Для открытия сессии X11 через SSH подключение, воспользуйтесь командой в следующей форме:

```
$ ssh -Y username@hostname
```

Например, для входа на удаленную машину (**c7-client02.ll-100.local**) от имени пользователя **student**, введите:

```
$ ssh -Y student@c7-client02.ll-100.local
```

Когда программа X запускается из строки безопасной оболочки (SSH), клиент SSH и сервер создают новый безопасный канал и данные программы X прозрачно пересылаются через этот канал.



ПРИМЕЧАНИЕ:

Обратите внимание, что X Windows System должна быть установлена на удаленной системе до перенаправления X11. Введите следующую команду от имени пользователя root для установки группы пакетов X11:

```
# yum group install "X Window System"
```

Перенаправление X11 может быть очень удобным. Например, перенаправление X11 может быть использовано для создания безопасной интерактивной сессии с утилитой Print Settings. Чтобы сделать это, подключитесь к серверу при помощи **ssh** и введите:

```
$ system-config-printer &
```

Утилита Print Settings отобразится, позволив безопасно настроить печать на удаленной системе.

Перенаправление порта.

SSH может обеспечить безопасность прочих небезопасных протоколов TCP/IP через перенаправление портов. При использовании этой техники, сервер SSH выступает в роли зашифрованного тоннеля для клиента SSH.

Перенаправление порта работает за счет привязки локального порта клиента к удаленному порту на сервере. SSH может привязывать любой порт с сервера к любому порту на клиенте. Номера портов не обязаны соответствовать, чтобы эта техника работала.



ПРИМЕЧАНИЕ:
Настройка перенаправления портов на прослушивание портов меньше 1024, требует уровня доступа root.

Чтобы создать канал перенаправления порта TCP/IP на локальной системе (**localhost**), используйте команду в следующей форме:

```
$ ssh -L local-port:remote-hostname:remote-port username@hostname
```

Например, для проверки поты на сервере **c7-server01.ll-100.local**, при помощи POP3 через зашифрованное подключение, используйте следующую команду:

```
$ ssh -L 1100:c7-server01.ll-100.local:110 c7-server01.ll-100.local
```

После установки канала перенаправления порта между клиентской машиной и почтовым сервером, направьте почтового клиента POP3 на использование порта **1100** на **localhost** для проверки новых почтовых сообщений. Любые запросы, отправленные на порт **1100** на клиентской системе, будут безопасно направлены на сервер **c7-server01.ll-100.local**.

Если на сервере **c7-server01.ll-100.local** не запущен сервер SSH, но на других машинах в той же сети запущен, SSH может быть использован для обеспечения безопасности части подключения. Однако, потребуется частично изменить команду:

```
$ ssh -L 1100:c7-server01.ll-100.local:110 c7-server02.ll-100.local
```

В примере, POP3 запрашиваемый на порту **1100** на клиентской машине, перенаправляется на порт **22** на сервере SSH (**c7-server02.ll-100.local**). Затем, **c7-server02.lll-00.local** соединяется с портом **110** на **c7-server01.ll-100.local** для проверки нового почтового сообщения. Обратите внимание что при таком подключении безопасное соединение существует только между клиентской системой и **c7-server02.lll-00.local**.

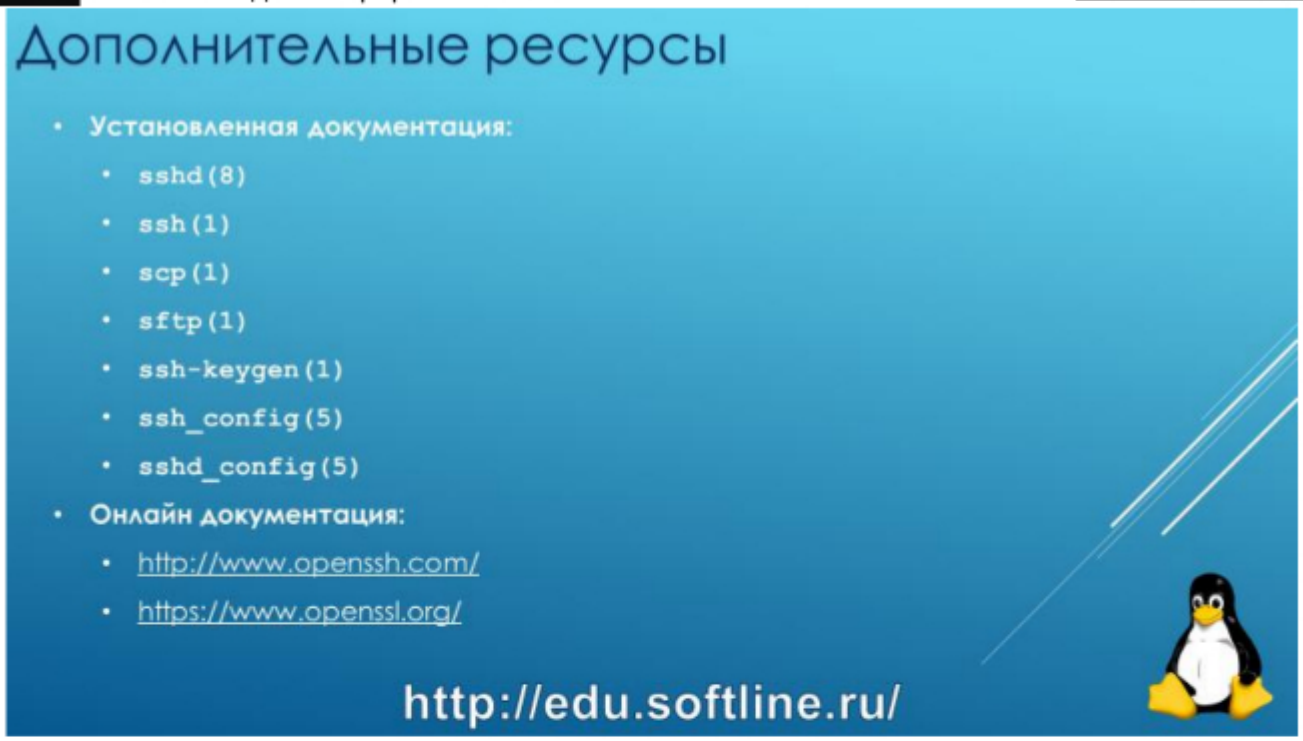
Перенаправление порта может также быть использовано для безопасного получения информации через сетевые экраны. Если сетевой экран настроен для пропуска трафика SSH через стандартный порт (**22**), но блокирует доступы к прочим портам, подключение между двумя хостами возможно за счет перенаправления коммуникаций через установленное SSH подключение.



ПРИМЕЧАНИЕ:

Использование перенаправления порта для перенаправления подключения в конечном счете позволяет любому пользователю на клиентской системе подключаться к службе. Если клиентская система будет компрометирована, злоумышленник получит доступ к перенаправленным службам.

Системные администраторы знакомые с перенаправлением портов могут отключить эту возможность на сервере, указав **`No`** *для параметра* **`AllowTcpForwarding`** *в конфигурационном файле* **`/etc/ssh/sshd_config`** *и перезапустив службу* **`sshd`**.



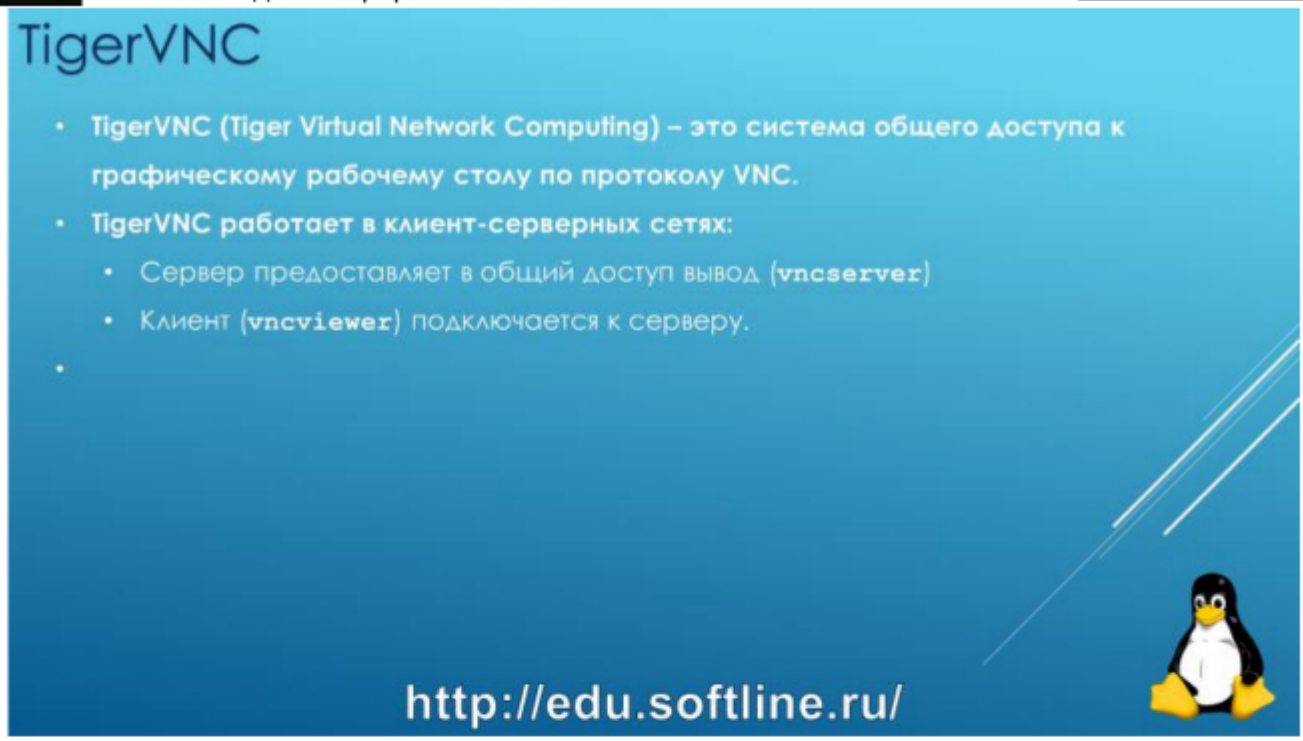
Для получения дополнительной информации по настройке или подключению к серверу OpenSSH на системе CentOS (RHEL), обратитесь к ресурсам, перечисленным ниже.

Установленная документация.

- **`sshd(8)`** – Страница руководства демона **sshd**, содержит доступные опции командной строки и предоставляет полный список поддерживаемых конфигурационных файлов и каталогов.
- **`ssh(1)`** – Страница руководства клиентского приложения **ssh**, предоставляет полный список доступных опций командной строки, поддерживаемых конфигурационных файлов и каталогов.
- **`scp(1)`** – Страница руководства утилиты **scp**, предоставляет более детальное описание утилиты и ее использования.
- **`sftp(1)`** – Страница руководства утилиты **sftp**.
- **`ssh-keygen(1)`** – Страница руководства утилиты **ssh-keygen**, детально описывает ее использование для генерации, управления и конвертации ключей аутентификации, используемых **ssh**.
- **`ssh_config(5)`** – Страница руководства **ssh_config**, описывает доступные опции настройки клиента SSH.
- **`sshd_config(5)`** – Страница руководства **sshd_config**, предоставляет полное описание доступных опций настройки демона SSH.

Онлайн документация.

- http://www.openssh.com/ – Домашняя страница OpenSSH, содержит подробную документацию, часто задаваемые вопросы, ссылки на списки рассылок, отчеты об ошибках и прочие полезные ресурсы.
- https://www.openssl.org/ – Домашняя страница OpenSSL, содержит подробную документацию, часто задаваемые вопросы, ссылки на списки рассылок, отчеты об ошибках и прочие полезные ресурсы.



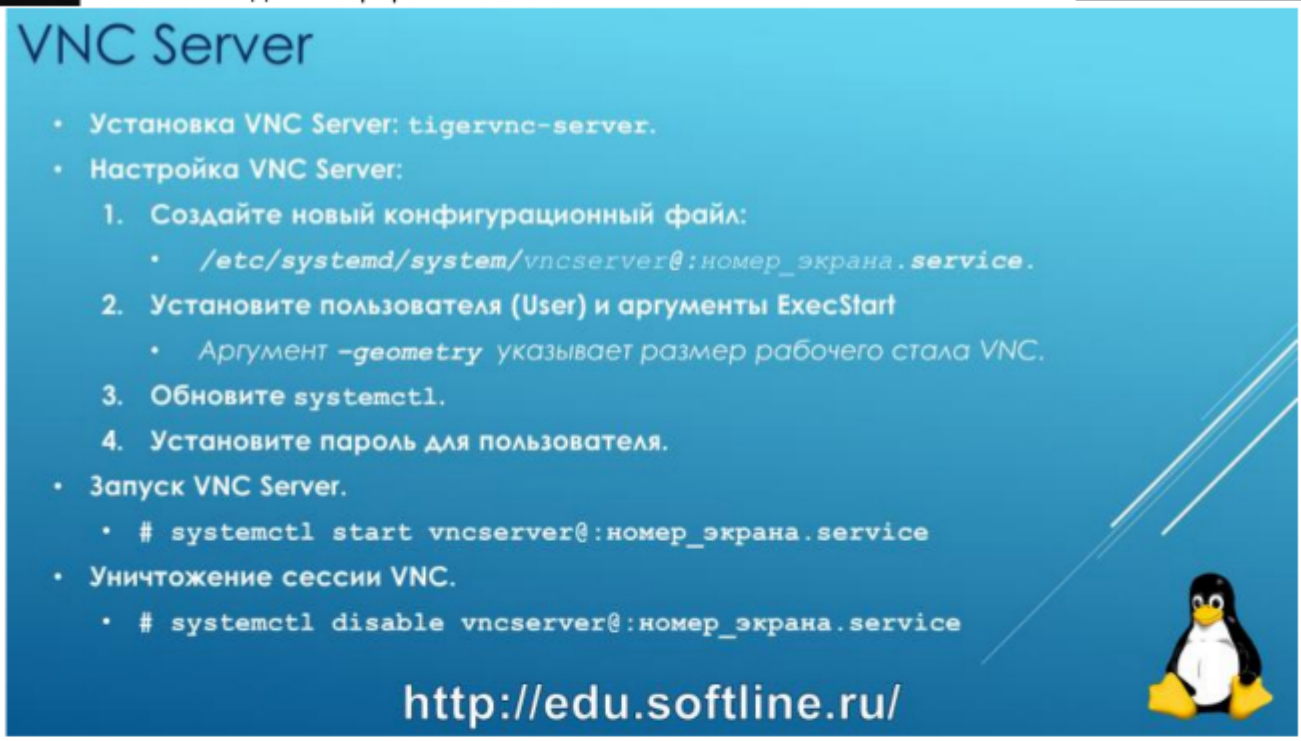
TigerVNC (Tiger Virtual Network Computing) – это система общего доступа к графическому рабочему столу, которая позволяет удаленно управлять другими компьютерами.

TigerVNC работает в клиент-серверных сетях: сервер предоставляет в общий доступ его вывод (**`vncserver`**), а клиент (**`vncviewer`**) подключается к серверу.



ПРИМЕЧАНИЕ:

В отличии от предыдущих дистрибутивов CentOS (RHEL), текущий TigerVNC использует для настройки демон управления системой ***`systemd`****. Конфигурационный файл* ***`/etc/sysconfig/vncserver`*** *был заменен на* ***`/lib/system/system/vncserver@.service`****.*



vncserver – это утилита, которая запускает рабочий стол VNC (Virtual Network Computing). Она запускает Xvnc с необходимыми опциями и оконный менеджер на рабочем столе VNC. **vncserver** позволяет пользователям запускать полностью параллельные сессии на машине, которая доступна любому числу клиентов откуда угодно.

Установка VNC Server.

Для установки сервера TigerVNC, запустите следующую команду от имени пользователя root:

```
# yum install tigervnc-server
```

Если вы планируете использовать TigerVNC вместе с клиентом, запустите следующую команду, которая установит серверные компоненты вместе с клиентским:

```
# yum install vnc
```

Настройка VNC Server.

Процедура начальной настройки VNC подключений:

1. Создайте новый конфигурационный файл **/lib/systemd/system/vncserver@:***номер_экрана***.service** для каждого номера экрана, который вы хотите включить. В следующем примере настраивается экран номер 3, который включен в имя конфигурационного файла. Не обязательно создавать новый файл, можно копировать контент файла **/lib/systemd/system/vncserver@.service**:

   ```
   # cp /lib/systemd/system/vncserver@.service
   /lib/systemd/system/vncserver@:3.service
   ```

2. Отредактируйте файл **/lib/systemd/system/vncserver@:***номер_экрана***.service**, установив пользователя (**User**) и аргументы **ExecStart**, как в примере ниже. Аргумент –geometry указывает размер рабочего стала VNC. По умолчанию используется **1024x768**. Пример указания аргументов в конфигурационном файле:

   ```
   User=joe
   ```

```
ExecStart=/sbin/runuser -l joe -c "/usr/bin/vncserver %i -geometry 1280x1024"
```

3. Сохраните изменения.
4. Обновите systemctl, чтобы убедиться, что изменения вступили в силу незамедлительно:

```
# systemctl daemon-reload
```

5. Установите пароль для пользователя или пользователей, определенных в конфигурационных файлах.

```
# vncpasswd user
Password:
Verify:
```

Повторите процедуру, чтобы установить пароль для остальных пользователей:

```
# su – имя_пользователя
```



ПРИМЕЧАНИЕ:

Сохраненный пароль не шифруется безопасно. Любой, кто получит доступ к файлу паролей, может найти в нем пароль в открытом виде.

Запуск VNC Server.

Для запуска службы с указанным номером экрана, выполните следующую команду:

```
# systemctl start vncserver@:номер_экрана.service
```

Для настройки службы на автоматический запуск во время запуска системы, используйте следующую команду:

```
# systemctl enable vncserver@:номер_экрана.service
```

В данный момент остальные пользователи могут использовать программу **vncviewer** для подключения к серверу при помощи номера экрана и определенного пароля.

Уничтожение сессии VNC.

Подобно запуску сервиса с указанием номера экрана, можно отключить автоматический запуск сервиса при загрузке системы:

```
# systemctl disable vncserver@:номер_экрана.service
```

Или остановить службу, если операционная система запущена, при помощи следующей команды:

```
# systemctl stop vncserver@:номер_экрана.service
```

Межсетевой экран.

Когда используется незашифрованное подключение, firewalld может заблокировать его. Для нормальной работы необходимо открыть указанный порт.

ПРИМЕЧАНИЕ:

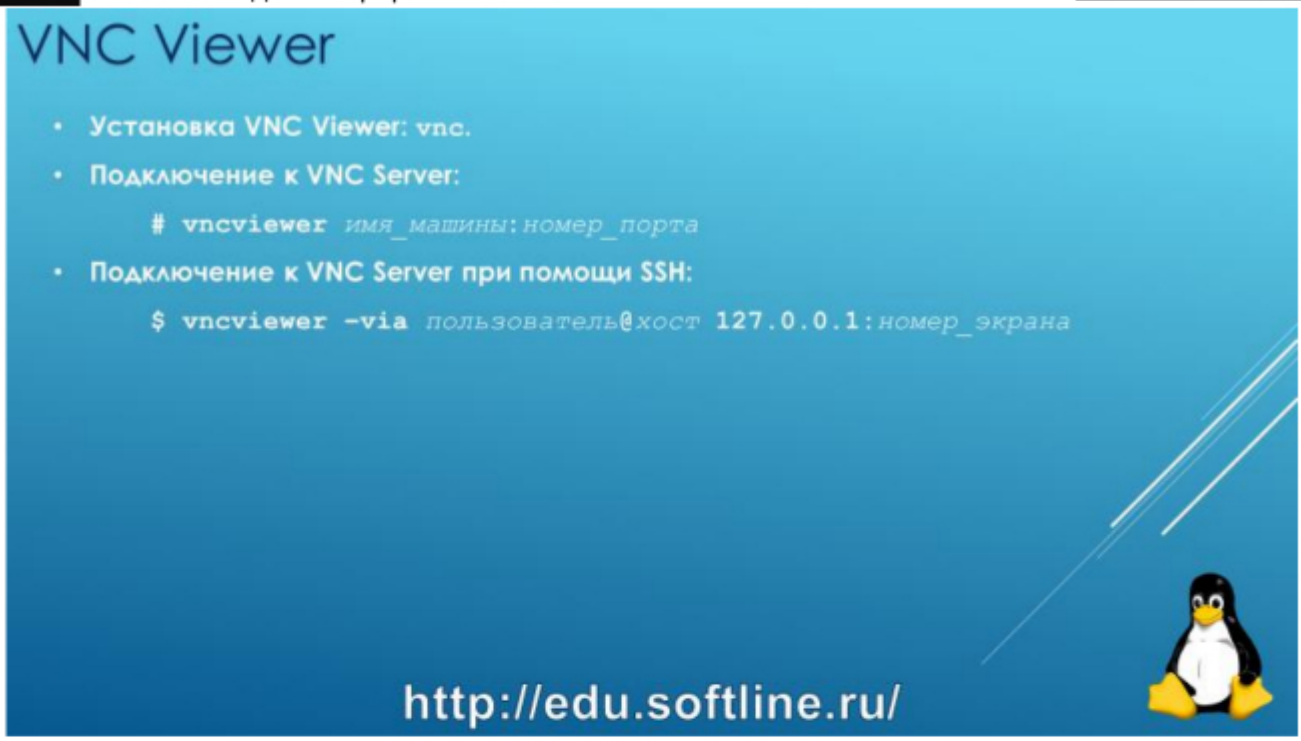
*По умолчанию сервер VNC использует порт **5900**. Чтобы вычислить порт, через который функционирует удаленный рабочий стол, необходимо сложить порт по умолчанию и назначенный номер экрана. Например, второй экран:**2** + **5900** = **5902**.*

1. Для получения информации о параметрах **firewalld**, выполните следующую команду:

   ```
   # firewall-cmd --list-all
   ```

2. Чтобы убедиться что выбранный порт открыт используйте команду:

   ```
   # firewall-cmd --query-port=номер_порта/tcp
   ```



vncviewer – это программа, которая отображает предоставленный в общий доступ графический пользовательский интерфейс и управляет сервером.

Для работы в **vncviewer** присутствует всплывающее меню, содержащее записи, которые выполняют различные действия, такие как переключение в полноэкранный режим и из него или выхода из режима просмотра. В качестве альтернативы можно использовать терминал для управления **vncviewer**. Для просмотра списка параметров **vncviewer** можно использовать команду **vncviewer -h**.

Установка VNC Viewer.

Для установки TigerVNC Viewer, выполните следующую команду от имени пользователя root:

```
# yum install vnc
```

Подключение к VNC Server.

Для подключения к серверу VNC, запустите команду **vncviewer** в следующем формате:

```
# vncviewer имя_машины:номер_порта
```

Например, для подключения к IP-адресу **192.168.0.4**, номеру экрана **3** от имени пользователя **joe**, используйте следующую команду:

```
# vncviewer joe 192.168.0.4 :3
```

Подключение к VNC Server при помощи SSH.

Протокол VNC – открытый сетевой протокол, который не имеет средств предотвращения сетевых атак на коммуникации. Для обеспечения безопасных коммуникаций, можно использовать туннель SSH для шифрования коммуникаций между клиентом и сервером VNC.

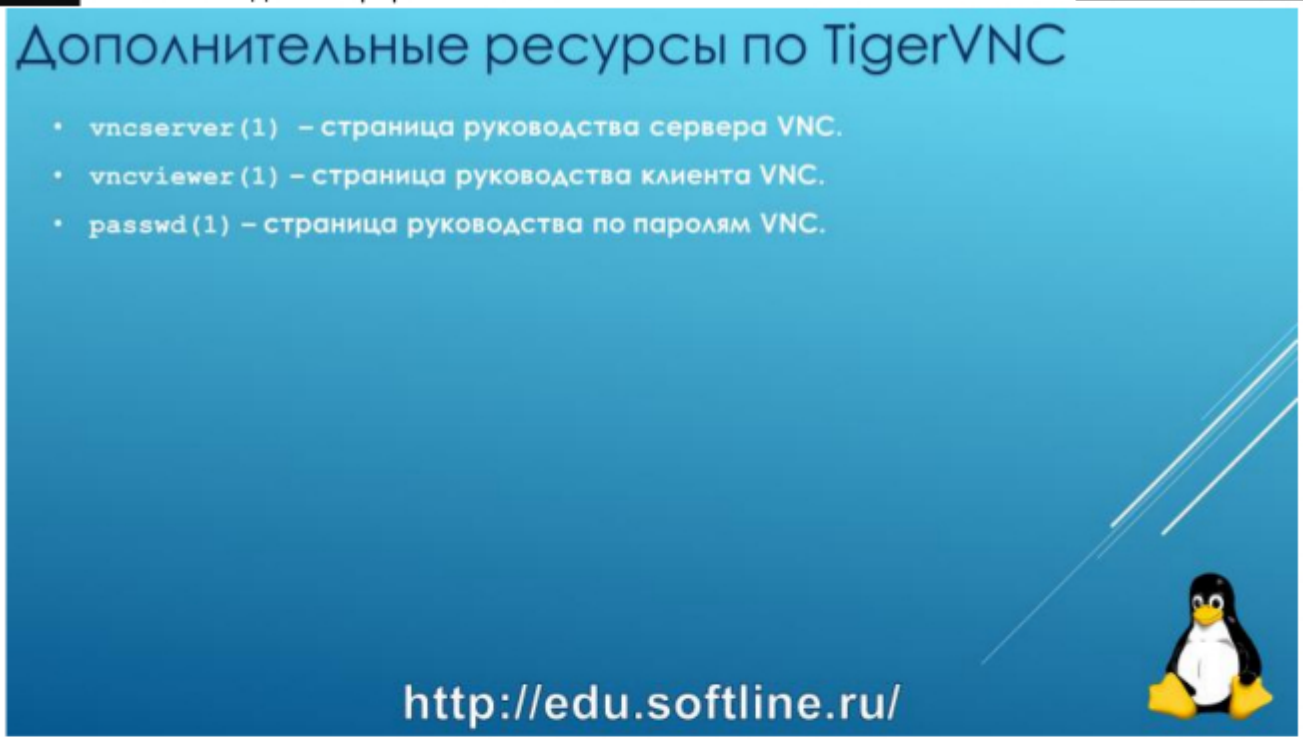
Для этого необходимо использовать опцию **-via**:

```
$ vncviewer -via пользователь@хост 127.0.0.1:номер_экрана
```

Пример использования опции **-via**:

```
$ vncviewer -via joe@192.168.2.101 127.0.0.1:3
```

Для получения информации по OpenSSH, обратитесь к соответствующему разделу.



Установленная документация:

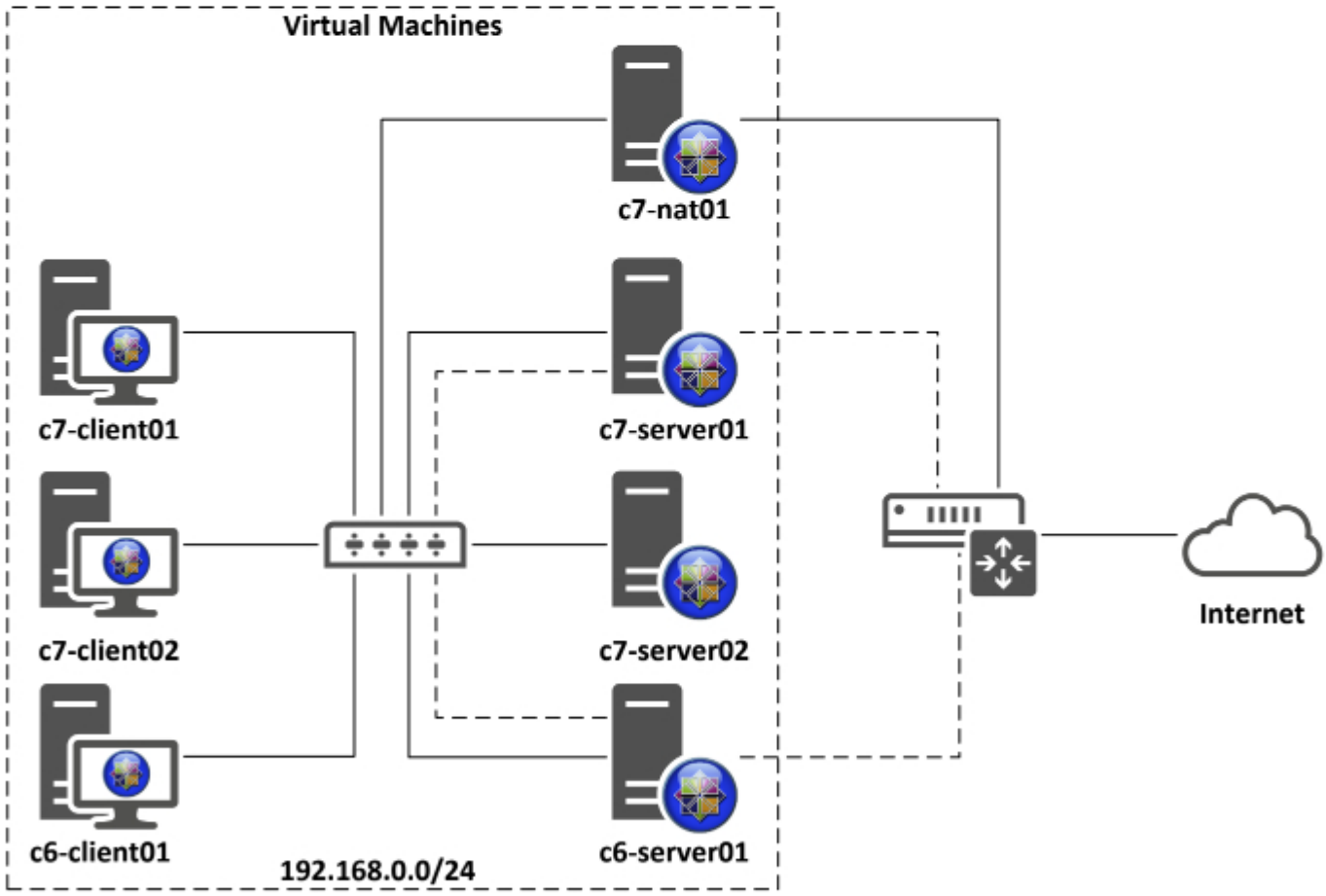
- **`vncserver(1)`** – страница руководства сервера VNC.
- **`vncviewer(1)`** – страница руководства клиента VNC.
- **`passwd(1)`** – страница руководства по паролям VNC.

Лабораторная работа

Лабораторная работа 7

- **Упражнения:**
  - Управление Journal.
  - Настройка сервера Rsyslog на CentOS 7.
  - Настройка аутентификации OpenSSH на базе ключей.
- **Необходимые машины:**
  - `c7-nat01`.
- **Учетные данные:**
  - Пользователь: `root`.
  - Пароль: **`Pa$$w0rd`**.

http://edu.softline.ru/





Упражнение 1: Управление Journal.

Задание 1: Просмотр сообщений в Journal.

1. Выведите 10 последних записей Journal.
2. Выведите всю доступную информацию по последним 5 событиям в Journal.
3. Выведите 5 последних записей Journal формате JSON.
4. Выведите 10 последних сообщений Journal с приоритетом не ниже ошибки (Error).
5. Выведите все сообщения Journal за последние 5 минут:
6. Выведите все сообщения Journal от устройства ядра **`hid-generic`**.
7. Выведите 10 последних сообщений Journal от пользователя root.

Задание 2: Настройка постоянного хранения сообщений Journal.

1. Создайте каталог для хранения файлов Journal.
2. Перезапустите сервис Journal.
3. Убедитесь, что сервис Journal запущен.
4. Убедитесь, что в каталоге для хранения файлов Journal появились файлы.



Ответы к упражнению 1: Управление Journal.

Задание 1: Просмотр сообщений в Journal.

1. Выведите 10 последних записей Journal.

```
# journalctl -n 10
```

2. Выведите всю доступную информацию по последним 5 событиям в Journal.

```
# journalctl -n 5 -o verbose
```

3. Выведите 5 последних записей Journal формате JSON.

```
# journalctl -n 5 -o json
```

4. Выведите 10 последних сообщений Journal с приоритетом не ниже ошибки (Error).

```
# journalctl -n 10 -p 3
```

5. Выведите все сообщения Journal за последние 5 минут:

```
# journalctl --since="2018-8-20 08:50:00"
```

6. Выведите все сообщения Journal от устройства ядра **hid-generic**.

```
# journalctl _KERNEL_DEVICE=+hid:0006:045E:0621.0001
```

7. Выведите 10 последних сообщений Journal от пользователя root.

```
# journalctl _UID=0 -n 10
```

Задание 2: Настройка постоянного хранения сообщений Journal.

1. Создайте каталог для хранения файлов Journal.

```
# mkdir -p /var/log/journal/
```

2. Перезапустите сервис Journal.

```
# systemctl restart systemd-journald
```

3. Убедитесь, что сервис Journal запущен.

```
# systemctl status systemd-journald
```

4. Убедитесь, что в каталоге для хранения файлов Journal появились файлы.

```
# ls -R /var/log/journal/
```



Упражнение 2: Настройка сервера Rsyslog на CentOS 7.

Задача 1: Настройка сервера Rsyslog.

1. Установите Rsyslog (**`rsyslog`**) на машине **c7-server01**.

> *ПРИМЕЧАНИЕ:*
> *Rsyslog (**`rsyslog`**) уже установлен в CentOS 7 и поэтому может быть только обновлен.*

2. При помощи конфигурационного файла **`/etc/rsyslog.conf`** настройте прием сообщений syslog по протоколам TCP и UDP.
3. Перезапустите службу rsyslog.
4. Убедитесь, что служба запущена и настроена на автоматический запуск.
5. Разрешите входящий траффик на порту 514 для UDP и TCP в FirewallD.

```
# firewall-cmd --add-port=514/udp --permanent
# firewall-cmd --add-port=514/tcp --permanent
# firewall-cmd --reload
```

6. Запустите интерактивный вывод сообщений из файла журнала **`/var/log/messages`** при помощи команды **`tail`**.

Задача 2: Настройка клиента Rsyslog.

1. Установите Rsyslog (**`rsyslog`**) на машине **c7-server02**.

> *ПРИМЕЧАНИЕ:*
> *Rsyslog (**`rsyslog`**) уже установлен в CentOS 7 и поэтому может быть только обновлен.*

2. При помощи конфигурационного файла **`/etc/rsyslog.conf`** настройте правило отправки всех сообщений syslog на сервер **c7-server01**.
3. Перезапустите службу rsyslog.
4. Убудеитесь, что служба запущена и настроена на автоматический запуск.
5. Убедитесь, что сообщения syslog начали поступать в файл **`/var/log/messages`** на систему **c7-server01**.

> *ПРИМЕЧАНИЕ:*
> *Для отправки тестового сообщения в журнал можно воспользоваться командой **`logger`**:*
> ***`# logger -t test "Test Log Message"`***

Задача 3: Настройка фильтрации журналов Rsyslog.

1. При помощи конфигурационного файла **`/etc/rsyslog.conf`** настройте правило записи всех сообщений syslog от хоста **c7-server01** в файл **`/var/log/c7-server01`**, а сообщений syslog от хоста c7-server02 в файл **`/var/log/c7-server02`**.
2. Перезапустите службу **`rsyslog`**.
3. Убедитесь, что служба **`rsyslog`** запущена.
4. Убедитесь, что файлы журнала **`/var/log/c7-server01`** и **`/var/log/c7-server02`** создались и содержат сообщения syslog.



Ответы к упражнению 2: Настройка сервера Rsyslog на CentOS 7.

Задача 1: Настройка сервера Rsyslog.

1. Установите Rsyslog (**rsyslog**) на машине **c7-server01**.

```
# yum install rsyslog
```



ПРИМЕЧАНИЕ:
*Rsyslog (**rsyslog**) уже установлен в CentOS 7 и поэтому может быть только обновлен.*

2. При помощи конфигурационного файла **/etc/rsyslog.conf** настройте прием сообщений syslog по протоколам TCP и UDP.

```
# vi /etc/rsyslog.conf
# Provides UDP syslog reception
$ModLoad imudp
$UDPServerRun 514

# Provides TCP syslog reception
$ModLoad imtcp
$InputTCPServerRun 514
```

3. Перезапустите службу **rsyslog**.

```
# systemctl restart rsyslog.service
```

4. Убедитесь, что служба **rsyslog** запущена и настроена на автоматический запуск.

```
# systemctl status rsyslog.service
# systemctl is-enabled  rsyslog.service
```

5. Разрешите входящий траффик на порту 514 для UDP и TCP в FirewallD.

```
# firewall-cmd --add-port=514/udp --permanent
# firewall-cmd --add-port=514/tcp --permanent
# firewall-cmd --reload
```

6. Запустите интерактивный вывод сообщений из файла журнала **/var/log/messages** при помощи команды **tail**.

```
# tail -f /var/log/messages
```

Задача 2: Настройка клиента Rsyslog.

1. Установите Rsyslog (**rsyslog**) на машине **c7-server02**.

```
# yum install rsyslog
```

ПРИМЕЧАНИЕ:
*Rsyslog (**rsyslog**) уже установлен в CentOS 7 и поэтому может быть только обновлен.*

2. При помощи конфигурационного файла **/etc/rsyslog.conf** настройте правило отправки всех сообщений syslog на сервер **c7-server01**.

```
# vi /etc/rsyslog.conf
#### RULES ####

*.* @c7-server01:514
```

3. Перезапустите службу **rsyslog**.

```
# systemctl restart rsyslog.service
```

4. Убедитесь, что служба **rsyslog** запущена и настроена на автоматический запуск.

```
# systemctl status rsyslog.service
# systemctl is-enabled  rsyslog.service
```

5. Убедитесь, что сообщения syslog начали поступать в файл **/var/log/messages** на систему **c7-server01**.

```
# tail -f /var/log/messages
…
Sep 27 05:22:05 c7-server02 test: Test Log Message
```

ПРИМЕЧАНИЕ:
*Для отправки тестового сообщения в журнал можно воспользоваться командой **logger**:*
 # logger -t test "Test Log Message"

Задача 3: Настройка фильтрации журналов Rsyslog.

1. При помощи конфигурационного файла **/etc/rsyslog.conf** настройте правило записи всех сообщений syslog от хоста **c7-server01** в файл **/var/log/c7-server01**, а сообщений syslog от хоста c7-server02 в файл **/var/log/c7-server02**.

```
# vi /etc/rsyslog.conf
#### RULES ####

:hostname, isequal, "c7-server01" /var/log/c7-server01
:hostname, isequal, "c7-server02" /var/log/c7-server02
```

2. Перезапустите службу **rsyslog**.

```
# systemctl restart rsyslog.service
```

3. Убедитесь, что служба **rsyslog** запущена.

```
# systemctl status rsyslog.service
```

4. Убедитесь, что файлы журнала **/var/log/c7-server01** и **/var/log/c7-server02** создались и содержат сообщения syslog.

```
# ls /var/log/c7-server0?
# tail -n 1 /var/log/c7-server01
# tail -n 1 /var/log/c7-server02
```



Упражнение 3 – Настройка аутентификации OpenSSH на базе ключей.

1. Войдите на машину **c7-client01** от имени пользователя root.
2. Проверьте возможность подключения к **c7-server01** от имени пользователя root.
3. Сгенерируйте пару ключей RSA, сохраните идентификатор в файле **`/root/.ssh/id_rsa`** и защитите ключ при помощи пароля **`Pa$$w0rd`**.
4. Сделайте копию открытого ключа на машину **c7-server01** под пользователем root.
5. Дважды проверьте возможность подключения к **c7-server01**, при первом подключении введите пароль доступа к ключу (**`Pa$$w0rd`**).
6. На машине **c7-client01** проверьте возможность подключения к **c7-server02** от имени пользователя root.
7. Сгенерируйте пару ключей RSA, сохраните идентификатор в файле **`/root/.ssh/id_ecdsa`** и не используйте пароль для защиты ключа.
8. Сделайте копию открытого ключа на машину **c7-server01** под пользователем root.
9. Проверьте возможность подключения к **c7-server01** и к **c7-server02**.
10. Просмотрите содержимое файла **`~/.ssh/known_hosts`**.



Ответы к упражнению 3 – Настройка аутентификации OpenSSH на базе ключей.

1. Войдите на машину **c7-client01** от имени пользователя root.
2. Проверьте возможность подключения к **c7-server01** от имени пользователя root.

```
# ssh root@server01
```

3. Сгенерируйте пару ключей RSA, сохраните идентификатор в файле **/root/.ssh/id_rsa** и защитите ключ при помощи пароля **Pa$$w0rd**.

```
# ssh-keygen -t rsa
```

4. Сделайте копию открытого ключа на машину **c7-server01** под пользователем root.

```
# ssh-copy-id root@server01
```

5. Дважды проверьте возможность подключения к **c7-server01**, при первом подключении введите пароль доступа к ключу (**Pa$$w0rd**).

```
# ssh root@server01
```

6. На машине **c7-client01** проверьте возможность подключения к **c7-server02** от имени пользователя root.

```
# ssh root@server02
```

7. Сгенерируйте пару ключей RSA, сохраните идентификатор в файле **/root/.ssh/id_ecdsa** и не используйте пароль для защиты ключа.

```
# ssh-keygen -t ecdsa
```

8. Сделайте копию открытого ключа на машину **c7-server01** под пользователем root.

```
# ssh-copy-id root@server02
```

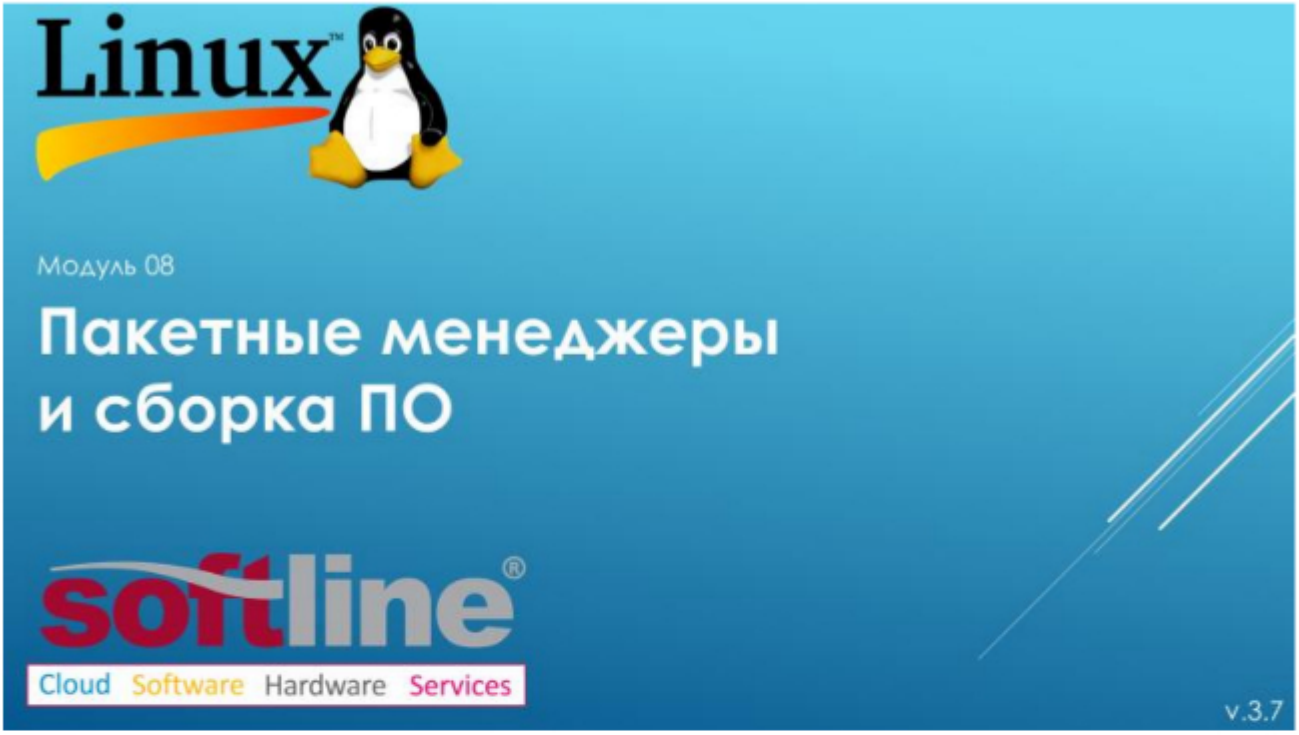
9. Проверьте возможность подключения к **c7-server01** и к **c7-server02**.

```
# ssh root@server01
# ssh root@server02
```

10. Просмотрите содержимое файла **~/.ssh/known_hosts**.

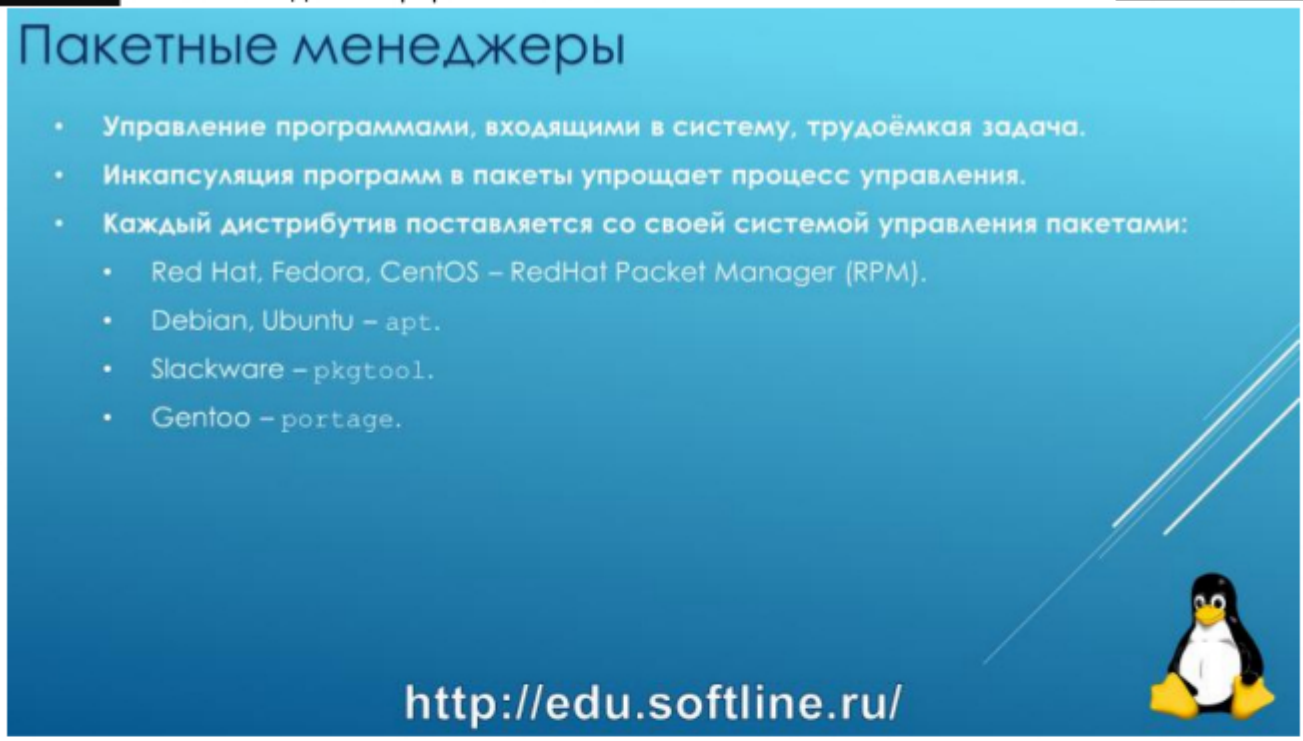
```
# cat ~/.ssh/known_hosts.
```

Модуль 8: Пакетные менеджеры и сборка ПО.



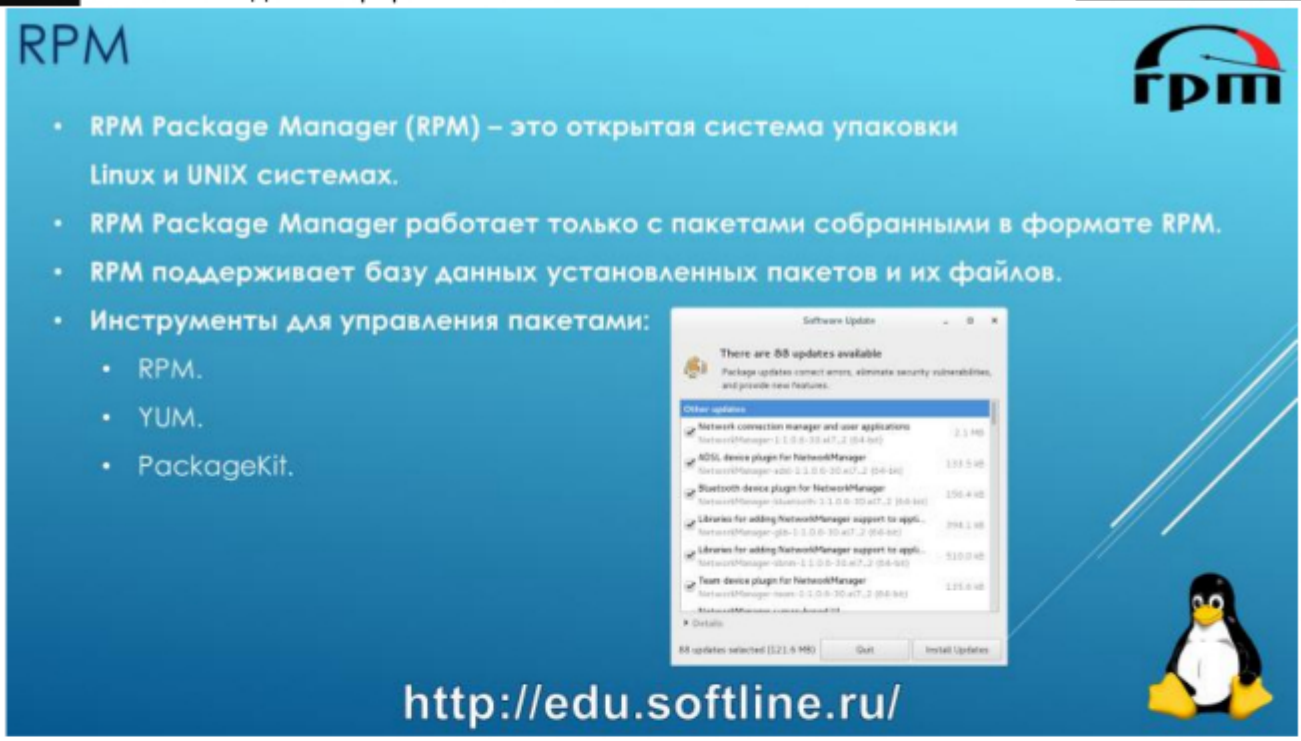
В модуле представлены:

- Управление пакетами при помощи RPM.
- Управление репозиториями при помощи YUM.
- Компиляция программ из исходных кодов.
- Запуск приложений Windows в среде Linux.



Основной целью всех пакетных менеджеров является удобство управления программными пакетами, предоставляя возможность устанавливать и удалять их, обновлять. Получать информацию о программе, отслеживать необходимые зависимости.

Существует большое множество форматов пакетов и систем управления ими. Одним из основоположников является "RedHat Packet Manager (RPM). Каждый пакет содержит описание, архитектуру, для которой программа была скомпилирована, зависимости и т.д. Основным конкурентом RPM является формат пакетов DEB, используемый в дистрибутиве Debian.



RPM Package Manager (RPM) – это открытая система упаковки, которая используется в CentOS (RHEL) и других Linux и UNIX системах. Корпорация Red Hat и проект Fedora, а также многие другие производители, в том числе и CentOS, выбрали RPM для своих продуктов. RPM распространяется под лицензией GPL (GNU General Public License).

RPM Package Manager работает только с пакетами собранными в формате RPM. RPM предоставляется как предустановленный пакет. Для конечных пользователей RPM упрощает обновление системы. Установка, удаление и обновление пакетов RPM может быть выполнена при помощи короткой команды. RPM поддерживает базу данных установленных пакетов и их файлов, что позволяет выполнять запросы и проверять систему. Также в системе присутствует ряд приложений, таких как Yum или PackageKit, которые упрощают работу с пакетами в формате RPM.

ВНИМАНИЕ:

Для большинства задач управления пакетами, пакетный менеджер Yum предлагает эквивалент, как правило с большими возможностями. Yum также отслеживает и разрешает зависимости пакетов. Yum поддерживает целостность системы, принудительно проверяя целостность при установке или удалении пакетов при помощи сторонних приложений, таких как RPM. Настоятельно рекомендуется использовать Yum для задач управления пакетами вместо RPM.

Если вы предпочитаете графический интерфейс – используйте графическое приложение PackageKit, которое является графической надстройкой над Yum.

ПРИМЕЧАНИЕ:

При установке пакета, убедитесь, что он совместим с вашей архитектурой процессора. Обычно это легко определить при помощи имени пакета. Например, имя файла пакета, скомпилированного под архитектуру AMD64/Intel 64 заканчивается на **`x86_64.rpm`**

В процессе обновления пакета, RPM осторожно обрабатывает конфигурационные файлы таким образом, чтобы вы не потеряли ваши изменения – то, чего не возможно добиться с обычными файлами **`.tar.gz`**.

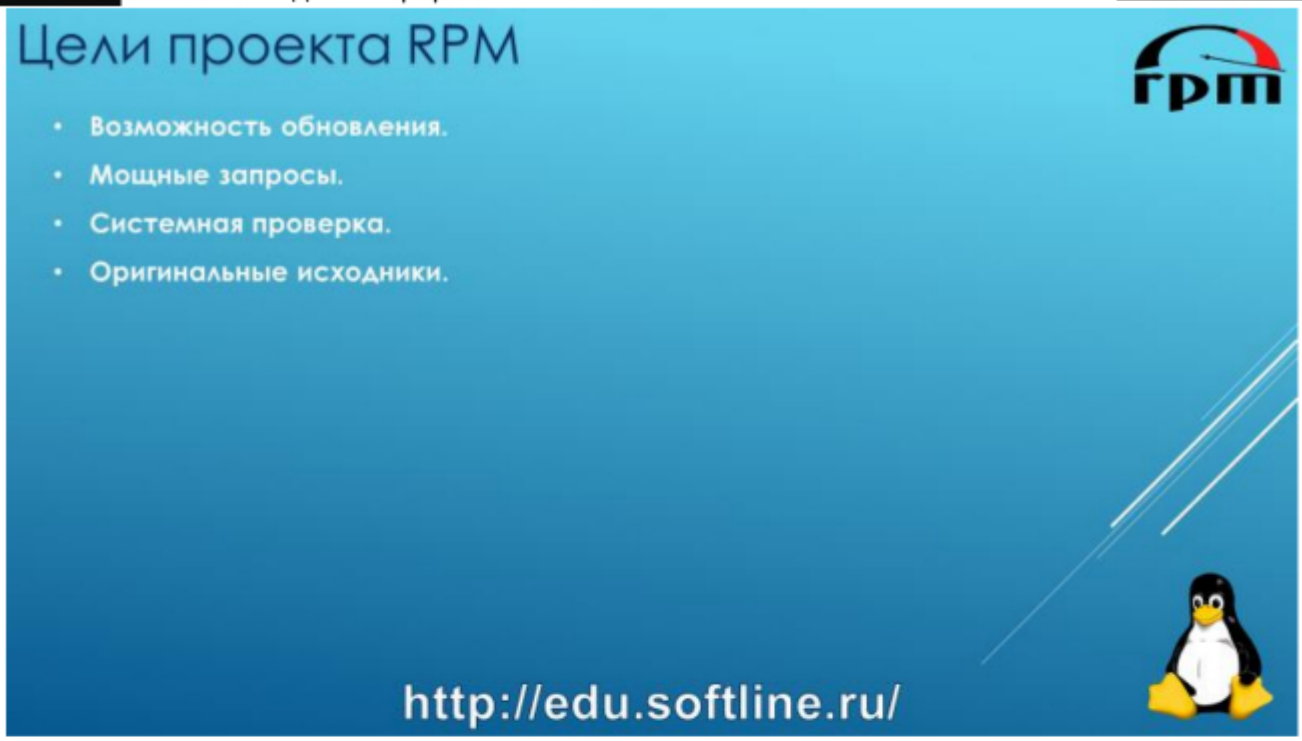
Для разработчиков, RPM позволяет взять исходные коды и упаковать их в исходные и бинарные пакеты для конечных пользователей. Этот процесс существенно упрощает доставку приложений и предоставляет единый

файл и опциональные обновления. Возможность упаковки исходных файлов и инструкций по обновлению упрощает поддержку пакетов и обновление до последних версий.



ПРИМЕЧАНИЕ:

Так как RPM может вносить изменения в систему, выполняя установку, обновление, понижение версий и установку бинарных системных пакетов, в большинстве случаев требуются привилегии root. В настройках по умолчанию исключения составляют только домашний каталог и каталоги, созданные администратором (с соответствующими правами).



Для того чтобы понять, как использовать RPM, будет полезно понять цели проекта RPM.

Возможность обновления.

При помощи RPM можно обновить отдельный компонент системы без полной переустановки системы. После релиза новой версии операционной системы RPM позволяет интеллектуально, полностью автоматически обновить систему на месте (In-place Upgrade). Дополнительно, конфигурационные файлы будут сохранены неизменными в процессе обновления, таким образом, чтобы вы не потеряли ваши настройки. Для обновления не требуются дополнительные файлы, так как и обновление и установка производится из одних и тех же файлов.

Мощные запросы.

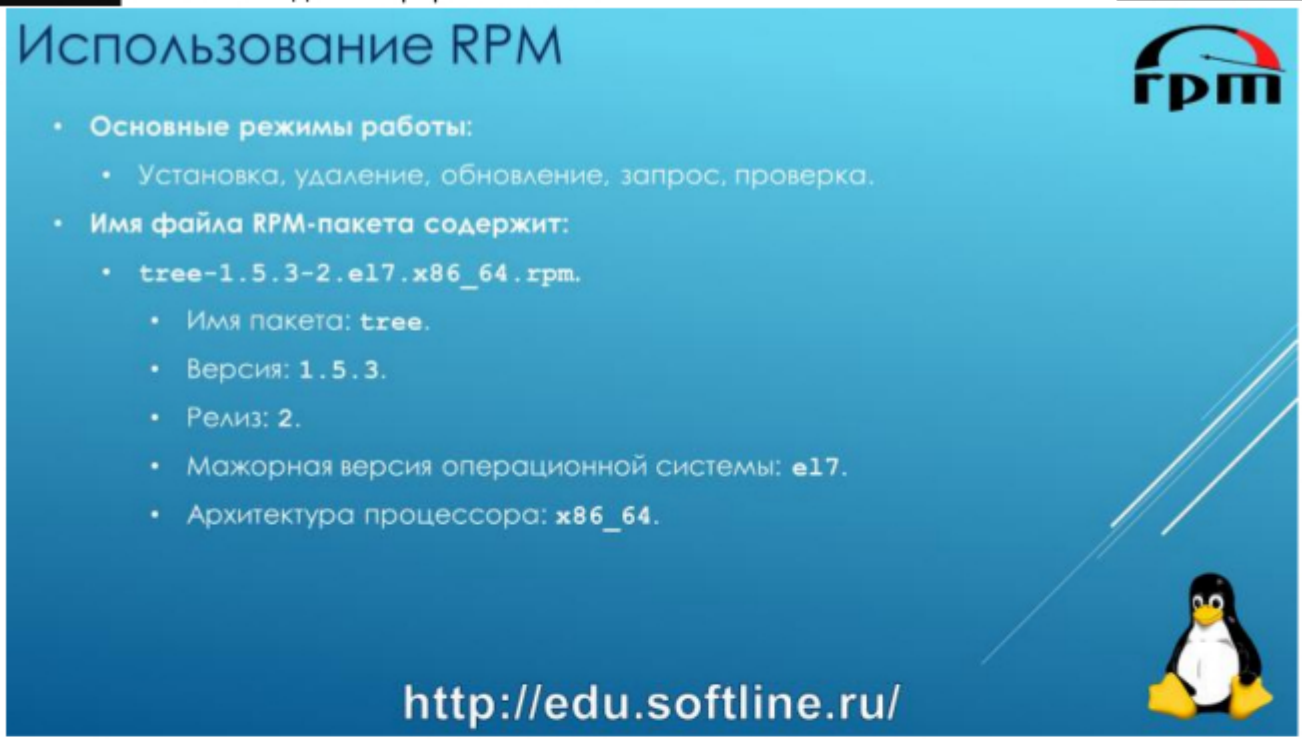
RPM спроектирован для предоставления мощной системы создания запросов. Присутствует возможность создания запросов из базы данных по установленным пакетам и или их файлам. Вы можете легко определить из какого пакета взялся тот или иной файл, а также откуда взялся тот или иной пакет. Файлы, содержащиеся в RPM-пакете, хранятся в сжатом архиве с бинарным заголовком, содержащем полезную информацию о пакете и его содержимом, что позволяет легко и быстро делать запросы к отдельному пакету.

Системная проверка.

Другая мощная возможность RPM – это возможность проверять пакеты. Если случайно был удален важный файл, какого-либо установленного пакета, вы можете проверить пакет. Вы будете уведомлены обо всех аномалиях и сможете переустановить пакет. Любые конфигурационные файлы, которые вы изменяли, будут сохранены в процессе переустановки.

