Учебный курс

Безопасность сетей на базе Linux (Unix)

(**SecurL**)

Автор: Лесковец В.В.

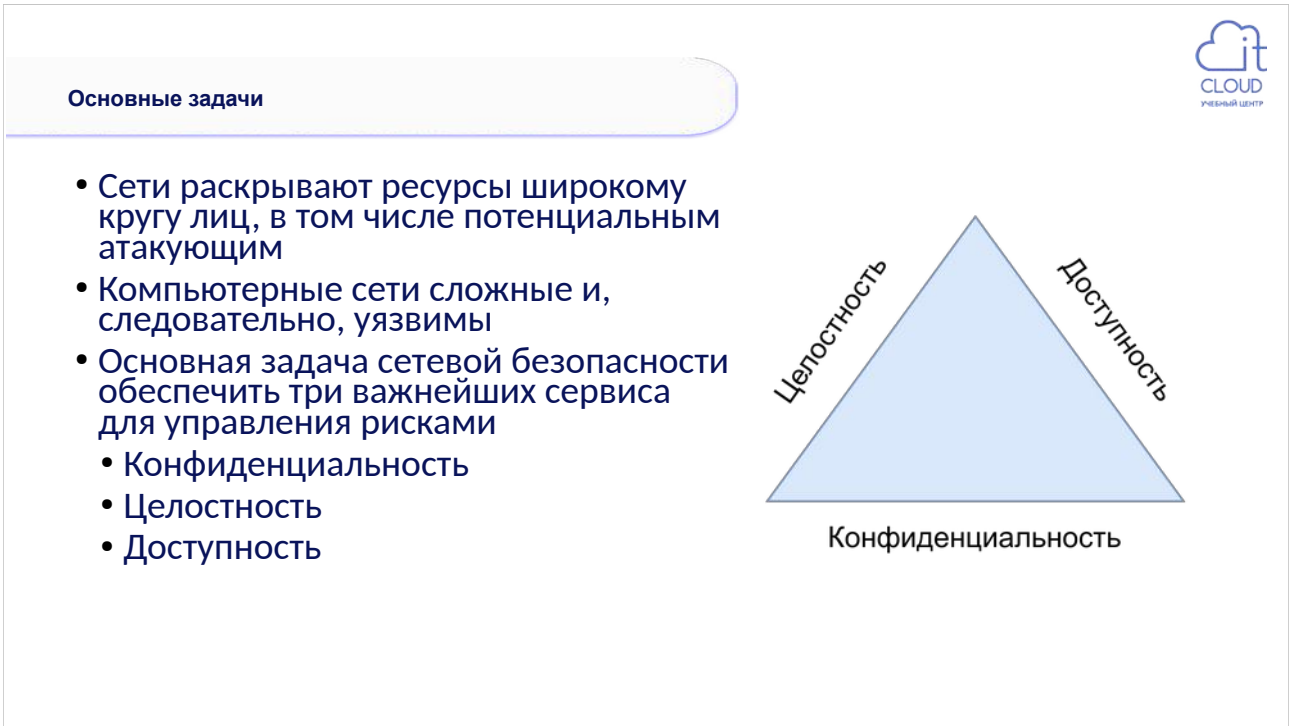
Версия: 2.0

г. Екатеринбург
2025

Оглавление

Глава 1. Введение.

1.1 Задачи, решаемые в ходе защиты компьютерных систем.

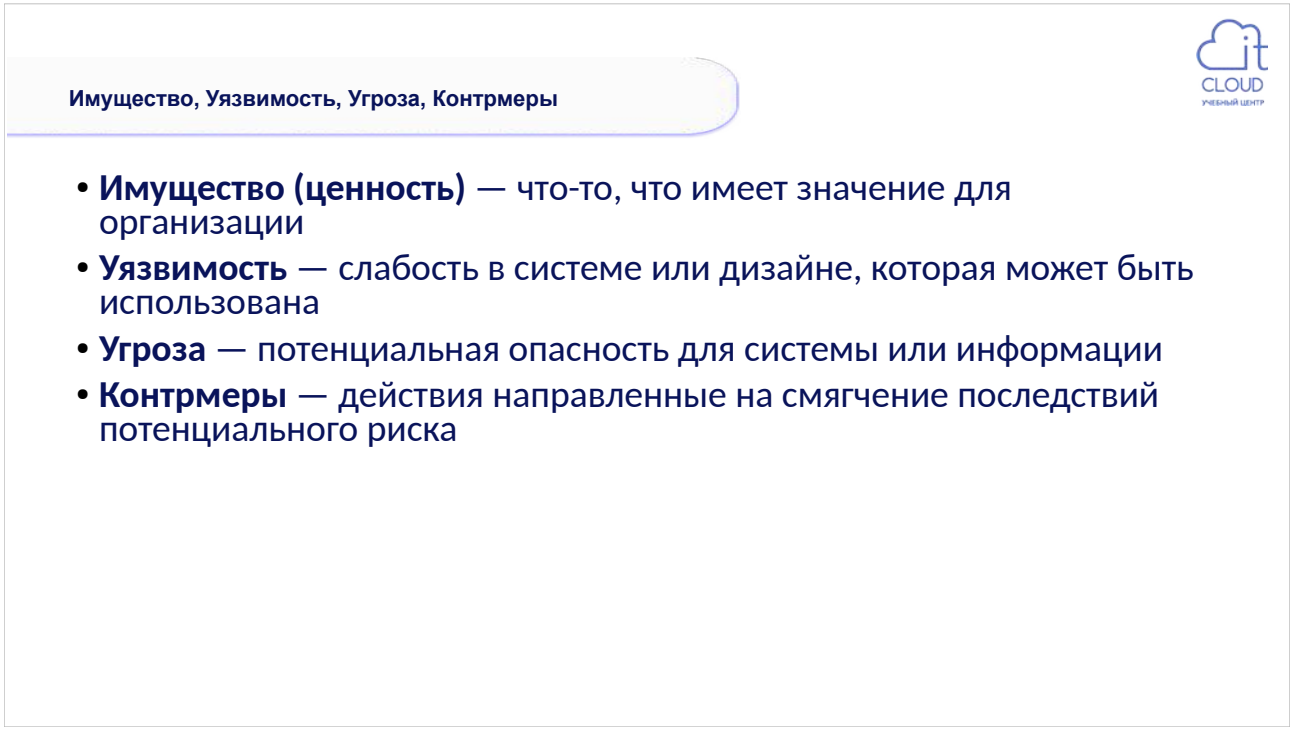


С одной стороны, современные условия предполагают, что сеть является неотъемлемой частью бизнеса. С другой стороны предполагается, что подключение к сети должно быть простым и постоянным. Поэтому обеспечение безопасности как самой сети, так и всех систем, подключенных к ней является жизненно важным. При этом необходимо исходить из следующих предположений:

- Сети являются сложными и тесно связанными, что дает возможность злоумышленнику легко подключиться и непосредственно в вашу сеть, и удалено
- Современные компьютерные системы и приложения становятся все сложнее, и, значит, их анализ, обеспечение безопасности и тестирование становится все сложнее и сложнее

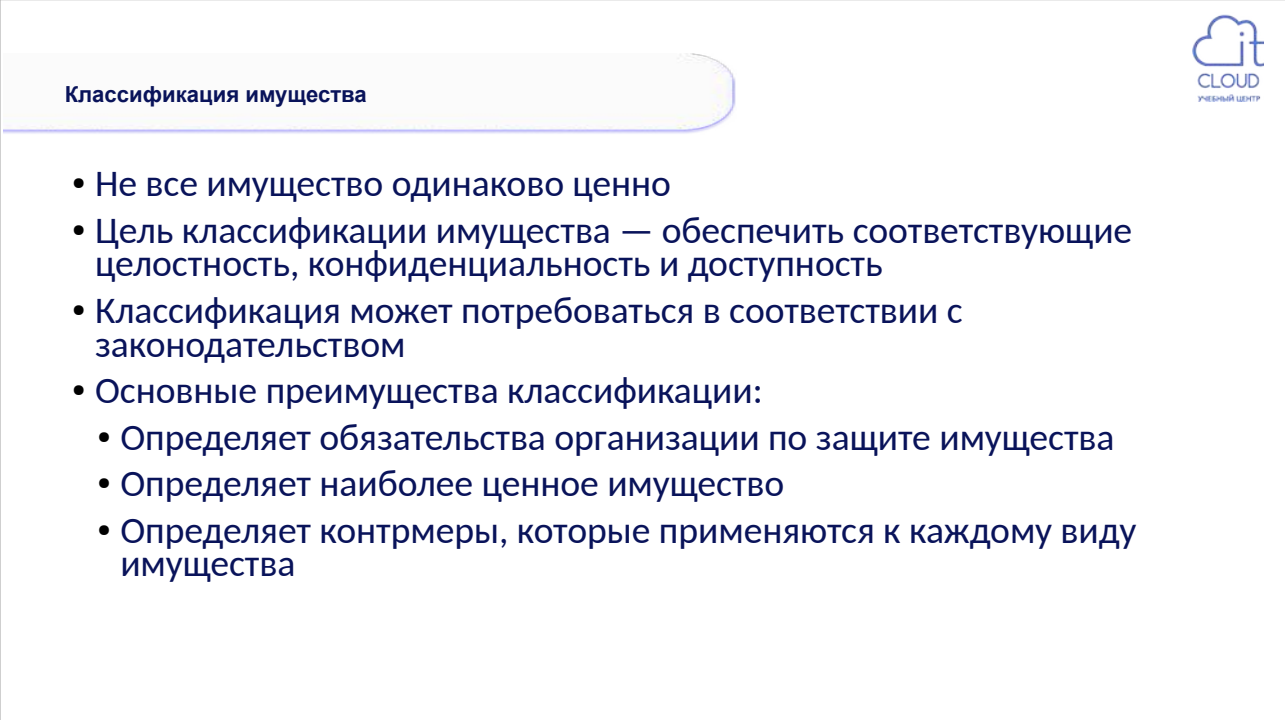
Основными элементами обеспечения безопасности являются:

- **Конфиденциальность** — предоставление информации только авторизованным лицам.
- **Целостность** — только авторизованные лица могут изменять данные и аутентичность данных.
- **Доступность** — гарантирует непрерывный доступ авторизованным лицам к системам и данным в них.



Хотя вирусы, черви и хакеры являются заметной и часто обсуждаемой частью информационной безопасности, наиболее важной частью является управление рисками. Управление рисками основано на определенных принципах защиты имущества и управлении безопасностью.

- **Имущество (ценность, asset)** — то, что имеет значение для организации. К этой категории относятся: материальные ресурсы, информация, репутация.
- **Уязвимость (vulnerability)** — слабость в системе или дизайне, которая может быть использована угрозой. Уязвимости могут быть на любом уровне: протокол, операционная система, приложение, настройка, порядок использования …
- **Угроза (threat)** — потенциальная опасность для системы или информации. Угроза реализуется когда кто-то или что-то идентифицирует уязвимость и использует ее.
- **Контрмеры (countermeasure)** — действия направленные на смягчение последствий потенциального риска. Контрмеры минимизируют или устраняют уязвимость, или минимизируют вероятность использования уязвимости.



Для обеспечения конфиденциальности, целостности и доступности важно правильно классифицировать имущество, что даст возможность правильно распределить ресурсы направленные на защиту данных. Иногда данные должны быть классифицированы в соответствии с законодательством.

Существуют разные подходы к классификации, например в Российской Федерации используется следующий :

- **Не секретные**: информация не требует или почти не требует соблюдения конфиденциальности, целостности и доступности.
- **Секретные**: к секретным сведениям следует относить все иные сведения из числа сведений, составляющих государственную тайну. Ущербом безопасности Российской Федерации в этом случае считается ущерб, нанесённый интересам предприятия, учреждения или организации в военной, внешнеполитической, экономической, научно-технической, разведывательной, контрразведывательной или оперативно-розыскной области деятельности.
- **Совершенно секретные**: к совершенно секретным сведениям следует относить сведения в области военной, внешнеполитической, экономической, научно-технической, разведывательной, контрразведывательной и оперативно-розыскной деятельности, распространение которых может нанести ущерб интересам министерства (ведомства) или отрасли экономики Российской Федерации в одной или нескольких из перечисленных областей.
- **Особой важности**: к сведениям особой важности следует относить сведения в области военной, внешнеполитической, экономической, научно-технической, разведывательной, контрразведывательной и оперативно-розыскной деятельности,

распространение которых может нанести ущерб интересам Российской Федерации в одной или нескольких из перечисленных областей.

Другой подход к классификации (может быть использован коммерческой организацией):

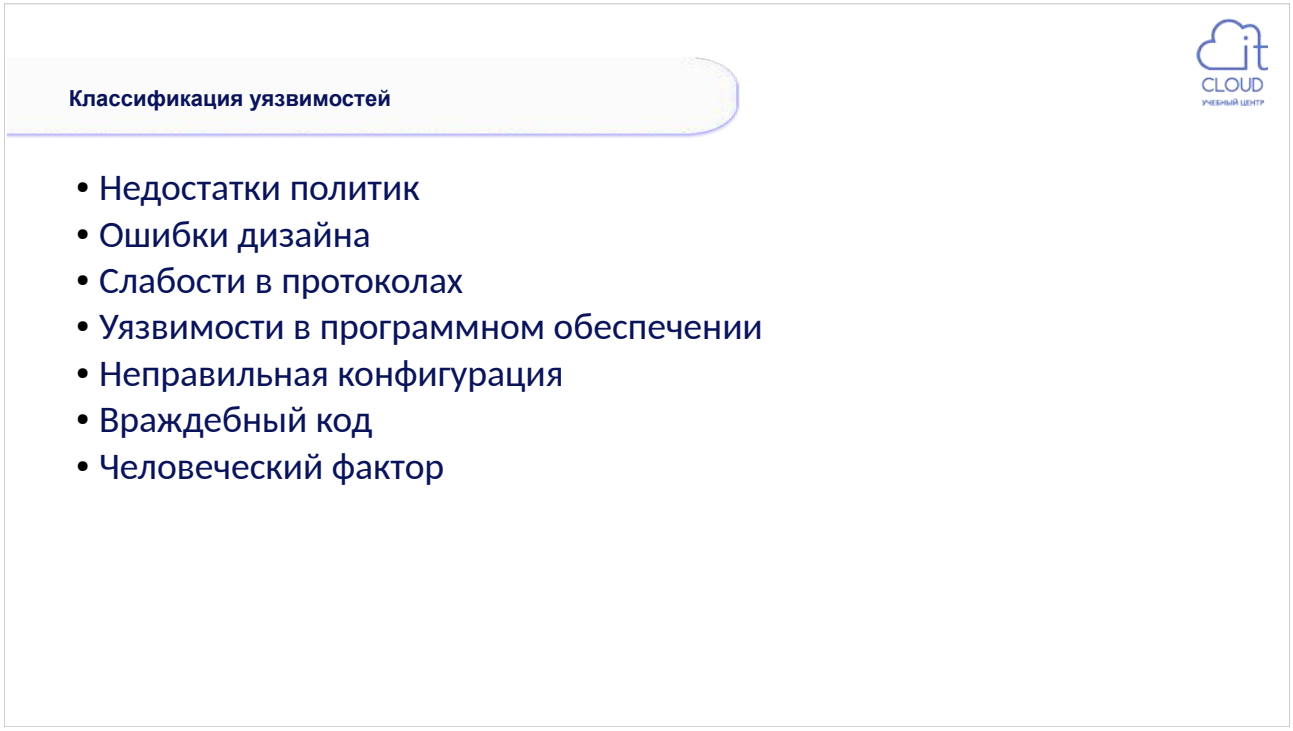
- **Публичная** — рекламные буклеты, веб-сайт и т. п. Требует соблюдения целостности и доступности.
- **Важная** — утечка информации не может нанести серьезного ущерба организации. Помимо целостности и доступности требует определенных усилий по ограничению распространения. Например: внутренние объявления.
- **Частная** — данные важны для правильного функционирования организации. Основные усилия — целостность и доступность для авторизованных лиц. Пример: складская база данных
- **Конфиденциальная** — пример: торговые секреты или данные на сотрудников. Требует наибольших усилий по защите.

Основные факторы влияющие на классификацию:

- **Значимость** — выраженная, к примеру, в деньгах. Значимость можно оценивать на только стоимостью.
- **Время жизни** — как долго информация должна защищаться.
- **Период использования** — период времени, во время которого, данные нужны для работы.
- **Персональная привязка** — лицо, с которым, ассоциированы данные.

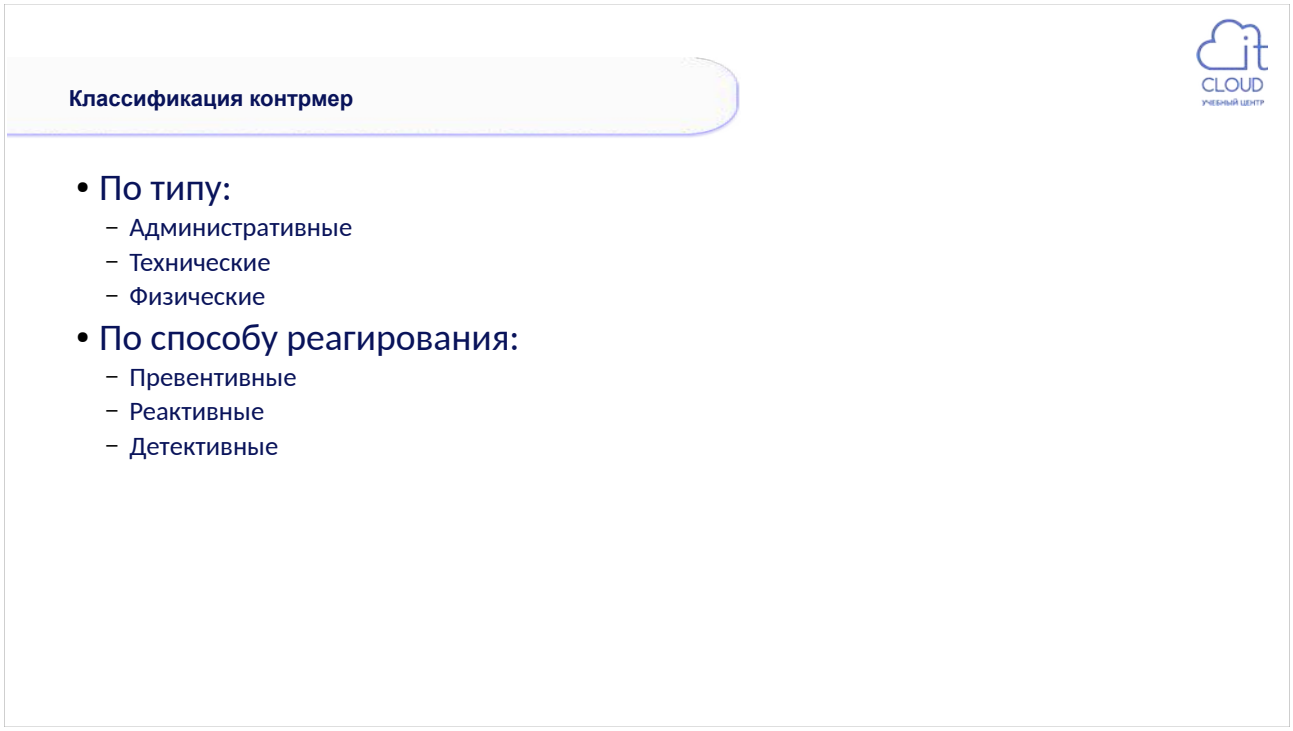
Еще один аспект связанный с классификацией роли:

- **Владелец** — лицо ответственное за имущество (данные).
- **Хранитель** — лицо обеспечивающее защиту.
- **Пользователь** — лицо использующие данные.



Оценка уязвимости помогает применять правильные методы защиты. Здесь приведен не полный список возможных уязвимостей. Важно внимательно и аккуратно проанализировать список возможных уязвимостей для именно вашей системы или сети.

1.2 Обзор механизмов и принципов защиты.

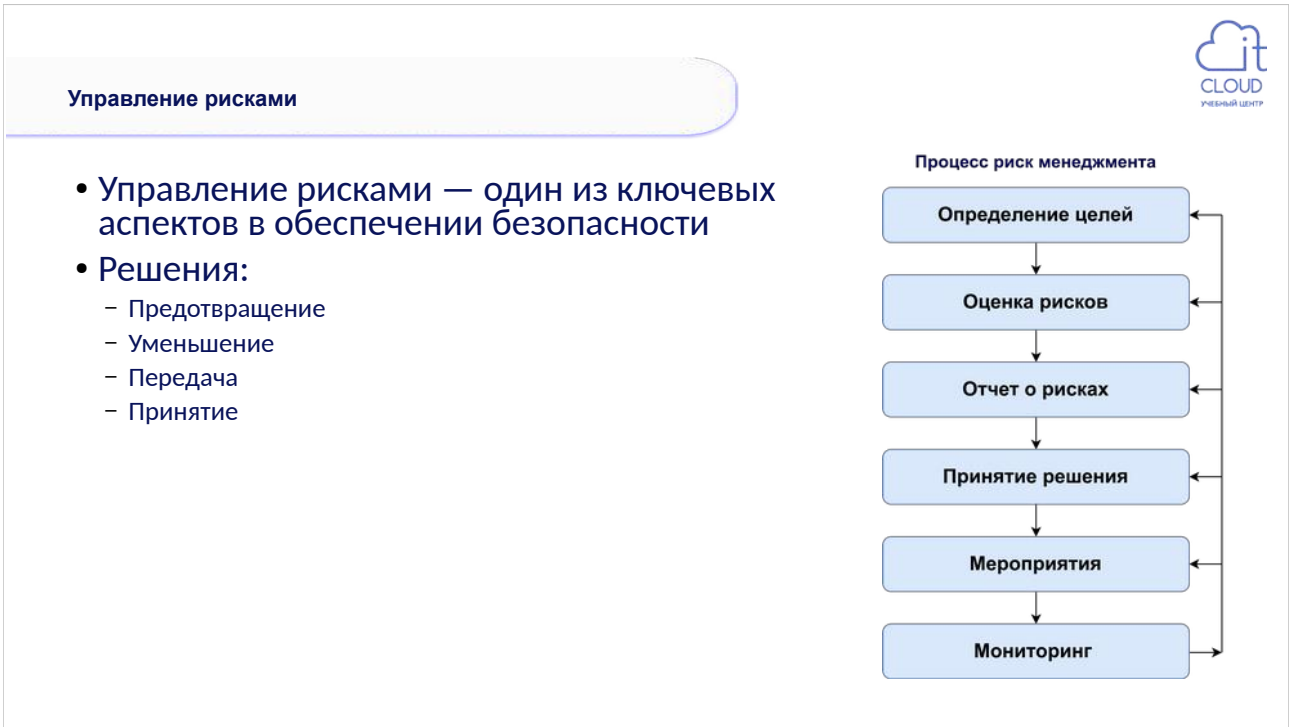


Административный, технический и физический контроль

- **Административный**: в основном состоит из политик и процедур. Политики безопасности, тренировки персонала, контроль изменений, аудит…
- **Технический**: электронный, аппаратный, программный и т. п. Брандмауэры, ВПН, биометрия…
- **Физический**: защита физической инфраструктуры. Охрана, замки, ИБП…

Превентивные, реактивные, детективные меры

- **Превентивные**: меры применяются заранее до нарушения безопасности. *Профилактика*.
- **Реактивные**: меры призванные уменьшить ущерб, вызванный нарушением безопасности. *Лечение*.
- **Детективные**: меры по поиску причин нарушения и методов их нарушения. *Поиск вакцины*.

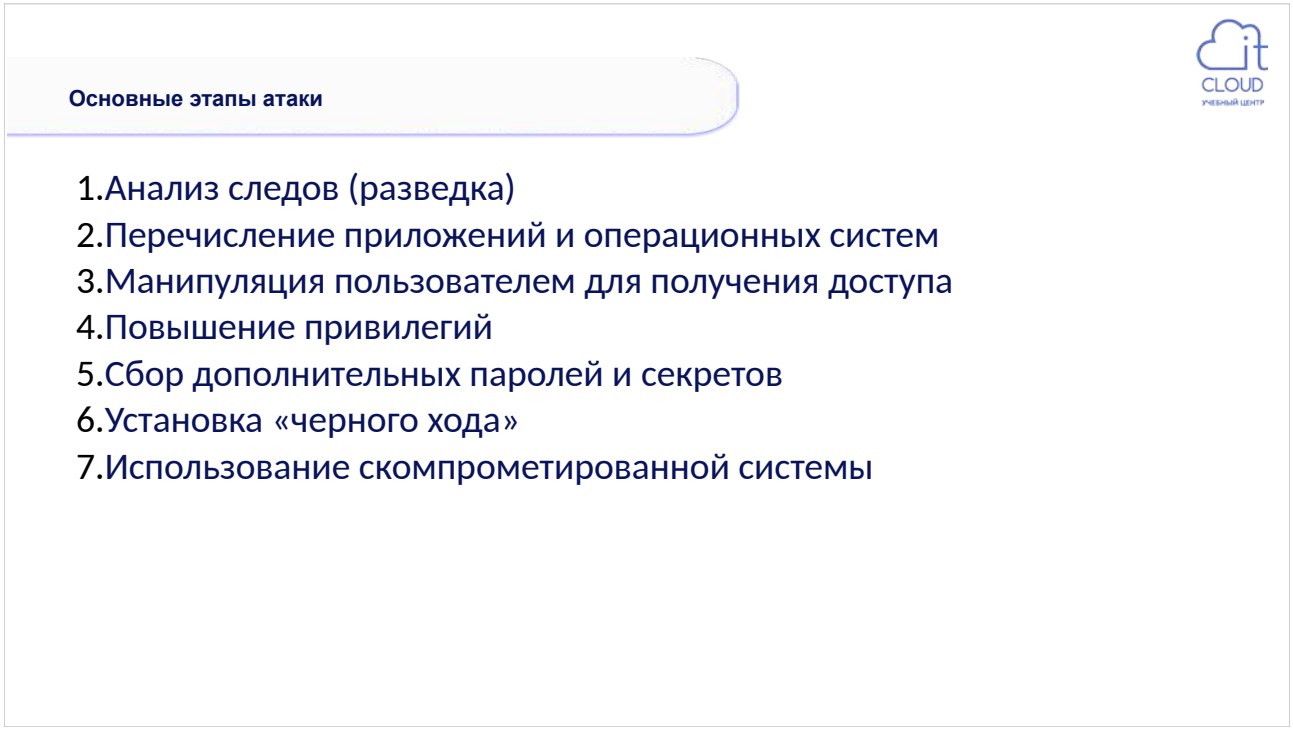


Риск — потенциальная проблема. Риски можно оценить по вероятности наступления негативных событий, возможному ущербу, потенциальному времени и многому другому.

Управление рисками — процесс в котором ищут, оценивают и, по возможности, предотвращают потенциальные проблемы. После применения мероприятий по уменьшению рисков важно наблюдать за результатов и применять корректировки на периодической основе.

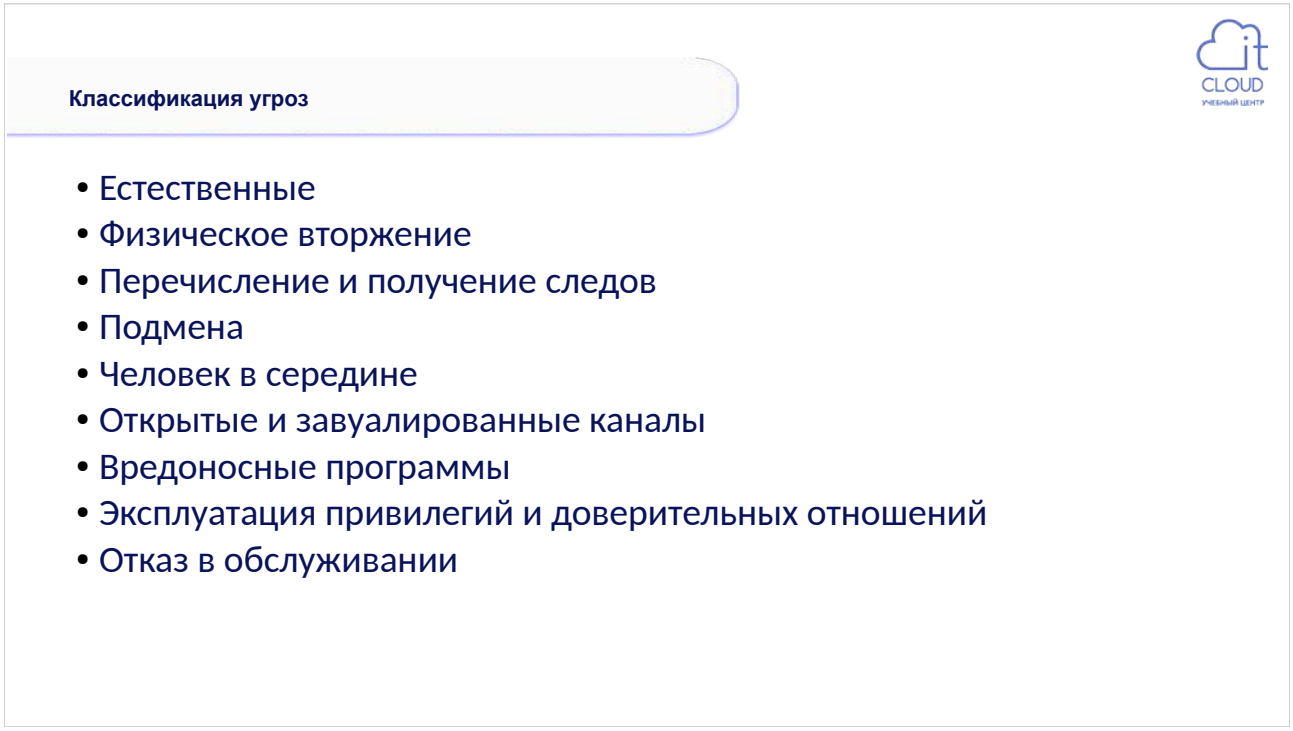
Список возможных решений классифицируется следующим образом:

- **Предотвращение**: меры не позволяющие использовать угрозу, связанную с риском. Блокирование писем с приложением.
- **Уменьшение**: уменьшение возможных последствий. Установка антивируса на почтовый шлюз.
- **Передача**: передача рисков третьем лицам. Страхование на случай заражения вирусом, переданным по почте.
- **Принятие**: принятие риска вследствие невозможности или нецелесообразности его предотвращения, уменьшения или передачи. Ничего не делаем.



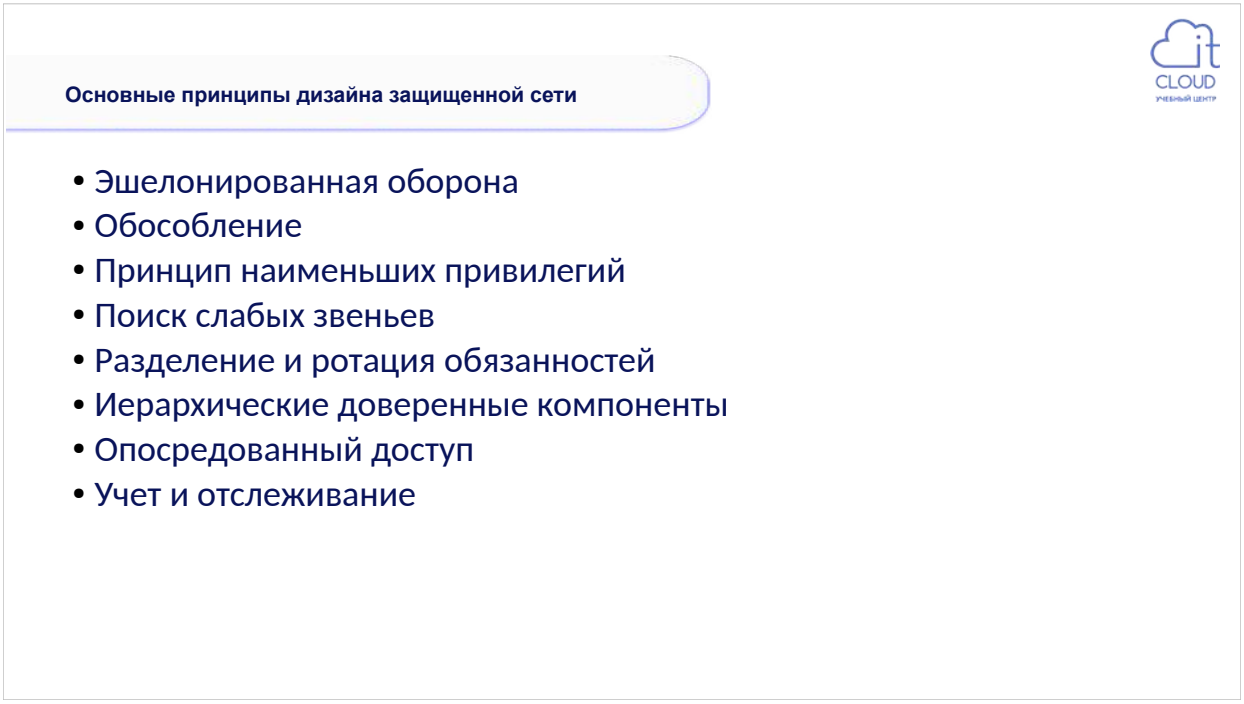
Атакующий, как правило, начинает атаку с минимальной или нулевой информацией о цели атаки. Постепенно он получает больше сведений о цели и использует либо известные уязвимости или пользователей для вторжения в систему.

Здесь описана примерная схема атаки. Общая ситуация такова, что атакующему надо выявить хотя бы *одну уязвимость*, а защищающемуся надо закрыть *все уязвимости*.



Одна из задач в обеспечении безопасности выявить все возможные угрозы системе и минимизировать их возможное использование. Что требует тщательного и обстоятельного анализа. Некоторые из угроз:

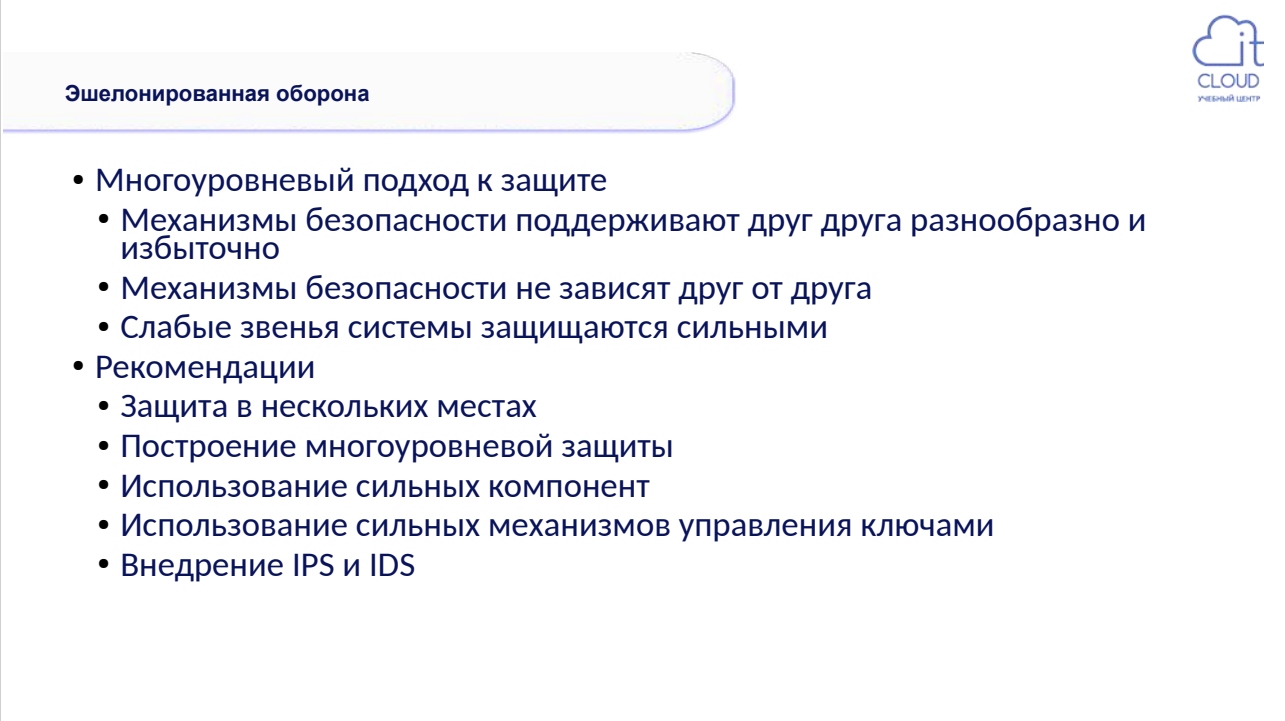
- Естественные
- Физическое вторжение
- Перечисление и получение следов
- Подмена
- Человек в середине
- Открытые и завуалированные каналы
- Вредоносные программы
- Эксплуатация привилегий и доверительных отношений
- Отказ в обслуживании



При планировании стратегии защиты следует придерживаться нескольких основополагающих принципов:

- **Эшелонированная оборона** — каждый элемент должно защищать несколько уровней защитных мер. Система настолько защищена, насколько защищено его самое слабое звено.
- **Обособление** — различное имущество с разной значимостью должно находится в разных доменах безопасности.
- **Принцип наименьших привилегий** — чем меньше привилегий выдается, тем меньше риск нарушения безопасности. Но привилегии должны быть выданы в должном объеме.
- **Поиск слабых звеньев** — система настолько слаба, насколько слабо его самое слабое звено. Поэтому слабые звенья следует защищать сильными.
- **Разделение и ротация обязанностей** — не должно быть одного человека, который выполняет какую-то работу.
- **Иерархические доверенные компоненты** — иерархический подход к обособлению и принципу наименьших привелегий.
- **Опосредованный доступ** — доступ к разным доменам безопасности осуществляется через централизованный компонент.
- **Учет и отслеживание** — позволит оценить принятые контрмеры и попытки их нарушения.

1.3 Концепция глубоко эшелонированной (многоуровневой) защиты.



Эшелонированная оборона универсальный принцип построения безопасности в любой сфере человеческой деятельности. Основная идея — многоуровневый подход к защите, главная цель которой не остановка наступления, но его замедление и распыления сил. Главные компоненты эшелонированной обороны:

- Механизмы безопасности поддерживают друг друга разнообразно и избыточно
- Механизмы безопасности не зависят друг от друга
- Слабые звенья системы защищаются сильными

Рекомендации для построения эшелонированной обороны:

- **Защита в нескольких местах:** необходимо обеспечивать защиту на всех возможных направлениях (и невозможных тоже). В том числе:
  - Защита сети и инфраструктуры. Защита от атак возможности сетевой коммуникации и обеспечение ее целостности и конфиденциальности.
  - Обеспечение защиты границ (внутренних и внешних) сети.
  - Защита вычислительных устроцств
- **Построение многоуровневой защиты:** использование нескольких механизмов защиты каждого компонента. Более того каждый механизм должен не только защищать, но и определять нарушение безопасности.
- **Использование сильных компонент:** защитные механизмы должны быть максимально стойкими.

- **Использование сильных механизмов управления ключами:** используйте такие ключи и алгоритмы управления ключами, которые обеспечивают максимальную критостойкость.
- **Внедрение IPS и IDS:** Системы предотвращения вторжения (*Intusion Prevention System, IPS, активная*) и системы обнаружения вторжения (*Intusion Detection System, IDS, пассивная*) позволяют получить ответы на ряд важнейших вопросов:
  - Меня атакуют?
  - Кто атакует?
  - Что является целью?
  - Кто-нибудь еще атакован?
  - Какие у меня варианты действий?