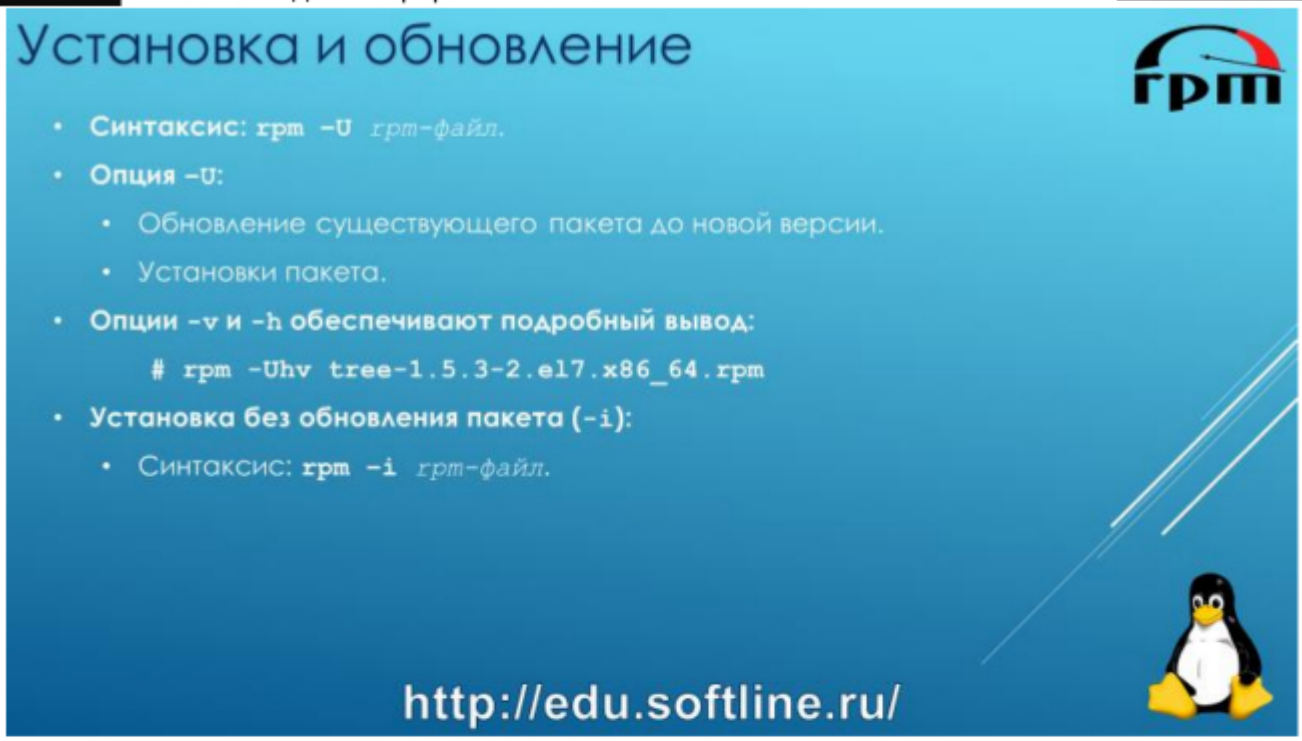
Оригинальные исходники.

Критически важная цель в проектировании RPM – обеспечение возможности использования нетронутых исходных кодов программного обеспечения, такими, какими их заложили авторы. Вместе с RPM у вас имеются все исходные коды, распространяемые вместе со всеми обновлениями, которые могут быть использованы с инструкциями по сборке. Это важное преимущество, которое может быть использовано для различных целей. Может показаться, что данное преимущество имеет значение только для разработчиков, но на самом деле это гарантирует высокое качество программного обеспечения.



RPM имеет пять основных режимов работы (за исключением сборки пакета): установка, удаление, обновление, запрос и проверка. Далее мы рассмотрим каждый из этих режимов. Для получения полной информации обо всех опциях используйте **`rpm --help`** или **`man rpm`**.

RPM-пакеты, как правило имеют имя похожее на **`tree-1.5.3-2.el7.x86_64.rpm`**. Имя файла содержит имя пакета (**`tree`**), версию (**`1.5.3`**), релиз (**2**), мажорную версию операционной системы (**`el7`**) и архитектуру CPU (**`x86_64`**).



Вы можете использовать опцию **–U** для:

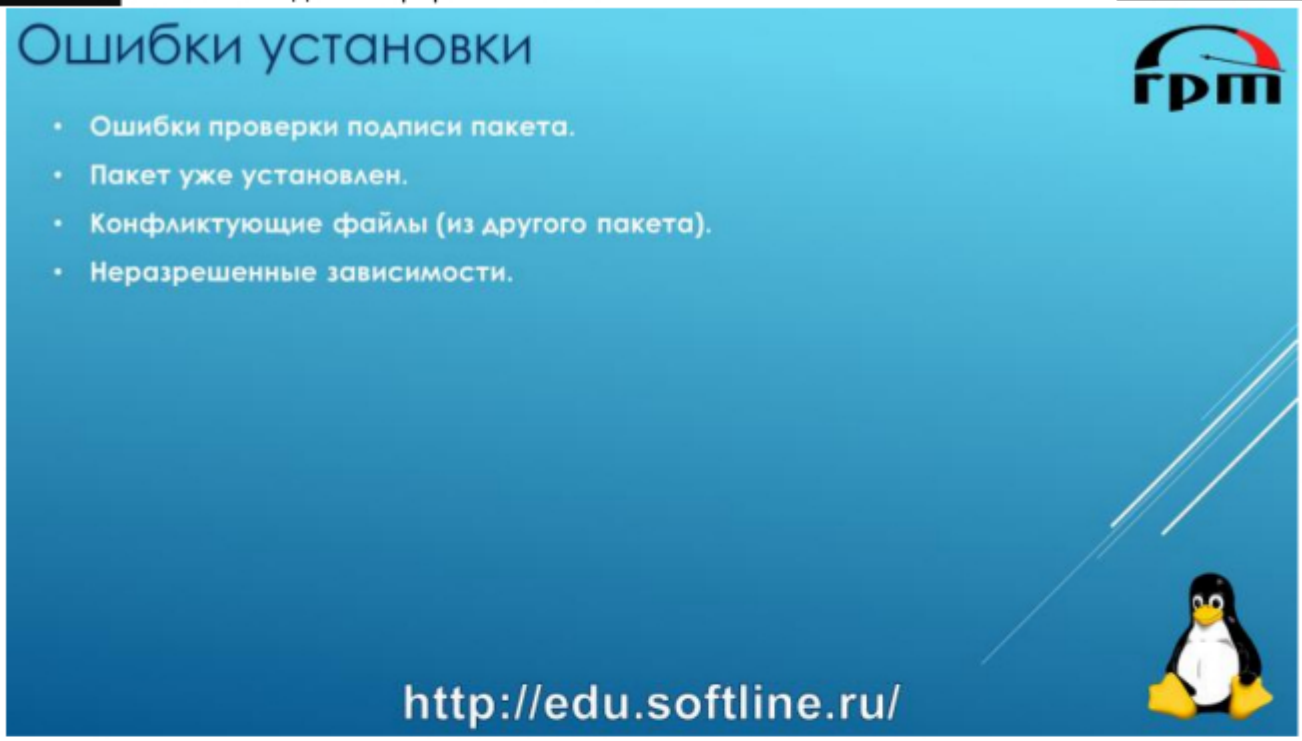
- Обновления существующего, старого пакета до новой версии.
- Установки пакета.

Синтаксис: **rpm –U *<rpm_file>***

Для установки или обновления пакета **tree-1.5.3-2.el7.x86_64.rpm**, выполните следующую команду с правами пользователя root:

```
# rpm -Uhv tree-1.5.3-2.el7.x86_64.rpm
```

ПРИМЕЧАНИЕ:
*Опции **-v** и **-h** дают указание **rpm** вывести более подробный вывод, отображая прогресс при помощи **#**.*



Если обновление/установка прошли успешно, следующий вывод будет отображен.

```
Preparing...  ########################################### [100%]
1:tree        ########################################### [100%]
```

ВНИМАНИЕ:

rpm *предоставляет две разные опции для установки пакетов: опцию* ***-U*** *(исторически обозначающую update) и опцию* ***-i****, предназначенную для установки. Настоятельно рекомендуется использовать опцию* ***-U*** *для установки всех пакетов за исключением ядра операционной системы Linux.*

Всегда используйте опцию ***-i*** *(install) для установки пакета нового ядра, вместо обновления. Так как опция* ***-U*** *обновляет ядро, удалив старое ядро и, если что-то пойдет не так вы не сможете загрузиться с использованием старого ядра. Поэтому всегда используйте команду* ***rpm -i*** *<пакет_ядра>.*

Проверка подписи пакета.

Подпись пакета проверяется автоматически при установке и обновлении пакета. Подпись подтверждает, что пакет подписан авторизованной стороной. Например, если проверка подписи завершится неудачно, вы получите следующее сообщение:

```
error: tree-1.5.3-2.el7.x86_64.rpm : Header V3 RSA/SHA256 signature: BAD, key
ID d22e77f2
```

Если на системе не установлен необходимый ключ для проверки подписи, вы получите следующее сообщение, содержащее слово **NOKEY**:

```
warning: tree-1.5.3-2.el7.x86_64.rpm : Header V3 RSA/SHA1 signature: NOKEY,
key ID 57bbccba
```

Пакет уже установлен.

Если пакет с таким же именем и такой же версией уже установлен вы получите следующий вывод:

```
Preparing...  ########################################### [100%]
package tree-1.5.3-2.el7.x86_64 is already installed
```

Однако, если вы хотите переустановить пакет, используйте опцию **--replacepkgs**, которая заставляет RPM игнорировать ошибки.

```
# rpm -Uhv --replacepkgs tree-1.5.3-2.el7.x86_64.rpm
```

Эта опция полезна, если файлы из пакета были удалены или вы хотите вернуть оригинальные конфигурационные файлы из пакета.

Конфликтующие файлы.

Если вы попытаетесь установить пакет, который содержит файлы, которые уже были установлены из другого пакета, вы получите следующее сообщение:

```
Preparing... ##################################################
file /usr/bin/foobar from install of foo-1.0-1.el7.x86_64 conflicts
with file from package bar-3.1.1.el7.x86_64
```

Для того, чтобы проигнорировать такую ошибку, воспользуйтесь опцией **--replacefiles**:

```
# rpm -Uvh --replacefiles foo-1.0-1.el7.x86_64.rpm
```

Неразрешенные зависимости.

Иногда RPM-пакет может зависеть от других пакетов, это значит, что другие пакеты должны быть установлены на системе, чтобы приложение запускалось корректно. Если вы попробуете установить пакет, который содержит неразрешенные зависимости, вы получите следующий вывод:

```
error: Failed dependencies:
bar.so.3()(64bit) is needed by foo-1.0-1.el7.x86_64
```

Если вы производите установку с установочного накопителя вы можете найти зависимые пакеты на нем или выгрузить их из иных источников. Для того чтобы установить пакет вместе с зависимостью, добавить название зависимого пакета в команду **rpm**:

```
# rpm -Uvh foo-1.0-1.el7.x86_64.rpm bar-3.1.1.el7.x86_64.rpm
```

Если установка обоих пакетов завершится успешно, вы получите следующий вывод:

```
Preparing... ########################################### [100%]
1:foo        ########################################### [ 50%]
2:bar        ########################################### [100%]
```

Вы можете использовать опцию **--whatprovides** для определения пакета, содержащего необходимый файл:

```
# rpm -q --whatprovides "bar.so.3"
```

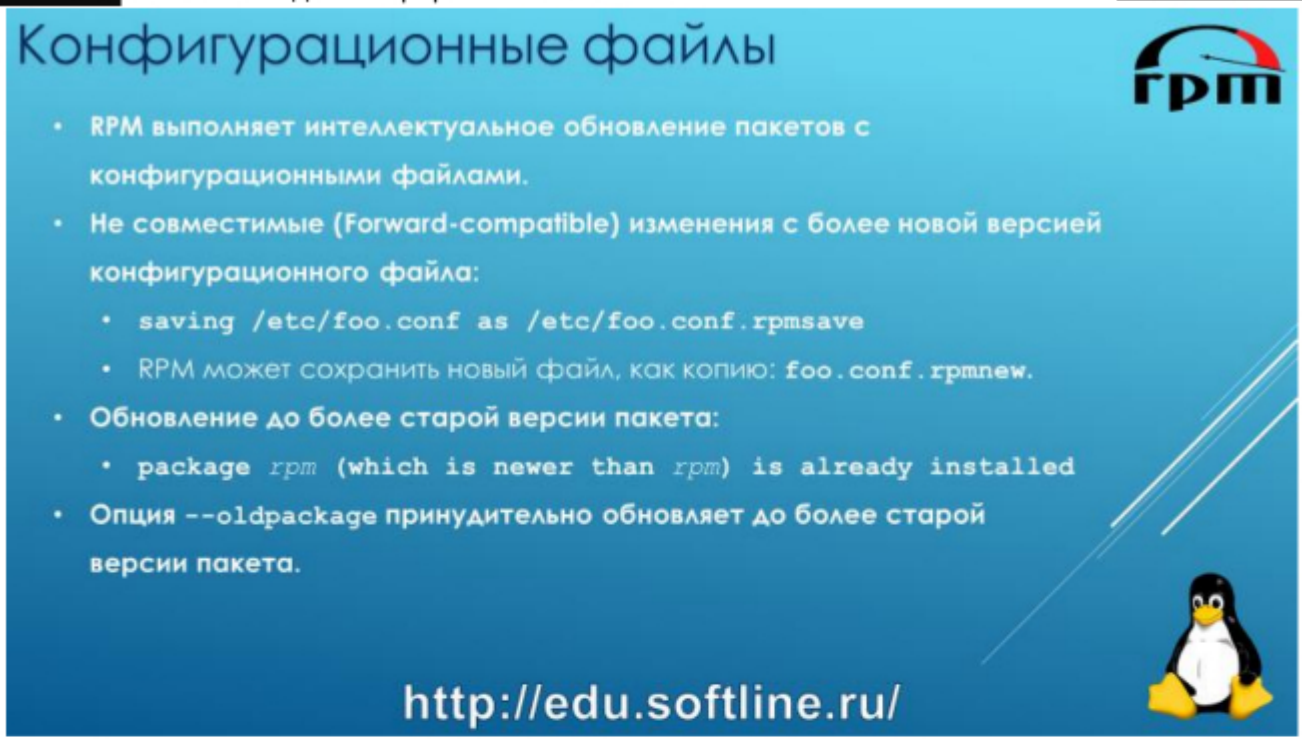
Если пакет, который содержит файл **bar.so.3** содержится в базе данных RPM, имя пакета будет отображено:

```
bar-3.1.1.el7.i586.rpm
```



ВНИМАНИЕ:

*Также вы можете принудительно поставить пакет, который содержит неразрешенные зависимости (при помощи опции **--nodes**), но этого не рекомендуется делать, так как обычно такая установка не позволяет запускаться приложению. Установка или удаление пакетов с использованием опции --node может привести к неисправности приложений и серьезным проблемам с управлением пакетов, в том числе к системным ошибкам.*



Так как RPM выполняет интеллектуальное обновление пакетов с конфигурационными файлами, вы можете увидеть одно или несколько подобных сообщений:

```
saving /etc/foo.conf as /etc/foo.conf.rpmsave
```

Данное сообщение обозначает, что изменения, которые вы внесли в конфигурационный файл, не совместимы (Forward-compatible) с более новой версией конфигурационного файла в пакете, RPM в свою очередь сохранил измененный конфигурационный файл и установил новый. Далее необходимо вручную исправить новый конфигурационный файл, чтобы система оставалась настроенной.

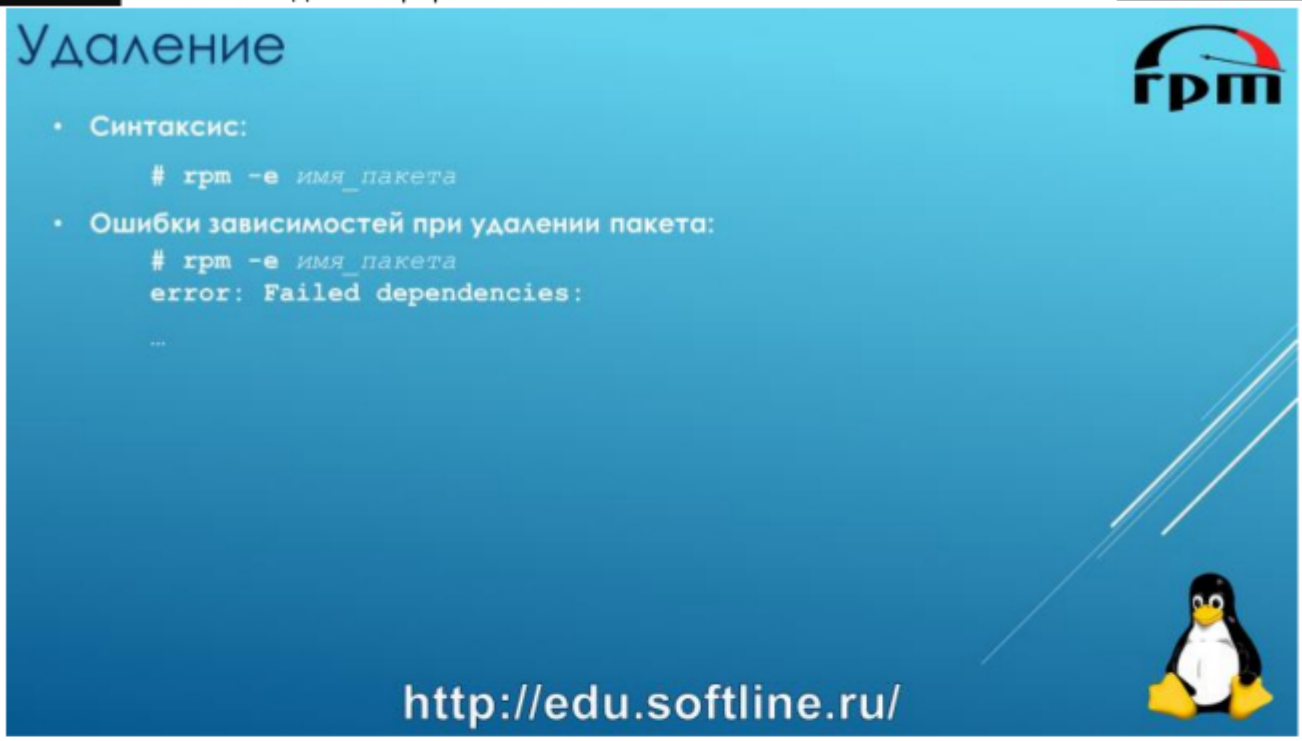
Альтернативно RPM может сохранить новый конфигурационный файл, как копию, например, **foo.conf.rpmnew**, и оставить измененный конфигурационный файл нетронутым. Далее необходимо будет разрешить конфликты конфигурационного файла, например, совместив различия при помощи программы **diff**.

Если вы пробуете обновиться до более старой версии пакета (чем установленная в системе), вы получите следующее сообщение:

```
package foo-2.0-1.el7.x86_64.rpm (which is newer than foo-1.0-1) is already
installed
```

Для принудительного обновления до более старой версии пакета используйте опцию **--oldpackage**:

```
# rpm -Uvh --oldpackage foo-1.0-1.el7.x86_64.rpm
```



Удаление пакета происходит также просто, как и установка. При помощи следующей команды:

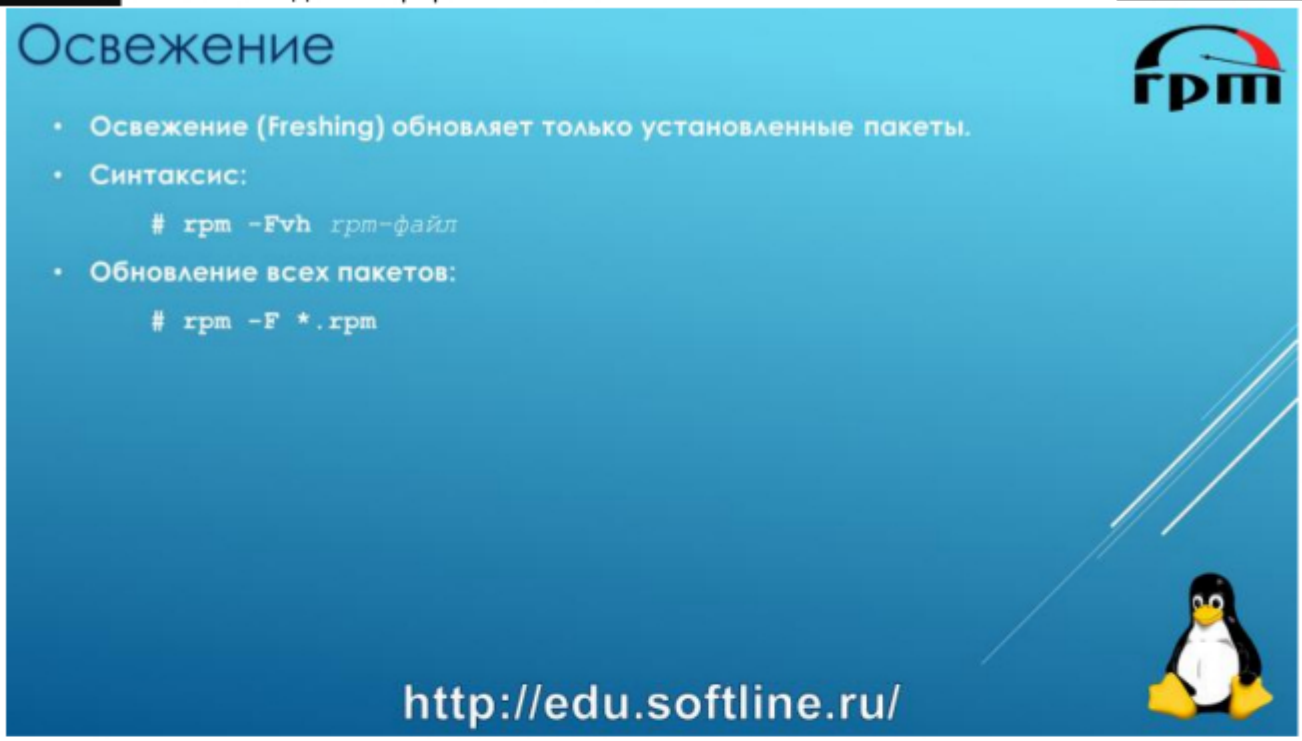
```
# rpm -e foo
```

ПРИМЕЧАНИЕ:

Обратите внимание, что команда использует имя пакета ***foo****, а не имя оригинального файла пакета* ***foo-1.0-1.el7.x86_64****. Если вы попытаетесь удалить пакет при помощи команды* ***rpm -e*** *и полного имени файла, то вы получите ошибку.*

Вы можете столкнуться с ошибками зависимостей при удалении пакета, если от него зависят другие пакеты. Например:

```
# rpm -e ghostscript
error: Failed dependencies:
libgs.so.8()(64bit) is needed by (installed) libspectre-0.2.2-3.el7.x86_64
libgs.so.8()(64bit) is needed by (installed) foom atic-4.0.3-1.el7.x86_64
libijs-0.35.so()(64bit) is needed by (installed) gutenprint-5.2.4-
5.el7.x86_64
ghostscript is needed by (installed) printer-filters-1.1-4.el7.noarch
```



Освежение (Freshing) похоже на обновление, за исключением того, что обновляются только установленные пакеты. Для освежения используйте следующую команду:

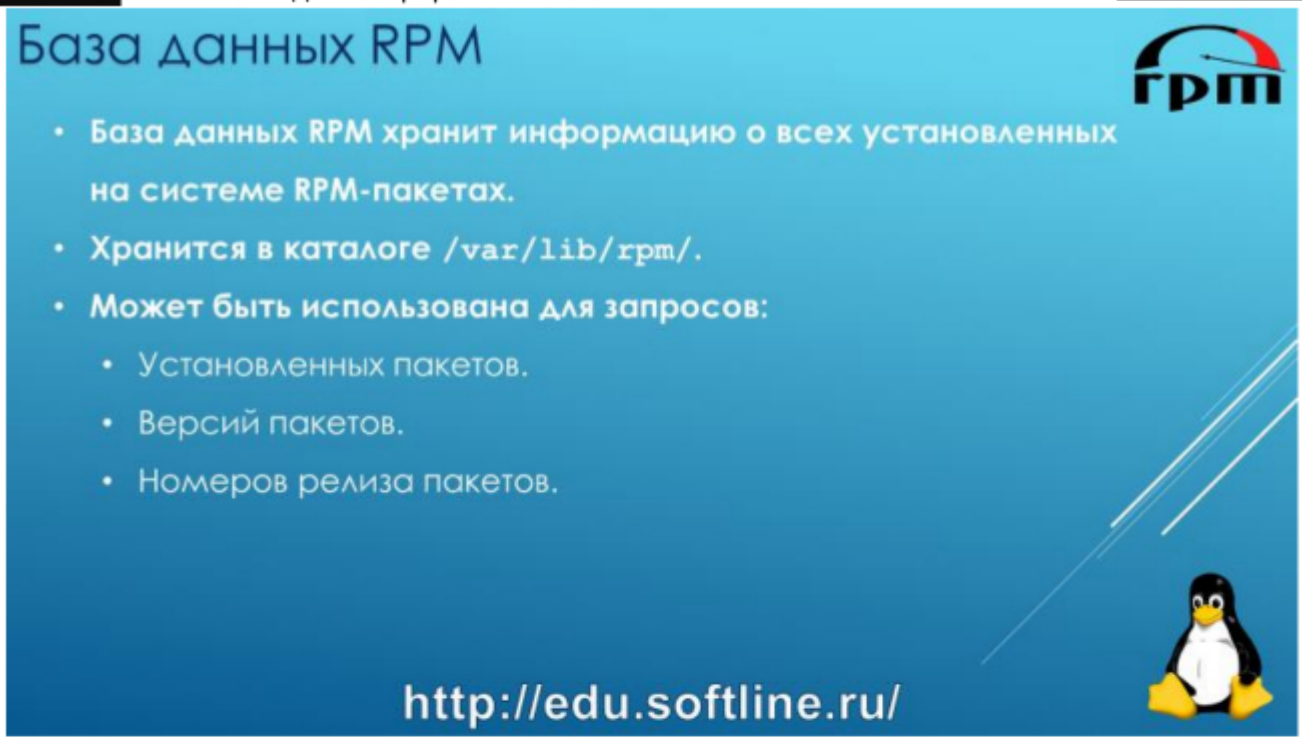
```
# rpm -Fvh foo-2.0-1.el7x86_64.rpm
```

Опция освежения пакета проверяет версию пакета, указанного в команде и сравнивает с версией установленного в системе пакета. Когда опция освежения обрабатывает пакет более новой версии, чем установленный в системе, пакет будет обновлен до новой версии. Однако, опция освежения не устанавливает пакет, если он отсутствует в системе, этим опция освежения (**-F**) отличается от опции обновления (**-U**).

В следующем примере обновляются все пакеты, для которых доступны обновления в текущей папке:

```
# rpm -F *.rpm
```

При этом обновляются только установленные в системе пакеты.



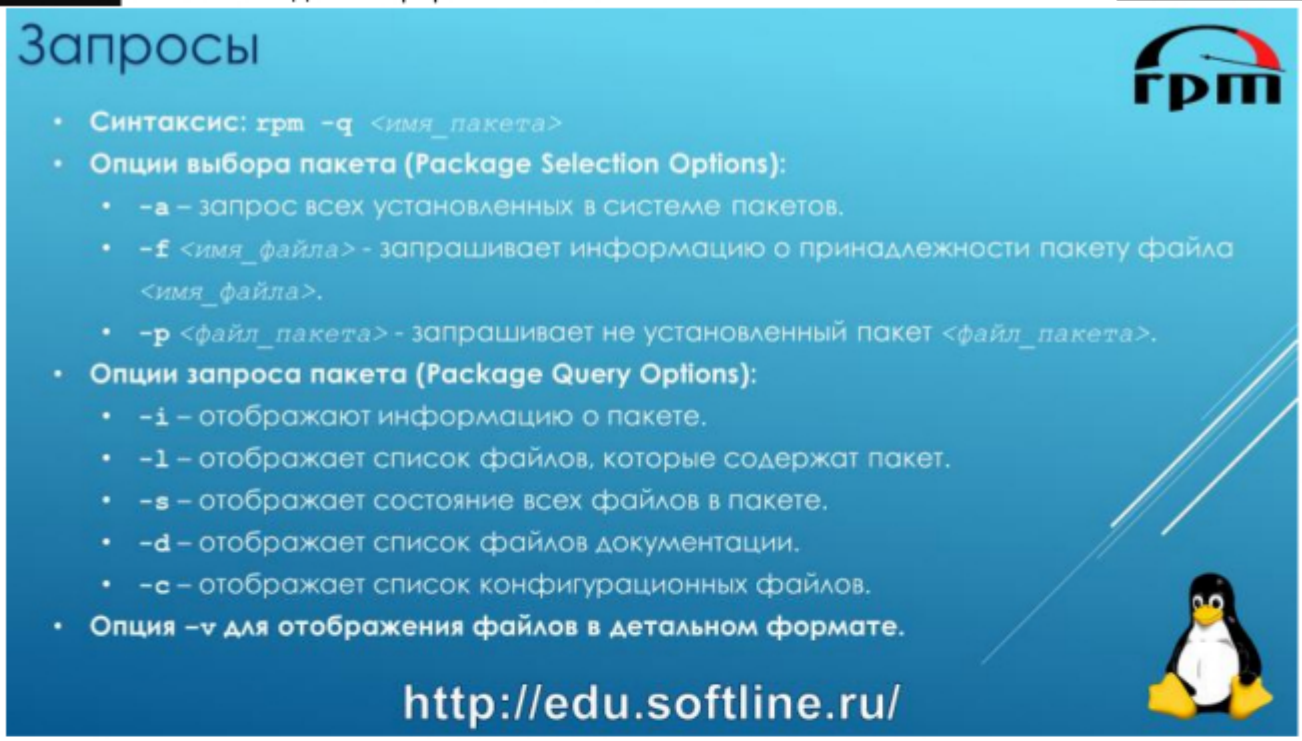
База данных RPM хранит информацию о всех установленных на системе RPM-пакетах. Она хранится в каталоге **/var/lib/rpm/**:

```
[root@c7-server01]# ls /var/lib/rpm -1
Basenames
Conflictname
__db.001
__db.002
__db.003
Dirnames
Group
Installtid
Name
Obsoletename
Packages
Providename
Requirename
Sha1header
Sigmd5
Triggername
```

Каждый файл – это отдельная база данных в формате Berkeley DB, за исключением нескольких файлов данных **__db**.

ПРИМЕЧАНИЕ:

Библиотеки Berkeley DB предоставляют простой API базы данных. Это не традиционная реляционная база данных. В ней значения данных зранятся в виде хеш-таблицы в парах имя/значение. Подобный тип баз данных очень быстр для просмотра именованных записей (таких как имена пакетов), но совсем не такой быстрый при итеративных запросах по всем записям.



К базе данных RPM можно обращаться, например, при помощи **rpm -q tree**, можно запросить информацию об установленном пакете **tree** и должны получить следующий вывод:

```
tree-1.5.2.2-4.el7.x86_64
```

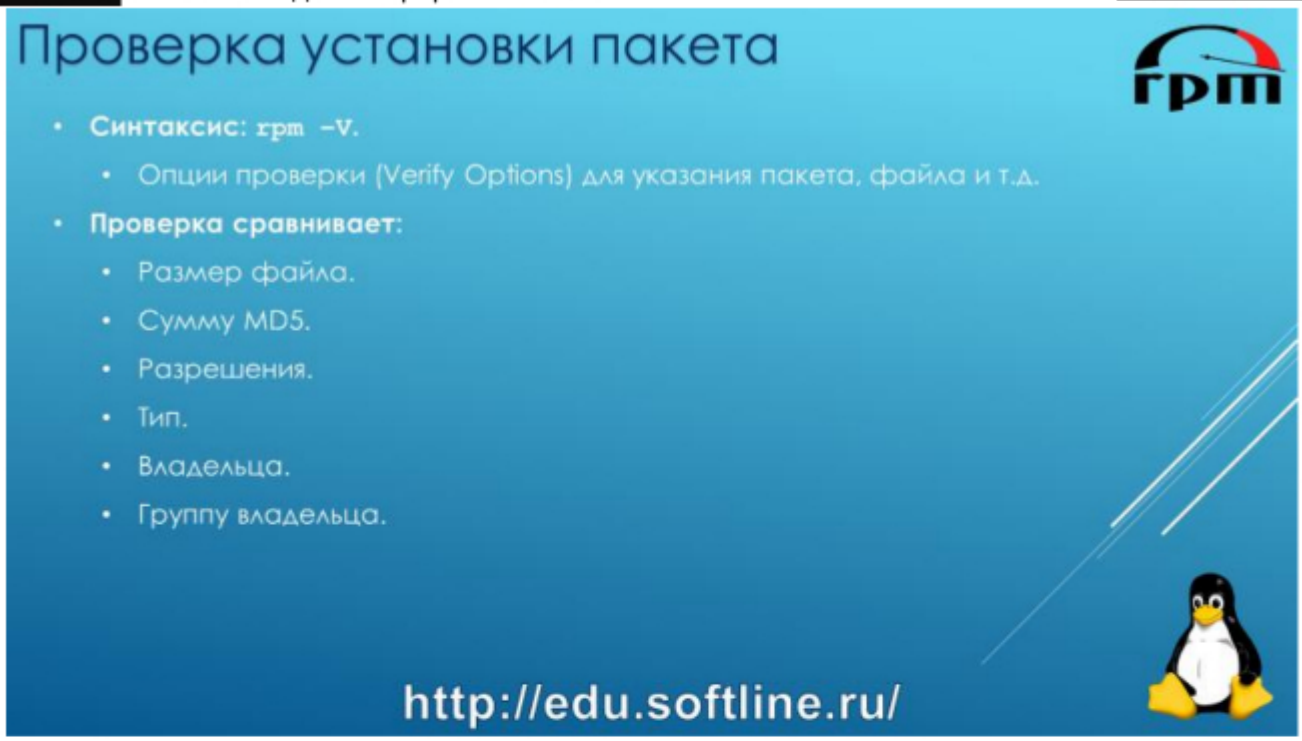
Вы можете также использовать следующие опции выбора пакета (Package Selection Options), которые описаны в страницах руководства man.

- **-a** – запрос всех установленных в системе пакетов.
- **-f** *<имя_файла>* - запрашивает из базы данных RPM информацию о принадлежности пакету файла *<имя_файла>* (Например, **rpm -qf /bin/ls**).
- **-p** *<файл_пакета>* - запрашивает не установленный пакет *<файл_пакета>*.

Есть несколько способов повлиять на отображаемую информацию о запрашиваемом пакете. Следующие опции используются для выбора типа информации для которой построен запрос. Они называются опции запроса пакета (Package Query Options).

- **-i** – Отображает информацию о пакете, в том числе имя, описание, релиз, размер, дату создания, дату установки и производителя.
- **-l** – Отображает список файлов, которые содержат пакет.
- **-s** – Отображает состояние всех файлов в пакете.
- **-d** – Отображает список файлов, помеченных как документация (страницы руководства **man**, страницы **info**, файлы **README** и т.д.).
- **-c** – отображает список файлов, помеченных как конфигурационные файлы. Это файлы, которые вы изменяете после установки для адаптации и настройки пакета на вашей системе (Например: **sendmail.cf**, **passwd**, **initttab** и так далее).

Для опций, который отображают список файлов, добавьте опцию **-v**, чтобы отображать файлы в формате похожем на команду **ls -l**.



Проверка пакета, сравнивает информацию о файлах, установленных из пакета с информацией о файлах из оригинального пакета. Среди прочего, проверка сравнивает размер файла, сумму MD5, разрешения, тип, владельца и группу владельца каждого фала.

Команда **rpm -V** проверяет пакет. Вы можете использовать любую опцию проверки (Verify Options) из раздела про запросы, чтобы указать пакеты, которые необходимо проверить. Простое использование проверки – это: rpm -V tree, которое проверяет, что все файлы в пакете **tree** такие же, какими они были сразу после установки. Например:

```
# rpm -Vf /usr/bin/tree
```

В данном примере **/usr/bin/tree** – это абсолютный путь до файла, который необходимо проверить.

Для проверки всех установленных на системе пакетов (потребуется некоторое количество времени), используйте следующую команду:

```
# rpm -Va
```

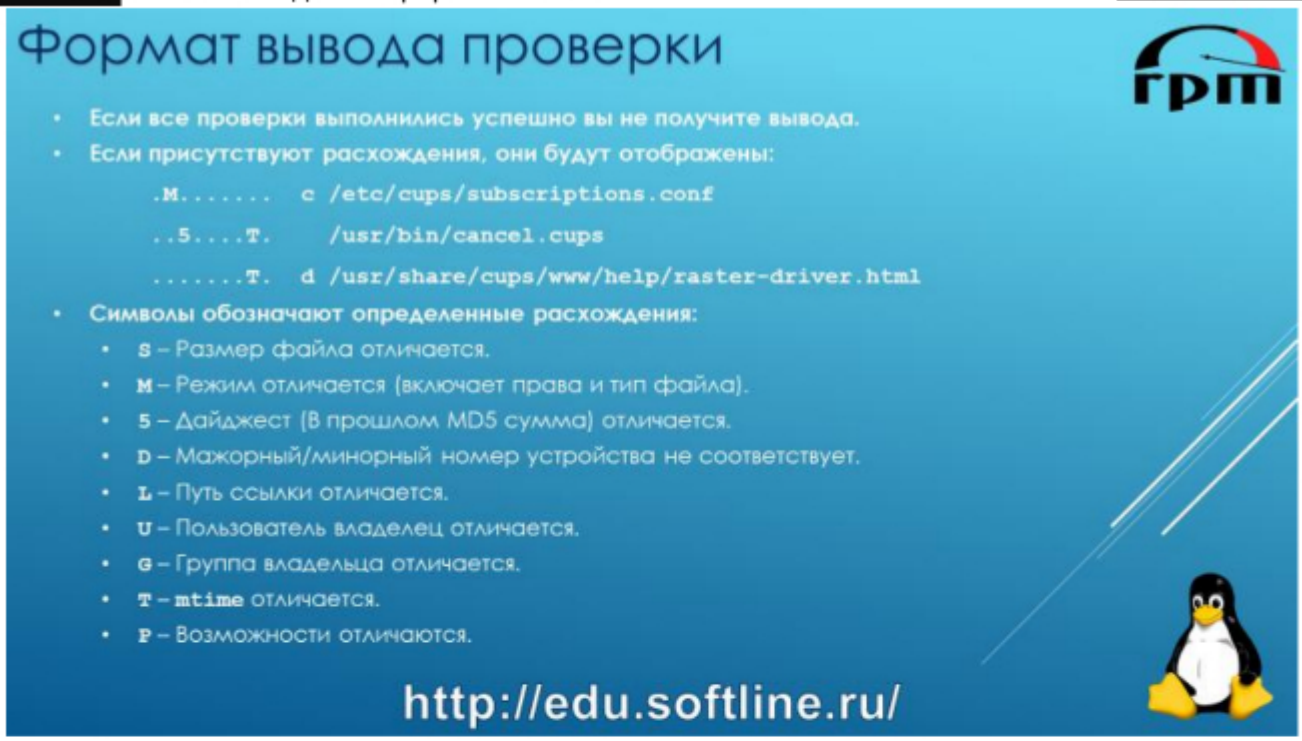
Для проверки установленного пакета при помощи файла пакета, используйте следующую команду:

```
# rpm -Vp tree-1.5.3-2.el7.x86_64.rpm
```

Данная команда может быть полезна, если вы подозреваете, что база данных RPM повреждена.

Если все проверки выполнились успешно вы не получите вывода. Если присутствуют расхождения, они будут отображены:

```
# rpm -Vp  cups-1.7.1-6.fc20.x86_64.rpm
.M.......  c /etc/cups/subscriptions.conf
..5....T.    /usr/bin/cancel.cups
.......T.  d /usr/share/cups/www/help/raster-driver.html
```



Формат вывода – это строка из девяти символов, идущая перед опциональным маркирующим атрибутом и именем файла.

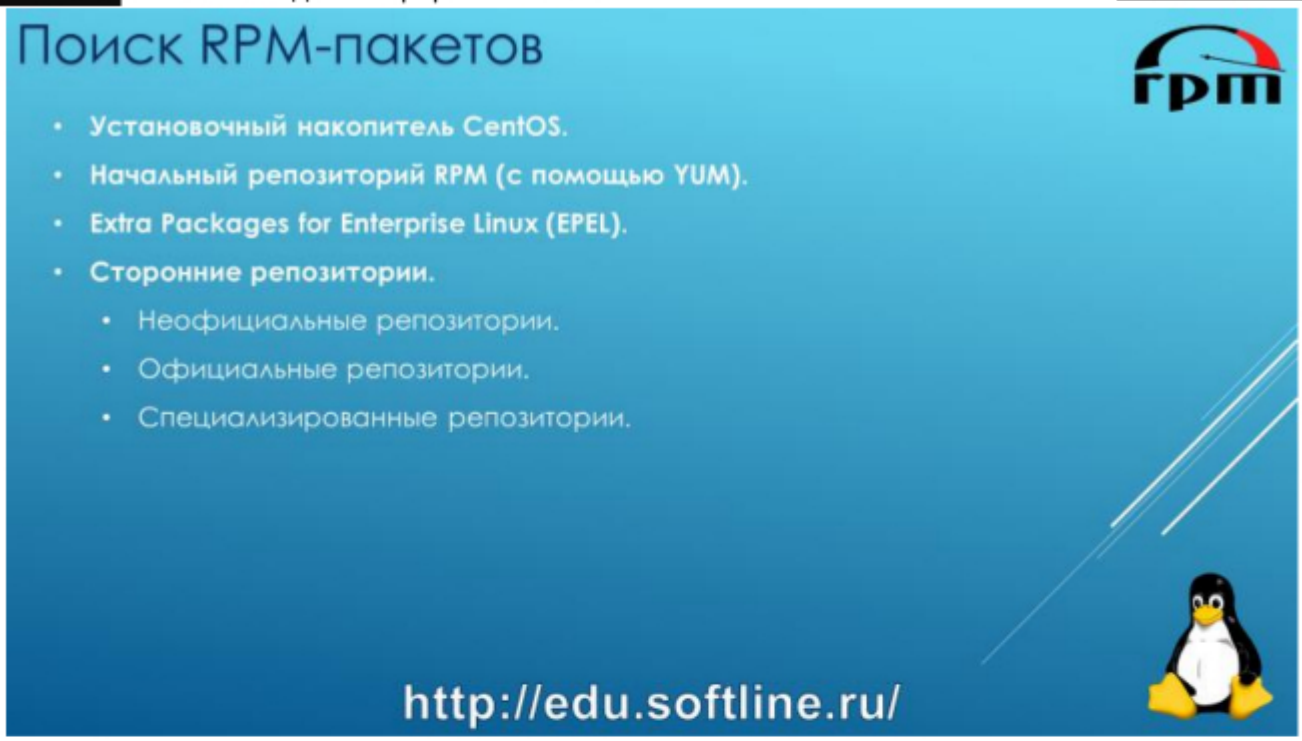
Первые 9 символов – это результат проверки, выполненной над файлом. Каждый тест – это сравнение одного атрибута файла со значением, записанным в базе данных RPM. Отдельно стоящая точка (**.**) обозначает, что тест пройден. Следующие символы обозначают определенные расхождения:

- **S** – Размер файла отличается.
- **M** – Режим отличается (включает права и тип файла).
- **5** – Дайджест отличается (в прошлом контрольная сумма MD5).
- **D** – Мажорный/минорный номер устройства отличается.
- **L** – Путь ссылки отличается.
- **U** – Пользователь владелец отличается.
- **G** – Группа владельца отличается.
- **T** – **mtime** отличается (время последнего изменения файла).
- **P** – Возможности отличаются.

Маркирующие атрибуты, если представлен, описывает назначение файла. Маркеры атрибутов включают в себя:

- **c** – Конфигурационный файл.
- **d** – Файл документации.
- **l** – Файл лицензии.
- **r** – Файл README.

В случае отображения любого вывода, используйте любой метод проверки установки пакета, если он удален, переустановите его для устранения проблемы.



Прежде чем использовать какие-либо RPM-пакеты, необходимо научиться находить их и узнать, как доверять им.

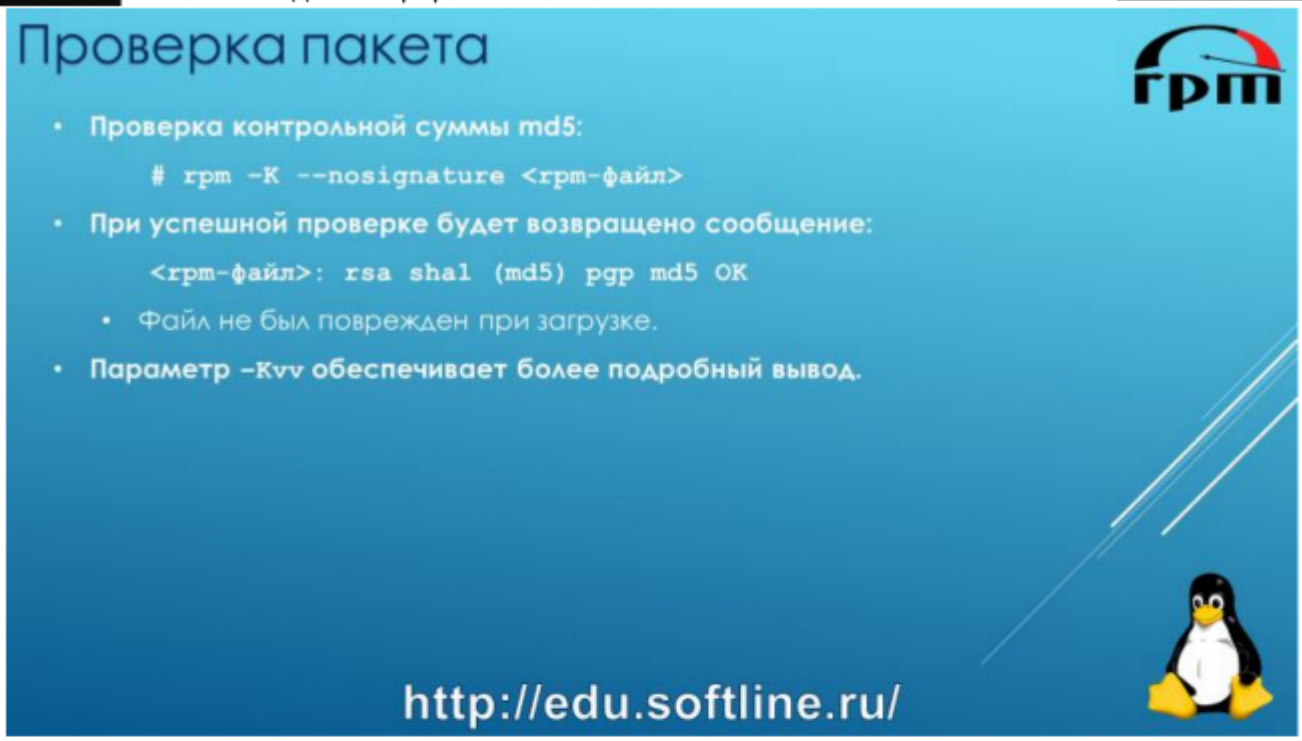
Поиск RPM-пакетов.

Поиск в интернете возвращает множество репозиториев RPM, но если вы посмотрите на RPM-пакеты CentOS, они могут быть найдены в следующих расположениях:

- Установочный накопитель CentOS.
- Начальный репозиторий RPM предоставляемый с помощью пакетного менеджера YUM.
- Поддерживаемый сообществом, высококачественный источник дополнительных пакетов Extra Packages for Enterprise Linux (EPEL) Для получения дополнительной информации о EPEL, воспользуйтесь следующей ссылкой: http://fedoraproject.org/wiki/EPEL.
- Сторонние репозитории.
    - Неофициальные репозитории (например, REMI).
    - Официальные репозитории (например, RedHat и Oracle).
    - Специализированные репозитории (например, ZABBIX или MariaDB).

ПРИМЕЧАНИЕ:

Когда вы думаете об использовании стороннего репозитория для вашей системы с CentOS, внимательно ознакомьтесь с веб-сайтом репозитория, обратив внимание на совместимость, прежде чем добавлять этот репозиторий в качестве источника пакетов. Репозиторий может содержать отличные версии пакетов от репозитория по умолчанию, что в конечном итоге может привести к проблемам совместимости на вашей системе.



Если вы хотите убедиться, что пакет не был поврежден или специально модифицирован, проверьте контрольную сумму **md5** при помощи следующей команды (где *<rpm-файл>* - это имя файла RPM-пакета):

 # rpm -K --nosignature *<rpm-файл>*

Сообщение **<rpm-файл>: rsa sha1 (md5) pgp md5 OK** будет отображено. Это сообщение обозначает, что файл не был поврежден при загрузке. Для получения более подробного вывода замените параметр **-K** на **-Kvv**.

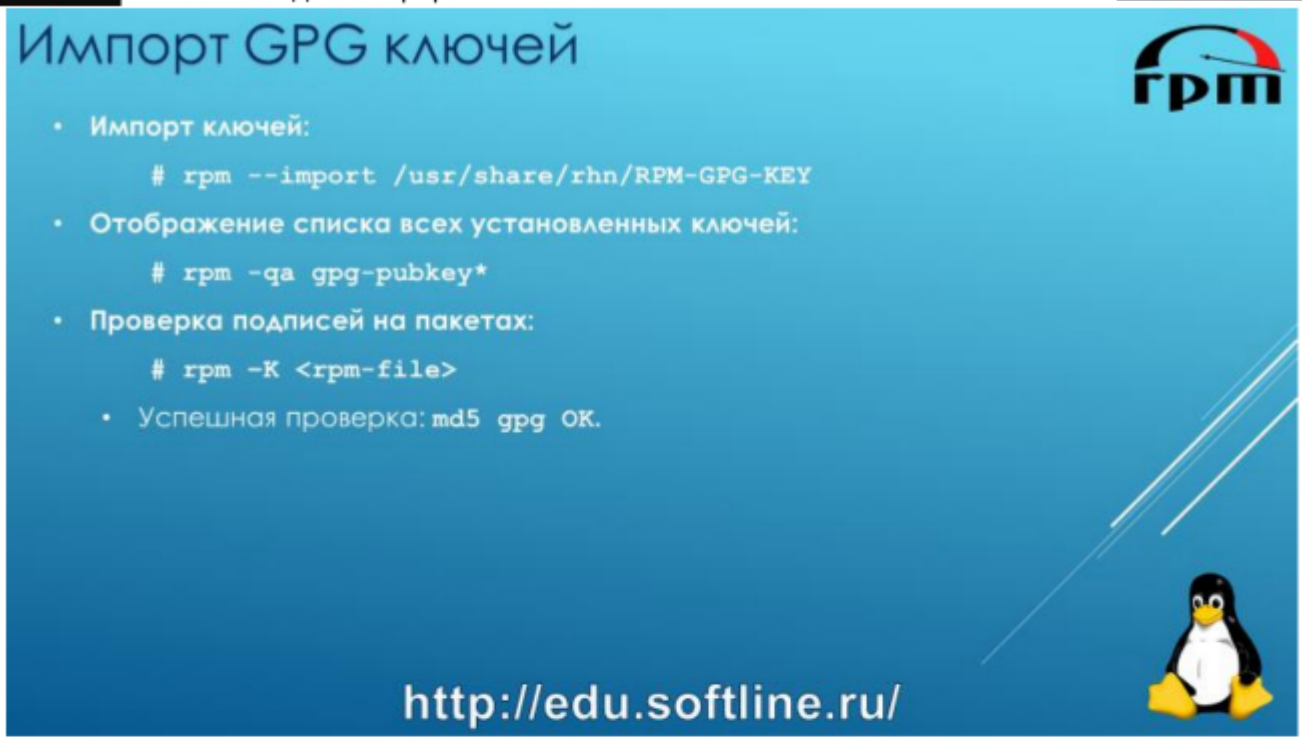
В остальных случаях возникает вопрос, как доверять разработчику, создавшему пакет? Если пакет был подписан ключом GnuPG разработчика, вы знаете, кем является разработчик на самом деле.



RPM-пакет может быть подписан при помощи GNU Privacy Guard (или GnuPG) для помощи в принятии решения о доверии скачанному пакету.

GnuPG – это инструмент для безопасных коммуникаций; это полная и бесплатная замена для технологий шифрования PGP. При помощи GnuPG вы можете аутентифицировать подлинность документа и зашифровать/расшифровать данные от других получателей. Также GnuPG имеет возможность расшифровывать и проверять файлы PGP 5.x.

В процессе установки GnuPG устанавливается по умолчанию. Поэтому вы можете немедленно приступить к использованию GnuPG для проверки любых пакетов, которые вы получили от CentOS. Прежде чем это делать, вы должны импортировать публичный ключ CentOS.



Для проверки пакетов, вы должны импортировать GnuPG ключ CentOS. Для этого выполните следующую команду:

```
rpm --import /usr/share/rhn/RPM-GPG-KEY
```

Для отображения списка всех установленных ключей для проверки RPM, используйте следующую команду:

```
rpm -qa gpg-pubkey*
```

Для ключа Red Hat, вывод будет следующим:

```
gpg-pubkey-db42a60e-37ea5438
```

Для отображения поддробностей об указанном ключе, используйте команду rpm -qi с указанием вывода предыдущей команды:

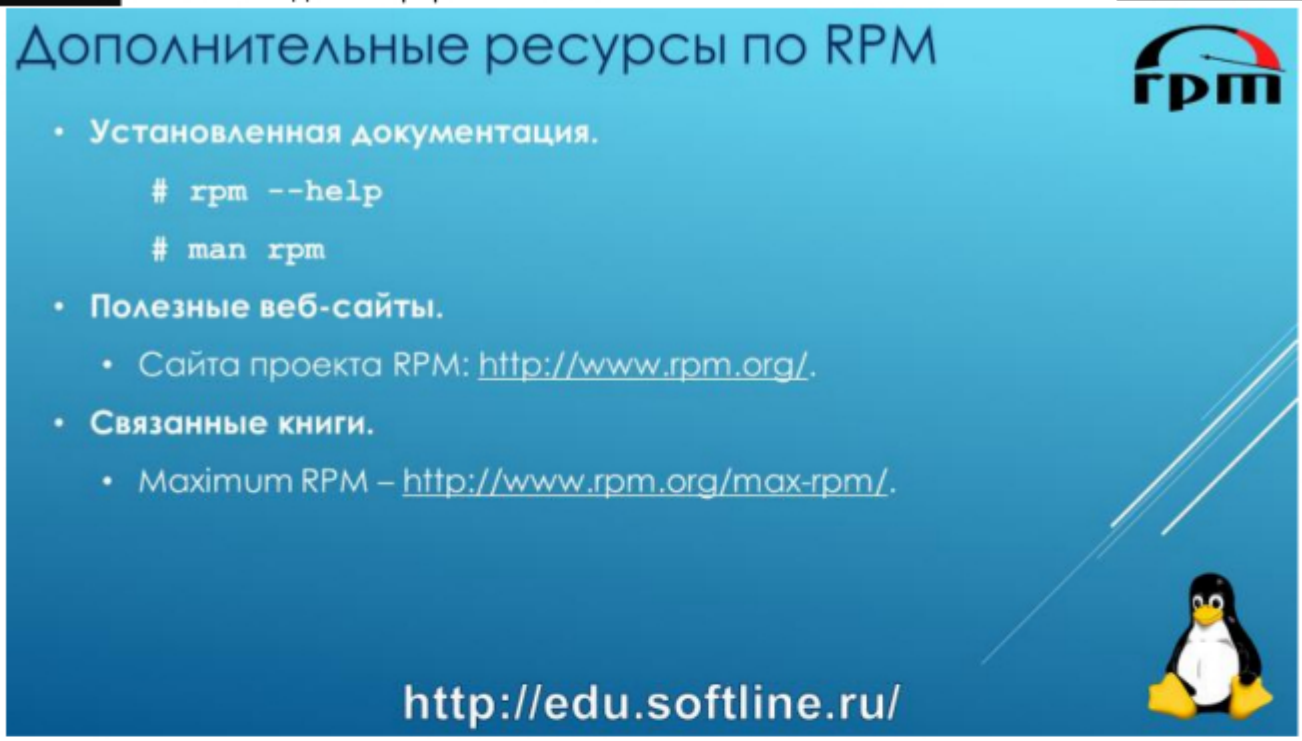
```
rpm -qi gpg-pubkey-db4 2a60e-37ea54 38
```

Проверка подписей на пакетах.

Для проверки подписи GnuPG файла RPM после импорта ключа GnuPG создателя пакета, используйте следующую команду (Замените *rpm-file* на имя файла RPM-пакета):

```
# rpm -K rpm-file
```

Если все пройдет успешно, следующее сообщение будет отображено: **md5 gpg OK**. Это значит, что подпись пакета проверена, он не изменен и безопасен для установки и использования.



RPM это экстремально комплексная утилита с множеством опций и методов для запросов, установок, обновлений и удалений пакетов. Используйте следующие ресурсы для изучения возможностей RPM.

Установленная документация:

- **`rpm --help`** – отображает быструю подсказку по параметрам RPM.
- **`man rpm`** – отображает страницу руководства man, содержащую более детальную информацию о параметрах RPM.

Полезные веб-сайты:

- Сайта проекта RPM: http://www.rpm.org/.

Связанные книги:

- **Maximum RPM** – http://www.rpm.org/max-rpm/.

Книга Maximum RPM, доступная для чтения в режиме онлайн, содержит полную информацию, начиная от использования RPM до создания своих собственных RPM-пакетов и программирования при помощи **`rpmlib`**.



Yum – это пакетный менеджер Red Hat, который может запрашивать информацию о доступных пакетах, загружать пакеты из репозитория, устанавливать и удалять их, обновлять всю систему до последней доступной версии. Yum выполняет автоматическое разрешение зависимостей, при установке, удалении или обновлении пакетов.

Yum может быть настроен на работу с дополнительными репозиториями или источниками пакетов, а также предоставляет множество плагинов, которые улучшают и расширяют его возможности. Yum во многом повторяет функционал RPM, дополнительно, также многие опции командной строки похожи друг на друга. Yum позволяет легко и быстро управлять пакетами на одной машине или на группе машин.



ПРИМЕЧАНИЕ:

Yum предоставляет безопасное управление пакетами за счет включения проверки подписей GPG (Gnu Privacy Guard, также известный как GnuPG) на пакетах подписанных при помощи GPG. Проверка подписей GPG может быть включена как на всех репозиториях так и на отдельных из них. Это обозначает, что вы доверяете RPM-пакету, скачанному из репозитория, убедившись в том, что он не был модифицирован.

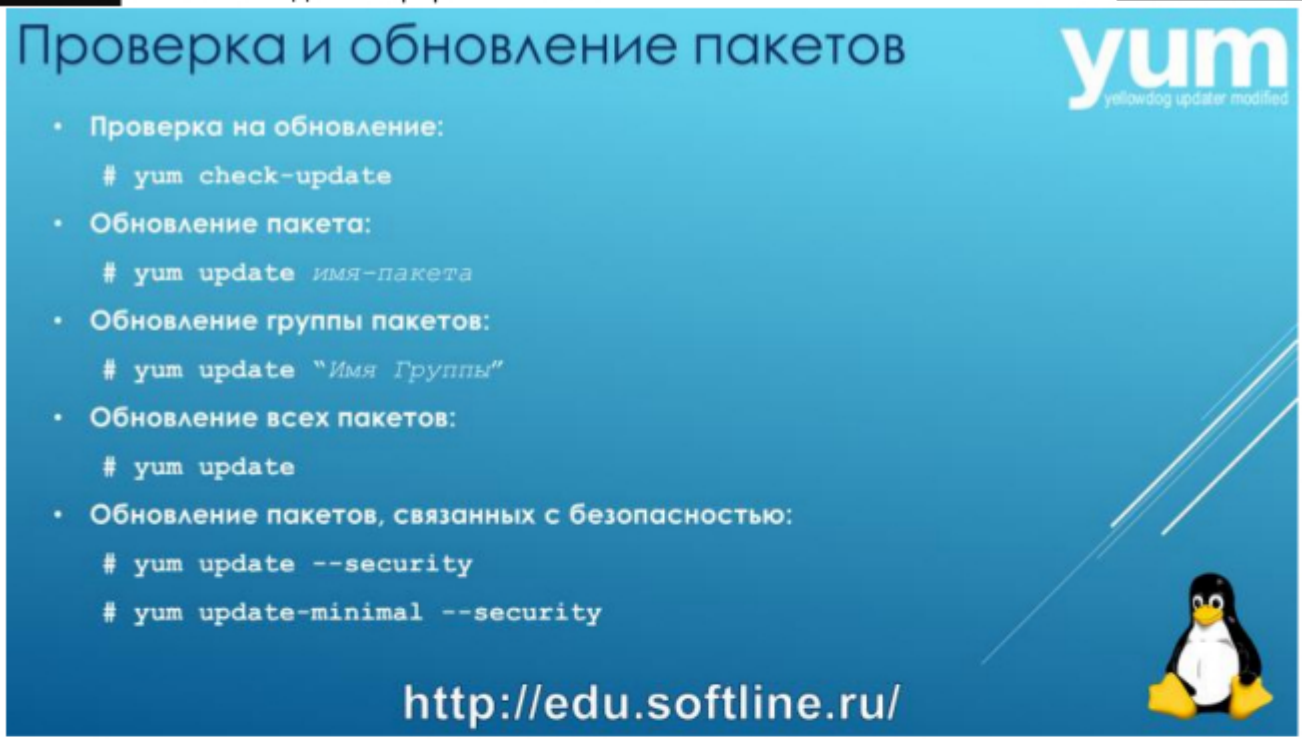
Yum также позволяет вам настроить ваш собственный репозиторий пакетов для загрузки и установки на прочих машинах. По возможности Yum использует параллельную загрузку нескольких пакетов и метаданных для увеличения скорости загрузки.

Изучение Yum крайне важно, так как это самый быстрый путь выполнения задач системного администрирования, а также он предоставляет значительно больше возможностей, чем графический инструмент управления пакетами PackageKit.



ПРИМЕЧАНИЕ:

Для установки, удаления или обновления пакетов на системе при помощи yum необходимо иметь привилегии супер-пользователя. Для получения необходимых прав можно воспользоваться **su** *или* **sudo**.



Yum позволяет проверить наличие доступных обновлений для вашей системы. Вы можете вывести список обновлений пакетов доступных для вашей системы и обновить их все, либо выборочно обновить отдельные пакеты.

Проверка на обновление.

При помощи команды **yum check-update** вы можете вывести список доступных обновлений пакетов для вашей системы:

```
[root@c7-server01 ~]# yum check-update
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.digitalhusky.com
 * epel: mirror.yandex.ru
 * extras: mirror.digitalhusky.com
 * updates: mirror.digitalhusky.com

ModemManager-glib.x86_64              1.1.0-8.git20130913.el7          base
NetworkManager.x86_64                 1:1.0.6-29.el7_2                 updates
NetworkManager-adsl.x86_64            1:1.0.6-29.el7_2                 updates
NetworkManager-bluetooth.x86_64       1:1.0.6-29.el7_2                 updates
NetworkManager-glib.x86_64            1:1.0.6-29.el7_2                 updates
NetworkManager-libnm.x86_64           1:1.0.6-29.el7_2                 updates
NetworkManager-team.x86_64            1:1.0.6-29.el7_2                 updates
NetworkManager-tui.x86_64             1:1.0.6-29.el7_2                 updates
NetworkManager-wifi.x86_64            1:1.0.6-29.el7_2                 updates
NetworkManager-wwan.x86_64            1:1.0.6-29.el7_2                 updates
abrt.x86_64                           2.1.11-36.el7.centos             base
abrt-addon-ccpp.x86_64                2.1.11-36.el7.centos             base
abrt-addon-kerneloops.x86_64          2.1.11-36.el7.centos             base
```

Первый доступный для обновления пакет – это **ModemManager-glib**. Строку вывода можно интерпретировать следующим образом:

- `ModemManager-glib` – имя пакета.
- `x86_64` – архитектура процессора, под которую был скомпилирован пакет.

- `1.1.0-8` – Версия обновленного пакета.
- `base` – репозиторий, из которого пакет будет загружаться.

Обновление пакетов.

Вы можете обновить один пакет, несколько пакетов или все пакеты сразу. Если у зависимых пакетов обновляемого пакета присутствуют обновления, они будут обновлены автоматически.

Обновление одного пакета.

Для обновления оного пакета, выполните следующую команду **`yum update`** ***`имя_пакета`*** от имени пользователя root:

Пример обновления пакета **`rpm`**:

```
[root@c7-server01 ~]# yum update rpm
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.digitalhusky.com
 * epel: mirror.yandex.ru
 * extras: mirror.digitalhusky.com
 * updates: mirror.digitalhusky.com
Resolving Dependencies
--> Running transaction check
---> Package rpm.x86_64 0:4.11.1-25.el7 will be updated
--> Processing Dependency: rpm = 4.11.1-25.el7 for package: rpm-python-4.11.1-
25.el7.x86_64
--> Processing Dependency: rpm = 4.11.1-25.el7 for package: rpm-libs-4.11.1-25.el7.x86_64
---> Package rpm.x86_64 0:4.11.3-17.el7 will be an update
…
---> Package rpm-build-libs.x86_64 0:4.11.3-17.el7 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                Arch           Version                Repository    Size
================================================================================
Updating:
 rpm                x86_64         4.11.3-17.el7          base         1.2 M
Updating for dependencies:
 rpm-build-libs     x86_64         4.11.3-17.el7          base         103 k
 rpm-libs           x86_64         4.11.3-17.el7          base         272 k
 rpm-python         x86_64         4.11.3-17.el7          base          79 k

Transaction Summary
================================================================================
Upgrade  1 Package (+3 Dependent packages)

Total download size: 1.6 M
Is this ok [y/d/N]:
```

Вывод содержит следующие элементы:

1. **`Loaded plugins: fastestmirror, langpacks`** – yum всегда информирует о том, какие плагины установлены и включены.
2. **`Resolving Dependencies`** – Yum использует транзакционную проверку для разрешения зависимостей.

3. Yum представляет информацию об обновлении и затем запрашивает подтверждение о начале процесса обновления, по умолчанию yum запускается в интерактивном режиме. Если вы уже знаете о том, что будет делать команда yum, вы можете воспользоваться опциональным ключом **-y**, чтобы автоматически отвечать **yes** на все запросы, которые выдаст yum (в данном случае yum запускается не интерактивно). Однако, вы должны быть всегда в курсе о том какие изменения yum собирается внести в систему, это позволит проще предотвращать возможные неисправности. Вы также можете выбрать загрузку пакетов без установки, для этого необходимо выбрать опцию **d** в строке запроса загрузки. Это запустит фоновый процесс загрузку выбранного пакета.

Аналогичным образом вы может обновить группу пакетов. Для этого воспользуйтесь командой **yum group update *Имя Группы***, от имени пользователя root:

```
[root@c7-server01~]# yum group update Development Tools
Loaded plugins: fastestmirror, langpacks
There is no installed groups file.
Maybe run: yum groups mark convert (see man yum)
Loading mirror speeds from cached hostfile
 * base: mirror.digitalhusky.com
 * epel: mirror.yandex.ru
 * extras: mirror.digitalhusky.com
 * updates: mirror.digitalhusky.com
Warning: group Tools does not exist.
Resolving Dependencies
--> Running transaction check
---> Package autoconf.noarch 0:2.69-11.el7 will be installed
…
```

Yum также поддерживает команду **upgrade**, которая эквивалентна команде **update** с включенной опцией настройки **obsolete**. По умолчанию опция **obsolete** включена в **/etc/yum.conf**, что делает эти две команды эквивалентными.



ПРИМЕЧАНИЕ:
Подробная информация о группах пакетов и опциях настройки представлена дальше в модуле.

Обновление всех пакетов и их зависимостей.

Для обновления всех пакетов и их зависимостей введите команду **yum update** без каких-либо опций.

```
[root@c7-server01~]# yum update
```

Обновление пакетов, связанных с безопасностью.

Обнаружение пакетов, имеющих обновления безопасности и последующее обновление этих пакетов максимально быстро крайне важно. Для этих целей yum предоставляет специальный плагин. Плагин **security** расширяет возможности команды **yum** при помощи удобного набора команд, подкоманд и опций, связанных с безопасностью.



ПРИМЕЧАНИЕ:
Подробная информация о плагинах представлена дальше в модуле.

Для обновления пакета, до последней версии, если у него было обновление безопасности, используйте следующую команду:

```
[root@c7-server01~]# yum update --security
```

Для обновления пакета, до версии с обновлением безопасности, используйте следующую команду:

```
[root@c7-server01~]# yum update-minimal --security
```

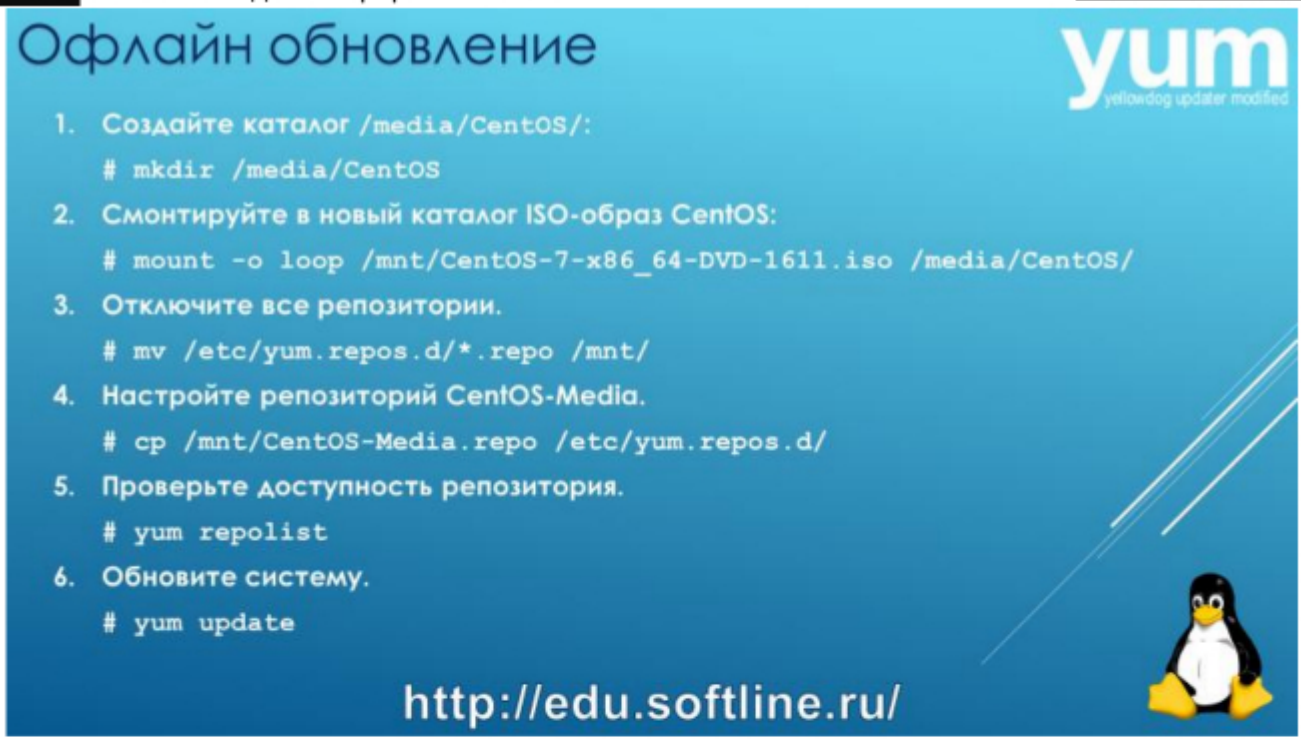
Рассмотрим пример:

- Пакет ***kernel-4.1.6-1*** установлен на системе.
- Пакет ***kernel-4.1.6-2*** содержит обновление безопасности.
- Пакет ***kernel-4.1.6-3*** содержит исправление ошибки.

Команда **yum update-minimal --security**, обновит пакет до ***kernel-4.1.6-2***, а команда **yum update --security** обновит пакет до ***kernel-4.1.6-3***.

Предотвращение изменений конфигурационных файлов.

Вы неизбежно вносите изменения в конфигурационные файлы, установленные вместе с пакетами в системе. RPM, который использует YUM для внесения изменений в систему, предоставляет механизм проверки и сохранения целостности изменений.



Для систем, которые не имеют доступа к сети интернет или иным источникам репозиториев Yum, можно использовать команду yum update с установочным образом CentOS. Это простой и быстрый способ обновить систему до последней минорной версии. Последующие шаги демонстрирую процесс обновления:

1. Создайте целевой каталог для монтирования образа ISO (каталог не создается автоматически).

```
[root@c7-server01~]# mkdir /media/iso
```

2. Смонтируйте установочный образ ISO в созданный каталог:

```
[root@c7-server01 ~]# mount -o loop ~/CentOS-7-1511.iso /media/iso
```

Опция **`-o loop`** необходима для монтирования файла в качестве блочного устройства.

3. Скопируйте файл **`media.repo`** из смонтированного дистрибутива в каталог **`/etc/yum.repos.d/`**.

```
[root@c7-server01~]# cp /media/iso/media.repo /etc/repos.d/c7.repo
```

Для корректной работы, конфигурационные файлы в каталоге **`/etc/yum.repos.d/`** должны обладать расширением **`.repo`**.

4. Отредактируйте новый конфигурационный файл, указав на смонтированный установочный образ ISO. Добавьте следующую строку в файл **`/etc/repos.d/c7.repo`**:

```
baseurl=file://media/iso
```

5. Проверьте доступность нового репозитория Yum:

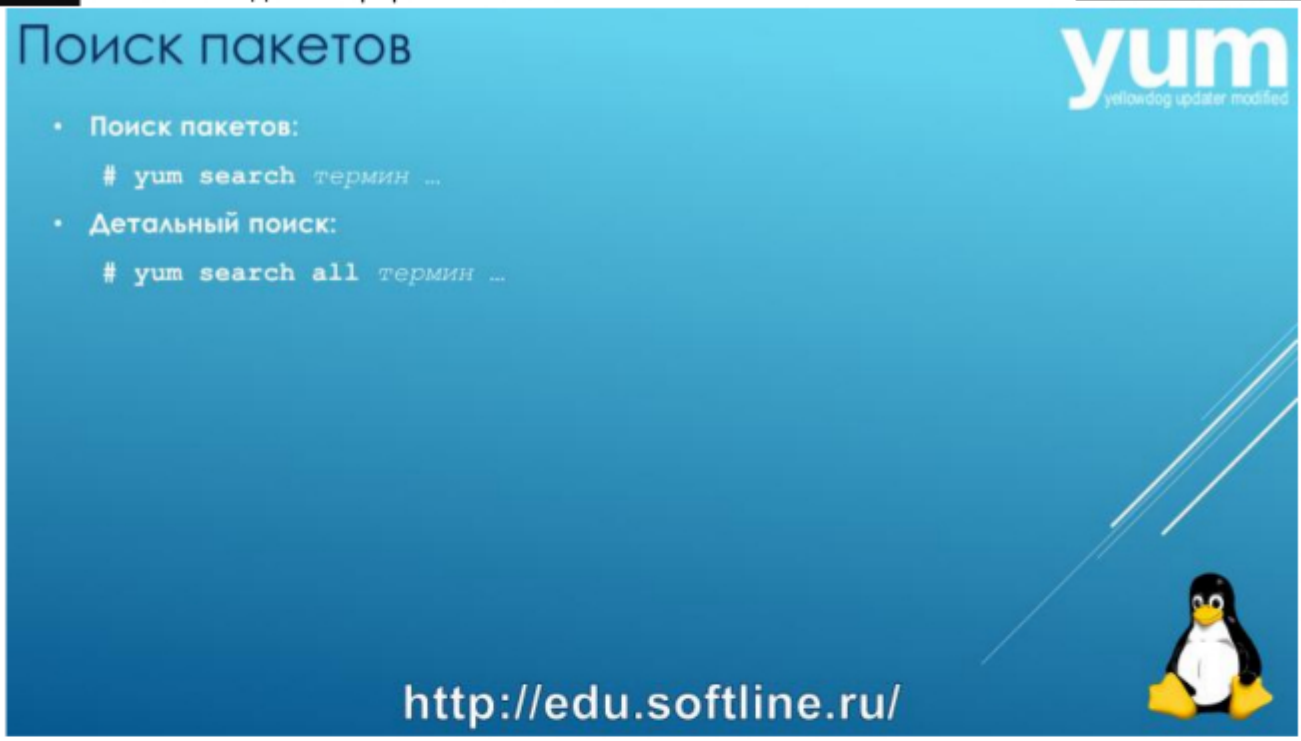
```
[root@c7-server01~]# yum repolist
```

6. **Обновите систему при помощи нового репозитория Yum:**

```
[root@c7-server01~]# yum update
```

7. Если вы более не планируете использовать репозиторий установочного образа ISO, можно отмонтировать образ и удалить каталог монтирования и конфигурационный файл:

```
[root@c7-server01 ~]# umount /media/iso
[root@c7-server01 ~]# rmdir /media/iso
[root@c7-server01 ~]# rm /etc/repos.d/c72.repo
```



Вы можете искать все имена, описания и сводную информацию об RPM-пакетах при помощи следующей команды:

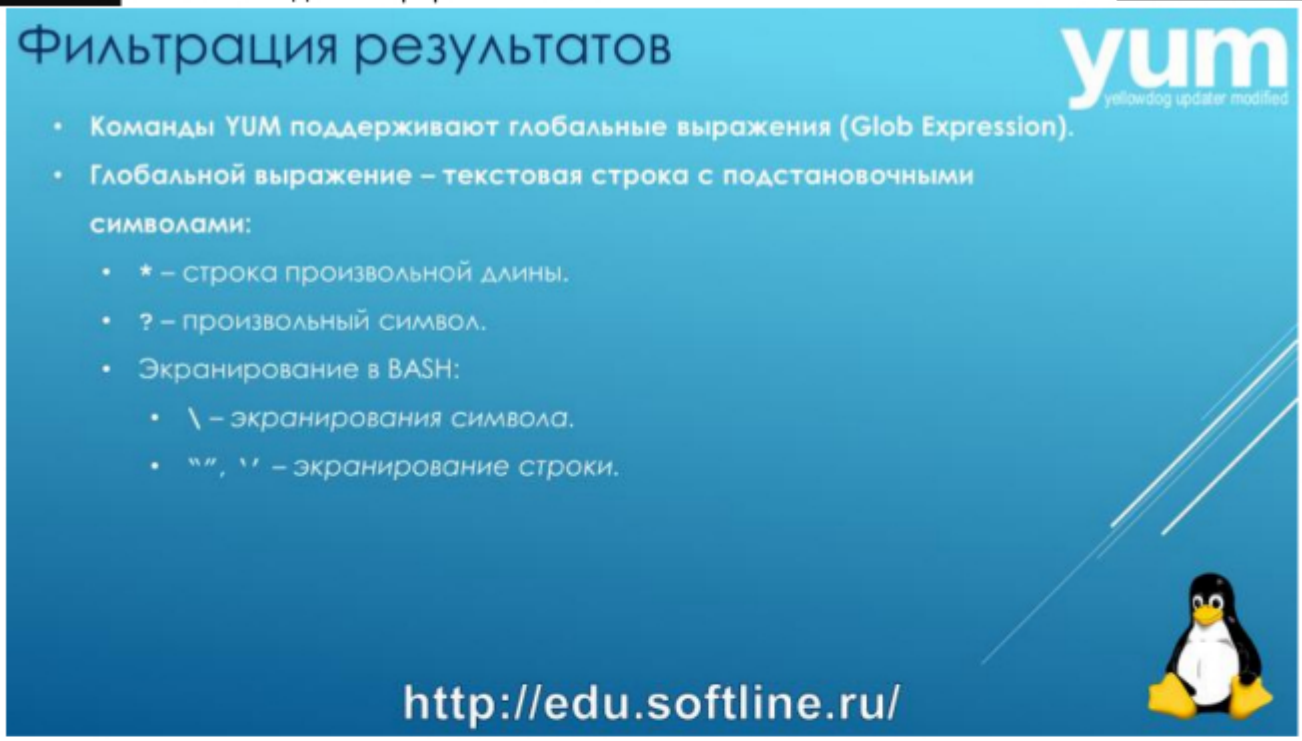
yum search *термин …*

Этак команда отображает все совпадения с искомым значением или значениями.

Пример поиска пакетов, которые соответствуют терминам «**rpm**» или «**tool**»:

```
[root@c7-server01 ~]# yum search rpm tool
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.digitalhusky.com
 * epel: mirror.yandex.ru
 * extras: mirror.digitalhusky.com
 * updates: mirror.digitalhusky.com
========================== N/S matched: rpm, tool ==========================
cabal-rpm.x86_64 : RPM packaging tool for Haskell Cabal-based packages
rpmconf.noarch : Tool to handle rpmnew and rpmsave files
rpmdevtools.noarch : RPM Development Tools
rpmlint.noarch : Tool for checking common errors in RPM packages
rpmreaper.x86_64 : A tool for removing packages from system
rpmrebuild.noarch : A tool to build rpm file from rpm database
tito.noarch : A tool for managing rpm based git projects

  Full name and summary matches only, use "search all" for everything.
```

Команда **yum search** полезна для поиска пакетов, имя которых вам не известно, но вы знаете связанный термин. Учтите, что по умолчанию **yum search** возвращает совпадения в имени пакета или сводке, что делает поиск более быстрым. Используйте команд **yum search all** для исчерпывающего и более длительного поиска.



Все команды Yum, возвращающие список значений поддерживают фильтрацию результатов при помощи одного или нескольких глобальных выражений (Glob Expression) в качестве аргументов. Глобальные выражения – это обычные строки символов, которые содержат один или более символов подстановки ***** (обозначает строку произвольной длины) и **?** (обозначает один произвольный символ).

Будьте осторожны при передаче подстановочных символов, так как в первую очередь их обрабатывает Bash. Для передачи глобальных выражений используйте следующие методы:

- Для передачи *****, поставьте перед ним обратный слеш (\).
- Заключите все глобальное выражение в двойные или одинарные кавычки.

Далее в примерах демонстрируется оба этих метода.



Для вывода списка всех установленных и доступных пакетов используйте следующую команду:

```
[root@c7-server01 ~]# yum list all
```

Для вывода установленных и доступных пакетов, которые соответствуют введенному глобальному выражению, используйте следующую команду:

yum list *глобальное_выражение…*

Пример использования глобального выражения, указывающего на все пакеты, начинающиеся на **rpm-b**:

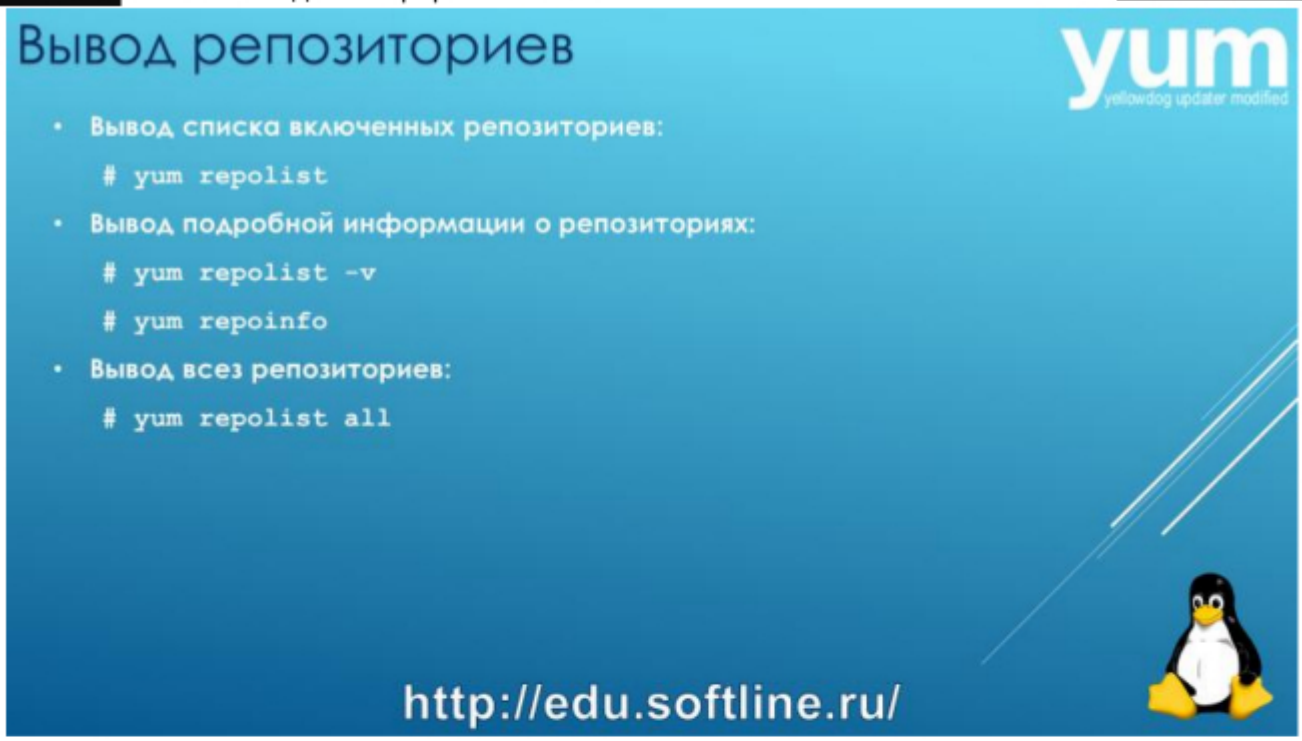
```
[root@c7-server01 ~]# yum list rpm-b\*
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.digitalhusky.com
 * epel: mirror.yandex.ru
 * extras: mirror.digitalhusky.com
 * updates: mirror.digitalhusky.com
Installed Packages
rpm-build-libs.x86_64                4.11.1-25.el7                @base
Available Packages
rpm-build.x86_64                     4.11.3-17.el7                base
rpm-build-libs.i686                  4.11.3-17.el7                base
rpm-build-libs.x86_64                4.11.3-17.el7                base
```

Для вывода всех установленных на системе пакетов используйте ключевое слово **installed**. Соответствующий список пакетов будет возвращен:

```
[root@c7-server01 ~]# yum list installed rpm-b\*
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.digitalhusky.com
 * epel: mirror.yandex.ru
 * extras: mirror.digitalhusky.com
 * updates: mirror.digitalhusky.com
Installed Packages
rpm-build-libs.x86_64                4.11.1-25.el7                @base
```

Для вывода списка всех пакетов, доступных для установки из репозиториев, используйте ключевое слово **availbale**:

```
[root@c7-server01 ~]# yum list available rpm-b\*
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.digitalhusky.com
 * epel: mirror.yandex.ru
 * extras: mirror.digitalhusky.com
 * updates: mirror.digitalhusky.com
Available Packages
rpm-build.x86_64                          4.11.3-17.el7                   base
rpm-build-libs.i686                       4.11.3-17.el7                   base
rpm-build-libs.x86_64                     4.11.3-17.el7                   base
```



Для отображения идентификатора, имени и числа пакетов во всех включенных репозиториях используйте следующую команду:

```
[root@c7-server01 ~]# yum repolist
```

Для отображения более подробной информации о репозиториях используйте опцию **–v**. При использовании этой опции, информация будет включать имя файла, общий размер, дату последнего обновления и базовый URL для каждого репозитория в списке. В качестве альтернативы, можно использовать команду repoinfo, которая имеет аналогичный вывод:

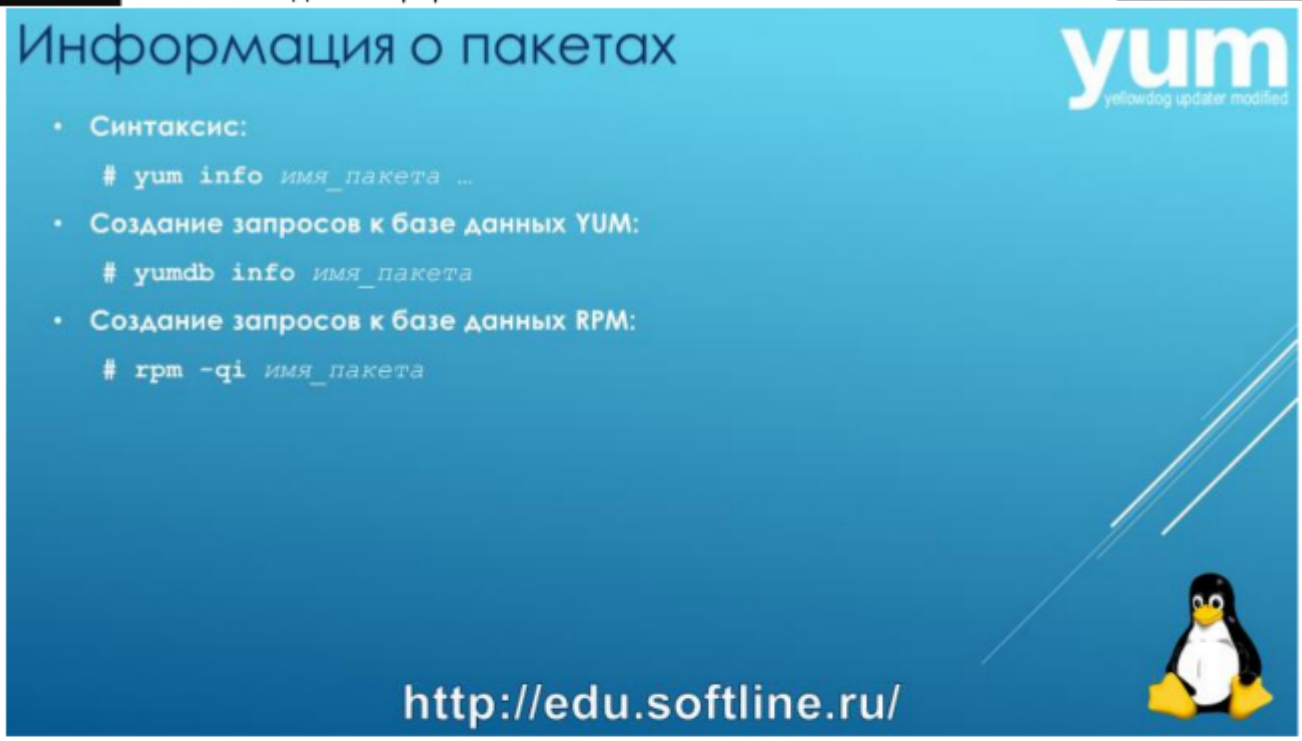
```
[root@c7-server01 ~]# yum repolist -v
[root@c7-server01 ~]# yum repoinfo
```

Для вывода списка всех и включенных, и отключенных репозиториев, используйте следующую команду:

```
[root@c7-server01 ~]# yum repolist all
```

Колонка со статусом (Status) будет добавлена в список вывода, отображая какие репозитории включены.

Передав **disabled**, в качестве первого аргумента, вы сократите вывод до отключенных репозиториев. Можно также передавать идентификатор или имя репозитория или глобальное выражение в качестве аргумента. Обратите внимание, что используется полное соответствие между именем и идентификатором введенного аргумента и репозиторий будет отображен, даже если он не соответствует фильтру **enabled** или **disabled**.



Для отображения информации об одном или нескольких пакетах, используйте следующую команду (глобальные выражения также поддерживаются).

yum info *имя_пакета…*

Пример отображения информации о пакете rpm:

```
[root@c7-server01 ~]# yum info rpm
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.digitalhusky.com
 * epel: mirror.yandex.ru
 * extras: mirror.digitalhusky.com
 * updates: mirror.digitalhusky.com
Installed Packages
Name        : rpm
Arch        : x86_64
Version     : 4.11.1
Release     : 25.el7
Size        : 2.5 M
Repo        : installed
From repo   : base
Summary     : The RPM package management system
URL         : http://www.rpm.org/
License     : GPLv2+
Description : The RPM Package Manager (RPM) is a powerful command line driven
            : package management system capable of installing, uninstalling,
            : verifying, querying, and updating software packages. Each software
            : package consists of an archive of files along with information
            : about the package like its version, a description, etc.

Available Packages
Name        : rpm
Arch        : x86_64
Version     : 4.11.3
Release     : 17.el7
Size        : 1.2 M
Repo        : base/7/x86_64
Summary     : The RPM package management system
URL         : http://www.rpm.org/
```

```
License     : GPLv2+
Description : The RPM Package Manager (RPM) is a powerful command line driven
            : package management system capable of installing, uninstalling,
            : verifying, querying, and updating software packages. Each software
            : package consists of an archive of files along with information
            : about the package like its version, a description, etc.
```

Команда **yum info** *имя_пакета* похожа на команду **rpm -q -info** *имя_пакета*, но предоставляет дополнительную информацию: идентификатор репозитория Yum, в котором был найден RPM-пакет (строка **From repo:** в выводе).

Использование yumdb.

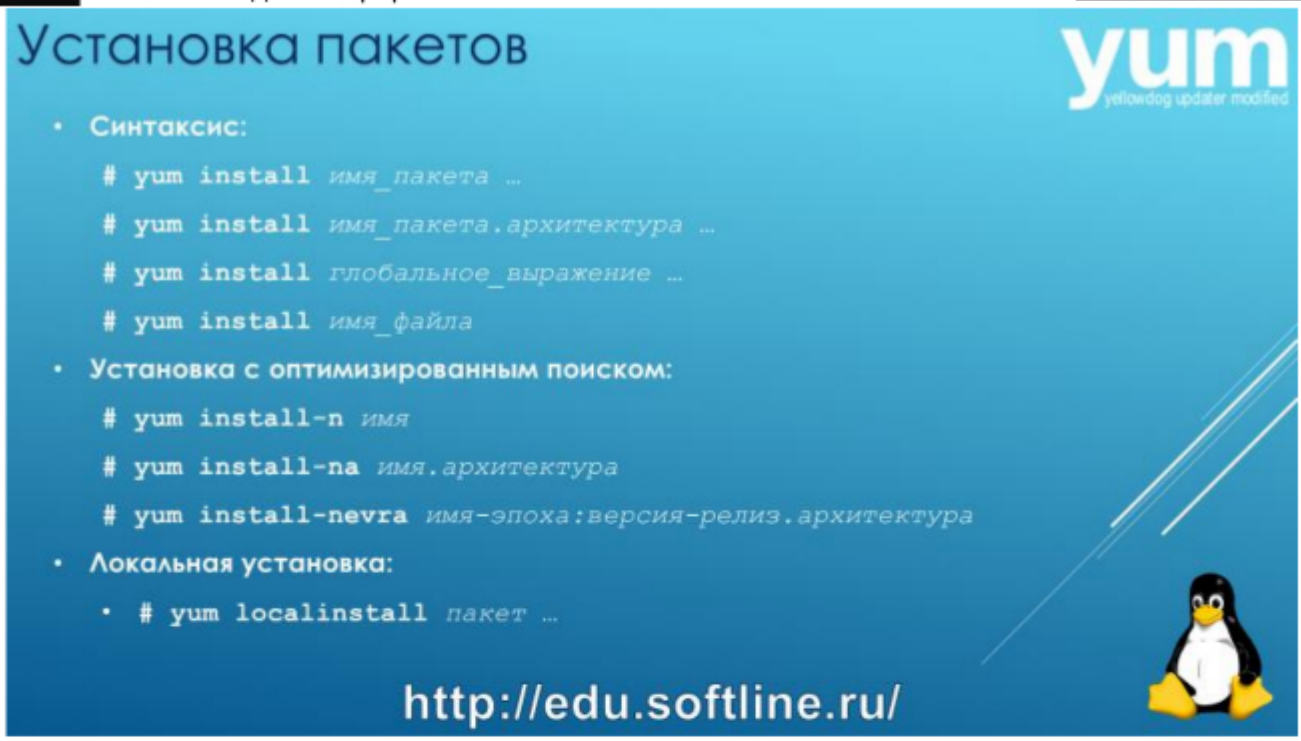
Имеется возможность создавать запросы к базе данных Yum, в качестве альтернативного и полезного источника информации о пакетах при помощи следующей команды:

yumdb info *имя_пакета*

Эта команда предоставляет дополнительную информацию о пакете, в том числе контрольную сумму пакета (и алгоритм, использованный для вычисления, такой как SHA-256), команду, которая была использована для установки пакета, цель установки пакета в системе (где **user** обозначает установку пользователем, а **dep** обозначает установку в качестве зависимости).

```
[root@c7-server01 ~]# yumdb info rpm
Loaded plugins: fastestmirror, langpacks
rpm-4.11.1-25.el7.x86_64
     changed_by = 0
     checksum_data = 43671fd5e1073e3589058a24efbbab9139384893da8d7f2e7502d48181f53e9d
     checksum_type = sha256
     command_line = update
     from_repo = base
     from_repo_revision = 1427842153
     from_repo_timestamp = 1427842246
     installed_by = 4294967295
     origin_url  =  http://mirror.yandex.ru/centos/7.1.1503/os/x86_64/Packages/rpm-4.11.1-
25.el7.x86_64.rpm
     reason = user
     releasever = 7
     var_uuid = 777cc08f-59ee-4df5-b1b5-00e24477a0af
```

Для получения дополнительной информации по команде **yumdb**, обратитесь к странице руководства **yumdb(8)**.



Для установки отдельного пакета и всех его неустановленных зависимостей, введите следующую команду от имени пользователя root:

yum install *имя_пакета*

Вы можете установить несколько пакетов, добавляя их имена в качестве аргументов:

yum install *имя_пакета имя_пакета …*

Если вы устанавливаете пакет на системе с несколькими наборами библиотек (Multilib), таких как машины с архитектурой AMD64 или Intel64 вы можете указать архитектуру пакета (если она доступна в репозитории), добавив *.архитектура* к имени пакета.

yum insyall *имя_пакета.архитектура*

Пример установки Wireshark с архитектурой i686 в CentOS 7 (x64):

```
[root@c7-server01 ~]# yum install wireshark.i686
```

Также можно использовать глобальные выражения (Glob Expressions) при установке нескольких идентичных пакетов

yum install *глобальное_выражение …*

Пример установки всех пакетов, имя которых начинается на **openssl**:

```
[root@c7-server01 ~]# yum install openssl\*
```

В дополнение к имени пакета и глобальным выражениям вы можете передавать имя файла команде **yum install**. Если вы знаете имя бинарного файла, который хотите установить, но не знаете имя пакета, вы можете указать абсолютный путь для команды **yum install**:

```
[root@c7-server01 ~]# yum install /usr/sbin/named
```

В таком случае **yum** ищет в своем списке пакет, который содержит файл **/usr/sbin/named**, при нахождении запрашивает у вас необходимость установки.

Как можно видеть из примера выше, команда **yum install** не требует прямого указания аргументов. Он может обрабатывать различные форматы имен пакетов и глобальных выражений, которые упрощают установку для пользователей. С другой стороны, это требует некоторого времени, чтобы **yum** распознал ввод корректно, особенно если вы указали большое количество пакетов. Для оптимизации поиска пакетов, вы можете использовать следующие команды, для прямого указания, как распознавать аргументы:

```
yum install-n имя
yum install-na имя.архитектура
yum install-nevra имя-эпоха:версия-релиз.архитектура
```

При помощи команды **install-n** указывается на прямое соответствие имени пакета. Команда `install-na` указывает, что последующие аргументы содержат имя и архитектуру разделенные символом точки. Команды **install-nevra**, **yum** указывает, что последующие аргументы имеют следующую форму: *имя-эпоха:версия-релиз.архитектура*. Аналогично вы можете использовать команды **yum rmove-n**, **yum rmove-na** и **yum rmove-nevra** для поиска пакетов на удаление.



ПРИМЕЧАНИЕ:

Если вы хотите установить пакет, который содержит бинарный файл ***named****, но не знаете в каком каталоге* `bin` *или* `sbin` *он должен находиться, используйте команду yum provides с глобальным выражением:*

```
[root@MAINSERVER ~]# yum provides *bin/named
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.digitalhusky.com
 * epel: mirror.yandex.ru
 * extras: mirror.digitalhusky.com
 * updates: mirror.digitalhusky.com
32:bind-9.9.4-29.el7.x86_64 : The Berkeley Internet Name Domain (BIND) DNS
                            : (Domain Name System) server
Repo        : base
Matched from:
Filename    : /usr/sbin/named
…
```

По аналогии можно использовать команду следующего вида:

yum provides "*/имя_файла"

Следующий пример предоставляет обзор установки Apache HTTP Server при помощи yum. Для загрузки и установки последней версии **httpd**, используйте следующую команду:

```
[root@c7-server01 ~]# yum install httpd
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.digitalhusky.com
 * epel: mirror.yandex.ru
 * extras: mirror.digitalhusky.com
 * updates: mirror.digitalhusky.com
Resolving Dependencies
--> Running transaction check
---> Package httpd.x86_64 0:2.4.6-31.el7.centos will be updated
---> Package httpd.x86_64 0:2.4.6-40.el7.centos.1 will be an update
--> Processing Dependency: httpd-tools = 2.4.6-40.el7.centos.1 for package: httpd-
2.4.6-40.el7.centos.1.x86_64
--> Running transaction check
```

```
---> Package httpd-tools.x86_64 0:2.4.6-31.el7.centos will be updated
---> Package httpd-tools.x86_64 0:2.4.6-40.el7.centos.1 will be an update
--> Finished Dependency Resolution

Dependencies Resolved
```

После выполнения выше указанной команды, **yum** загружает необходимые плагины и выполняет транзакционную проверку. В данном примере, **httpd** уже установлен. Так как установленный пакет старше последнего доступного пакета, он будет обновлен. Тоже самое касается и пакета **httpd-tools**. Затем отображается результат транзакции:

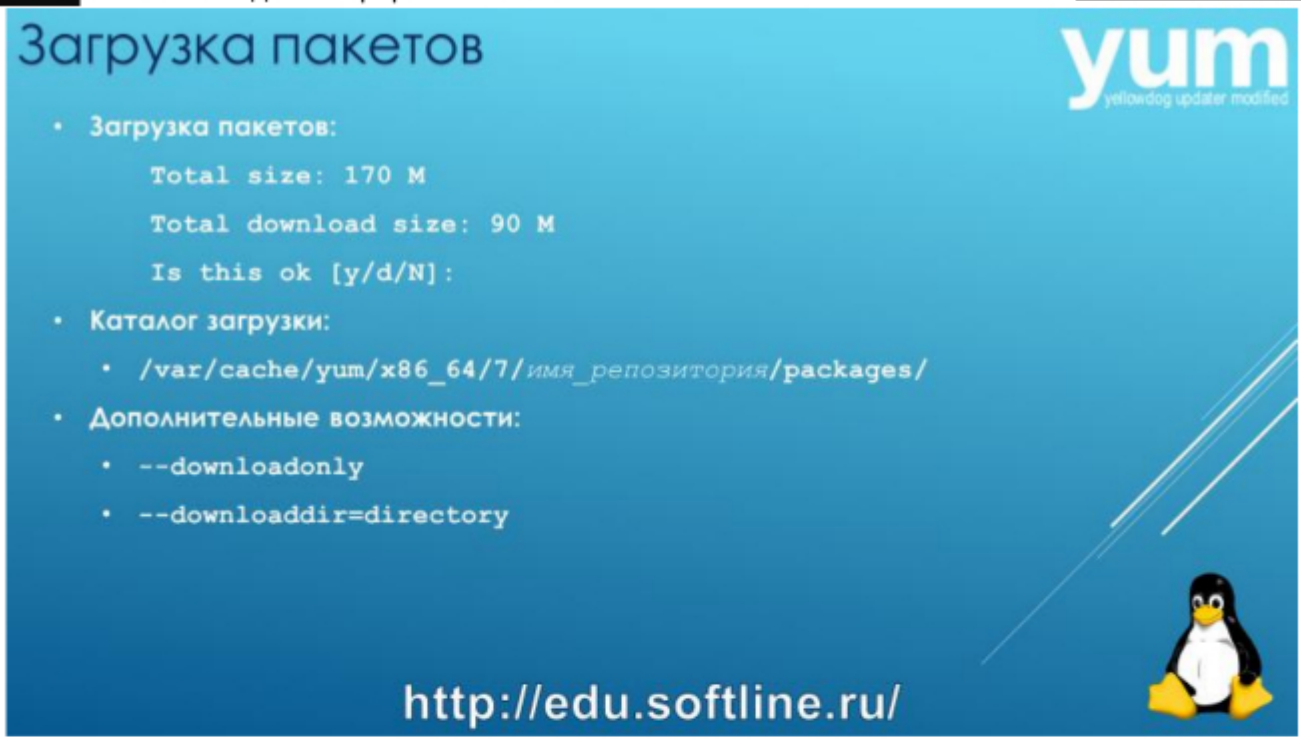
```
================================================================================
 Package            Arch         Version                      Repository     Size
================================================================================
Updating:
 httpd              x86_64       2.4.6-40.el7.centos.1        updates       2.7 M
Updating for dependencies:
 httpd-tools        x86_64       2.4.6-40.el7.centos.1        updates        82 k

Transaction Summary
================================================================================
Upgrade  1 Package (+1 Dependent package)

Total download size: 2.8 M
Is this ok [y/d/N]:
```

На этом шаге yum запрашивает у вас подтверждение установки. Вне зависимости от опций **y** (да) и **N** (нет), вы можете выбрать **d** (только загрузка) для загрузки, но не установки пакетов. Если вы выберите **y**, процесс установки продолжится и выдаст следующие сообщения:

```
Downloading packages:
Delta RPMs disabled because /usr/bin/applydeltarpm not installed.
(1/2): httpd-tools-2.4.6-40.el7.centos.1.x86_64.rpm        |  82 kB   00:00
(2/2): httpd-2.4.6-40.el7.centos.1.x86_64.rpm              | 2.7 MB   00:02
--------------------------------------------------------------------------------
Total                                          1.1 MB/s | 2.8 MB  00:02
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Updating   : httpd-tools-2.4.6-40.el7.centos.1.x86_64                    1/4
  Updating   : httpd-2.4.6-40.el7.centos.1.x86_64                          2/4
  Cleanup    : httpd-2.4.6-31.el7.centos.x86_64                            3/4
  Cleanup    : httpd-tools-2.4.6-31.el7.centos.x86_64                      4/4
  Verifying  : httpd-2.4.6-40.el7.centos.1.x86_64                          1/4
  Verifying  : httpd-tools-2.4.6-40.el7.centos.1.x86_64                    2/4
  Verifying  : httpd-tools-2.4.6-31.el7.centos.x86_64                      3/4
  Verifying  : httpd-2.4.6-31.el7.centos.x86_64                            4/4

Updated:
  httpd.x86_64 0:2.4.6-40.el7.centos.1

Dependency Updated:
  httpd-tools.x86_64 0:2.4.6-40.el7.centos.1

Complete!
```

Для установки предварительно загруженных пакетов из локального каталога на вашей системе, используйте следующую команду:

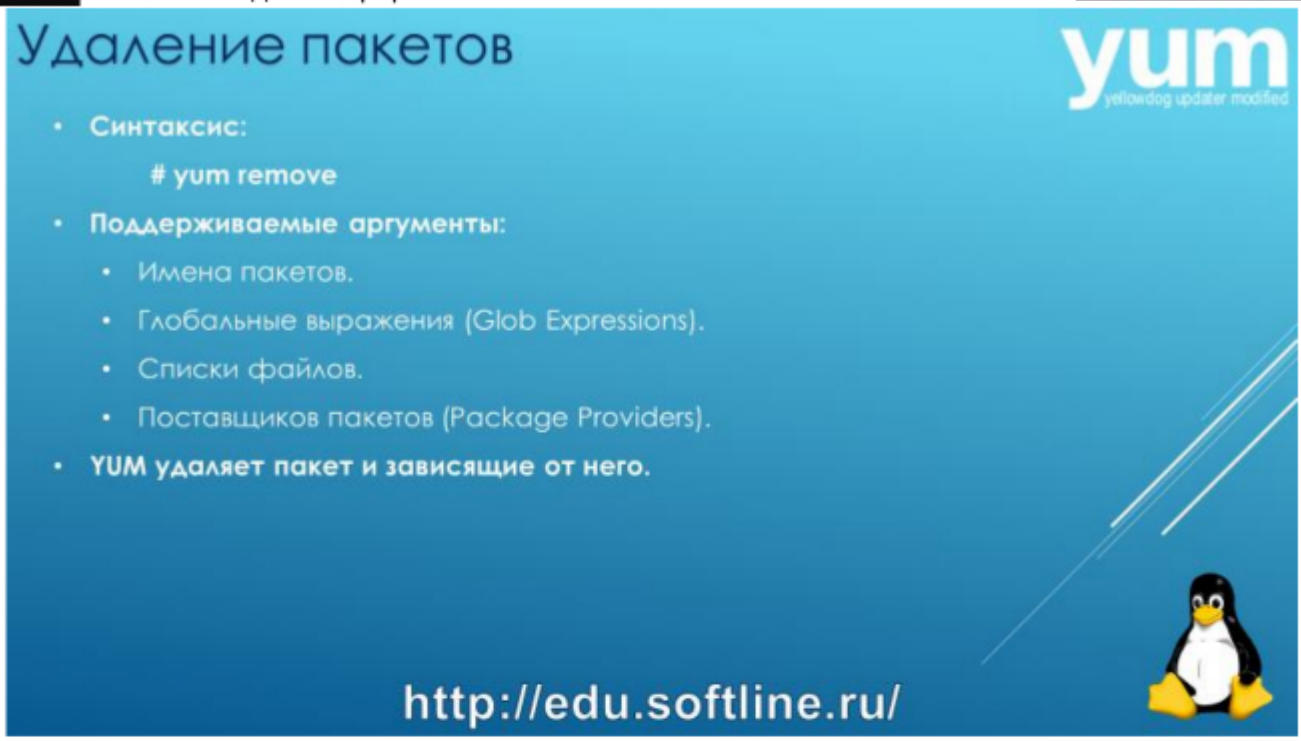
yum localinstall *путь*



Как было показано в предыдущем примере, в процессе установки пакетов у вас запрашивается подтверждение установки:

```
Total download size: 2.8 M
Is this ok [y/d/N]:
```

При помощи опции **d**, yum скачивает пакеты без установки. Вы можете позже установить эти пакеты в режиме офлайн, при помощи команды **yum localinstall** или вы можете распространить их между другими устройствами. Загруженные пакеты сохраняются в одном из подкаталогов каталога кэширования, по умолчанию: **/var/cache/yum/***архитектура***/***релиз***/packages**. Загрузка может выполняться в фоновом режиме, таким образом, оставляя возможность параллельно использовать yum для других операций.



Подобно установке пакетов, yum позволяет их удалять. Для удаления пакета, также, как и зависящих от него, выполните следующую команду от имени пользователя root:

yum remove *имя_пакета…*

Пример удаления пакета **proftpd**:

```
[root@c7-server01 ~]# yum remove proftpd
Loaded plugins: fastestmirror, langpacks
Resolving Dependencies
--> Running transaction check
---> Package proftpd.x86_64 0:1.3.5b-1.el7 will be erased
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package          Arch            Version                  Repository      Size
================================================================================
Removing:
 proftpd          x86_64          1.3.5b-1.el7             @epel          9.7 M

Transaction Summary
================================================================================
Remove  1 Package

Installed size: 9.7 M
Is this ok [y/N]: y
Downloading packages:
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Erasing    : proftpd-1.3.5b-1.el7.x86_64                                  1/1
warning: /etc/ftpusers saved as /etc/ftpusers.rpmsave
  Verifying  : proftpd-1.3.5b-1.el7.x86_64                                  1/1

Removed:
  proftpd.x86_64 0:1.3.5b-1.el7
```

```
Complete!
```

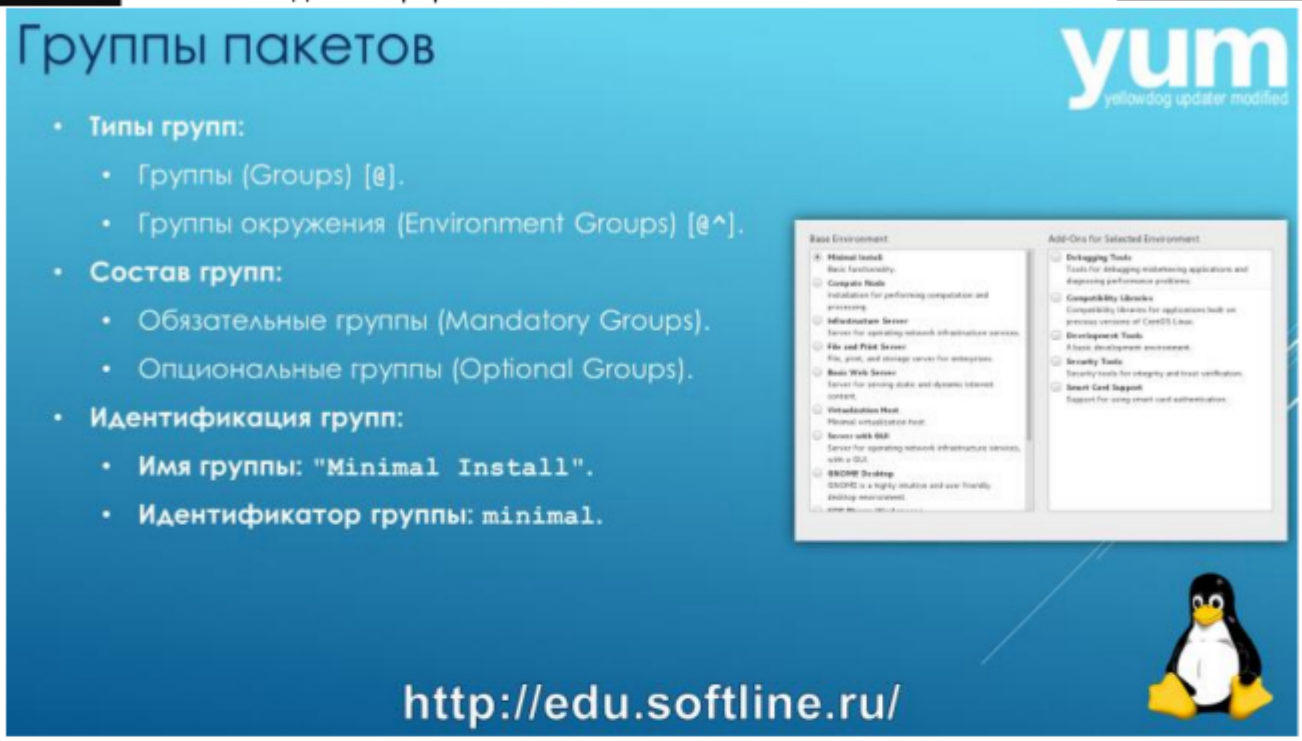
Подобно тому, как это работает с командой yum install, yum remove может принимать следующие аргументы:

- Имена пакетов.
- Глобальные выражения (Glob Expressions).
- Списки файлов.
- Поставщиков пакетов (Package Providers).

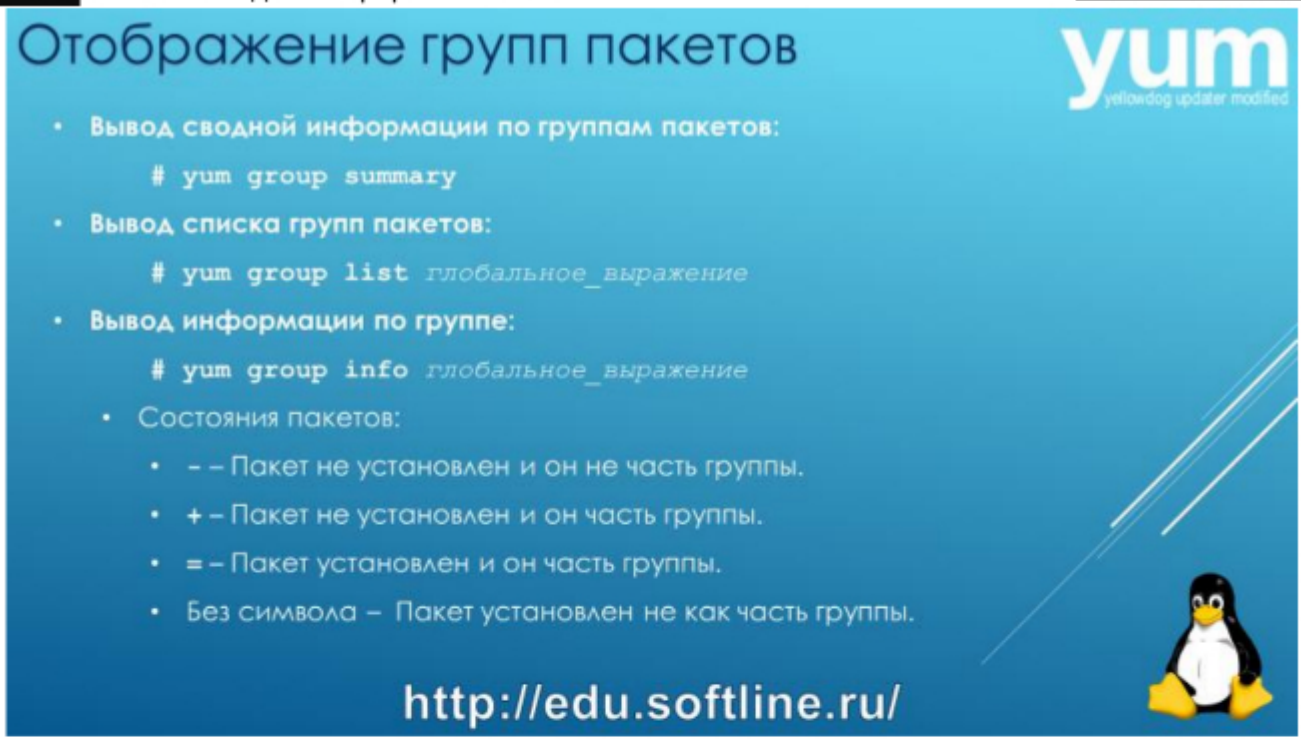


ПРИМЕЧАНИЕ:

Yum не может удалить пакет, без удаления зависящего от него пакета. Это тот тип операций, который может быть выполнен только при помощи RPM, такая возможность не рекомендуется к использованию и потенциально может повлечь за собой некорректную работу вашей системы или аварию.



Группа пакетов (Package Group) – это коллекция пакетов, которые имеют одинаковое назначение, например, **`System Tools`** или **`Sound and Video`**. Установка группы пакетов, выстраивает в очередь набор зависимых пакетов и значительно экономит время. Команда yum groups – это команда верхнего уровня, позволяющая выполнять все возможные операции над группами пакетов в yum.



Опция **summary** используется для просмотра количества установленных групп, доступных групп, доступных групп окружения, а также установленных и доступных языковых групп:

```
[root@c7-server01 ~]# yum group summary
Loaded plugins: fastestmirror, langpacks
There is no installed groups file.
Maybe run: yum groups mark convert (see man yum)
Loading mirror speeds from cached hostfile
 * base: mirror.digitalhusky.com
 * epel: mirror.yandex.ru
 * extras: mirror.digitalhusky.com
 * updates: mirror.digitalhusky.com
Available environment groups: 11
Available Groups: 38
Done
```

Для отображения всех групп пакетов из репозиториев yum используйте опцию **list**. Также можно фильтровать вывод команды при помощи имен групп.

yum group list *глобальное выражение*

Несколько опциональных аргументов могут быть переданы этой команде, в том числе **hidden** для отображения групп, не помеченных как видимых для пользователя (User Visible) и **ids** для вывода идентификаторов (ID) групп. Также можно добавить опции **language**, **environment**, **installed** и **available** для сокращения вывода команды до указанного типа групп.

Для вывода более подробной информации о группе пакетов, в том числе списков обязательных и опциональных пакетов, если это поддерживается, используйте следующую команду:

yum group info *глобальное выражение*

Пример вывода информации о группе пакетов **Development Tools**:

```
[root@MAINSERVER ~]# yum group info 'Development Tools'
Loaded plugins: fastestmirror, langpacks
There is no installed groups file.
Maybe run: yum groups mark convert (see man yum)
Loading mirror speeds from cached hostfile
```

```
 * base: mirror.digitalhusky.com
 * epel: mirror.yandex.ru
 * extras: mirror.digitalhusky.com
 * updates: mirror.digitalhusky.com

Group: Development Tools
 Group-Id: development
 Description: A basic development environment.
 Mandatory Packages:
   +autoconf
   +automake
    binutils
…
   +rpm-sign
 Default Packages:
   +byacc
   +cscope
   +ctags
…
   +systemtap
 Optional Packages:
   ElectricFence
…
   translate-toolkit
```

Как можно видеть в примере, у пакетов группы возможны различные состояния, которые маркируются следующими символами:

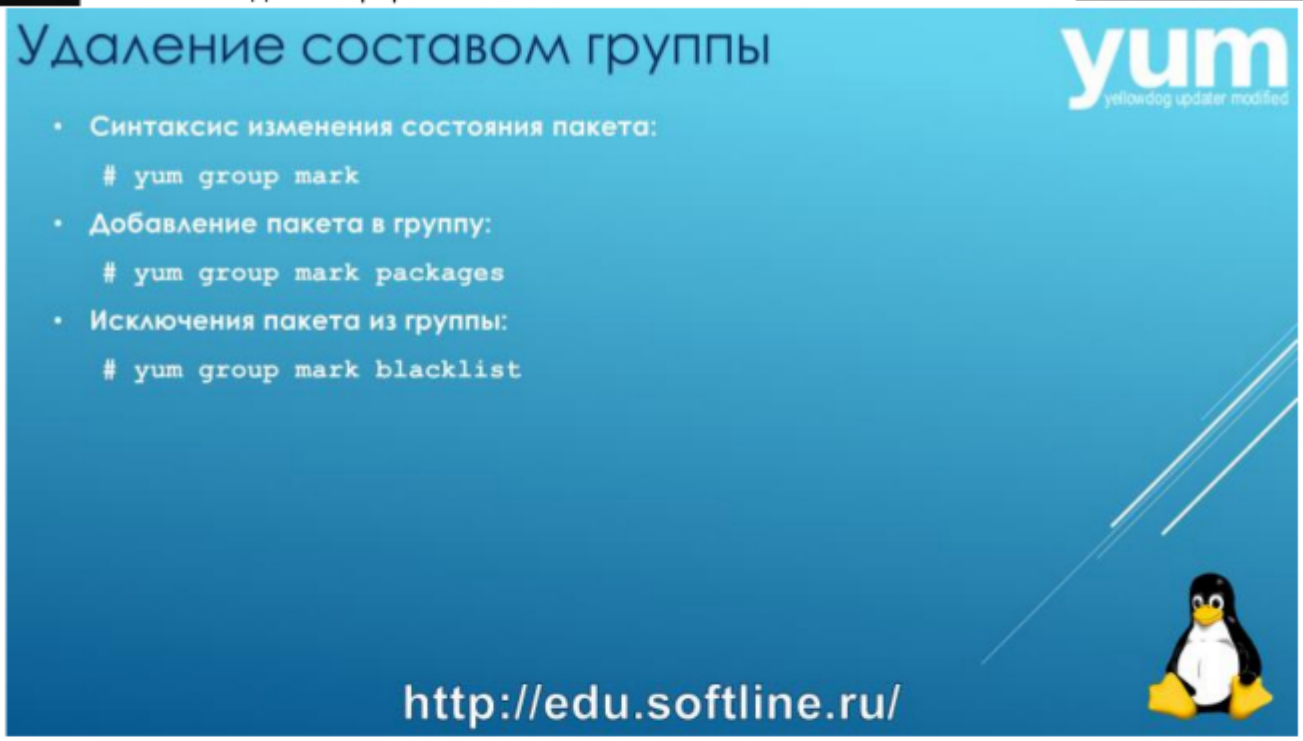
- **-** – Пакет не установлен, и он не будет установлен, как часть группы пакетов.
- **+** – Пакет не установлен, но он будет установлен, при следующем выполнении **`yum upgrade`** или **`yum group upgrade`**.
- **=** – Пакет установлен и был установлен как часть группы пакетов.
- Без символа – Пакет установлен на системе, но не в составе группы. Это значит, что он не будет удален командой **`yum group remove`**.

Эти различия имеют значения, только когда параметр настройки **`group_command`** выставлен в значение **`objects`** (параметр по умолчанию для CentOS/RHEL). Изменение значения этого параметра приведет к отключению отслеживания вхождения пактов в группы.

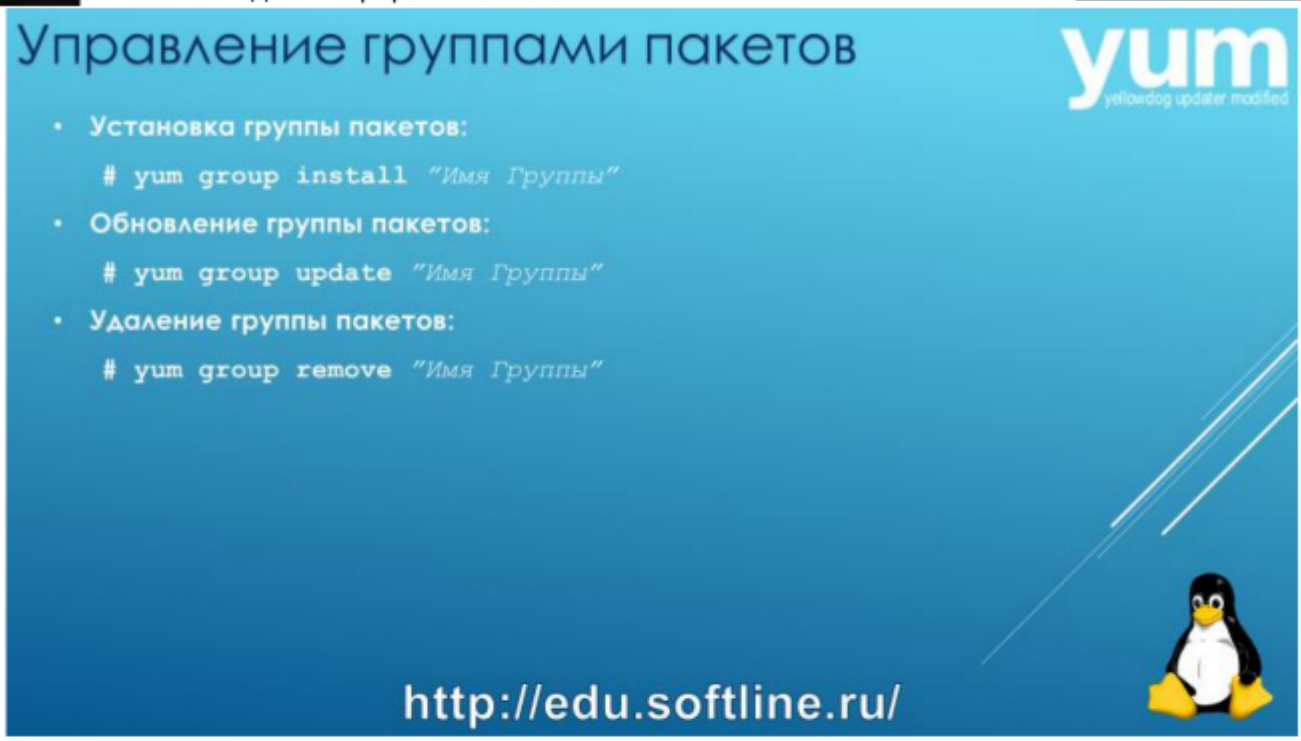


ПРИМЕЧАНИЕ:

*Вы можете определить группу окружения при помощи префикса **@^** и группу пакетов при помощи **@**. Когда используется **`yum group list`**, **`info`**, **`install`** или **`remove`** передайте **@***имя_группы* *для указания группы пакетов, **@^***имя_группы* *для указания группы окружения или* *имя_группы* *для указания обоих типов групп.*



Вы можете изменить состояния пакета при помощи команды **yum group mark**. Например, **yum group mark packages** сделает любые указанные пакеты членами указанной группы. Для того чтобы избежать установки новых пакетов при обновлении группы, используйте команду **yum group mark blacklist**. Чтобы больше узнать о возможностях **yum group mark**, обратитесь к страницам руководства **yum(8)**.