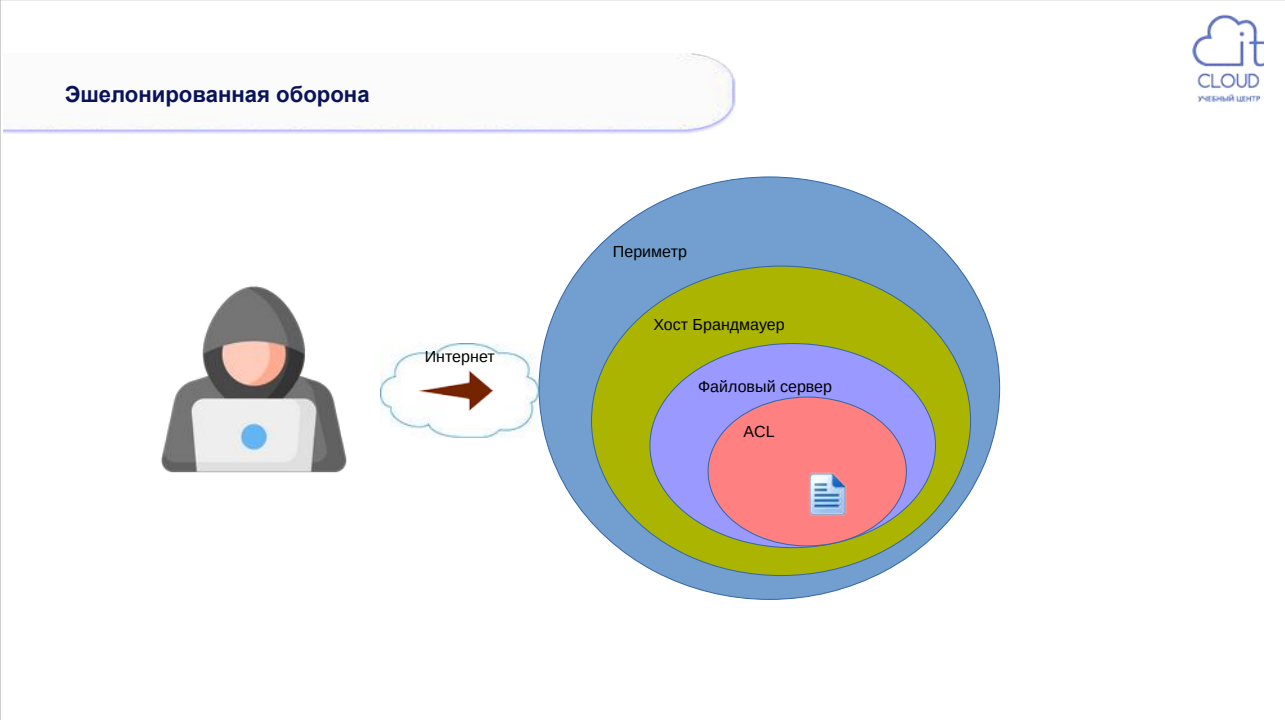
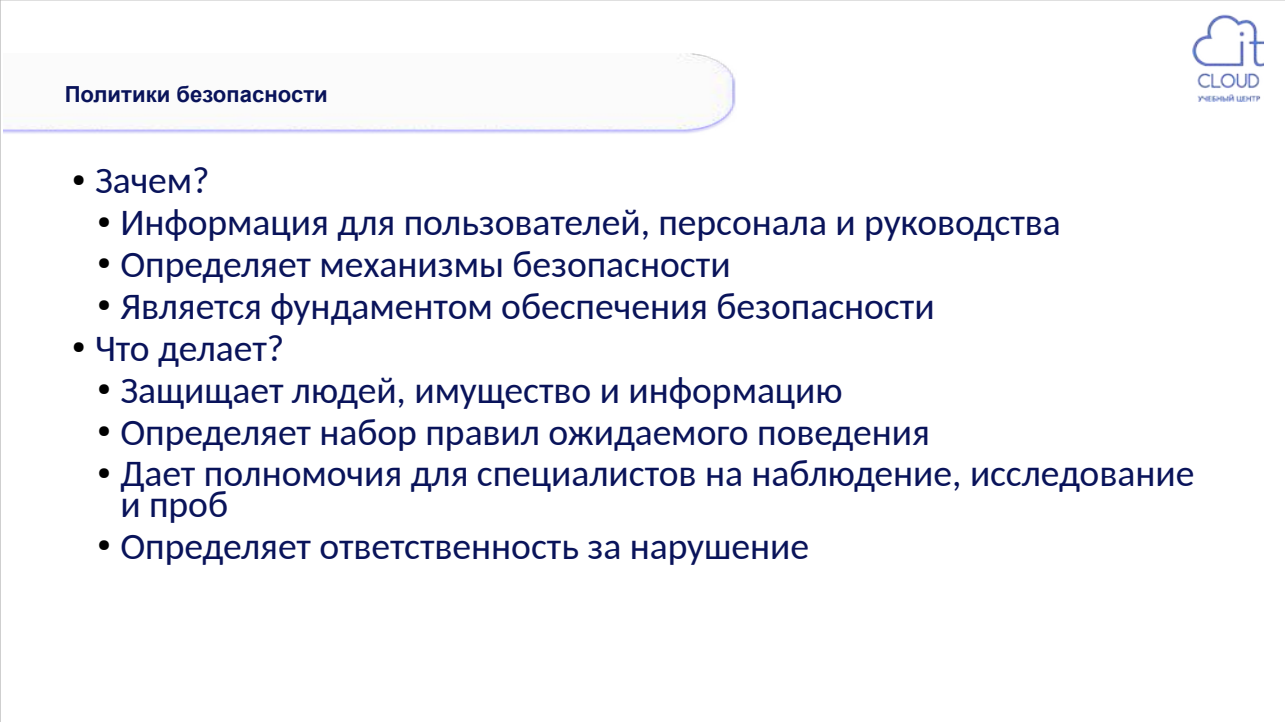


Иллюстрация принципа эшелонированной обороны:

Атакующему для получения важных данных на файловом сервере необходимо преодолеть несколько уровней защиты. Сначала ему нужно обойти правила доступа к сети описанные на пограничных устройствах. Затем преодолеть правила доступа заданные на самом файловом сервере. Сервис предоставляющий доступ должен также разрешить доступ. И, наконец, файловая система на диске должна авторизовать доступ к файлу.

Но если злоумышленник получит физический доступ к серверу, описанные механизмы не подействуют, тогда должны быть другие для данного направления атаки.

1.4 Политики безопасности.



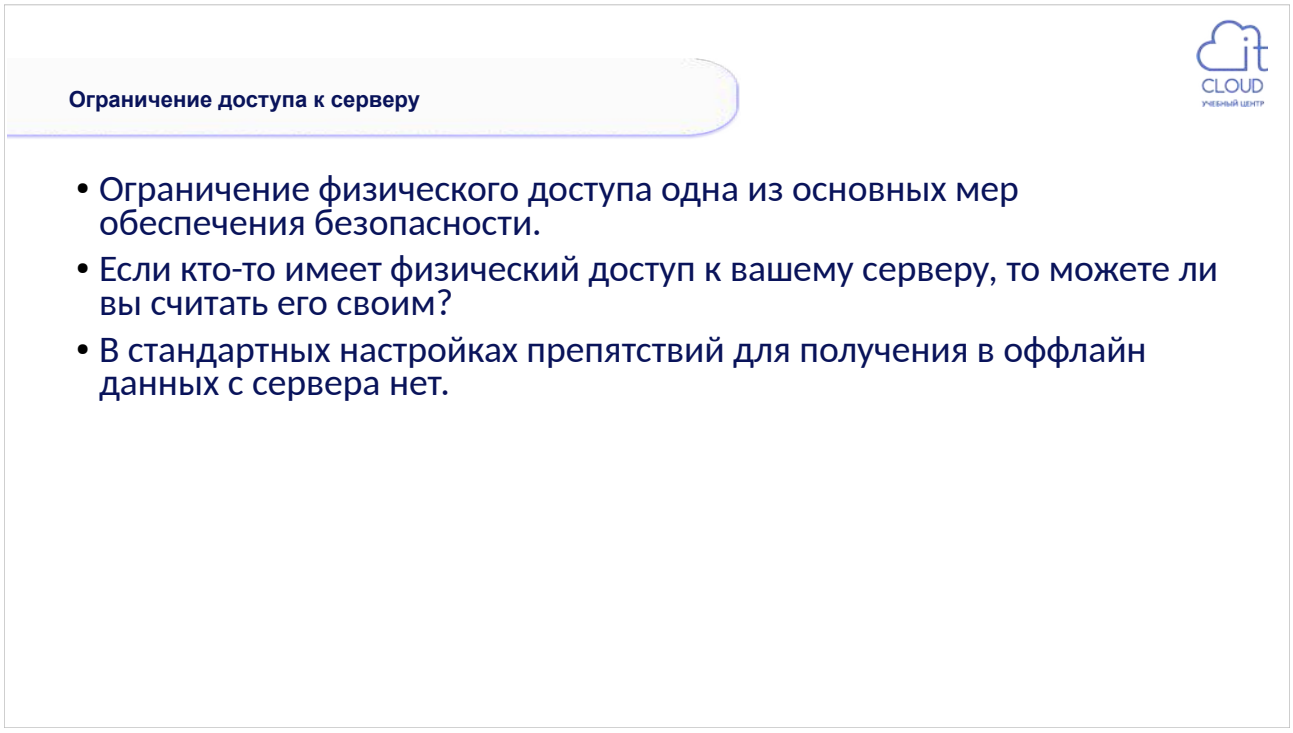
Политика безопасности — набор задач для организации, правил поведения для пользователей и администраторов, требований к системам и управлению ими. Политика безопасности основа построения безопасности в организации. Это «живой» документ, работа над которым никогда не должна заканчиваться.

Политика безопасности транслирует, проясняет и сообщает позицию руководства в отношении безопасности, и определяет основные принципы обеспечения безопасности.

Политика безопасности определяет приемлемое и неприемлемое поведение любого сотрудника организации, а так же лиц не относящихся к организации.

Глава 2. Обеспечение физической безопасности Linux-сервера.

2.1 Ограничение доступа к серверу.

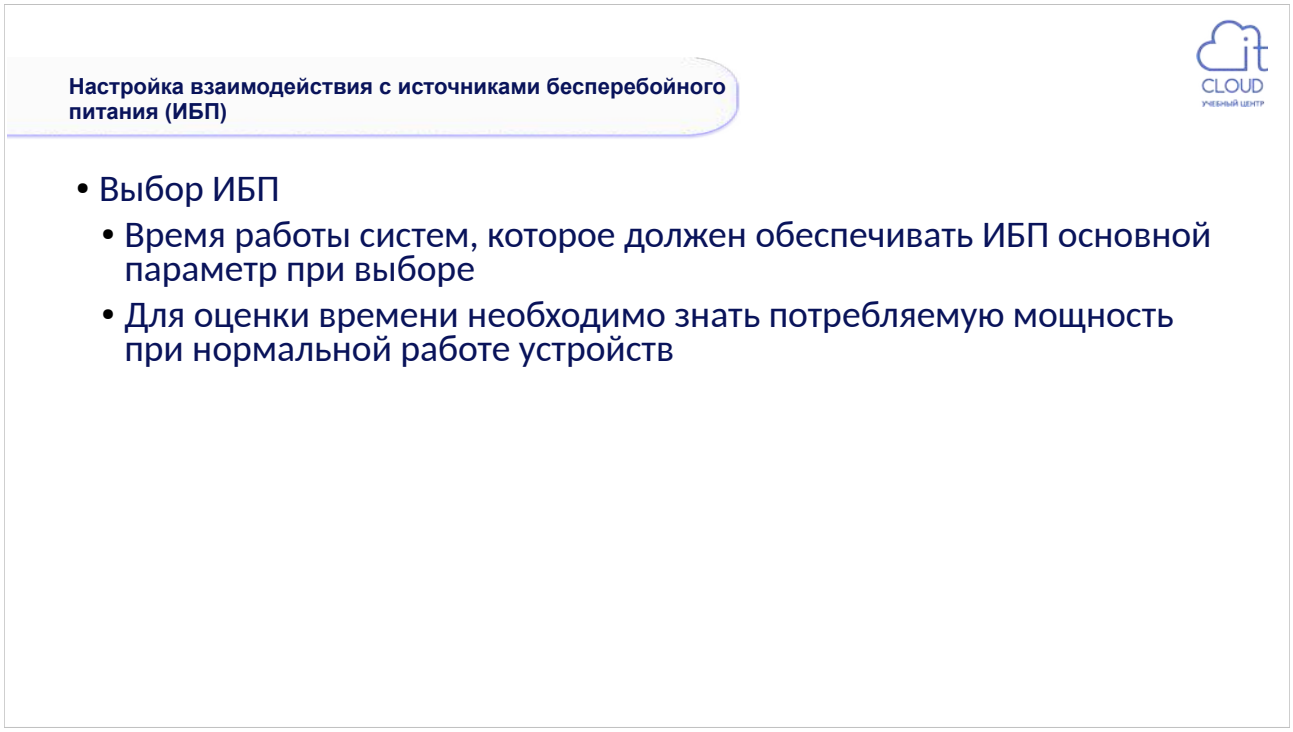


Ограничение физического доступа к вычислительным устройствам одна из основополагающих мер по обеспечению безопасности. Почти во всех операционных системах, в т.ч. в GNU/Linux, нет оффлайн защиты в стандартной инсталляции.

Поучив физический доступ к вашему компьютеру злоумышленник сможет:

- Поменять пароль любому пользователю, в том числе root.
- Скопировать любые файлы
- Установить программы
- ...

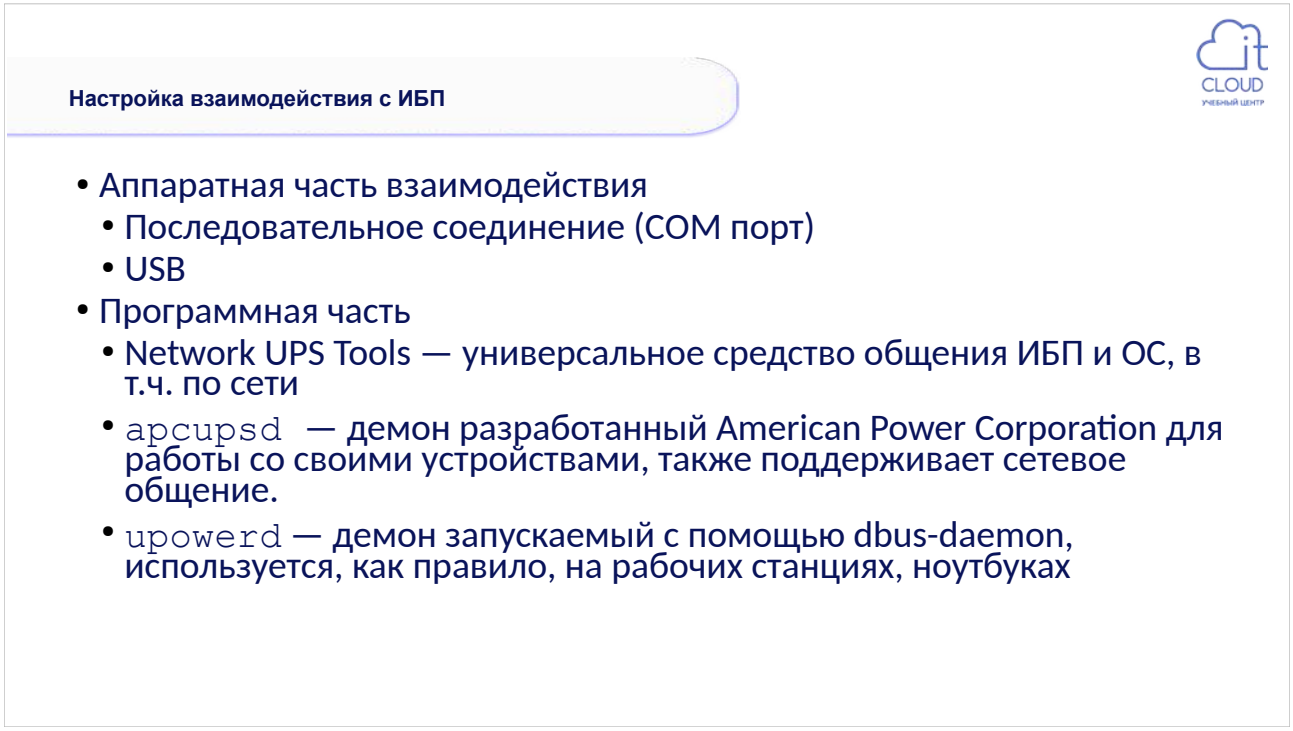
2.2 Настройка взаимодействия с источниками бесперебойного питания.



Обеспечение бесперебойного питания один из ключевых элементов в работе компьютерных систем.

Источник бесперебойного питания (ИБП) должен обеспечивать адекватное время работы в отсутствие основного питания. Прежде чем закупать ИБП, оцените время, в течение которого, ИБП должен поддерживать работу устройств. Чтобы оценить время работы устройств, необходимо знать потребляемую мощность устройств, подключенных к ИБП.

Возможно потребуется поддержка длительной работы устройств от альтернативного питания, тогда можно рассмотреть установку генератора.

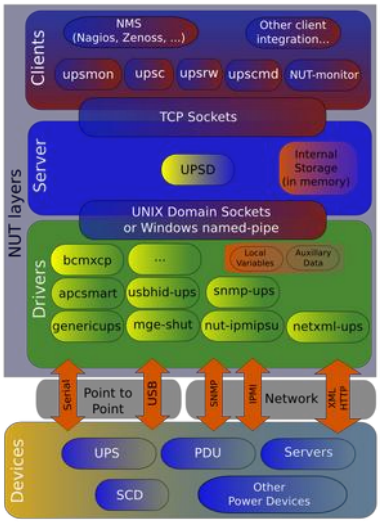


Для обратной связи ИБП и систем, которые он обслуживает, необходимо настроить взаимодействие между ними. Обычно ИБП для взаимодействия с ОС используют последовательное соединение или USB интерфейс. При этом один ИБП, как правило, обслуживает несколько устройств.

Демоны, такие как Network UPS Tools (NUT) или apcupsd позволяют получить подробные сведения о работе ИБП, в том числе тем устройствам, которые не имеют непосредственного управляющего соединения с ИБП.

Демон upowerd прежде всего предназначен для работы в ноутбуках, которые монопольно используют батарею.

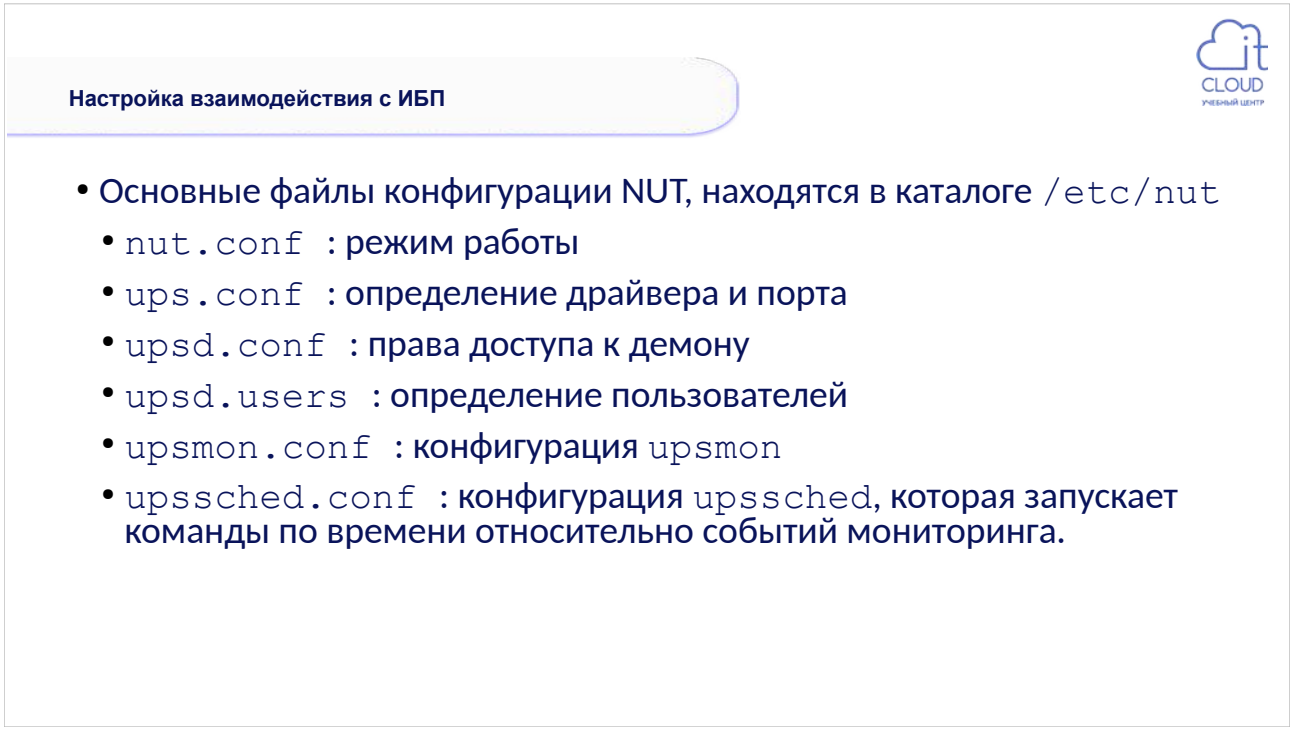
Схема работы NUT



http://networkupstools.org/docs/developer-guide.chunked/ar01s02.html#_the_layering

Основная схема работы NUT: Устройства взаимодействуют с ОС посредством драйвера, который зависит от выбранного устройства. Драйверы предоставляют доступ к функциям ИБП демону upsd, который агрегирует и предоставляет информацию клиентам через стандартное сетевое взаимодействие.

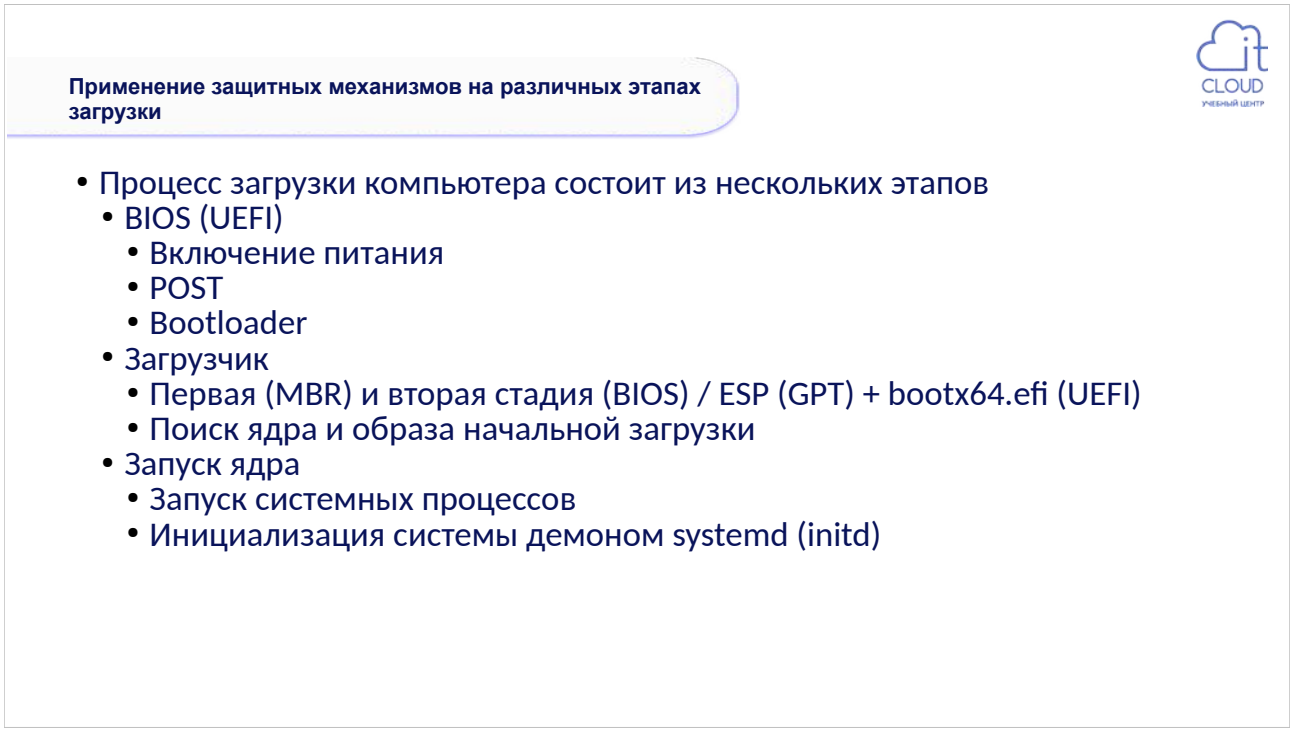
http://networkupstools.org/documentation.html



Для настройки пакета nut необходимо определить параметры в нескольких файлах в каталоге `/etc/nut` (в некоторых системах `/etc/ups`):

- `nut.conf` — определяется режим работы NUT, с помощью параметра `MODE`:
  - `none` — не настроен, т. е. ничего не запускается.
  - `standalone` — локальная конфигурация, стартуют `driver`, `upsd`, `upsmon`
  - `netserver` — тоже что и режим standalone, но требуется указать параметры сетевого доступа к upsd
  - `netclient` — запускается только `upsmon` в режиме сетевого клиента.
- `ups.conf` — определяет какой драйвер обслуживает работу с ИБП и через какой порт. Поддерживаются последовательные порты, USB и сетевые (SNMP, IPMI, XML, HTTP) устройства. Чтобы определить драйвер просмотрите файл `/usr/share/nut/driver.list`.
- `upsd.conf` — определяет параметры доступа к демону `upsd`.
- `upsmon.conf` – настройка работы программы `upsmon`.
- `upssched.conf` — конфигурация `upssched`, которая запускает команды по времени относительно событий мониторинга

2.3 Применение защитных механизмов на различных этапах загрузки.

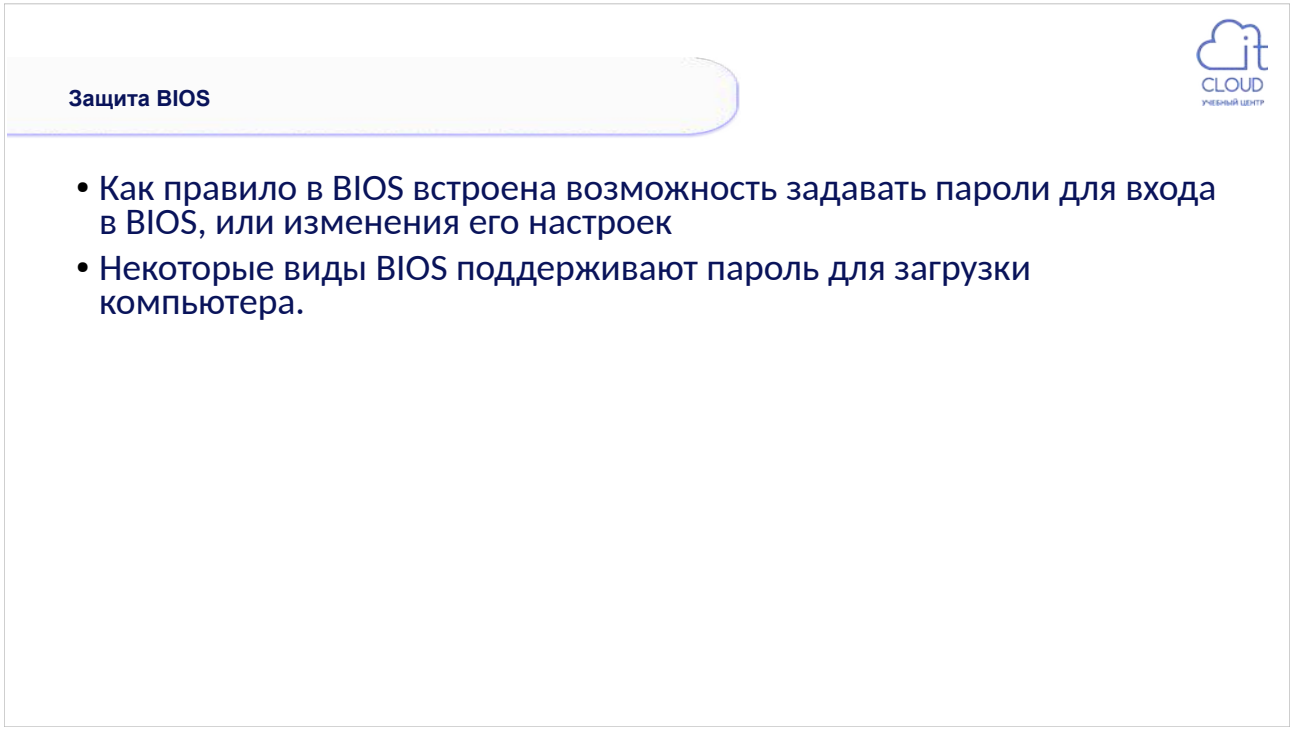


Запуск компьютера и инициализация операционной системы, как правило происходит в три этапа

1. При включении питания начинают работать программы вшитые в ПЗУ компьютера. В их задачу входит:
    1. Проверка общей работоспособности компьютера (POST)
    2. Выбор устройства для загрузки ОС и запуск загрузчика (bootloader)
2. Запуск загрузчика ОС многоэтапный процесс:
    1. Сначала bootloader находит первую стадию загрузчика в MBR или нулевом секторе активного раздела. И запускает ее. Первая стадия — очень простая программа, в задачу которой входит поиск и запуск второй стадии загрузчика. В системах с UEFI bootloader запускает на разделе ESP (EFI System Partition) файл загрузчика `bootx64.efi`.
    2. Вторая стадия ищет и запускает ядро операционной системы и использует образ начальной загрузки в качестве первоначальной корневой файловой системы.
3. Ядро ОС после инициализации выполняет две операции:
    1. Запускает системные процессы
    2. Находит и запускает программу инициализации ОС (init)

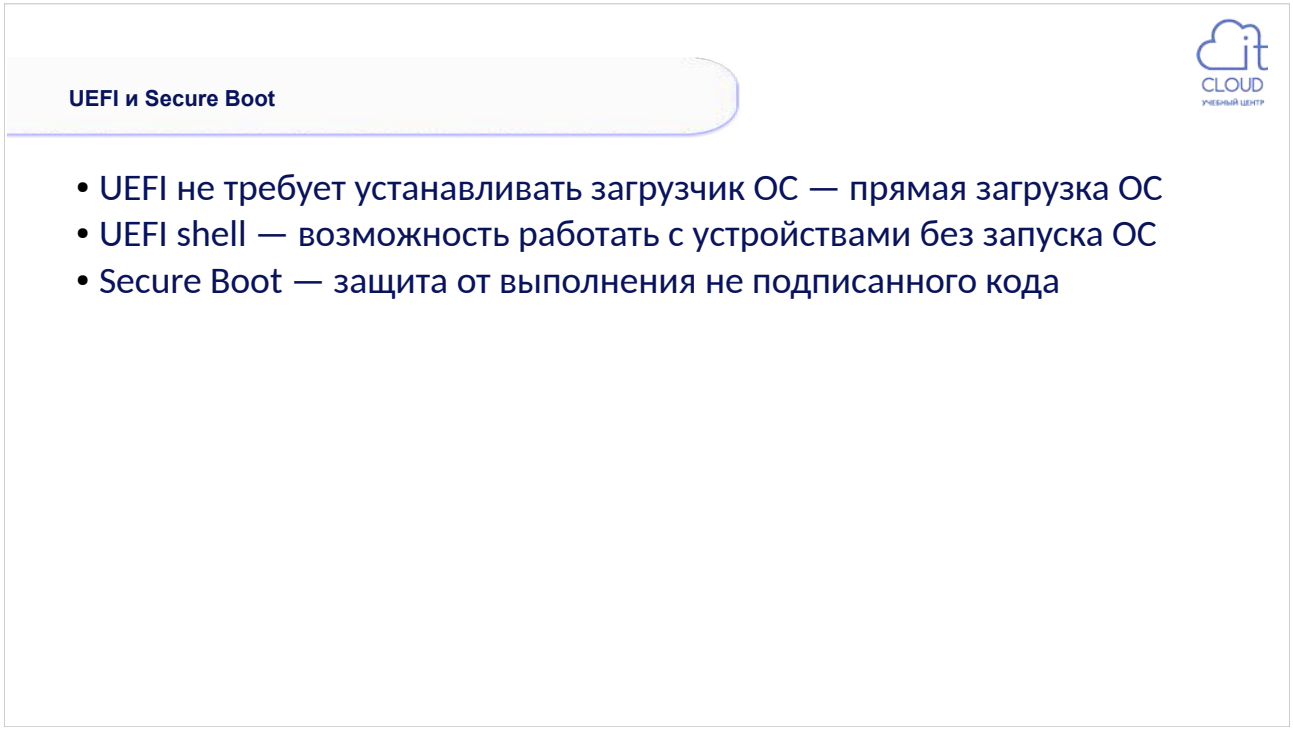
Глава 2. Обеспечение физической безопасности Linux-сервера.

На любом из этапов загрузки злоумышленник может вмешаться в этот процесс, если получит физический доступ к компьютеру. В результате чего может быть нарушена безопасность системы.



BIOS (Basic Input Output System) — имеет несколько механизмов защиты. Как правило вы можете настроить пароль на доступ в CMOS (complementary metal-oxide-semiconductor) BIOS или на изменение его настроек. В некоторых BIOS можно установить пароль на запуск компьютера.

Если злоумышленнику известна модель системной платы, то он может легко найти инженерные пароли для обхода такой защиты.

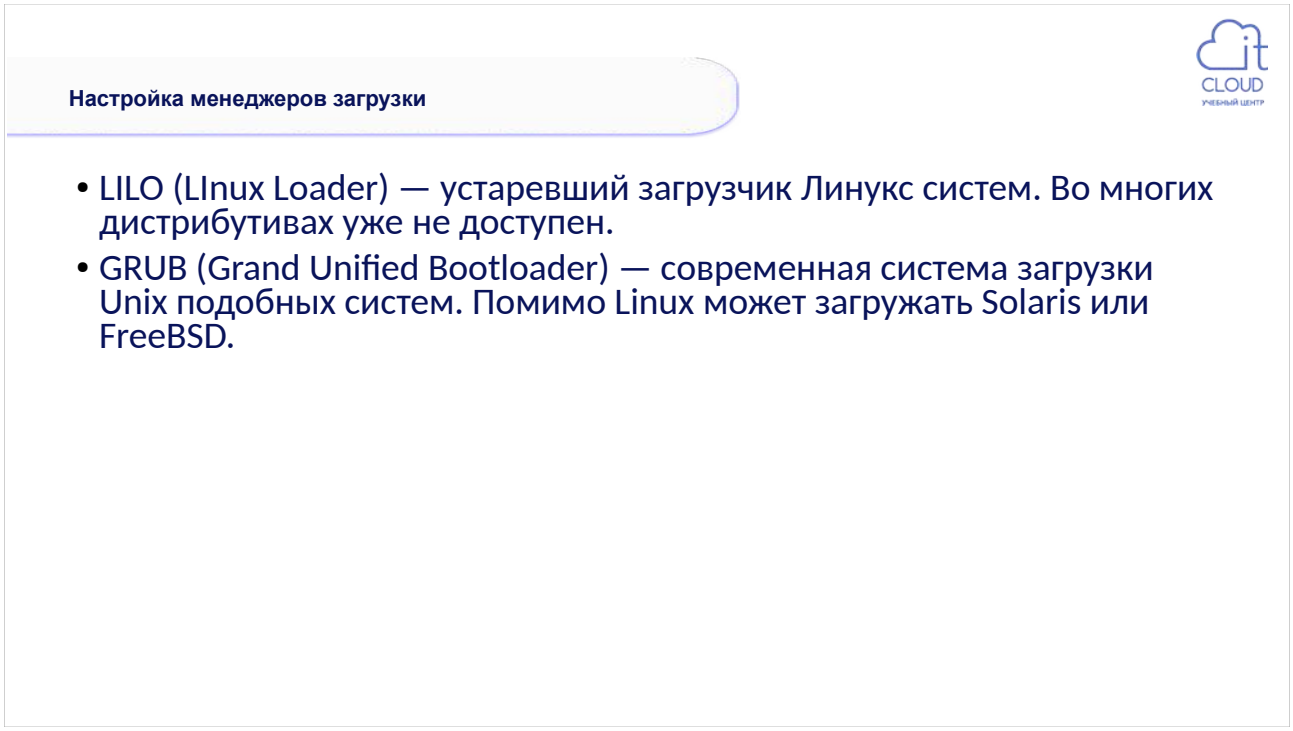


UEFI (Unified Extensible Firmware Interface) — новое поколение интерфейса между операционной системой и микропрограммами, управляющими низкоуровневыми функциями оборудования. В UEFI имеется несколько дополнительных механизмов, в том числе Secure Boot.

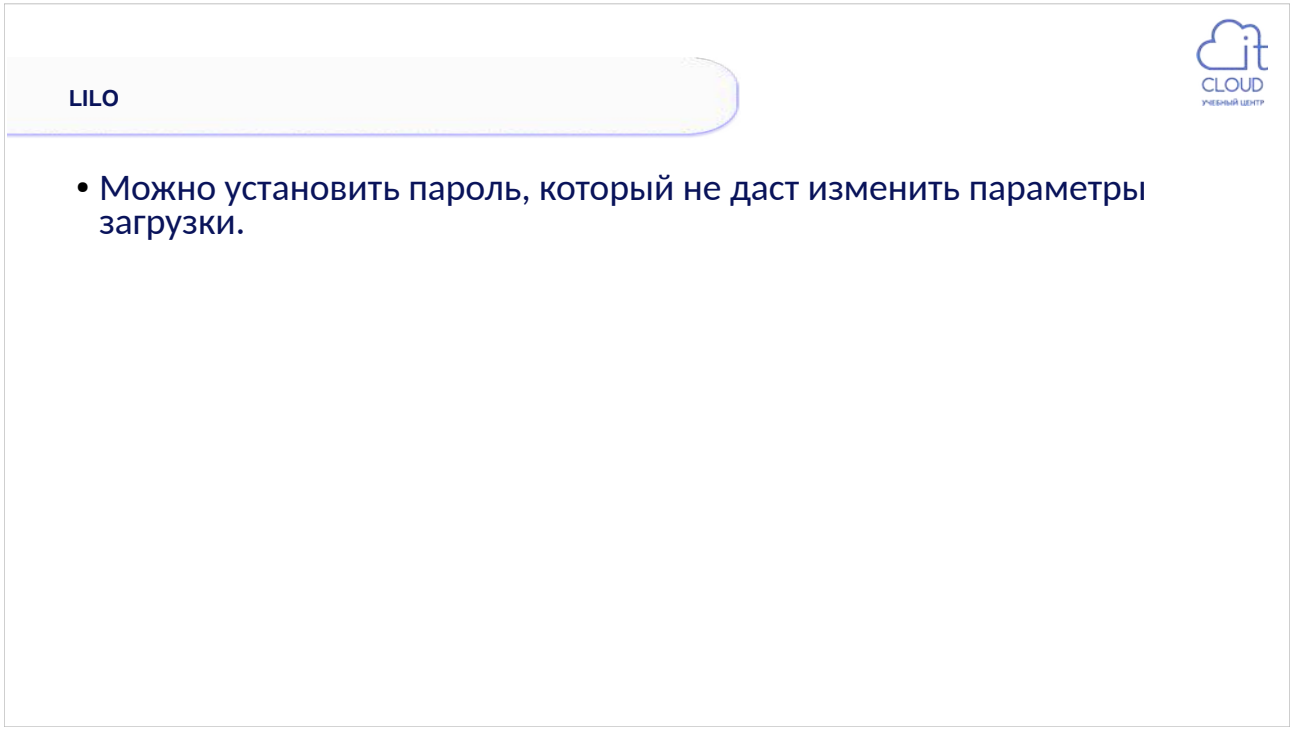
Secure Boot проверяет наличие цифровых подписей компонентов ядра при их загрузке в память.

UEFI умеет самостоятельно загружать ОС, в этом случае на компьютер можно не устанавливать загрузчик.

2.4 Настройка менеджеров загрузки LILO, GRUB.

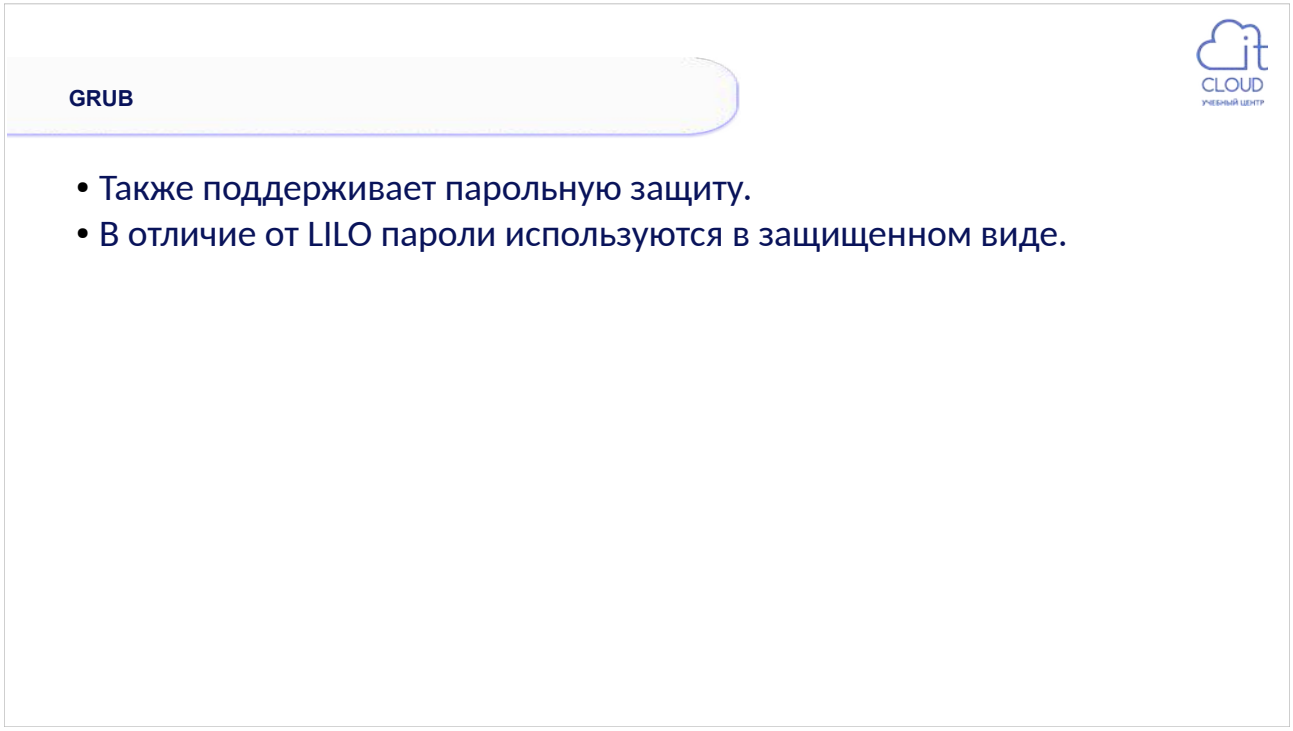


В Линукс применяются два вида загрузчиков LILO (устарело) и GRUB.



В LILO можно лишь установить пароль на изменение параметров загрузки ядра.

Пароль можно задать опцией `password=` и `restrict`. Если не указывать опцию `restrict`, то пароль необходимо будет вводить для запуска из выбранного пункта меню.



GRUB также поддерживает защиту в виде паролей на доступ к параметрам загрузки ОС.

Для установки пароля на доступ в GRUB имеется специальная команда `grub2-setpassword`, которая создает файл `/boot/grub2/user.cfg` с паролем для пользователя root

Пример:

```
# grub-mkpasswd-pbkdf2
Введите пароль:
Повторно введите пароль:
Хэш PBKDF2 вашего пароля:
grub.pbkdf2.sha512.10000.BC259FB35201B07343A9FC3E917E8EA9A577BE0DB929CC9B1BC052A
9104E7F84A49AC614040C331B4DE710F3D33115F171C2129DA3EF75544E57924198AF0C61.CA042C
5A4D9596D43768E5D0B6C01670B160099608B9604B3EA13B83861DE959C20169CDC23DDCD3BD5B1F
1E9270590F7ACFB12EDEE2B6E357677FF35BD5437E
```

Далее внести строки для настройки пользователя и пароля в файл `/etc/grub.d/40_custom`:

```
#!/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.
set superusers="grub"
password_pbkdf2 grub grub.pbkdf2.sha512.10000.BC259FB35201B07343A9FC3E917E8EA9A
577BE0DB929CC9B1BC052A9104E7F84A49AC614040C331B4DE710F3D33115F171C2129DA3EF7554
4E57924198AF0C61.CA042C5A4D9596D43768E5D0B6C01670B160099608B9604B3EA13B83861DE9
59C20169CDC23DDCD3BD5B1F1E9270590F7ACFB12EDEE2B6E357677FF35BD5437E
```

В файле `/etc/grub.d/10_linux` изменить строку:

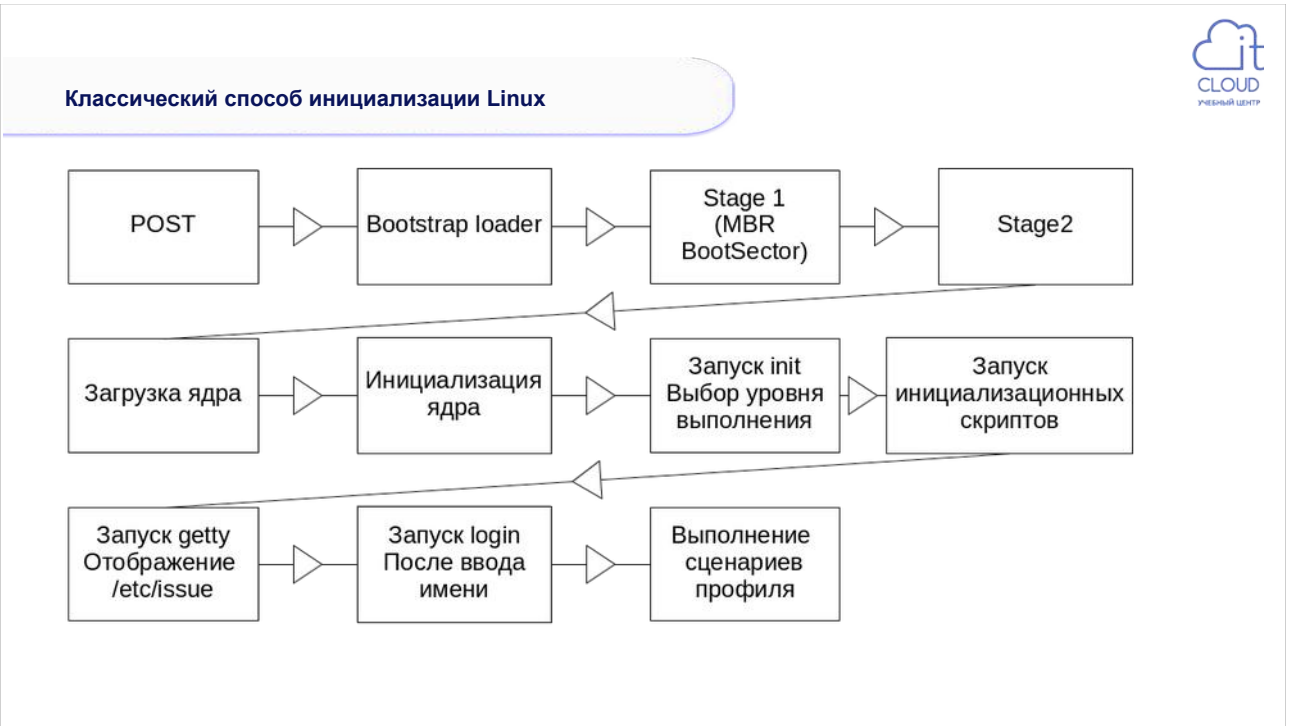
```
CLASS="--class gnu-linux --class gnu --class os"
```

на

```
CLASS="--class gnu-linux --class gnu --class os —unrestricted"
```

Если этого не сделать, то при каждом запуске потребуется вводить имя и пароль, чтобы продолжить загрузку.

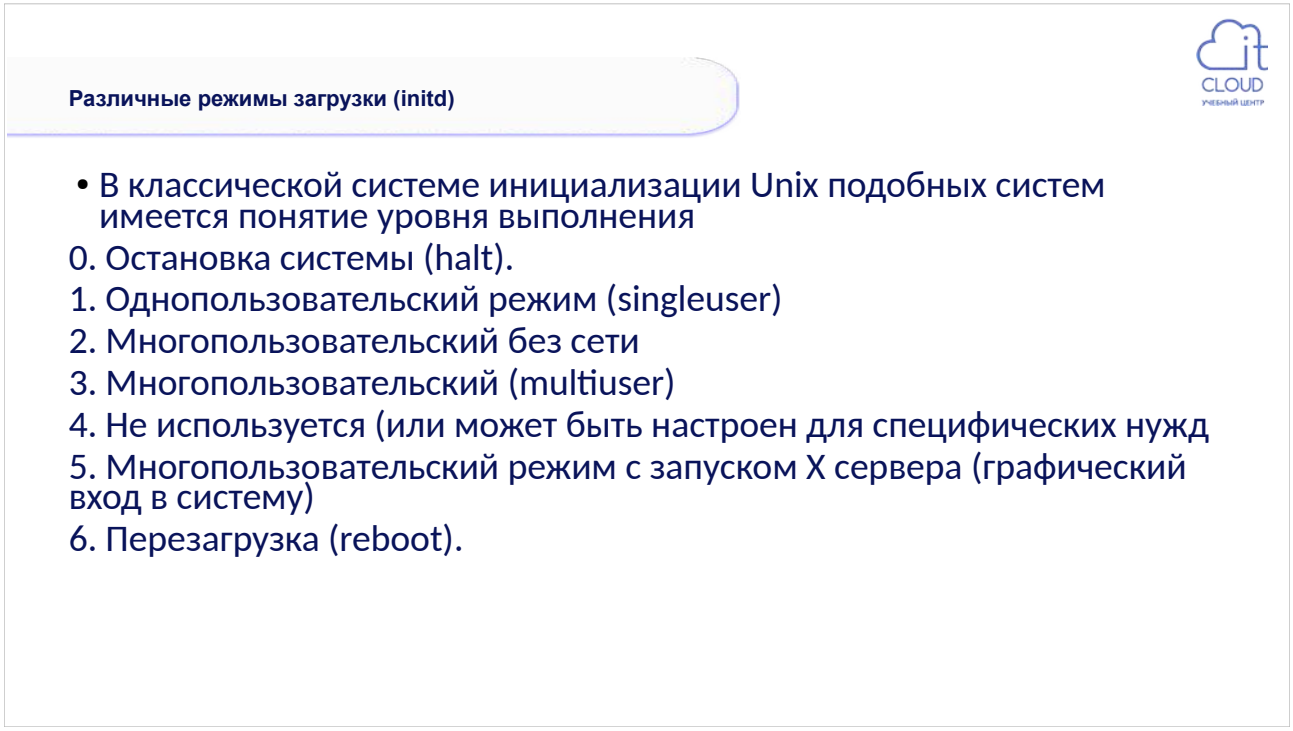
2.5 Различные режимы загрузки.



В классической системе инициализации Unix-подобных систем предусматривается запуск специального процесс с PID=1 под названием init.

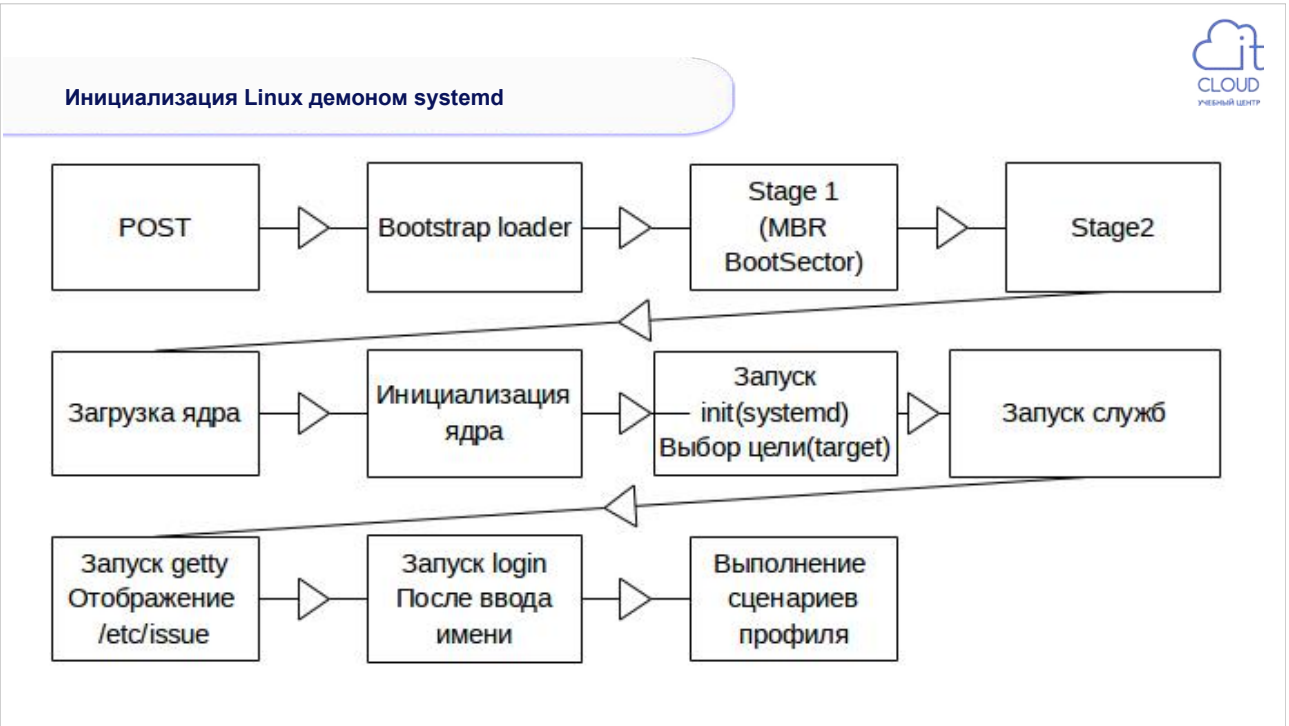
Задача init инициализация операционной системы и поддержание ее работоспособности.

Одна из решаемых задач — перевод ОС на нужный уровень выполнения (run level). Уровень выполнения означает режим функционирования операционной системы компьютера, в которой реализована инициализация в стиле ОС UNIX System V.

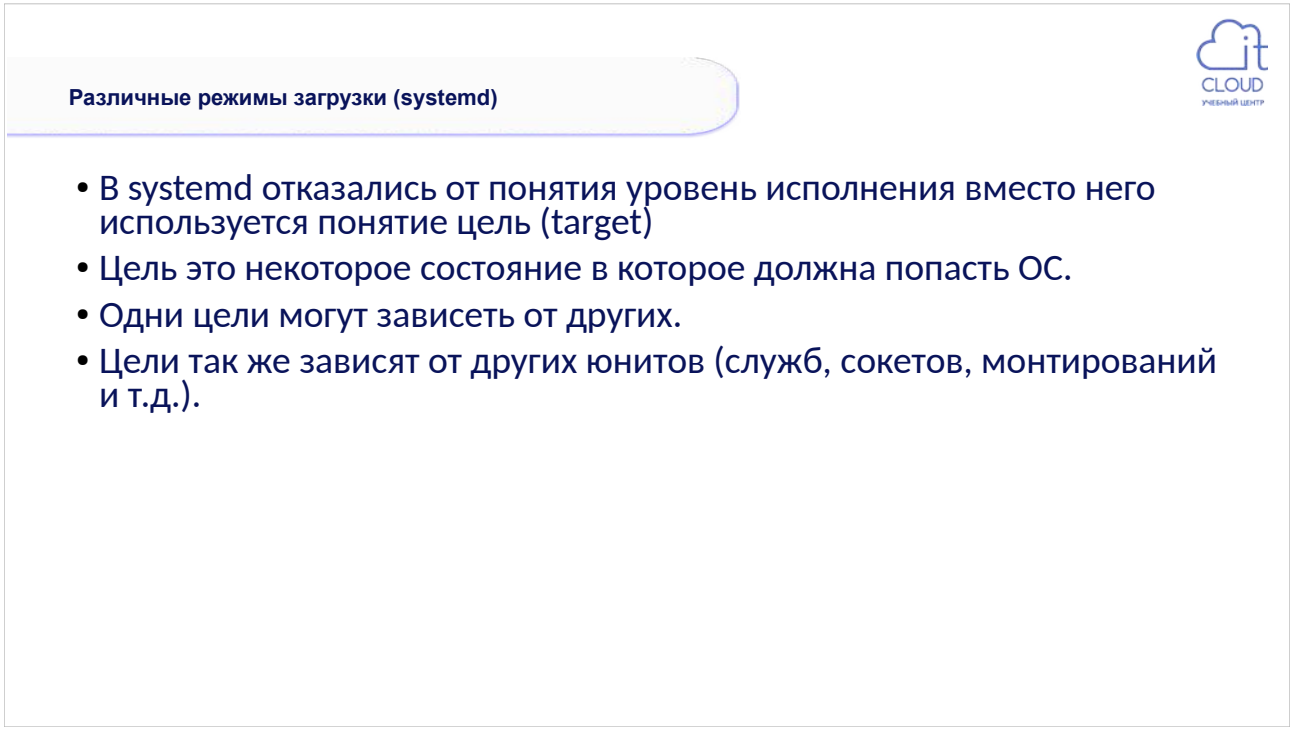


Команда `runlevel` позволяет узнать уровень исполнения

Команда `telinit` или `init` может сменить уровень исполнения. В качестве аргумента указывается нужный уровень выполнения.



В системах использующих демон `systemd` все функции `init` выполняет `systemd`. В таких системах понятие уровень выполнения оставлено для обратной совместимости.



В системах с `systemd` используется понятие цель (target).

Цель — промежуточное или конечное состояние, в которое должна перейти ОС. Для достижения нужной цели sysytemd запускает различные юниты (сервисы, сокеты, цели, слайсы, монтирования и др.) от которых зависит данная цель.

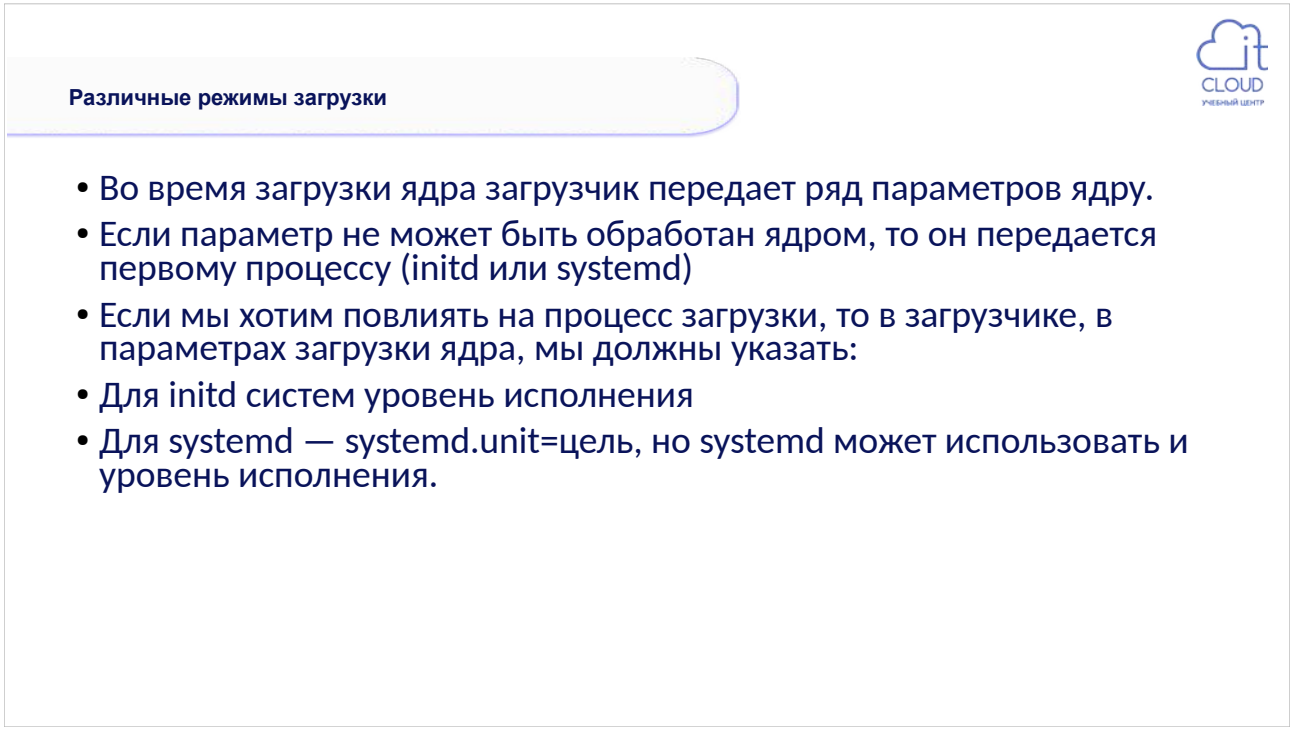
Цель по умолчанию (default target) это состояние, в котором ОС должна находиться после загрузки. Например multi-user.target или graphical.target

Команда `systemctl get-default` показывает текущую цель по умолчанию

Команда `systemctl set-default <`*`название_цели`*`>` устанавливает цель по умолчанию.

Пример:

```
root@s10:~# systemctl get-default
multi-user.target
root@s10:~# systemctl set-default emergency.target
Removed symlink /etc/systemd/system/default.target.
Created symlink from /etc/systemd/system/default.target to
/usr/lib/systemd/system/emergency.target.
root@s10:~#
```

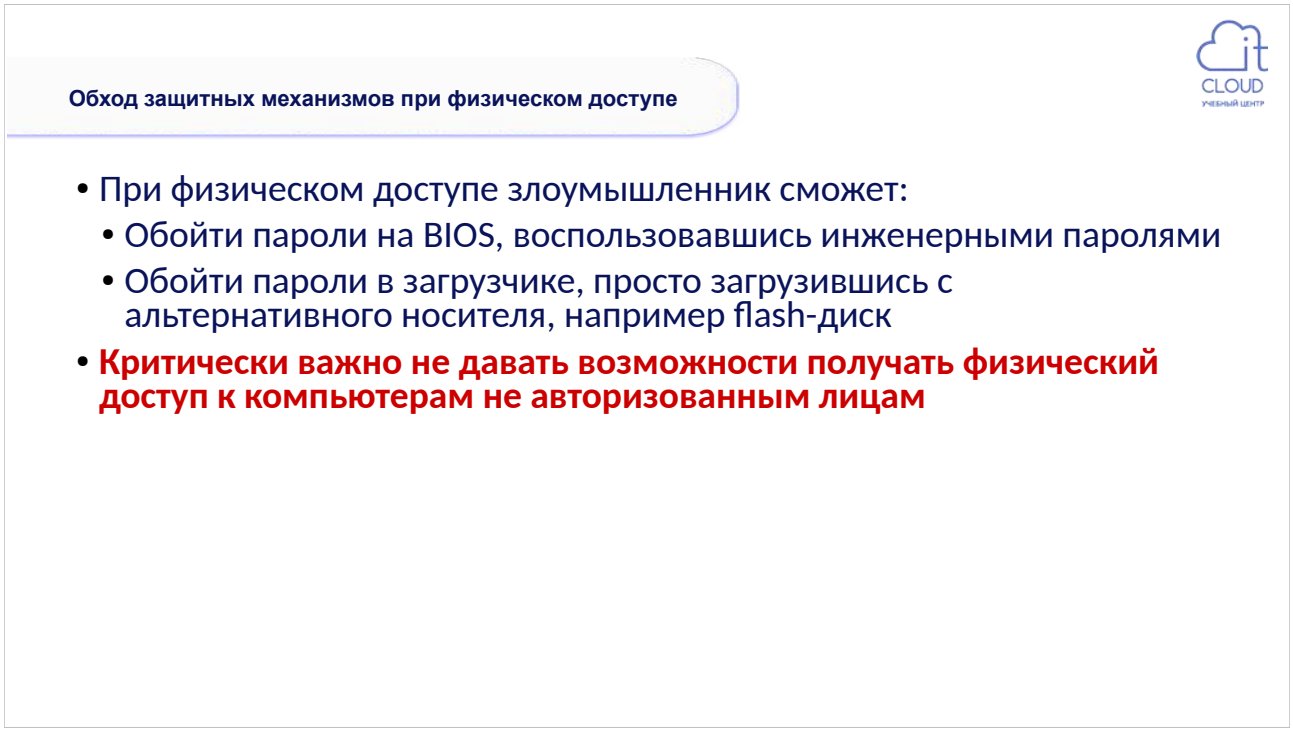


Если мы хотим временно использовать другой уровень исполнения или цель, то в процессе загрузки нужный вариант указывается как параметр загрузки ядра.

Пример:

```
linux   /boot/vmlinuz-6.1.0-25-amd64 root=UUID=7190a38a-7fac-4376-a220-1
9f8814fa8f3 ro  quiet systemd.unit=emergency.target
```

2.6 Обход защитных механизмов при физическом доступе.



Важно помнить, что получив физических доступ к компьютеру злоумышленник может легко обойти защитные механизмы загрузки, такие как пароли на BIOS или загрузчик.

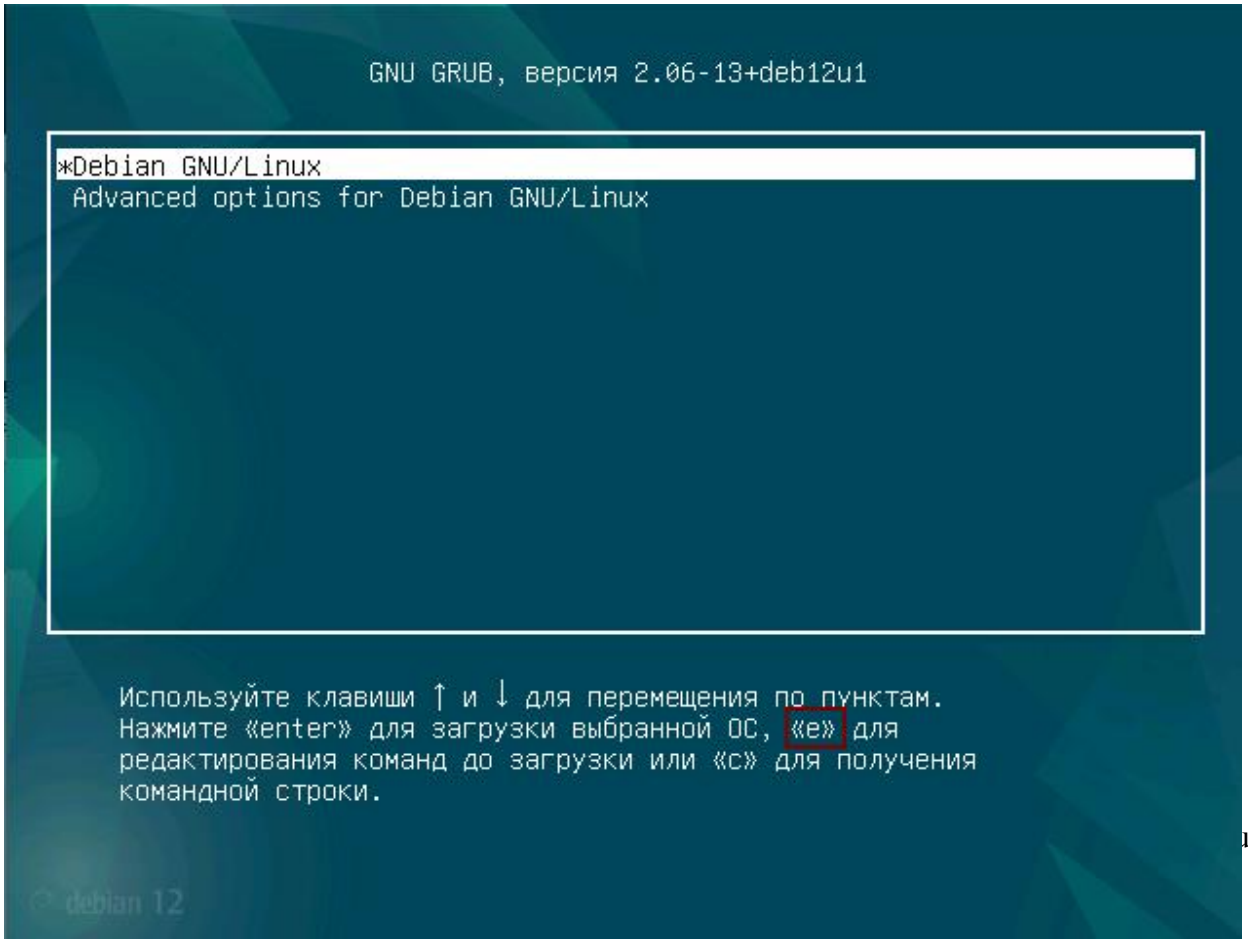
2.7 Возможности по восстановлению пароля пользователя root при физическом доступе к серверу.

Возможности по восстановлению пароля пользователя root при физическом доступе к серверу

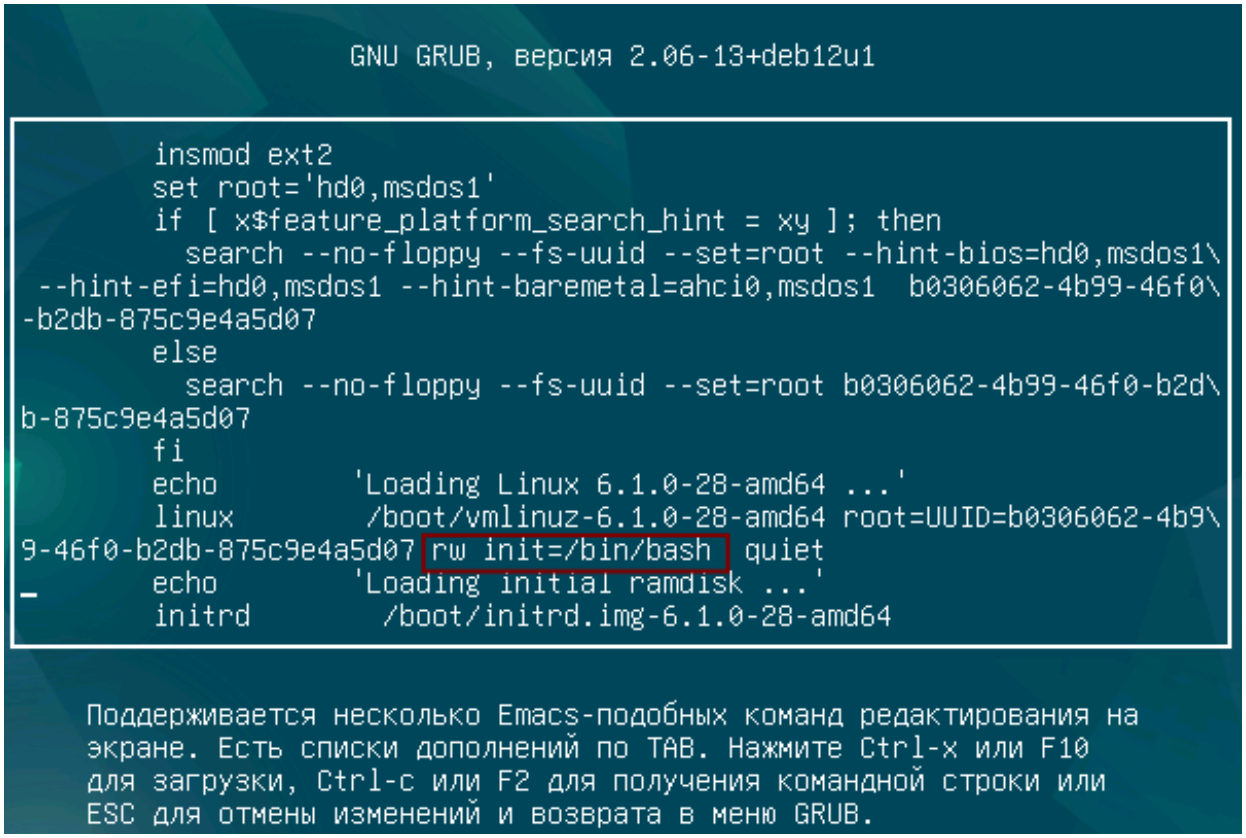
- Если вы забыли пароль суперпользователя, то вы можете провести процедуру сброса пароля

Если вы забыли пароль суперпользователя (или любого другого локального пользователя), то можно легко произвести процедуру сброса пароля:

1. Во время старта компьютера, в загрузчике, в строке выбора ОС нажмите клавишу **«e»**



2. Найдите строку, которая описывает какое ядро и с какими параметрами загружается. В конце строки напишите параметр **`init=/bin/bash`**. Затем нажмите `Ctrl-x`.



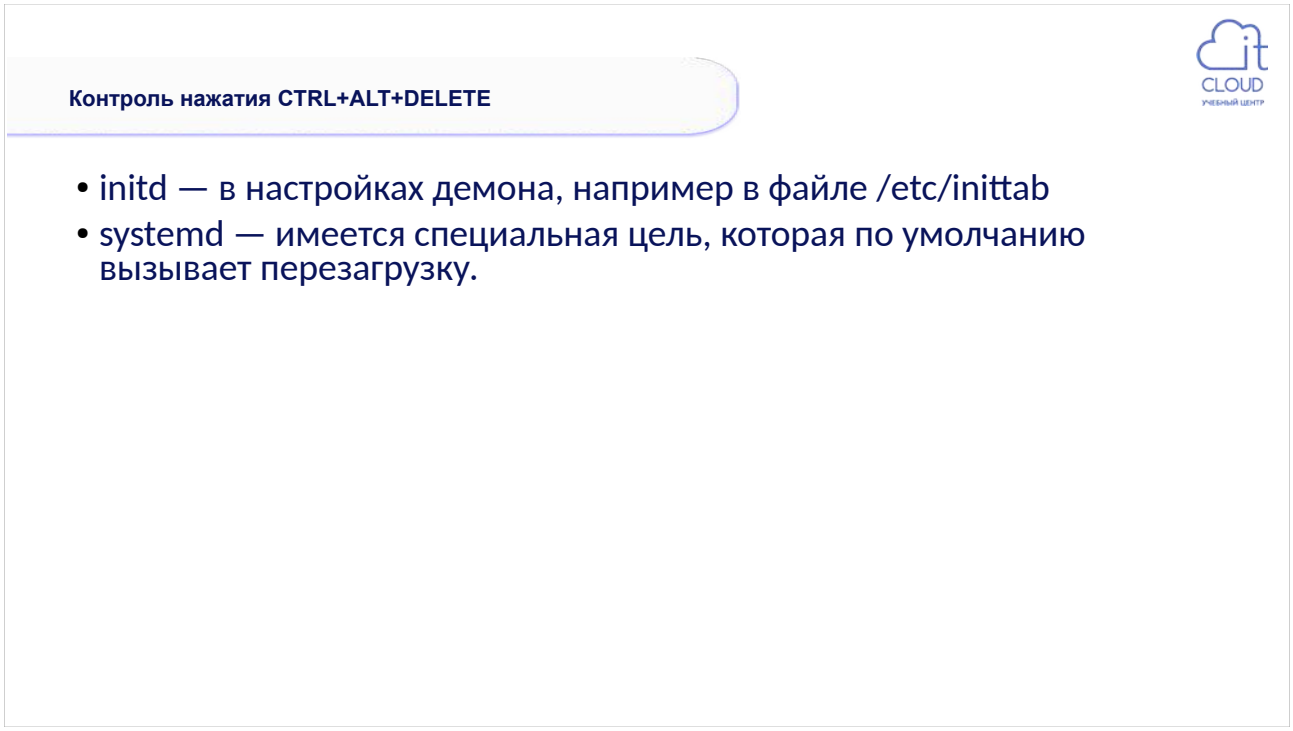
3. Выполните команду **`mount -o remount,rw /`**, которая перемонтирует корневой раздел в режиме чтения-записи, если вы не указали параметр ядра rw.
4. Командой `passwd` изменяете пароль суперпользователю.

```
bash-4.2# passwd
Changing password for user root.
New password:
BAD PASSWORD: The password fails the dictionary check - it is too
simplistic/systematic
Retype new password:
passwd: all authentication tokens updated successfully.
```

5. Выполняете команды **`sync`** и **`mount -o remount,ro /`**
6. Перезапустите систему.

Примечание: Если у вас включена система SELinux, то такая процедура приведет к тому, что ни одни пользователь не сможет войти в систему. Чтобы это не произошло, необходимо создать в корневом каталоге файл `/.autorelabel` перед выполнением пункта 5.

2.8 Контроль нажатия CTRL+ALT+DELETE.



В системах с `initd` в файле `/etc/inittab` настраивается действие `ctrlaltdel`

Пример:

```
ca:12345:ctrlaltdel:
```

В системах `systemd` используется цель `ctrl-alt-del.target`.

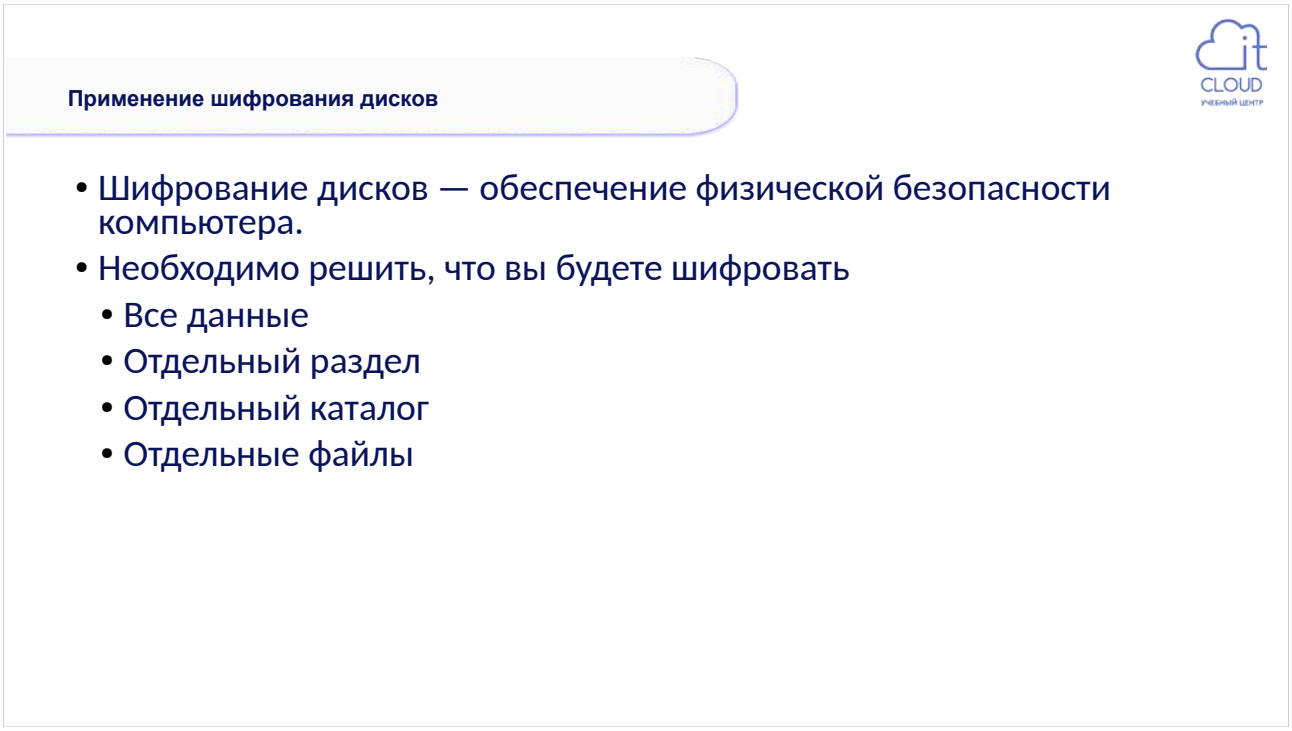
Пример:

```
root@s10:~# systemctl status ctrl-alt-del.target
● reboot.target - Reboot
   Loaded: loaded (/usr/lib/systemd/system/reboot.target; disabled; vendor
preset: disabled)
   Active: inactive (dead)
     Docs: man:systemd.special(7)
```

Для блокировки перезагрузки по CTRL+ALT+DELETE замаскируйте эту цель.

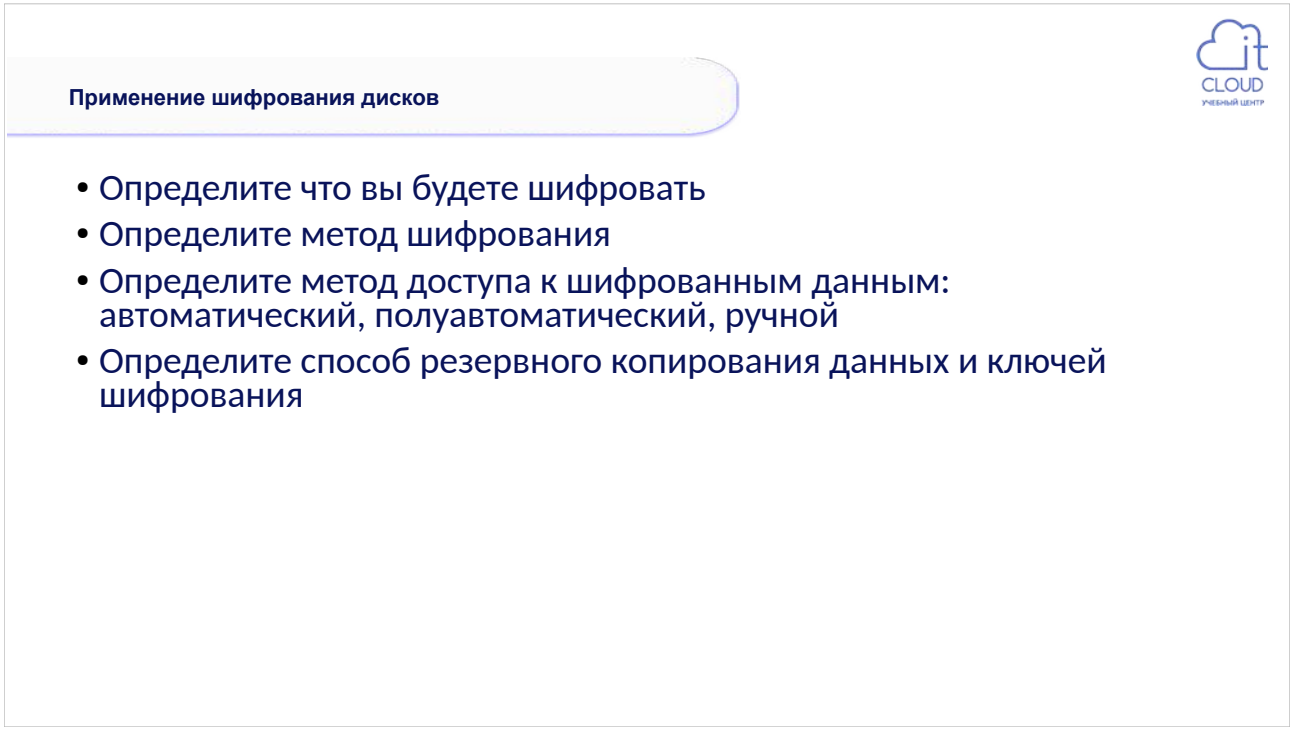
```
root@s10:~# systemctl mask ctrl-alt-del.target
Created symlink /etc/systemd/system/ctrl-alt-del.target → /dev/null.
```

2.9 Применение шифрования дисков.