Установка групп пакетов.

Каждая группа пакетов имя и идентификатор группы (*groupid*). Для вывода имен всех групп пакетов и идентификаторов групп, воспользуйтесь следующей командой:

```
yum group list ids
```

Для поиска имени и идентификатора группы пакетов, связаных с окружением рабочего стола KDE, воспользуйтесь следующей командой:

```
[root@MAINSERVER ~]# yum group list ids kde\*
Loaded plugins: fastestmirror, langpacks
There is no installed groups file.
Maybe run: yum groups mark convert (see man yum)
Loading mirror speeds from cached hostfile
 * base: mirror.digitalhusky.com
 * epel: mirror.yandex.ru
 * extras: mirror.digitalhusky.com
 * updates: mirror.digitalhusky.com
Available environment groups:
   KDE Plasma Workspaces (kde-desktop-environment)
Done
```

Некоторые группы скрыты параметрами в настроенных репозиториях. Для их отображения можно использовать опцию **hidden**.

```
[root@MAINSERVER ~]# yum group list hidden ids kde\*
Loaded plugins: fastestmirror, langpacks
There is no installed groups file.
Maybe run: yum groups mark convert (see man yum)
Loading mirror speeds from cached hostfile
 * base: mirror.digitalhusky.com
 * epel: mirror.yandex.ru
 * extras: mirror.digitalhusky.com
 * updates: mirror.digitalhusky.com
Available environment groups:
   KDE Plasma Workspaces (kde-desktop-environment)
Available Groups:
   KDE (kde-desktop)
```

```
   KDE Applications (kde-apps)
   KDE Multimedia Support (kde-media)
Done
```

Для установки группы пакетов, можно использовать ее имя:

yum group install *"Имя группы"*

Также для установки можно использовать идентификатор группы (*groupid*):

yum group install *groupid*

Если вы используете символ @ для указания группы пакетов или @^ для указания группы окружения, можно опустить слово group при установке:

yum install @*группа*
yum install @^*группа*

Пример четырех способов установки группы пакетов:

```
[root@MAINSERVER ~]# yum group install "KDE Plasma Workspaces"
[root@MAINSERVER ~]# yum group install kde-desktop-environment
[root@MAINSERVER ~]# yum install @^"KDE Plasma Workspaces"
[root@MAINSERVER ~]# yum install @^kde-desktop-environment
```

Удаление групп пакетов.

Для удаления группы пакетов используется синтаксис аналогичный install с указанием имени группы или идентификатора:

yum group remove *"Имя группы"*

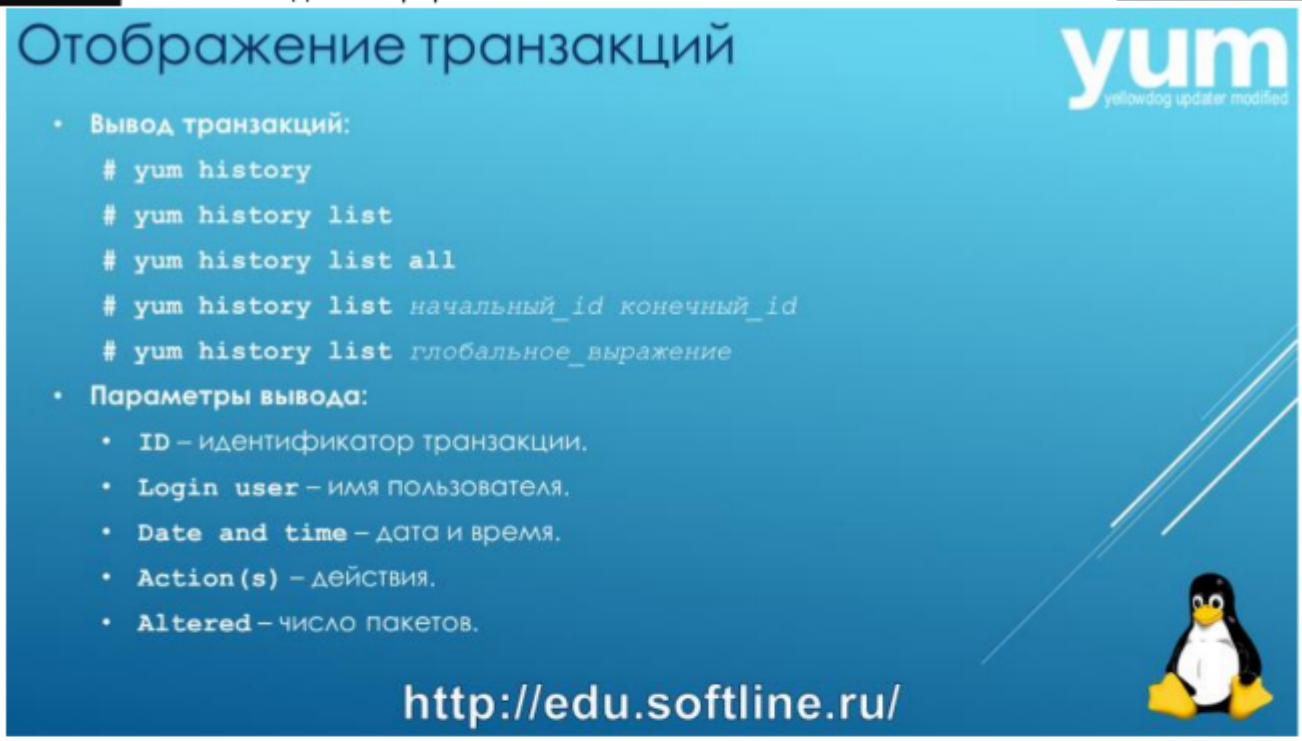
yum group remove *groupid*

Также можно использовать имя группы в кавычках или идентификатор группы с символом **@** (для указания группы пакетов) или **@^** (для указания группы окружения), без использования команды **group**:

yum remove @*группа*
yum remove @^*группа*

Пример четырех способов удаления группы пакетов:

```
[root@MAINSERVER ~]# yum group remove "KDE Plasma Workspaces"
[root@MAINSERVER ~]# yum group remove kde-desktop-environment
[root@MAINSERVER ~]# yum remove @^"KDE Plasma Workspaces"
[root@MAINSERVER ~]# yum remove @^kde-desktop-environment
```



Команда **yum history** позволяет пользователям просмотреть информацию о временных шкалах транзакций yum, датах и времени их выполнения, числу задействованных пакетов, когда транзакция успешно выполнилась или была прервана и изменениях в базе данных RPM между транзакциями. Дополнительно, эта команда может быть использована для отмены или повторения указанной транзакции. Все данные истории сохраняются в базе данных истории (History DB) в каталоге **/var/lib/yum/history/**.

Отображение транзакций.

Для отображения списка из двадцати последних транзакций, от имени пользователя root, выполните одну из следующих команд:

```
# yum history
# yum history list
```

Для отображения всех транзакций, добавьте ключевое слово **all**:

```
# yum history list all
```

Для отображения транзакций из указанного диапазона, используйте следующую форму:

```
# yum history list начальный_ID конечный_ID
```

Также можно отобразить транзакции соответствующую указанному пакету или пакетам. Для этого используйте команд **yum history list** с именем пакета или глобальным выражением (Global Expression):

```
# yum history list глобальное_выражение
```

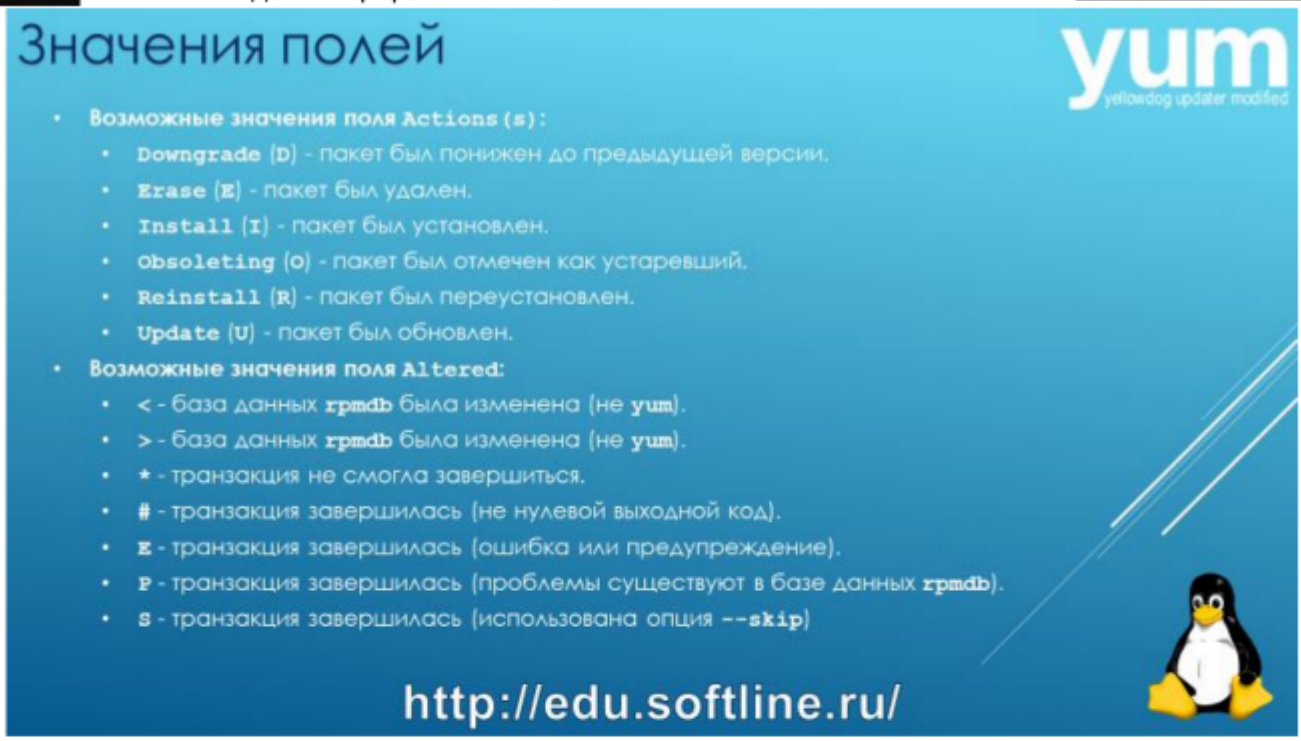
Пример вывода 5 самых старых транзакций.

В выводе команды yum history list самые последние транзакции отображаются в начале списка. Для отображения информации о пяти самых старых транзакций, хранящихся в базе данных истории, используйте следующую команду:

```
[root@MAINSERVER ~]# yum history list 1..5
Loaded plugins: langpacks, product-id, subscription-manager
ID      | Login user          | Date and time          | Action(s)     | Altered
-------------------------------------------------------------------------------
5       | User <user>         | 2013-07-29 15:33       | Install       | 1
4       | User <user>         | 2013-07-21 15:10       | Install       | 1
3       | User <user>         | 2013-07-16 15:27       | I, U          | 73
2       | System <unset>      | 2013-07-16 15:19       | Update        | 1
1       | System <unset>      | 2013-07-16 14:38       | Install       | 1106
history list
```

Все формы команды yum history list создают табличный вывод, в котором каждая строка состоит из следующих колонок:

- **ID** – Целочисленное значение, определяющее транзакцию.
- **Login user** –Имя пользователя, чья сессия была использована для начала транзакции. Эта информация обычно предоставляется в форме: **Full Name <username>**. Для транзакций, которые были вызваны не пользователем (такие как автоматическое обновление системы) используется **System <unset>**.
- **Date and time** – Дата и время вызова транзакции.
- **Action(s)** – список действий, которые были выполнены в рамках транзакции.
- **Altered** –Число пакетов, которые были задействованы в транзакции, возможно будет сопровождаться дополнительной информацией.

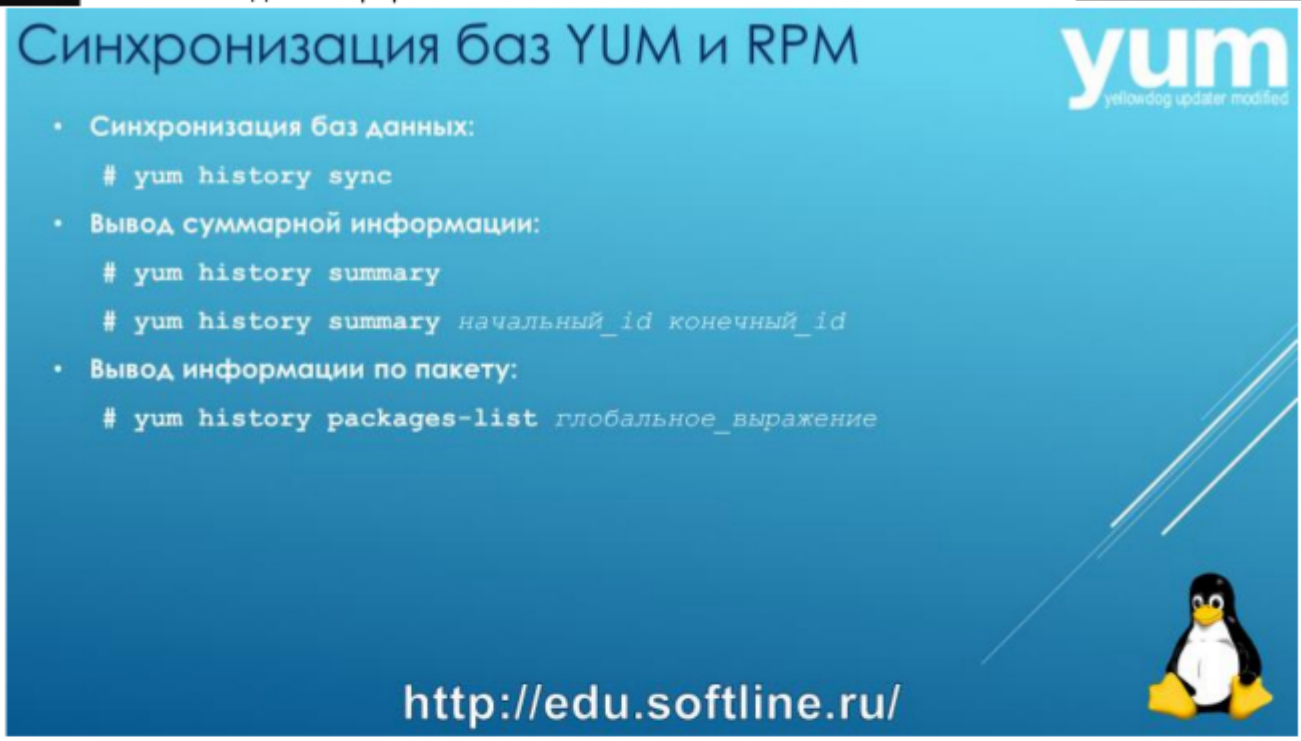


Возможные значения поля **Action(s)**:

Действие	Аббревиатура	Описание
Downgrade	**D**	Как минимум один пакет был понижен до предыдущей версии.
Erase	**E**	Как минимум один пакет был удален.
Install	**I**	Как минимум один пакет был установлен.
Obsoleting	**O**	Как минимум один пакет был отмечен как устаревший.
Reinstall	**R**	Как минимум один пакет был переустановлен.
Update	**U**	Как минимум один пакет был обновлен.

Возможные значения поля **Altered**:

Символ	Описание
<	Прежде чем транзакция завершилась база данных **rpmdb** была изменена (не yum).
>	После завершения транзакции база данных **rpmdb** была изменена (не yum).
*****	Транзакция не смогла завершиться.
#	Транзакция успешно завершилась, но **yum** вернул не нулевой выходной код.
E	Транзакция успешно завершилась, но ошибка или предупреждение была отображена.
P	Транзакция успешно завершилась, но проблемы уже существуют в базе данных **rpmdb**.
S	Транзакция успешно завершилась, но была использована опция --skip-



Для синхронизации контента базы данных rpmdb или yumdb для любых установленных пакетов с используемой сейчас rpmdb или yumdb, введите следующую команду:

```
# yum history sync
```

Для отображения общей статистики из используемой на данный момент базы данных введите следующую команду:

```
[root@MAINSERVER ~]# yum history stats
Loaded plugins: langpacks, product-id, subscription-manager
File         : //var/lib/yum/history/history-2012-08-15.sqlite
Size         : 2,766,848
Transactions : 41
Begin time   : Wed Aug 15 16:18:25 2012
End time     : Wed Feb 27 14:52:30 2013
Counts       :
   NEVRAC    : 2,204
   NEVRA     : 2,204
   NA        : 1,759
   NEVR      : 2,204
   rpm DB    : 2,204
   yum DB    : 2,204
history stats
```

Yum также позволяет отобразить сумму всех последних транзакций. Для этого воспользуйтесь следующую командой:

```
# yum history summary
```

Для отображения транзакций только в указанном диапазоне. Введите следующую команду:

```
# yum history summary начальный_ID..конечный_ID
```

По аналогии с командой **yum history list**, можно отобразить суммарную информацию по транзакциям в которых использовался указанный пакет или пакеты, за счет передачи имени пакета или глобального выражения (Glob Expression):

```
# yum history list глобальное_выражение
```

Пример получения суммарной информации по пяти самым старым транзакциям:

```
[root@MAINSERVER ~]# yum history summary 1..5
Loaded plugins: langpacks, product-id, subscription-manager
Login user | Time | Action(s) |
Altered
------------------------------------------------------------------------
---------
Jaromir ... <jhradilek> | Last day | Install | 1
Jaromir ... <jhradilek> | Last week | Install | 1
Jaromir ... <jhradilek> | Last 2 weeks | I, U | 73
System <unset> | Last 2 weeks | I, U | 1107
history summary
```

Все формы команды yum history summary выдают упрощенный табличный вывод, похожий на вывод команды yum history list.

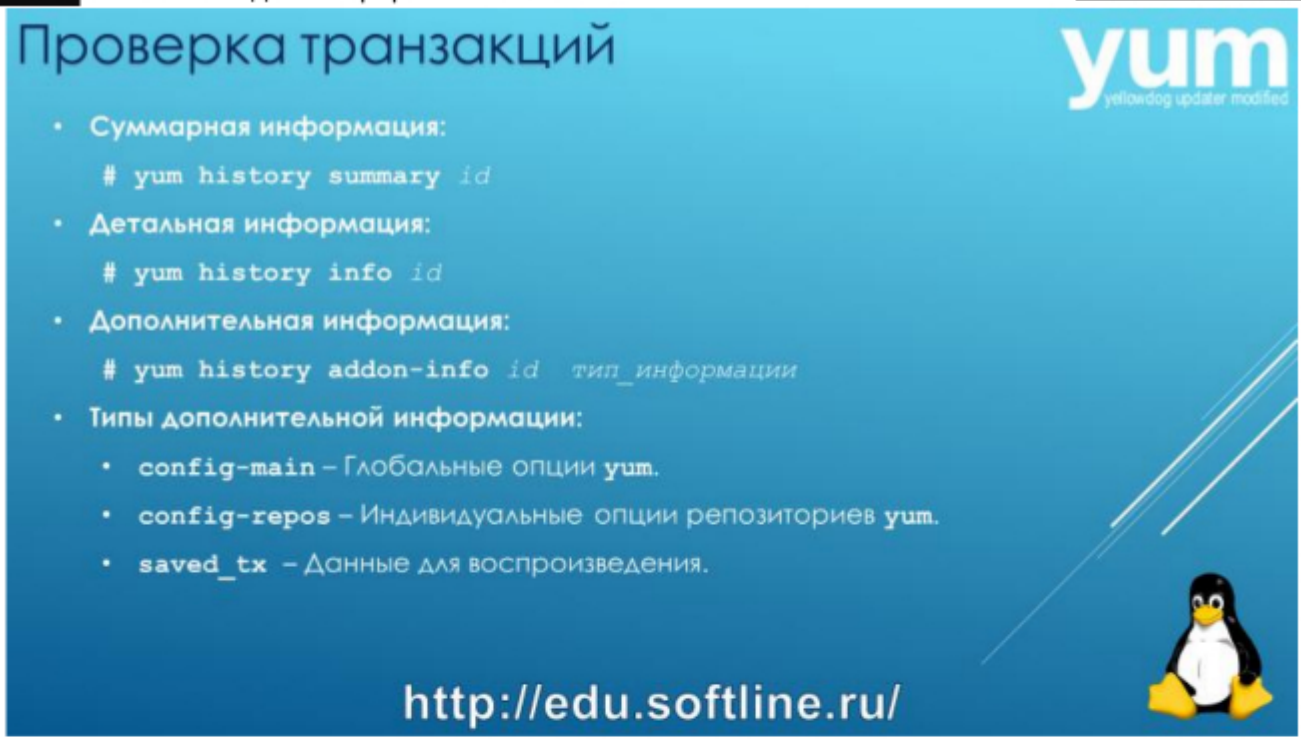
Как показано выше, команды **yum history list** и **yum history summary** ориентированы на транзакции и, несмотря на возможность отобразить только транзакции, связанные с указанным пакетом или пакетами, они не отображают важных подробностей, таких как версии пакетов. Для вывода списка транзакций с точки зрения пакета, выполните следующую команду от имени пользователя root:

```
# yum history package-list глобальное_выражение…
```

Например, для отслеживания пакета **rpm** и связанных пакетов в истории, введите следующую команду:

```
[root@MAINSERVER ~]# yum history packages-list rpm\*
Loaded plugins: fastestmirror, langpacks
Repodata is over 2 weeks old. Install yum-cron? Or run: yum makecache fast
ID     | Action(s)      | Package
------------------------------------------------------------------------------
     2 | Updated        | rpm-4.11.1-16.el7.x86_64                             EE
     2 | Update         |     4.11.1-25.el7.x86_64                             EE
     2 | Updated        | rpm-build-libs-4.11.1-16.el7.x86_64                  EE
     2 | Update         |                 4.11.1-25.el7.x86_64                 EE
     2 | Updated        | rpm-libs-4.11.1-16.el7.x86_64                        EE
     2 | Update         |           4.11.1-25.el7.x86_64                       EE
     2 | Updated        | rpm-python-4.11.1-16.el7.x86_64                      EE
     2 | Update         |             4.11.1-25.el7.x86_64                     EE
     1 | Install        | rpm-4.11.1-16.el7.x86_64
     1 | Dep-Install    | rpm-build-libs-4.11.1-16.el7.x86_64
     1 | Dep-Install    | rpm-libs-4.11.1-16.el7.x86_64
     1 | Dep-Install    | rpm-python-4.11.1-16.el7.x86_64
history packages-list
```

В данном примере 4 пакета установлено в процессе начальной установки системы: **rpm-4.11.1-16.el7.x86_64**, **rpm-build-libs-4.11.1-16.el7.x86_64**, **rpm-libs-4.11.1-16.el7.x86_64**, **rpm-python-4.11.1-16.el7.x86_64**. Во второй транзакции все эти пакеты были обновлены.



Для отображения суммарной информации об отдельной транзакции используйте команду yum history summary в следующей форме:

```
# yum history summary id
```

Где *id* – это идентификатор транзакции.

Для более детальной проверки транзакции или транзакций, используйте следующую команду от имени пользователя root:

```
# yum history info id…
```

Аргумент id опциональный, поэтому, если вы его не указываете, будет выведена информация о последней транзакции. Также для получения информации по нескольким транзакциям вы можете использовать диапазон:

```
# yum history info начальный_id..конечный_id
```

Следующий пример демонстрант вывод двух транзакций:

```
[root@MAINSERVER ~]# yum history info 4..5
Loaded plugins: fastestmirror, langpacks
Transaction ID : 4..5
Begin time     : Mon Apr  6 11:11:12 2015
Begin rpmdb    : 484:6156b656808c1ad77d27d0bd16057e74a2b082c4
End time       :          20 08:30:43 2015 (13 days)
End rpmdb      : 490:5d68ce8d29f71639555cd26f668212164c99ce57
User           : root <root>
Return-Code    : Success
Command Line   : install bind-chroot
Command Line   : install httpd
Transaction performed with:
    Installed     rpm-4.11.1-25.el7.x86_64                    @base
    Installed     yum-3.4.3-125.el7.centos.noarch             @base
    Installed     yum-plugin-fastestmirror-1.1.31-29.el7.noarch @base
Packages Altered:
    Dep-Install apr-1.4.8-3.el7.x86_64                @base
    Dep-Install apr-util-1.5.2-6.el7.x86_64           @base
```

```
    Install     bind-chroot-32:9.9.4-18.el7_1.1.x86_64 @updates
    Install     httpd-2.4.6-31.el7.centos.x86_64       @base
    Dep-Install httpd-tools-2.4.6-31.el7.centos.x86_64 @base
    Dep-Install mailcap-2.1.41-2.el7.noarch            @base
history info
```

Также можно просматривать дополнительную информацию, такую как использованные опции настройки во время транзакции или репозиторий, из которого были установлены пакеты. Для определения доступной информации для определенной транзакции, выполните следующую команду от имени пользователя root:

```
# yum history addon-info id
```

Подобно команде yum history info, когда не указан *id*, yum автоматически использует транзакцию. Другой путь получить последнюю транзакцию – это использовать ключевое слово **last**:

```
# yum history addon-info last
```

Для четвертой транзакции в истории команда **yum history addon-info** предоставляет следующий вывод:

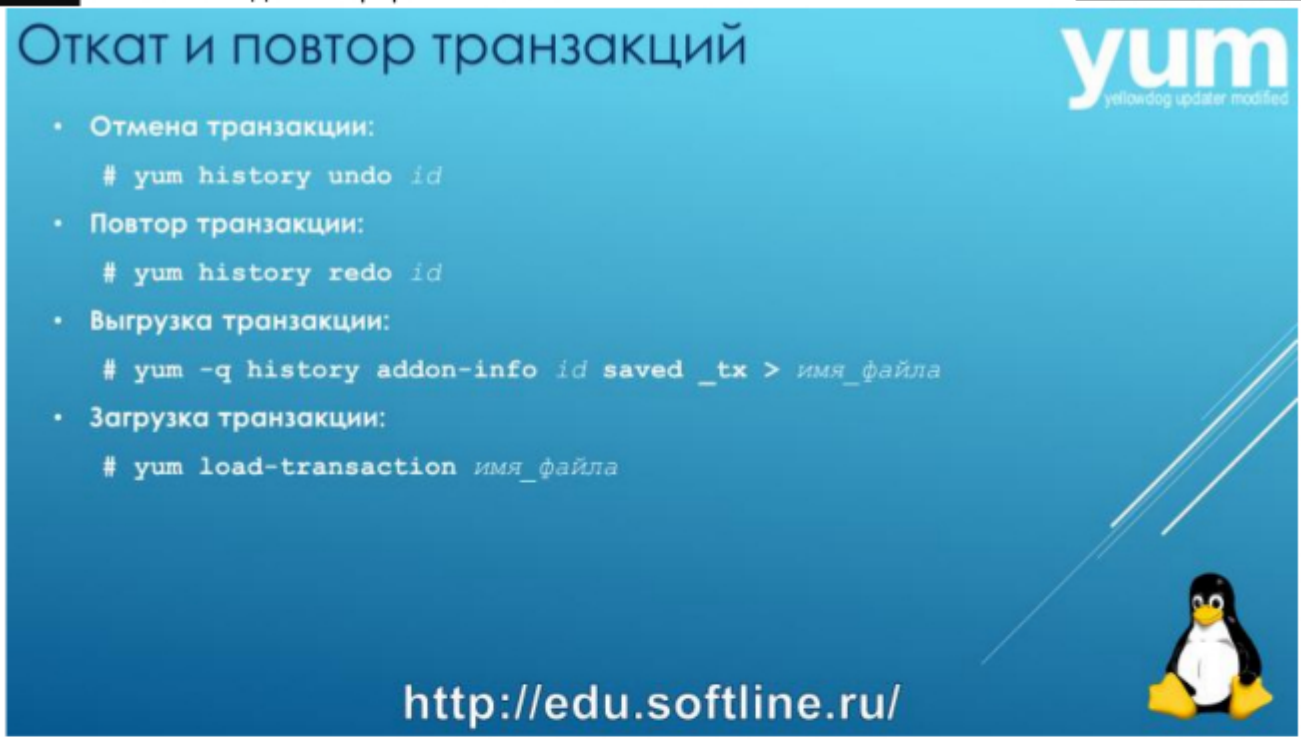
```
[root@MAINSERVER ~]# yum history addon-info 4
Loaded plugins: fastestmirror, langpacks
Transaction ID: 4
Available additional history information:
  config-main
  config-repos
  saved_tx

history addon-info
```

В выводе команды **yum history addon-info** доступно три типа информации:

- **config-main** – Глобальные опции yum (располагаются в секции **[main]**, конфигурационного файла **/etc/yum.conf**), которая была использована в процессе выполнения транзакции.
- **config-repos** – Индивидуальные опции репозиториев yum (располагаются в секции **[repository]** конфигурационного файла репозитория.)
- **saved_tx** – Данные, которые могут быть использованы командой **yum load-transaction** в порядке повторения транзакции на другой машине.

Для отображения выбранного типа информации, выполните следующую команду от имени пользователя root:

```
# yum history addon-info id тип_информации
```



Независимо от просмотра истории транзакций, команда **yum history** предоставляет возможность отката и повторения выбранной транзакции. Для отката транзакции, воспользуйтесь следующей командой от имени пользователя root:

```
# yum history undo id
```

Для повторения выбранной транзакции, воспользуйтесь следующей командой:

```
# yum history redo id
```

Обе команды также принимают ключевое слово **last** для отката или повторения предыдущей транзакции.

Обратите внимание, что обе команды **yum history undo** и **yum history redo** откатывают или повторяют только те шаги, которые были выполнены в процессе выполнения транзакции. Если транзакция устанавливала пакет, то команда **yum history undo** удалит его, а если пакет был удален во время транзакции, то команда **yum history undo** установит его. Также эта команда попробует понизить версию всех обновленных пакетов до предыдущей версии, если более старая версия пакета по-прежнему доступна.

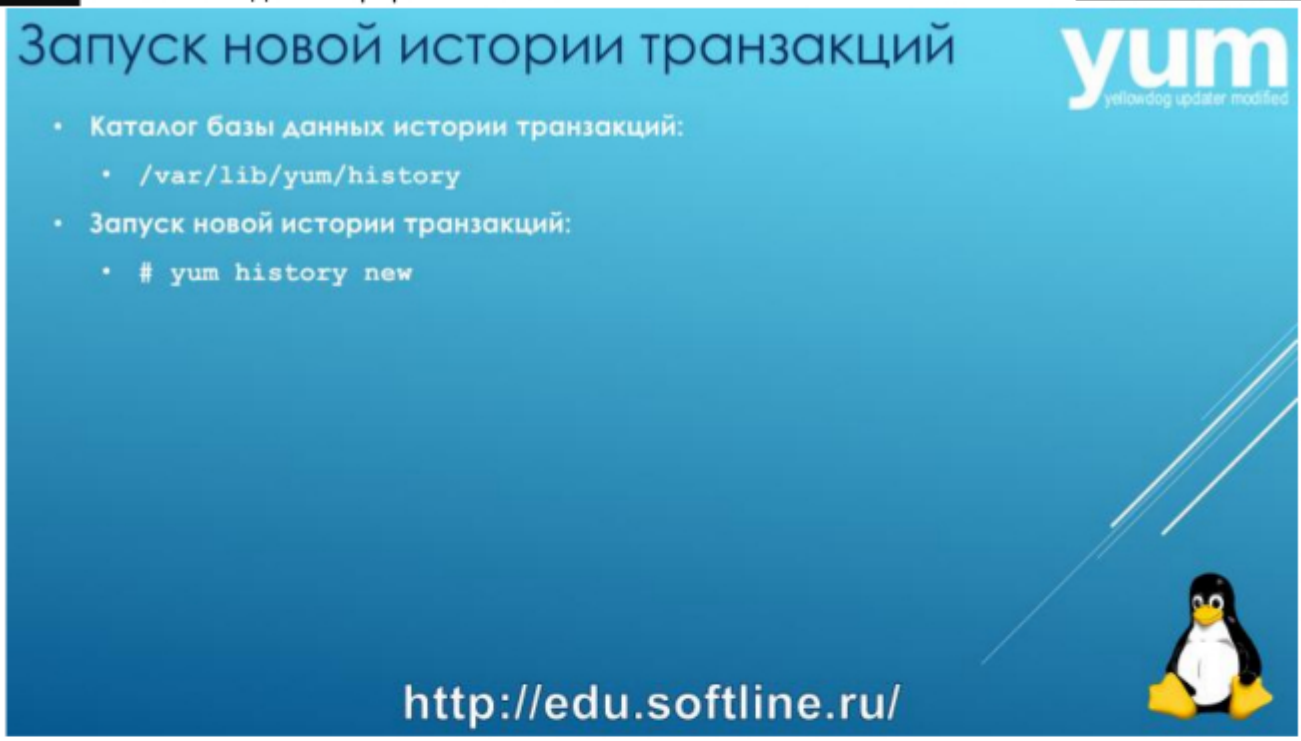
Когда требуется управлять несколькими одинаковыми системами, yum также позволяет выполнять транзакцию на одной из них, сохранять подробности транзакции в файле и после проверки, повторять эту транзакцию на других системах. Для сохранения подробностей транзакции в файле, выполните следующую команду от имени пользователя root:

```
# yum -q history addon-info id saved_tx > имя_файла
```

После того, как вы скопируете файл на целевую систему, вы можете повторить транзакцию при помощи следующей команды:

```
# yum load-transaction имя_файла
```

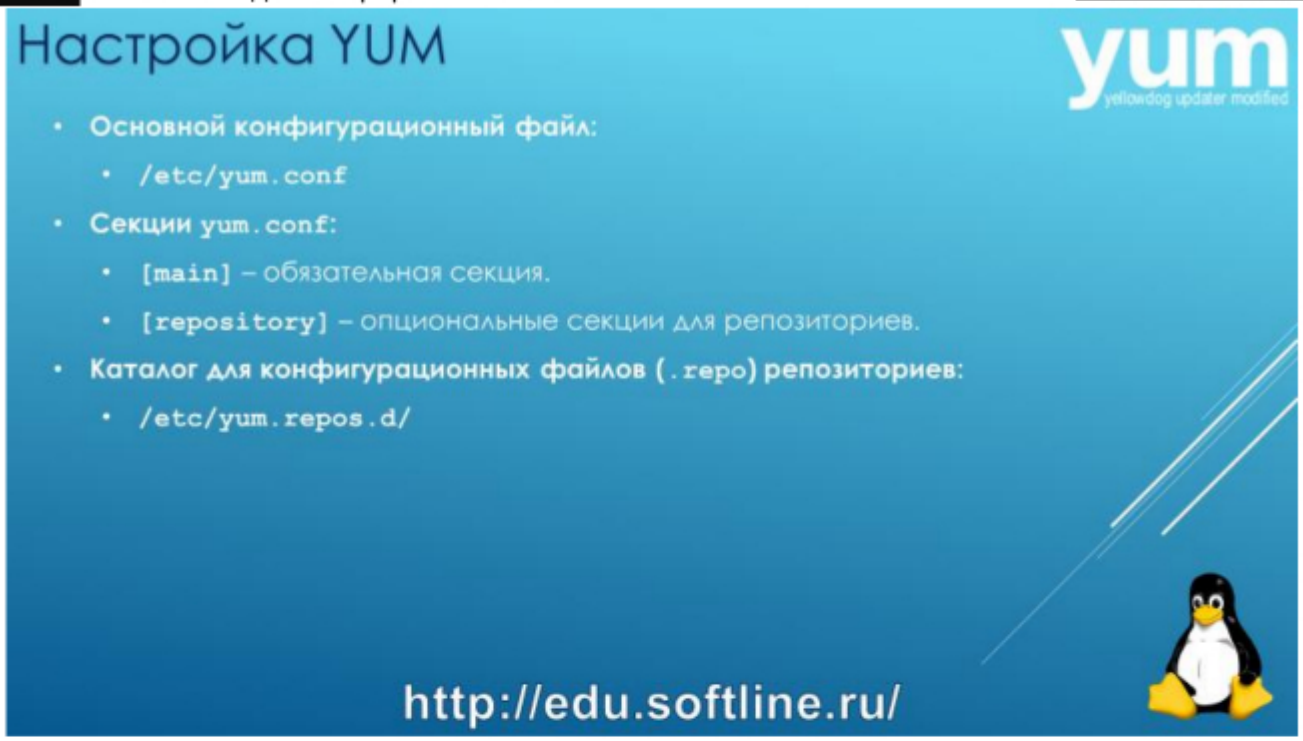
Вы можете настроить **load-transaction** на игнорирование недостающих пакетов или версию **rpmdb**. Дополнительную информацию об этом можно найти на странице руководства **yum.conf(5)**.



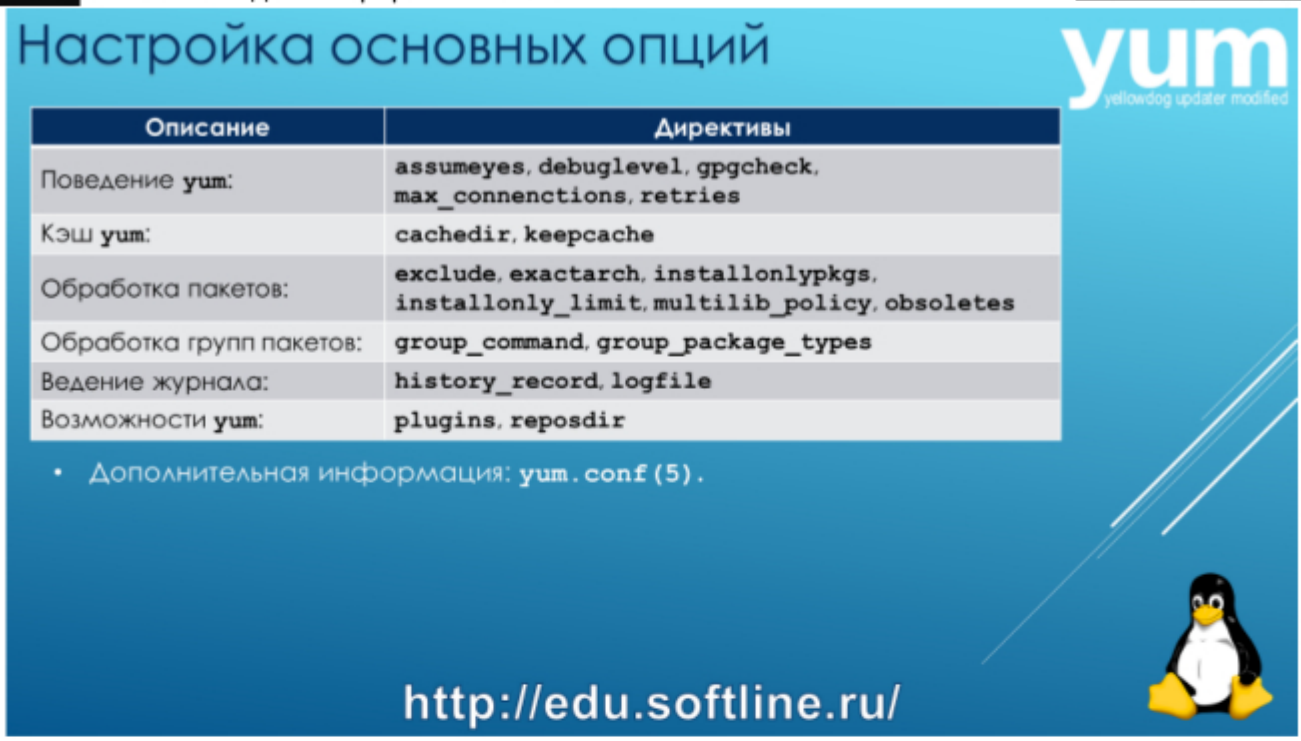
Yum хранит историю транзакций в отдельном файле базы данных SQLite. Для запуска новой истории транзакций, выполните следующую команду от имени пользователя root:

```
# yum history new
```

Это создаст новый, пустой файл базы данных в каталоге **/var/lib/yum/history**. Старый файл истории транзакции будет сохранен, но будет недоступен до тех пор, пока новый файл базы данных представлен в каталоге.



Информация по настройке для yum и связанных утилит располагается в **/etc/yum.conf**. Этот файл содержит одну обязательную секцию **[main]**, которая позволяет настраивать глобальные опции yum и может содержать одну или более секций **[repository]**, которые позволяют устанавливать специфичные опции для отдельных репозиториев. Однако, рекомендуется определять репозитории в новых или существующих файлах с расширением **.repo** в каталоге **/etc/yum.repos.d/**. Значения опций, определенные в секции **[repository]** переписывают значения, установленные в секции **[main]** файла **/etc/yum.conf**.



Конфигурационный файл **/etc/yum.conf** содержит только одну секцию **[main]**, которая состоит из пар ключей и значений, которые определяют, как будет работать yum. Вы можете добавить множество дополнительных опций в секцию **[main]**.

Пример конфигурационного файла **/etc/yum.conf**:

```
[main]
cachedir=/var/cache/yum/$basearch/$releasever
keepcache=0
debuglevel=2
logfile=/var/log/yum.log
exactarch=1
obsoletes=1
gpgcheck=1
plugins=1
installonly_limit=5
bugtracker_url=http://bugs.centos.org/set_project.php?project_id=23&ref=http
 ://bugs.centos.org/bug_report_page.php?category=yum
distroverpkg=centos-release

#  This is the default, …
```

Наиболее часто используемые опции в секции **[main]**:

assumeyes=*значение*

Опция **assumeyes** определяет запрашивать или не запрашивать подтверждение при критических действиях. Поддерживаемые значения:

- 0 (по умолчанию) – yum запрашивает подтверждение перед выполнением всех критических действий.
- 1 – yum не запрашивает подтверждение при критических действия также, как если бы команда **yum** выполнялась с опцией **-y** или **--assumeyes**.

cachedir=*каталог*

Данная опция устанавливает каталог, в котором Yum сохраняет свой кэш и файлы базы данных. По умолчанию используется каталог **/var/cache/yum/$basearch/$releasever**.

debuglevel=*значение*

Данная опция указывает детализацию отладочного вывода, производимого yum. Значение может быть установлено в диапазоне от **1** до **10**. Установка более высокого значения **debuglevel** заставит yum отображать более подробный отладочный вывод. По умолчанию используется значение **2**, значение **0** отключит отображение отладочной информации.

exactarch=*значение*

Данная опция указывает определяет точно соответствие архитектуре при обновлении уже установленного пакета. Поддерживаемые значения:

- **0** – не соблюдает архитектуру при обновлении пакетов.
- **1** (по умолчанию) – не позволяет обновлять уже установленные пакеты, пакетами другой архитектуры. Например, нельзя установить более новый пакет с архитектурой 32 бита, если уже установлен пакет с архитектурой 64 бита.

exclude=*имя_пакета [имя_пакета]*

Опция **exclude** позволяет исключить пакеты, по ключевым словам, в процессе установки или обновления системы. Перечисление множества пакетов может быть заключено в кавычки и разделено пробелами. Также поддерживаются глобальные выражения (Glob Expressions) с подстановочными знаками (***** и **?**).

gpgcheck=*значение*

Опция **gpgcheck** заставляет yum выполнять проверку подписи GPG на пакетах. Поддерживаемые значения:

- 0 – Отключает проверку GPG подписей на пакетах во всех репозиториях, в том числе при локальной установке пакетов.
- 1 (по умолчанию) – Включает проверку подписей GPG на всех пакетах во всех репозиториях, в том числе при локальной установке.

group_command=*значение*

Данная опция указывает поведение команд **yum group install** , **yum group upgrade** и **yum group remove** при обработке группы пакетов. Поддерживаемые значения:

- **simple** – Устанавливать всех членов группы пакетов. Обновлять только ранее установленные пакеты, но не устанавливать пакеты, которые были добавлены в группу позже.
- **compat** – Похоже на **simple** , но yum upgrade также устанавливает пакеты, которые были добавлены в группу с момента предыдущего обновления.
- **objects** (по умолчанию) – yum продолжает отслеживать ранее установленные группы и пакеты установленные в группе и отдельно.

group_package_types=package_type [more_package_types]

Указывает какой тип пакета (**optinal**, **default** или **mandatory**) устанавливается, когда вызывается команда **yum group install**. По умолчанию используется тип пакета **mandatory**.

history_record=*значение*

Включает или отключает ведение истории транзакций.

- **0** – yum не записывает историю транзакций.
- **1** (по умолчанию) – yum записывает историю транзакций. Эта операция требует дополнительное место на диске и дополнительное время в транзакции, но предоставляет много информации о прошедших операциях, которая может быть отображена при помощи команды **yum history**.



ПРИМЕЧАНИЕ:

*Yum использует запись истории для определения изменений в базе данных **rpmdb**, которые были внесены без участия yum. В таких случаях yum отображает предупреждение и автоматически ищет проблему, вызванную изменением **rpmdb**. Когда опция **history_record** отключена, yum не может определить изменения и не выполняет автоматическую проверку.*

installonlypkgs=*разделенный пробелами список пакетов*

Указывает список пакетов, разделенных пробелами, которые yum может установить, но не может обновить.

installonly_limit=*значение*

Указывает как много пакетов, указанных в директиве **installonlypkgs** может быть установлено одновременно.

keepcache=*значение*

Определяет удержание заголовков и пакетов после успешной установки в кэше.

Значение может принимать:

- **0** (по умолчанию) – не хранить заголовки и пакеты кэше после успешной установки.
- **1** – хранить заголовки и пакеты кэше после успешной установки.

logfile=*имя_файла*

Определяет расположение вывода журнала, определяется указанием абсолютного пути. По умолчанию используется: **/var/log/yum.log**.

max_connenctions=*число*

Определяет максимальное число конкурентных подключений. По умолчанию 5.

multilib_policy=*значение*

Определяет поведение установки, в случае доступности нескольких архитектур.

Значение может принимать:

- **best** – устанавливать пакеты соответствующие архитектуре системы в первую очередь. Например, для систем на базе AMD64 всегда будет устанавливаться 64-битная версия пакета.
- **all** – устанавливать пакеты всех доступных архитектур. Например, для систем на базе AMD64 всегда будет устанавливаться обе версии пакета, если доступны i686 и AMD64.

obsoletes=*значение*

Определяет обновления пакетов до более новой версии. Когда один пакете определяет устаревание другого пакета, который при установке будет удален.

Значение может принимать:

- **0** – отключает логику обработки устаревания yum в процессе выполнения обновления.
- **1** (по умолчанию) – включает логику обработки устаревания yum в процессе выполнения обновления.

plugins=*значение*

Данный глобальный параметр включает или отключает плагины yum.

Значение может принимать:

- **0** – Глобально отключает все плагины yum.
- **1** (по умолчанию) - Глобально включить все плагины yum, при этом остается возможность отключить необходимые плагины выставив параметр **enabled=0** в конфигурационном файле плагина.



ПРИМЕЧАНИЕ:

Не рекомендуется отключать все плагины, так как некоторые плагины предоставляют важные службы yum. Глобальное отключение как правило используется в целях диагностики и выявления неисправностей.

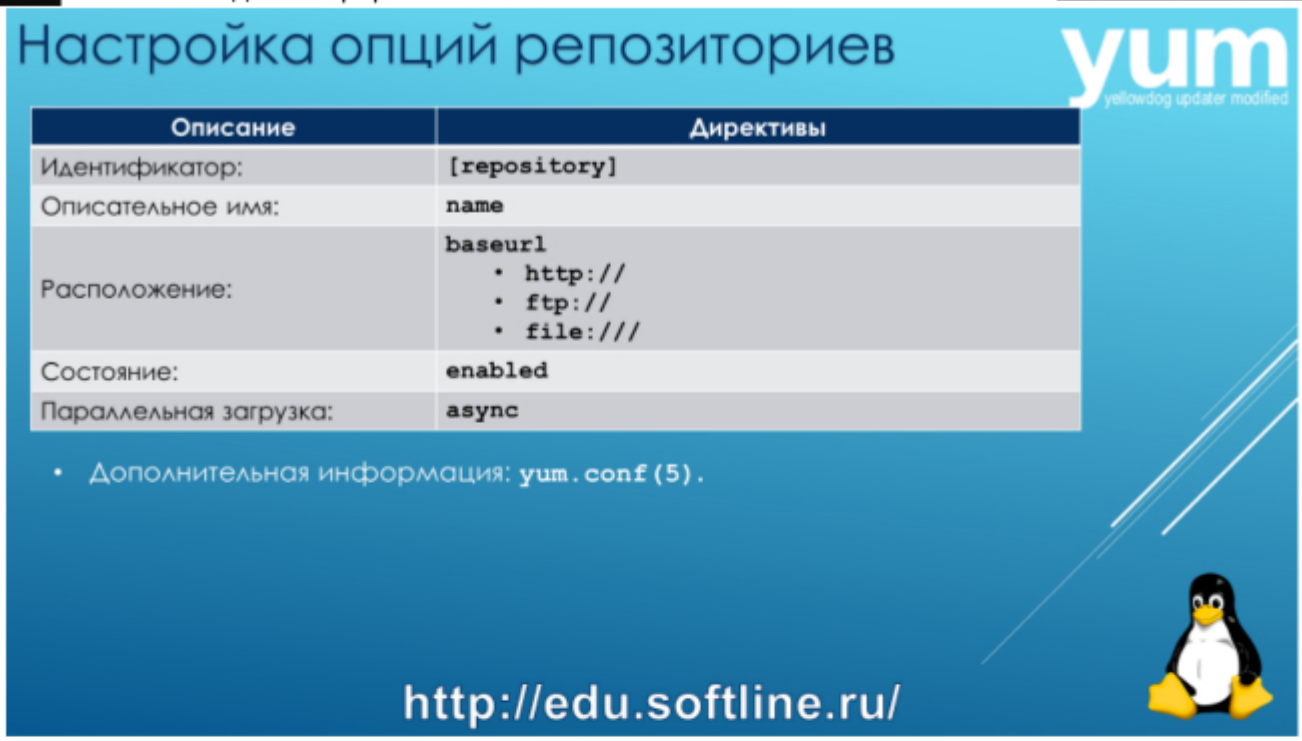
reposdir=*каталог*

Указывает абсолютный путь к каталогу с конфигурационными файлами репозиториев (**.repo**). Все **.repo** файлы содержат информацию о репозиториях (подобно секции **[repository]** файла **/etc/yum.conf)**. Yum собирает информацию из всех файлов .repo и секции **[repository]** файла **/etc/yum.conf** и создает список репозиториев, который используется в транзакциях. Если опция **reposdir** не указана используется каталог по умолчанию **/etc/yum.repos.d/**.

retries=*значение*

Указывает число попыток получить доступ к файлу перед выдачей ошибки. Значение должно быть целочисленным, указанным в диапазоне от **0** и больше. Значение **0** заставляет делать бесконечно число попыток. Значение по умолчанию – **10**.

Для получения полного списка всех доступных опций **[main]**, влспльзуйтесь секцией **[main] OPTIONS** страницы руководства **yum.conf**(5).



Секция **[***repository***]**, в которой *repository* – это уникальный идентификатор, такой как **c72_x64_repo** (пробелы не разрешены), позволяет вам определить индивидуальные репозитории yum. Во избежание конфликтов, настраиваемые репозитории не должны использовать имена репозиторриев CentOS.

Следующий пример демонстрирует минимальную форму определения секции **[***repository***]**:

```
[repository]
name=имя_репозитория
baseurl=url_репозитория
```

Каждая секция **[***repository***]** должна содержать следующие директивы:

name=*имя_репозитория*

Указывает описательное имя для репозитория.

baseurl=*url_репозитория*

Указывает URL до каталога, содержащего репозиторий:

- URL для репозитория доступного по протоколу HTTP: **http://***путь/до/репозитория*
- URL для репозитория доступного по протоколу FTP: **ftp://***путь/до/репозитория*
- URL для репозитория на локальном диске: **file:///***путь/до/репозитория*

Если указанный онлайн репозиторий требует базовую аутентификацию HTTP, вы можете указать имя пользователя и пароль в строке URL, в следующем формате:

имя_пользователя:*пароль*@*ссылка*.

Например: **http://user:password@www.example.com/repo/**.

Обычно URL – это HTTP-ссылка, следующего вида:

baseurl=http://*сервер***/releases/$releasever/server/$basearch/os/**

Где **$releasever** и **$basearch** – это переменные.

Прочие полезные директивы секции **[***repository***]**:

enabled=*значение*

Определяет использование репозитрия (включен/отключен).

Значение может принимать:

- 0 – не использовать репозиторий в качестве источника пакетов.
- 1 - использовать репозиторий в качестве источника пакетов.

Включать и откалючать репозитории можно при помощи опций команды yum:

--enablerepo=*имя_репозитория* и **--disablerepo=***имя_репозитория* и окна **Add/Remove Software** утилиты **PackageKit**.

async=*значение*

Управляет параллельной загрузкой пакетов из репозитория.

Значение может принимать:

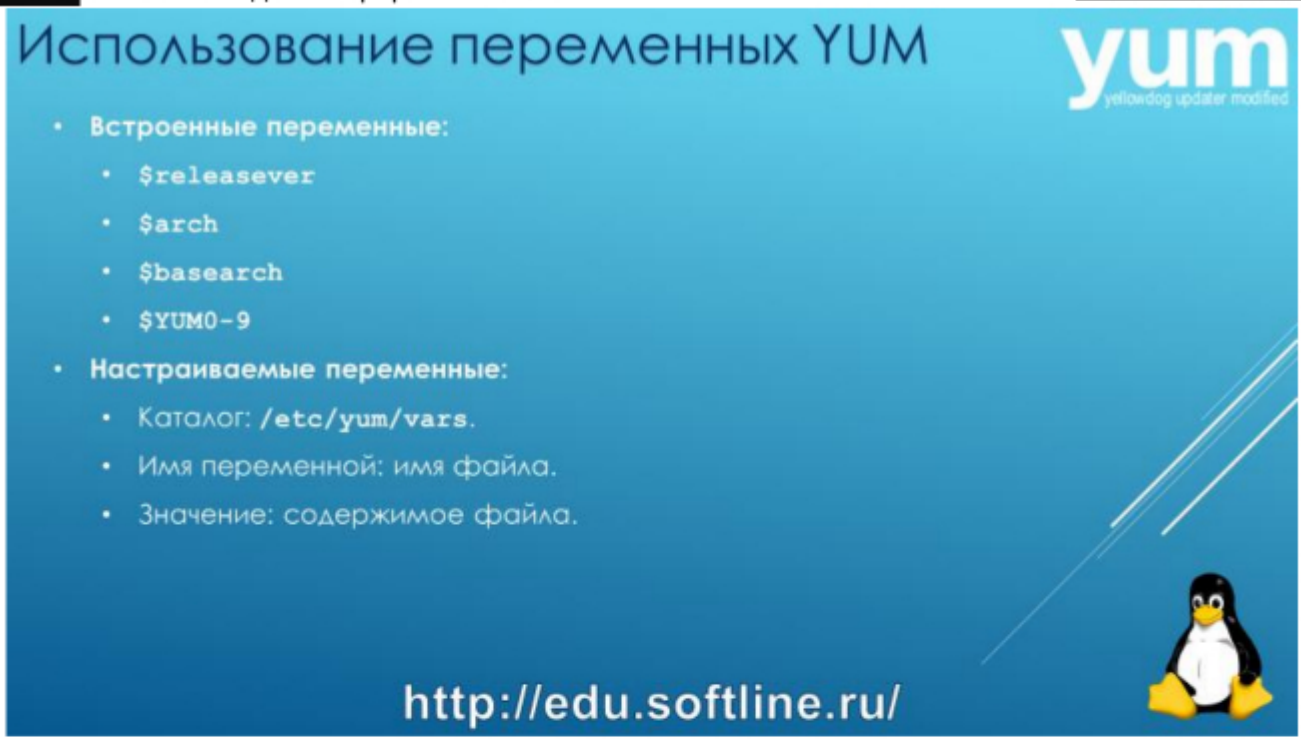
- auto (по умолчанию) – если доступна, используется параллельная загрузка. Yum автоматически отключает параллельную загрузку для репозиториев созданных при помощи плагинов.
- on – включение параллельной загрузки.
- off – отключение параллельной загрузки.

Существует множество других опций для секции **[***repository***]**, часть из них полностью повторяет опции секции **[main]**. Для получения полного списка доступных опций, обратитесь к секции **[repository] OPTIONS** страницы руководства **yum.conf**(5).

Пример конфигурационного файла **/etc/yum.repos.d/ CentOS-Base.repo**:

```
# CentOS-Base.repo
#
# The mirror system uses the connecting IP address of the client and the
# update status of each mirror to pick mirrors that are updated to and
# geographically close to the client.  You should use this for CentOS updates
# unless you are manually picking other mirrors.
#
# If the mirrorlist= does not work for you, as a fall back you can try the
# remarked out baseurl= line instead.
#
#

[base]
name=CentOS-$releasever - Base
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&r
epo=os&infra=$infra
#baseurl=http://mirror.centos.org/centos/$releasever/os/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7

#released updates
[updates]
name=CentOS-$releasever - Updates
```

```
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&r
epo=updates&infra=$infra
#baseurl=http://mirror.centos.org/centos/$releasever/updates/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7

#additional packages that may be useful
[extras]
name=CentOS-$releasever - Extras
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&r
epo=extras&infra=$infra
#baseurl=http://mirror.centos.org/centos/$releasever/extras/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7

#additional packages that extend functionality of existing packages
[centosplus]
name=CentOS-$releasever - Plus
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&r
epo=centosplus&infra=$infra
#baseurl=http://mirror.centos.org/centos/$releasever/centosplus/$basearch/
gpgcheck=1
enabled=0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
```



Вы можете использовать и ссылаться на следующие встроенные переменные в командах **yum** и всех конфигурационных файлах:

$releasever

Вы можете использовать эту переменную для ссылки на версию релиза CentOS. Yum применяет значение **$releasever** из строки **distroverpkg=value** конфигурационного файла **/etc/yum.conf**. Если строка не определена в файле **/etc/yum.conf**, в таком случае yum выбирает номер версии из пакета **centos-release-версия**, который предоставляет файлы **/etc/centos-release** и **/etc/redhat-release**.

$arch

Вы можете использовать эту переменную для ссылки на архитектуру процессора в том же виде как возвращает функция Python **os.uname()**. Корректные значения для **$arch** включают **i586**, **i686** и **x86_64**.

$basearch

Вы можете использовать эту переменную для ссылки на базовую архитектуру системы. Например, архитектуры **i586** и **i686** базируются на архитектуре **i386**, а AMD64 и Intel64 базируются на архитектуре **x86_64**.

$YUM0-9

Эти 10 переменных заменяются значениями переменных окружения оболочки с теми же именами. Если переменная определена в конфигурационном файле (например, в /etc/yum.conf) и переменная отсутствует в окружении оболочки, в таком случае переменная конфигурационного файла не заменяется.

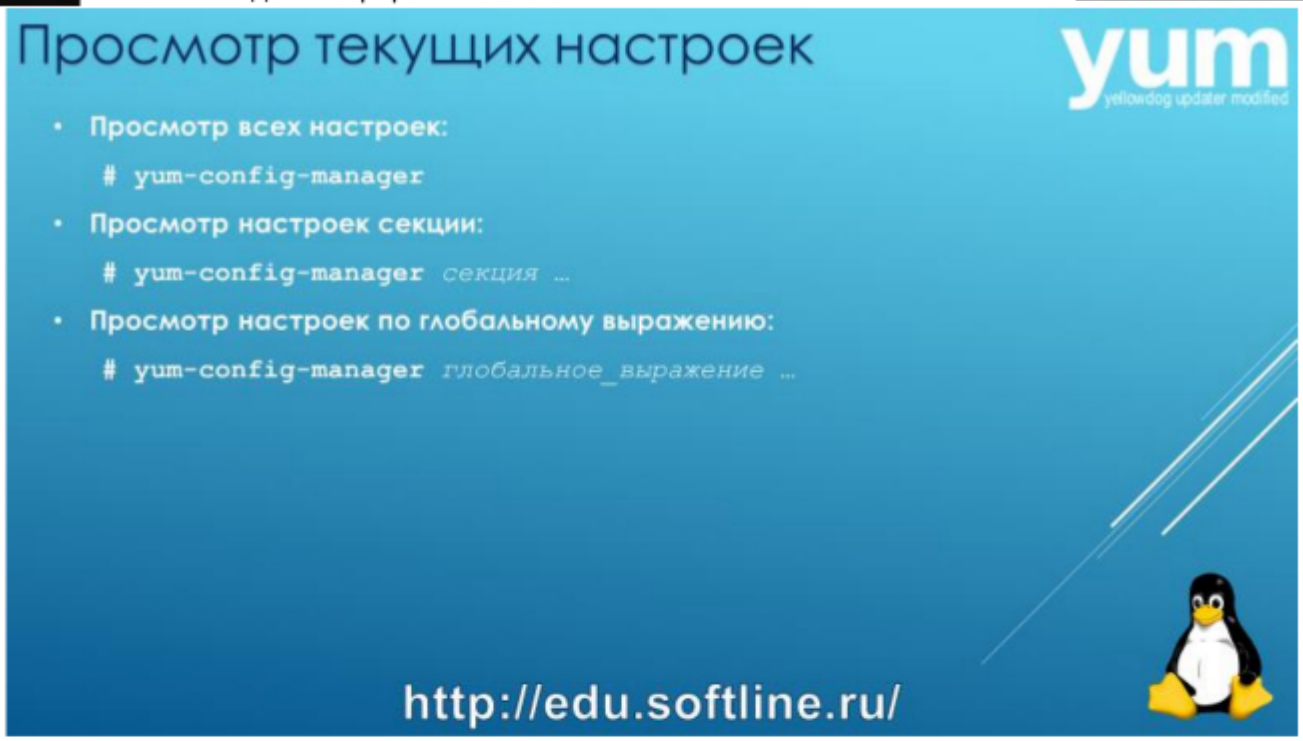
Для определения настраиваемой переменной или перезаписи существующей создайте файл с таким де именем (без знака **$**) в каталоге **/etc/yum/vars** и добавьте в него необходимое значение в первую строку.