Шифрование данных один из элементов по обеспечению физической безопасности компьютера. Даже если злоумышленник получит физический доступ к системе, то ему еще потребуется расшифровать файлы на диске.

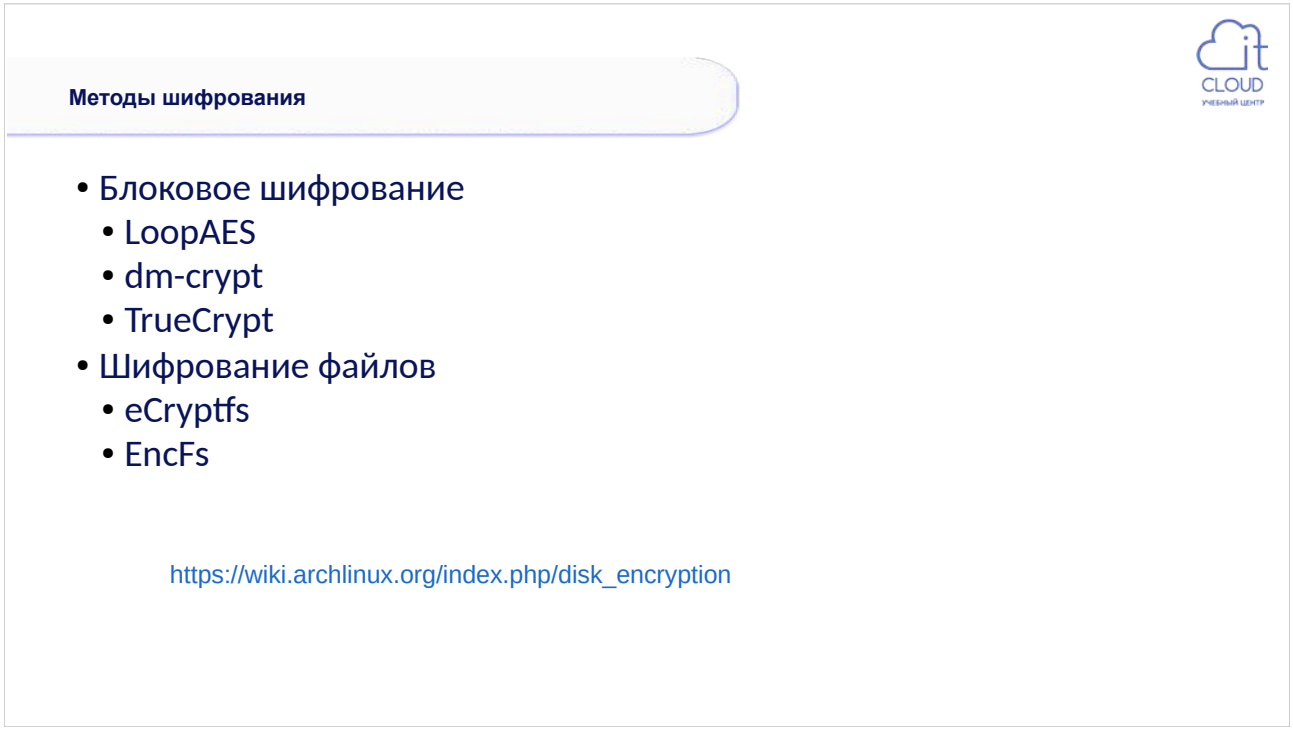
В Линукс вы можете выбрать, какой объем данных следует зашифровать.



Прежде чем шифровать данные, необходимо провести предварительное планирование. Следует учесть следующие моменты:

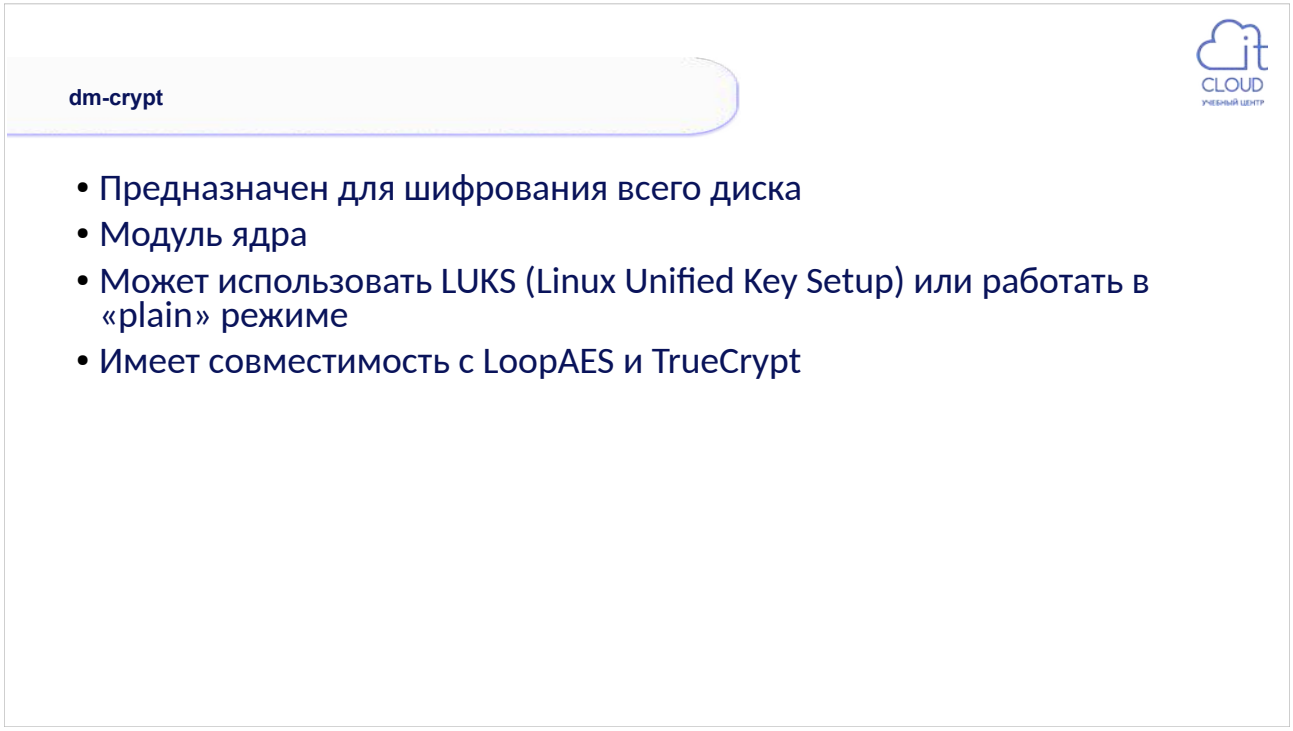
- Один из ключевых вопросов — что нужно шифровать? И нужно ли вообще что-либо шифровать? Чтобы ответь на эти вопросы нужно определить возможные угрозы системе и способно ли шифрование от них защитить.
- Определите метод шифрования.
- Какой способ будет использоваться для доступа к шифрованным данным
  - Автоматический. Ключи шифрования находятся в файле. Наименее безопасный способ.
  - Полуавтоматический. Ключ шифрования находятся на внешних устройствах.
  - Ручной. Помимо ключа шифрования или вместо него используется пароль.
- Как будет производиться резервное копирование данных и ключей. Архивация данных может быть как в шифрованном виде так и в открытом.

Надо учитывать накладные расходы на шифрование. Процесс шифрования и расшифровки требует значительных вычислительных ресурсов.



Существует два принципиально разных подхода к шифрованию:

1. Блоковое шифрование, в котором шифруются блоки данных на диске и поверх этих блоков создается файловая система.
2. Шифрование на уровне файлов. В такой системе шифруется содержимое файлов.

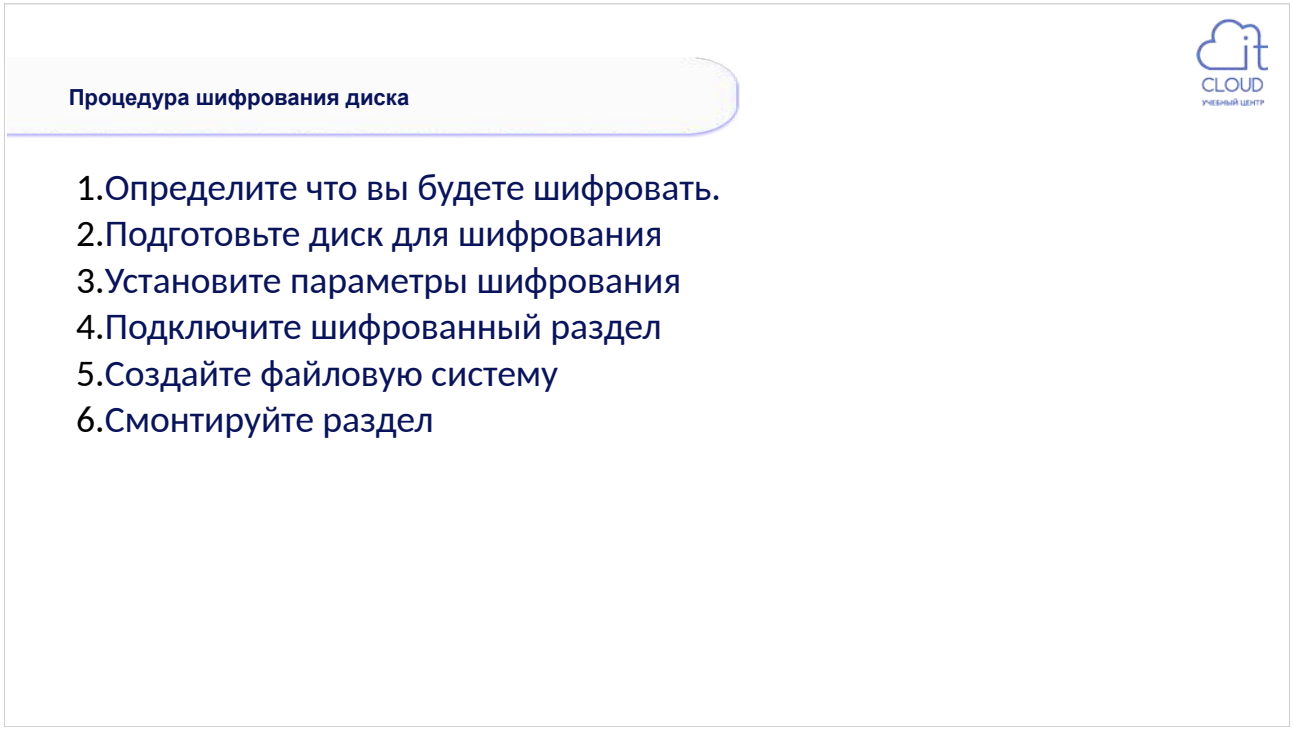


Система dm-crypt используется для блокового шифрования.

Система работает посредством модуля ядра, который производит отображение шифрованного диска в виртуальное устройство. Такое устройство можно использовать как обычное устройство хранения.

Для управления ключами предполагается использовать систему LUKS. Но dm-crypt может работать и без LUKS.

dm-crypt имеет обратную совместимость с системами LoopAES и TrueCrypt. Но все же не является их полной заменой.



Перед шифрованием диска убедитесь что у вас установлены соответствующие инструменты. Если пакет cryptsetup не установлен, то установите его.

Пример:

```
root@s10:~# apt list cryptsetup
Вывод списка… Готово
cryptsetup/stable 2:2.6.1-4~deb12u2 amd64

root@s10:~# apt install cryptsetup
Чтение списков пакетов… Готово
Построение дерева зависимостей… Готово
Чтение информации о состоянии… Готово
Будут установлены следующие дополнительные пакеты:
  cryptsetup-bin
Предлагаемые пакеты:
  cryptsetup-initramfs keyutils
Следующие НОВЫЕ пакеты будут установлены:
  cryptsetup cryptsetup-bin
...
```

1. Определите что вы будете шифровать это может быть целый диск, раздел, LVM том. Например, можно зашифровать весь USB диск `/dev/sdb`.
2. Подготовка диска — заполнение диска случайными данными. Этот этап можно пропустить, но тогда данные будут уязвимыми.

Пример:

```
shred -v --iterations=1 /dev/sdb
```

3. Утилитой `cryptsetup` определите параметры шифрования.

Пример:

```
root@s10:~# cryptsetup --cipher=aes-cbc-essiv:sha256 --key-size 256 \
```

```
luksFormat /dev/sdb

ПРЕДУПРЕЖДЕНИЕ!
========
Данные на /dev/sdb будут перезаписаны без возможности восстановления.

Вы уверены? (введите «yes» заглавными буквами): YES
Введите парольную фразу для /dev/sdb:
Парольная фраза повторно:
```

4. Создайте виртуальное устройство для шифрованного диска

Пример:

```
root@s10:~# cryptsetup open /dev/sdb encsdb
Введите парольную фразу для /dev/sdb:

root@s10:~# ls -l /dev/mapper/
итого 0
crw------- 1 root root 10, 236 фев  3 12:07 control
lrwxrwxrwx 1 root root       7 фев  3 12:16 encsdb -> ../dm-0
```

5. Создайте файловую систему

Пример:

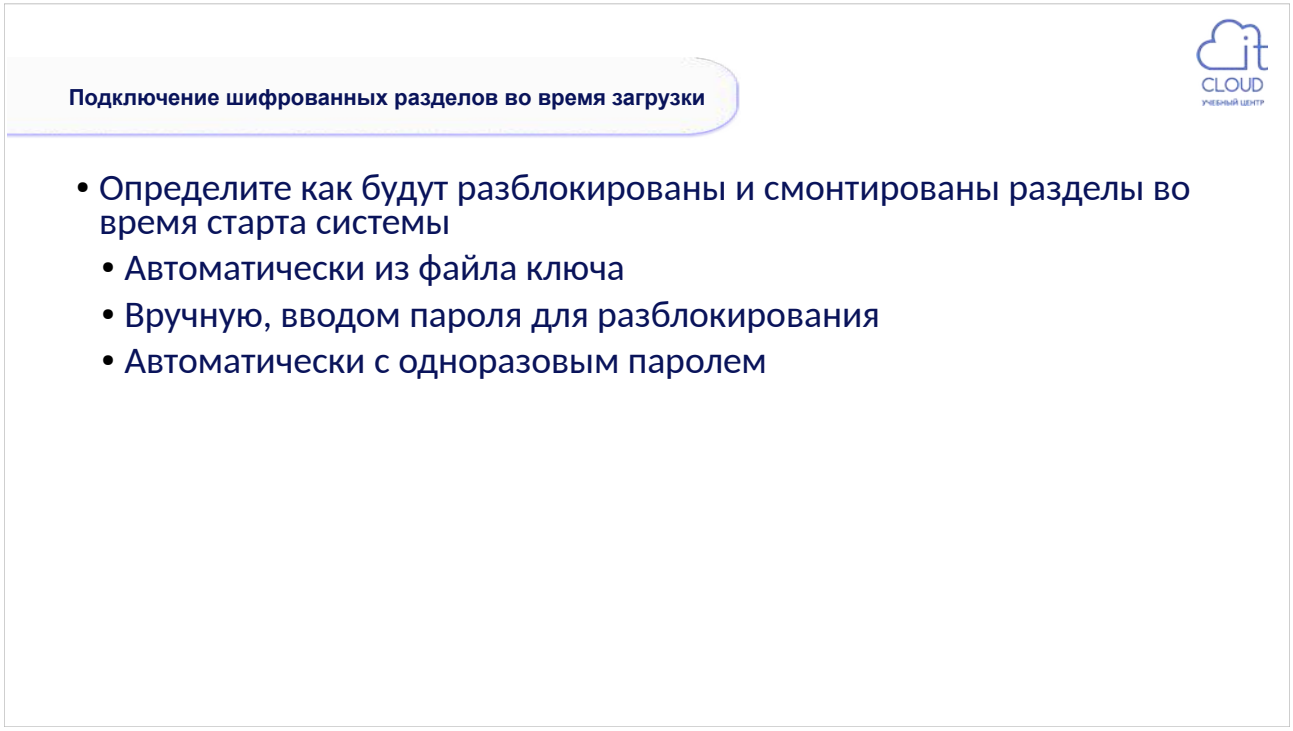
```
root@s10:~# mkfs.ext4 /dev/mapper/encsdb
mke2fs 1.47.0 (5-Feb-2023)
Creating filesystem with 5238784 4k blocks and 1310720 inodes
Filesystem UUID: 06f2298a-b253-41d7-b294-dbe0373a87f6
Superblock backups stored on blocks:
      32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
      4096000

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
```

6. Смонтируйте файловую систему.

Пример:

```
root@s10:~# mount /dev/mapper/encsdb /mnt/
root@s10:~# echo 123 > /mnt/test.file
root@s10:~# umount /mnt

root@s10:~# cryptsetup close encsdb
root@s10:~# ls -l /dev/mapper/
итого 0
crw------- 1 root root 10, 236 фев  3 12:07 control

root@s10:~# blkid /dev/sdb
/dev/sdb: UUID="50658916-4e42-467b-b2bd-21c919644feb" TYPE="crypto_LUKS"
root@s10:~# mount /dev/sdb /mnt/
mount: /mnt: unknown filesystem type 'crypto_LUKS'.
       dmesg(1) may have more information after failed mount system call.
```

Показанный выше пример показывает как применить шифрование в ручном виде.



Подключение шифрованных разделов во время старта системы контролируется целью cryptsetup.target

Пример:

```
root@s10:~# systemctl status cryptsetup.target
● cryptsetup.target - Local Encrypted Volumes
     Loaded: loaded (/lib/systemd/system/cryptsetup.target; static)
     Active: active since Mon 2025-02-03 12:07:38 +05; 12min ago
       Docs: man:systemd.special(7)
```

Для настройки этой цели используется файл `/etc/crypttab`.

Пример:

Создаем файл со случайными данными, который будем использовать как ключ

```
root@s10:~# dd if=/dev/random of=/root/mykey_encsdb bs=1 count=1024
1024+0 records in
1024+0 records out
1024 bytes (1,0 kB, 1,0 KiB) copied, 0,0349917 s, 29,3 kB/s
```

Добавляем ключ в хранилище

```
root@s10:~# cryptsetup luksAddKey /dev/sdb /root/mykey_encsdb
Введите любую существующую парольную фразу:
```

Настраиваем `/etc/crypttab` для автоматического подключения шифрованного диска.

```
root@s10:~# blkid /dev/sdb
/dev/sdb: UUID="50658916-4e42-467b-b2bd-21c919644feb" TYPE="crypto_LUKS"
root@s10:~# cat /etc/crypttab
# <target name>   <source device>         <key file>  <options>
encsdb UUID=50658916-4e42-467b-b2bd-21c919644feb /root/mykey_encsdb auto
```

1. Первое поле файла имя виртуальное устройство

2. Второе поле имя шифрованного устройство
3. Третье поле файл с паролем для разблокирования. Вместо имени файла можно указать none или «-». Еще один вариант указать имя файла устройства, которое генерирует случайные данные. Последний случай применяется для подключения шифрованных разделов swap или tmp. Файл с ключом может иметь любое содержание, но его надо добавить командой `cryptsetup luksAddKey`
4. Четвертое поле опции подключения.

Для автоматического монтирования при старте системы вносим строку в `/etc/fstab`:

```
root@s10:~# tail -1 /etc/fstab
/dev/mapper/encsdb /secret_data ext4 defaults 0 0
```

После перезагрузки можем проверить, что все подключилось:

```
root@s10:~# cryptsetup status encsdb
/dev/mapper/encsdb is active and is in use.
  type:    LUKS2
  cipher:  aes-cbc-essiv:sha256
  keysize: 256 bits
  key location: keyring
  device:  /dev/sdb
  sector size:  512
  offset:  32768 sectors
  size:    41910272 sectors
  mode:    read/write

root@s10:~# findmnt -t ext4
TARGET         SOURCE             FSTYPE OPTIONS
/              /dev/sda1          ext4   rw,relatime,errors=remount-ro
└─/secret_data /dev/mapper/encsdb ext4   rw,relatime
```

Выше приведенный пример позволит автоматически подключить шифрованный диск и файловую систему на нем. Это не очень безопасно, т. к. при получении физического доступа к компьютеру мы можем сменить пароль любому пользователю и получить доступ к защищаемым данным.

Более правильным решением будет или указать опцию `none` в файле `/etc/crypttab` или использовать модуль TPM для защиты данных.

Управлять ключами шифрованных дисков удобно утилитой `systemd-cryptenroll`.

```
root@s10:~# systemd-cryptenroll --password /dev/sdb
🔐 Please enter current passphrase for disk /dev/sdb: ******
🔐 Please enter new passphrase for disk /dev/sdb: ***
🔐 Please enter new passphrase for disk /dev/sdb (repeat): (press TAB for no
ech***
```

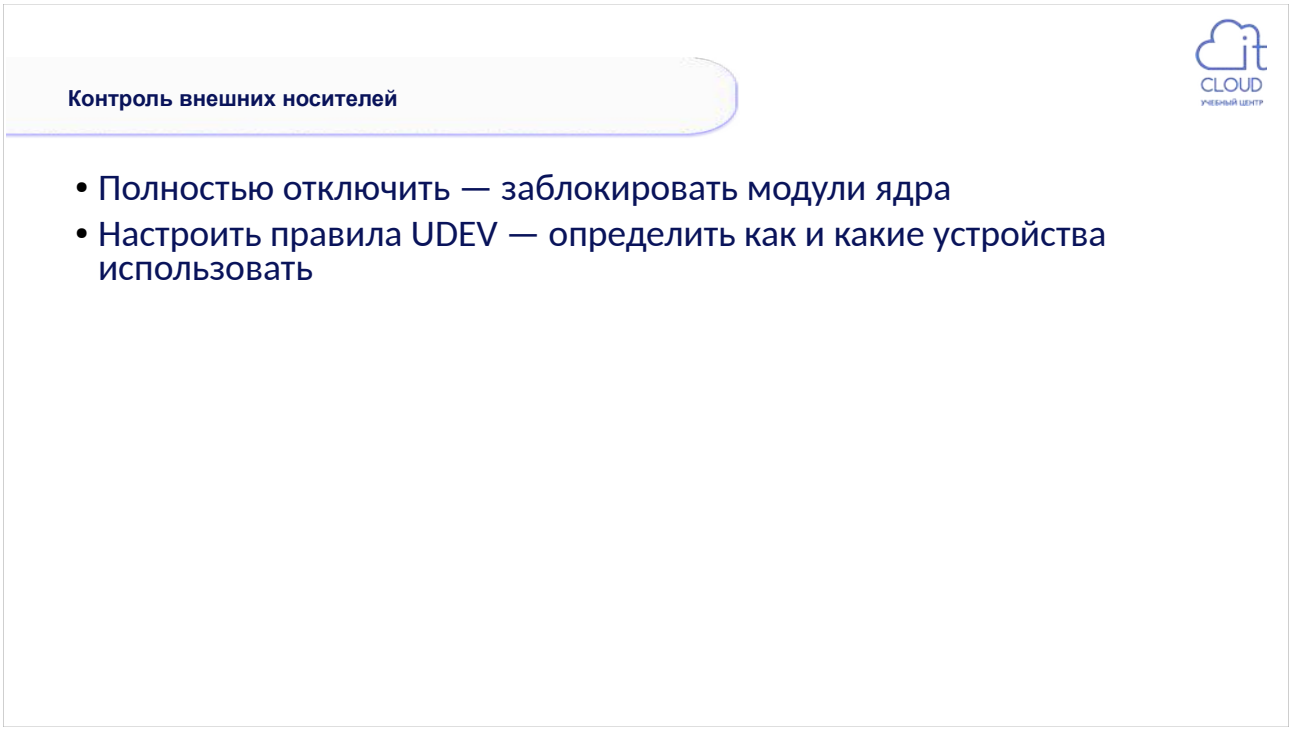
Посмотреть состояние заголовка LUKS в шифрованном диске можем командой:

```
root@s10:~# cryptsetup luksDump /dev/sdb
LUKS header information
Version:        2
Epoch:          5
Metadata area:  16384 [bytes]
Keyslots area:  16744448 [bytes]
UUID:           50658916-4e42-467b-b2bd-21c919644feb
```

```
Label:            (no label)
Subsystem:        (no subsystem)
Flags:            (no flags)

Data segments:
  0: crypt
      offset: 16777216 [bytes]
      length: (whole device)
      cipher: aes-cbc-essiv:sha256
      sector: 512 [bytes]

Keyslots:
  0: luks2
      Key:        256 bits
      Priority:   normal
      Cipher:     aes-cbc-essiv:sha256
      Cipher key: 256 bits
      PBKDF:      argon2id
      Time cost:  4
      Memory:     665548
      Threads:    2
      Salt:       6b 62 5f 35 7c ef 6c d7 47 24 e9 62 73 99 71 e4
                  c5 23 04 8d 96 01 cb 78 7d f0 1e 8e 6f c4 50 ab
      AF stripes: 4000
      AF hash:    sha256
      Area offset:32768 [bytes]
      Area length:131072 [bytes]
      Digest ID:  0
  1: luks2
      Key:        256 bits
      Priority:   normal
      Cipher:     aes-cbc-essiv:sha256
      Cipher key: 256 bits
      PBKDF:      argon2id
      Time cost:  4
      Memory:     749192
      Threads:    2
      Salt:       e2 d8 c5 47 76 63 1d 4d 77 ca 84 ee ed 99 61 7b
                  6b 00 8f 1a 1e c3 81 46 f1 d8 df 81 d6 72 57 20
      AF stripes: 4000
      AF hash:    sha256
      Area offset:163840 [bytes]
      Area length:131072 [bytes]
      Digest ID:  0
  2: luks2
      Key:        256 bits
      Priority:   normal
      Cipher:     aes-cbc-essiv:sha256
      Cipher key: 256 bits
      PBKDF:      argon2id
      Time cost:  4
      Memory:     867099
      Threads:    2
      Salt:       88 3f 7e 7b 3c e9 c2 78 b6 64 22 15 30 61 29 b3
                  cf fe 8f 81 ec 91 b9 19 08 31 7e 58 99 1a 37 51
      AF stripes: 4000
      AF hash:    sha256
      Area offset:294912 [bytes]
      Area length:131072 [bytes]
```

```
      Digest ID:  0
Tokens:
Digests:
  0: pbkdf2
      Hash:       sha256
      Iterations: 112604
      Salt:       8d 9d b4 7d 44 4b b6 6c ea 00 ab c4 7a 0e ea b3
                  e5 d0 f7 3c 0d 7d ee 47 df 12 f9 a9 1a cb 71 0b
      Digest:     8b 36 16 55 45 b2 63 34 0e b4 a9 b0 82 8f 38 11
                  d4 a7 93 ef 20 6e 4f 00 3e c0 5f f5 9e de 87 5d
```

2.10 Контроль внешних носителей.



Для контроля внешних носителей вы можете использовать несколько подходов.

1. Можно полностью отключить возможность подключать внешние накопители. Для отключения определенного типа носителей заблокируйте модуль ядра. Так для USB носителей вы можете запретить загрузку модуля ядра usb-storage, а для запрета подключения оптических носителей — sr_mod.

Пример: Запрет внешних носителей через блокировку модулей ядра.

Смотрим начальное состояние и настраиваем блокировку «ненужных» модулей.

```
root@client:~# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda      8:0    0   40G  0 disk
├─sda1   8:1    0   39G  0 part /
├─sda2   8:2    0    1K  0 part
└─sda5   8:5    0  975M  0 part [SWAP]
sdb      8:16   0   20G  0 disk
sr0     11:0    1 50,6M  0 rom

root@client:~# cat /etc/modprobe.d/blacklist.conf
blacklist uas
blacklist usb-storage
blacklist sr_mod

root@client:~# update-initramfs -k all -u
update-initramfs: Generating /boot/initrd.img-6.1.0-25-amd64
update-initramfs: Generating /boot/initrd.img-6.1.0-9-amd64
root@client:~# reboot
```

Результат настройки, после перезагрузки. Диска «нет», но в списке устройств USB присутствует.

```
root@client:~# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda      8:0    0   40G  0 disk
├─sda1   8:1    0   39G  0 part /
├─sda2   8:2    0    1K  0 part
└─sda5   8:5    0  975M  0 part [SWAP]

root@client:~# lsusb
Bus 001 Device 002: ID 80ee:0030 VirtualBox USB Harddisk
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 002: ID 80ee:0021 VirtualBox USB Tablet
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
root@client:~# lsusb -t
/:  Bus 02.Port 1: Dev 1, Class=root_hub, Driver=ohci-pci/12p, 12M
    |__ Port 1: Dev 2, If 0, Class=Human Interface Device, Driver=usbhid, 12M
/:  Bus 01.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/12p, 480M
    |__ Port 1: Dev 2, If 0, Class=Mass Storage, Driver=, 480M
```

Как альтернатива предложенному выше, можно задать параметр загрузки ядра module_blacklist:

```
root@client:~# grep module_blacklist /etc/default/grub
GRUB_CMDLINE_LINUX_DEFAULT="quiet module_blacklist=sr_mod,uas,usb-storage"

root@client:~# update-grub
```

2. Настроить правила UDEV, которые дадут возможность, только определенной группе пользователей работать с подключенными накопителями.

Пример:

1. Создайте группу пользователей usb-users и добавьте пользователя в эту группу.

```
root@client:~# groupadd usb-users
root@client:~# gpasswd -a sa usb-users
```

2. Создайте правило UDEV, которое назначит группу usb-users на любой подключенный USB диск.

```
root@client:~# cat /etc/udev/rules.d/99zz-usbdisks.rules
ACTION=="add", SUBSYSTEM=="block", ENV{ID_USB_DRIVER}=="usb-storage",
RUN+="/usr/bin/chgrp usb-users /dev/%k", ENV{UDISKS_AUTO}="0",
ENV{UDISKS_SYSTEM}="1"
```

Запускаемая команда позволит членам группы usb-users управлять диском, например создавать файловую систему. Назначение переменных UDISKS_AUTO и UDISKS_SYSTEM разрешит только администраторам подключать диск в графике.

---

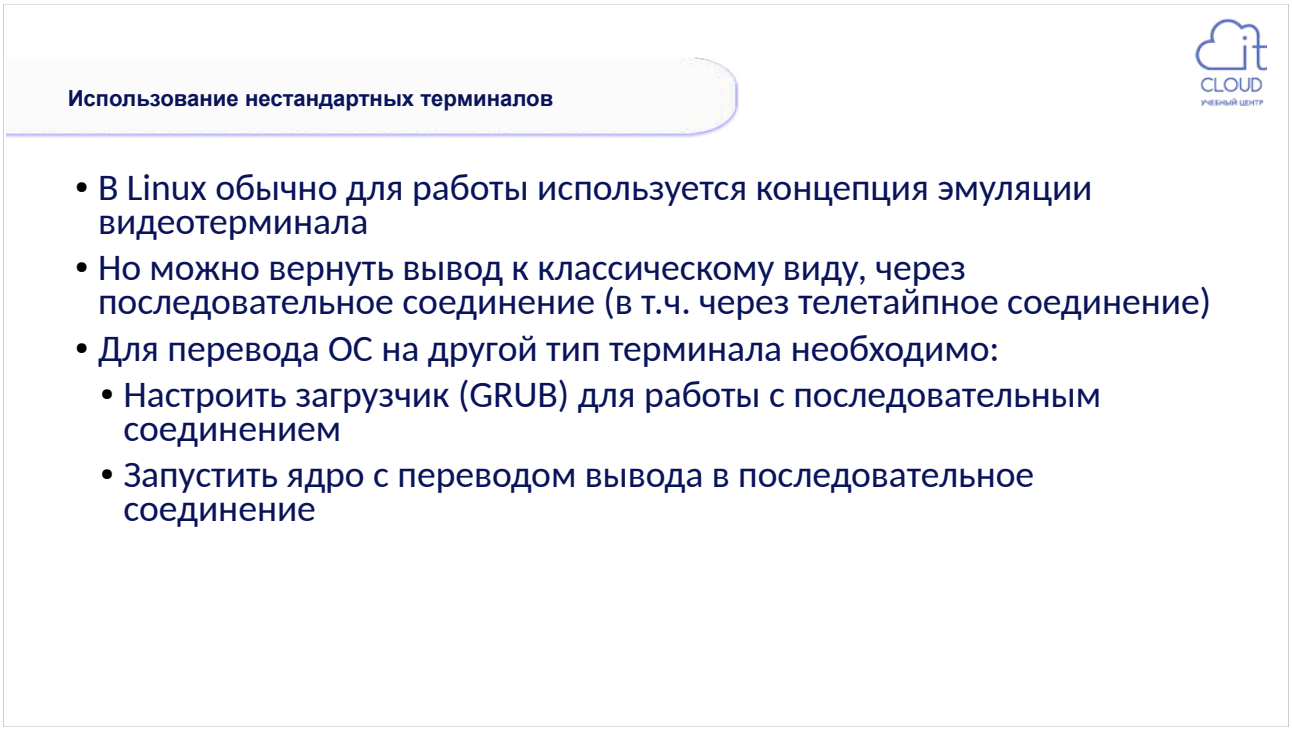
3. Перечитайте правила UDEV и проверьте как будет работать подключение USB носителя.

```
root@client:~# udevadm control --reload
root@client:~# udevadm test -a add /dev/sdb 2>/dev/stdout | grep usbdisk
Reading rules file: /etc/udev/rules.d/99zz-usbdisks.rules
sdb: /etc/udev/rules.d/99zz-usbdisks.rules:1 RUN '/usr/bin/chgrp usb-users
/dev/%k'
```

4. Отключите и вновь подключите USB диск.

```
root@client:~# ls -l /dev/sdb
brw-rw---- 1 root usb-users 8, 16 фев 12 09:16 /dev/sdb
```

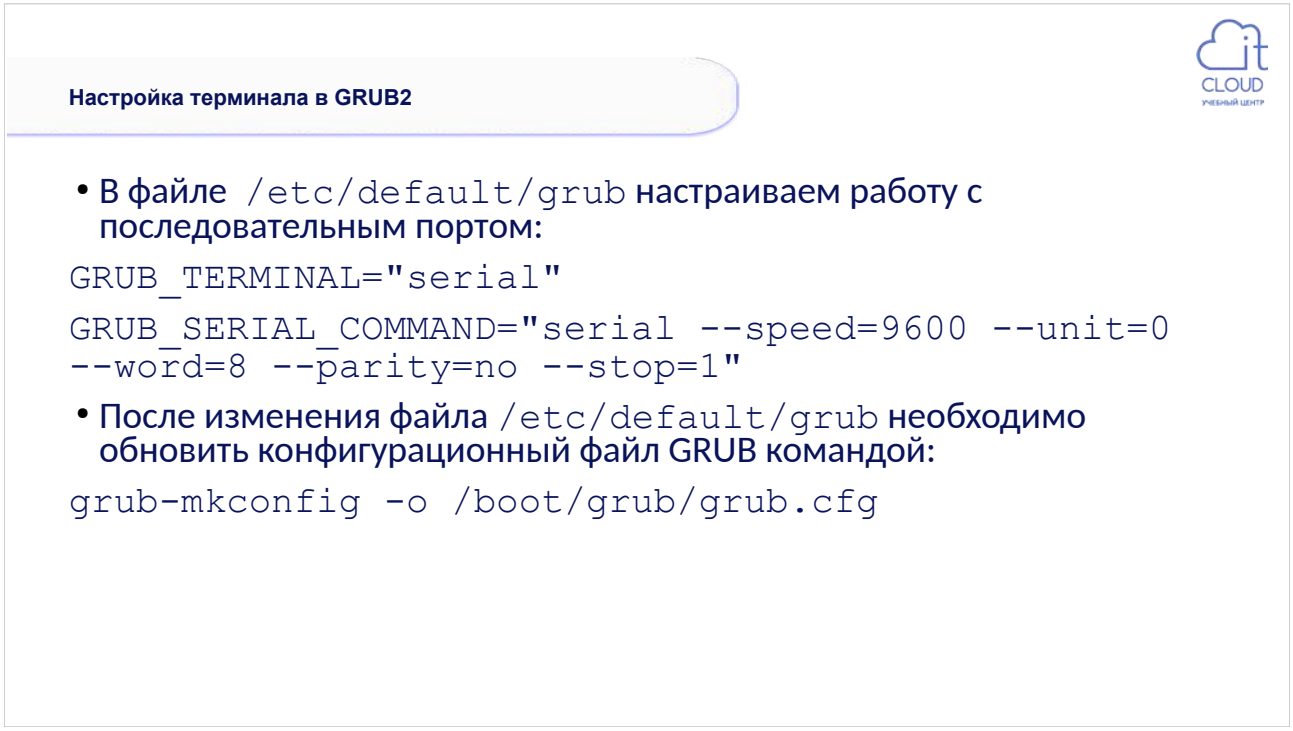
2.11 Использование нестандартных терминалов.



Изначально Unix подобные системы использовали последовательные соединения для работы в качестве терминальной линии. С появлением видео терминалов эти системы стали использовать эмуляторы классических терминалов (tty) с набором расширенных функций.

Параметр ядра `console=`*`терминал`* определяет тип терминала.

Чтобы перевести ОС на использование нестандартного терминала необходимо настроить загрузчик.

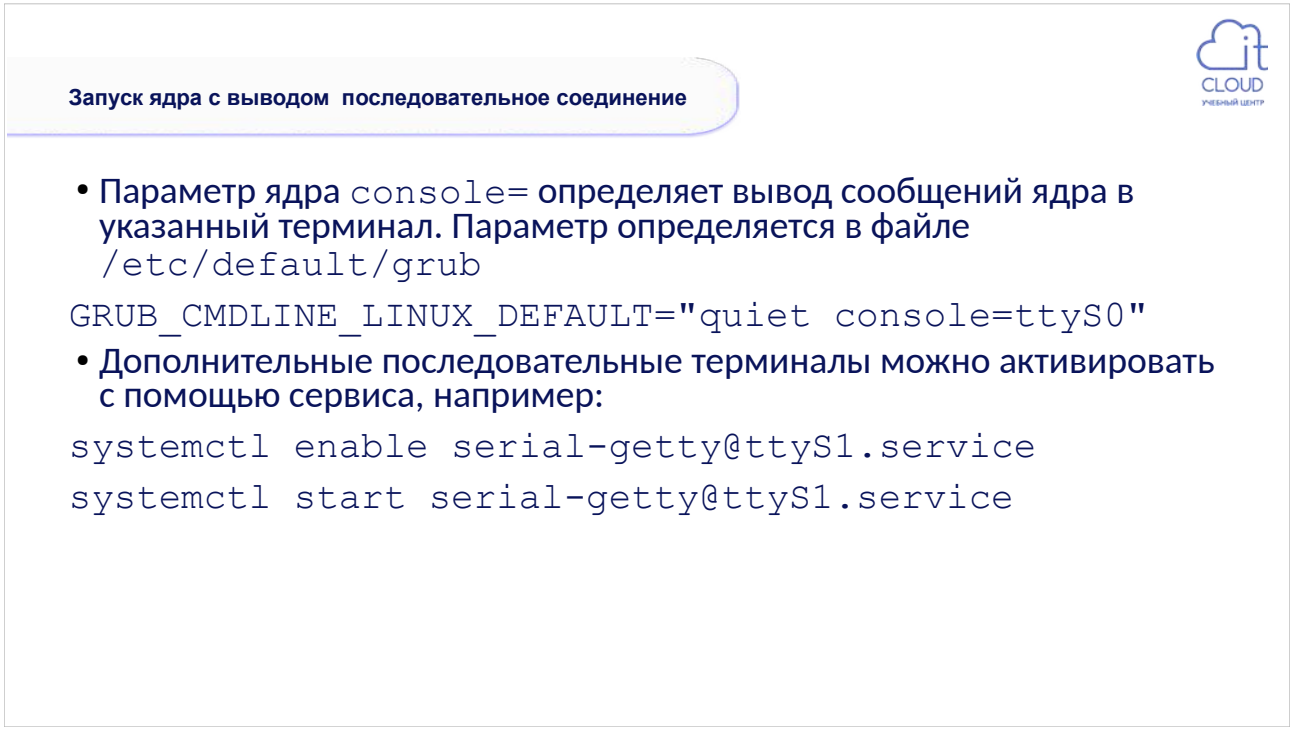


Файл `/etc/default/grub` используется для определения параметров файла конфигурации загрузчика.

Пример: Перевод вывода на первое последовательное соединение (COM1)

```
root@s10:~# grep -E 'serial|console' /etc/default/grub
GRUB_CMDLINE_LINUX_DEFAULT="quiet console=ttyS0"
GRUB_TERMINAL=serial
GRUB_SERIAL_COMMAND="serial --speed=9600 --unit=0 --word=8 --parity=no --stop=1"
```

После изменения файла `/etc/default/grub` следует выполнить команду `grub-mkconfig -o /boot/grub/grub.cfg`



Можно указывать несколько консолей, в которые нужно выводить данные о загрузке.

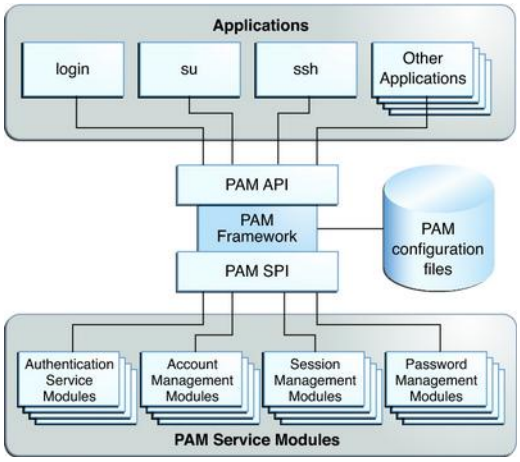
Если вам необходимо запустить дополнительные последовательные терминалы, то их можно активировать и запустить через службу `serial-getty@.service`.

Глава 3. Подключаемые модули аутентификации (PAM).

3.1 Система безопасности PAM

Система безопасности PAM

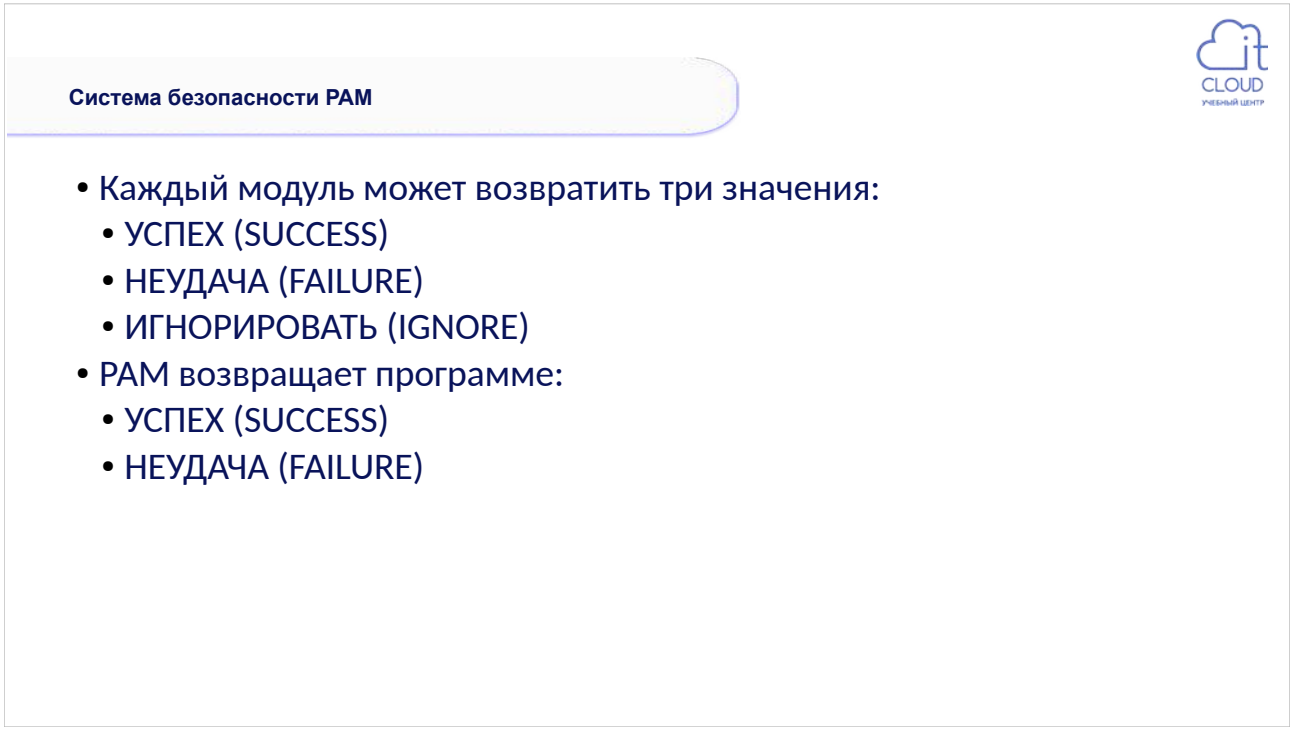
- Pluggable Authentication Modules
- Гибкая система аутентификации и авторизации
- Схема работы не зависит от применяемых механизмов
- Впервые была реализована в Sun Solaris 8.2.3
- Состоит из модулей и библиотек



https://docs.oracle.com/cd/E53394_01/html/E54787/pam-2.html

Одна из проблем при смены системы авторизации заключается в том, что все приложения так или иначе работающие с паролями, должны быть (заново) скомпилированы с поддержкой новой системы, иначе они будут неработоспособны. Однако компания Sun предложила решение этой проблемы, впервые появившееся в операционной системе Solaris 8.2.3, которое затем было адаптировано и для Linux.

Система называется PAM (Pluggable Authentication Modules) и состоит из модулей и библиотек. Приложению, собранному с использованием этих библиотек, не нужно подстраиваться под систему авторизации, используемую в системе в данный момент, так как PAM берет все эти вопросы на себя. Такой подход делает систему намного более гибкой.

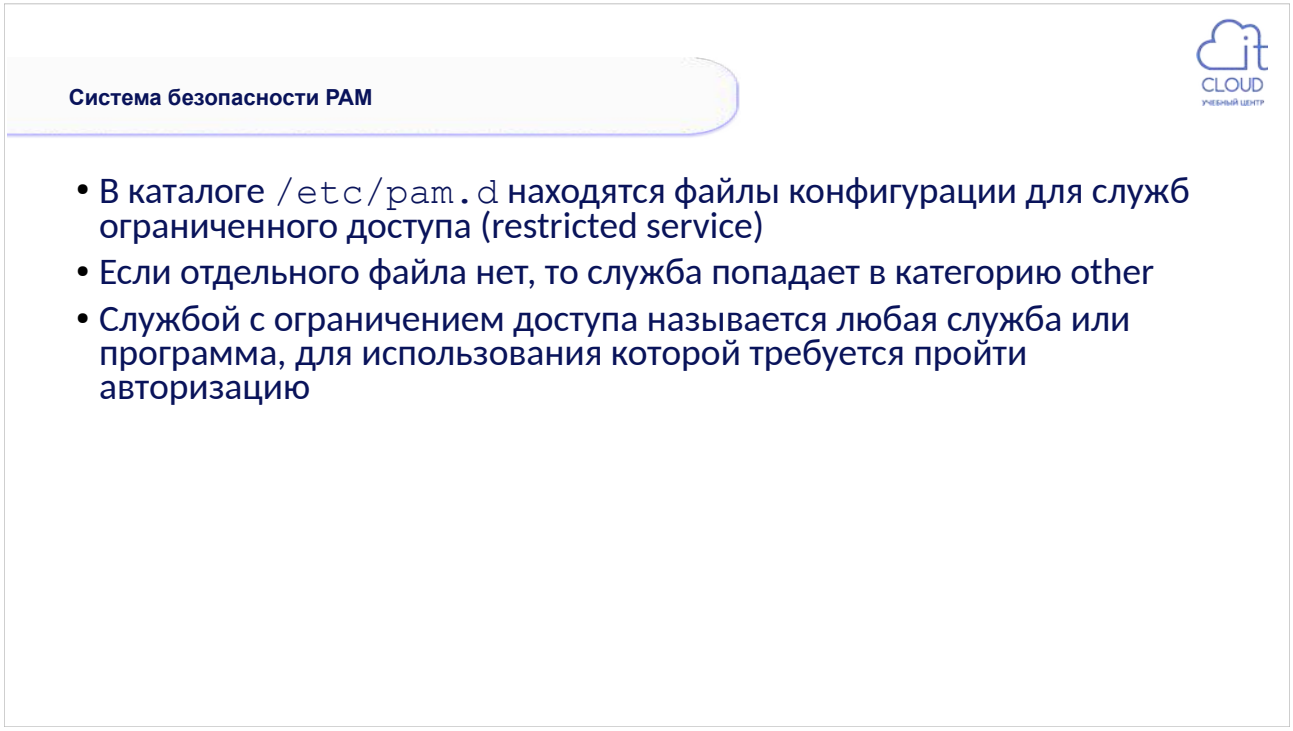


Основная идея PAM состоит в том, что всегда можно написать новый модуль безопасности, который бы обращался к файлу или устройству за информацией и возвращал результат выполнения процедуры авторизации: УСПЕХ (SUCCESS), НЕУДАЧА (FAILURE) или ИГНОРИРОВАТЬ (IGNORE).

PAM, в свою очередь, возвратит УСПЕХ (SUCCESS) или НЕУДАЧА (FAILURE) вызвавшей ее службе.

Таким образом, неважно, какие пароли, теневые или обычные, используются в вашей системе, коль скоро в ней есть PAM: все поддерживающие PAM программы будут прекрасно работать и с теми и другими. Более того, вы можете сравнительно легко добавлять и другие механизмы авторизации. Допустим, у вас есть устройство распознавания речи, способное сравнивать голоса с эталоном из файла. Тогда чтобы получить авторизацию на основе распознавания речи, достаточно написать модуль, который бы сравнивал голоса с помощью устройства и возвращал одно из трех значений: УСПЕХ (SUCCESS), ИГНОРИРОВАТЬ (FAILURE) или НЕУДАЧА (IGNORE) в зависимости от результата. Аналогично можно добавить авторизацию на основе сканирования сетчатки глаза, смарт-карты и т. д. дело лишь в соответствующем модуле.

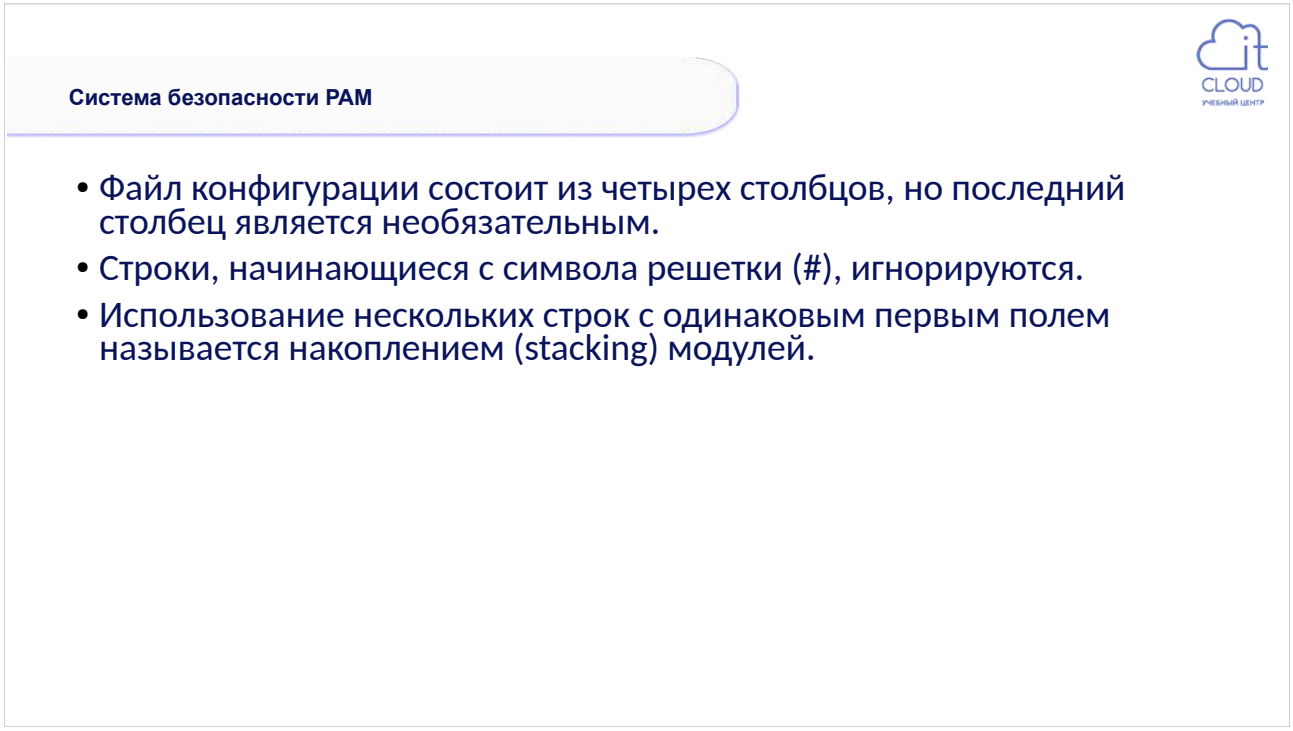
Модули можно комбинировать. Например так, чтобы для успешной авторизации требовалось прохождение нескольких процедур авторизации или же чтобы было достаточно прохождения любой из них. Тем самым можно создавать достаточно сложные системы авторизации: пароль, дополняемый сканированием сетчатки глаза, или же смарт-карта и распознавание голоса и т. д.



В каталоге /etc/pam.d содержатся файлы конфигурации для служб ограниченного доступа (restricted service).

Если такового нет, то данная служба с ограничением доступа попадает в категорию other, с файлом конфигурации other.

Службой с ограничением доступа называется любая служба или программа, для использования которой требуется пройти авторизацию. Иными словами, если при нормальных условиях служба запрашивает у вас имя пользователя и пароль, она является службой с ограничением доступа.

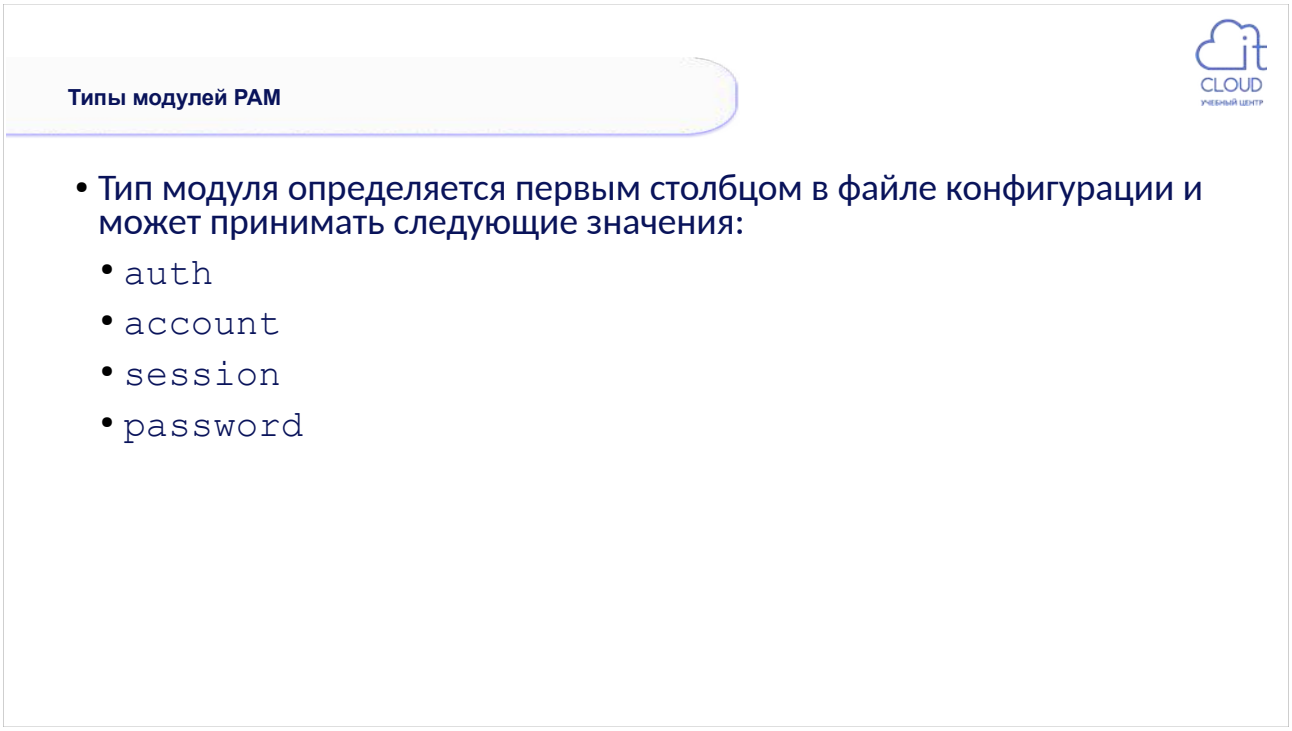


Файл конфигурации состоит из четырех столбцов, но последний столбец является необязательным.

Строки, начинающиеся с символа решетки (#), игнорируются.

Использование нескольких строк с одинаковым первым полем называется накоплением (stacking) модулей и позволяет получать много шаговую авторизацию (стек модулей), включающую несколько различных процедур авторизации.

3.2 Типы модулей PAM



Первый столбец в файле конфигурации PAM является столбцом типа. Тип определяется одной из четырех символьных меток: `auth, account, session и password.`

Содержимое всех столбцов рассматривается без учета регистра.

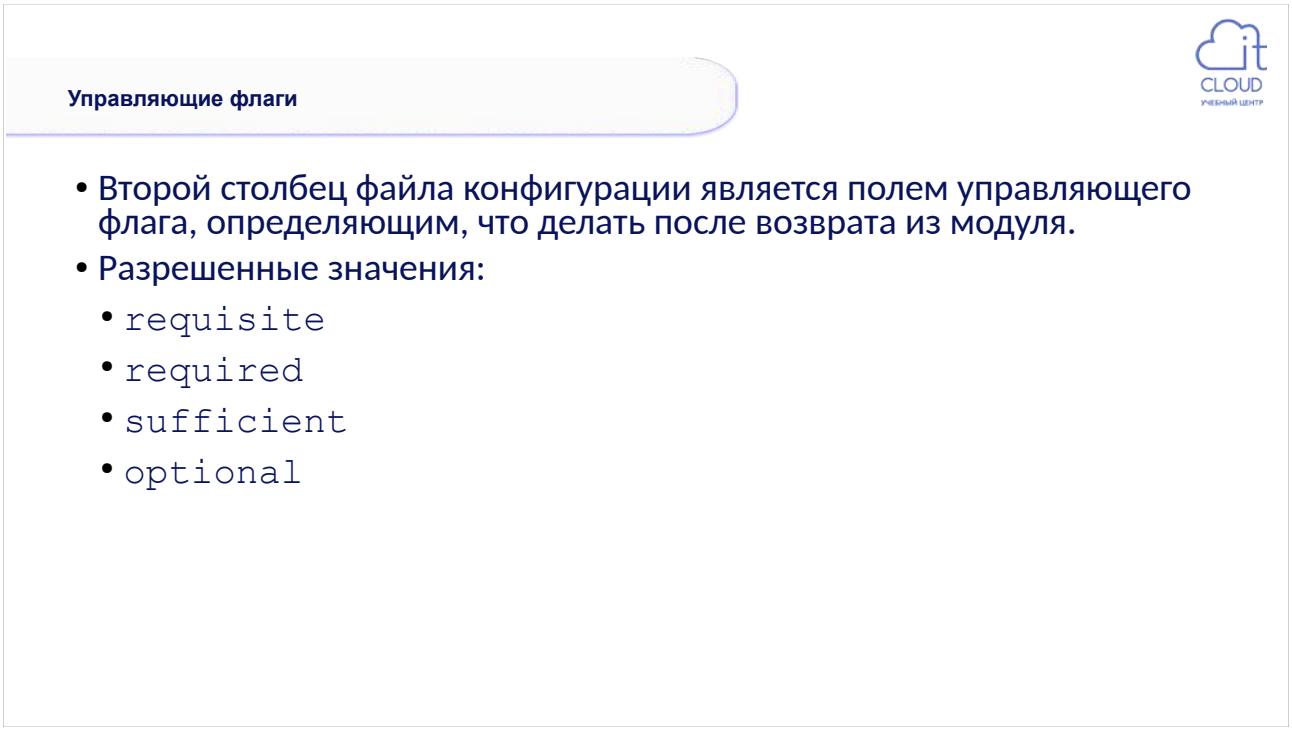
Тип **auth** (authentication — аутентификация) используется для выяснения, является ли пользователь тем, за кого он себя выдает. Как правило, это достигается сравнением введенного и хранимого паролей, но возможны и другие варианты. Например, при помощи смарт-карты или каким-либо иным способом, для которого имеется соответствующий модуль.

Тип **account** (учетная запись) проверяет дозволено ли использовать службу данному пользователю, на каких условиях, не устарел ли пароль и т. д.

Тип **password** (пароль) используется для обновления маркеров авторизации. Например управляет процессом смены пароля, проверяя его на соответствие требованиям сложности.

Тип **session** (сеанс) выполняет определенные действия при входе пользователя в систему и при выходе пользователя из системы. Таковыми могут быть, например, монтирование и демонтирование каталогов, подготовка пользовательского окружения и т. д.

3.3 Управляющие флаги



Второй столбец файла конфигурации является полем управляющего флага, определяющим, что делать после возврата из модуля, то есть реакцию PAM на значения УСПЕХ (SUCCESS), ИГНОРИРОВАТЬ (IGNORE) и НЕУДАЧА (FAILURE).

Разрешенные значения: requisite, required, sufficient и optional.

От значения в этом поле зависит, будут ли обработаны остальные строки файла. Например, если модуль, помеченный как optional (достаточный), вернет значение УСПЕХ (SUCCESS), то работа PAM на нем успешно закончится и пользователь будет допущен в систему. Аналогичным образом возврат НЕУДАЧА (FAILURE) модулем, помеченным как requisite (обязательный), означает прекращение авторизации и возврат значения НЕУДАЧА (FAILURE). И в том и в другом случае остальные строки обрабатываться не будут.

- **requisite** (**обязательный**) задает наиболее жесткое поведение. Обработка любой строки с флагом requisite, модуль которой вернул значение НЕУДАЧА (FAILURE), будет прекращена и вызвавшей ее службе будет возвращен статус НЕУДАЧА (FAILURE). Никакие другие строки рассматриваться не будут.

***Примечание**: Вообще говоря, этот флаг используется достаточно редко. Дело в том, что если помеченный им модуль выполняется самым первым, то следующие за ним модули могут и не выполниться, в том числе и отвечающие за протоколирование, поэтому вместо него обычно применяется флаг required (необходимый).*

- **required** (**необходимый**) не прерывает выполнение модулей. Каков бы ни был результат выполнения помеченного им модуля: УСПЕХ (SUCCESS), ИГНОРИРОВАТЬ (IGNORE) или НЕУДАЧА (FAILURE), PAM всегда переходит к обработке следующего модуля. Это наиболее часто используемый флаг, так как

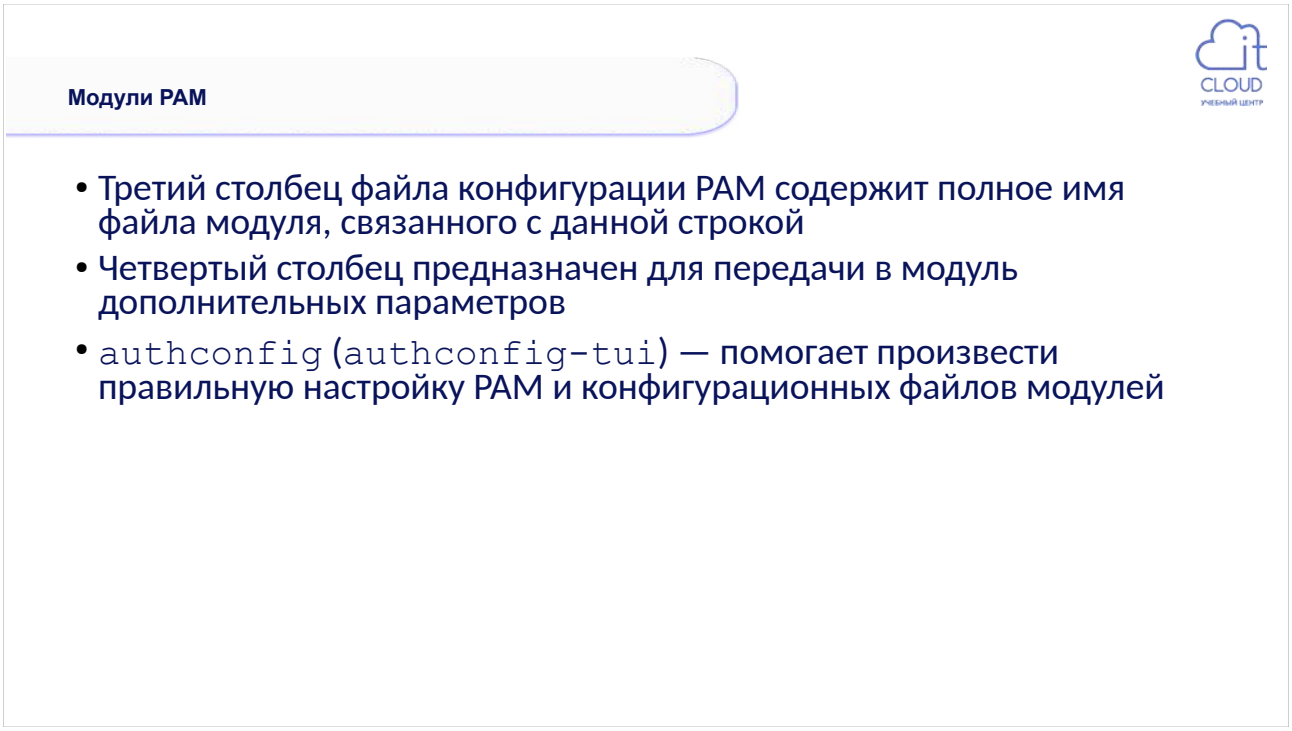
результат выполнения модуля не возвращается до тех пор, пока не отработают все остальные модули, а значит, модули, отвечающие за протоколирование, обязательно выполнятся.

- **sufficient** (**достаточный**) приводит к немедленному завершению обработки строки и возврату значения УСПЕХ (SUCCESS) при условии, что помеченный им модуль вернул значение УСПЕХ (SUCCESS) и ранее не встречалось модуля с флагом required, вернувшего статус НЕУДАЧА (FAILURE). Если такой модуль встречался, то флаг sufficient игнорируется. Если помеченный этим флагом модуль возвратил значение ИГНОРИРОВАТЬ (IGNORE) или НЕУДАЧА (FAILURE), то флаг sufficient рассматривается аналогично флагу optional.
- **optional** (**необязательный**) принимается во внимание лишь тогда, когда он является единственным модулем в стеке, вернувшим значение УСПЕХ (SUCCESS). В противном случае результат его выполнения игнорируется. Таким образом, не успешное выполнение помеченного им модуля не влечет за собой неуспех всего процесса авторизации.

Чтобы пользователь смог получить доступ к системе, модули, помеченные флагами requisite и required, не должны возвращать значения НЕУДАЧА (FAILURE).

Результат выполнения модуля с флагом optional принимается в рассмотрение, лишь если он является единственным модулем в стеке, вернувшим УСПЕХ (SUCCESS).

3.4 Модули PAM



Третий столбец файла конфигурации PAM содержит полное имя файла модуля, связанного с данной строкой.

Модули могут располагаться где угодно, однако если они размещены в предопределенном каталоге для модулей, то можно указывать одно лишь имя, в противном случае нужен еще и путь.

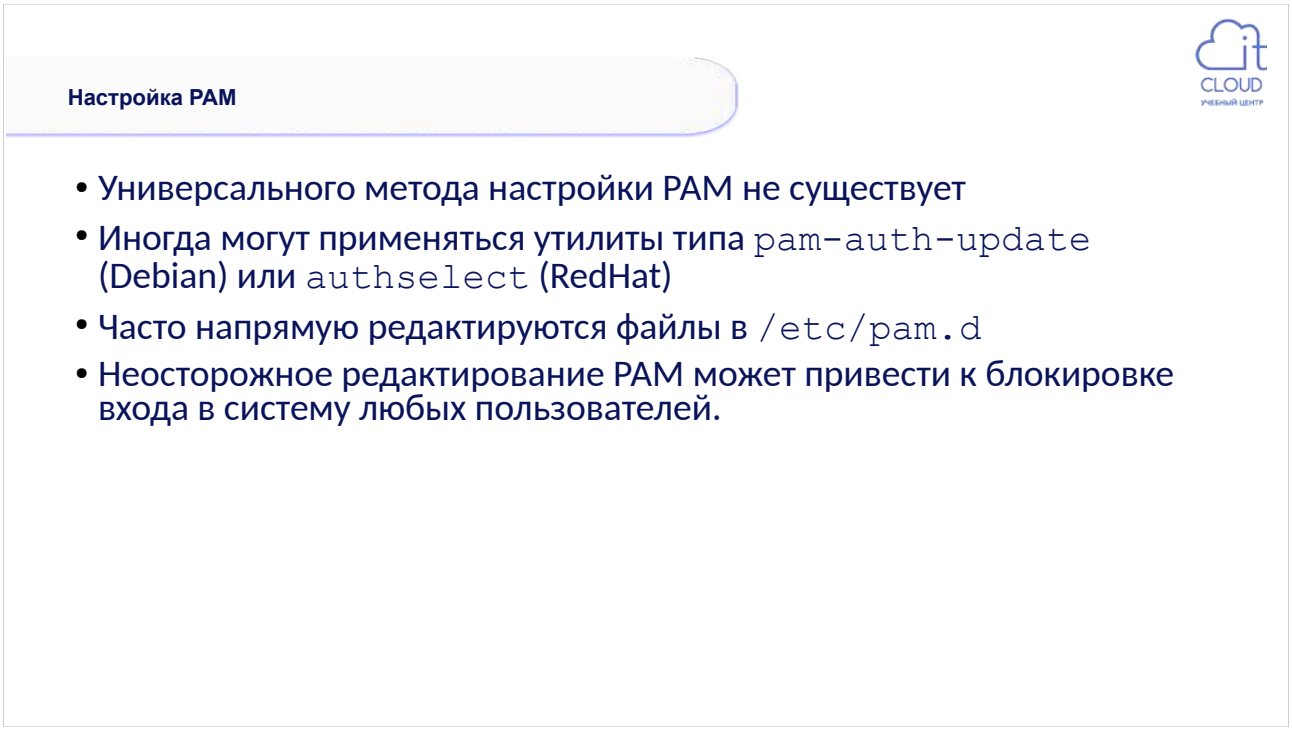
Каталог содержащий модули PAM это обычно `/lib/security` или `/lib64/security`.

Четвертый столбец предназначен для передачи в модуль дополнительных параметров.

Не у всех модулей есть параметры, а если есть, то они могут и не использоваться.

Передача параметра модулю позволяет изменить его поведение тем или иным образом.

3.5 Настройка PAM.



Устоявшегося метода настройки модулей PAM не существуют.

Часто изменение конфигурационных файлов в каталоге `/etc/pam.d` недостаточно. Необходимо менять также конфигурационные файлы модулей. Что делает процесс настройки аутентификации сложным и запутанным. Например, при настройке аутентификации в LDAP нужно менять файлы `/etc/ldap/ldap.conf` и `/etc/nsswitch.conf`.

В RedHat подобных Linux имеется утилита для решения этих проблем: `authselect`.

Эта программа позволяет настроить получение информации о пользователях и аутентификацию в стандартных службах:

1. В системе теневых паролей
2. В службе Kerberos
3. В службе LDAP
4. В службе SAMBA

В Debian подобных утилита — `pam-auth-update`. Но она только включает или выключает модули, их настройку надо производить отдельно.

Важно, при работе с PAM, помнить, что неправильная конфигурация PAM приводит к блокировке входа всех пользователей. Если вы редактируете конфигурацию PAM, то не закрывая сеанса суперпользователя попробуйте войти в систему.

Пример: настроим проверку качества пароля

1. Установим пакет libpam-pwquality

```
root@s10:~# apt install libpam-pwquality
```

2. Проверим настройки в конфигурации PAM:

```
root@s10:~# grep -r pam_pwquality /etc/pam.d/
/etc/pam.d/common-password:password requisite   pam_pwquality.so retry=3
```

Мы видим, что в файл /etc/pam.d/common-password была добавлена строка вызова модуля pam_pwquality.so

3. Отключим модуль с помощью pam-auth-update и проверим конфигурацию:

```
root@s10:~# pam-auth-update
root@s10:~# grep -r pam_pwquality /etc/pam.d/
```

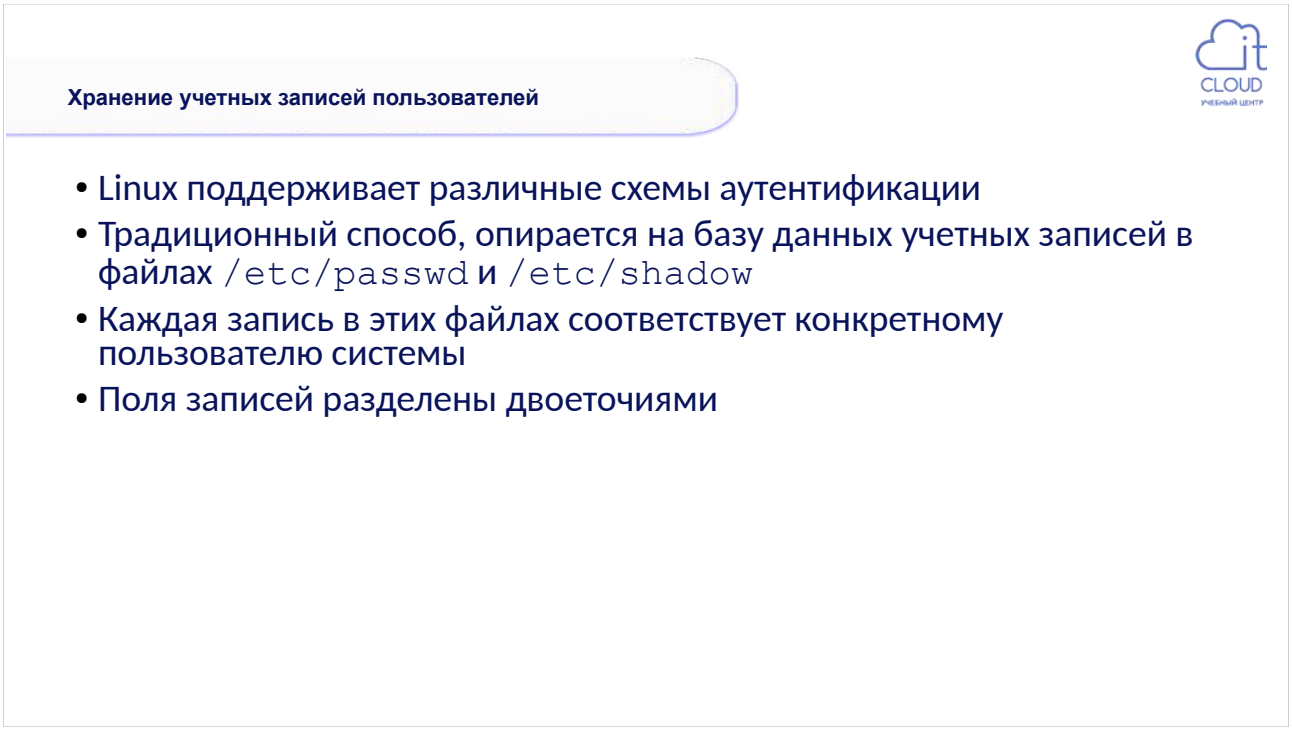
Вторая команда ничего не выводит.

4. Вновь включим модуль командой `pam-auth-update`.

Обратите внимание на параметр модуля `retry=3`, в `man pam_pwquality` мы можем прочитать про другие опции, а так же узнать, что имеется еще и конфигурационный файл `/etc/security/pwquality.conf`, в котором можно описать политику в описании сложностей паролей, вместо того, чтобы усложнять конфигурацию модулей PAM.

Глава 4. Аутентификация.

4.1 Хранение учетных записей пользователей.



В GNU/Linux процедура аутентификации пользователя при входе в сеанс может быть проведена разными способами. Вот некоторые из них:

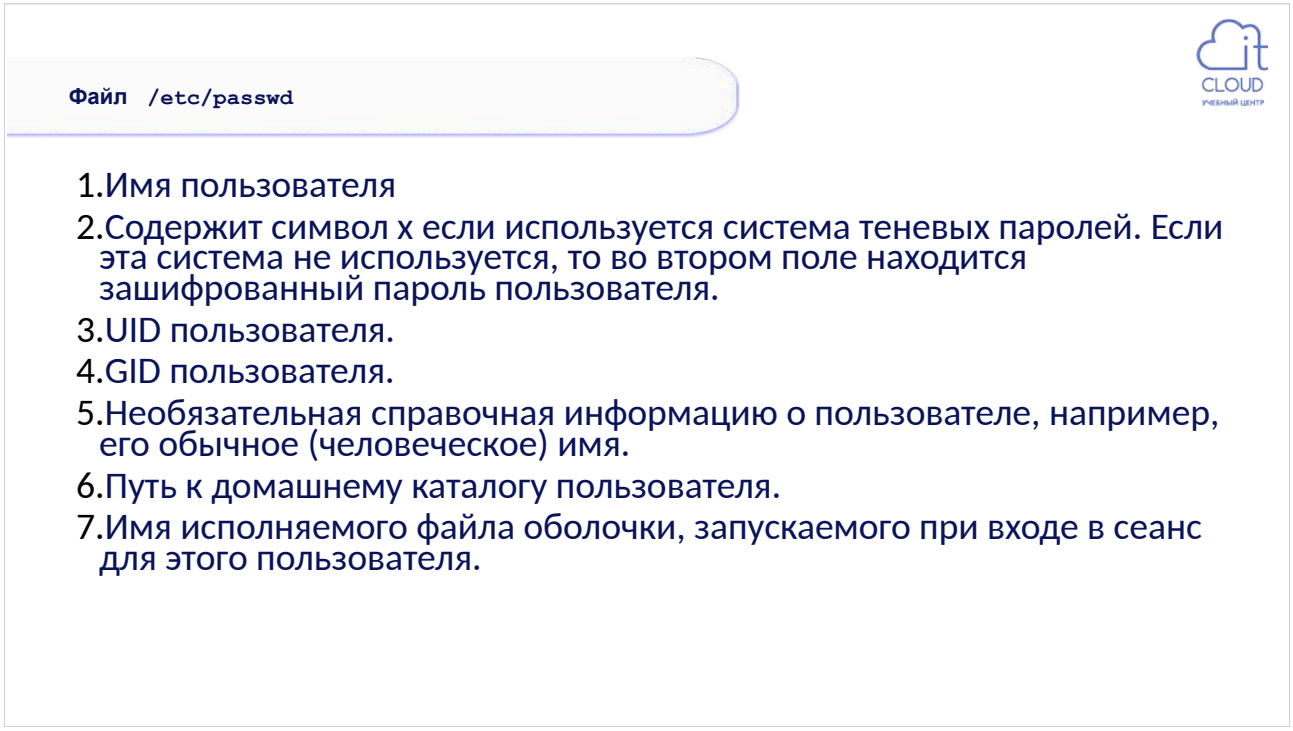
- Традиционный способ, опирающийся на база данных учетных записей в файлах `/etc/passwd` и `/etc/shadow`.
- Аутентификация с помощью системы Kerberos.
- Аутентификация в NIS/NIS+.
- Аутентификация в LDAP.
- Использование специализированных систем аутентификации (например, TCB - trusted computing base) и т.п.

Не смотря на наличие различных систем аутентификации наиболее простым, а следовательно, и наиболее распространенным способом является традиционный, использующий текстовые файлы учетных записей пользователей.

Файл `/etc/passwd` для аутентификации пользователей используется еще с первых версий Unix, а файл `/etc/shadow` стал использоваться с появлением системы теневых паролей.