Например, описание репозитория часто использует имя операционной системы. Для определения новой переменной **$osname**, создайте новый файл **/etc/yum/vars/osname** со значением «Community Enterprise Operating System» в первой строке:

```
# echo " Community Enterprise Operating System" > /etc/yum/vars/osname
```

Теперь вы можете использовать ее в файлах **.repo**:

```
name=$osname 7.2
```



Для отображения текущих значений глобальных опций yum (опции, указанные в секции [main] конфигурационного файла **/etc/yum.conf**) выполните команду **yum-config-manager**:

```
# yum-config-manager
```

Для отображения информации из другой конфигурационной секции или секций, используйте команду в следующем виде:

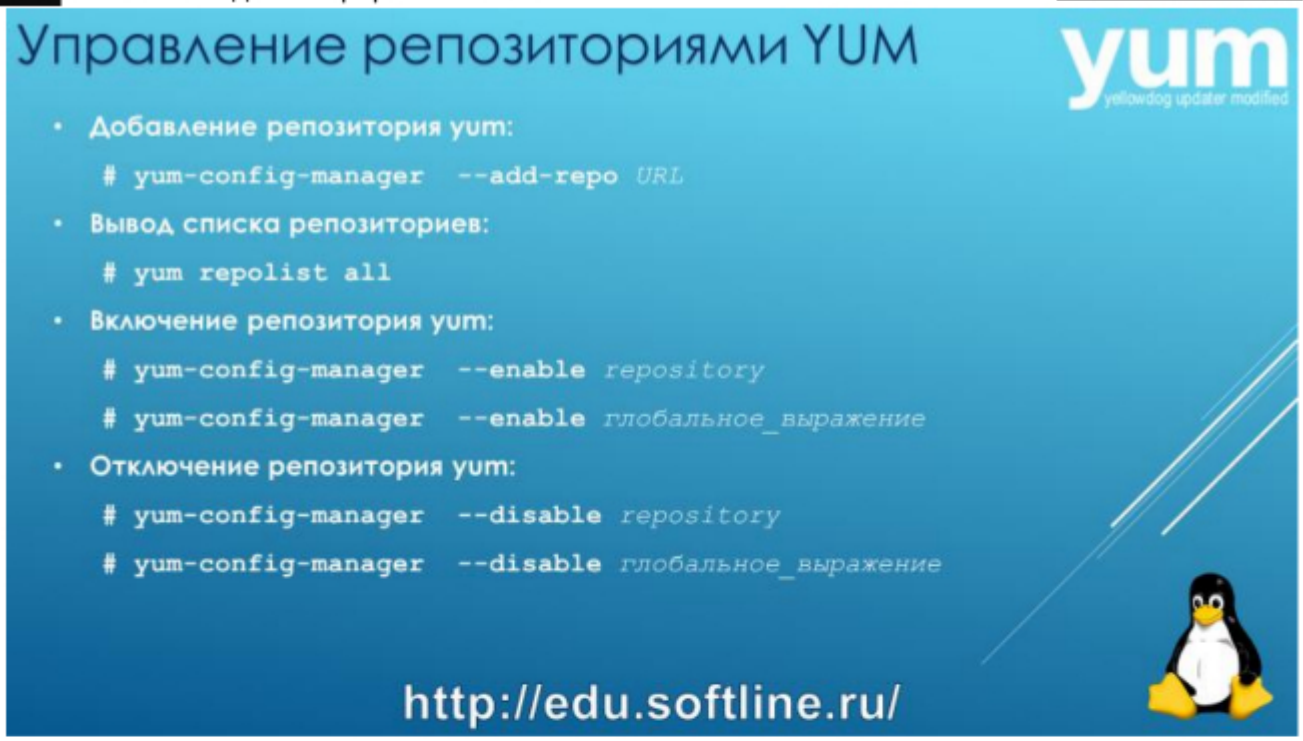
```
# yum-config-mamager секция…
```

Также можно использовать глобальные выражения (Glob Expression):

```
# yum-config-mamager глобальное_выражение…
```

Пример вывода настроек секции **[main]** конфигурационного файла **/etc/yum.conf**:

```
[root@MAINSERVER named]# yum-config-manager main
Loaded plugins: fastestmirror, langpacks
================================== main ==================================
[main]
alwaysprompt = True
assumeno = False
assumeyes = False
autocheck_running_kernel = True
bandwidth = 0
cache = 0
cachedir = /var/cache/yum/x86_64/7
check_config_file_age = True
clean_requirements_on_remove = False
color = auto
color_list_available_downgrade = dim,cyan
color_list_available_install = normal
color_list_available_reinstall = bold,underline,green
color_list_available_running_kernel = bold,underline
color_list_available_upgrade = bold,blue
…
```



Добавление репозитория yum.

Для определения нового репозитория Yum, вы можете добавить секцию **[***repository***]** в файл **/etc/yum.conf** или или файл **.repo** в каталог **/etc/yum.repos.d/**. Все файлы с расширением **.repo** в этом каталоге читаются yum, и это рекомендованный способ определения репозитория.

Как правило, репозитории yum предоставляют собственные файл **.repo**. Для добавления репозитория в систему и включения его используйте следующую команду:

 # yum-config-manager --add-repo *URL*

Пример добавления репозитория:

```
[root@c7-nat01 ~]# yum-config-manager --add-repo http://172.20.70.251/repo/centos72
Loaded plugins: fastestmirror, langpacks
adding repo from: http://172.20.70.251/repo/centos72

[172.20.70.251_repo_centos72]
name=added from: http://172.20.70.251/repo/centos72
baseurl=http://172.20.70.251/repo/centos72
enabled=1
```

Вывод списка репозиториев yum.

Для вы вода полного списка репозиториев воспользуйтесь командой

 # yum repolist all

Включение репозитория yum.

Для включения репозитория yum, воспользуйтесь следующей командой:

 # yum-config-manager --enable *repository…*

Также можно использовать глобальные выражения (Glob Expression):

 # yum-config-mamager --enable *глобальное_выражение…*

Пример включения репозитория:

```
[root@c7-nat01 ~]# yum-config-manager --enable c7-me\*
Loaded plugins: fastestmirror, langpacks
================================ repo: c7-media ================================
[c7-media]
async = True
bandwidth = 0
base_persistdir = /var/lib/yum/repos/x86_64/7
baseurl = file:///media/CentOS/,
   file:///media/cdrom/,
   file:///media/cdrecorder/
cache = 0
cachedir = /var/cache/yum/x86_64/7/c7-media
check_config_file_age = True
cost = 1000
deltarpm_metadata_percentage = 100
deltarpm_percentage =
enabled = 1
…
```

После успешного включения репозитория, команда **yum-config-manager** отображает его текущие параметры.

Отключение репозитория yum.

Для отключения репозитория yum, воспользуйтесь следующей командой:

```
# yum-config-manager --disable repository…
```

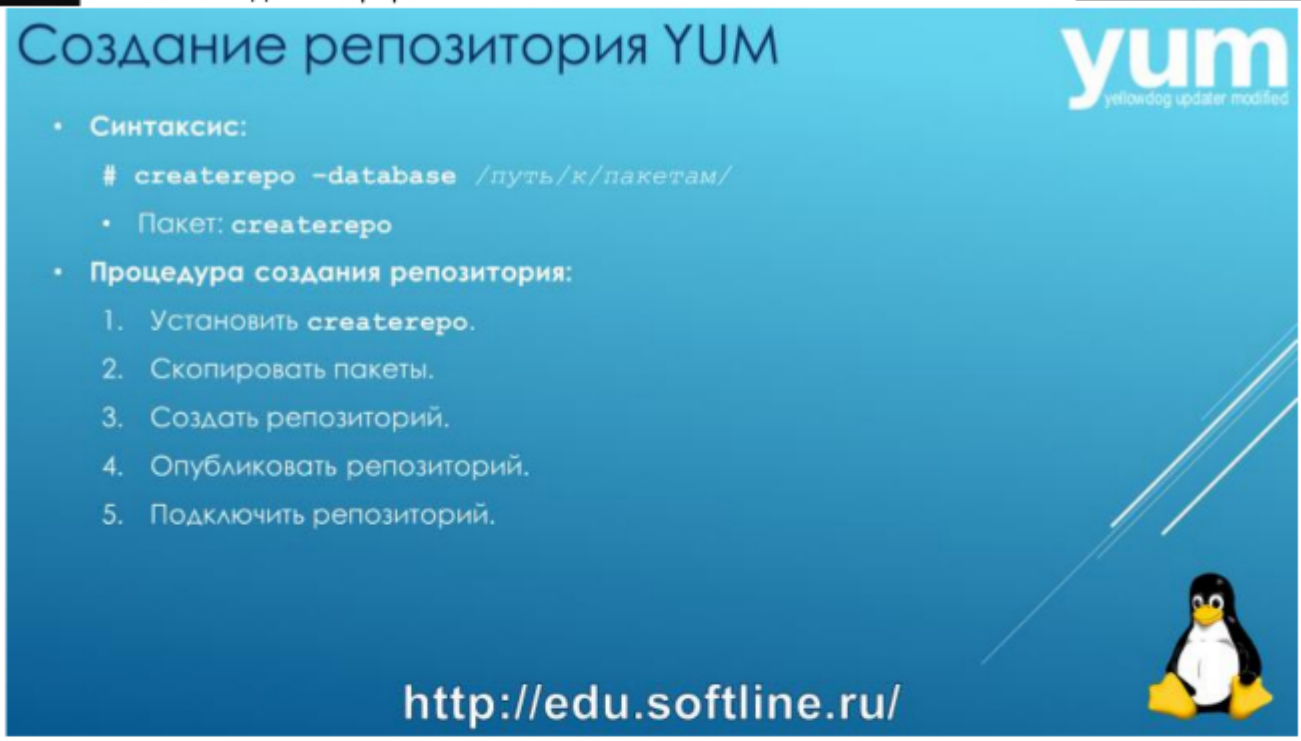
Также можно использовать глобальные выражения (Glob Expression):

```
# yum-config-mamager --disable глобальное_выражение…
```

Пример отключения репозитория:

```
[root@c7-nat01 ~]# yum-config-manager --disable c7-me\*
Loaded plugins: fastestmirror, langpacks
================================ repo: c7-media ================================
[c7-media]
async = True
bandwidth = 0
base_persistdir = /var/lib/yum/repos/x86_64/7
baseurl = file:///media/CentOS/,
   file:///media/cdrom/,
   file:///media/cdrecorder/
cache = 0
cachedir = /var/cache/yum/x86_64/7/c7-media
check_config_file_age = True
cost = 1000
deltarpm_metadata_percentage = 100
deltarpm_percentage =
enabled = 0
…
```

После успешного отключения репозитория, команда **yum-config-manager** отображает его текущие параметры.



Для настройки репозитория yum воспользуйтесь следующими командами:

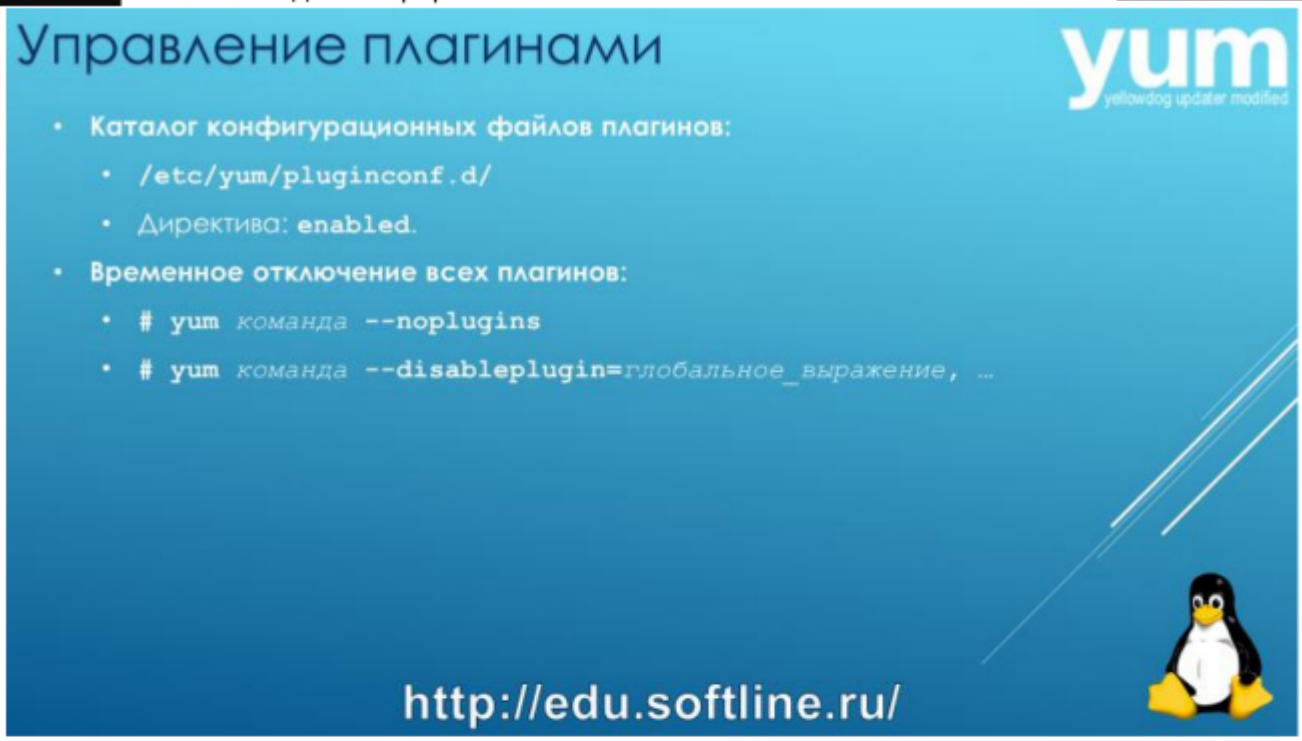
1. Установите пакет **createrepo**:

```
# yum install createrepo
```

2. Скопируйте все необходимые пакеты в единое расположение, например **/mnt/1repo**.
3. Создайте репозиторий в каталоге:

```
# createrepo --database /mnt/1repo
```

Это создаст все необходимые файлы в каталоге, в том числе базу данных sqlite для ускорения операций yum.



Yum предоставляет плагины, которые расширяют и улучшают его возможности. Набор необходимых плагинов устанавливается по умолчанию. Yum всегда информирует о том, какие плагины используются во время любой операции, например:

```
[root@c7-nat01 ~]# yum info yum
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.nsu.ru
```

Обратите внимание, что имена, указанные после слов **Loaded plugins**, это имена, которые можно использовать в командах, например **--disableplugin=***plugin_name*.

Включение, настройка и отключение плагинов Yum.

Для того чтобы включить плагины, убедитесь, что в строке **plugins=** в секции **[main]** конфигурационного файла **/etc/yum.conf** установлено значение **1**.

```
plugins=1
```

Для отключения всех плагинов выставьте значение **plugins=0**.



ПРИМЕЧАНИЕ:

Не рекомендуется отключать все плагины, так как некоторые плагины предоставляют важные службы yum. Глобальное отключение как правило используется в целях диагностики и выявления неисправностей.

Для каждого установленного пллагина имеется свой собственный конфигурационный файл в каталоге **/etc/yum/pluginconf.d/**. Вы можете установить специальные параметры плагинов в этих файлах. Пример конфигурационного файла плагина **aliases – aliases.conf**

```
[main]
enabled=1
```

Подобно конфигурационному файлу **/etc/yum.conf**, конфигурационные файлы плагинов содержит секцию **[main]**, в которой присутствует опция **enabled=**, позволяющая отключить плагин. Если данная опция отсутствует в файле вы можете добавить ее вручную.

Если плагины отключены глобально, в секции **[main]** конфигурационного файла **/etc/yum.conf**, значение опции **enabled=** в конфигурационных файлах плагинов игнорируется.

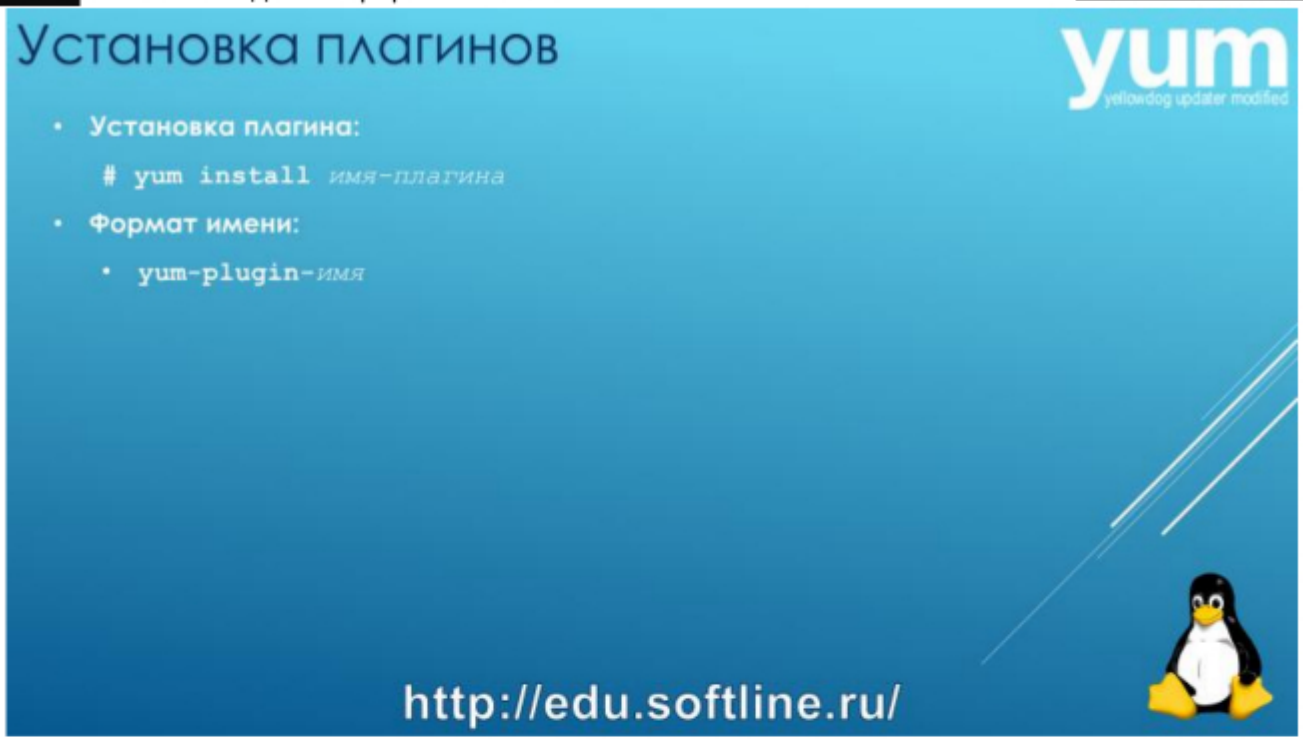
Для разового отключения всех плагинов можно использовать опцию **--noplugins** команды **yum**.

Для разового отключения о дного или нескольких плагинов воспользуйтесь опцией **--disableplugin** команды **yum**. Например, для отключения плагина aliases, в процессе обновления, воспользуйтесь следующей командой:

```
# yum update --disableplugin=aliases
```

Для указания нескольких плагинов, их нужно указывать через запятую, также поддерживаются глобальные выражения (Glob Expressions):

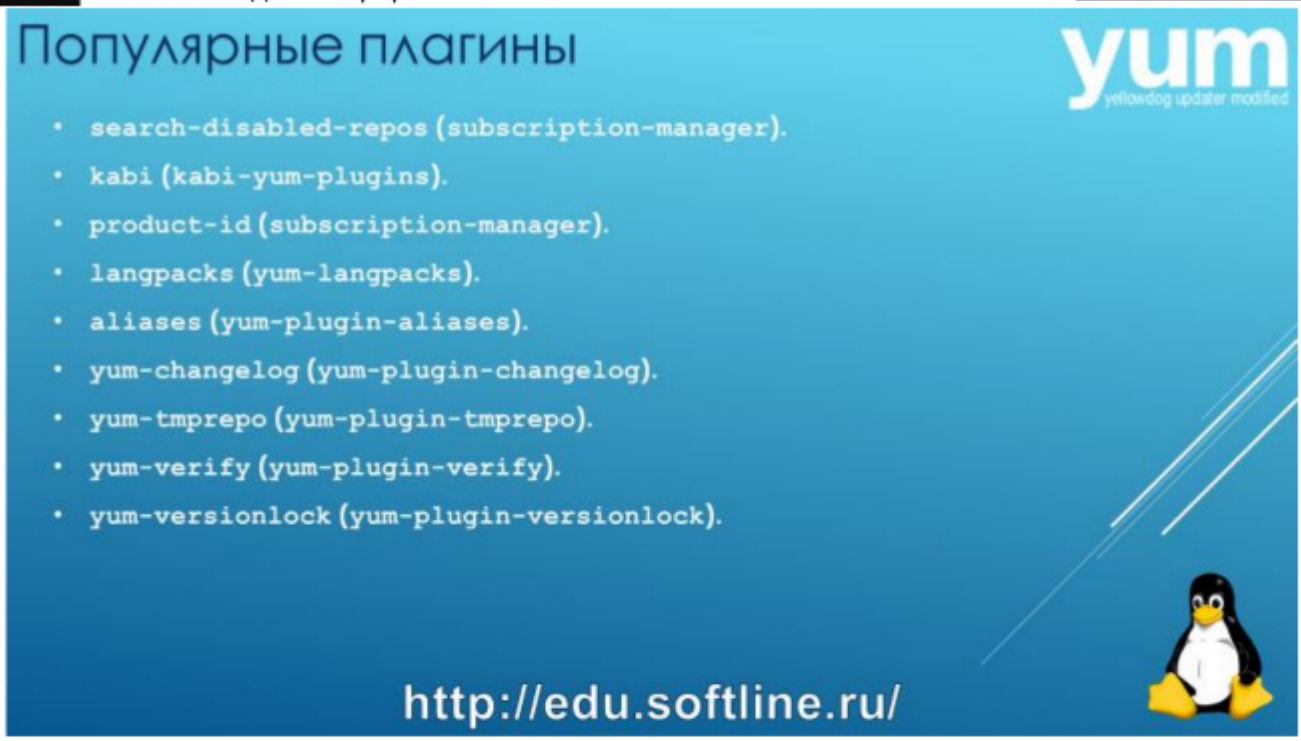
```
# yum update --disableplugin=aliases, lang*
```



Плагины yum распространяются в пакетах, соответствующих следующему соглашению именования: **yum-plugin-***имя-плагина*, но не всегда. Например, пакет предоставляющий плагин **kabi** называется **kabi-yum-plugins**.

Пример установки плагина **aliases**:

```
# yum install yum-plugin-aliases
```



Следующий список предоставляет описание и инструкции по использованию популярных плагинов yum. В скобках указаны имена пакетов, предоставляющих плагин.

search-disabled-repos (**subscription-manager**)

Плагин **search-disabled-repos** позволяет временно или на постоянной основе включить отключенный репозиторий, чтобы помочь с разрешением зависимостей. Когда этот плагин включен и yum не может завершить установку из-за неразрешенных зависимостей, плагин временно включает отключенные репозитории и повторно пробует выполнить операцию. Если установка проходит успешно, yum предлагает оставить включенным использованный репозиторий. Обратите внимание, что плагин работает только с репозиториями, управляемыми через **subscription-manager** и не работает с настраиваемыми репозиторриями.

ПРИМЕЧАНИЕ:
*Если yum выполняется с опциями **--assumeyes** или **-y**, или директива **assumeyes** включена в **/etc/yum.conf** плагин включает репозитории временно и на постоянной основе без запросов подтверждения. Это может привести к проблемам, связанным с включением репозиториев, которые вы не хотите использовать.*

Поддерживаемые директивы конфигурационного файла **search-disabled-repos.conf**:

Директивы	Описание
enabled=*значение*	Позволяет включать или отключать плагин, поддерживаемые значения: **1** (включен) и **0** (отключен). По умолчанию включен.
notify_only=*значение*	Позволяет ограничить функционал только до уведомлений, поддерживаемые значения: 1 (только уведомления) и 0 (позволяет управлять поведением yum). По умолчанию только уведомляет.
ignored_repos=*репозитории*	Позволяет указать репозитории, которые не будут включены плагином.

kabi (**kabi-yum-plugins**)

Плагин **kabi** проверяет пакеты обновления драйверов на соответствие официальному прикладному бинарному интерфейсу ядра (Kernel Application Binary Interface, KABI). При включенном плагине, когда пользователь попытается установить пакет, который использует символы ядра, не представленные в списке разрешенных, сообщение с предупреждением будет записано в системный журнал. Дополнительно, если настроить плагин на запуск в принудительном режиме это не позволит таким пакетам устанавливаться.

Для настройки плагина **kabi**, отредактируйте конфигурационный файл, расположенный в файле **/etc/yum/pluginconf.d/kabi.conf**. Список директив, которые можно использовать в секции **[main]** представлен в таблице:

Директивы	Описание
enabled=*значение*	Позволяет включать или отключать плагин, поддерживаемые значения: **1** (включен) и **0** (отключен). Когда установлен, по умолчанию включен.
whitelists=*каталог*	Позволяет указать каталог, который содержит файлы с поддерживаемыми символами ядра. По умолчанию, плагин **kabi** использует файлы, предоставляемые пакетом **kernel-abi-whitelists** (каталог **/usr/lib/modules/kabi-rhel70/**).
enforce=*значение*	Позволяет включить или отключить принудительный режим, поддерживаемые значения: **1** (включен) и **0** (отключен). По умолчанию данная опция закомментирована и плагин **kabi** отображает только предупреждающее сообщение.

product-id (**subscription-manager**)

Плагин **product-id** управляет сертификатами идентификации продуктов, устанавливаемыми из Content Delivery Network. Плагин **product-id** установлен по умолчанию.

langpacks (**yum-langpacks**)

Плагин **langpacks** используется для поиска языковых пакетов выбранных языков для каждого устанавливаемого пакета. Плагин **langpacks** установлен по умолчанию.

aliases (**yum-plugin-aliases**)

Плагин **aliases** добавляет опцию командной строки **alias**, которая позволяет настройку и использование псевдонимов для команд **yum**.

yum-changelog (**yum-plugin-changelog**)

Плагин **aliases** добавляет опцию командной строки **--changelog**, которая позволяет просматривать журнал изменения пакета до и после обновления.

yum-tmprepo (**yum-plugin-tmprepo**)

Плагин **yum-tmprepo** добавляет опцию командной строки **--tmprepo**, которая берет URL файла репозитория, загружает и включает его только на одну транзакцию. Данный плагин пытается

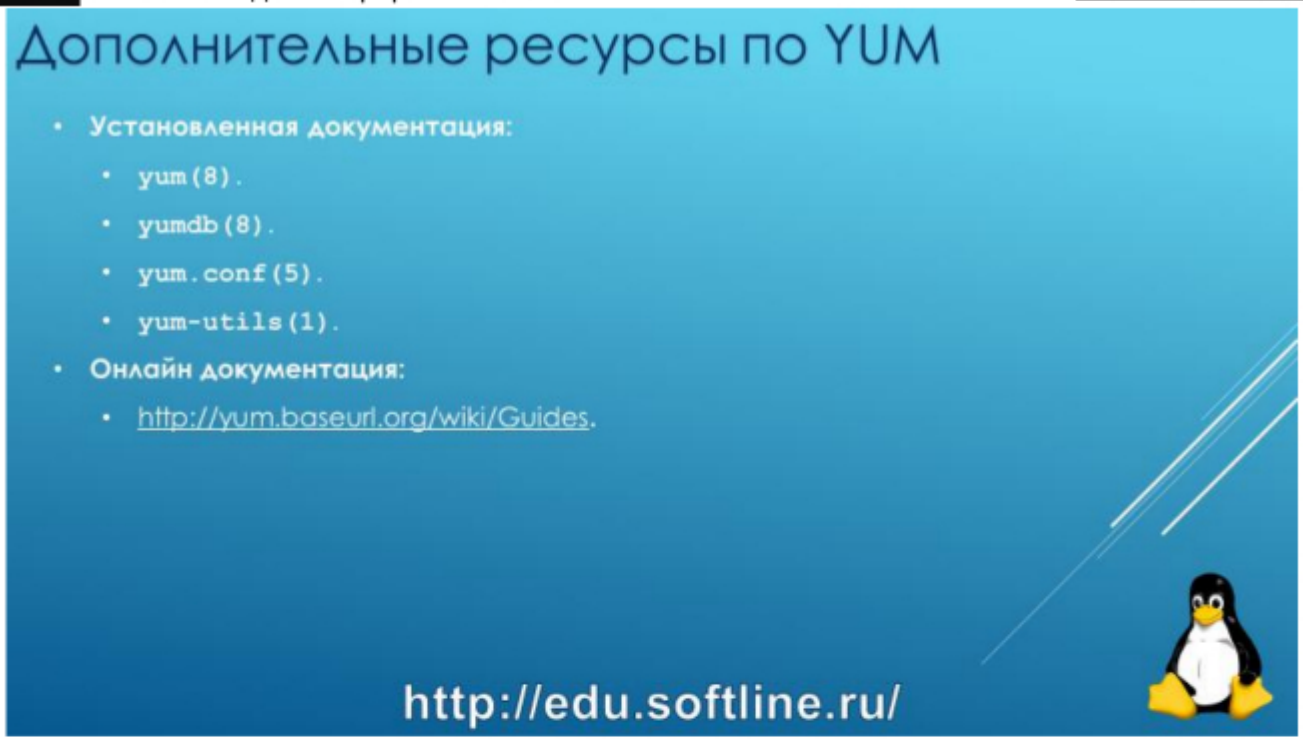
обеспечить безопасное временное использование репозитория. По умолчанию не позволяет отключать проверку GPG.

`yum-verify` (**`yum-plugin-verify`**)

Плагин **`yum-verify`** добавляет опции командной строки **`verify`**, **`verify-rpm`** и **`verfy-all`** для просмотра данных проверки на системе.

`yum-versionlock` (**`yum-plugin-versionlock`**)

Плагин **`yum-versionlock`** исключает прочие версии выбранного пакета, что позволяет защитить пакеты от обновления новыми версиями. При помощи опции командной строки **`versionlock`** можно просматривать и редактировать список заблокированных пакетов.



Для получения дополнительной информации по управлению пакетами в CentOS (RHEL) обратитесь к следующим ресурсам.

Установленная документация:

yum(8) – страница руководства утилиты командной строки **yum**, предоставляет полный список поддерживаемых опций и команд.

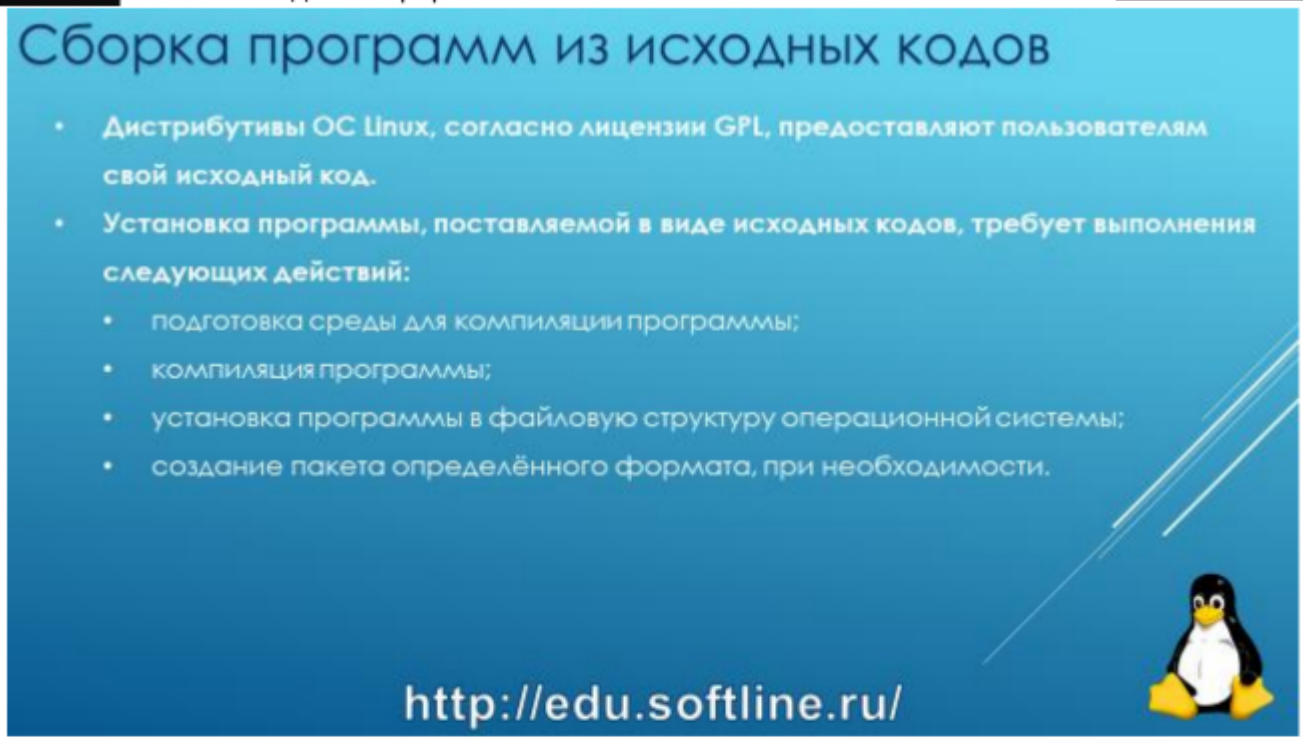
yumdb(8) – страница руководства утилиты командной строки **yumdb**, описывает использование утилиты для запросов и, при необходимости, изменения базы данных.

yum.conf(5) – страница руководства конфигурационного файла **yum.conf**, содержащая все доступные опции настройки yum.

yum-utils(1) – страница руководства, содержащая краткое описание дополнительных утилит управления настройками yum, управления репозиториями и работы с базой данных yum.

Онлайн документация:

Руководства yum (http://yum.baseurl.org/wiki/Guides).



Операционная система UNIX появилась на свет одновременно с языком программирования C, более того, язык C был разработан для создания ОС UNIX. В последнее время, всё чаще используется и C++.

Лицензия GPL обязывает разработчиков предоставлять исходные коды своих программных продуктов, а соответственно, используя операционную систему Linux, рано или поздно придётся столкнуться с установкой программы, поставляемой в виде исходных кодов.

Для выполнения процедуры сборки программы используется программа gcc (GNU Compiler Collection). На сегодняшний день, GCC объединяет компиляторы для различных языков программирования: C, C++, Fortran, Ada, Java и т.д. GCC состоит из 4-х основных компонент:

- Препроцессор модифицирует код программы для подготовки к компиляции.
- Компилятор обрабатывает исходный код программы и преобразует его в код на языке ассемблера.
- Ассемблер формирует объектный код.
- Компоновщик (линкер) производит сборку всех частей в конечные файлы.

Для создания программы, программист может использовать стандартные библиотеки, предоставляющие необходимый уровень абстракции. Например, основная библиотека **`libc`**, предоставляет функции для работы с памятью, устройствами ввода/вывода. Библиотеки создаются программистами и могут быть статическими, разделяемыми и динамическими:

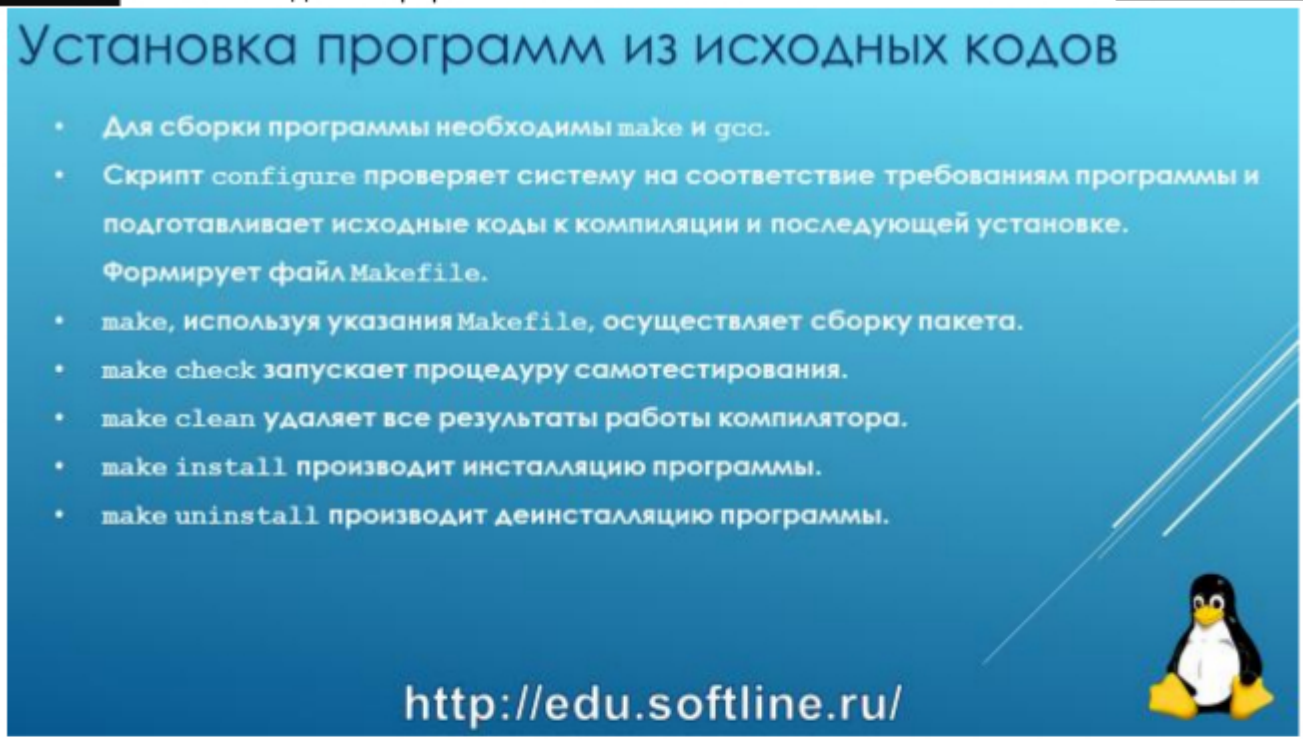
- Статическая библиотека – код статической библиотеки, при компиляции, встраивается в код программы, тем самым, программа становится самодостаточной, имея в себе все необходимые для работы функции. Имя такой библиотеки имеет расширение "**`.a`**".
- Разделяемая библиотека – код разделяемой библиотеки не встраивается в код программы, загружаясь в память одновременно с запуском программы, в свою очередь, программа получает доступ к необходимым функциям.

- Динамическая библиотека – разновидность разделяемой библиотеки, но она загружается в память только тогда, когда из программы поступит вызов какой-либо функции, находящейся в динамической библиотеке. Имя такой библиотеки имеет расширение "**.so**".

Как правило, библиотеки располагаются в каталогах: **/lib**, **/usr/lib**, **/usr/local/lib**. Про все библиотеки, находящиеся в файловой системе, должна знать программа **ldconfig**, которая производит поиск библиотек во всех каталогах, указанных в файле **/etc/ld.so.conf** и заносит их в свой кэш, впоследствии, компоновщик обращается к ldconfig за информацией о необходимых библиотеках.

Обычно программы состоят из множества компонент и при сборке таких продуктов, очень важно отследить, какие файлы уже были откомпилированы, а какие нет, какие файлы необходимо подключать и так далее.

Чтобы управлять сборкой программ была создана утилита **make**, которая определяет необходимость сборки того или иного элемента, основываясь на дате последней модификации. Утилита производит действия, основываясь на файле **Makefile**, в котором есть описание элементов программы, указывающих, от каких элементов они зависят.



Чтобы установить программу, необходимо получить её с сайта разработчика, либо из любого публичного хранилища. Исходные коды поставляются в виде архивных файлов, сжатых программой **gzip** или **bzip2**. Для начала установки программы, распакуйте архив и перейдите в появившийся каталог:

```
# tar xzpf ./mc-4.6.1.tar.gz
# cd mc-4.6.1
```

Теперь необходимо выполнить процедуру конфигурирования в соответствии с вашей системой. Скрипт configure проверит данные системы и соответствие требованиям собираемого пакета. С помощью configure, пользователь может выполнить настройку программы, включить или отключить функции, указать пути для установки, поиска библиотек и т.д. Для просмотра всех возможных параметров, выполните скрипт configure:

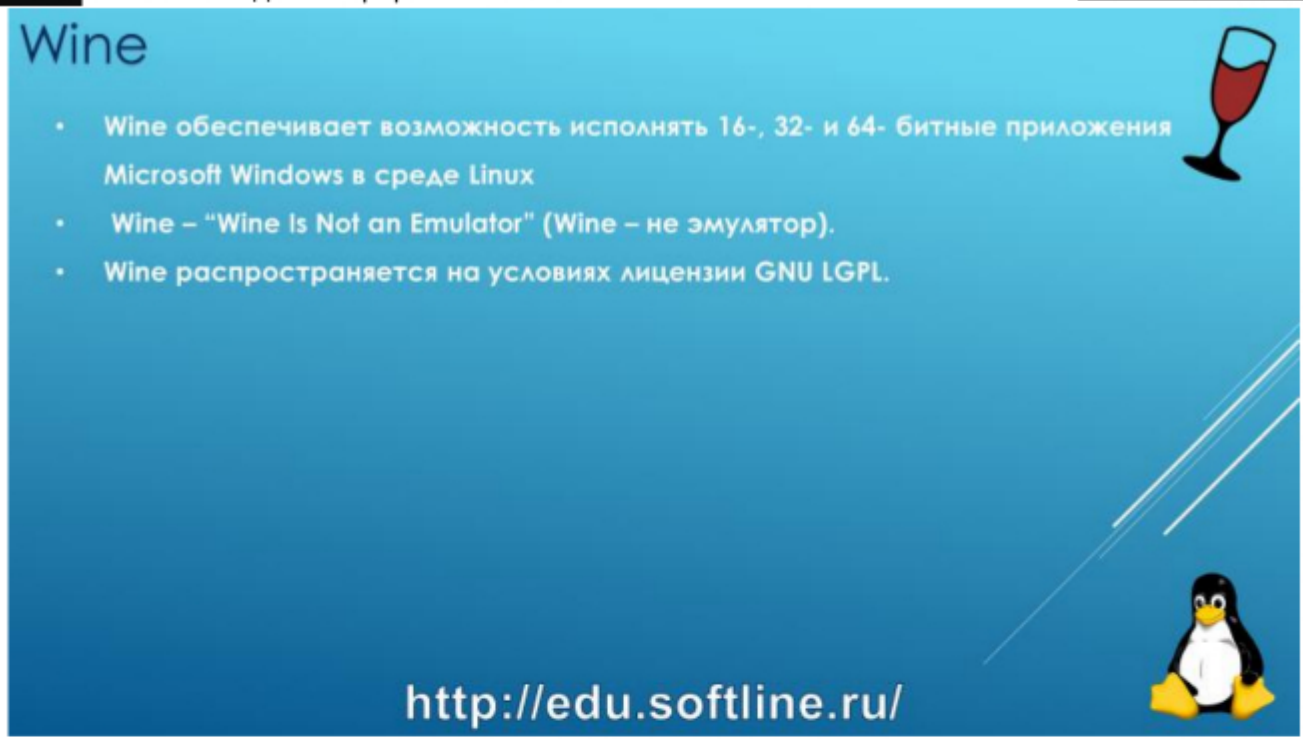
```
# configure –help
```

Одним из наиболее часто используемых параметров, является параметр "**--prefix**", который указывает каталог, в который необходимо произвести установку, по умолчанию, как правило, установка произведётся в **/usr/local**. Давайте соберём программу **mc**, запустим скрипт configure и дождёмся завершения выполнения, если ошибок не будет, то можно приступать к сборке:

```
# configure
# make
# make install
```

Всё, программа установлена в каталог **/usr/local** и вы можете её запустить, дав команду:

```
# mc
```



Wine – свободное программное обеспечение, позволяющее пользователям UNIX-подобных систем архитектуры x86 (и других архитектур, при наличии совместимости, например, AMD64) исполнять 16-, 32- и 64- битные приложения Microsoft Windows (64-битные приложения находятся в стадии ранней реализации). W

Wine также предоставляет программистам библиотеку программ Winelib, при помощи которой они могут компилировать Windows-приложения для портирования их в UNIX-подобные системы.

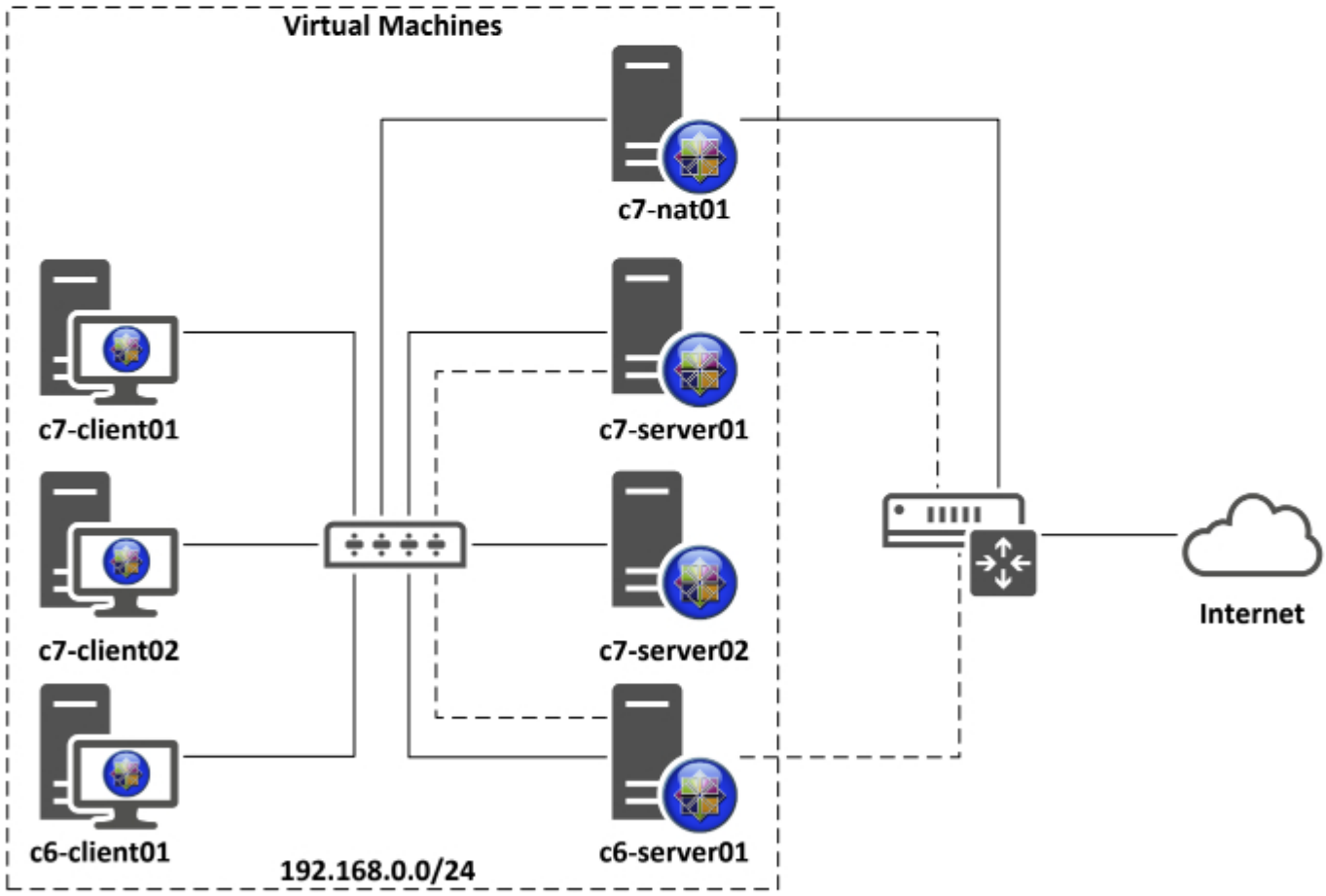
Wine распространяется на условиях лицензии GNU LGPL.

Лабораторная работа.

Лабораторная работа 8

- **Упражнения:**
  - Работа с пакетным менеджером RPM.
  - Сборка программы из исходных кодов.
  - Управление репозиториями RPM-пакетов при помощи YUM.
- **Необходимые машины:**
  - `c7-nat01`
- **Учетные данные:**
  - Пользователь: `root`.
  - Пароль: `Pa$$w0rd`.

http://edu.softline.ru/





Упражнение 1 – Работа с пакетным менеджером RPM.

1. Запустите виртуальную машину **c7-server01**.
2. При помощи команды **rpm** выясните сколько на системе установлено пакетов?
3. При помощи команды **rpm** проверьте, установлен ли пакет **openssh** на системе?
4. При помощи команды **rpm** выясните сколько файлов содержится в пакете **openssh**?
5. При помощи команды **rpm** выясните какой/какие конфигурационные файлы содержится в пакете **openssh**?
6. При помощи команды **rpm** выясните из каких пакетов были установлены следующие файлы:
   - **/usr/bin/rpm**?
   - **/usr/bin/yum**?
   - **/etc/resolv.conf**?
   - **/etc/ssh/ssh_config**?
   - **/etc/ssh/sshd_config**?
7. Загрузите пакеты **openssh** и **zziplib**.

```
# wget http://URL/openssh-7.4p1-16.el7.x86_64.rpm
# wget http://URL/zziplib-0.13.62-8.fc26.x86_64.rpm
```



ПРИМЕЧАНИЕ:
Ссылку для загрузки уточните у инструктора.

8. При помощи команды **rpm** сравните содержимое загруженного пакета **openssh** с установленным на системе?
   - Какие файлы в пакете отличаются размером от установленных на системе?
9. При помощи команды **rpm** получите описание к загруженному пакету **zziplib**.
   - Кто выпустил пакет?
   - Под какой лицензией выпущен пакет?
   - Под какую архитектуру выпущен пакет?
10. При помощи команды **rpm** установите загруженный пакет **zziplib**.
11. Убедитесь, что пакет **zziplib** успешно установлен.
12. При помощи команды **rpm** удалите пакет **zziplib**.



Ответы к упражнению 1 – Работа с пакетным менеджером RPM.

1. Запустите виртуальную машину **c7-server01**.
2. При помощи команды **rpm** выясните сколько на системе установлено пакетов?

```
# rpm -qa | wc -l
```

3. При помощи команды **rpm** проверьте, установлен ли пакет **openssh** на системе?

```
# rpm -q openssh
```

4. При помощи команды **rpm** выясните сколько файлов содержится в пакете **openssh**?

```
# rpm -q openssh -l |wc -l
```

5. При помощи команды **rpm** выясните какой/какие конфигурационные файлы содержится в пакете **openssh**?

```
# rpm -q openssh -c
```

6. При помощи команды **rpm** выясните из каких пакетов были установлены следующие файлы:
   - **/usr/bin/rpm**?

```
# rpm -qf /usr/bin/rpm
```

   - **/usr/bin/yum**?

```
# rpm -qf /usr/bin/yum
```

   - **/etc/resolv.conf**?

```
# rpm -qf /etc/resolv.conf
```

   - **/etc/ssh/ssh_config**?

```
# rpm -qf /etc/ssh/ssh_config
```

   - **/etc/ssh/sshd_config**?

```
# rpm -qf /etc/ssh/sshd_config
```

7. Загрузите пакеты **openssh** и **zziplib**.

```
# wget http://URL/openssh-7.4p1-16.el7.x86_64.rpm
# wget http://URL/zziplib-0.13.62-8.fc26.x86_64.rpm
```



ПРИМЕЧАНИЕ:

Ссылку для загрузки уточните у инструктора.

8. При помощи команды **rpm** сравните содержимое загруженного пакета **openssh** с установленным на системе?

```
# rpm -Vp openssh-7.4p1-16.el7.x86_64.rpm | grep ^S
```

   - Какие файлы в пакете отличаются размером от установленных на системе?

```
# rpm -Vp openssh-7.4p1-16.el7.x86_64.rpm | grep ^S
```

9. При помощи команды **rpm** получите описание к загруженному пакету **zziplib**.

```
# rpm -qip zziplib-0.13.62-8.fc26.x86_64.rpm
```

   - Кто выпустил пакет?

```
# rpm -qip zziplib-0.13.62-8.fc26.x86_64.rpm | grep ^Vendor
```

   - Под какой лицензией выпущен пакет?

```
# rpm -qip zziplib-0.13.62-8.fc26.x86_64.rpm | grep ^License
```

   - Под какую архитектуру выпущен пакет?

```
# rpm -qip zziplib-0.13.62-8.fc26.x86_64.rpm | grep ^Arch
```

10. При помощи команды **rpm** установите загруженный пакет **zziplib**.

```
# rpm -ihv zziplib-0.13.62-8.fc26.x86_64.rpm
```

11. Убедитесь, что пакет **zziplib** успешно установлен.

```
# rpm -q zziplib
```

12. При помощи команды **rpm** удалите пакет **zziplib**.

```
# rpm -ehv zziplib
```



Упражнение 2 – Сборка программ из исходных кодов.

1. Запустите виртуальную машину **c7-server01**.
2. Установите GNU Compiler Collection (**`gcc`**) для компиляции программ, написанных на C, C++, Objective-C, Java и других языках.
3. Загрузите архив с исходными кодами программы **`sipcalc`**.

ПРИМЕЧАНИЕ:
Ссылку для загрузки уточните у инструктора.

4. Распакуйте архив с исходными кодами **`sipcalc`**.
5. Перейдите в каталог с исходными кодами.
6. Выполните настройку переменных и создайте **`Makefiles`** при помощи скрипта **`configure`**.
7. Произведите сборку бинарных файлов программы.
8. Установите программу.
9. При помощи страницы руководства **`sipcalc(1)`** познакомьтесь с возможностями программы.
10. Проверьте работоспособность программы.
11. Удалите программу.
12. Установите программу в каталог **`/usr/local/sipcalc`**.



Ответы к упражнению 2 – Сборка программ из исходных кодов.

1. Запустите виртуальную машину **c7-server01**.
2. Установите GNU Compiler Collection (**gcc**) для компиляции программ, написанных на C, C++, Objective-C, Java и других языках.

```
# yum install gcc -y
```

3. Загрузите архив с исходными кодами программы **sipcalc**.

```
# wget http://URL/files/sipcalc-1.1.4.tar.gz
```

ПРИМЕЧАНИЕ:
Ссылку для загрузки уточните у инструктора.

4. Распакуйте архив с исходными кодами **sipcalc**.

```
# tar xzf sipcalc-1.1.4.tar.gz
```

5. Перейдите в каталог с исходными кодами.

```
# cd sipcalc-1.1.4/
```

6. Выполните настройку переменных и создайте **Makefiles** при помощи скрипта **configure**.

```
# ./configure
```

7. Произведите сборку бинарных файлов программы.

```
# make
```

8. Установите программу.

```
# make install
```

9. При помощи страницы руководства **sipcalc(1)** познакомьтесь с возможностями программы.

```
# man sipcalc
```

10. Проверьте работоспособность программы.

```
# sipcalc -a 192.168.0.0/24
```

11. Удалите программу.

```
# make uninstall
```

12. Установите программу в каталог **/usr/local/sipcalc**.

```
./configure -prefix=/usr/local/sipcalc && make && make install
```



Упражнение – 3: Управление репозиториями RPM-пакетов при помощи YUM.

Задание – 1: Подготовка сервера.

1. Подключите ISO-образ с дистрибутивом CentOS к машине **c7-server01** и смонтируйте образ в каталог **/mnt**.



ПРИМЕЧАНИЕ:
Уточните расположение ISO-образа у инструктора.

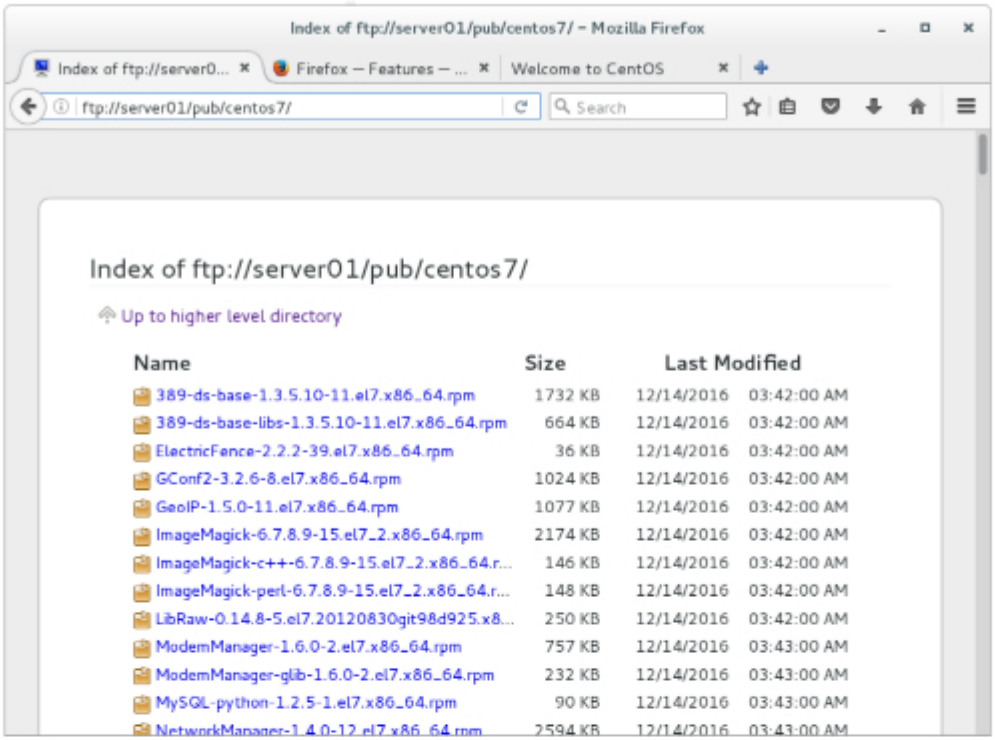
2. Установите пакет FTP-сервера (**vsftpd**) из каталога **/mnt/Packages**.
3. Настройте автоматический запуск службы FTP-сервера (**vsftpd**).
4. Добавьте исключение для службы FTP в FirewallD (21/TCP, 22/TCP).
5. Установите пакет **createrepo** и необходимые для него зависимости.

Задание – 2: Создание локального репозитория.

1. Создайте каталог для локального репозитория **/var/ftp/pub/centos7** на машине **c7-server01**.
2. Скопируйте все RPM-пакеты из каталога **/mnt/Packages/** в каталог локального репозитория.
3. Создайте репозиторий YUM в каталоге **/var/ftp/pub/centos7/**.
4. Переместите все файлы из каталога **/etc/yum.repos.d/** в домашнюю папку пользователя root.
5. Создайте конфигурационный файл локального репозитория, со следующими параметрами:
   - Конфигурационный файл: **/etc/yum.repos.d/centos7.repo**.
   - Идентификатор репозитория: `c7-local`.
   - Имя репозиттория: `Course LL-100`.
   - Путь к репозиторию: `file:///var/ftp/pub/centos7`.
   - Проверка ключей репозитория: отключена (`0`).
6. Убедитесь в работоспособности локального репозитония.
7. Установите любой пакет с зависимостями из репозитория (например, редактор Emacs).

Задание – 3: Настройка удаленного репозитория.

1. На машине c7-client01 проверьте доступ к FTP (ftp://server01/pub/centos7).



2. Отключите все репозитории YUM на системе.
3. Подключите вновь созданный репозиторий.
4. Убедитесь в работоспособности локального репозитония.
5. Установите любой пакет с зависимостями из репозитория (например, редактор Emacs).

Ответы к упражнению – 3: Управление репозиториями RPM-пакетов при помощи YUM.

Задание – 1: Подготовка сервера.

1. Подключите ISO-образ с дистрибутивом CentOS к машине **c7-server01** и смонтируйте образ в каталог **/mnt**.

```
# mount /dev/cdrom /mnt/
```

ПРИМЕЧАНИЕ:
Уточните расположение ISO-образа у инструктора.

2. Установите пакет FTP-сервера (**vsftpd**) из каталога **/mnt/Packages**.

```
# cd /mnt/Packages/
# rpm -ihv vsftpd-3.0.2-21.el7.x86_64.rpm
```

3. Настройте автоматический запуск службы FTP-сервера (**vsftpd**).

```
# systemctl enable vsftpd
# systemctl start vsftpd
```

4. Добавьте исключение для службы FTP в FirewallD (21/TCP, 22/TCP).

```
# firewall-cmd --add-service=ftp
# firewall-cmd --add-service=ftp --permanent
```

5. Установите пакет **createrepo** и необходимые для него зависимости.

```
# rpm -ihv createrepo-0.9.9-26.el7.noarch.rpm
# rpm -ihv deltarpm-3.6-3.el7.x86_64.rpm
# rpm -ihv python-deltarpm-3.6-3.el7.x86_64.rpm
```

Задание – 2: Создание локального репозитория.

1. Создайте каталог для локального репозитория **/var/ftp/pub/centos7** на машине **c7-server01**.

```
# mkdir /var/ftp/pub/centos7
```

2. Скопируйте все RPM-пакеты из каталога **/mnt/Packages/** в каталог локального репозитория.

```
# cp /mnt/Packages/*.rpm /var/ftp/pub/centos7/
```

3. Создайте репозиторий YUM в каталоге **/var/ftp/pub/centos7/**.

```
# createrepo /var/ftp/pub/centos7/
```

4. Переместите все файлы из каталога **/etc/yum.repos.d/** в домашнюю папку пользователя root.

```
# mv /etc/yum.repos.d/* ~
```

5. Создайте конфигурационный файл локального репозитория, со следующими параметрами:
   - Конфигурационный файл: **/etc/yum.repos.d/centos7.repo**.
   - Идентификатор репозитория: `c7-local`.
   - Имя репозиттория: `Course LL-100`.
   - Путь к репозиторию: `file:///var/ftp/pub/centos7`.
   - Проверка ключей репозитория: отключена (`0`).

```
# vi /etc/yum.repos.d/centos7.repo
[c7-local]
name=Course LL-100
baseurl=file:///var/ftp/pub/centos7
gpgcheck=0
enabled=1
```

6. Убедитесь в работоспособности локального репозитония.