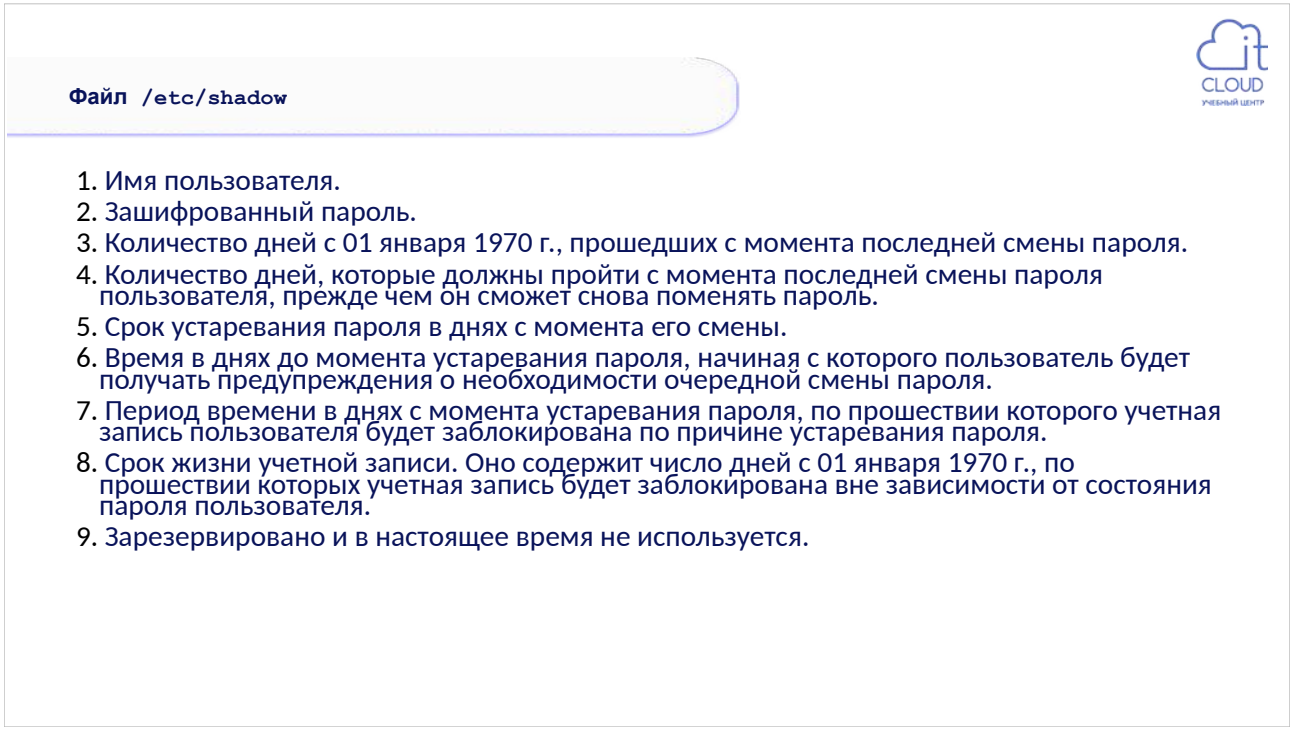
Каждая запись в этих файлах соответствует конкретному пользователю системы.

Поля записей разделены двоеточиями.



Структура записей в файле /etc/passwd следующая:

- Первое поле – имя пользователя.
- Второе поле содержит символ x если используется система теневых паролей /etc/shadow (в man 5 passwd указано, что при использовании теневых паролей во втором поле должен находиться символ *). Если эта система не используется, то во втором поле находится зашифрованный пароль пользователя.
- Третье поле – UID пользователя.
- Четвертое поле – GID пользователя.
- Пятое поле содержит необязательную справочную информацию о пользователе, например, его обычное (человеческое) имя.
- Шестое поле указывает домашний каталог пользователя.
- Седьмое поле соответствует имени исполняемого файла оболочки, запускаемой при входе в сеанс для этого пользователя.



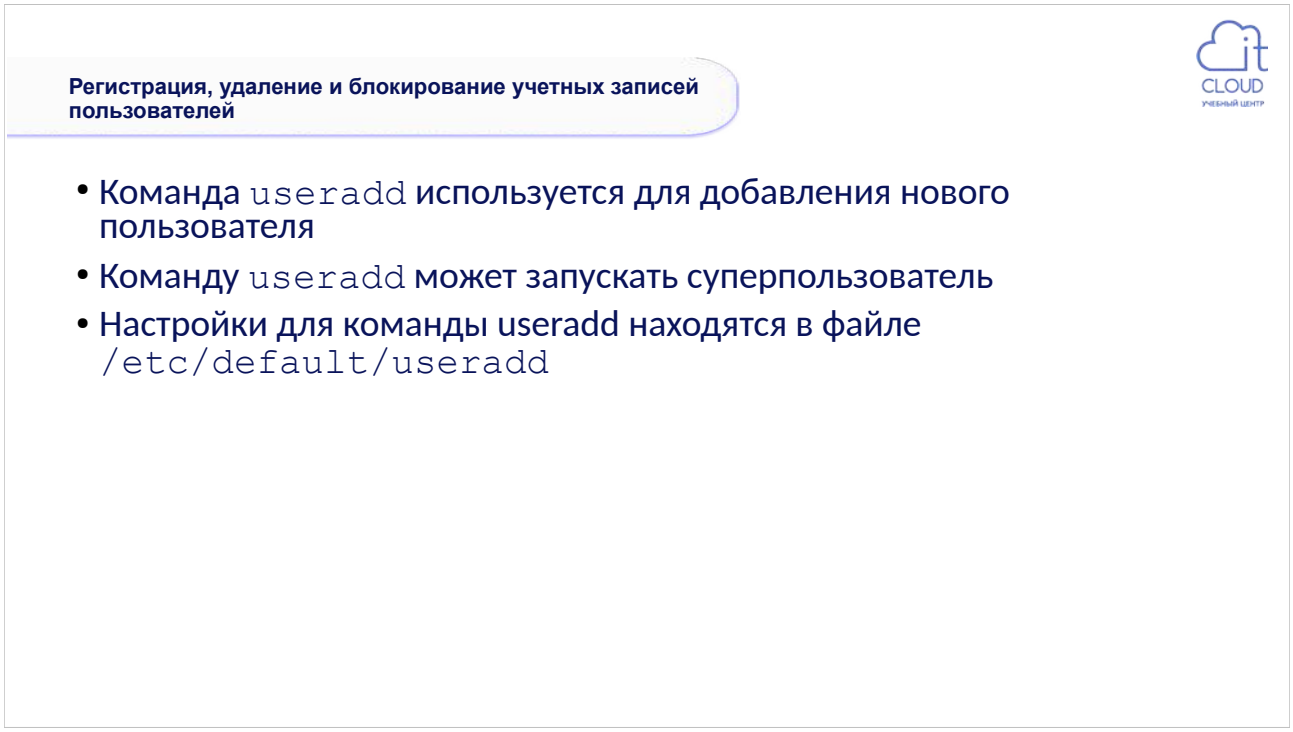
Права доступа, устанавливаемые на файл /etc/passwd позволяют читать этот файл всем пользователям. Поэтому при хранении шифрованного пароля во втором поле этого файла представляет реальную угрозу безопасности, так как любой злоумышленник, имеющий доступ к данной системе может воспользоваться программами подбора паролей для взлома системы.

Использование системы теневых паролей существенно снижает эту опасность, так как файл, где хранятся шифрованные пароли — /etc/shadow не позволяет его читать никому, кроме суперпользователя. Структура этого файла такова:

- Первое поле – имя пользователя.
- Второе поле – зашифрованный пароль.
- Третье поле – количество дней с 01 января 1970 г., прошедших с момента последней смены пароля.
- Четвертое поле – количество дней, которые должны пройти с момента последней смены пароля пользователя, прежде чем он сможет снова поменять пароль.
- Пятое поле – срок устаревания пароля в днях с момента его смены. До истечения этого срока пароль обязательно должен быть изменен.
- Шестое поле – время в днях до момента устаревания пароля, начиная с которого пользователь будет получать предупреждения о необходимости очередной смены пароля.
- Седьмое поле – период времени в днях с момента устаревания пароля, по прошествии которого учетная запись пользователя будет заблокирована по причине устаревания пароля.

- Восьмое поле устанавливает срок жизни учетной записи и предназначен для создания временных учетных записей. Оно содержит число дней с 01 января 1970 г., по прошествии которых учетная запись будет заблокирована вне зависимости от состояния пароля пользователя.
- Девятое поле зарезервировано и в настоящее время не используется.

4.2 Регистрация, удаление и блокирование учетных записей пользователей.



Правами регистрации пользователей в системе обладает суперпользователь.

Для добавления учетной записи пользователя используется команда useradd. В качестве аргмента для этой команды должно быть указано имя пользователя.

Пример:

```
root@s10:~# useradd user1
root@s10:~# id user1
uid=1001(user1) gid=1001(user1) groups=1001(user1)
```

Примечание: *В этом примере в системе был зарегистрирован новый пользователь – user1. Для него была зарегистрирована его приватная группа пользователей – user1. Приватная группа пользователей состоит из единственного пользователя. Она является первичной группой для этого пользователя.*

3. Регистрация пользователя приводит к появлению соответствующих записей в файлах `/etc/passwd` и `/etc/shadow`.

Пример:

```
root@s10:~# grep user1 /etc/passwd
user1:x:1001:1001::/home/user1:/bin/bash

root@s10:~# grep user1 /etc/shadow
user1:!!:17374:0:99999:7:::
```

Также для этого пользователя не был создан его домашний каталог:

```
root@s10:~# ls -a /home/
```

. .. sa

Создание приватной группы и домашнего каталога для пользователя характерно для Red Hat Linux и подобных дистрибутивов. В других дистрибутивах создание домашнего каталога производится только при использовании опции `-m` команды `useradd`, либо при использовании опции `-d`, указывающей путь к домашнему каталогу.

Создание приватной группы можно запретить, используя опцию `-n`. При этом для вновь зарегистрированных пользователей в системе будет установлена группа по умолчанию (см. ниже) в качестве первичной группы.

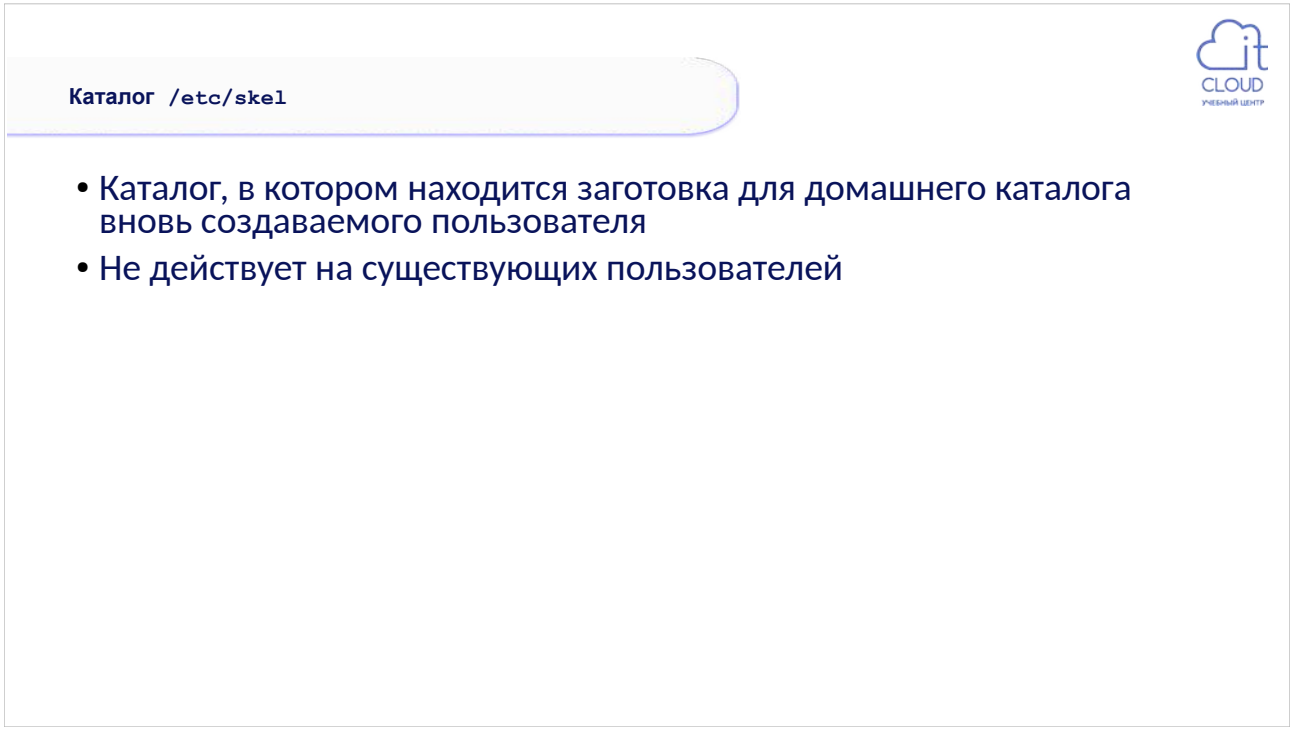
Настройки для команды `useradd` находятся в файле `/etc/default/useradd` и могут быть получены с помощью опции `-D` команды `useradd` .

Пример:

```
root@s10:~# useradd -D
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/sh
SKEL=/etc/skel
CREATE_MAIL_SPOOL=no
LOG_INIT=yes
```

Выведенная информация командой `useradd -D` говорит о следующем:

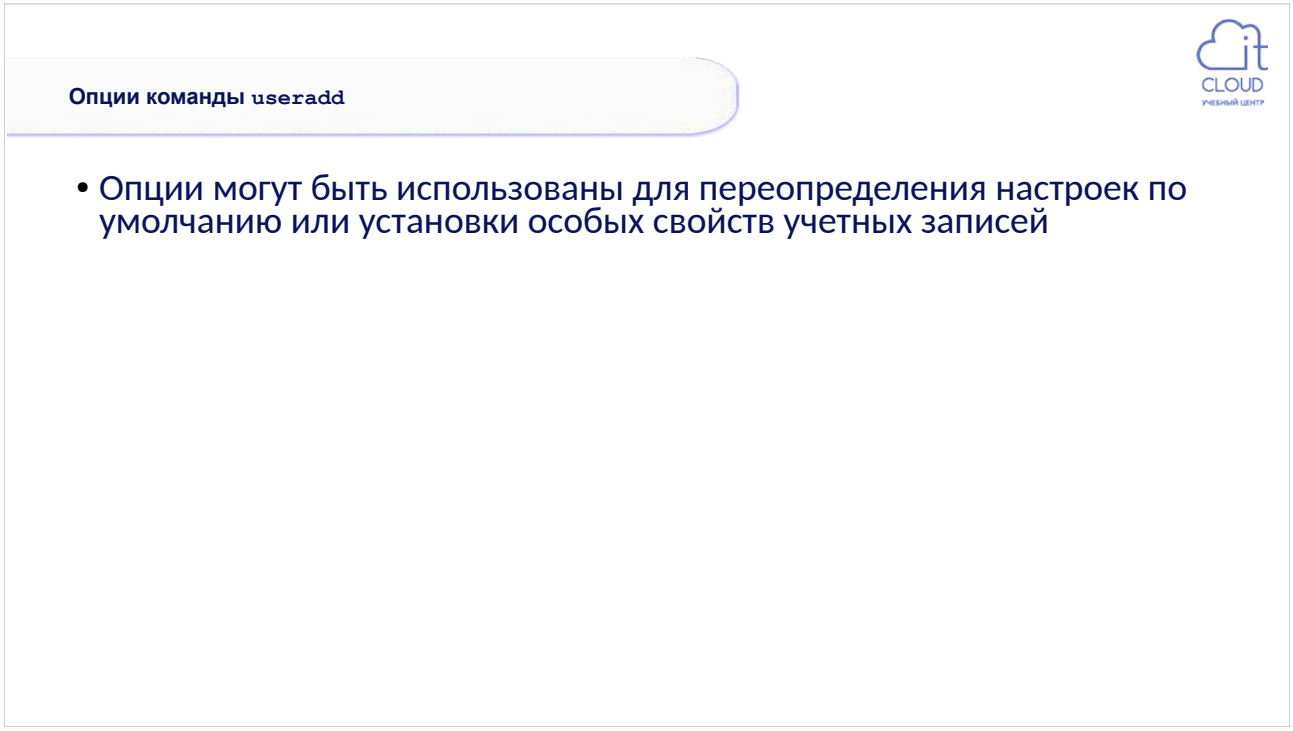
- `GROUP=100` - GID для вновь регистрируемых пользователей - 100 (эта настройка игнорируется при создании приватной группы пользователя);
- `HOME=/home` - домашние каталоги для пользователей будут создаваться в каталоге /home;
- `INACTIVE =-1` – блокирование учетной записи пользователя при устаревании его пароля не будет;
- `SHELL=/bin/sh` – оболочка для вновь регистрируемых пользователей по умолчанию;
- `SKEL=/etc/skel` – каталог “скелета”, из которого в домашние каталоги вновь регистрируемых пользователей копируются файлы, необходимые для каждого пользователя.
- `CREATE_MAIL_SPOOL=no` не требуется создавать почтовый ящик в каталоге `/var/spool/mail`



Каталог `/etc/skel` обычно содержит файлы профиля для вновь регистрируемых пользователей и другие служебные файлы, которые копируются при регистрации пользователя в его домашний каталог.

Пример:

```
root@s10:~# ls -a /etc/skel/
.  ..  .bash_logout  .bashrc  .face  .face.icon  .profile
```



Наиболее часто используемые опции команды useradd :

-s – указывает исполняемый файл оболочки по умолчанию;

-d – путь к домашнему каталогу пользователя;

-m – опция, указывающая на необходимость создать домашний каталог;

-M – не создавать домашний каталог;

-k – путь к альтернативному каталогу скелета;

-u – назначить UID пользователю;

-g – назначить GID (первичную группу) пользователю;

-G – список групп, в которых принимает участие пользователь (разделены запятыми);

-e – календарная дата, после которой учетная запись будет заблокирована (срок жизни учетной записи);

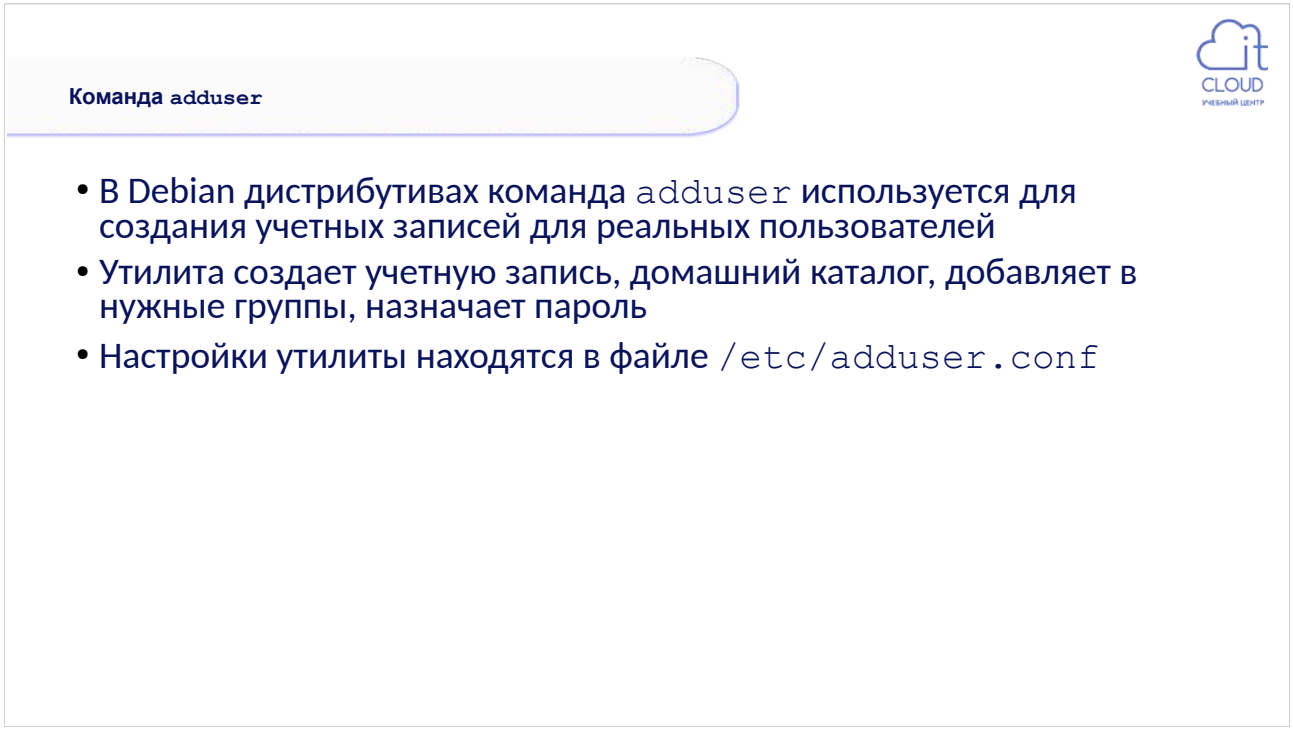
-f – количество дней, которое должно пройти после срока устаревания пароля до блокировки учетной записи.

Пример:

```
root@s10:~# useradd -M -G users -s /sbin/nologin -e 2025-05-01 tempuser
root@s10:~# id tempuser
uid=1002(tempuser) gid=1002(tempuser) группы=1002(tempuser),100(users)
root@s10:~# ls -a /home/
.  ..  sa
root@s10:~# ls -a /var/spool/mail
.  ..
root@s10:~# getent passwd tempuser
tempuser:x:1002:1002::/home/tempuser:/sbin/nologin
root@s10:~# getent shadow tempuser
```

```
tempuser:!:20122:0:99999:7::20209:
root@s10:~# echo $((`date -d "02 May 2025" +%s`/24/3600))
20209
```

Примечание: *Приведенная выше команда регистрирует пользователя без создания для него домашнего каталога (опция -M) с добавлением в группу users (опция -G). Для этого пользователя запрещен вход в сеанс, так как в качестве оболочки для этого пользователя указан файл /sbin/nologin . Учетная запись пользователя будет заблокирована после 1 мая 2025 г. (опция -e). Предпоследняя команда примера демонстрирует запись в файле теневых паролей для пользователя tempuser. Восьмое поле записи содержит число дней с 1 января 1970 г. до дня, когда учетная запись пользователя будет заблокирована. Последняя команда — проверка номера дня.*



В Debian подобных дистрибутивах предполагается, что утилита `useradd` используется для добавления системных пользователей. Если вы хотите создать учетную запись для человека, то удобней воспользоваться командой `adduser`.

Пример: создание пользователя командой `adduser`

```
root@s10:~# adduser user2
Добавляется пользователь «user2» ...
Добавляется новая группа «user2» (1003) ...
Adding new user `user2' (1003) with group `user2 (1003)' ...
Создаётся домашний каталог «/home/user2» ...
Копирование файлов из «/etc/skel» ...
Новый пароль:
НЕУДАЧНЫЙ ПАРОЛЬ: Пароль должен содержать не менее 8 символов
Повторите ввод нового пароля:
passwd: пароль успешно обновлён
Изменение информации о пользователе user2
Введите новое значение или нажмите ENTER для выбора значения по умолчанию
      Полное имя []:
      Номер комнаты []:
      Рабочий телефон []:
      Домашний телефон []:
      Другое []:
Данная информация корректна? [Y/n]
Adding new user `user2' to supplemental / extra groups `users' ...
Добавляется пользователь «user2» в группу «users» ...
```

Далее проверим, что произошло в системе:

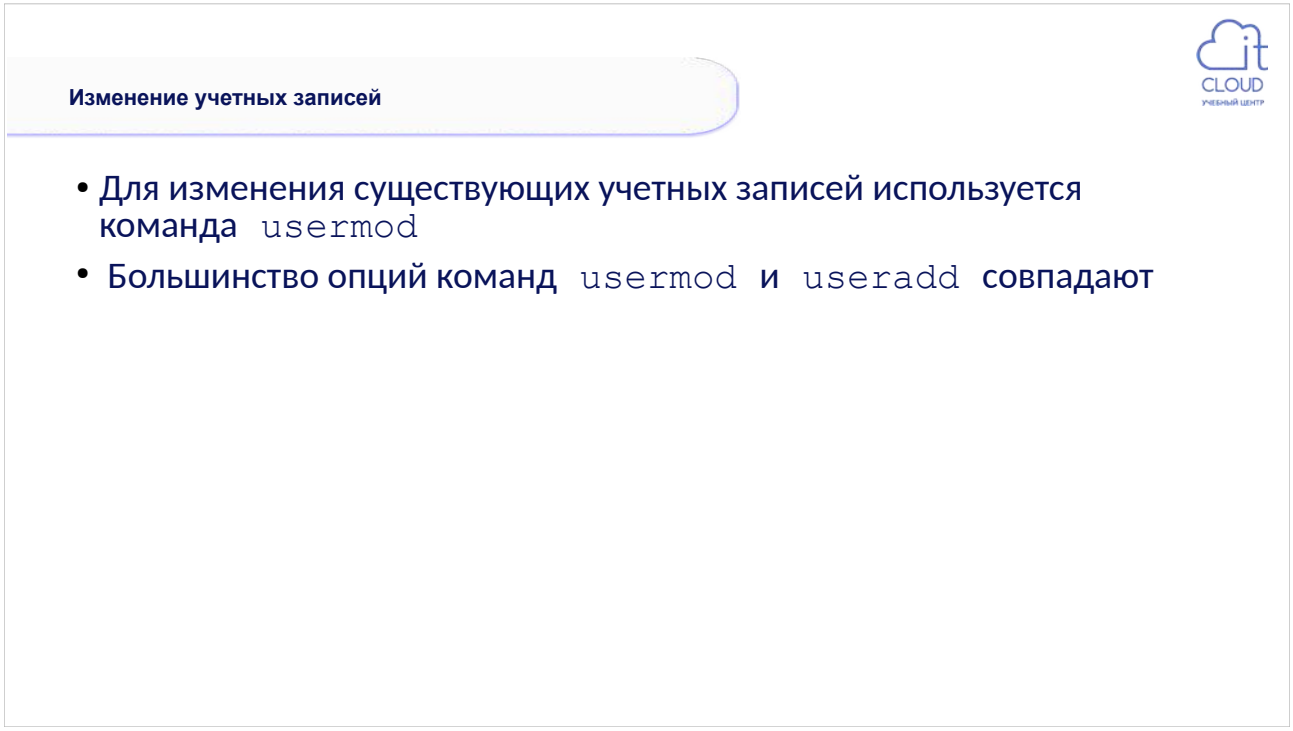
```
root@s10:~# id user2
uid=1003(user2) gid=1003(user2) группы=1003(user2),100(users)
root@s10:~# ls -a /home/
.  ..  sa  user2
root@s10:~# getent passwd user2
user2:x:1003:1003:,,,:/home/user2:/bin/bash
```

```
root@s10:~# getent shadow user2
user2:$y$j9T$8YWDau32zLa949o3C1A0Z1$gL2vS3zXY97H0l8OZSlHFlXmQt2rEMoYMf6VNELWmB/:
20122:0:99999:7:::
```

Настройки команды `adduser` находятся в файле `/etc/adduser.conf`.

Пример: изменим файл `/etc/adduser.conf` и проверим, как будут создаваться пользователи.

```
root@s10:~# tail -8 /etc/adduser.conf
EXTRA_GROUPS="cdrom floppy video"

# Setting this to something other than 0 will cause adduser to add
# newly created non-system users to the list of groups defined by
# EXTRA_GROUPS.
# Default: ADD_EXTRA_GROUPS=0
ADD_EXTRA_GROUPS=1

root@s10:~# adduser user3
Добавляется пользователь «user3» ...
Добавляется новая группа «user3» (1004) ...
Adding new user `user3' (1004) with group `user3 (1004)' ...
Создаётся домашний каталог «/home/user3» ...
Копирование файлов из «/etc/skel» ...
Новый пароль:
НЕУДАЧНЫЙ ПАРОЛЬ: Пароль должен содержать не менее 8 символов
Повторите ввод нового пароля:
passwd: пароль успешно обновлён
Изменение информации о пользователе user3
Введите новое значение или нажмите ENTER для выбора значения по умолчанию
      Полное имя []:
      Номер комнаты []:
      Рабочий телефон []:
      Домашний телефон []:
      Другое []:
Данная информация корректна? [Y/n]
Adding new user `user3' to supplemental / extra groups `users, users, cdrom,
floppy, video' ...
Добавляется пользователь «user3» в группу «users» ...
Добавляется пользователь «user3» в группу «cdrom» ...
Добавляется пользователь «user3» в группу «floppy» ...
Добавляется пользователь «user3» в группу «video» …

root@s10:~# grep USERS_GROUP /etc/adduser.conf
# Default: USERS_GID=undefined, USERS_GROUP=users
#USERS_GROUP=users
```



Если необходимо произвести некоторые изменения в учетной записи уже зарегистрированного пользователя, то для этого предназначена команда `usermod` .

Пример: смена оболочки по умолчанию для пользователя:

```
root@s10:~# usermod -s /bin/false tempuser
root@s10:~# getent passwd tempuser
tempuser:x:1002:1002::/home/tempuser:/bin/false
```

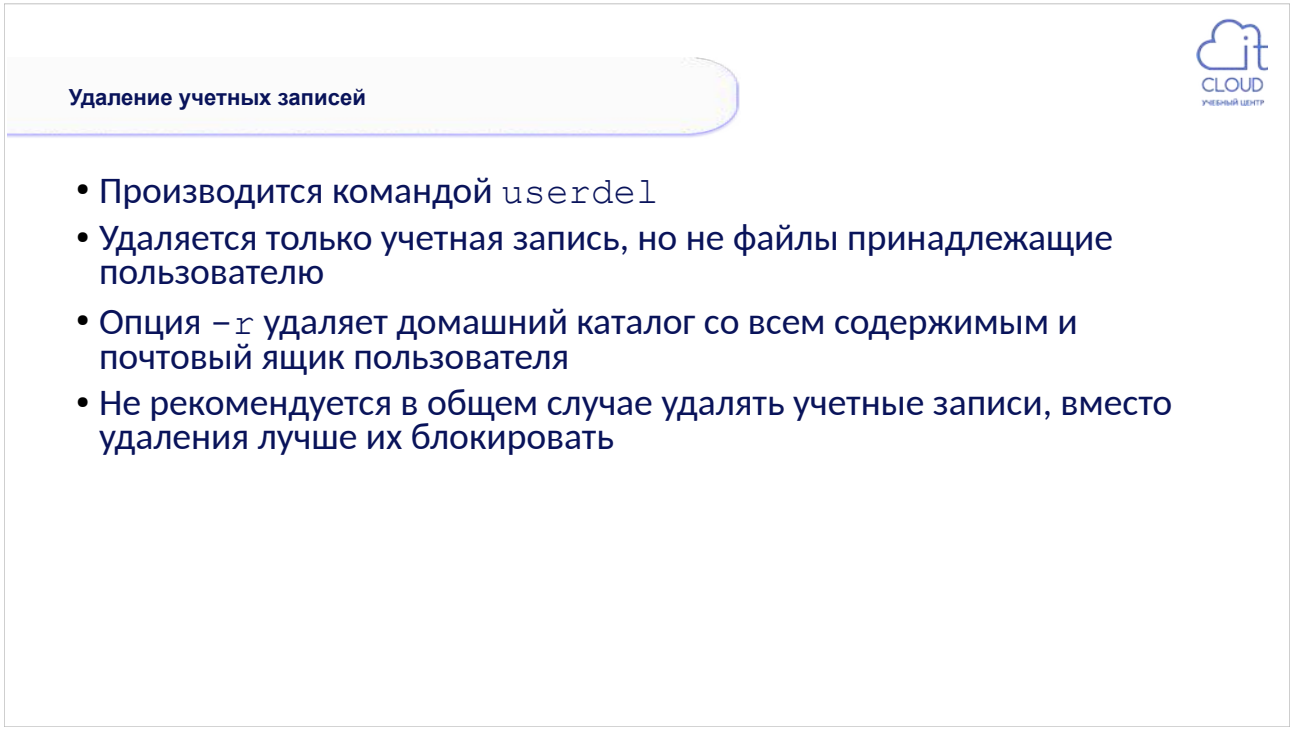
Большая часть опций команд `useradd` и `usermod` совпадают.

Пример: для добавления новой группы, в которых участвует пользователь, можно использовать следующую команду:

```
root@s10:~# usermod -aG "sudo,adm" tempuser
root@s10:~# id tempuser
uid=1002(tempuser) gid=1002(tempuser)
группы=1002(tempuser),4(adm),27(sudo),100(users)
```

Используя команду `usermod` можно также указать для пользователя его новое имя с помощью опции `-l`.

Опции `-L` и `-U` позволяют, соответственно, блокировать и разблокировать возможность входа в сеанс для данного пользователя.



Для удаления учетной записи пользователя следует воспользоваться командой `userdel`.

Пример:

```
root@s10:~# userdel user2
root@s10:~# ls -l /home/
итого 12
drwx------ 13 sa    sa    4096 фев  3 12:28 sa
drwx------  2  1003  1003 4096 фев  3 15:27 user2
drwx------  2 user3 user3 4096 фев  3 15:40 user3
```

Перед удалением учетной записи пользователя необходимо решить, что делать с файлами пользователя, если таковые в системе имеются.

Сама команда `userdel`, по умолчанию, удаление файлов пользователя не производит. Поэтому все файлы пользователя, учетная запись которого подлежит удалению, должны быть найдены и либо удалены, либо сохранены в архив, либо переданы другому пользователю.

Пример:

```
#find / -user 1003 -exec rm -rf {} \;
```

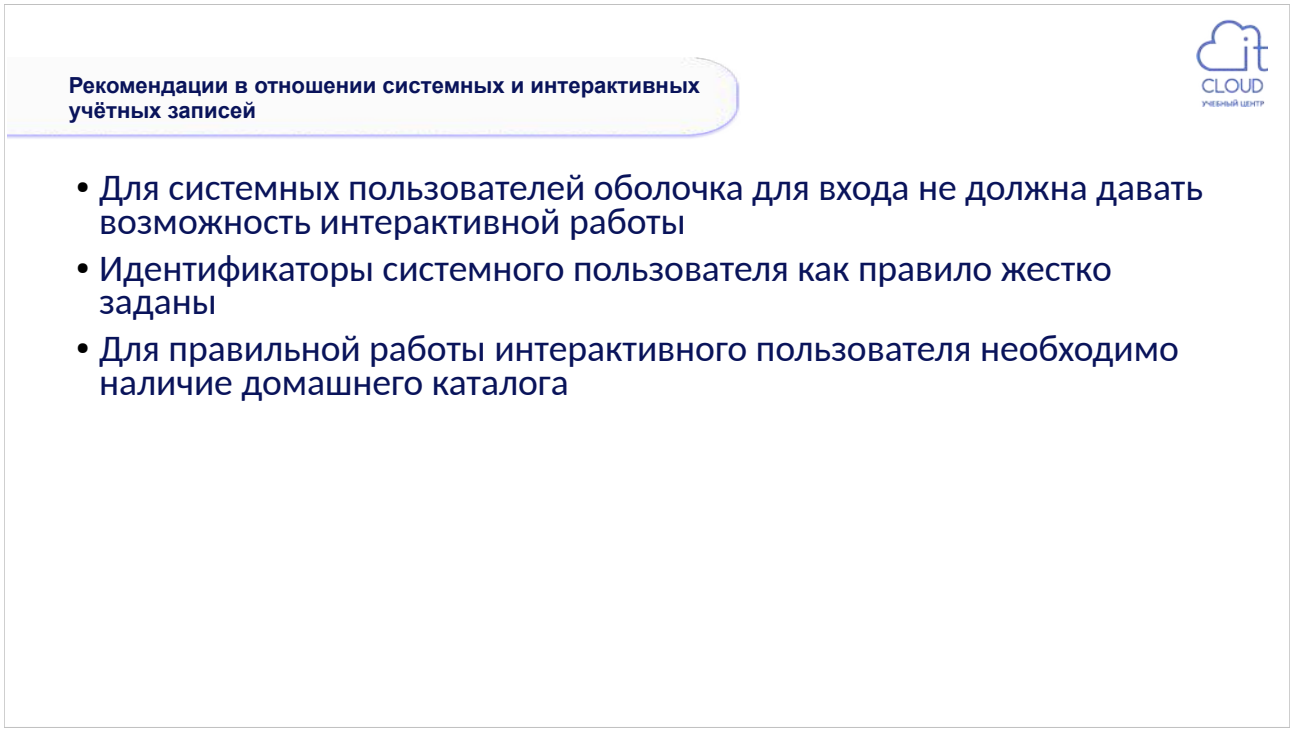
***Примечание**: Здесь вместо имени пользователя использовался UID, поскольку пользователь с таким UID был уже удален.*

Опция `–r` удаляет домашний каталог со всем содержимым и почтовый ящик пользователя

Удаление учетных записей может приводить к «осиротевшим» файлам. Более того идентификатор учетной записи может быть повторно использован, а это означает, что

потенциально все «осиротевшие» файлы удаленного пользователя могут стать доступны новому пользователю. Поэтому не рекомендуется производить удаление учетных записей, вместо этого ненужные учетные записи лучше блокировать.

4.3 Рекомендации в отношении системных и интерактивных учётных записей



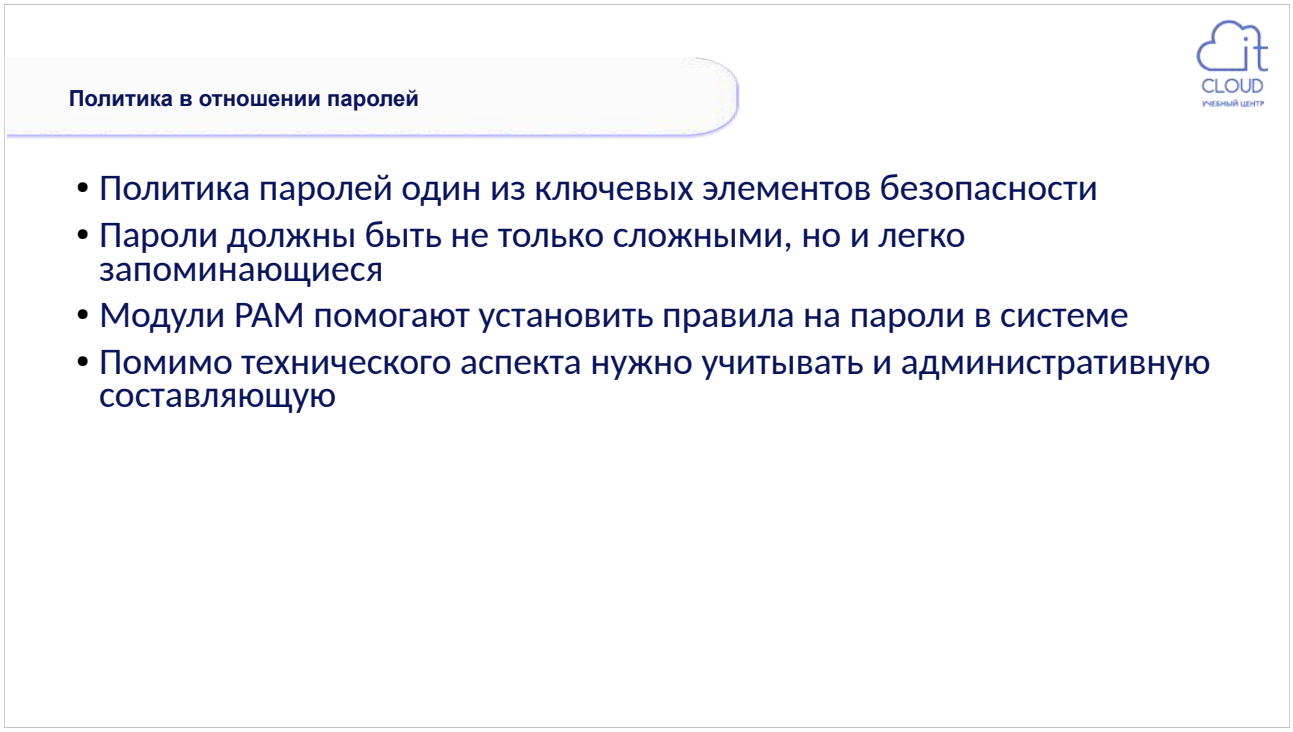
Если пользователь не имеет право входить в сеанс (как правило это специальные системные или служебные пользователи), то в качестве оболочки должен быть установлен один из следующих вариантов:

- `/bin/false` – системная команда, всегда возвращающая код 1 (код ошибочного завершения);
- `/dev/null` – специальный файл символьного устройства, при попытке запуска которого возникает ошибка;
- `/sbin/nologin` – системная команда, возвращающая при запуске код ошибки и сообщение о невозможности входа в сеанс.