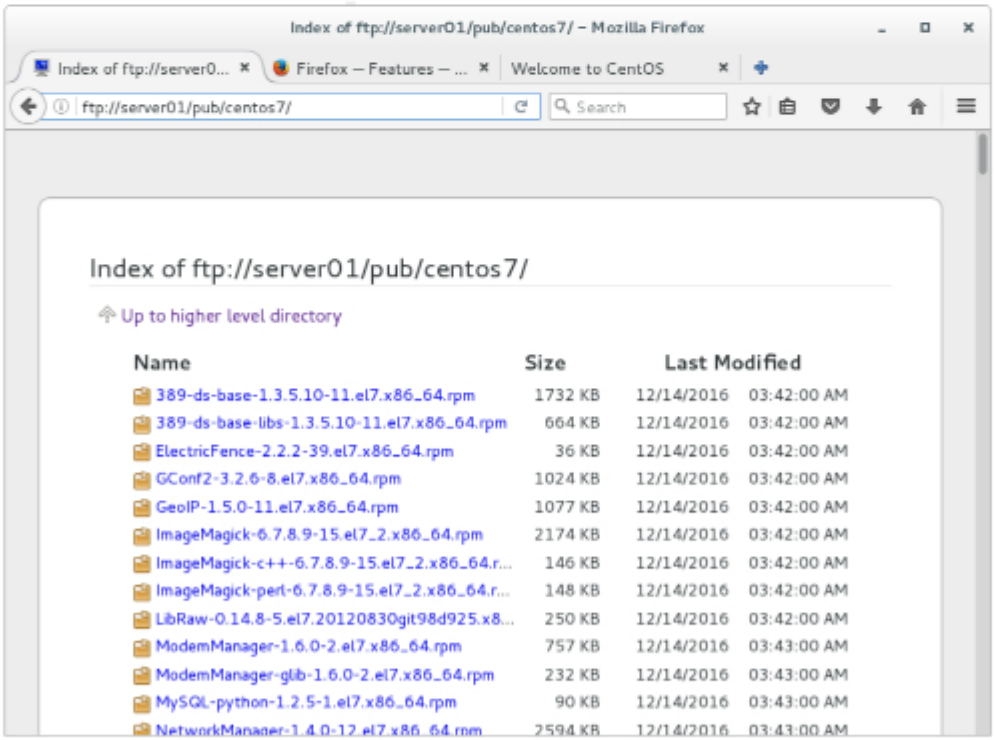
```
# yum repolist
```

7. Установите любой пакет с зависимостями из репозитория (например, редактор Emacs).

```
# yum install emacs
```

Задание – 3: Настройка удаленного репозитория.

1. На машине c7-client01 проверьте доступ к FTP (ftp://server01/pub/centos7).



2. Отключите все репозитории YUM на системе.

```
# yum-config-manager --disable "CentOS-7 - Base"
# yum-config-manager --disable "CentOS-7 - Extras"
# yum-config-manager --disable "CentOS-7 - Updates"
```

3. Подключите вновь созданный репозиторий.

```
# yum-config-manager --add-repo ftp://server01/pub/centos7
```

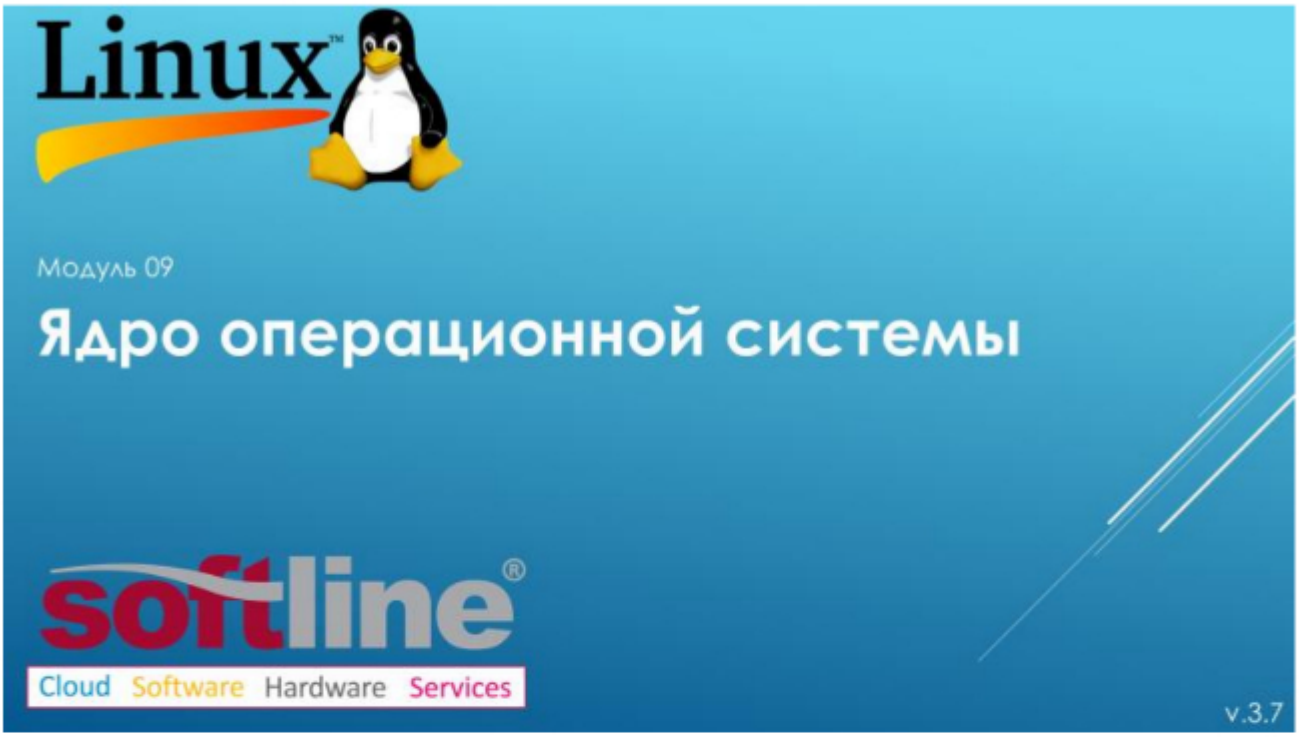
4. Убедитесь в работоспособности локального репозитония.

```
# yum repolist
```

5. Установите любой пакет с зависимостями из репозитория (например, редактор Emacs).

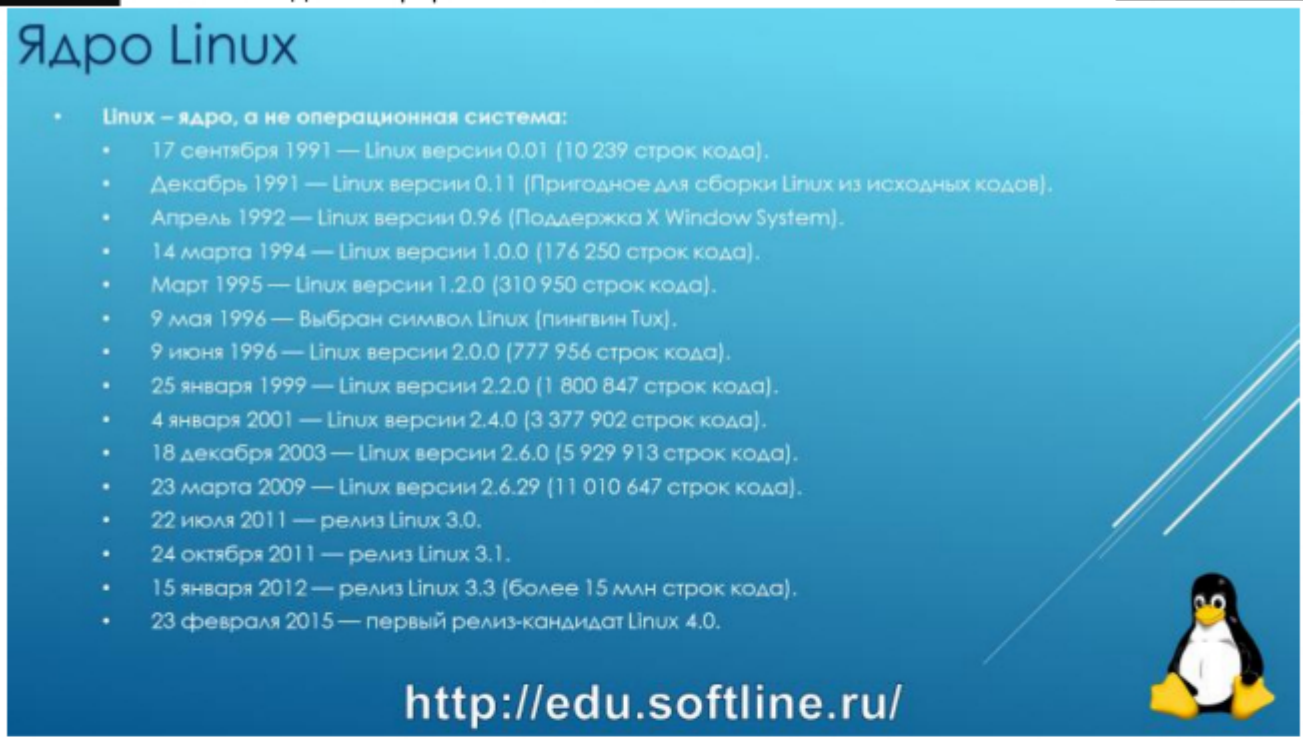
```
# yum install emacs
```

Модуль 9: Ядро операционной системы



В модуле представлены:

- Ядро операционной системы Linux.
- Установка и обновление ядра Linux.
- Сборка ядра из исходных кодов.
- Ручное обновление ядра Linux.



Ядро Linux было создано финским студентом Линусом Торвальдсом. В 1991 году в новостной группе comp.os.minix появилось сообщение следующего содержания:

> *Привет всем, кто использует minix — я делаю (бесплатную) операционную систем (всего лишь хобби, не будет большой и профессиональной как GNU) для клонов 386(486) AT...*

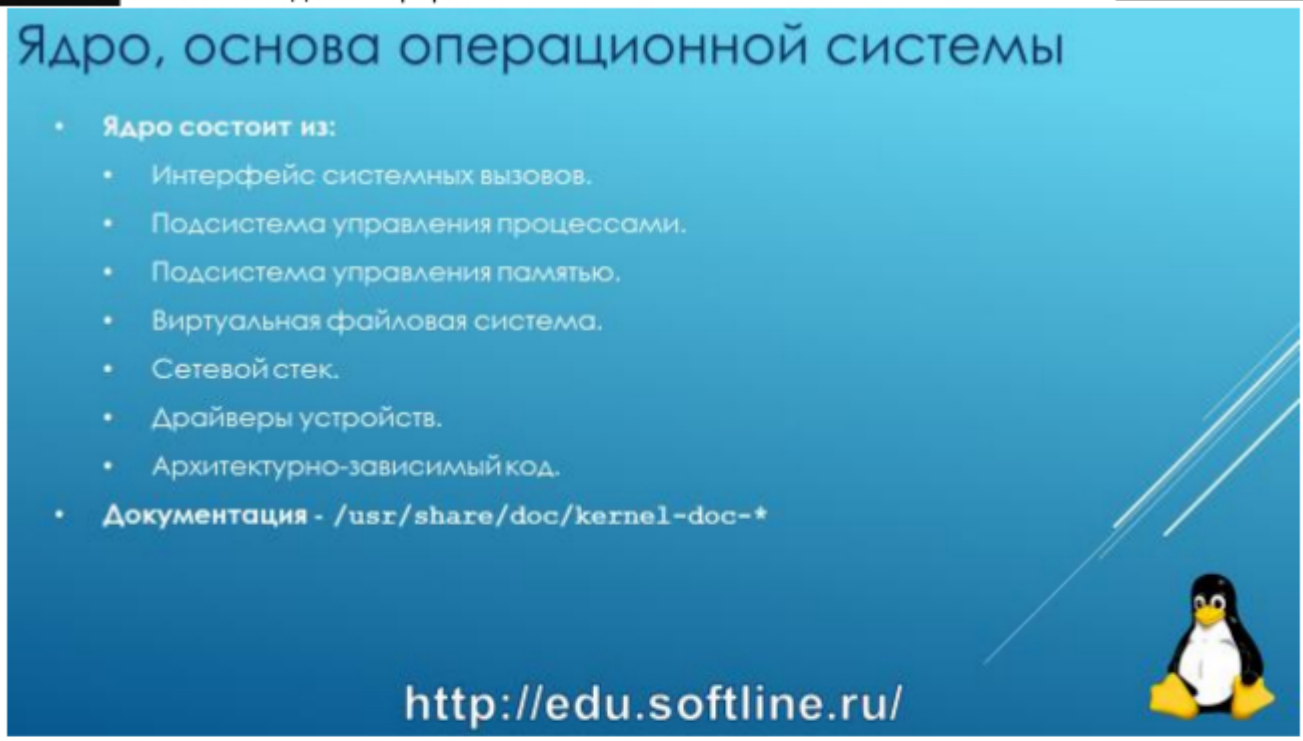
Linux — название ядра, а не операционной системы, но поскольку ядро является основной составляющей операционной системы, то и операционную систему называют Linux. Большинство систем Linux на самом деле необходимо называть GNU/Linux, так как они состоят из ядра Linux и множества программ и библиотек, разработанных в рамках проекта GNU.

В настоящее время, Торвальдс курирует разработку ядра, внося изменения и объединяя изменения, вносимые множеством других программистов со всего мира. Как правило, основные поставщики Linux-дистрибутивов создают свои ответвления от ядра, дополняя их драйверами, которых ещё нет в официальной версии.

Ядро имеет свою нумерацию, состоящую из 3-х чисел — *A*.*B*[.*C*]:

- Число *A* обозначает версию ядра, изменяется только при существенных изменениях в код и концепцию ядра, за всю историю ядра но изменялось трижды;
- Число *B* обозначает старшую версию ревизии ядра.
- Число *C* обозначает младшую версию ревизии ядра, изменяется при добавлении новых драйверов и улучшений. Если число *C* отсутствует, это означает, что данная версия ядра является разрабатываемой. Как только разработчики стабилизируют эту версию, она получит номер *C*=*0*, а разрабатываемая версия станет *A*.*B*+*1*

Также, часть версий ядра помечены как Long Term Support и поддерживаются дольше, чем обычные версии.



Ядро — очень сложная система, для простоты понимания, разделим его на логические составляющие:

Интерфейс системных вызовов (SCI) — это уровень, предоставляющий средства для вызова функций ядра из пространства пользователей.

Подсистема управления процессами — основой управления процессами является их выполнение. В пространстве ядра процесс называется потоком (thread). Одной из основных задач подсистемы управления процессами является организация совместного использования процессора множеством потоков.

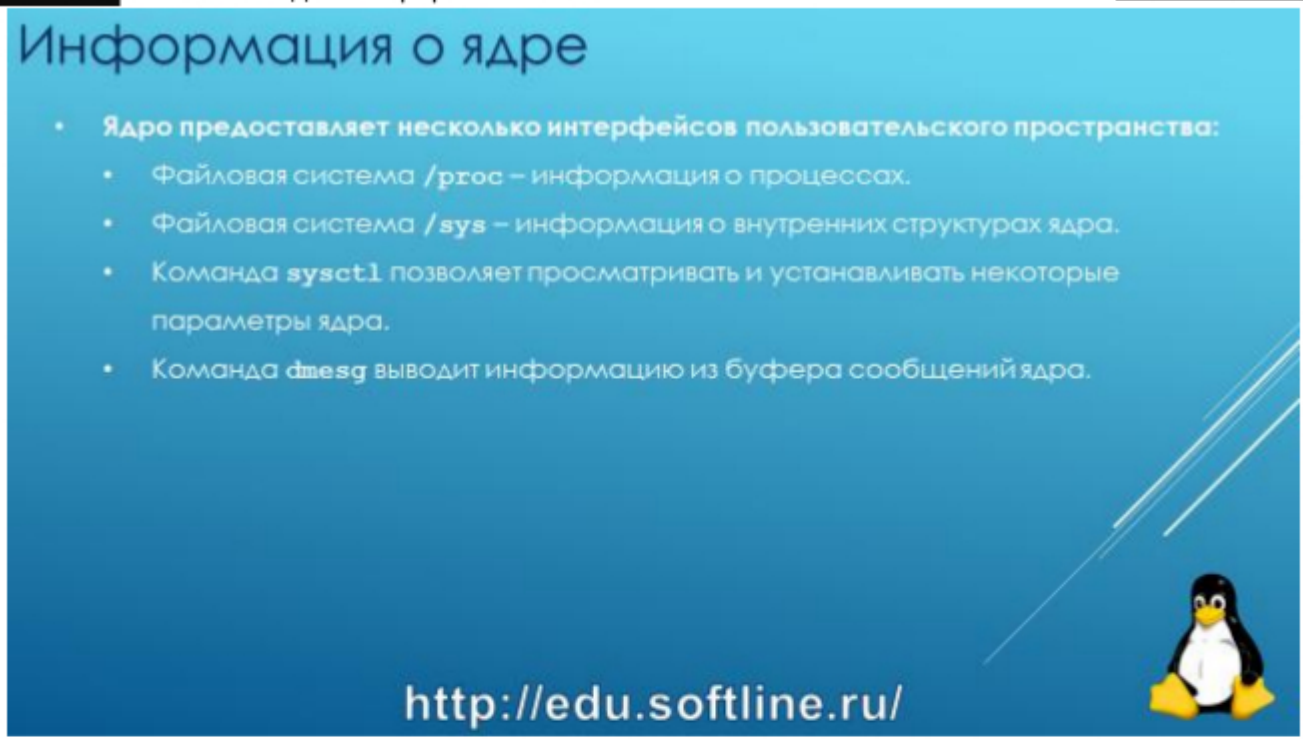
Подсистема управления памятью — ядро, помимо организации работы потоков, не должно забывать и про управление памятью, которая, для повышения эффективности, организуется в виде страниц (для большинства архитектур размер страницы составляет 4 кб). Linux управляет имеющейся памятью и имеет механизмы для установления соответствия между физической и виртуальной памятью. В случае, когда в системе слишком много пользователей памяти, последняя может закончиться, в таком случае, ядро должно вычислить те страницы памяти, которые можно перенести на диск, замедлив работу с ними, но освободив часть оперативной памяти. Такой процесс называется подкачкой.

Виртуальная файловая система — данная подсистема предоставляет общую абстракцию интерфейса к файловым системам. VFS предоставляет уровень коммутации между SCI и файловыми системами, поддерживаемыми ядром. На верхнем уровне VFS располагается API, предоставляющий общую абстракцию к таким функциям файловой системы, как открытие, закрытие, чтение и запись файлов. На нижнем уровне VFS находится сама файловая система.

Сетевой стек — представляет многоуровневую архитектуру, повторяющую структуру самих протоколов. Уровень сокетов представляет собой API к сетевой подсистеме. Он предоставляет пользовательский интерфейс к различным сетевым протоколам.

Драйверы устройств — большей частью исходного кода ядра являются коды драйверов устройств, обеспечивающих возможность работы с различными устройствами.

Архитектурно-зависимый код — основная часть ядра независима от архитектуры, на которой работает операционная система. Тем не менее, некоторые части требуют привязки к архитектуре системы для того, чтобы повысить производительность.



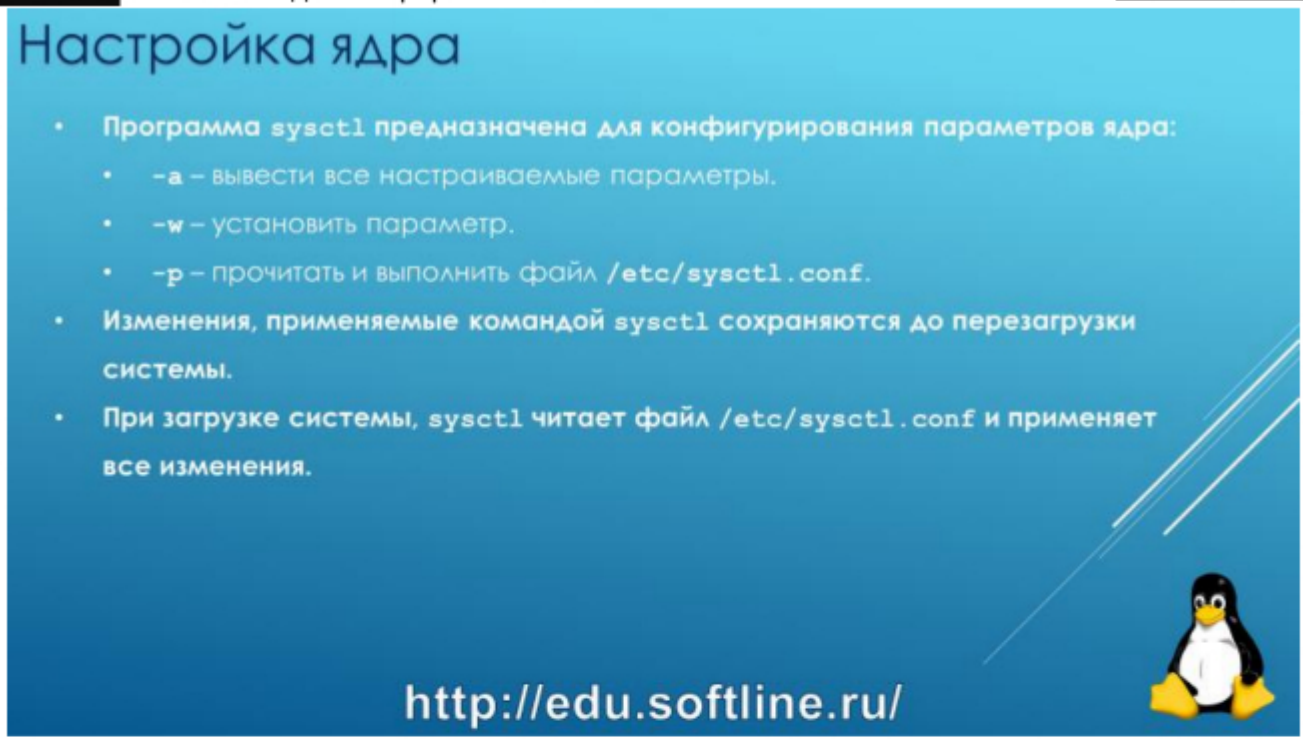
Виртуальная файловая система **/proc** предоставляет доступ к детальной информации о жизнедеятельности ядра, запущенных процессах. Каждый файл содержит информацию о каком-либо элементе.

В **/proc** есть несколько очень интересных файлов:

- **/proc/cmdline** — команды, которые были переданы загрузчику в командной строке.
- **/proc/cpuinfo** — информация о процессоре.
- **/proc/meminfo** — информация об использовании памяти.
- **/proc/modules** — список загруженных модулей.
- **/proc/partitions** — информация о разделах.
- **/proc/net/arp** — ARP таблица.
- **/proc/net/route** — таблица маршрутизации.

Файловая система **/sys** появилась относительно недавно и призвана вытеснить устаревшую файловую систему **/proc**:

- **/sys/block/*** - информация о блочных устройствах;
- **/sys/bus/*** - информация об устройствах;
- **/sys/modules/*** - информация о модулях.

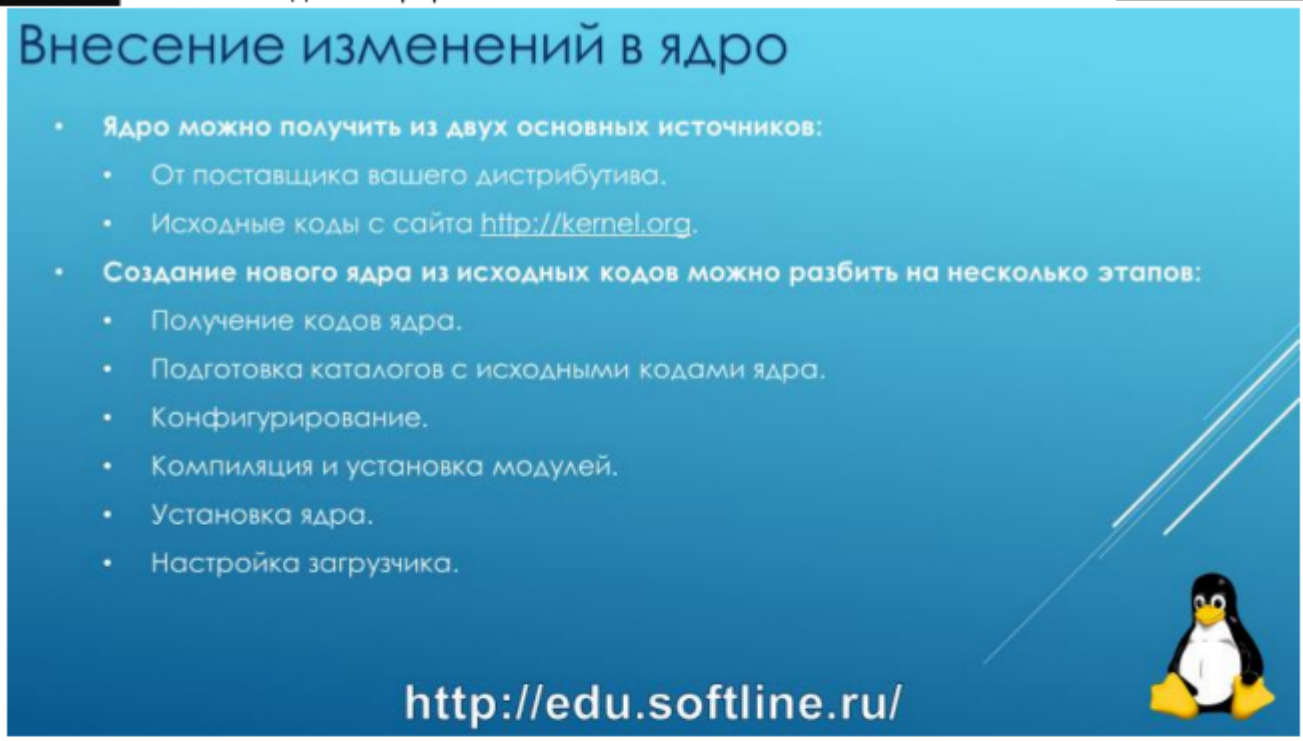


Ядро, поставляемое с системой настроено таким образом, чтобы удовлетворять среднестатистические потребности, но в процессе работы вам может понадобиться более тонкая настройка некоторых параметров ядра. Документация, поставляемая с ядром, содержит описание параметров. Команда **sysctl** предоставляет администратору системы возможность настройки тех параметров, которые ему кажутся не оптимальными. Для получения списка всех параметров, воспользуйтесь командой:

```
# sysctl -a
```

Чтобы установить параметр, используйте параметр "**-w**". Например, чтобы включить перенаправление сетевых пакетов, можно воспользоваться двумя разными командами, приводящими в одному и тому же результату:

```
# sysctl -w net.ipv4.ip_forward=1
# echo 1 > /proc/sys/net/ipv4/ip_forward
```



Как правило ядро, поставляемое с дистрибутивом операционной системы, включает в себя полный набор всех драйверов, которые могут понадобиться для установки на различные системы. Естественно, что для каждой системы лучше включить в ядро только то, что действительно необходимо, тем самым избежав лишних потерь производительности.

Чтобы собрать ядро необходимо его получить. Все версии ядер находятся на сервере kernel.org, откуда вы можете свободно загрузить исходные коды ядра.

Чтобы получить исходные коды, воспользуйтесь командной **wget**:

```
# cd /usr/src
# wget http://kernel.org/pub/linux/kernel/v2.6/linux-2.6.22.9.tar.bz2
```

Теперь необходимо перейти в распакованный каталог с исходными кодами ядра Linux и запустить командой **make** программу конфигурирования. Существует несколько способов настройки ядра:

- **menuconfig** — цветное текстовое меню с диалоговыми окнами, является основным способов настройки ядра;
- **xconfig** — то же самое, что и menuconfig, но в GUI (X-сервер). Основано на библиотеке Qt;
- **gconfig** — то же самое, что и xconfig, но основано на библиотеке Gtk;
- **oldconfig** — конфигурирование основывается на настройках в файле **./.config**. Пользователь должен настроить только новые функции.
- **allmodconfig** — конфигурирование будет произведено таким образом, что везде, где это возможно, будет использована модульная структура.

Основным конфигурационным файлом является файл **.config**, который появляется в результате произведённых настроек. По умолчанию, во многих дистрибутивах, в каталоге **/boot** помимо файла ядра находится его конфигурация:

```
# ls /boot
...
```

```
config-2.6.21-1.3194.fc7      vmlinuz-2.6.21-1.3194.fc7
...
```

В случае, когда вы хотите произвести обновление ядра с использованием текущей конфигурации, скопируйте конфигурационный файл в дерево исходных кодов ядра и выполните команду:

```
# cp config-2.6.21-1.3194.fc7 /usr/src/linux-2.6.22.9/.config
# cd /usr/src/linux-2.6.22.9/
# make oldconfig
```

После выполнения этой команды начнётся пройцесс конфигурирования ядра, при котором на все вопросы конфигуратора ответы будут получены из файла **./.config**, на все вопросы, касаемо новых функций ядра, будет предложено ответить администратору.

Порой необходимо не только обновить ядро, но и внести какие-то изменения, будь то отключение ненужных модулей или изменение типа ядра. Чтобы настроить ядро так, как вам требуется, дайте следующую команду:

```
# make menuconfig
```

Всё, что вам необходимо сделать, отметить необходимые пункты, в соответствии с подсказкой, данной в верхней части меню.

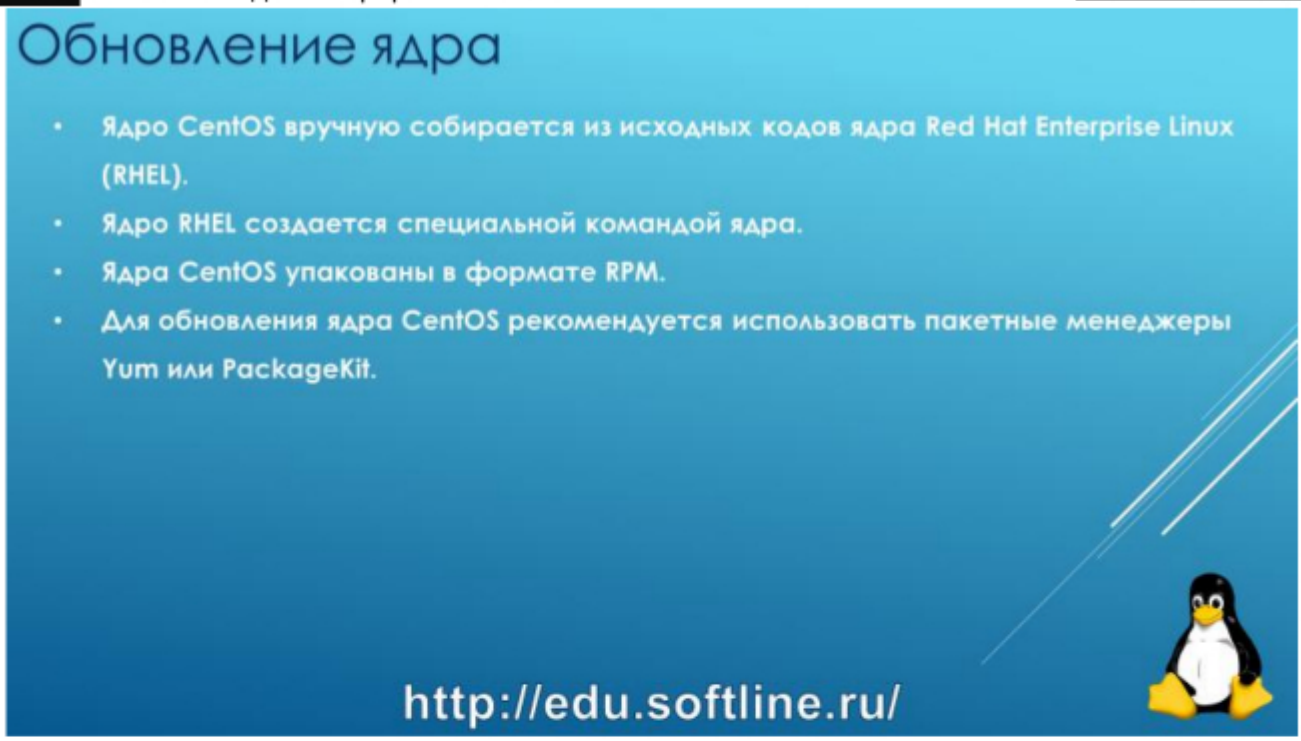
После того, как ядро было настроено необходимо произвести компиляцию и установку. Для компиляции необходимо выполнить, уже известную, команду make, после чего начнётся процесс сборки, который может затянуться на некоторое время. Если компиляция была завершена без ошибок, значит, что ядро было собрано (**/usr/src/linux-2.6.32.59/arch/i386/boot/bzImage**), его уже можно использовать, но перед этим требуется установить все модули в каталог **/usr/lib**, выполните команду:

```
# make
# make modules_install
```

Теперь в каталоге **/usr/lib** появилось дерево модулей, необходимых для работы нового ядра. Остаётся только скопировать файл ядра в каталог /boot и внести соответствующие изменения в конфигурационный файл загрузчика grub. Также, возможна автоматическая установка ядра:

```
# make install
```

Не забывайте, что новое ядро может не загрузиться по каким-либо причинам, поэтому всегда оставляйте возможность загрузки с предыдущей версией ядра.



Ядро CentOS вручную собирается из исходных кодов ядра Red Hat Enterprise Linux (RHEL). В свою очередь ядро RHEL создается специальной командой ядра, чтобы убедиться в целостности и совместимости с поддерживаемым оборудованием. Перед каждым релизом ядра, Red Hat проводит комплексное оценочное тестирование.

Ядра CentOS упакованы в формате RPM, таким образом, что они могут быть легко обновлены и проверены при помощи пакетных менеджеров Yum или PackageKit.

Далее мы детально разберем управляемый процесс обновления пакета ядра при помощи команды **rpm** вместо **yum**.



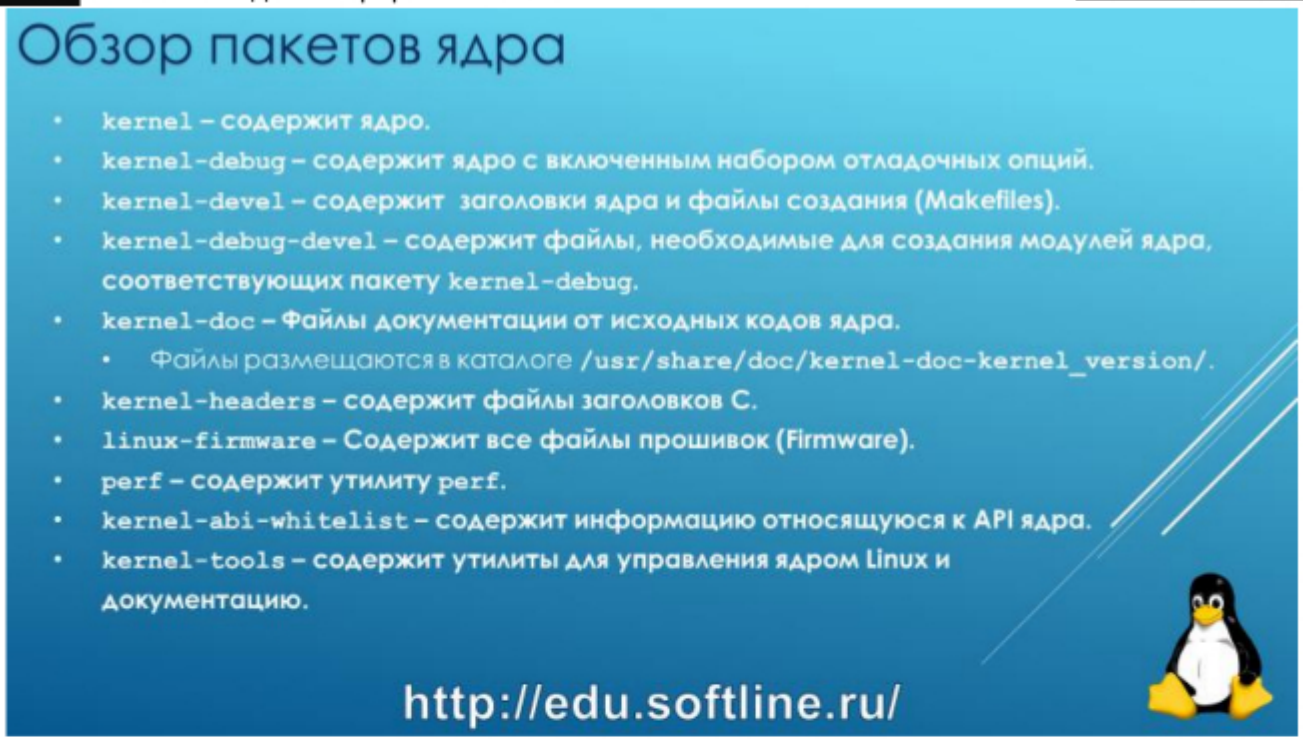
ВНИМАНИЕ:

По возможности используйте пакетный менеджер Yum или PackageKit для установки новых версий ядра, так как он устанавливает новое ядро, вместо замены текущего, что позволяет защититься от невозможности загрузиться.



ВНИМАНИЕ:

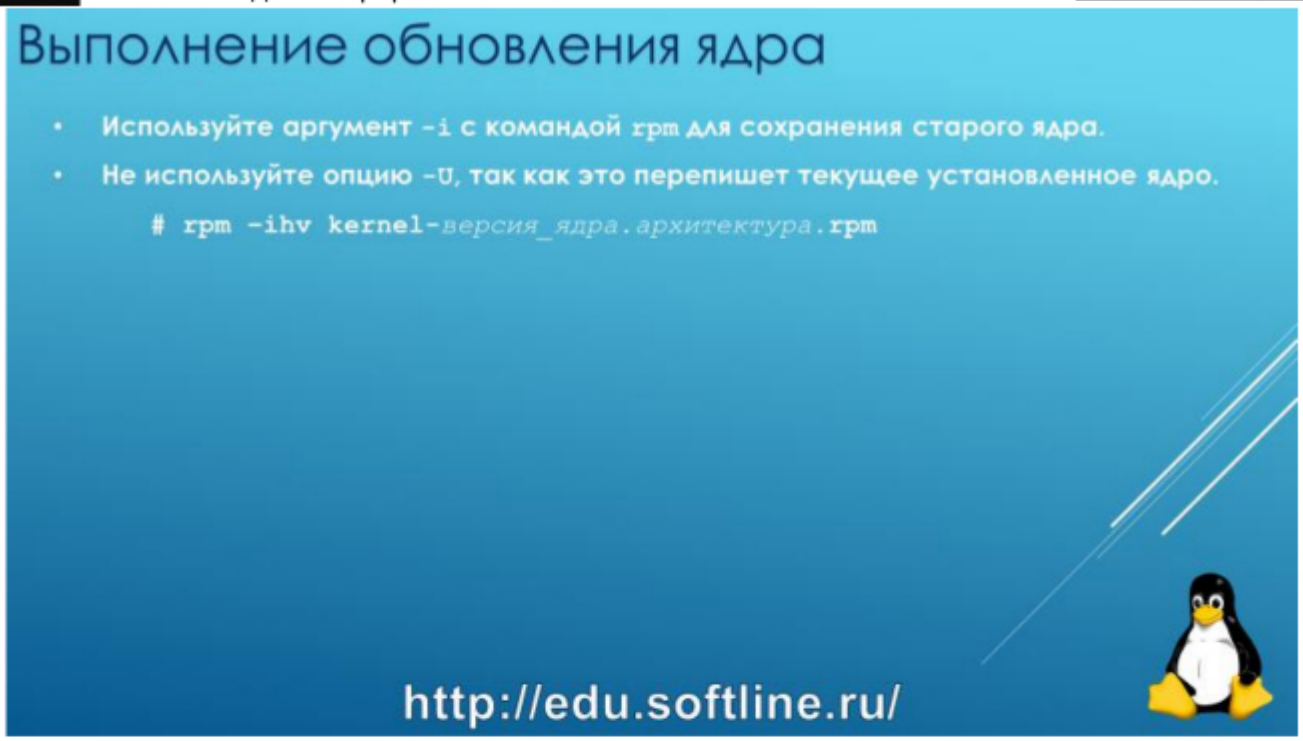
Сборка ядра из исходных кодов не рекомендуется для дистрибутивов CentOS (RHEL).



- **kernel** – содержит ядро, для одноядерных, многоядерных и многопроцессорных систем.
- **kernel-debug** – содержит ядро с включенным набором отладочных опций для диагностики ядра и с меньшей производительностью.
- **kernel-devel** – содержит заголовки ядра и файлы создания (Makefiles) необходимые для создания модулей ядра.
- **kernel-debug-devel** – содержит файлы, необходимые для создания модулей ядра, соответствующих пакету **kernel-debug**.
- **kernel-doc** – Файлы документации от исходных кодов ядра. Различные секции ядра Linux и поставляемых драйверов устройств описаны в этих файлах документации. Установка этих файлов предоставляет описание к опциям, которые могут быть переданы модулям ядра Linux во время загрузки.

  По умолчанию эти файлы размещаются в каталоге **/usr/share/doc/kernel-doc-kernel_version/**.

- **kernel-headers** – содержит файлы заголовков C, которые определяют интерфейс между ядром Linux и пользовательскими библиотеками, и программами. Файлы заголовков определяют структуру и константы необходимые для создания большинства стандартных программ.
- **linux-firmware** – Содержит все файлы прошивок (Firmware), которые необходимы различным устройствам для работы.
- **perf** – этот пакет, содержит утилиту **perf**, которая обеспечивает мониторинг ядра Linux.
- **kernel-abi-whitelist** – содержит информацию, относящуюся к API ядра CentOS (RHEL), в том числе список символов ядра, которые необходимы внешним модулям ядра Linux и плагину yum для обеспечения применения.
- **kernel-tools** – содержит утилиты для управления ядром Linux и документацию.



После загрузки всех необходимых пакетов, пришло время для обновления существующего ядра.

ПРИМЕЧАНИЕ:
Настоятельно рекомендуется сохранить старое ядро на случай возникновения проблем с новым ядром.

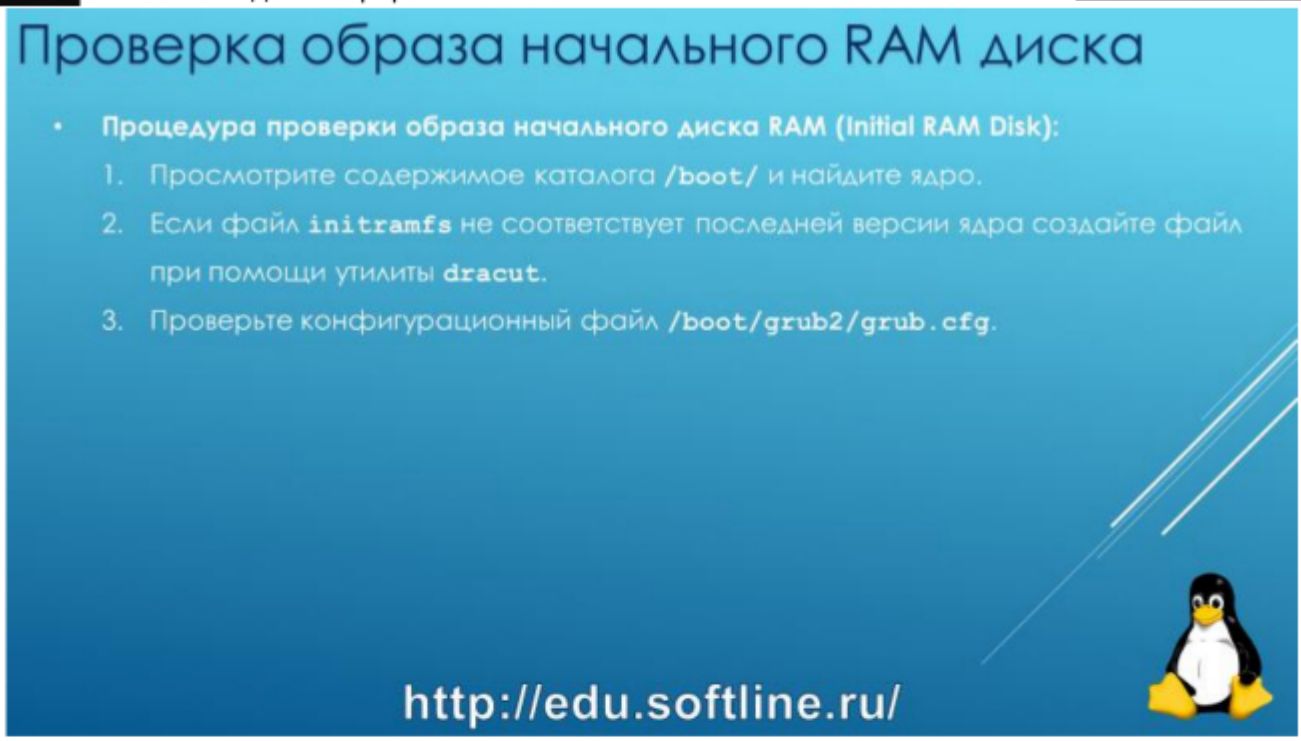
В командной строке перейдите в каталог с загруженным RPM пакетом. Используйте аргумент **-i** с командой **rpm** для сохранения старого ядра. Не используйте опцию **-U**, так как это перепишет текущее установленное ядро, что может в дальнейшем вызвать проблемы с загрузкой.

 # rpm -ihv kernel-*версия_ядра.архитектура***.rpm**

Пример установки нового ядра:

```
[root@lpc01 KERNEL]# rpm -ihv kernel-3.10.0-327.36.3.el7.x86_64.rpm
Preparing...                          ################################# [100%]
Updating / installing...
   1:kernel-3.10.0-327.36.3.el7       ################################# [100%]
```

Следующим шагом необходимо проверить, что начальный диск RAM (Initial RAM Disk) был создан.



Основная задача начального RAM диска заключается в загрузку модулей блочных устройств, таких как IDE, SCSI или RAID, так как корневая файловая система обычно располагается на них, они должны быть доступны и смонтированы. В системах CentOS 7 (RHEL 7), когда новое ядро устанавливается при помощи пакетных менеджеров Yum, PackageKit или RPM, утилита Dracut всегда вызывает установочный скрипт для создания **initramfs** (образ начального диска RAM).

На всех архитектурах кроме other than IBM eServer System i можно создать **initramfs** при помощи утилиты **dracut**. Однако, обычно нет необходимости создавать **initramfs** вручную, так как этот шаг выполняется автоматически если ядро и связанные пакеты установлены или обновляются из пакетов RPM.

Вы можете проверить, что **initramfs** соответствует текущей версии ядра и он корректно указан в конфигурационном файле **grub.cfg**, при помощи следующей процедуры:

Процедура проверки образа начального диска RAM (Initial RAM Disk).

1. От имени пользователя root просмотрите содержимое каталога **/boot/** и найдите ядро (**vmlinux-***версия_ядра*) и **initramfs-***версия_ядра* с последними версиями.

```
[root@lpc01 KERNEL]# ls /boot/
config-3.10.0-327.22.2.el7.x86_64
config-3.10.0-327.36.3.el7.x86_64
grub
grub2
initramfs-0-rescue-bef274407d4d409ab35ad045ffa59ab3.img
initramfs-3.10.0-327.22.2.el7.x86_64.img
initramfs-3.10.0-327.36.3.el7.x86_64.img
initrd-plymouth.img
symvers-3.10.0-327.22.2.el7.x86_64.gz
symvers-3.10.0-327.36.3.el7.x86_64.gz
System.map-3.10.0-327.22.2.el7.x86_64
System.map-3.10.0-327.36.3.el7.x86_64
vmlinuz-0-rescue-bef274407d4d409ab35ad045ffa59ab3
vmlinuz-3.10.0-327.22.2.el7.x86_64
vmlinuz-3.10.0-327.36.3.el7.x86_64
```

Предыдущий пример показывает:

- Одно или несколько ядер установлены (если точнее, то следующие файлы представлены в каталоге **/boot**).
- Последняя версия ядра это **vmlinuz-3.10.0-327.36.3.el7.x86_64**.
- Файл **initramfs**, соответствующий версии ядра, **initramfs-3.10.0-327.36.3.el7.x86_64.img** также существует.



ПРИМЕЧАНИЕ:

*В каталоге **/boot/** вы можете найти несколько файлов **initrd**-версия**kdump.img**. Это специальные файлы созданные механизмом Kdump в целях отладки ядра, не используются для загрузки системы и могут быть безопасно проигнорированы.*

2. Если файл **initramfs-***версия_ядра* не соответствует последней версии ядра в каталоге /boot/, или в иных подобных ситуациях, вам может потребоваться создать файл initramfs при помощи утилиты **dracut**. В таком случае просто вызовите **dracut** с правами пользователя root, без указания параметров для создания файла **initramfs** в каталоге **/boot/** для последней версии ядра представленной в каталоге:

```
# dracut
```

Необходимо использовать опцию **-f**, **--force**, если вы хотите переписать существующие версии **initramfs** (например, когда **initramfs** поврежден). Иначе **dracut** проигнорирует существующие файлы **initramfs**:

```
[root@lpc01 KERNEL]# dracut
Will not override existing initramfs (/boot/initramfs-3.10.0-
327.22.2.el7.x86_64.img) without --force
```

Вы можете создать **initramfs** в текущем каталоге при помощи **dracut**.

```
# dracut "initramfs-$(uname -r).img" $(uname -r)
```

Если необходимо указать определенные модули ядра, чтобы они были предварительно загружены, нужно добавить имена этих модулей (без указания каких-либо суффиксов, таких как **.ko**) в директиве **add_dracutmodules+="***модуль* **[***дополнительные_модули***]"** конфигурационного файла **/etc/dracut.conf**. Вы можете просмотреть содержимое образа **initramfs** созданного утилитой **dracut**, при помощи команды **lsinitrd** *файл_initramfs*:

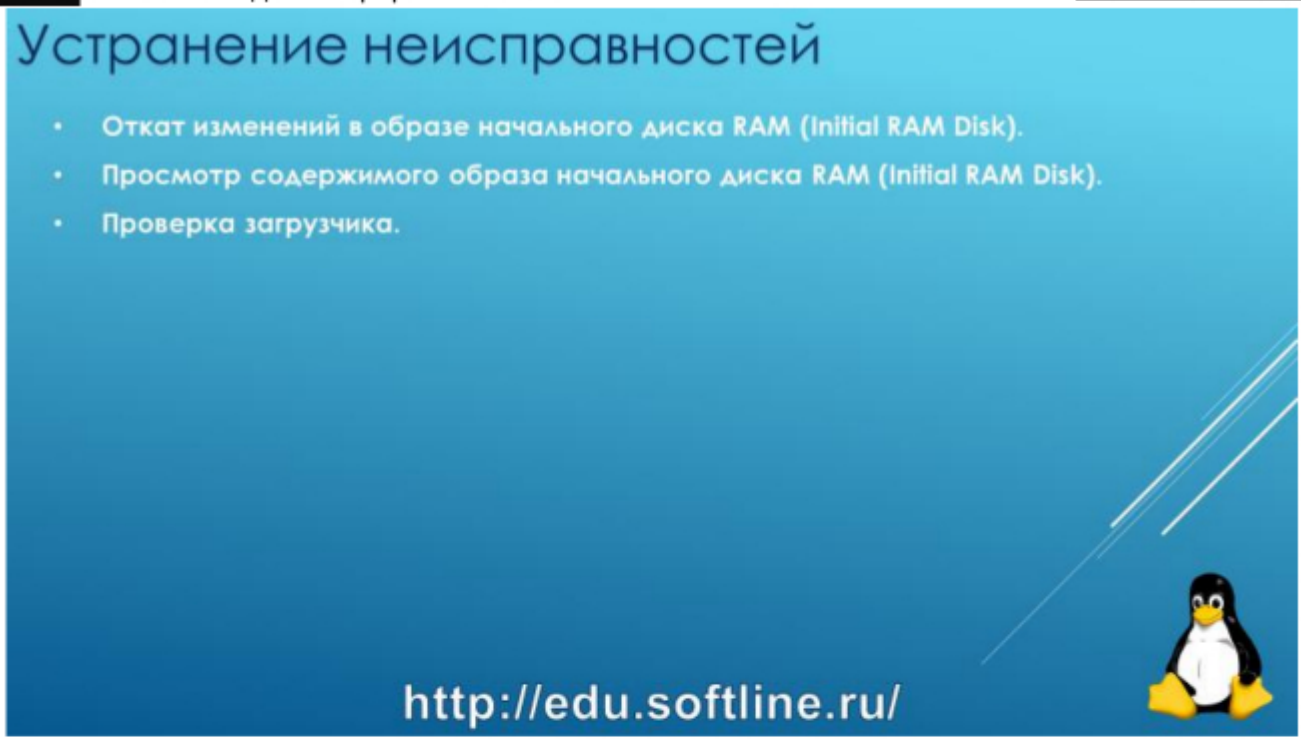
```
[root@lpc01 KERNEL]# lsinitrd /boot/initramfs-3.10.0-327.36.3.el7.x86_64.img
Image: /boot/initramfs-3.10.0-327.36.3.el7.x86_64.img: 29M
========================================================================
Early CPIO image
========================================================================
drwxr-xr-x   3 root     root            0 Dec  4 08:29 .
-rw-r--r--   1 root     root            2 Dec  4 08:29 early_cpio
drwxr-xr-x   3 root     root            0 Dec  4 08:29 kernel
drwxr-xr-x   3 root     root            0 Dec  4 08:29 kernel/x86
drwxr-xr-x   2 root     root            0 Dec  4 08:29 kernel/x86/microcode
-rw-r--r--       1   root           root                  10240   Dec     4   08:29
kernel/x86/microcode/GenuineIntel.bin
========================================================================
Version: dracut-033-360.el7_2.1
```

...

Для получения дополнительной информации по опциям и использованию воспользуйтесь **man dracut** и **man dracut.conf**.

3. Проверьте конфигурационный файл **/boot/grub2/grub.cfg** чтобы убедиться что **initrd initramfs-***версия_ядра***.img** существует для версии ядра, которое необходимо загрузить. Например:

```
[root@lpc01 KERNEL]# grep initramfs /boot/grub2/grub.cfg
        initrd16 /initramfs-3.10.0-327.36.3.el7.x86_64.img
        initrd16 /initramfs-3.10.0-327.22.2.el7.x86_64.img
        initrd16 /initramfs-0-rescue-bef274407d4d409ab35ad045ffa59ab3.img
```



Откат изменений в образе начального диска RAM (Initial RAM Disk).

В некоторых случаях, например, например, при некорректной настройке системы, она может перестать загружаться, может потребоваться откат изменений, сделанных в образе начального диска RAM (Initial RAM Disk).

Процедура отката изменений в образе начального диска RAM (Initial RAM Disk):

1. Перезагрузите систему, выбрав аварийное ядро в меню GRUB.
2. Измените некорректные настройки, которые привели к неисправности **initramfs**.
3. Пересоздайте **initramfs** с корректными настройка при помощи следующей команды:

```
# dracut --kver версия_ядра --force
```

Описанная выше процедура может быть полезна в случае, если, например, в файле **sysctl.conf** некорректно задан параметр **vm.nr_hugepages**. Так как файл **sysctl.conf** включается в **initramfs**, новая настройка **vm.nr_hugepages** применится к файлу **initramfs** только после его пересоздания.

ПРИМЕЧАНИЕ:

Большинство современных архитектур процессоров поддерживают большие страницы памяти (что позволяет сократить количество просмотров процессором/ОС), эти страницы называются Huge Pages (в Linux), Super Pages (в BSD) и Large Pages (в Windows), но все они делают одно и то же.

Просмотр содержимого образа начального диска RAM (Initial RAM Disk).

Для вывода списка файлов, включенных в **initramfs**, выполните следующую команду от имени пользователя root:

```
# lsinitrd
```

Например, для вывода содержимого каталога **/etc**, введите следующую команду:

```
# lsinitrd | grep etc/
```

Для вывода содержимого указанного файла, сохраненного в **initramfs** для текущего ядра, используйте опцию **-f**:

```
# lsinitrd -f имя_файла
```

Например, для вывода содержимого **sysctl.conf**, используйте следующую команду:

```
# lsinitrd -f /etc/sysctl.conf
```

Чтобы указать версию ядра, воспользуйтесь опцией **--kver**:

```
# lsinitrd --kver версия_ядра -f /etc/sysctl.conf
```

Например, для вывода содержимого **sysctl.conf**, из **initramfs** для ядра версии **3.10-327.10.1.el7.x86_64** используйте следующую команду:

```
# lsinitrd --kver 3.10.0-327.10.1.el7.x86_64 -f /etc/sysctl.conf
```

Проверка загрузчика.

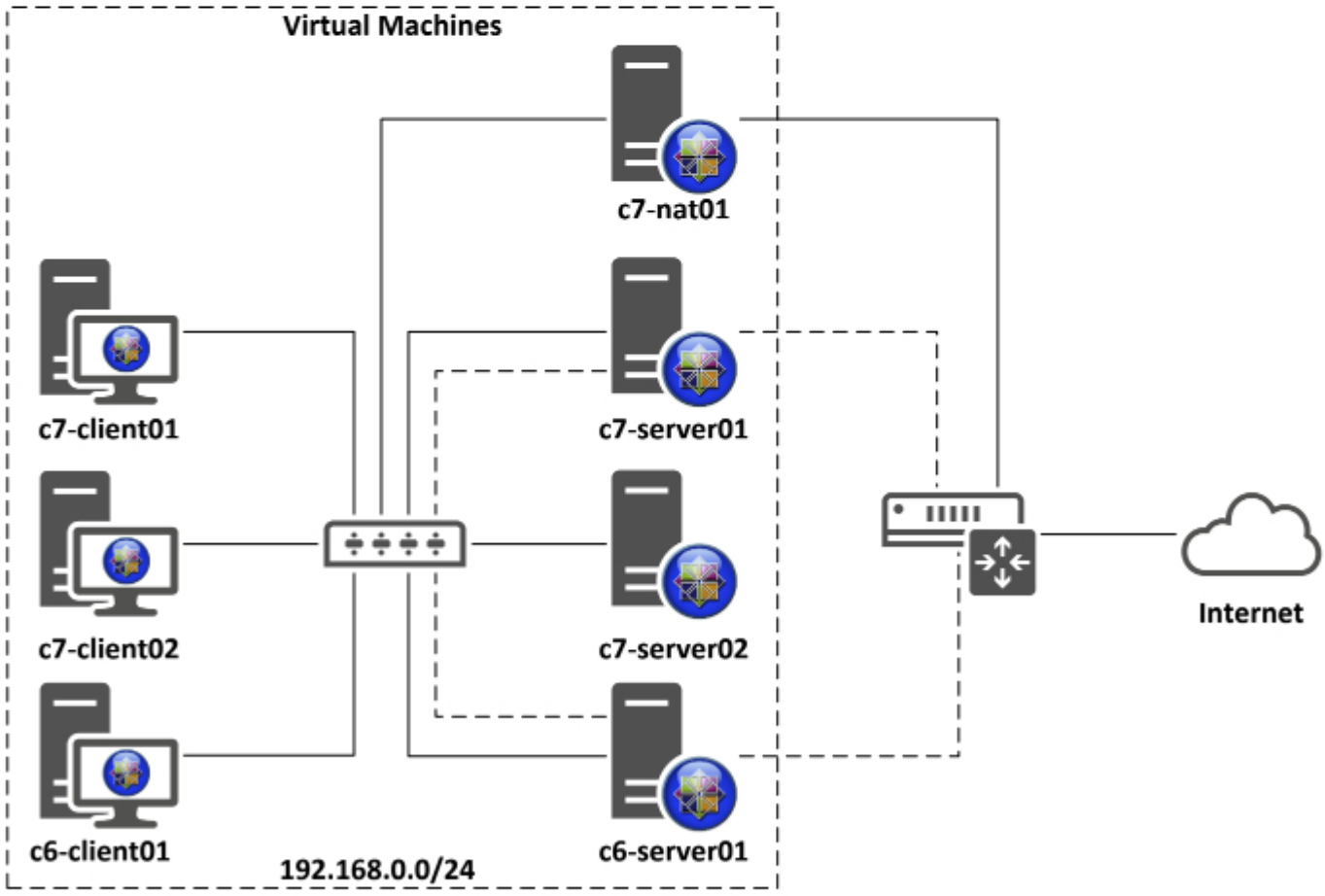
Когда вы устанавливаете ядро при помощи **rpm**, пакет ядра изменяет записи в конфигурационном файле ядра на новое ядро. Однако, **rpm** не настраивает новое ядро на загрузку по умолчанию. Вы должны это сделать вручную, когда устанавливаете ядро при помощи **rpm**.

Рекомендуется всегда проверять конфигурационный файл загрузчика после установки нового ядра при помощи **rpm**, чтобы убедиться, что конфигурация верна. В противном случае система может не загрузить CentOS (RHEL) корректно. Если это произойдет, необходимо загрузить систему при помощи загрузочного накопителя, приготовленного заранее.

Лабораторная работа 9

- **Упражнения:**
  - Установка стабильного ядра Linux 4.x в CentOS 7.
  - Сборка ядра Linux из исходных кодов в CentOS 7.
- **Необходимые машины:**
  - `c7-nat01`
- **Учетные данные:**
  - Пользователь: `root`.
  - Пароль: **`Pa$$w0rd`**.

http://edu.softline.ru/





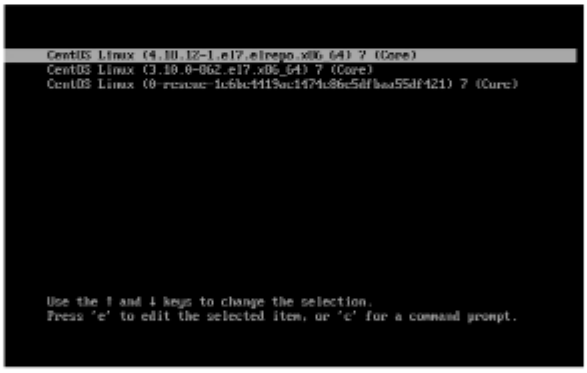
Упражнение 1 – Установка стабильного ядра Linux 4.x в CentOS 7.

Задача 1: Установка нового ядра Linux из RPM-пакета.

1. Запустите виртуальную машину **c7-server02**.
2. Посмотрите текущую версию ядра Linux.
3. Загрузите пакет с ядром Linux **`kernel-ml-4.18.12-1.el7.elrepo.x86_64.rpm`**.

> ⚠ *ПРИМЕЧАНИЕ:*
> *Уточните ссылку для загрузки у инструктора.*

4. Установите ядро Linux из пакета.
5. Убедитесь, что в каталоге **`/boot`** присутствуют файл ядра (**`vmlinuz`**) и образ начального диска RAM (**`initramfs`**).
6. Убедитесь, что в конфигурационном фале GRUB2 присутствует запись для загрузки установленного ядра Linux.
7. Перезагрузите систему, выбрав загрузку с новым ядром.

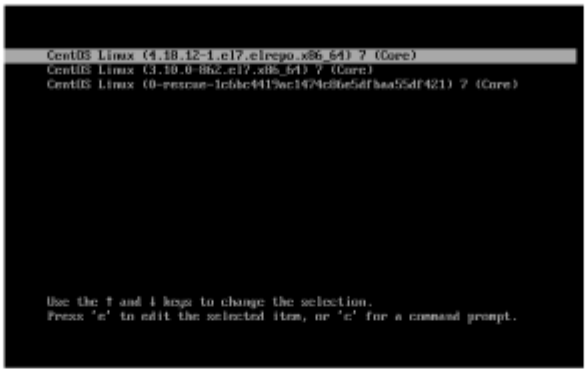


8. Убедитесь, что система запущена с ядром Linux версии 4.

Задача 2: Управление образом начального диска RAM (Initial RAM Disk).

1. Перейдите в каталог **`/boot`**.
2. Убедитесь, что в каталоге присутствуют файл ядра (**`vmlinuz`**) и образ начального диска RAM (**`initramfs`**) для ядра, установленного на предыдущем шаге.
3. Просмотрите содержимое образа начального диска RAM (**`initramfs`**) для ядра, установленного на предыдущем шаге.
4. Предположим, что мы нашли ошибку в образе начального диска RAM (**`initramfs`**), удалите его.
5. Вручную создайте образ начального диска RAM (**`initramfs`**) вместо удаленного.
6. Убедитесь, что в каталоге **`/boot`** присутствует образ начального диска RAM (**`initramfs`**).

7. Пересоздайте конфигурационный файл загрузчика GRUB2.
8. Перезагрузите систему, выбрав загрузку с ядром Linux версии 4.



9. Убедитесь, что система успешно запущена с ядром Linux версии 4.



Упражнение 1 – Установка стабильного ядра Linux 4.x в CentOS 7.

Задача 1: Установка нового ядра Linux из RPM-пакета.

1. Запустите виртуальную машину **c7-server02**.
2. Посмотрите текущую версию ядра Linux.

```
# uname -r
```

3. Загрузите пакет с ядром Linux **kernel-ml-4.18.12-1.el7.elrepo.x86_64.rpm**.

```
# wget http://URL/files/kernel-ml-4.18.12-1.el7.elrepo.x86_64.rpm
```

ПРИМЕЧАНИЕ:
Уточните ссылку для загрузки у инструктора.

4. Установите ядро Linux из пакета.

```
# rpm -ihv kernel-ml-4.18.12-1.el7.elrepo.x86_64.rpm
```

5. Убедитесь, что в каталоге **/boot** присутствуют файл ядра (**vmlinuz**) и образ начального диска RAM (**initramfs**).

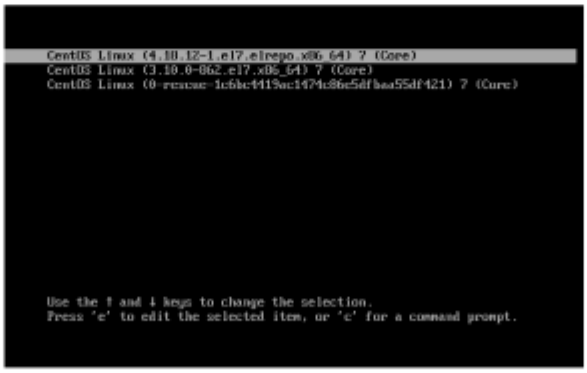
```
# ls -1 /boot/
```

6. Убедитесь, что в конфигурационном фале GRUB2 присутствует запись для загрузки установленного ядра Linux.

```
# cat /boot/grub2/grub.cfg | grep vmlinuz-4
```

7. Перезагрузите систему, выбрав загрузку с новым ядром.

```
# reboot
```



8. Убедитесь, что система запущена с ядром Linux версии 4.

```
# uname -r
```

Задача 2: Управление образом начального диска RAM (Initial RAM Disk).

1. Перейдите в каталог **/boot**.

   ```
   # cd /boot/
   ```

2. Убедитесь, что в каталоге присутствуют файл ядра (**vmlinuz**) и образ начального диска RAM (**initramfs**) для ядра, установленного на предыдущем шаге.

   ```
   # ls -1
   ```

3. Просмотрите содержимое образа начального диска RAM (**initramfs**) для ядра, установленного на предыдущем шаге.

   ```
   # lsinitrd initramfs-4.18.12-1.el7.elrepo.x86_64.img
   ```

4. Предположим, что мы нашли ошибку в образе начального диска RAM (**initramfs**), удалите его.

   ```
   # rm initramfs-4.18.12-1.el7.elrepo.x86_64.img
   ```

5. Вручную создайте образ начального диска RAM (**initramfs**) вместо удаленного.

   ```
   # dracut
   ```

6. Убедитесь, что в каталоге **/boot** присутствует образ начального диска RAM (**initramfs**).

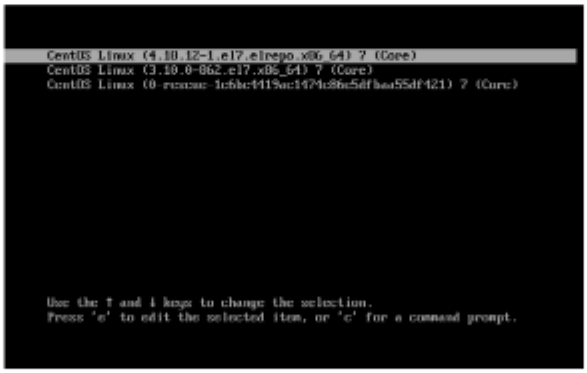
   ```
   # ls -1
   ```

7. Пересоздайте конфигурационный файл загрузчика GRUB2.

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

8. Перезагрузите систему, выбрав загрузку с ядром Linux версии 4.

   ```
   # reboot
   ```



9. Убедитесь, что система успешно запущена с ядром Linux версии 4.

   ```
   # uname -r
   ```



Упражнение 2: Сборка ядра Linux из исходных кодов в CentOS 7.

9. Запустите виртуальную машину **c7-server01**.
10. Посмотрите текущую версию ядра Linux.
11. Установите следующие пакеты на системе:
    - **`ncurses-devel`**.
    - **`make`**.
    - **`gcc`**.
    - **`bc`**.
    - **`openssl-devel`**.
    - **`elfutils-libelf-devel`**.
    - **`rpm-build`**.
    - **`bison`**.
    - **`flex`**.
12. Загрузите архив (**`linux-4.18.12.tar.xz`**) с исходными кодами ядра Linux.

⚠ *ПРИМЕЧАНИЕ:*
Уточните ссылку для загрузки у инструктора.

13. Распакуйте архив с ядром в текущий каталог.
14. Перейдите в каталог с распакованным ядром.
15. Скопируйте конфигурационный файл текущего ядра (**`/boot/config-3.10.0-862.el7.x86_64`**) в папку с исходными файлами и переименуйте его в **`.config`**.
16. Сохраните текущие настройки в файле **`.config`**.
17. Скомпилируйте ядро.

⚠ *ПРИМЕЧАНИЕ:*
В зависимости от мощности вашей машины это может занять от 30 до 60 минут.

18. Установите ядро и модули.
19. Перезагрузите машину и проверьте ее работу: версию ядра, работу сети и файловых систем.



Ответы к упражнению 2: Сборка ядра Linux из исходных кодов в CentOS 7.

1. Запустите виртуальную машину **c7-server01**.
2. Посмотрите текущую версию ядра Linux.

```
# uname -r
```

3. Установите следующие пакеты на системе:
   - **ncurses-devel**.
   - **make**.
   - **gcc**.
   - **bc**.
   - **openssl-devel**.
   - **elfutils-libelf-devel**.
   - **rpm-build**.
   - **bison**.
   - **flex**.

```
# yum install \
ncurses-devel \
make \
gcc \
bc \
openssl-devel \
elfutils-libelf-devel \
rpm-build \
bison \
flex
```

4. Загрузите архив (**linux-4.18.12.tar.xz**) с исходными кодами ядра Linux.

```
# wget http://URL/files/linux-4.18.12.tar.xz
```



ПРИМЕЧАНИЕ:
Уточните ссылку для загрузки у инструктора.

5. Распакуйте архив с ядром в текущий каталог.

```
# tar -xvf linux-4.18.12.tar.xz
```

6. Перейдите в каталог с распакованным ядром.

```
# cd linux-4.18.12
```

7. Скопируйте конфигурационный файл текущего ядра (**/boot/config-3.10.0-862.el7.x86_64**) в папку с исходными файлами и переименуйте его в **.config**.

```
# cp /boot/config-3.10.0-862.el7.x86_64 .config
```

8. Сохраните текущие настройки в файле **.config**.

```
# make oldconfig
```

9. Скомпилируйте ядро.

```
# make
```



ПРИМЕЧАНИЕ:

В зависимости от мощности вашей машины это может занять от 30 до 60 минут.

10. Установите ядро.

```
# make modules_install install
```



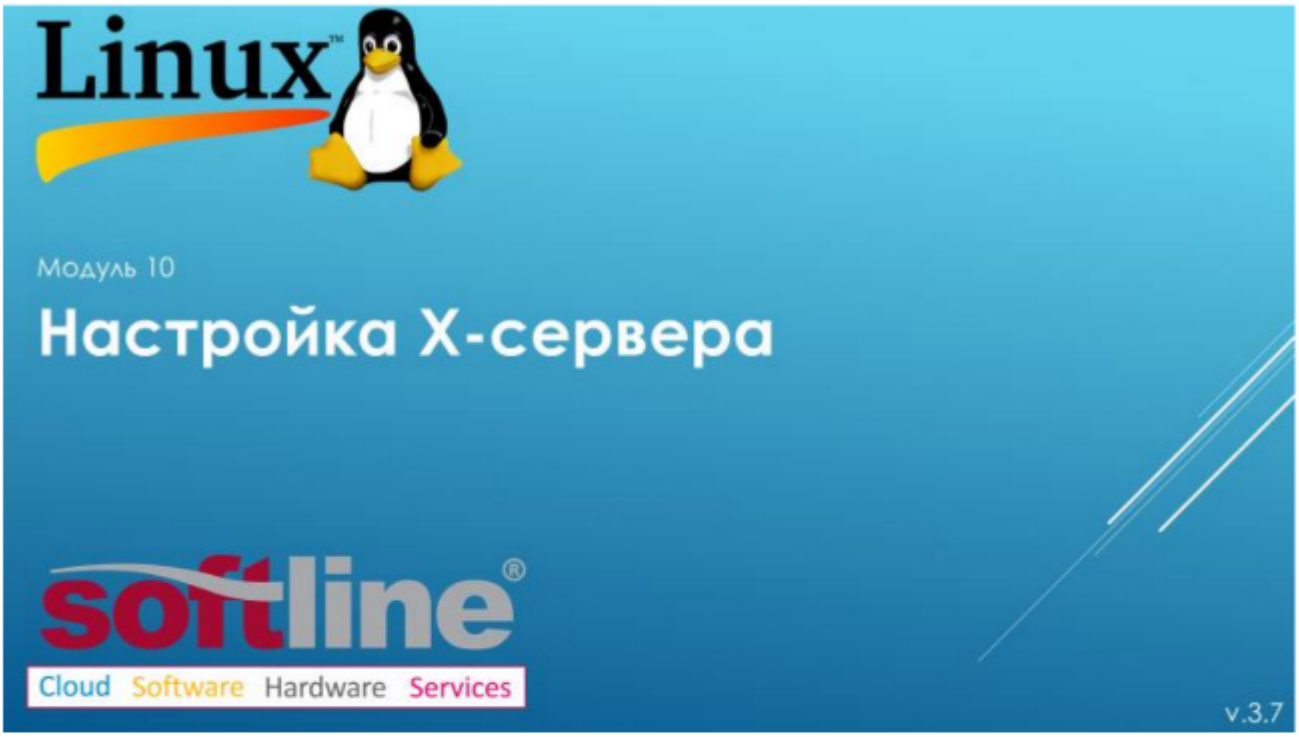
ПРИМЕЧАНИЕ:

*Также эта процедура автоматически добавит необходимые записи в GRUB 2. Но все же не забудьте проверить наличие файлов **vmlinuz** и **initramfs** (обратите внимание на разницу с файлами ядра 3.0) и соответствующей записи **menuentry** в **/etc/grub/grub.cfg**.*

11. Перезагрузите машину и проверьте ее работу: версию ядра, работу сети и файловых систем.

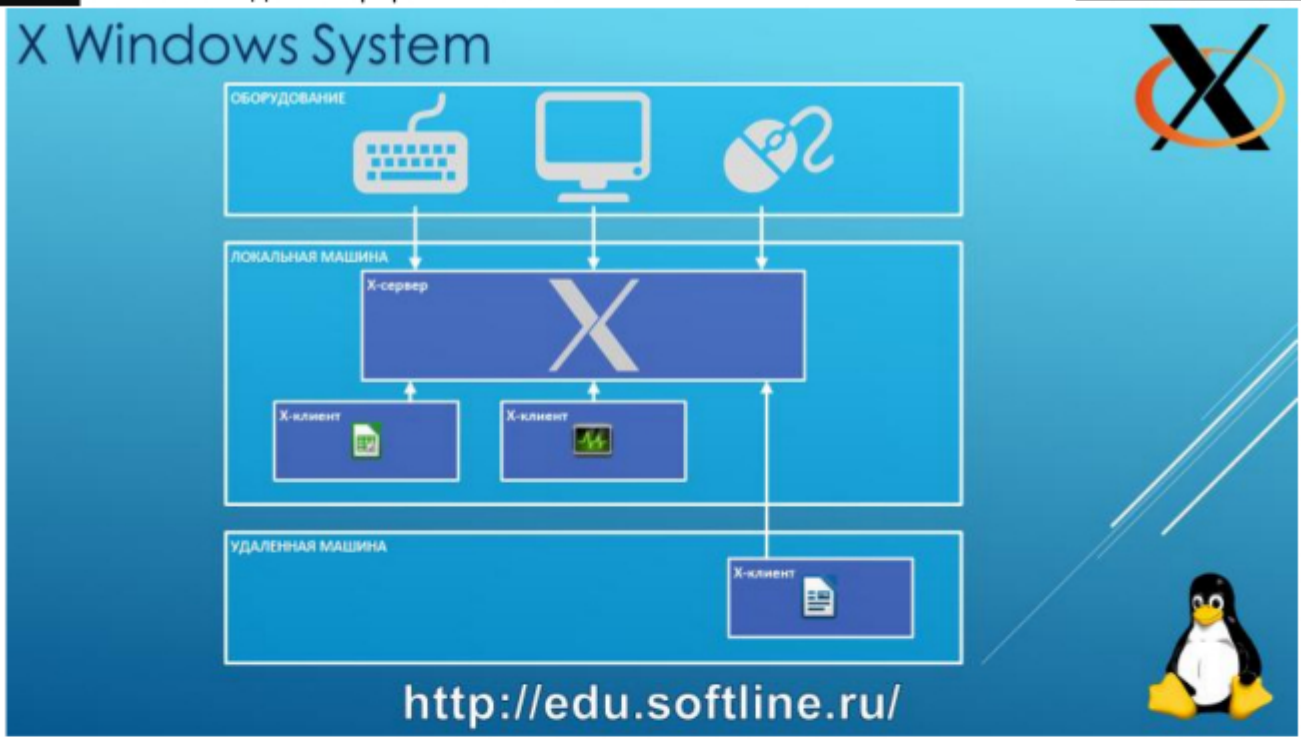
```
# uname -r
```

Модуль 10: Настройка X-сервера



В модуле представлены:

- X Windows System.
- Сервер X (X Server).
- Окружение рабочего стола.
- Оконные менеджеры.
- Каталог **`xorg.conf.d`**.
- Файл **`xorg.conf`**.
- Шрифты.

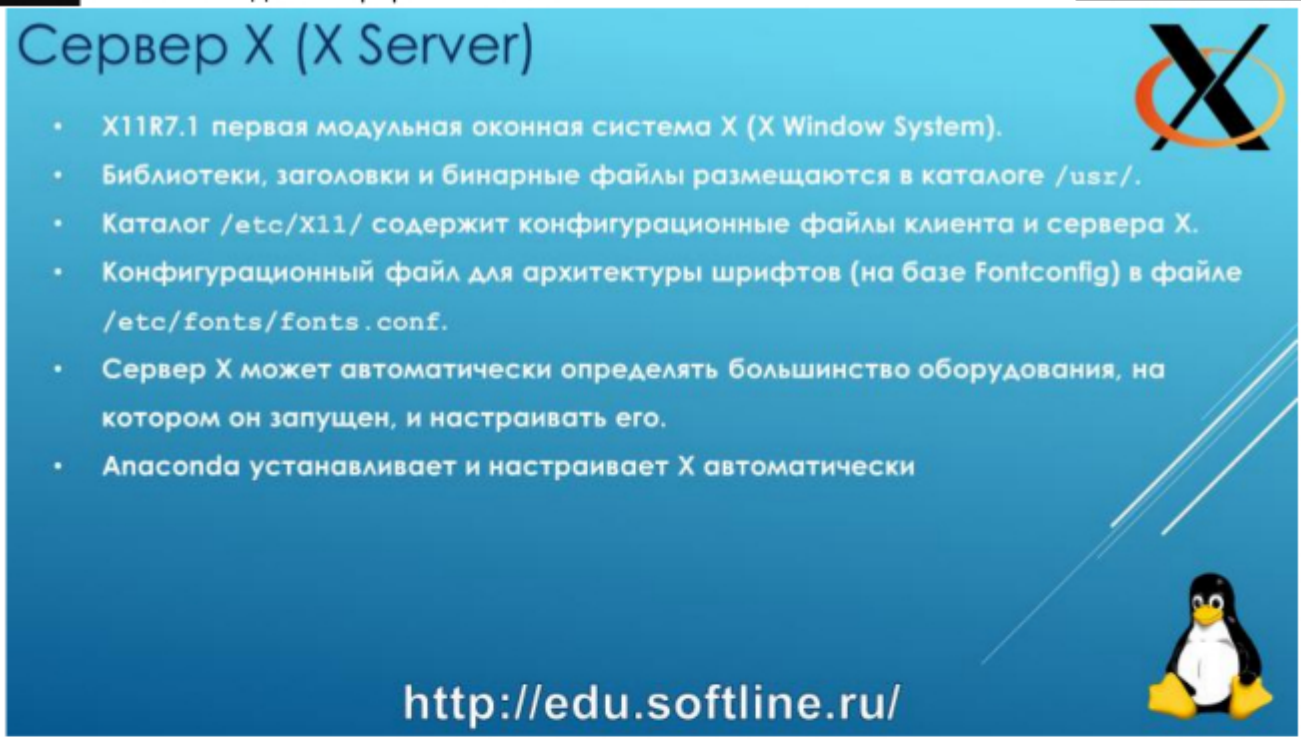


В то время, как ядро является сердцем CentOS, для многих пользователей, лицом операционной системы является графическое окружение, предоставляемое по средствам X Windows System, также называемое X.

Другие оконные окружения, существуют в мире UNIX, включая некоторые из тех, которые предшествовали созданию X Window System в июне 1984. Несмотря на это, X является графической оболочкой по умолчанию для большинства UNIX-подобных операционных систем, в том числе CentOS (RHEL) уже много лет.

Графическое окружение CentOS (RHEL) поставляется X.Org Foundation, организации, созданной для управления разработкой и стратегического планирования X Window System и связанных технологий. X.Org – это широко масштабируемый, быстро разрабатываемый проект с сотнями разработчиков по всему миру. Его возможности являются фундаментом для поддержки различных аппаратных устройств и архитектур, и запуска на тысячах операционных систем и платформ.

X Window System использует клиент-серверную архитектуру. Ее основное назначение заключается в предоставлении прозрачной сетевой оконной системы, поддерживающей запуска на широком диапазоне вычислительных и графических машин. X-сервер (бинарный Xorg) прослушивает в ожидании подключений от приложений клиентов X по сети или локально через интерфейс-петлю (Loopback), обеспечивает серверное взаимодействие с оборудованием, таким как: видеокарта, монитор, клавиатура и мышь. Приложения клиенты X существуют в пользовательском пространстве, создают графический пользовательский интерфейс (Graphical User Interface, GUI) для пользователей и передают пользовательские запросы серверу X.



Начиная с CentOS 6, CentOS (RHEL) использует версию X-сервера, которая включает в себя несколько видео драйверов, EXA и платформу, поддерживающую расширения, в отличии от предыдущих релизов. В дополнение, данный релиз включает несколько возможностей автоматической настройки для сервера X и общий драйвер ввода: **evdev**, который поддерживает большинство устройств ввода, о которых известно ядру, включая большинство клавиатур и мышей.

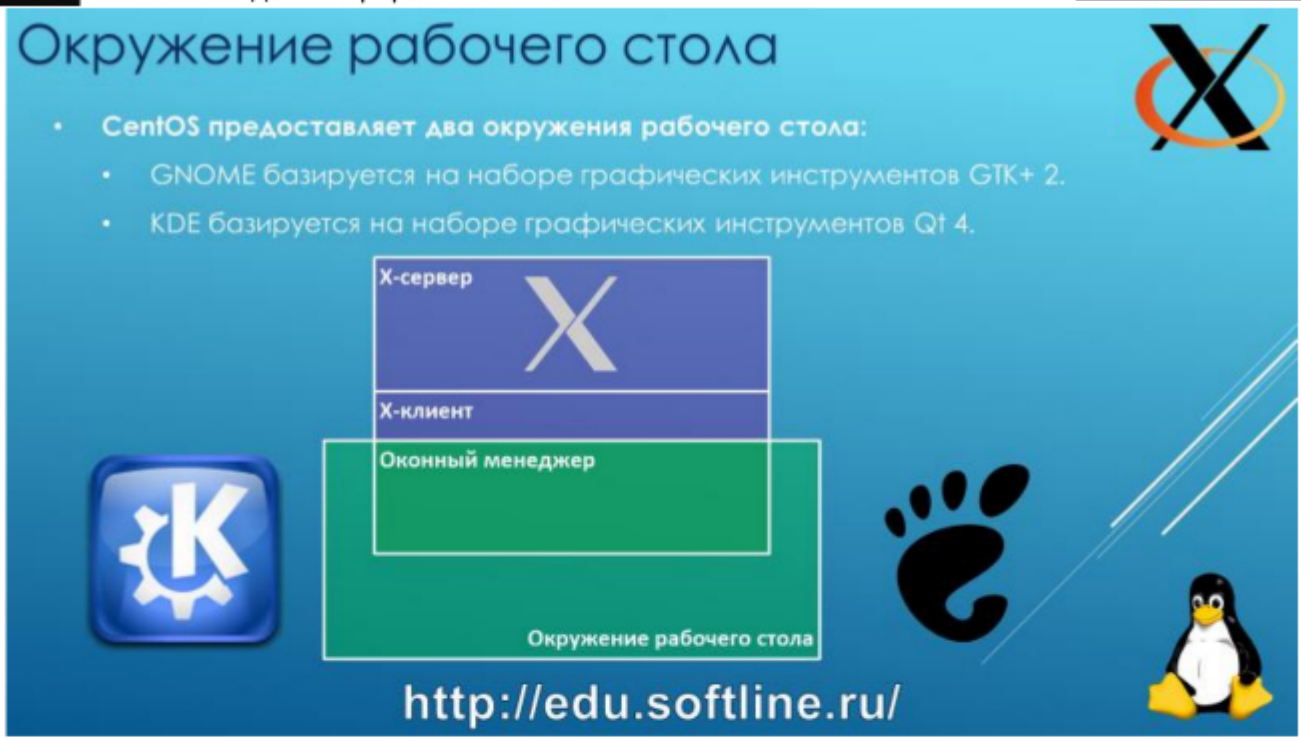
X11R7.1 является первым релизом, предоставляющим дополнительные преимущества, делая оконную систему X (X Window System) модульной. Данный релиз разделяет X на отдельные логические модули, что упрощает разработчикам открытого программного обеспечения встраивать код в систему.

В текущем релизе все библиотеки, заголовки и бинарные файлы размещаются в каталоге **/usr/**. Каталог **/etc/X11/** содержит конфигурационные файлы приложение клиенту и серверу X. Он содержит конфигурационные файлы сервера X, менеджера экрана X и множества прочих основных компонентов.

Конфигурационный файл для новой архитектуры шрифтов, основанной на **Fontconfig**, остался в файле **/etc/fonts/fonts.conf**.

В связи с тем, что сервер X выполняет продвинутые задачи на широком наборе оборудования, это требует детальной информации об оборудовании и работе с ним. Сервер X может автоматически определять большинство оборудования, на котором он запущен, и настраивать его соответственно. В качестве альтернативы, вы можете указать его в конфигурационном файле вручную.

Системный установщик CentOS – Anaconda устанавливает и настраивает X автоматически, за исключением случаев, когда пакет X не выбран для установки. Если случаются какие-либо изменения в мониторе, видеокарте или иных устройствах, которыми управляет сервер X, в большинстве случаев сервер X автоматически определяет их и перенастраивается. Только в очень редких случаях требуется перенастраивать X вручную.



Как только сервер X запущен, приложения клиенты X могут подключаться к нему и создавать GUI для пользователей. Набор GUI доступен в CentOS от независимого оконного менеджера, такого как metacity, до высокоуровневого интерактивного окружения рабочего стола GNOME и KDE, с которыми знакомы большинство пользователей CentOS.

Для дальнейшего создания более обширного GUI, два основных класса приложений клиентов X должны подключиться к серверу X: оконный менеджер (Window Manager) и окружение рабочего стола (Desktop Environment).

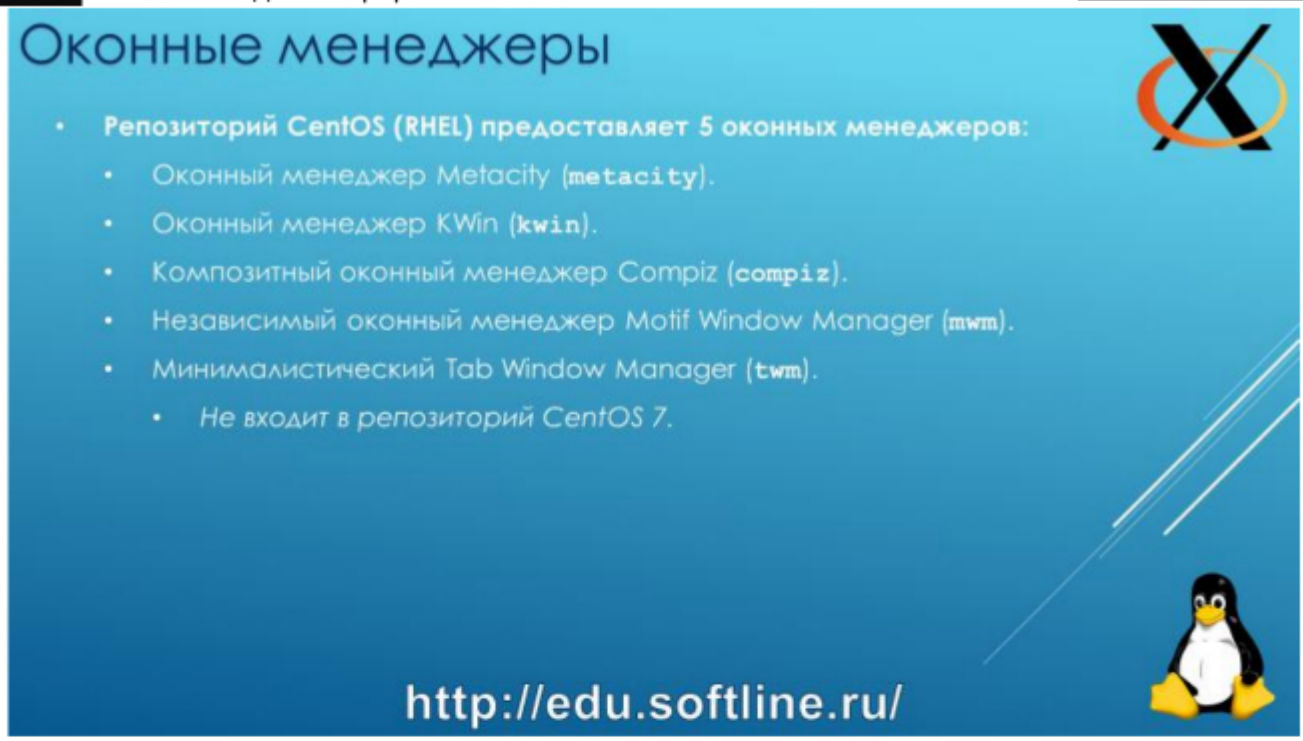
Окружения рабочего стола (Desktop Environments).

Окружение рабочего стола интегрирует различных клиентов X для создания общего графического пользовательского окружения и платформы разработки.

Окружения рабочего стола имеют расширенные возможности, позволяющие клиентам X и остальным запущенным процессам взаимодействовать между собой, выполняя расширенные задачи, такие как операции перетаскивания (Drag-and-drop).

CentOS предоставляет два окружения рабочего стола:

- GNOME – окружение рабочего стола по умолчанию для CentOS, базируется на наборе графических инструментов GTK+ 2.
- KDE – альтернативное окружение рабочего стола базируется на наборе графических инструментов Qt 4.



Оконные менеджеры – это клиенты X, которые являются частью окружения рабочего стола или, в некоторых случаях, независимы. Основное назначение – это управление позиционированием графических окон, изменением размеров или перемещением. Оконные менеджеры также управляют панелями заголовков, оконным фокусом и определенными пользовательскими привязки клавиш и кнопок мыши.

Репозиторий CentOS (RHEL) предоставляет 5 различных оконных менеджеров:

metacity

Оконный менеджер Metacity является оконным менеджером GNOME по умолчанию. Он является простым и эффективным оконным менеджером, который поддерживает настраиваемые темы. Этот оконный менеджер автоматически устанавливается, как зависимость при установке рабочего стола GNOME.

kwin

Оконный менеджер KWin – это оконный менеджер по умолчанию для KDE. Он является эффективным оконным менеджером, который поддерживает настраиваемые темы. Этот оконный менеджер автоматически устанавливается, как зависимость при установке рабочего стола KDE.

compiz

Композитный оконный менеджер Compiz, основанный на OpenGL, может использовать 3D-оборудование для создания быстрых композитных эффектов рабочего стола для оконного менеджера. Расширенные возможности, такие как кубический рабочий стол, применяются в качестве загружаемых плагинов. Для запуска этого оконного менеджера необходимо установить пакет **compiz**.

mwm

Motif Window Manager (**mwm**) – это независимый оконный менеджер, который разработан для независимой работы и не требует объединения с GNOME или KDE. Для запуска этого оконного менеджера необходимо установить пакет **openmotif**.

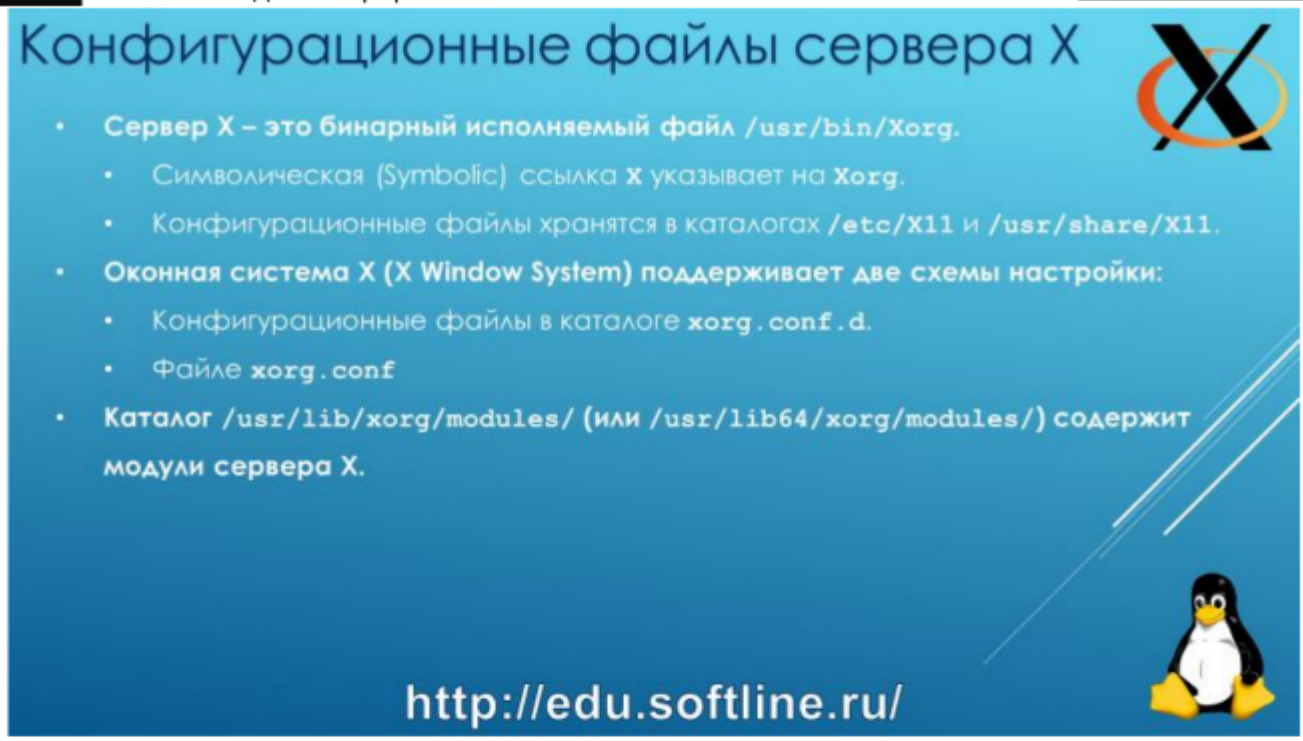
twm

Минималистический Tab Window Manager (**twm**), который предоставляет большинство базовых наборов утилит, доступных оконным менеджерам, может быт использован независимо или вместе с окружением рабочего стола. Для запуска этого оконного менеджера вам необходимо установить пакет **xorg-x11-twm**.



ПРИМЕЧАНИЕ:

Tab Window Manager не входит в дистрибутив и репозитории CentOS 7, но может быть установлен из репозитория или дистрибутива CentOS 6, это не гарантирует его стабильной работы, поэтому рекомендуется использовать Motif Window Manager.



Сервер X – это один бинарный исполняемый файл **/usr/bin/Xorg**; Символическая (Symbolic) ссылка **X** указывающая на **Xorg**, также предоставляется. Связанные конфигурационные файлы хранятся в каталогах **/etc/X11 и /usr/share/X11**.

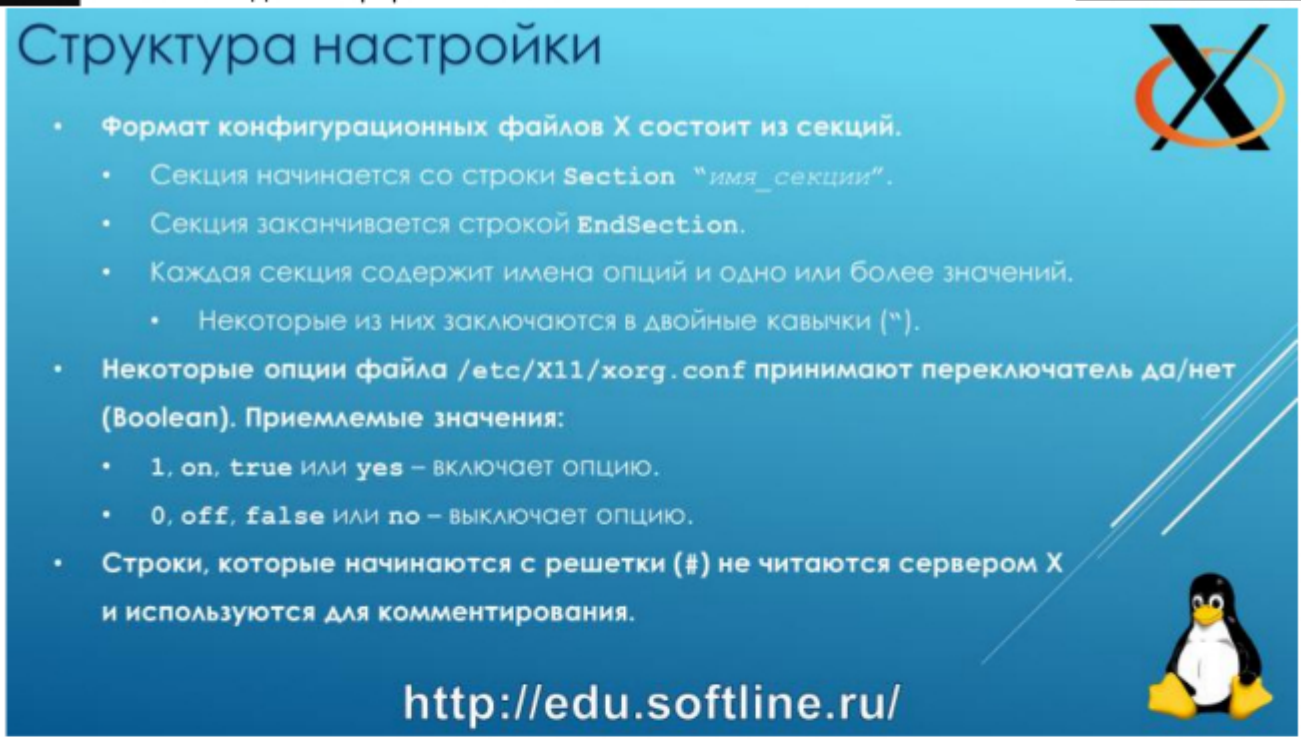
Оконная система X (X Window System) поддерживает две различных схемы настройки. Конфигурационные файлы в каталоге **xorg.conf.d** содержат заранее настроенные параметры от производителей и дистрибьюторов, эти файлы не должны редактироваться вручную. Настройка в файле **xorg.conf** – второй вариант настройки, выполняется полностью вручную, но как правило это не требуется.

ПРИМЕЧАНИЕ:

*Все необходимые параметры экрана и периферии автоматически определяются и настраиваются в процессе установки. Конфигурационный файл сервера X – **/etc/X11/xorg.conf**, который был необходим в предыдущих релизах, не предоставляется в текущем релизе X Window System. Может потребоваться создание файла вручную для настройки нового оборудования, настройки окружения с несколькими видеокартами или в целях отладки.*

Каталог **/usr/lib/xorg/modules/** (или **/usr/lib64/xorg/modules/**) содержит модули сервера X, которые могут быть загружены динамически в процессе работы. По умолчанию только некоторые модули в **/usr/lib/xorg/modules/** загружаются автоматически сервером X.

Когда CentOS 7 (RHEL 7) устанавливается, конфигурационные файлы для X создаются при помощи информации, собранной о системном оборудовании в процессе установки при помощи фоновой настройки HAL (уровень аппаратной абстракции). Каждый раз, когда сервер X запускается, он опрашивает HAL для получения списка устройств ввода и добавляет каждое с соответствующим драйвером. Когда новое устройство ввода подключается или существующее отключается, HAL уведомляет сервер X об изменении. Благодаря этой системе уведомления, устройства, использующие драйвера mouse, **kbd** или **vmmouse**, настроенные в файле **xorg.conf** по умолчанию, игнорируются сервером X. Дополнительные настройки, сделанные в каталоге **/etc/X11/xorg.conf.d/** перепишут настройки применяемы через HAL.



Формат конфигурационных файлов X состоят из множества различных секций, которые адресованы отдельным аспектам системного оборудования. Каждая секция начинается со строки Section "*имя_секции*", секция заканчивается строкой **EndSection**. Каждая секция содержит имена опций и одно или более значений. Некоторые из них заключаются в двойные кавычки (").

Некоторые опции файла **/etc/X11/xorg.conf** принимают переключатель да/нет (Boolean), которые включают компонент или отключают. Приемлемые значения:

- **1**, **on**, **true** или **yes** – включает опцию.
- **0**, **off**, **false** или **no** – выключает опцию.

Следующий пример показывает типичный конфигурационный файл для клавиатуры. Строки, которые начинаются с решетки (#) не читаются сервером X и используются для комментирования.

```
# Read and parsed by systemd-localed. It's probably wise not to edit this file
# manually too freely.
Section "InputClass"
        Identifier "system-keyboard"
        MatchIsKeyboard "on"
        Option "XkbLayout" "us"
EndSection
```



Сервер X поддерживает два каталога настройки. Каталог **/usr/share/X11/xorg.conf.d/** предоставляет отдельные конфигурационные файлы от производителей. Изменения в файлах этого катала могут быть переписаны параметрами, указанными в файле **/etc/X11/xorg.conf**. Каталог **/etc/X11/xorg.conf.d/** хранит относящиеся к пользователю настройки.

Файлы, заканчивающиеся на **.conf** в каталогах настройки разбираются сервером X во время запуска, подобно частям традиционного конфигурационного файла **xorg.conf**. Эти файлы могут содержать одну или более секций.



В предыдущих релизах X window System, файл **/etc/X11/xorg.conf** использовался для хранения начальной настройки для X. Когда происходит изменение монитора, видеокарты или иного устройства, управляемого сервером X необходимо вносить изменение вручную. В CentOS не часто приходится создавать и редактировать файл **/etc/X11/xorg.conf**. Несмотря на это, очень важно понимать различные секции и опциональные параметры доступные специально для устранения неисправностей или настройки необычного оборудования.

Процедура создания xorg.conf.

Если необходимо внести изменения в конфигурацию X вручную, в первую очередь потребуется создать конфигурационный файл **/etc/X11/xorg.conf**.

Для создания конфигурационного файла, воспользуйтесь командой:

```
# Xorg :1 -configure
```

Это создать файл **/root/xorg.conf**.new, который вы затем можете скопировать в **/etc/X11/xorg.conf**:

```
# cp /root/xorg.conf.new /etc/X11/xorg.conf
```

Далее описаны важные секции в том же порядке, как они применяются в файле **/etc/X11/xorg.conf**. Больше подробной информации о конфигурационном файле сервера X можно найти на странице руководства man: **xorg.conf**(5).

Секция InputClass.

InputClass – это новый тип конфигурационной секции, который не применяется к отдельному устройству, вместо этого применяется к целому классу устройств, включая устройства, подключаемые на горячую. В секции **InputClass** область действия ограничена при помощи указанных соответствий, в

порядке применения к устройствам ввода, все соответствия должны быть применены к устройствам ввода, как показано в примере ниже:

```
Section "InputClass"
   Identifier       "touchpad catchall"
   MatchIsTouchpad  "on"
   Driver           "synaptics"
EndSection
```

В этом примере любому сенсорному экрану, представленному в системе, будет назначен драйвер **synaptics**.

Опции соответствия указывают, к каким устройствам могут примениться установки секции.

Популярные опции секции **InputClass**:

- **MatchIsPointer**, **MatchIsKeybord**, **MatchIsTouchpad**, **MatchIsTouchscreen**, **MatchIsJoystick** – булевская переменная, указывает тип устройства.
- **MatchProduct** "*имя_продукта*" – эта опция соответствует, если строка *имя_продукта* присутствует в имени продукта устройства.
- **MatchVendor** "*имя_производителя*" – эта опция соответствует, если строка имя_производителя присутствует в имени производителя устройства.
- **MatchDevicePath** "*/путь/к/устройству*" – эта опция соответствует любому устройству, если путь к устройству соответствует указанному шаблону, например **/dev/input/event***. Дополнительную информацию смотрите на странице руководства **fnmatch**(3).
- **MatchTag** "*шаблон_тэга*" – эта опция соответствует, если соответствует хотя бы один из тэгов, назначенных HAL.

Конфигурационный файл может иметь несколько секций **InputClass**. Эти секции опциональны и используются для настройки классов устройств ввода по мере их добавления. Одно устройство может соответствовать нескольким секциям **InputClass**. В порядке выполнения каждая последующая секция **InputClass** будет переписывать параметры предыдущей, если они определены. Поэтому старайтесь располагать более универсальные секции в начале.

Секция InputDevice.

Каждая секция **InputDevice** настраивает одно устройство ввода для сервера X. Ранее, системы обычно использовали как минимум одну секцию под клавиатуру, и большинство параметров мыши определялось автоматически.

Начиная с CentOS 6, настройка секции **InputDevice** не требуется в большинстве случаев, и пакет драйверов ввода **xorg-x11-drv-*** предоставляет автоматическую настройку через HAL. По умолчанию для клавиатуры и мыши используется драйвер **evdev**.

Следующий пример показывает стандартную секцию для клавиатуры:

```
Section "InputDevice"
  Identifier "Keyboard0"
  Driver "kbd"
  Option "XkbModel" "pc105"
  Option "XkbLayout" "us"
EndSection
```

Популярные опции секции **InputClass**:

- **Identifier** - указывает уникальное имя для секции **InputDevice**. Является обязательной опцией.
- **Driver** – указывает имя драйвера устройства, который X должен загрузить. Если опция **AutoAddDevice** включена (по умолчанию включена), любая секция устройств ввода с **Driver "mouse"** или **Driver "kbd"** будет игнорироваться. Это необходимо, для того чтобы избежать конфликтов между устаревшими драйверами мыши и клавиатуры с универсальным драйвером **evdev**. Любая ручная настройка устройств ввода должна быть перенесена из **xorg.xonf** в бэкэнд. В большинстве случае бэкэндом будет HAL и конфигурация должна располагаться в каталоге **/ext/X11/xorg.conf.d**.
- **Option** – Указывает необходимые опции относящиеся к устройству.

Мышь, может также быть указана для перезаписи любых значений для устройства, определенных автоматически. Следующие опции, как правило указываются, когда добавляют мышь в файл **xorg.conf**:

- **Protocol** – указывает протокол, используемый мышью, такой как **IMPS/2**.
- **Device** – Указывает расположение физического устройства.
- **Emulate3Buttons** – Позволяет указать режим работы двухкнопочной мыши как трехкнопочной, когда одновременно нажимаются обе кнопки.

Для получения полного списка доступных опций этой секции обратитесь к странице руководства: **xorgh.conf(5)**.

Секция ServerFlags.

Опциональная секция **ServerFlags** содержит прочие глобальные настройки сервера X. Любые настройки данной секции могут быть переписаны опциями, размещенными в секции **ServerLayout**.

Каждая запись в секции **ServerFlags** занимает одну строку и начинается с термина Option сопровождающегося опцией, заключенной в двойные кавычки (").

Пример секции **ServerFlags**:

```
Section "ServerFlags"
   Option "DontZap" "true"
EndSection
```

Список наиболее полезных опций:

- **"DontZap"** "*boolean*" – когда значение установлено в **true**, этот параметр предотвращает использование сочетания клавиш **<Ctrl>+<Alt>+<Backspace>** для немедленного уничтожения сервера X.

ПРИМЕЧАНИЕ:

*Даже если опция **DontZap** включена, комбинация клавиш должна быть сконфигурирована в карте X Keyboard Extenstion (XKB), прежде чем она может быть использована. Чтобы добавить комбинацию клавиш можно воспользоваться следующей командой:*

```
# setxkbmap -option "terminate:ctrl_alt_bksp"
```

- **"DontZoom"** *"boolean"* – когда значение установлено в **true**, эта настройка предотвращает циклическое переключение между настроенными разрешениями видео при помощи комбинаций клавиш **<Ctrl>+<Alt>+<Keypad-Plus>** и **<Ctrl>+<Alt>+<Keypad-Minus>**.
- **"AutoAddDevices"** *"boolean"* – когда значение установлено в **false**, сервер не подключает устройства ввода на горячую, вместо этого полагается на устройства, сконфигурированные в файле **xorg.conf**. Данная опция включена по умолчанию, а для фонового обнаружения устройств используется HAL (уровень аппаратной абстракции).

Секция ServerLayout.

Секция **ServerLayout** связывает вместе устройства ввода и вывода, управляемые сервером X. Как минимум данная секция должна указывать одно устройство ввода и одно устройство вывода. По умолчанию устройство вывода – монитор, устройство ввода – клавиатура.

Пример обычной секции **ServerLayout**:

```
Section "ServerLayout"
  Identifier "Default Layout"
  Screen 0 "Screen0" 0 0
  InputDevice "Mouse0" "CorePointer"
  InputDevice "Keyboard0" "CoreKeyboard"
EndSection
```

Список наиболее часто используемых опций:

- **Identifier** – указывает уникальное имя секции **ServerLayout**.
- **Screen** – указывает имя секции **Screen** для использования с сервером X. В конфигурационном файле может быть представлено несколько секций **Screen**.

Пример обычной записи **Screen**:

```
Screen 0 "Screen0" 0 0
```

Первое число в примере записи **Screen** определяет первое подключение к монитору или выход (head) на видеокарте, использующее настройку, определенную в секции **Screen** с идентификатором **"Screen0"**.

Если у видеокарты несколько выходов (heads), необходима еще одна запись **Screen** с другим числом и отличающимся идентификатором секции **Screen**.

Числа справа от **"Screen0"** дают абсолютные координаты X и Y, определяющие верхний левый угол экрана (по умолчанию **0 0**).

- **InputDevice** – указывает имя секции **InputDevice**, которая будет использоваться сервером X.

Очевидно, что будет как минимум две записи **InpuDevice**: одна для мыши по умолчанию и одна для клавиатуры по умолчанию. Опции **CorePointer** и **CoreKeyboard** определяют основные мышь и клавиатуру. Если опция **AutoAddDevices** включена, эти записи не должны быть указаны в секции **ServerLayout**. Если опция **AutoAddDevices** отключена - мышь и клавиатура автоматически определяются со значениями по умолчанию.

- **Option** "*option-name*" – опциональная запись, которая указывает дополнительные параметры для секции. Любые опции указанные в этой секции переписывают свои определения в секции **ServerFlags**.

Доступные опции (*option-name*) для данной секции перечислены на странице руководства **xorg.conf(5)**.

Возможно разместить несколько секций **ServerLayout** в файле **/etc/X11/xorg.conf**. По умолчанию, сервер читает только первую найденную секцию. Однако, альтернативная секция **ServerLayout** может быть указана в качестве аргумента командной строки при запуске сессии X командой: **Xorg -layout** *layoutname*.

Секция Files.

Секция **Files** устанавливает пути для жизненно важных служб сервера X, таких как путь к шрифтам. Это опциональная секция, так как эти пути обычно определяются автоматически. Эта секция может быть использована для перезаписи автоматически определенных значений.

Пример секции **Files**:

```
Section "Files"
 RgbPath "/usr/share/X11/rgb.txt"
 FontPath "unix/:7100"
EndSection
```

Следующая запись как правило используют в секции **Files**:

- **ModulePath** – опциональный параметр, который указывает альтернативные каталоги с модулями сервера X.

Секция Monitor.

Каждая секция **Monitor** настраивает один тип монитора, используемого системой. Секция является опциональной, так как большинство мониторов сейчас определяются автоматически.

Пример обычной секции **Monitor** для монитора:

```
Section "Monitor"
 Identifier "Monitor0"
 VendorName "Monitor Vendor"
 ModelName "DDC Probed Monitor - ViewSonic G773-2"
 DisplaySize 320 240
 HorizSync 30.0 - 70.0
 VertRefresh 50.0 - 180.0
EndSection
```

Следующие записи как правило используются в секции **Monitor**:

- **Identifier** – указывает уникальное имя секции Monitor. Это обязательная запись.
- **VendorName** – опциональный параметр, который указывает производителя монитора.
- **ModelName** – опциональный параметр, который указывает модель монитора.
- **DisplaySize** – опциональный параметр, который указывает физически размер области изображения монитора в миллиметрах.

- **HorizSync** – Указывает в kHz диапазон горизонтальной частоты синхронизации, совместимый с монитором. Данные значения помогают серверу X определить корректность встроенных или указанных записей **Modeline** для монитора.
- **VertRefresh** – Указывает в kHz диапазон вертикальной частоты обновления, совместимый с монитором. Данные значения помогают серверу X определить корректность встроенных или указанных записей **Modeline** для монитора.
- **Modeline** – опциональный параметр, указывающий дополнительные режимы видео для монитора в конкретных разрешениях, с определенными горизонтальной синхронизацией и вертикальной частотой обновления. Подробная информация о записях **Modeline** доступна на странице руководства **xorg.conf(5)**.
- **Option** "*option-name*" – опциональная запись, которая указывает дополнительные параметры для секции. Доступные опции (*option-name*) для данной секции перечислены на странице руководства **xorg.conf(5)**.

Секция Device.

Каждая секция **Device** настраивает одну видеокарту на системе. Минимально должна присутствовать одна секция **Device**, но для каждой установленной на машине.

Пример обычной секции **Device**:

```
Section "Device"
 Identifier "Videocard0"
 Driver "mga"
 VendorName "Videocard vendor"
 BoardName "Matrox Millennium G200"
 VideoRam 8192
 Option "dpms"
EndSection
```

Следующие записи обычно используются в секции **Device**:

- **Identifier** – указывает уникальное имя для секции **Device**. Это обязательная запись.
- **Driver** – Указывает какой драйвер должен загрузить сервер X для использования видеокарты. Список драйвер можно найти в **/usr/share/hwdata.videodrivers**, который устанавливается вместе с пакетом **hwdata**.
- **VendorName** – Опциональный параметр, который указывает производителя видеокарты.
- **BoardName** – Опциональный параметр, который указывает имя видеокарты.
- **VideoRam** – Опциональный параметр, который указывает объем оперативной памяти в килобайтах доступный видеокарте. Данный параметр необходим только в том случае, когда сервер X не может самостоятельно определить объем видео памяти.
- **BusID** – запись, указывающая расположение шины видеокарты. На машинах с одной видеокартой запись **BusID** опциональна и может не быть представлена в файле **/etc/X11/xorg.conf** по умолчанию. Однако на машинах с несколькими видеокартами запись **BusID** обязательна.
- **Screen** – опциональная запись, указывающая какое подключение монитора или выход на видео карте настраивает секция **Device**.

Если несколько мониторов подключены к разным выходам на одной видео карте, должны присутствовать отдельные секции **Device,** и каждая секция должна иметь собственное значение **Screen**.

Значения записи **Screen** должны быть целыми числами. Первый выход на видео карте имеет значение **0**. Каждое следующее значение увеличивается на единицу.

- **Option** "*option-name*" – опциональная запись, которая указывает дополнительные параметры для секции. Доступные опции (*option-name*) для данной секции перечислены на странице руководства **xorg.conf(5)**.

Одна из наиболее популярных опций – это "**dpms**" (для Display Power Management Signaling – стандарт VESA), который активирует совместимые с Energy Star настройки электропитания для монитора.

Секция Screen.

Каждая секция **Screen** привязывает одну видеокарту (или выход на видеокарте) к одному монитору, ссылаясь на секции **Monitor** и **Device**. Присутствует как минимум одна секция **Screen**. Как минимум должна присутствовать одна секция **Screen**, дополнительные экземпляры будут присутствовать для каждой комбинации видео карты и монитора подключенных к машине.

Пример секции **Screen**:

```
Section "Screen"
 Identifier "Screen0"
 Device "Videocard0"
 Monitor "Monitor0"
 DefaultDepth 16

 SubSection "Display"
      Depth 24
      Modes "1280x1024" "1280x960" "1152x864" "1024x768" "800x600" "640x480"
 EndSubSection

 SubSection "Display"
      Depth 16
      Modes "1152x864" "1024x768" "800x600" "640x480"
 EndSubSection
EndSection
```

Следующие записи обычно используются в секции **Screen**:

- **Identifier** – Указывает уникальное имя секции **Screen** (Обязательная запись).
- **Device** – Указывает уникальное имя секции **Device** (Обязательная запись).
- **Monitor** – Указывает уникальное имя секции **Monitor**. Обязательной запись становится только при наличии специфической секции **Monitor** в файле **xorg.conf**. Обычно, мониторы определяются автоматически.
- **DefaultDepth** – Указывает глубину цвета в битах по умолчанию. Разрешено использовать только одну запись **DefaultDepth**, также она может быть переписана опцией командной строки **Xorg –depth** *n*, где *n* – указывает глубину.
- **SubSection "Display"** – указывает доступные режимы экрана для указанной глубины цвета. Секция **Screen** может содержать множество подсекций **Display**. Подсекции **Display** опциональны, так как режимы экрана определяются автоматически.

- **Option** "*option-name*" – опциональная запись, которая указывает дополнительные параметры для секции. Доступные опции (*option-name*) для данной секции перечислены на странице руководства **xorg.conf(5)**.

Секция DRI.

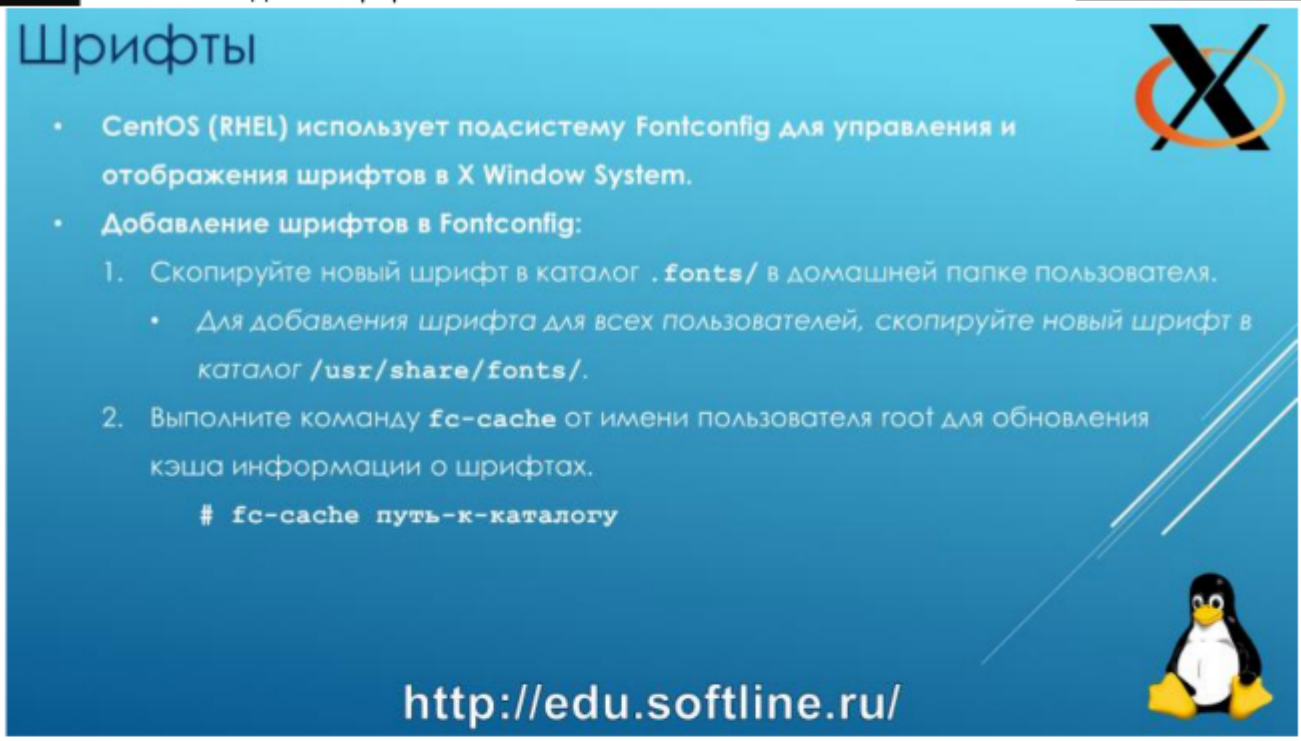
Опциональная секция **DRI** указывает параметры для Direct Rendering Infrastructure (DRI). DRI – это интерфейс, который позволяет программному обеспечению 3D приложений получать преимущества от возможностей аппаратного ускорения 3D, встроенного в большинство современного видеооборудования. Дополнительно, **DRI** может улучшить производительность 2D за счет аппаратного ускорения, если это поддерживается драйвером видеокарты.

Данная секция редко используется, так как группа (Group) и режим (Mode) DRI автоматически инициализируется в значения по умолчанию. Если необходимы разные группа (Group) и режим (Mode), то нужно добавить секцию **DRI** в файл **xorg.conf** для перезаписи значений по умолчанию.

Пример секции **DRI**:

```
Section "DRI"
Group 0
Mode 0666
EndSection
```

Так как разные видеокарты используют DRI по-разному, не добавляйте данную секцию без обращения к https://dri.freedesktop.org/wiki/.



CentOS (RHEL) использует подсистему Fontconfig для управления и отображения шрифтов в X Window System. Это упрощает управление шрифтами и предоставляет расширенные возможности отображения, такие как анти-совмещение имен (anti-aliasing). Данная система используется автоматически для приложений написанных с использованием графических наборов инструментов Qt 3 или GTK+ 2 или их более новых версий.

Подсистема шрифтов Fontconfig позволяет приложениям напрямую получать доступ к шрифтам на системе и использовать X FreeType interface library (Xft) или иные механизмы рендеринга для рендеринга шрифтов Fontconfig с расширенными возможностями, такими как анти-совмещение имен (anti-aliasing). Графические приложения могут использовать библиотеку Xft вместе с Dontconfig для отображения текста на экране.

ПРИМЕЧАНИЕ:
Fontconfig использует конфигурационный файл ***/etc/fonts/fonts.conf****, который не должен редактироваться вручную.*

ВНИМАНИЕ:
Любая система, где пользователь собирается запускать удаленные приложения X, должна обладать установленной группой ***fonts****. Это можно сделать при помощи следующей команды:*

```
# yum groupinstall fonts
```

Добавление шрифтов в Fontconfig.

Добавление нового шрифта в подсистему Fontconfig – это простой процесс:

1. Для добавления шрифта для отдельного пользователя, скопируйте новый шрифт в каталог .fonts/ в домашней папке пользователя.

Для добавления шрифта для всех пользователей, скопируйте новый шрифт в каталог /usr/share/fonts/. Для организации новых шрифтов рекомендуется создать новый подкаталог, например, local/.

2. Выполните команду fc-cache от имени пользователя root для обновления кэша информации о шрифтах.

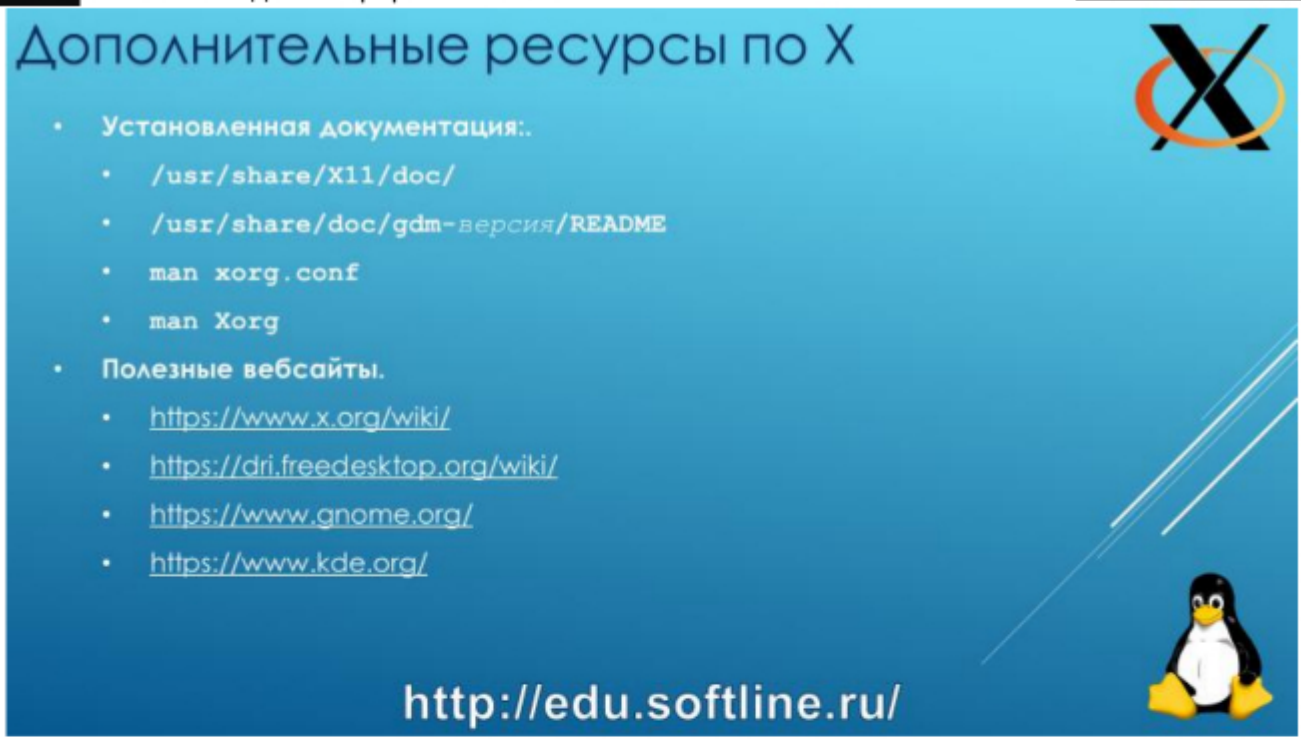
```
# fc-cache путь-к-каталогу
```

Необходимо заменить *путь-к-каталогу* каталогом содержащим новые шрифты, например (/usr/share/fonts/local или /home/student/.fonts/).



ПРИМЕЧАНИЕ:

Отдельные пользователи могут также установить шрифты интерактивно, введя в адресной строке Nautilus ***fonts:///*** *и переместив туда файлы новых шрифтов.*



Существует достаточно много подробной информации о сервере X, клиентах, которые подключаются к нему и связанному окружению рабочего стола с оконным менеджером.

Установленная документация.

- **`/usr/share/X11/doc/`** - содержит подробную документация по архитектуре X Window System, а также информацию о способах получения дополнительной информации о проекте Xorg.
- **`/usr/share/doc/gdm-`*`версия`*`/README`** – содержит информацию о том, как менеджеры экрана управляют аутентификацией пользователя.
- **`man xorg.conf`** – содержит информацию о конфигурационном файле **xorg.conf**, в том числе назначение и синтаксис различных секций файла.
- **`man Xorg`** – содержит описание сервера экрана Xorg.

Полезные вебсайты.

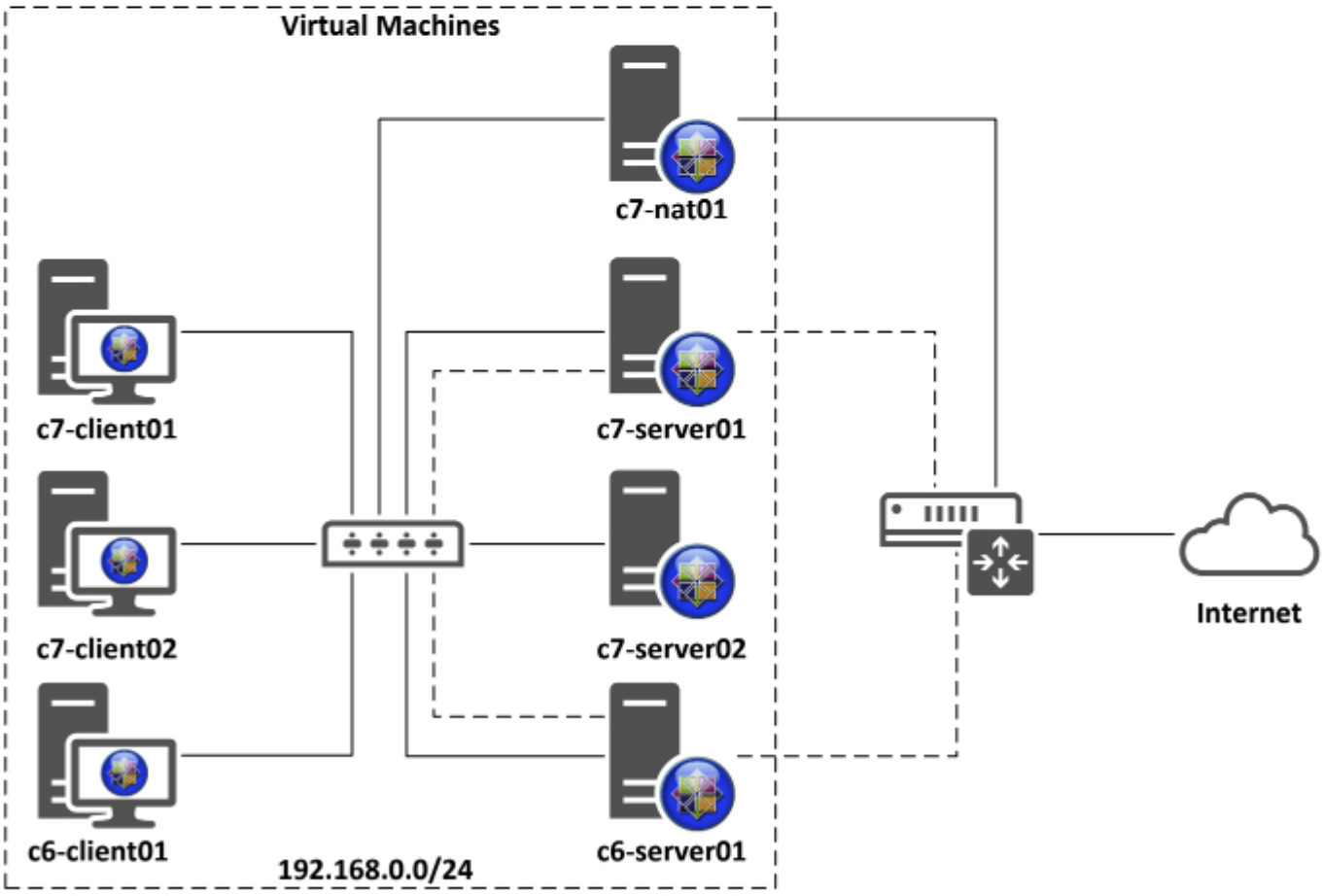
- https://www.x.org/wiki/ – домашняя страница X.Org Foundation, который выпускает основные релизы X Window System, встраиваемые в CentOS (RHEL) для управления необходимым оборудованием и предоставления окружения графического пользовательского интерфейса.
- https://dri.freedesktop.org/wiki/ – домашняя страница проекта DRI (Direct Rendering Infrastructure). DRI это основной компонент X для ускорения аппаратного 3D.
- https://www.gnome.org/ – домашняя страница проекта GNOME.
- https://www.kde.org/ – домашняя страница окружения рабочего стола KDE.

Лабораторная работа

Лабораторная работа 10

- **Упражнения:**
  - Установка X Window System и оконного менеджера MWM.
  - Установка и настройка окружения рабочего стола Xfce.
- **Необходимые машины:**
  - `c7-nat01`.
- **Учетные данные:**
  - Пользователь: `root`.
  - Пароль: **`Pa$$w0rd`**.

http://edu.softline.ru/





Упражнение 1: Установка X Window System и оконного менеджера Motif Window Manager (MWM).

1. На машине **c7-server01** установите группу пакетов '**X Window System**'.
2. Установите пакеты, необходимые для запуска оконного менеджера TWM:
   - **bitmap-fonts**
   - **gdm glx-utils**
   - **synaptics**
   - **vnc-server**
   - **xorg-x11-apps**
   - **xorg-x11-drivers**
   - **xorg-x11-fonts-100dpi**
   - **xorg-x11-fonts-75dpi**
   - **xorg-x11-fonts-ISO8859-1-100dpi**
   - **xorg-x11-fonts-ISO8859-1-75dpi**
   - **xorg-x11-fonts-Type1**
   - **xorg-x11-fonts-misc**
   - **xorg-x11-server-Xorg**
   - **xorg-x11-xauth**
   - **xorg-x11-xinit**
   - **xterm**



ПРИМЕЧАНИЕ:
Список необходимых пакетов можно получить у инструктора.
По умолчанию это: http://URL/files/mwm_pakages.txt.

3. Установите пакет оконного менеджера TWM (**openmotif**).
4. Перезагрузите машину.
5. Запустите X Window System.
6. Запустите оконный менеджер MWM.
7. Зажав правой кнопкой мыши на черном экране, откройте новое окно Xterm.
8. Проверьте работоспособность системы X Windows System при помощи приложения **xclock**.
9. Установите веб-браузер Mozilla Firefox (**firefox**).
10. Запустите веб-браузер Mozilla Firefox и проверьте возможность работы в интернете.



Ответы к упражнению 1: Установка X Window System и оконного менеджера Motif Window Manager (MWM).

1. На машине **c7-server01** установите группу пакетов '**X Window System**'.

```
# yum groupinstall 'X Window System'
```

2. Установите пакеты, необходимые для запуска оконного менеджера MWM.

```
yum install \
bitmap-fonts \
gdm glx-utils \
synaptics \
vnc-server \
xorg-x11-apps \
xorg-x11-drivers \
xorg-x11-fonts-100dpi \
xorg-x11-fonts-75dpi \
xorg-x11-fonts-ISO8859-1-100dpi \
xorg-x11-fonts-ISO8859-1-75dpi \
xorg-x11-fonts-Type1 \
xorg-x11-fonts-misc \
xorg-x11-server-Xorg \
xorg-x11-xauth \
xorg-x11-xinit \
xterm
```



ПРИМЕЧАНИЕ:
Список необходимых пакетов доступен по ссылке:
http://URL/files/mwm_pakages.txt.
Уточните ссылку у инструктора.

3. Установите пакет оконного менеджера MWM (**openmotif**).

```
# yum install openmotif
```

4. Перезагрузите машину.

```
# reboot
```

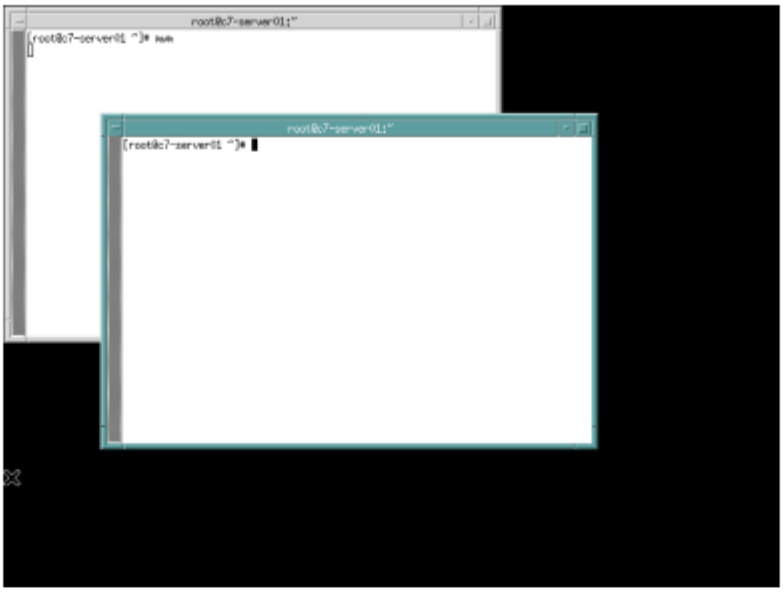
5. Запустите X Window System.

```
# xinit
```

6. Запустите оконный менеджер MWM.

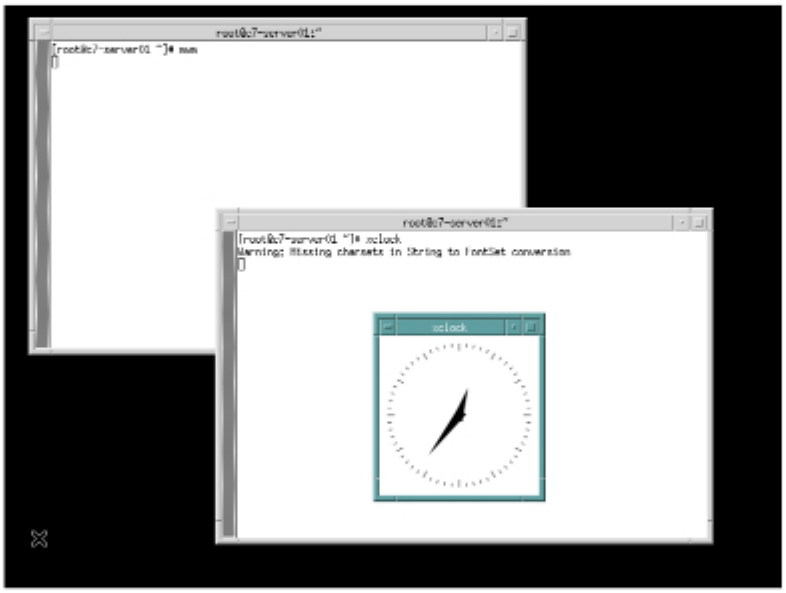
```
# mwm
```

7. Зажав левой кнопкой мыши на черном экране, откройте новое окно Xterm (New Window).



8. Проверьте работоспособность системы X Windows System при помощи приложения **xclock**.

```
# xclock
```

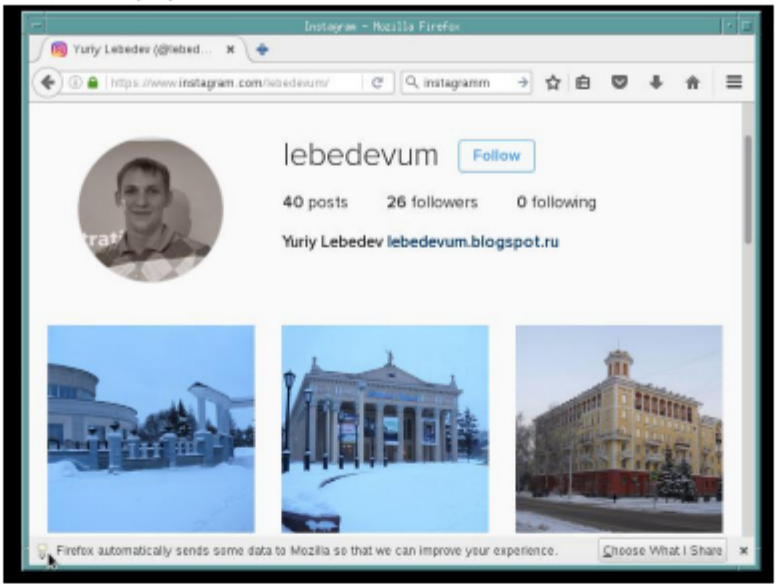


9. Установите веб-браузер Mozilla Firefox (**firefox**).

```
# yum install firefox
```

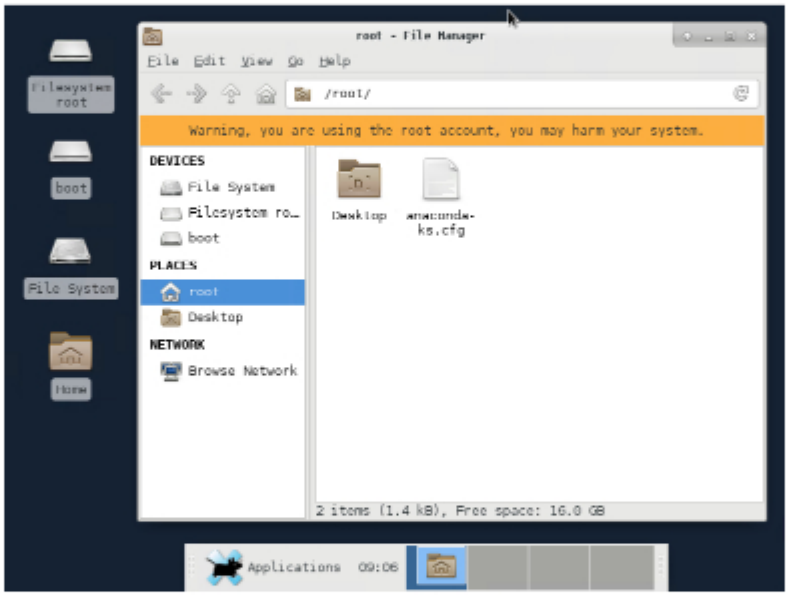
10. Запустите веб-браузер Mozilla Firefox и проверьте возможность работы в интернете.

```
# firefox
```


Instagram - Mozilla Firefox
Yuriy Lebedev (@lebed...
https://www.instagram.com/lebedevum/
instagramm
lebedevum
Follow
40 posts
26 followers
0 following
Yuriy Lebedev lebedevum.blogspot.ru
Firefox automatically sends some data to Mozilla so that we can improve your experience.
Choose What I Share

Упражнение 2: Установка и настройка окружения рабочего стола Xfce.

1. Запустите виртуальную машину **c7-server02**.
2. Установите группу пакетов «**`X Window system`**».
3. Подключите репозиторий EPEL.
4. Установите группу пакетов «**`Xfce`**».
5. Смените таргет по умолчанию на графический (**`graphical.target`**)
6. Перезагрузите систему.
7. Убедитесь, что окружение рабочего стола успешно установлено и готово к использованию.



Ответы к упражнению 2: Установка и настройка окружения рабочего стола Xfce.

1. Запустите виртуальную машину **c7-server02**.
2. Установите группу пакетов «**X Window system**».

```
# yum group install "X Window system" -y
```

3. Подключите репозиторий EPEL.

```
# yum install epel-release -y
```

4. Установите группу пакетов «**Xfce**».

```
# yum group install "Xfce" -y
```

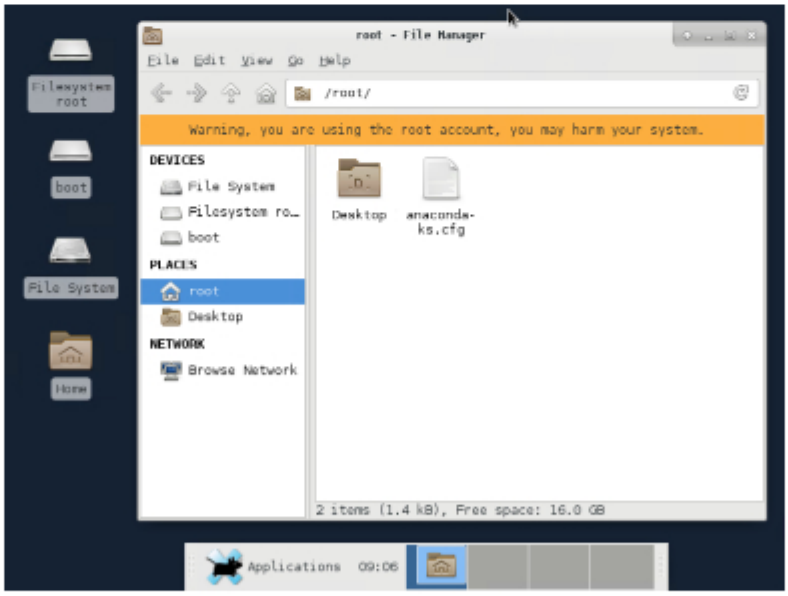
5. Смените таргет по умолчанию на графический (**graphical.target**)

```
# systemctl set-default graphical.target
```

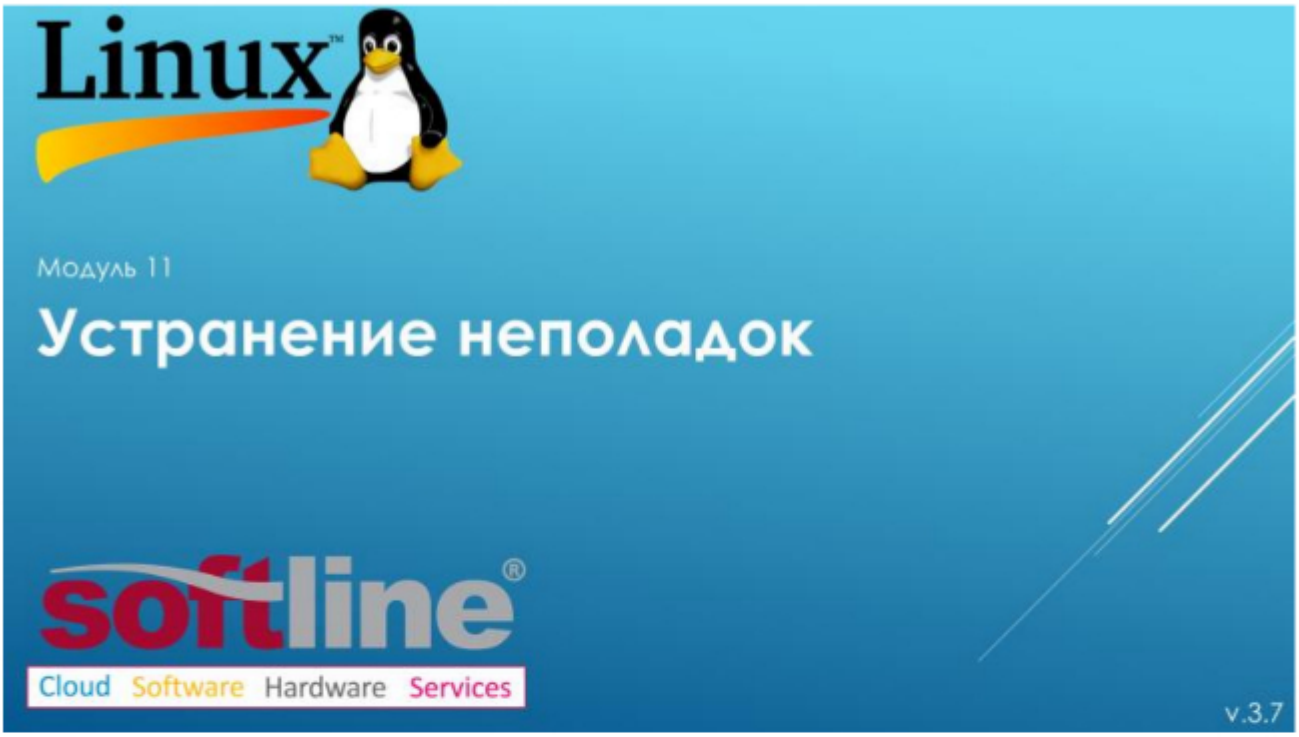
6. Перезагрузите систему.

```
# reboot
```

7. Убедитесь, что окружение рабочего стола успешно установлено и готово к использованию.

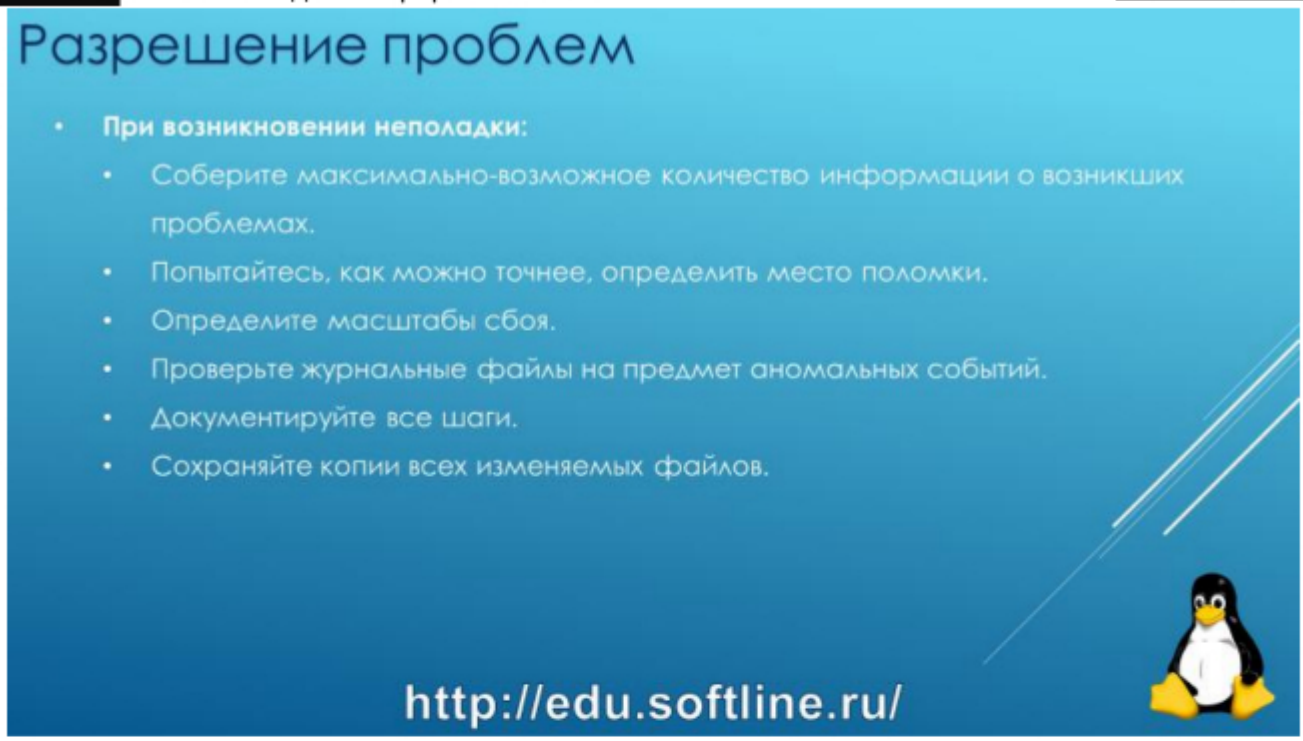


Модуль 11: Устранение неполадок



В модуле представлены:

- Возникающие проблемы и методика их решения.
- Использование среды восстановления системы.

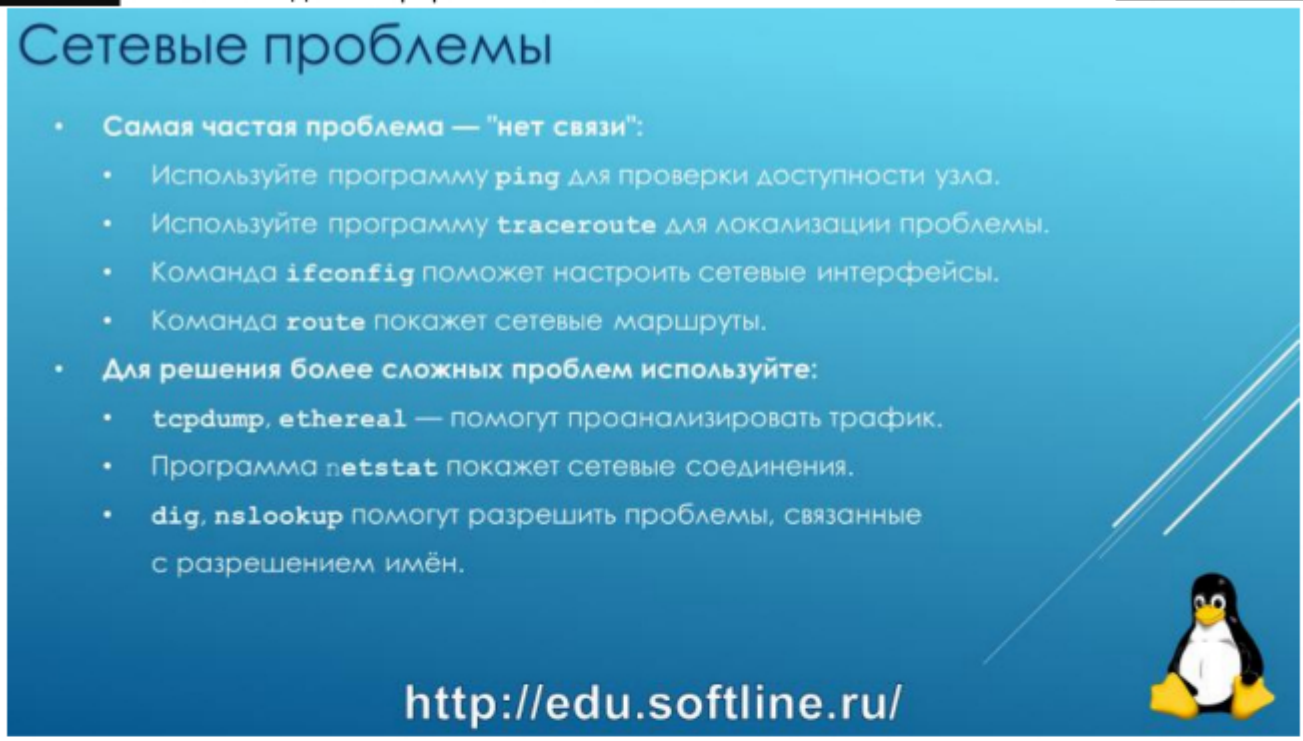


Linux — очень сложная система и подвергается правилу "надёжность системы зависит от её сложности", поэтому иногда случаются сбои в работе. Сбои бывают разные и приводят к разным результатам, начиная от неудобства работы охранника и заканчивая остановкой деятельности всего предприятия. Естественно, что чем грамотнее вы строили систему, тем она будет менее подвержена крупным сбоям, иначе вы рискуете получить риск масштабного краха даже при возникновении незначительных проблем в работе системы.

Устранение и поиск неполадок — искусство. Чтобы найти проблему, требуется не только хорошее понимание процессов, происходящих в системе, но и отличная смекалка и везение. Сложность поиска неисправности усугубляется тем, что система может не выполнять своих функций, а следовательно, что-то в организации не будет работать и вы получите начальника, стоящего в дверях с суровым выражением лица. Время — главный враг системного администратора, при устранении неполадок. Всегда старайтесь досконально разобраться в проблеме, но если организации грозят большие убытки, то постарайтесь в минимальные сроки восстановить работоспособность, пусть это будет временное и некрасивое решение, главное дать людям возможность работать, когда вы это сделаете, тогда появится время на изучение проблемы и более красивого её решения.

Проблемы могут носить самый разнообразный характер, это может быть и неправильная конфигурация, и сбой аппаратной части, это может быть повреждённая файловая система, как следствие удара молнии. Всё что угодно, администратор должен быть готов к самым большим неожиданностям. Иногда проблема бывает вызвана ошибкой в программном обеспечении, такие ошибки лучше предупреждать путём своевременного обновления системы до актуального состояния.

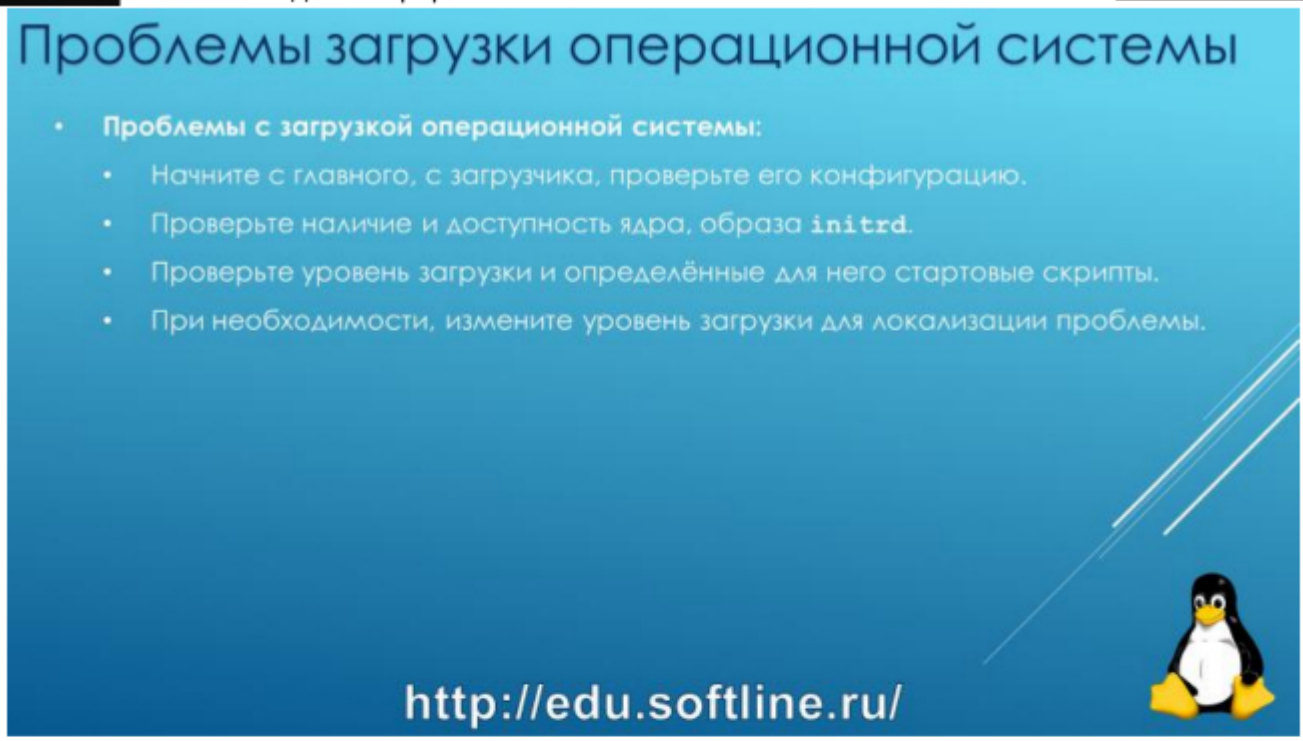
Никогда не забывайте о том, что все проблемы решаемы, но так утверждать может только администратор, у которого за спиной целый шкаф с резервными копиями всех важных данных.



Являетесь вы системным или сетевым администратором, вам всё равно придётся решать проблемы, связанные с функционированием сети. При поступлении заявки с текстом "не открывается сайт", сразу связывайтесь с автором и просите пояснить, что пишет web-браузер, в кодах ошибок очень многое говорится, ведь проблема может быть и в разрешении имён и в вашем маршрутизаторе, а может сервер, на котором находится сайт, и вовсе выключили.

Доступность удалённого узла вы всегда можете проверить программой ping, но всегда помните, как она работает. Правильно интерпретировать результаты работы программы вы сможете только тогда, когда поймёте, что делает эта программа. Если при отправке ICMP-пакета программой ping вы не получили ответного пакета, то это ни в коем случае не даёт 100% гаратнию, что узел не доступен. Это может говорить о том, что сетевой пакет отфильтровался на каком-то этапе, ведь сетевые брандмауэры могут фильтровать icmp-трафик, но пропускать весь остальной.

Случается, что проблемы не столько просты. В больших сетях иногда случаются "сетевые штормы", когда через сетевые устройства проходит такое количество трафика, что они становятся перегруженными, замедляется работа сети. В таком случае, команда ping поможет вам, разве что, убедиться в том, что в сети неполадки. Вам необходимо использовать анализаторы трафика и посмотреть, какие пакеты, откуда и куда они идут, чтобы локализовать проблему и перекрыть её источник.



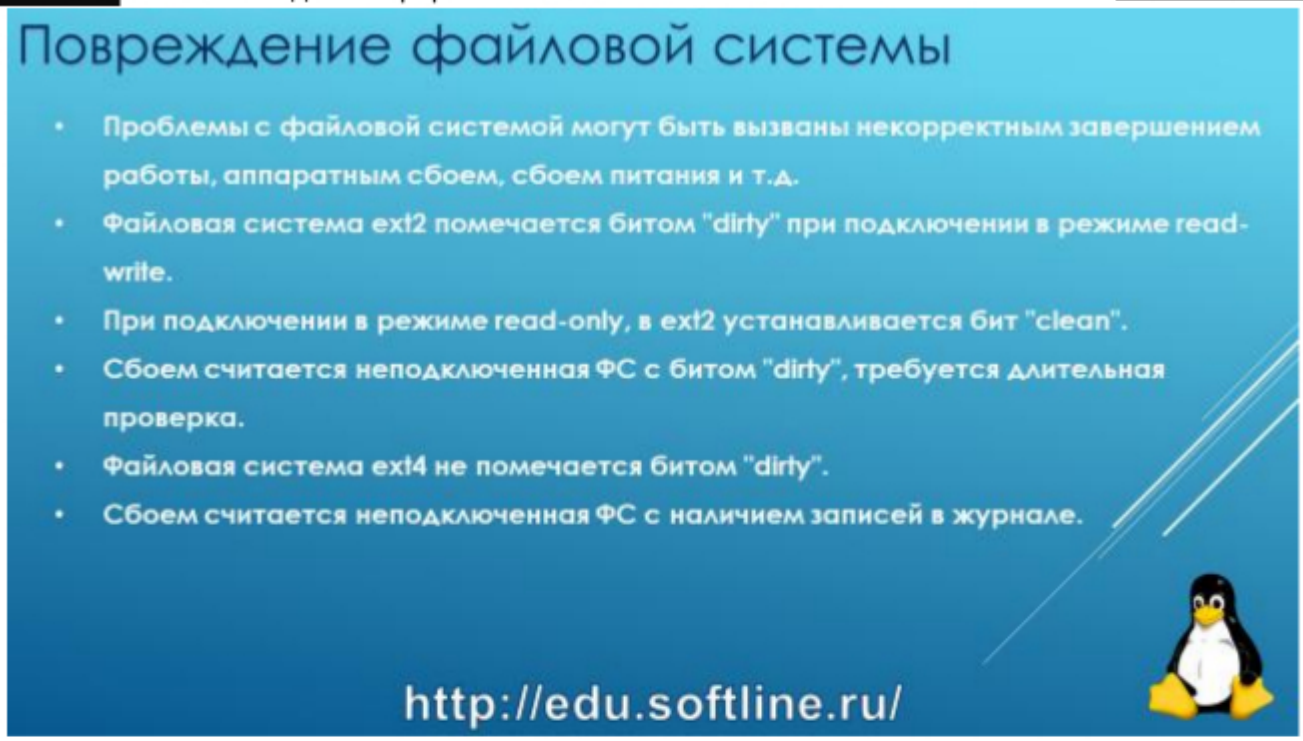
Проблемы, связанные с загрузкой системы, являются одними из самых сложных, так как сервер может не перезагружаться месяцами и действие, из-за которого появился сбой, могло быть выполнено в любой из дней работы сервера, а проблема проявится только при перезагрузке. Именно для этого необходимо документирование всех изменений, вносимых в систему.

В процессе решения проблем, связанных с загрузкой системы, очень важно понимать и помнить весь процесс запуска, чтобы быстро определить неисправный элемент.

Если загрузка оборвалась на этапе загрузки ядра, то проверьте сначала конфигурацию загрузчика GRUB, убедитесь, что в ней нет опечаток, что присутствуют обязательные параметры и они указывают на валидные объекты. Убедитесь в наличии и доступности ядра, а также образа **`initrd`**, отсутствие которого может привести к сбою загрузки.

Если загрузка была прервана после загрузки ядра, при вызове процесса **`init`**, то начинайте проверять его элементы, начните с файла **`/etc/inittab`**. Не исключено, что проблема была вызвана повреждённой файловой системой. Если вы загружаетесь в 5-м уровне запуска, и после загрузки всех элементов вы не можете попасть в систему, то обратите внимание на X-сервер, возможно проблема скрыта в нём.

Всегда обращайте пристальное внимание на сообщения, выдаваемые системой, в них может быть скрыта вся необходимая, для решения проблемы, информация.

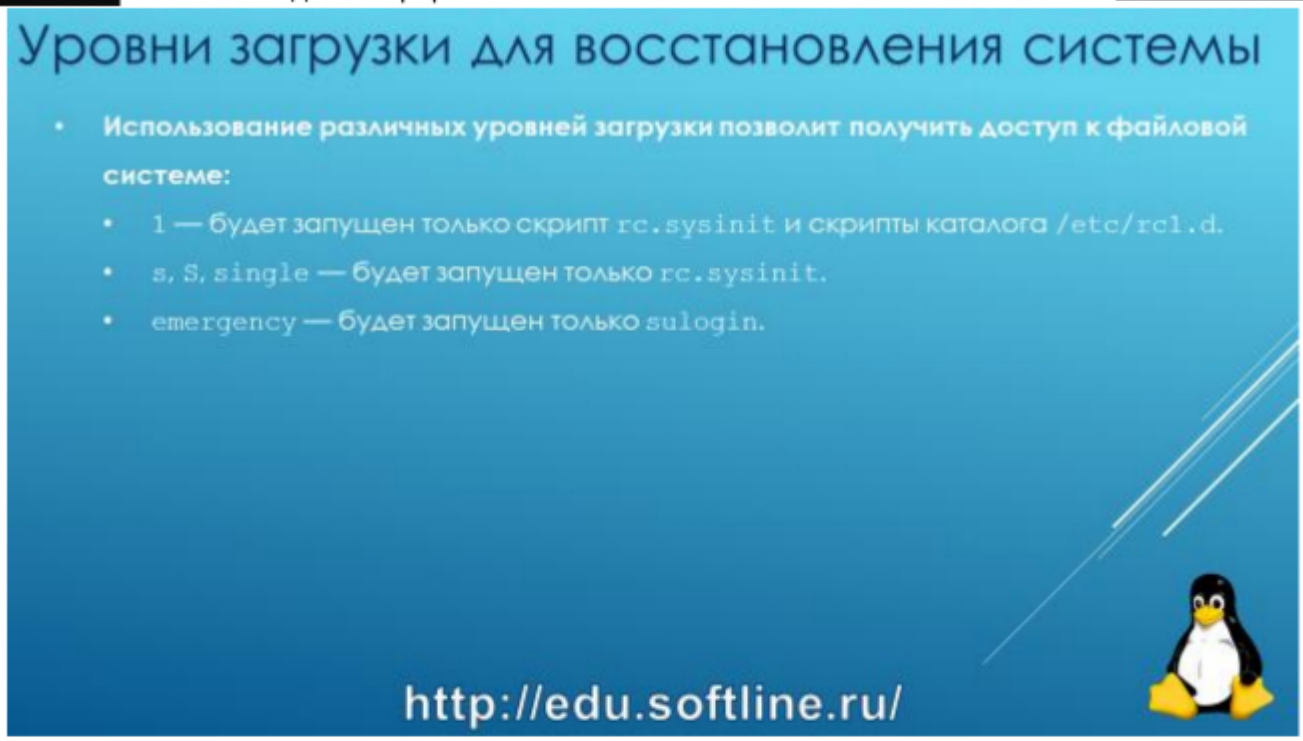


Повреждения файловой системы случаются очень часто, так как бывают вызваны крахом системы, либо сбоем питания, либо иными событиями, часто происходящими в процессе работы. Такого рода повреждения связаны с тем, что запись на диск производится не сразу, а постепенно. Сначала данные помещаются в буфер, после чего постепенно записываются на диск, естественно, что при исчезновении питания, содержимое этих буферов не сохраняется.

При подключении файловой системы ext2 она помечается битом "dirty", который будет снят при корректном отключении файловой системы, которое гарантирует синхронизацию данных между буфером и диском. Когда файловая система подключается и обнаруживается установленный бит "dirty", это означает, что файловая система была некорректно отключена и может иметь повреждённую информацию. В таком случае требуется длительная проверка данных.

Файловая система ext3 отличается от ext2 только наличием журнала, соответственно, при подключении такой файловой системы и обнаружении в журнале записей, делается заключение, что система повреждена. Но поскольку в журнале находится информация о заданиях, необходимых для выполнения, то процесс может быть продолжен.

В случае повреждения файловой системы, попробуйте восстановить её с помощью программы **`fsck`**.

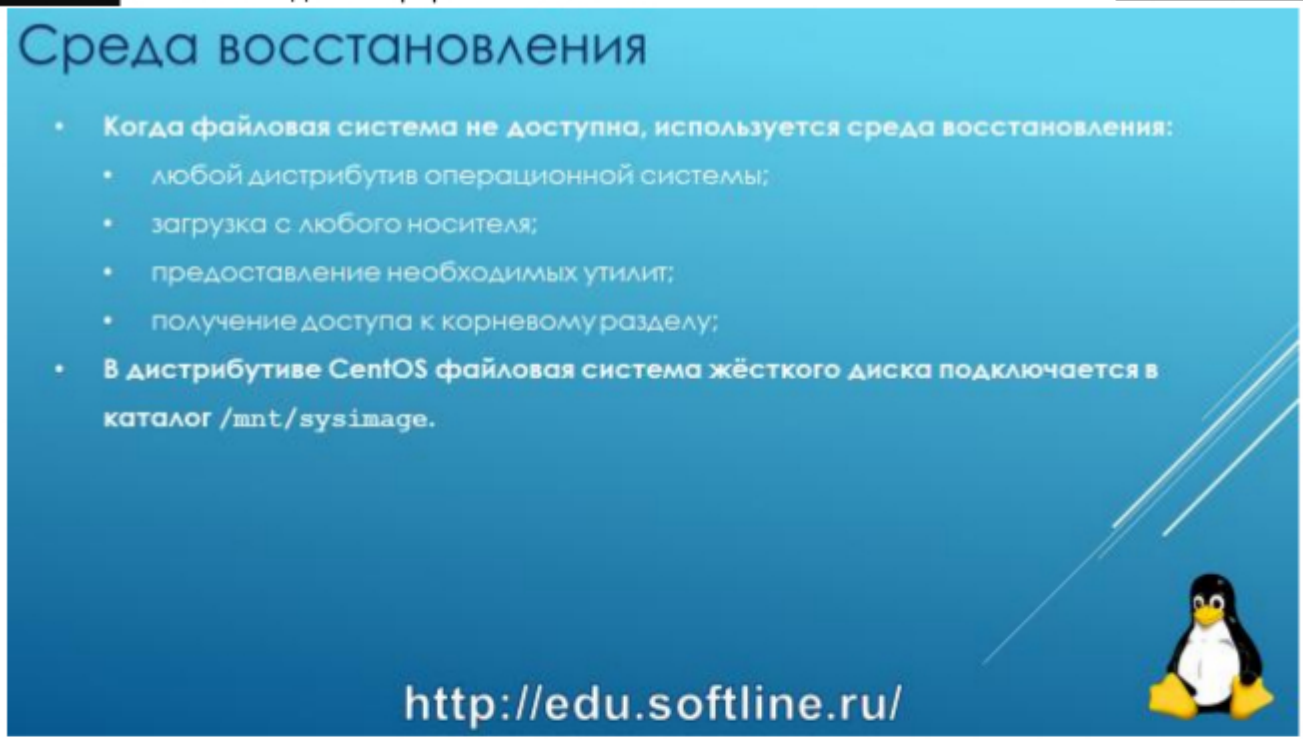


В случае, когда система не может загрузиться на любом этапе, следующем за загрузкой ядра, вы можете использовать различные уровни загрузки для того, чтобы получить контроль над системой до загрузки неисправного элемента.

Загрузка в 1-м уровне загрузки позволит загрузить операционную систему с минимальным набором необходимого ПО.

Загрузка в уровне **`single`** позволит получить контроль над системой без вызова программы **`init`**, на случай повреждений файла **`/etc/inittab`**, либо иного компонента программы **`init`**.

Загрузка в уровне **`emergency`** производит запуск программы sulogin для предоставления пользователю командной оболочки, ничего более запущено не будет.



Среда восстановления используется в случае, когда загрузка с жёсткого диска невозможна, по какой-либо причине. При загрузке среды восстановления, она пытается обнаружить установленные копии дистрибутива и предлагает администратору выбрать, с какой из копий необходимо работать.

Найденный корневой раздел установленной ОС будет подключен в каталог **/mnt/sysimage**. Естетственно, если файловая система повреждена, то среда восстановления может оказаться бессильной, при переключении на вторую виртуальную консоль вы получите доступ к программе **fdisk**.

Чтобы загрузиться в режим восстановления, вставьте установочный компакт-диск и при загрузке, в качестве параметра, укажите rescue:

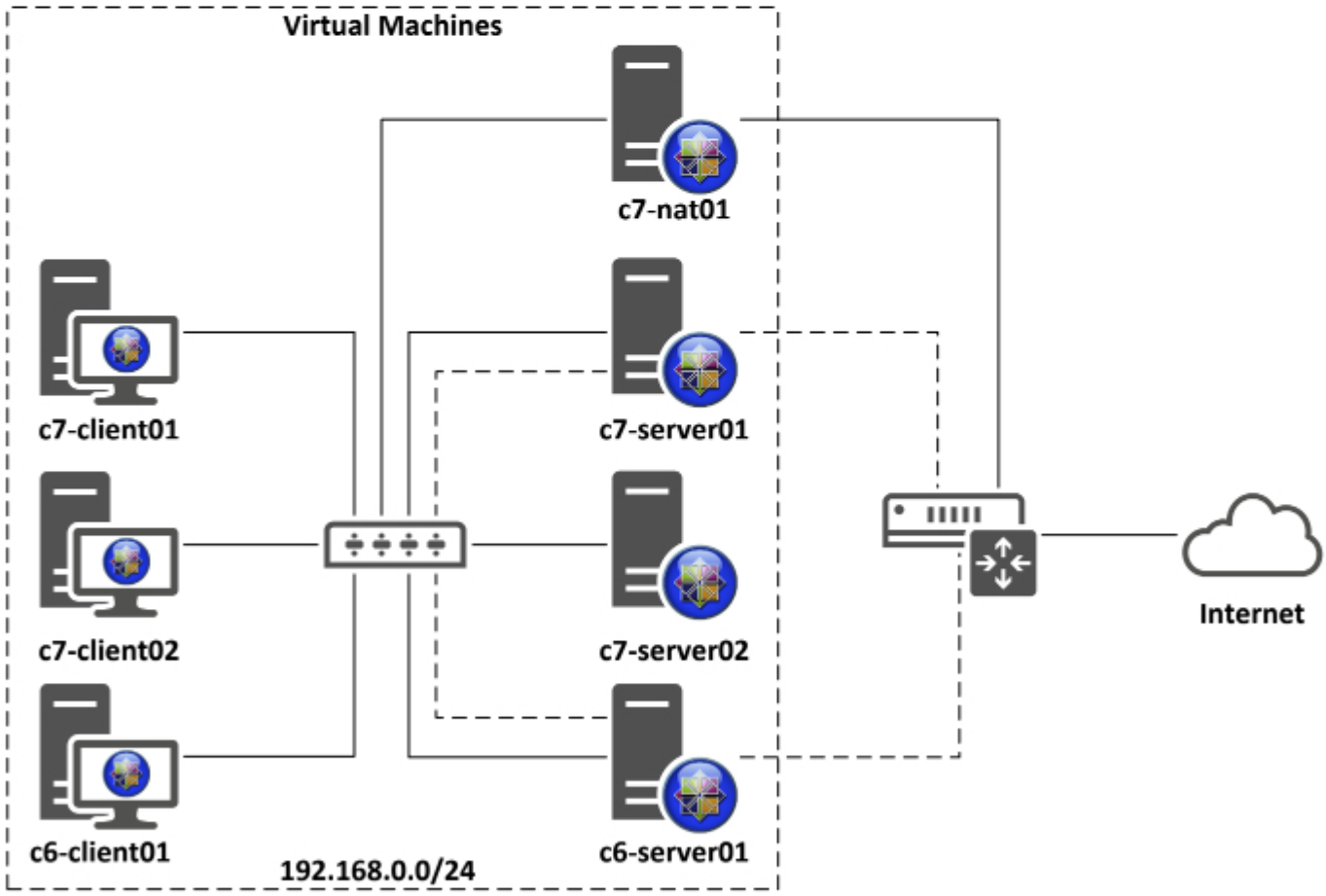
```
linux rescue
```

Лабораторная работа

Лабораторная работа 11

- **Упражнения:**
  - Восстановления доступа к системе.
  - Восстановления работы системы.
- **Необходимые машины:**
  - `c7-nat01`.
- **Учетные данные:**
  - Пользователь: `root`.
  - Пароль: **`Pa$$w0rd`**.

http://edu.softline.ru/





Упражнение 1: Восстановление пароля root в CentOS 7.

ПРИМЕЧАНИЕ:
Восстановление пароля пользователя root, это очень важная задача, в случае если вы его потеряли или забыли. В CentOS 7 (RHEL 7) вы можете сбросить его.
*Обратите внимание, что в GRUB2 сброс пароля в однопользовательском (**single-user**) режиме не поддерживается, как это было в CentOS 6 (RHEL 6) и предыдущих. Пароль пользователя root теперь требуется для работы в спасательном режиме (Emergency Mode), который пришел на смену однопользовательскому (**single-user**).*

1. Запустите виртуальную машину **c7-server01**.
2. Загрузите в домашнюю папку пользователя root скрипт (http://*URL*/files/lab11_ex01.sh).

```
# wget http://URL/files/lab11_ex01.sh
```

ПРИМЕЧАНИЕ:
Уточните ссылку у инструктора.

3. Добавьте право на выполнение загруженного файла.

```
# chmod +x lab11_ex01.sh
```

4. Выполните скрипт.

```
# lab11_ex01.sh
```

5. После перезагрузки системы убедитесь, что не можете войти под пользователем **root** с использованием пароля **Pa$$w0rd**.
6. Решите возникшую проблему.



Ответы к упражнению 1: Восстановление пароля root в CentOS 7.

Задача 1: Подготовка к лабораторной работе.

1. Запустите виртуальную машину **c7-server01**.
2. Загрузите в домашнюю папку пользователя root скрипт (http://*URL*/files/lab11_ex01.sh).

```
# wget http://URL/files/lab11_ex01.sh
```

ПРИМЕЧАНИЕ:
Уточните ссылку у инструктора.

3. Добавьте право на выполнение загруженного файла.

```
# chmod +x lab11_ex01.sh
```

4. Выполните скрипт.

```
# lab11_ex01.sh
```

5. После перезагрузки системы убедитесь, что не можете войти под пользователем root.

Задача 2: Восстановление пароля root в CentOS 7.

1. Перезагрузите систему **c7-server01**.

<**Ctrl**>+<**Alt**>+<**Del**>

2. В загрузочном меню GRUB 2 нажмите клавишу <**e**>.
3. Добавьте следующий параметр в конец строки **`linux`** или **`linuxefi`** на системах с UEFI:

```
init=/bin/sh
```

ПРИМЕЧАНИЕ:
*Ядро Linux запустит оболочку **`/bin/sh`** прежде чем демон **`systemd`**, при этом часть функционала будет ограничена, а часть недоступна.*
*Параметры **`rhgb`** и **`quiet`** должны быть отключены, чтобы включить системные сообщения.*

4. Нажмите комбинацию клавиш <**Ctrl**>+<**x**> для загрузки системы с параметрами.
5. Файловая система смонтируется в режиме только для чтения (Read-Only), необходимо ее перемонтировать для записи измененного пароля:

```
# mount -o remount,rw /
```

6. Воспользуйтесь командой **passwd** для смены пароля пользователя root.



ПРИМЕЧАНИЕ:

В случае если файловая система не смонтирована в режиме для записи вы получите ошибку:

```
passwd: Authentication token manipulation error
```

7. Если вы используете SELinux, для корректного запуска контекста измененных файлов воспользуйтесь командой:

```
# touch /.autorelabel
```

8. Для продолжения загрузки воспользуйтесь командой:

```
# exec /sbin/init
```



Упражнение 2: Восстановление загрузки CentOS 7.

8. Запустите виртуальную машину **c7-server02**.
9. Загрузите в домашнюю папку пользователя root скрипт (http://*URL*/files/lab11_ex02.sh).

```
# wget http://URL/files/lab11_ex02.sh
```

ПРИМЕЧАНИЕ:
Уточните ссылку у инструктора.

10. Добавьте право на выполнение загруженного файла.

```
# chmod +x lab11_ex02.sh
```

11. Выполните скрипт.

```
# lab11_ex02.sh
```

12. После перезагрузки системы убедитесь, что система не загружается.
13. Решите возникшую проблему.



Упражнение 2: Восстановление загрузки CentOS 7.

Задача 1: Подготовка к лабораторной работе.

1. Запустите виртуальную машину **c7-server02**.
2. Загрузите в домашнюю папку пользователя root скрипт (http://*URL*/files/lab11_ex02.sh).

```
# wget http://URL/files/lab11_ex02.sh
```

ПРИМЕЧАНИЕ:
Уточните ссылку у инструктора.

3. Добавьте право на выполнение загруженного файла.

```
# chmod +x lab11_ex02.sh
```

4. Выполните скрипт.

```
# lab11_ex02.sh
```

5. После перезагрузки системы убедитесь, что система не загружается.
6. Решите возникшую проблему.

Задача 2: Восстановление загрузки CentOS 7.

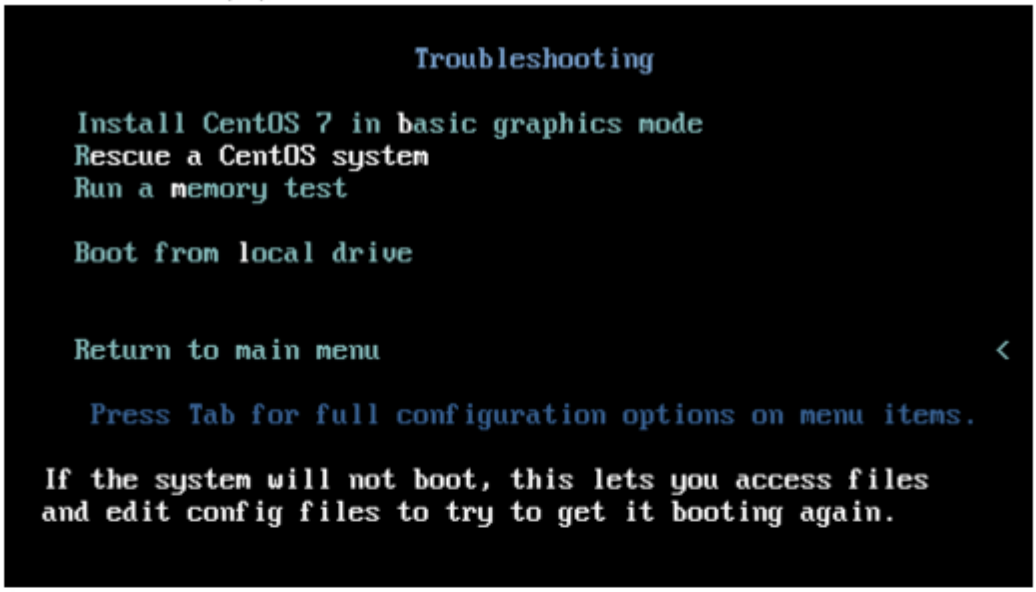
9. Подключите к виртуальной машине **c7-server02** установочный накопитель CentOS 7.

ПРИМЕЧАНИЕ:
Уточните расположение установочного накопителя CentOS 7.

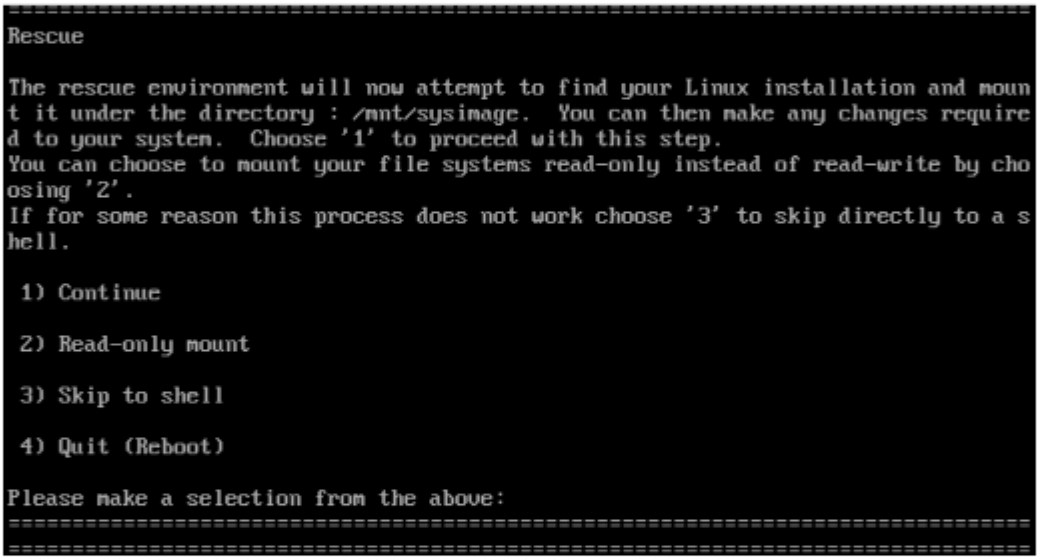
10. Перезагрузите систему **c7-server01**.

<**Ctrl**>+<**Alt**>+<**Del**>

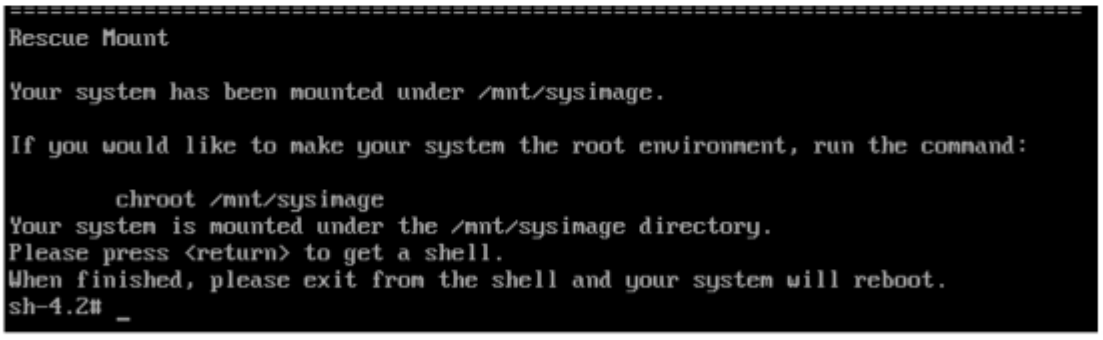
11. Запустите систему при помощи установочного диска в аварийном режиме (**Troubleshooting** => **Rescue a CentOS system**).



12. В аварийном окружении выберите продолжить (**Continue**) [1].



13. Перейдите к командной оболочке (**sh**).



14. Скопируйте пакет **util-linux-2.23.2-52.el7.x86_64.rpm** в домашнюю папку пользователя кщще.

```
# cp /run/install/repo/Packages/util-linux-2.23.2-
52.el7.x86_64.rpm /mnt/sysimage/root
```

15. Смените корневой каталог на **/mnt/sysimage**.

```
# chroot /mnt/sysimage
```

16. Выполните команду **mount** и проанализируйте вывод.

```
# mount
```

17. Проверьте установку пакета **util-linux**.

```
# rpm -V util-linux
```

18. Перейдите в домашний каталог пользователя root

```
# cd /root/
```

19. Переустановите пакет **util-linux-2.23.2-52.el7.x86_64.rpm**.

```
# rpm -ihv --replacepkgs util-linux-2.23.2-52.el7.x86_64.rpm
```

20. Выйдите из смены корневого каталога:

```
# exit
```

21. Перезагрузите систему и отключите установочный накопитель CentOS 7 от виртуальной машины.

```
# reboot
```

22. Убедитесь, что система запускается корректно.