UID и GID системных пользователей должны иметь определенные значения для правильного функционирования программ, для которых они создаются.

Для входа интерактивным пользователем необходимо наличие домашнего каталога. Иначе пользователь либо не может войти в сеанс или попадает в корневой каталог.

4.4 Политика в отношении паролей. Инструменты и методы создания, проверки и подбора паролей



Правила установки, использования и управления паролями являются важнейшей частью системной политики. Обычно они включают в себя, как минимум, следующее:

- Определение категорий пользователей, которые имеют право самостоятельного выбора паролей с помощью команды `passwd`.
- Правила выбора паролей и требования к их уровню сложности.
- Сроки устаревания паролей.
- Длительности периодов запрета на изменение паролей.

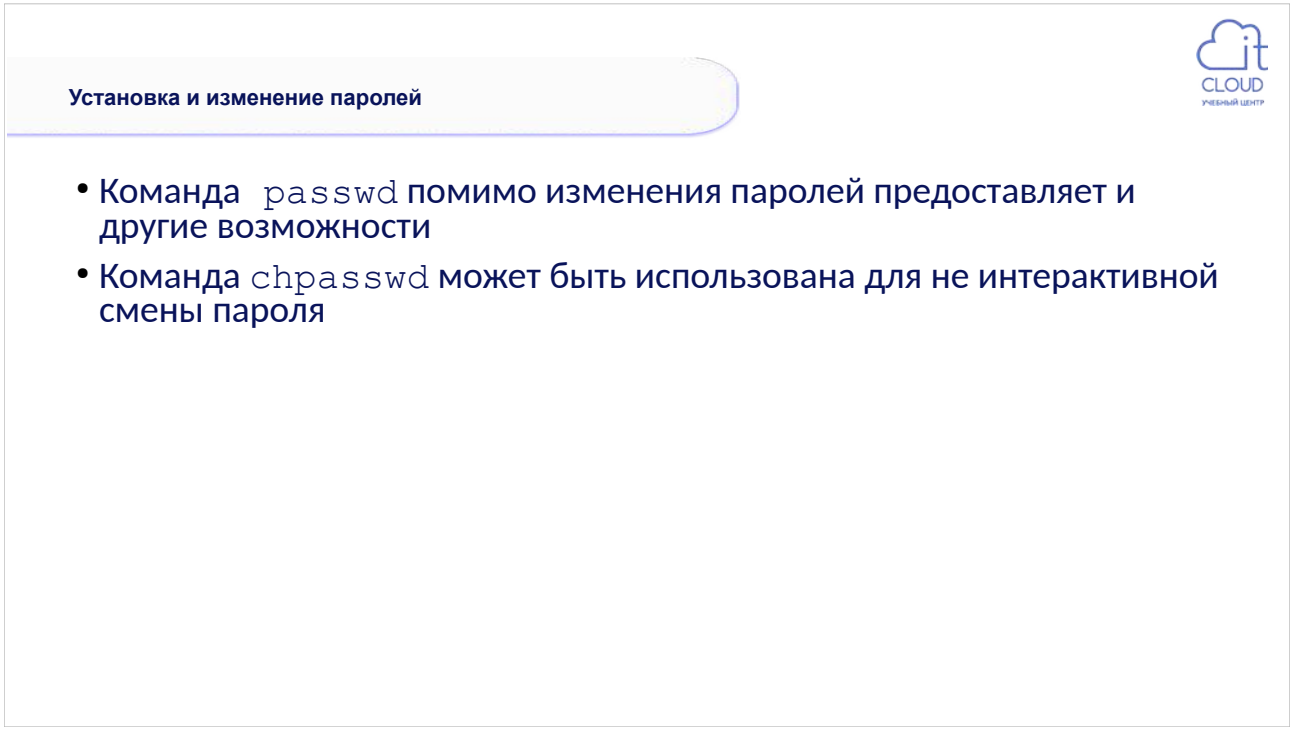
Четко сформулированная политика управления паролями значительно облегчает администрирование системы.

Для установки правил выбора паролей, их минимальной длины и требуемого уровня сложности, достаточно настроить модуль контроля паролей системы PAM для автоматической проверки соответствия выбираемого пользователем пароля системной политике.

В работе с паролями необходимо помнить, что, с одной стороны, пароли должны быть сложными, с другой стороны, они должны легко запоминаться пользователем. Если пользователь не может запомнить свой супер сложный пароль, то он его будет записывать и это приведет к снижению безопасности системы.

Настройки PAM для команды passwd обычно находятся в файлах `/etc/pam.d/passwd` и, возможно, в `/etc/pam.d/system-auth`.

Помимо технических механизмов, регулирующих правила использования паролей, следует учитывать и административную составляющую. Требуется проводить разъяснение, обучение, контроль, поощрение и наказание пользователей.



Команда `passwd` помимо изменения паролей предоставляет и другие возможности. Ниже приведен список наиболее часто применяемых опций команды `passwd` :

`-l` – блокирование учетной записи;

`-u` – разблокирование учетной записи;

`-S` – получение текущего состояния пароля;

`-d` – удалить пароль;

`-n` – установка периода запрета на смену пароля (минимальное время жизни пароля);

`-x` – установка максимального срока использования пароля;

`-w` – установка количества дней до момента устаревания пароля, начиная с которого пользователю будут выдаваться предупреждения о необходимости смены пароля;

`-i` – срок после устаревания пароля, по прошествии которого учетная запись блокируется.

Пример: создание пароля и блокирования учетной записи.

```
root@s10:~# id tempuser
uid=1002(tempuser) gid=1002(tempuser) groups=1002(tempuser),4(adm),100(users)
root@s10:~# getent shadow tempuser
tempuser:!!:17374:0:99999:7::17652:
root@s10:~# passwd tempuser
Changing password for user tempuser.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
root@s10:~# getent shadow tempuser
tempuser:$6$VIvms7KE$LWaXxUGHVufzCJ2U1YRfUfUcxU3P0wabbILSrO319M62O
LdEuf.nl4.MLNnn06OXOBY3aibvYx8Rq/pOlIAmB.:17374:0:99999:7::17652:
```

```
root@s10:~# passwd -l tempuser
Locking password for user tempuser.
passwd: Success
root@s10:~# getent shadow tempuser
tempuser:!!$6$VIvms7KE$LWaXxUGHVufzCJ2U1YRfUfUcxU3P0wabbILSrO319M62O
LdEuf.nl4.MLNnn06OXOBY3aibvYx8Rq/pOlIAmB.:17374:0:99999:7::17652:
```

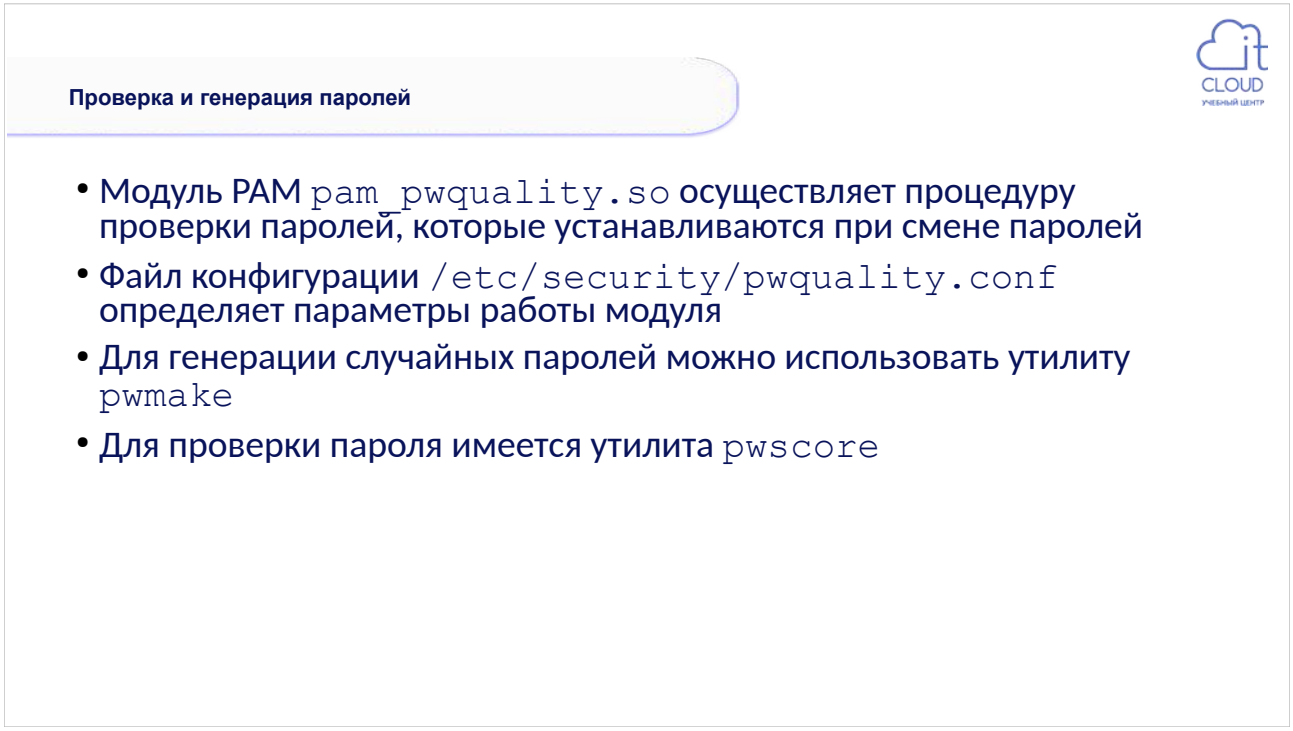
***Примечание**: После блокирования учетной записи в первых позициях второго поля файла /etc/shadow перед шифрованным паролем пользователя появляются знаки восклицания. При разблокировании учетной записи они исчезают.*

В случае, когда необходимо не интерактивно изменить пароль пользователю или выполнить массовую смену пароля, может быть использована команда `chpasswd`.

Пример:

```
root@s10:~# grep user1 /etc/shadow
user1:!:20122:0:99999:7:::
root@s10:~# echo 'user1:P@ssw0rd' | chpasswd
root@s10:~# grep user1 /etc/shadow
user1:$y$j9T$4CkVg3nSEYhPK9kUafKJe/
$c5zqnTFrKdObA83..fxah.Y9pNkCRc3.epQizH0JnRD:20122:0:99999:7:::
```

В примере выше назначается пароль P@ssw0rd для пользователя user1.



Пакет libpwquality-tools содержит инструменты для генерации и проверки паролей.

Модуль PAM pam_pwquality.so предназначен для проверки устанавливаемых паролей. Этот модуль подключается через тип модуля password:

```
root@s10:~# grep ^password /etc/pam.d/common-password
password    requisite               pam_pwquality.so retry=3
password    [success=1 default=ignore]    pam_unix.so obscure use_authtok
try_first_pass yescrypt
password    requisite               pam_deny.so
password    required                pam_permit.so
password    optional    pam_gnome_keyring.so
```

Настройки модуля находятся в файле `/etc/security/pwquality.conf`.

В некоторых системах используется похожий модуль `pam_cracklib.so`.

Утилита `pwmake` переназначена для генерации паролей. В качестве аргумента используется количество случайных бит для генерации пароля. Минимальное количество бит 56.

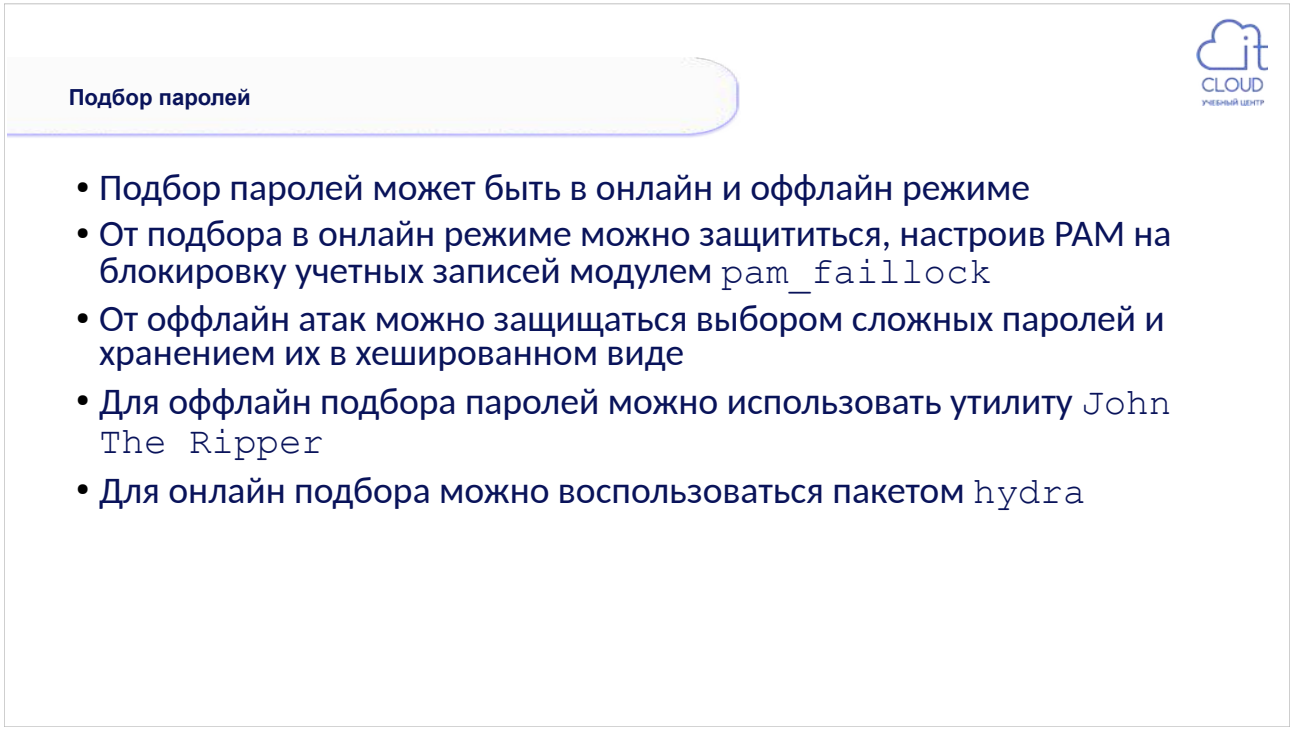
Утилита `pwscore` проверяет пароли, выставляя рейтинг. Пароль с рейтингом выше 50 считается хорошим.

Обе утилиты используют настройки из файла `/etc/security/pwquality.conf`.

Пример: Генерации и проверки паролей с разной степенью сложности.

```
root@s10:~# echo lin123 | pwscore
Проверка сложности пароля завершилась неудачей:
 Пароль должен содержать не менее 8 символов
root@s10:~# echo lin12345 | pwscore
Проверка сложности пароля завершилась неудачей:
 Пароль не прошел проверку орфографии - слишком простой
root@s10:~# echo Lin12345 | pwscore
```

```
Проверка сложности пароля завершилась неудачей:
 Пароль не прошел проверку орфографии - слишком простой
root@s10:~# echo 'Lin1234$' | pwscore
31
root@s10:~# echo 'L1n!2d3g4.' | pwscore
68
root@s10:~# pwmake 56
Qaxoc0xabzew
root@s10:~# echo Qaxoc0xabzew | pwscore
64
root@s10:~# pwmake 128
2iPEq4xcOliJzabiHaqej$4Rx4j
root@s10:~# echo '2iPEq4xcOliJzabiHaqej$4Rx4j' | pwscore
100
```



Существуют два подхода к подбору паролей онлайн и оффлайн атаки. Вне зависимости от подхода подбор может использовать или перебор по словарю, или посимвольный перебор (брутфорс).

Онлайн защита от подбора паролей может осуществляться как сервисами, предоставляющими доступ к системе, например в sshd имеются настройки на количество неудачных попыток входа для разрыва соединения, так и библиотеками PAM. Модуль `pam_faillock.so` позволяет настроить политику блокировки учетных записей.

Настройка модуля `pam_faillock.so` может производиться через редактирование файла `/etc/security/faillock.conf`:

```
…
deny=3
unlock_time=60
…
```

Эти же параметры можно передать и непосредственно в pam-файле:

```
…
auth        requisite           pam_faillock.so preauth deny=3 unlock_time=60
…
```

Основные параметры:

- `dir` — каталог, в котором хранятся пользовательские файлы с записями об ошибках аутентификации (по умолчанию `/var/run/faillock`);
- `audit` — записать в системный журнал имя пользователя, если данного пользователя не существует в системе;
- `silent` — не выводить информационные сообщения (не будет уведомлять пользователя о блокировке учётной записи и времени блокировки);

- `no_log_info` — не регистрировать информативные сообщения в системном журнале;
- `local_users_only` — включение данной опции в файл означает, что модуль будет применяться только для локальных пользователей, существующих в файле `/etc/passwd` (во избежание возможных проблем с централизованными средствами аутентификации: AD,IdM, LDAP, и т.д, у которых могут быть свои методы ограничения доступа к аутентификации);
- `deny` — количество неудачных попыток входа, после которых возможность аутентификации будет заблокирована (по умолчанию 3);
- `fail_interval` — интервал времени (в секундах), в течение которого должны произойти последовательные сбои аутентификации для блокировки учетной записи пользователя (по умолчанию 900 — 15 минут);
- `unlock_time` — интервал времени (в секундах), в течении которого возможность аутентификации для пользователя, превысившего количество попыток входа, будет заблокирована (по умолчанию 600 — 10 минут);
- `even_deny_root` — блокировать учётную запись root так же, как и обычные учетные записи;
- `root_unlock_time` — работает аналогично `unlock_time`. Применяется по отношению к пользователю root, используется совместно с `even_deny_root`;
- `admin_group` — члены группы, указанной в данном параметре, будут обрабатываться этим модулем так же, как и учетная запись root (к ним будут применяться опции `even_deny_root` и `root_unlock_time`).

Подробнее об этих и других существующих параметрах детально можно почитать в `man faillock.conf`.

Включение модуля в подсистеме PAM производится для типов auth и account.

```
auth ... pam_faillock.so {preauth|authfail|authsucc} [dir=/path/to/tally-
directory] [even_deny_root] [deny=n] [fail_interval=n] [unlock_time=n]
[root_unlock_time=n] [audit] [silent] [no_log_info]
account ... pam_faillock.so [dir=/path/to/tally-directory] [no_log_info]
```

Аргумент {preauth|authfail|authsucc} должен быть установлен в соответствии с положением этого экземпляра модуля в стеке PAM:

- `preauth` — должен использоваться, если модуль вызывается перед модулями, которые запрашивают учётные данные пользователя, такие как пароль. Модуль проверяет, заблокирован ли пользователь в случае, если в последнее время было аномальное количество неудачных последовательных попыток аутентификации. Этот вызов является необязательным, если используется `authsucc`;
- `authfail` — должен использоваться, если модуль вызывается после сбоя модулей, определяющих результат аутентификации. Если пользователь ещё не заблокирован из-

за предыдущих сбоев аутентификации, модуль запишет сбой в соответствующий файл с записями об ошибках;

- `authsucc` — должен использоваться, если модуль вызывается после успешного завершения модулей, определяющих результат аутентификации. Если пользователь ещё не заблокирован из-за предыдущих сбоев аутентификации, модуль удалит запись об ошибках в соответствующем файле с записями об ошибках. В противном случае он вернёт ошибку аутентификации. Если этот вызов не выполнен, pam_faillock не будет различать последовательные и непоследовательные неудачные попытки аутентификации. В таком случае необходимо использовать вызов `preauth`. Из-за сложности настройки стека PAM также можно вызвать pam_faillock в модуле account. В этом случае также должен вызываться `preauth`.

Примечание: Использование модуля в режиме preauth без параметра silent или с обязательным полем control приводит к утечке информации о существовании или отсутствии учётной записи пользователя в системе, т.к. сбои не регистрируются для неизвестных пользователей. Сообщение о блокировке учётной записи пользователя никогда не отображается для несуществующих учетных записей пользователей, позволяя злоумышленнику сделать вывод, что конкретная учетная запись не существует в системе.

Пример:

1. Настроим модули PAM:

```
root@s10:~# grep -A5 Primary /etc/pam.d/common-auth
# here are the per-package modules (the "Primary" block)
auth  required                        pam_faillock.so preauth
auth  [success=1 default=ignore]    pam_unix.so nullok
auth    [default=die]                    pam_faillock.so authfail
auth    sufficient                       pam_faillock.so authsucc
# here's the fallback if no module succeeds
root@s10:~# grep -A3 Primary /etc/pam.d/common-account
# here are the per-package modules (the "Primary" block)
account required                        pam_faillock.so
account     [success=1 new_authtok_reqd=done default=ignore]     pam_unix.so
# here's the fallback if no module succeeds
```

2. Настраиваем параметры блокировки:

```
root@s10:~# grep -v '^#' /etc/security/faillock.conf
dir = /var/run/faillock
audit
silent
local_users_only
deny = 3
fail_interval = 1800
unlock_time = 1800
```

3. Проверяем результат:

```
root@s10:~# faillock
faillock: Error reading tally directory: No such file or directory
root@s10:~# login user1
Пароль:

Неверное имя пользователя
s10 имя пользователя: user1
Пароль:
```

```
Неверное имя пользователя
s10 имя пользователя: user1
Пароль:

Неверное имя пользователя
s10 имя пользователя: user1
Пароль:

Неверное имя пользователя
s10 имя пользователя: user1
Пароль:

Неверное имя пользователя
Превышено максимальное число попыток (5)
root@s10:~# login sa
Пароль:

Неверное имя пользователя
s10 имя пользователя: sa
Пароль:
Linux s10 6.1.0-30-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.124-1 (2025-01-12)
x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Последний вход в систему: Пн фев  3 14:08:44 +05 2025 с 192.168.98.151 на pts/0
sa@s10:~$
выход
root@s10:~# faillock
sa:
When                Type  Source                                           Valid
user1:
When                Type  Source                                           Valid
2025-02-03 17:56:50 RHOST                                                     V
2025-02-03 17:56:56 RHOST                                                     V
2025-02-03 17:57:03 RHOST                                                     V
```

4. Сброс блокировки:

```
root@s10:~# faillock --user user1
user1:
When                Type  Source                                           Valid
2025-02-03 17:56:50 RHOST                                                     V
2025-02-03 17:56:56 RHOST                                                     V
2025-02-03 17:57:03 RHOST                                                     V
root@s10:~# faillock --user user1 --reset
root@s10:~# faillock --user user1
user1:
When                Type  Source                                           Valid

root@s10:~# login user1
Пароль:
Linux s10 6.1.0-30-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.124-1 (2025-01-12)
x86_64
```

```
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Последний вход в систему: Пн фев  3 18:00:39 +05 2025 на pts/1
$
```

От оффлайн атак блокировка учетных записей помочь не может. Поэтому следует использовать сложные пароли, которые хранятся в хешированном виде.

Для взлома или тестирования паролей мы можем воспользоваться утилитами типа `hydra` (онлайн тестинг) или John The Ripper (оффлайн тестирование).

Для RedHat подобных систем John The Ripper доступен только в исходных кодах. Для повышения производительности необходимо компилировать его под каждую отдельную систему. Исходный код можно получить на официальном сайте http://www.openwall.com/john

В Debian подобных системах имеются пакеты john и john-data. Второй пакет содержит словари для подбора паролей.

Пример:

1. Подготовим файл с паролем пользователя sa:

```
root@s10:~# unshadow /etc/passwd /etc/shadow | grep '^sa:' > passwd
```

2. Скопируем файл со словарем в текущий каталог, чтобы добавить в него свой предполагаемый пароль:

```
root@s10:~# cp /usr/share/john/password.lst .
root@s10:~# head -2 password.lst
lin123
#!comment: This list has been compiled by Solar Designer of Openwall Project
```

3. Запустим подбор паролей:

```
root@s10:~# john --format=crypt --wordlist=password.lst passwd
Created directory: /root/.john
Loaded 1 password hash (crypt, generic crypt(3) [?/64])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
lin123           (sa)
1g 0:00:00:00 100% 1.030g/s 98.96p/s 98.96c/s 98.96C/s lin123..pamela
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

4. Получим результат:

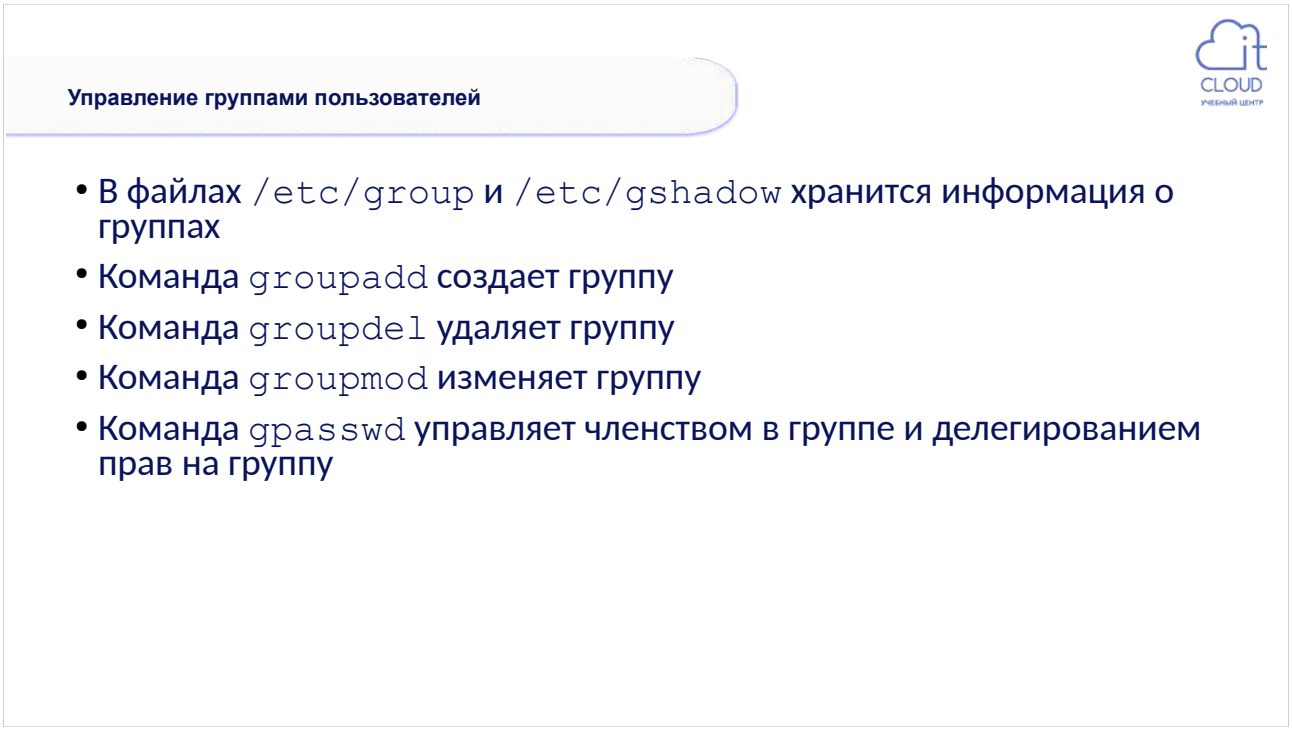
```
root@s10:~# john --show passwd
sa:lin123:1000:1000:sa,,,:/home/sa:/bin/bash

1 password hash cracked, 0 left

root@s10:~# cat .john/john.pot
$y$j9T$/08I6WtR4yBjypiK30k53.$OSa.yy8wMqH7UASihksXZrLpNEWKxM.jBuufr5nq/9.:lin123
```

Для онлайн подбора паролей или для проверки защиты от онлайн перебора можно использовать пакет hydra. Помимо консольной утилиты hydra можно установить hydra-gtk для запуска подбора в графической среде.

Пример:

```
root@s10:~# hydra -l sa -P password.lst ssh://127.0.0.1
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in
military or secret service organizations, or for illegal purposes (this is non-
binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-02-03
18:32:15
[WARNING] Many SSH configurations limit the number of parallel tasks, it is
recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 3560 login tries
(l:1/p:3560), ~223 tries per task
[DATA] attacking ssh://127.0.0.1:22/
[22][ssh] host: 127.0.0.1   login: sa   password: lin123
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-02-03
18:32:19
```

4.5 Управление группами пользователей.



С помощью создания групп пользователей системный администратор может эффективно управлять деятельностью в системе целыми коллективами пользователей, предоставляя им разрешения на доступ к системным ресурсам.

***Примечание**: Каждый файл располагает в метаданных триадой бит, кодирующей права доступа для группы пользователей. Следовательно, изменяя членство пользователя в группах, администратор изменяет, таким образом, привилегии пользователя на доступ к различным файлам в системе, не меняя при этом права пользователя на принадлежащие ему файлы.*

Информация о группах пользователей хранится в файле `/etc/group` в виде строк.

Формат записи: name:password:GID:user(s), где:

- Первое поле – имя группы.
- Второе поле – пароль группы. Если он не используется, в этом поле ставится звездочка.
- Третье поле - GID группы.
- Четвертое поле содержит список пользователей, принадлежащих к данной группе, разделенных запятыми.

Для добавления новой группы необходимо воспользоваться командой `groupadd` , которая добавляет новую запись в файл `/etc/group` .

Пример:

```
root@s10:~# groupadd class
root@s10:~# getent group class
class:x:1003:
```

Пользователи для которых группа является первичной имеют информацию об этом в GID, который хранится в четвертом поле файла `/etc/passwd`.

Имена пользователей, входящих в группу, которая не является для них первичной, записываются через запятую в четвертом поле файла `/etc/group` .

Пример:

```
root@s10:~# getent group users
users:x:100:tempuser,user
```

Для явного указания идентификатора группы необходимо воспользоваться опцией `-g`.

Пример:

```
root@s10:~# groupadd -g 1512 project
```

Для удаления группы необходимо воспользоваться командой `groupdel`.

Пример:

```
root@s10:~# groupdel project
```

Группа пользователей может быть создана для работы над каким-либо проектом. В таком случае бывает удобно одного из пользователей сделать администратором группы и делегировать ему право добавлять уже зарегистрированных в системе пользователей в эту группу и удалять их из группы при необходимости.

Для назначения администратора группы суперпользователю необходимо использовать команду `gpasswd -A`

Системный администратор может добавлять пользователей в группу с помощью команды `gpasswd -M`.

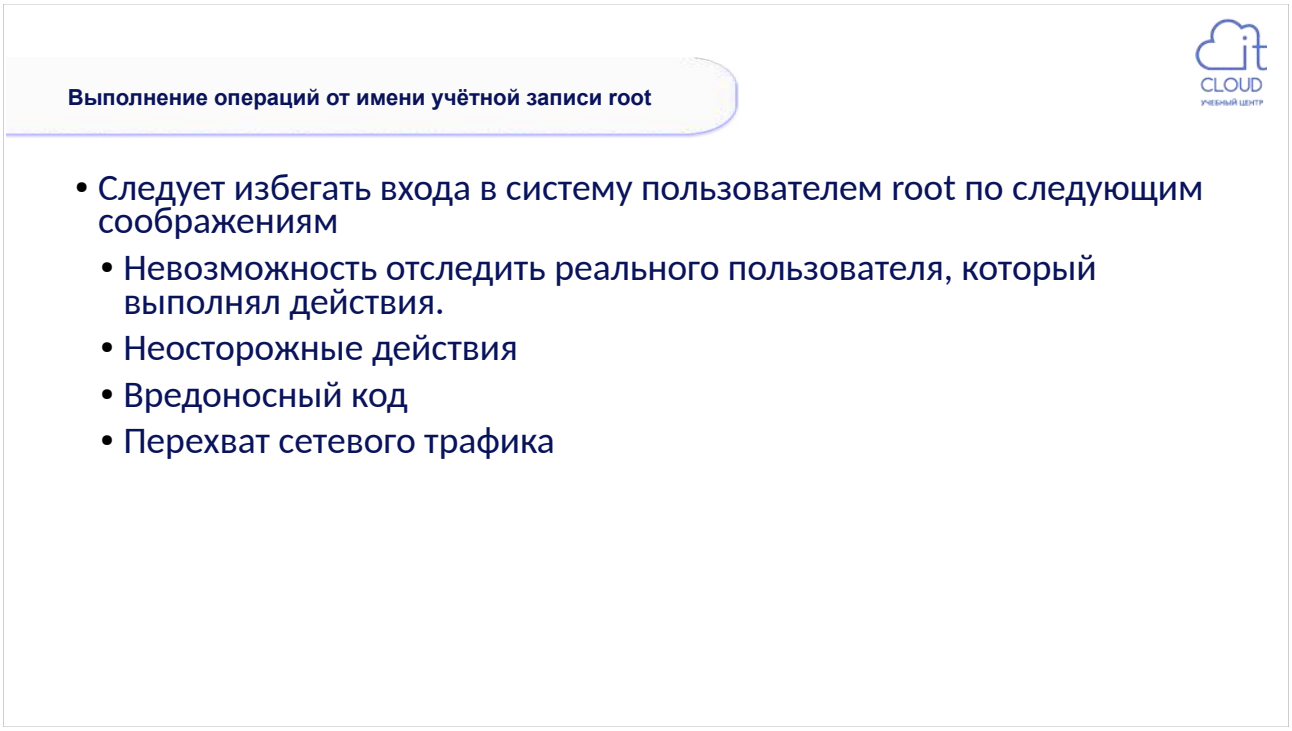
Администратор группы может:

- добавить пользователя в группу с помощью команды `gpasswd -a`;
- удалить пользователя из группы командой `gpasswd -d`.

Пример:

```
root@s10:~# groupadd developers
root@s10:~# gpasswd -A user developers
root@s10:~# su - user
...
user@sl0:~$ gpasswd -a tempuser developers
```

4.6 Выполнение операций от имени учётной записи root



Поскольку суперпользователь является таким же членом системы, как и остальные пользователи, можно войти в систему непосредственно под именем root. Однако оказывается, что это довольно неудачное решение.

Во-первых, не будет сделано никаких записей о том, какие операции выполнял суперпользователь.

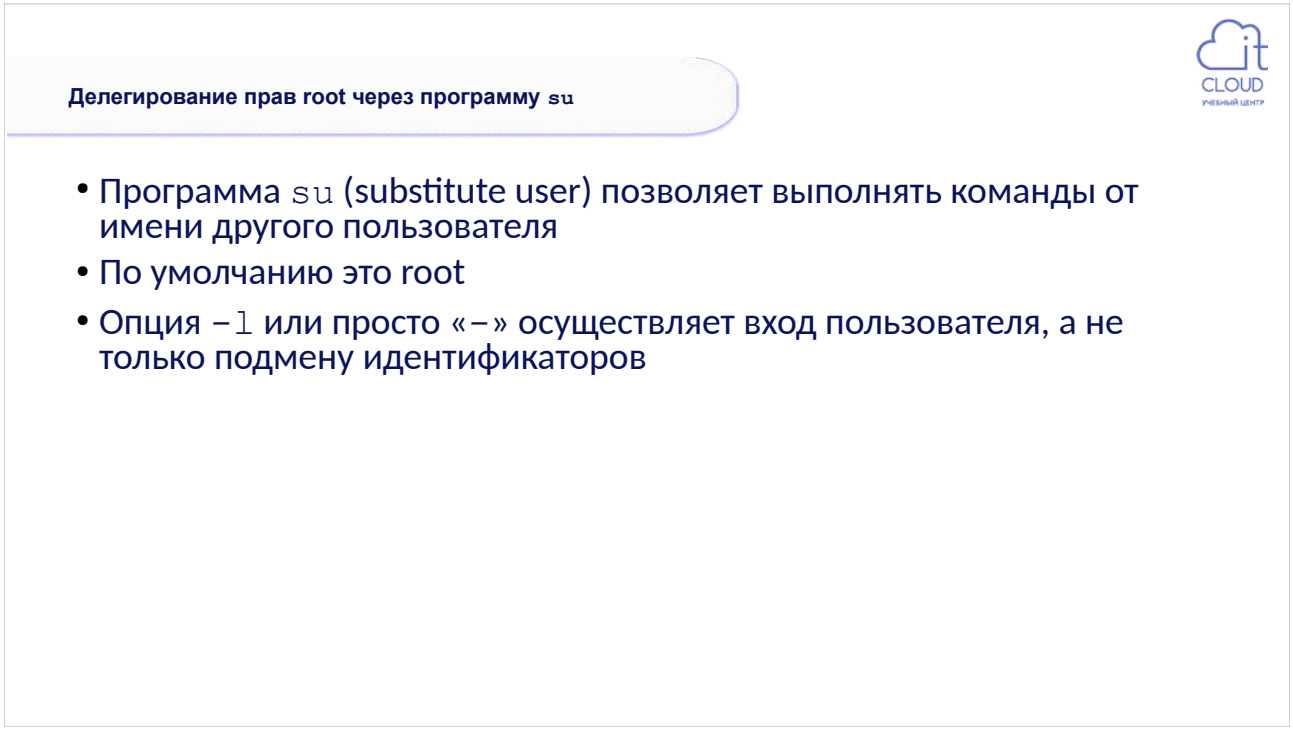
Во-вторых, сценарий регистрации суперпользователя не предполагает сбора дополнительной идентифицирующей информации. Когда под именем root в систему могут входить несколько пользователей, не существует способа определить, кто именно из них и когда это делал.

В-третьих, неосторожные действия могут привести к краху всей системы.

В-четвертых, вредоносный код может получить полный контроль над системой.

В-пятых, злоумышленники могут перехватывать сетевой трафик и выявить пароль суперюзера.

Вследствие упомянутых причин рекомендуется вход под именем root запрещать на терминалах и по сети, т.е. везде, кроме системной консоли.



С точки зрения безопасности лучше получать доступ к учетной записи root с помощью команды `su`.

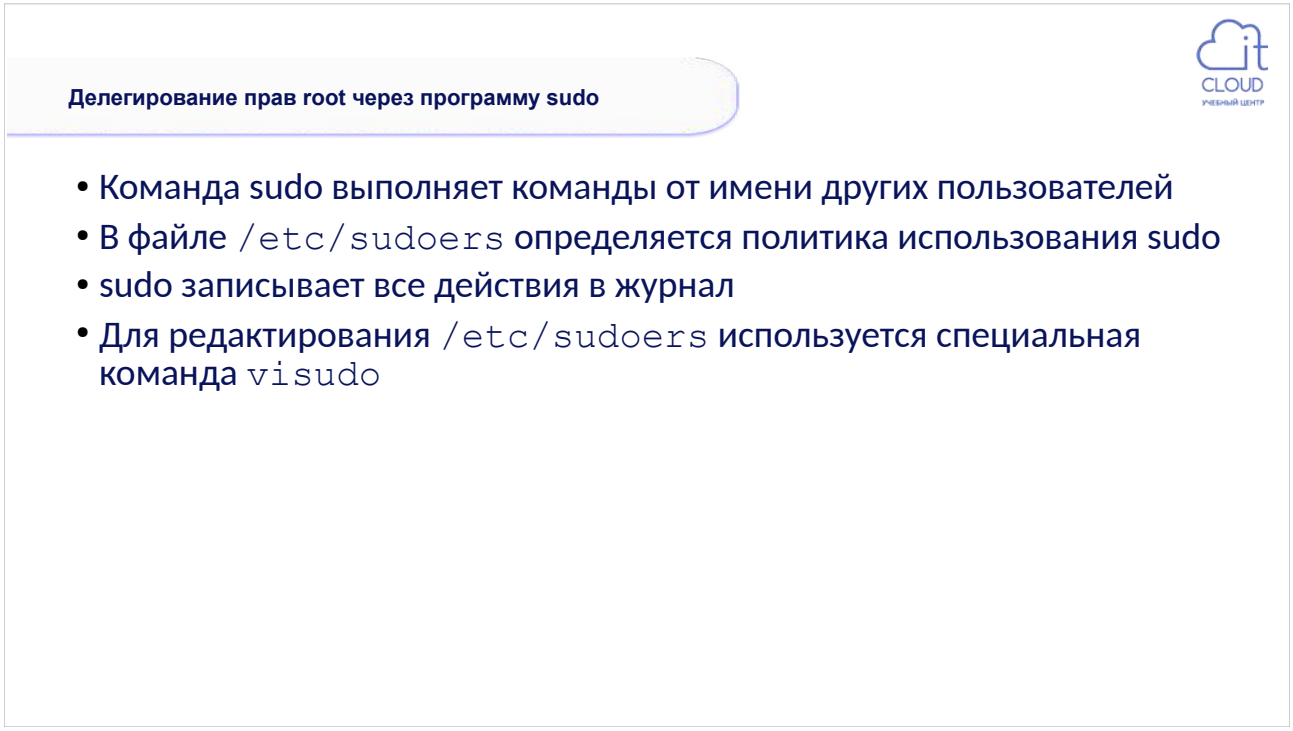
Будучи вызванной без аргументов, `su` выдает приглашение на ввод пароля суперпользователя, а затем запускает интерпретатор команд с правами пользователя root. Интерпретатор будет выполняться в привилегированном режиме, пока не завершит работу (по команде exit или при нажатии клавиш <CTRL+D>).

Команда `su` не фиксирует действия, производимые в среде интерпретатора, но добавляет запись в журнальный файл с указанием, кто и когда вошел в систему под паролем суперпользователя.

Команда `su -` производит вход пользователя с обработкой профильных файлов.

Команда `su` способна также подставлять вместо имени root имена других пользователей.

Рекомендуется взять за правило при вводе команды указывать полное имя, например `/bin/su`, а не просто `su`. Это послужит определенной защитой от тех программ с именем `su`, которые преднамеренно были прописаны в переменной среды PATH злоумышленником.



Использование утилиты `sudo` имеет следующие преимущества:

- благодаря регистрации команд значительно повышается степень административного контроля над системой,
- операторы могут выполнять специальные задачи, не имея неограниченных привилегий;
- настоящий пароль суперпользователя могут знать всего один-два человека;
- вызывать утилиту sudo быстрее, чем выполнять команду su или входить в систему под именем root,
- пользователя можно лишить привилегий, не меняя пароль суперпользователя
- ведется список всех пользователей с правами пользователя root;
- меньше вероятность того, что интерпретатор команд, запущенный суперпользователем, будет оставлен без присмотра;
- управлять доступом ко всей сети можно с помощью одного файла.

Утилита имеет `sudo` и недостатки. Самый большой из них заключается в том, что любая брешь в системе защиты того или иного привилегированного пользователя эквивалентна нарушению безопасности самой учетной записи root.

Утилита `sudo` в качестве аргумента принимает командную строку, которая подлежит выполнению от имени пользователя root (или другого уполномоченного пользователя).

Утилита обращается к файлу `/etc/sudoers`, где содержится список пользователей, имеющих разрешение на ее выполнение, и перечень команд, которые они могут вводить на конкретном компьютере. Если запрашиваемая команда разрешена, утилита sudo приглашает

пользователя ввести его собственный пароль и выполняет команду от имени суперпользователя.

Далее утилита `sudo` позволяет, не вводя пароль, выполнять другие команды, но только до тех пор, пока не наступит пятиминутный период бездействия (его продолжительность можно менять).

Утилита `sudo` ведет журнал, где регистрируются выполненные команды и вызвавшие их пользователи, а также каталоги, из которых запускались команды, и время их вызова. Эта информация может направляться в систему Syslog или сохраняться в любом журнальном файле по усмотрению пользователя.

Пример: Строка журнального файла, содержащая данные о пользователе user1, который выполнил команду `sudo /bin/cat /etc/sudoers`, может выглядеть следующим образом:

```
Dec 7 10:47:23 comp1 sudo: user1: TTY=ttyp0 ; PWD=/comp1/users/user1; USER=root;
COMMAND=/bin/cat /etc/sudoers
```

В файле `/etc/sudoers` можно указать псевдонимы:

- `Host_Alias` – для компьютеров с которых можно запускать команды;
- `Cmnd_Alias` – для определения команд;
- `User_Alias` – для определения групп пользователей;
- `Runas_Alias` – для определения пользователей от имени которых запускаются команды.

В каждую спецификацию прав доступа включается информация о:

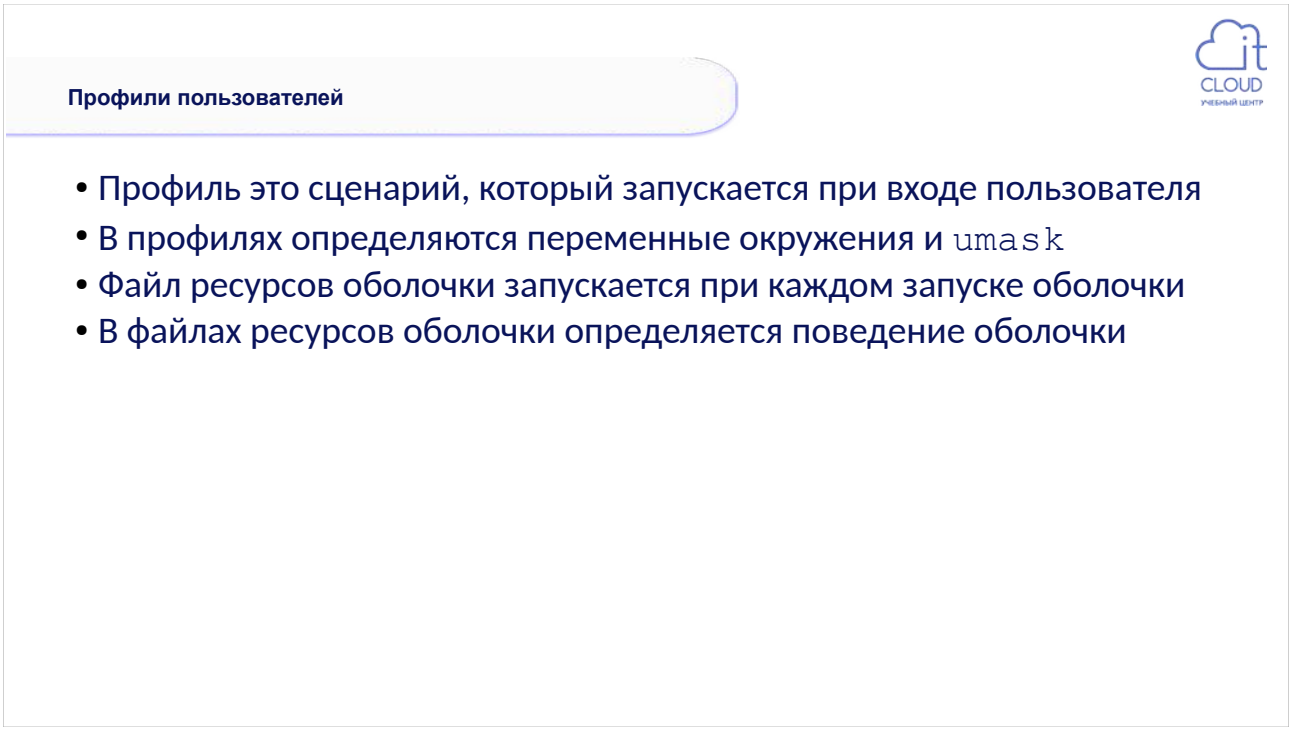
- пользователях, к которым относится запись;
- компьютерах, на которых пользователям разрешено выполнять соответствующие действия;
- командах, которые могут выполняться указанными пользователями;
- пользователях, от имени которых могут выполняться команды.

Пример: строка в `sudoerrs`, позволяющая пользователям группы users выключать компьютер

```
%users  localhost=/sbin/shutdown -h now
```

Для модификации файла `/etc/sudoers` предназначена специальная команда `visudo`, которая проверяет, не редактируется ли файл кем-то посторонним, затем открывает его в редакторе, а перед инсталляцией файла выполняет синтаксический контроль. Последний этап особенно важен, поскольку ошибка в файле `/etc/sudoers` может не позволить повторно вызвать утилиту `sudo` для исправления файла.

4.7 Профили пользователей.



При входе пользователей в сеанс автоматически выполняются специальные файлы сценариев, называемые профилями пользователей.

Обычный подход к хранению настроек оболочки состоит в разделении настроек (профилей) на глобальный профиль (Master Profile) и пользовательские профили (Login Profiles).

Кроме профилей имеются еще и специальные файлы настроек оболочек (resource files), которые также являются сценариями оболочек.

Отличие профилей от файлов ресурсов состоит в том, что сценарии профилей исполняются единожды при входе пользователя в сеанс, а файлы ресурсов запускаются при каждом запуске оболочки.

Если оболочка Bash запущена интерактивно при входе пользователя в сеанс (то есть является оболочкой по умолчанию), то сначала исполняется общий для всех пользователей файл `/etc/profile` , а затем индивидуальный профиль пользователя, находящийся в его домашнем каталоге.

Для оболочки Bash индивидуальный профиль находится в файле, который может называться одним из следующих имен:

- `~/.bash_profile`
- `~/.bash_login`
- `~/.profile`

В файлах профилей чаще всего устанавливаются такие переменные окружения, как:

- PATH - имена каталогов, в которых осуществляется поиск исполняемых файлов для запуска;
- TERM - тип терминала;
- USER - имя пользователя (устанавливается с помощью id -un);
- HOME - путь к домашнему каталогу пользователя;
- MAIL - путь к почтовому ящику пользователя;
- HOSTNAME - имя системы.

Переменные окружения, устанавливаемые в файлах профилей, должны быть экспортированы с помощью команды export.

Пример: к списку каталогов в переменной окружения PATH добавляется каталог bin, находящийся в домашнем каталоге пользователя:

```
PATH=$PATH:$HOME/bin
export PATH
```

Помимо переменных окружения в файлах профиля часто устанавливается значение umask.

При необходимости исполнить файл профиля из командной строки следует использовать команду source .

Пример:

```
# source /etc/profile
```

В противоположность профилям файл ресурсов оболочки ~/.bashrc выполняется только при интерактивном запуске оболочки Bash из командной строки, а не при входе в сеанс.

Для того, чтобы дополнительные настройки оболочки срабатывали не только при запуске оболочки из командной строки (то есть из уже запущенной оболочки), но и при запуске Bash по умолчанию при входе в сеанс, вызов инструкций в файле ~/.bashrc производится из пользовательского профиля.

Пример: Типичное содержимое файла пользовательского профиля таково:

```
root@s10:~$ cat .bash_profile
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
      . ~/.bashrc
fi

# User specific environment and startup programs

PATH=$PATH:$HOME/.local/bin:$HOME/bin

export PATH
```

***Примечание**: Здесь приведен пример содержимого файла пользовательского профиля, в котором проверяется наличие в домашнем каталоге пользователя файла ресурсов оболочки и, если он есть, содержимое его выполняется в контексте текущей оболочки. Это достигается с помощью так называемой inline подстановки - команды точка (.). Вызов . `~/.bashrc` приводит к тому (обратите внимание на точку, с которой начинается команда), что переменные, псевдонимы и функции, определенные в файле ресурсов будут доступны в текущей оболочке. Inline подстановка всегда используется для передачи из одного файла сценария оболочки в другой скрипт переменных, псевдонимов и функций.*

Довольно часто в файле `~/.bashrc` находится inline вызов общесистемного файла ресурсов `/etc/bashrc`.

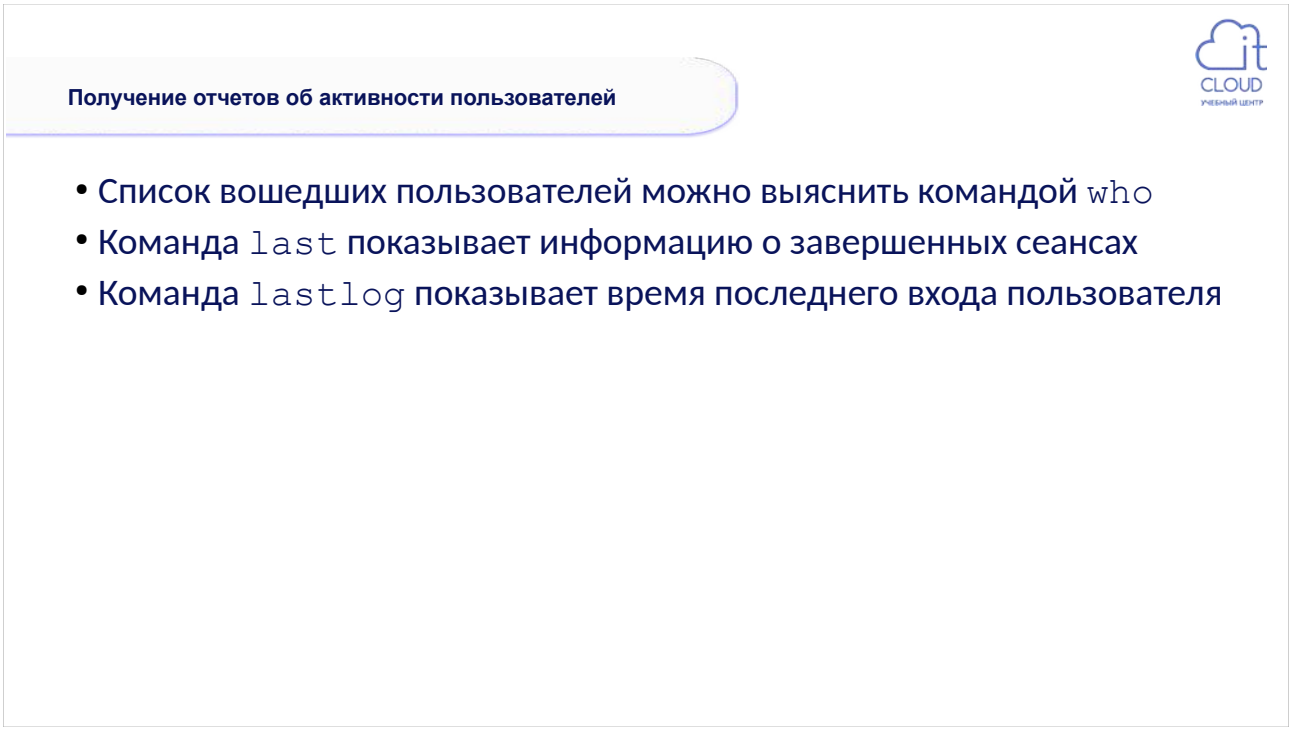
Это не обязательно, но очень удобно, так как в этом файле можно определить, например, псевдонимы для команд, которыми часто пользуются различные пользователи системы, вместо определения этих псевдонимов в частных файлов ресурсов оболочки ~/.bashrc.

Ниже приведен список действий, которые обычно выполняются автоматически при входе в сеанс Bash:

1. Исполняется общесистемный скрипт профиля `/etc/profile`.
2. Выполняется пользовательский скрипт профиля в его домашнем каталоге (например, `~/.bash_profile`).
3. В пользовательском профиле проверяется наличие в домашнем каталоге файла ресурсов оболочки `~/.bashrc` , и, при его наличии, он исполняется.
4. Если исполняется файл ресурсов оболочки, то обычно в нем вызывается общесистемный файл ресурсов `/etc/bashrc`.

При запуске оболочки из командной строки выполняются пункты 3 и 4 списка.

4.8 Получение отчетов об активности пользователей.



Команда `who` позволяет получить список пользователей, находящихся в настоящее время в сеансе.

Информация об этом берется из специального двоичного файла `/var/run/utmp` .

Пример:

```
user@s10:~$ who
user     ttyS0        2017-07-27 19:31
root     pts/0        2017-07-27 17:36 (gateway)
```

С помощью этой же команды, используя соответствующие опции, можно получать и другую информацию. Ниже приведены некоторые часто используемые опции команды `who` :

`-b` – время последней загрузки системы;

`-H` – печать заголовка;

`--login` – информация о системных процессах, контролирующих вход в сеанс;

`-q` – печатает всех имена пользователей в сеансе и их количество;

`-w` – текущий статус всех сеансов;

`-u` – подробная информация о сеансах;

`-a` – полная информация о статусе сеансов и процессов, контролирующих вход в сеанс.

Пример: приведенная ниже команда выведет информацию о сеансах пользователей:

```
user@s10:~$ who -uH
NAME     LINE         TIME             IDLE          PID COMMENT
```

```
user     ttyS0        2017-07-27 19:31   .          26596
root     pts/0        2017-07-27 17:36 02:24       19013 (gateway)
```

Для получения отчета о сеансах пользователей, которые уже завершились, необходимо воспользоваться информацией, сохраняемой в файле `/var/log/wtmp` .

Этот бинарный файл имеет ту же структуру, что и `/var/run/utmp` , поэтому его содержимое можно отобразить, указав его в качестве аргумента команды `who` . Однако, для этого предназначена специальная команда `last` .

Пример:

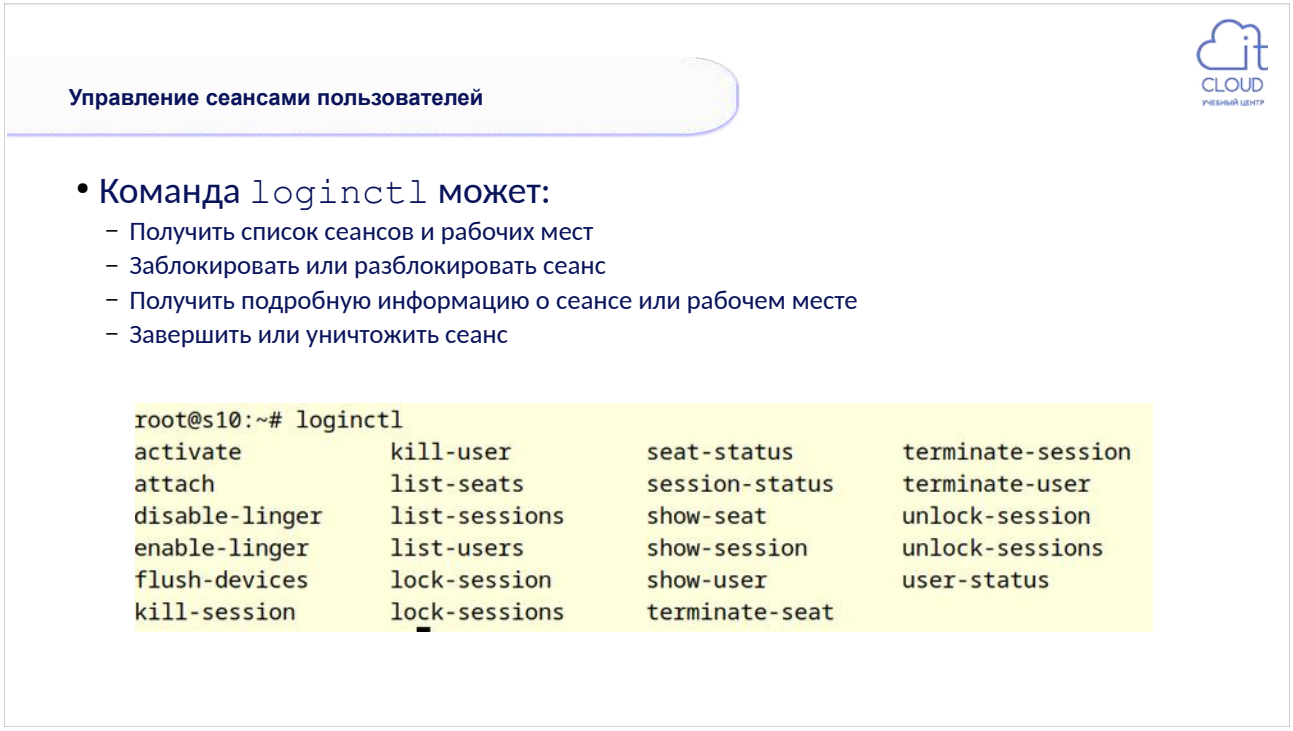
```
user@s10:~$ last | head
user     ttyS0                        Thu Jul 27 19:31   still logged in
root     ttyS0                        Thu Jul 27 17:45 - 19:31  (01:45)
user     ttyS0                        Thu Jul 27 17:43 - 17:44  (00:01)
root     ttyS0                        Thu Jul 27 17:41 - 17:43  (00:02)
root     tty1                         Thu Jul 27 17:40 - 17:40  (00:00)
root     pts/0        gateway         Thu Jul 27 17:36   still logged in
user     tty1                         Thu Jul 27 17:35 - 17:35  (00:00)
user     tty1                         Thu Jul 27 17:34 - 17:34  (00:00)
user     tty1                         Thu Jul 27 17:33 - 17:33  (00:00)
user     tty1                         Thu Jul 27 17:33 - 17:33  (00:00)
```

Имеется также стандартный файл журнала `/var/log/lastlog` , в котором также в бинарном виде хранится информация о последних входах в сеанс.

Для получения информации, находящейся в этом файле, требуется использовать команду `lastlog` .

Пример:

```
user@s10:~$ lastlog | egrep '(user|root)'
root             ttyS0                    Thu Jul 27 19:33:49 +0500 2017
user             ttyS0                    Thu Jul 27 20:03:54 +0500 2017
tempuser                                  **Never logged in**
```



В состав systemd входит утилита `loginctl`, предназначенная для управления сеансами пользователей.

Пример: Получение информации о сенсе

```
root@s10:~# loginctl
SESSION  UID USER SEAT  TTY
     17 1000 sa   seat0
      4 1000 sa         pts/0

2 sessions listed.
root@s10:~# loginctl session-status 17
17 - sa (1000)
           Since: Mon 2025-02-03 18:05:53 +05; 51min ago
          Leader: 3300 (lightdm)
            Seat: seat0; vc7
         Display: :0
         Service: lightdm; type x11; class user
         Desktop: lightdm-xsession
           State: active
            Unit: session-17.scope
                  ├─3300 lightdm --session-child 13 24
                  ├─3310 x-session-manager
                  ├─3358 /usr/bin/VBoxClient --clipboard
                  ├─3359 /usr/bin/VBoxClient --clipboard
                  ├─3374 /usr/bin/VBoxClient --seamless
                  ├─3375 /usr/bin/VBoxClient --seamless
                  ├─3381 /usr/bin/VBoxClient --draganddrop
                  ├─3382 /usr/bin/VBoxClient --draganddrop
                  ├─3390 /usr/bin/VBoxClient --vmsvga-session
                  ├─3391 /usr/bin/VBoxClient --vmsvga-session
                  ├─3394 /usr/bin/ssh-agent x-session-manager
                  ├─3424 /usr/bin/mate-settings-daemon
                  ├─3433 marco
                  ├─3454 mate-panel
```

```
                  ├─3487 /usr/bin/caja
                  ├─3500 nm-applet
                  ├─3503 mate-screensaver
                  ├─3505 mate-volume-control-status-icon
                  ├─3511 mate-power-manager
                  └─3531 /usr/libexec/polkit-mate-authentication-agent-1

фев 03 18:05:53 s10 systemd[1]: Started session-17.scope - Session 17 of User
sa.
фев 03 18:05:55 s10 x-session-manager[3310]: WARNING: Unable to find provider ''
of re>
фев 03 18:05:55 s10 gnome-keyring-daemon[3425]: discover_other_daemon: 1
фев 03 18:05:56 s10 gnome-keyring-daemon[3501]: discover_other_daemon: 1
фев 03 18:05:56 s10 gnome-keyring-daemon[3506]: discover_other_daemon: 1
фев 03 18:05:57 s10 gnome-keyring-daemon[3547]: discover_other_daemon: 1
root@s10:~# loginctl show-session 17
Id=17
User=1000
Name=sa
Timestamp=Mon 2025-02-03 18:05:53 +05
TimestampMonotonic=17832938529
VTNr=7
Seat=seat0
Display=:0
Remote=no
Service=lightdm
Desktop=lightdm-xsession
Scope=session-17.scope
Leader=3300
Audit=17
Type=x11
Class=user
Active=yes
State=active
IdleHint=no
IdleSinceHint=1738590621626705
IdleSinceHintMonotonic=20501509757
LockedHint=no
root@s10:~# loginctl seat-status seat0
seat0
        Sessions: *17
         Devices:
                  ├─/sys/devices/LNXSYSTM:00/LNXPWRBN:00/input/input2
                  │ input:input2 "Power Button"
                  ├─/sys/devices/LNXSYSTM:00/LNXSLPBN:00/input/input4
                  │ input:input4 "Sleep Button"

├─/sys/devices/LNXSYSTM:00/LNXSYBUS:00/PNP0A03:00/LNXVIDEO:00/input/>
                  │ input:input3 "Video Bus"

├─/sys/devices/pci0000:00/0000:00:01.1/ata2/host2/target2:0:0/2:0:0:>
                  │ block:sr0

├─/sys/devices/pci0000:00/0000:00:01.1/ata2/host2/target2:0:0/2:0:0:>
                  │ scsi_generic:sg0
                  ├─/sys/devices/pci0000:00/0000:00:02.0/drm/card0
                  │ [MASTER] drm:card0
                  │ ├─/sys/devices/pci0000:00/0000:00:02.0/drm/card0/card0-
Virtual-1
```

```
                    │ │ [MASTER] drm:card0-Virtual-1
                    │ ├/sys/devices/pci0000:00/0000:00:02.0/drm/card0/card0-
Virtual-2
                    │ │ [MASTER] drm:card0-Virtual-2
                    │ ├/sys/devices/pci0000:00/0000:00:02.0/drm/card0/card0-
Virtual-3
                    │ │ [MASTER] drm:card0-Virtual-3
                    │ ├/sys/devices/pci0000:00/0000:00:02.0/drm/card0/card0-
Virtual-4
                    │ │ [MASTER] drm:card0-Virtual-4
                    │ ├/sys/devices/pci0000:00/0000:00:02.0/drm/card0/card0-
Virtual-5
                    │ │ [MASTER] drm:card0-Virtual-5
                    │ ├/sys/devices/pci0000:00/0000:00:02.0/drm/card0/card0-
Virtual-6
                    │ │ [MASTER] drm:card0-Virtual-6
                    │ ├/sys/devices/pci0000:00/0000:00:02.0/drm/card0/card0-
Virtual-7
                    │ │ [MASTER] drm:card0-Virtual-7
                    │ └/sys/devices/pci0000:00/0000:00:02.0/drm/card0/card0-
Virtual-8
                    │   [MASTER] drm:card0-Virtual-8
                    ├/sys/devices/pci0000:00/0000:00:02.0/drm/renderD128
                    │ drm:renderD128
                    ├/sys/devices/pci0000:00/0000:00:02.0/graphics/fb0
                    │ graphics:fb0 "vmwgfxdrmfb"
                    ├/sys/devices/pci0000:00/0000:00:04.0/input/input8
                    │ input:input8 "VirtualBox mouse integration"
                    ├/sys/devices/pci0000:00/0000:00:05.0/sound/card0
                    │ sound:card0 "I82801AAICH"
                    ├/sys/devices/pci0000:00/0000:00:06.0/usb2
                    │ usb:usb2
                    │ └/sys/devices/pci0000:00/0000:00:06.0/usb2/2-1
                    │   usb:2-1
                    │   └/sys/devices/pci0000:00/0000:00:06.0/usb2/2-1/2-
1:1.0/0003:80E>
                    │     input:input6 "VirtualBox USB Tablet"
                    ├/sys/devices/pci0000:00/0000:00:0b.0/usb1
                    │ usb:usb1
                    ├/sys/devices/platform/i8042/serio0/input/input0
                    │ input:input0 "AT Translated Set 2 keyboard"
                    │
├/sys/devices/platform/i8042/serio0/input/input0/input0::capslock
                    │ │ leds:input0::capslock
                    │
├/sys/devices/platform/i8042/serio0/input/input0/input0::numlock
                    │ │ leds:input0::numlock
                    │
└/sys/devices/platform/i8042/serio0/input/input0/input0::scrolllo>
                    │   leds:input0::scrolllock
                    ├/sys/devices/platform/i8042/serio1/input/input5
                    │ input:input5 "ImExPS/2 Generic Explorer Mouse"
                    ├/sys/devices/platform/pcspkr/input/input7
                    │ input:input7 "PC Speaker"
                    └/sys/devices/virtual/misc/rfkill
                      misc:rfkill
root@s10:~# loginctl show-seat seat0
Id=seat0
ActiveSession=17
```

```
CanTTY=yes
CanGraphical=yes
Sessions=17
IdleHint=no
IdleSinceHint=1738590621626705
IdleSinceHintMonotonic=20501509757
```

Пример: блокировка и разблокирование сеанса

```
root@s10:~# loginctl lock-session 17
root@s10:~# loginctl show-session 17 | grep IdleHint
IdleHint=yes
root@s10:~# loginctl unlock-session 17
root@s10:~# loginctl show-session 17 | grep IdleHint
IdleHint=no
```

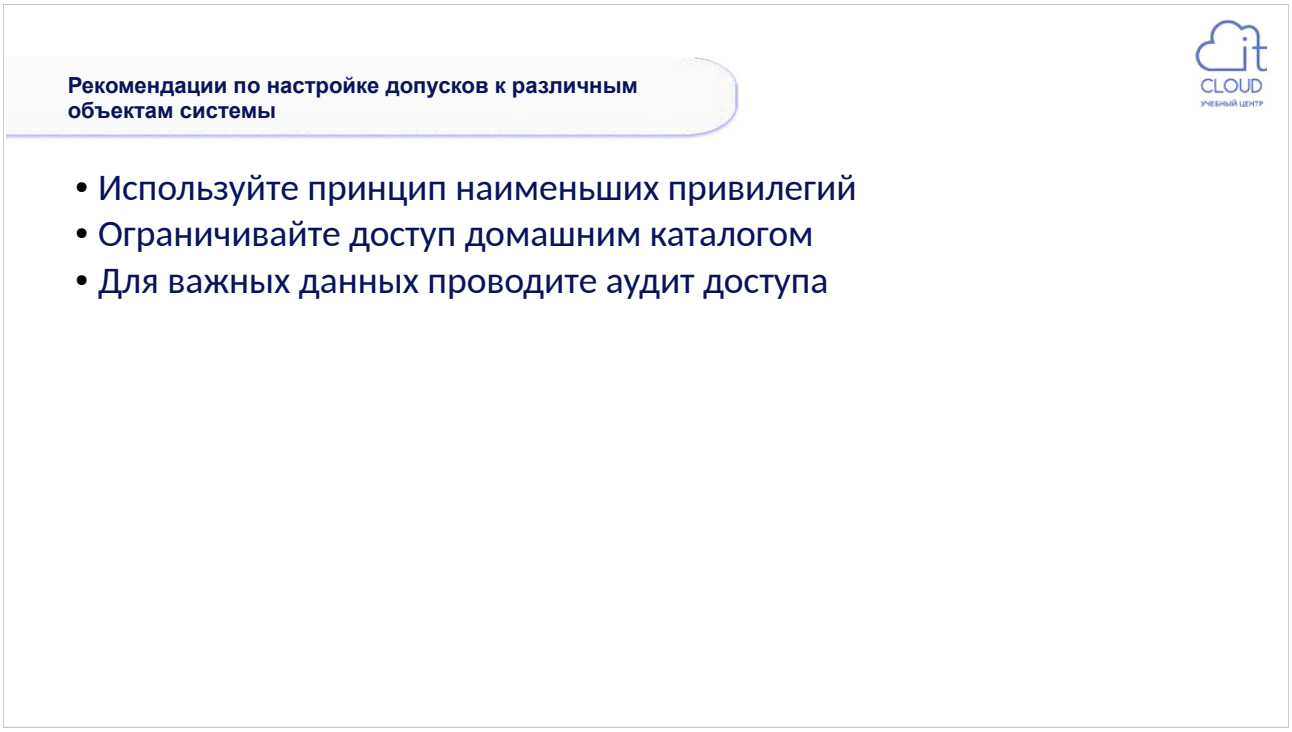
Пример: завершение сеанса

```
root@s10:~# loginctl session-status 17 | grep State
         State: active
root@s10:~# loginctl terminate-session 17 ; loginctl session-status 17 | grep
State
         State: closing
root@s10:~# loginctl session-status 17
17 - sa (1000)
           Since: Mon 2025-02-03 18:05:53 +05; 59min ago
          Leader: 3300 (lightdm)
            Seat: seat0; vc7
         Display: :0
         Service: lightdm; type x11; class user
         Desktop: lightdm-xsession
           State: closing
            Unit: session-17.scope
                  ├─3300 lightdm --session-child 13 24
                  └─3310 x-session-manager

фев 03 18:05:53 s10 systemd[1]: Started session-17.scope - Session 17 of User
sa.
фев 03 18:05:55 s10 x-session-manager[3310]: WARNING: Unable to find provider ''
of re>
фев 03 18:05:55 s10 gnome-keyring-daemon[3425]: discover_other_daemon: 1
фев 03 18:05:56 s10 gnome-keyring-daemon[3501]: discover_other_daemon: 1
фев 03 18:05:56 s10 gnome-keyring-daemon[3506]: discover_other_daemon: 1
фев 03 18:05:57 s10 gnome-keyring-daemon[3547]: discover_other_daemon: 1
фев 03 19:05:30 s10 systemd[1]: Stopping session-17.scope - Session 17 of User
sa...
фев 03 19:05:31 s10 x-session-manager[3310]: WARNING: Detected that screensaver
has le
```

Если сеанс по какой-то причине не заканчивается, то вы можете его уничтожить командой `loginctl kill-session`.

Глава 5. Доступ к файлам.

5.1 Рекомендации по настройке допусков к различным объектам системы.



При предоставлении доступа к ресурсам следует придерживаться принципа наименьших привилегий. Для этого необходимо:

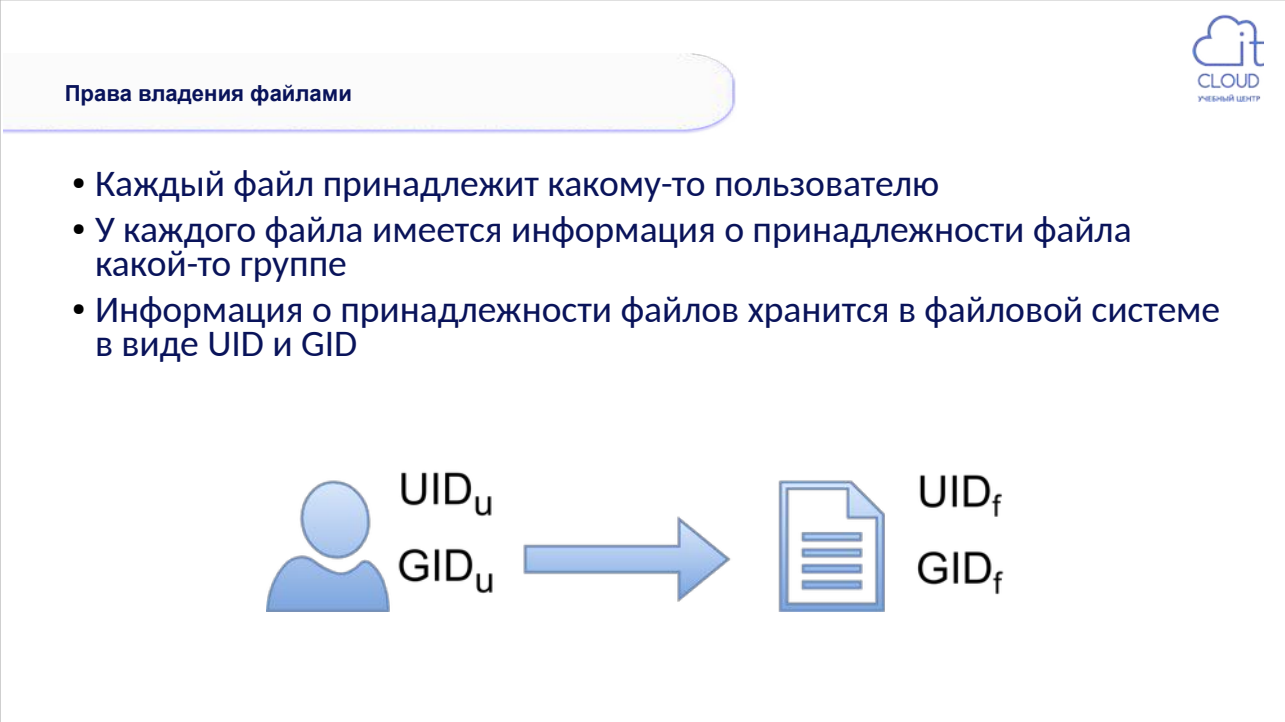
- Оценить круг задач, которые решает пользователь.
- Выяснить какие объекты системы ему необходимы для решения его задач.
- Предоставить минимально необходимые разрешения на доступ к нужным объектам.

Используя принцип наименьших привилегий мы минимизируем потенциальный ущерб от возможных уязвимостей системы. Минимизируем усилия по решению проблем и выяснению обстоятельств произошедших инцидентов.

В операционных системах Linux подразумевается выполнение всех действий пользователя в своем домашнем каталоге. Не следует пренебрегать этим принципом. Например, если пользователь запустит вредоносный код, то ущерб от него распространится только на данные в его домашнем каталоге.

При предоставлении доступа к важным ресурсам важно проводить аудит доступа. Аудит включает в себя протоколирование действий пользователя и постоянный анализ журналов аудита.

5.2 Права доступа к файловым ресурсам.



Каждый файл имеет два идентификатора определяющего его принадлежность – владелец файла и группа пользователей.

Эта информация сохраняется не в самом файле, а в его метаданных (inode).

Любой файл всегда принадлежит одному единственному пользователю. Этот пользователь называется владельцем (или пользователем - user) файла.

Когда любой пользователь системы создает файл, то права владения этим файлом принадлежат именно этому пользователю.

Первичная группа пользователя (GID) в обычных условиях определяет группу пользователей, которая будет установлена для вновь создаваемого файла.

Права доступа к файлам могут быть определены с помощью команды `ls -l`.

Пример:

```
user@s10:~$ id
uid=1000(user) gid=1000(user) groups=1000(user),10(wheel),100(users)
context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
user@s10:~$ > file
user@s10:~$ ls-l file
-bash: ls-l: command not found
user@s10:~$ ls -l file
-rw-rw-r--. 1 user user 0 Jul 28 09:36 file
```

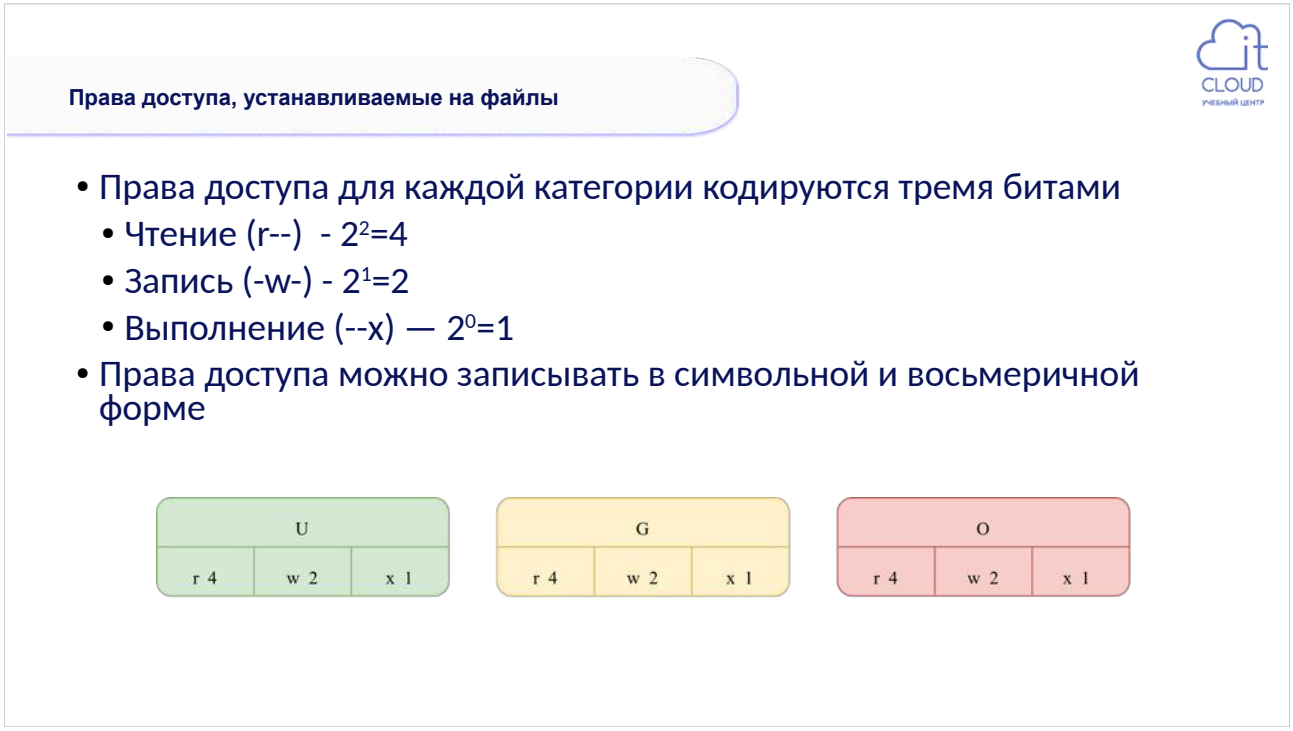
В реальности, в файловой системе сохраняется информация не об именах пользователя и группы, а UID и GID пользователя, создавшего файл.

В Linux существуют три базовых класса доступа к файлу:

- User access (u) – права доступа владельца файла;
- Group access (g) – права доступа группы владельцев файла;
- Other access (o) – права доступа для всех остальных.

Никаких других категорий не предусматривается, поэтому каждый пользователь может быть либо владельцем файла, либо входить в группу пользователей, либо относиться к категории всех остальных. Сами по себе права владения файлами не предоставляют информации о том, что может пользователь делать с данным файлом.

Система прав доступа к файлу, описываемая ниже, определяет какие возможности работы с файлом имеет конкретный пользователь системы.



Права доступа к файлу хранятся в метаданных файла и кодируются тремя триадами бит.

Права доступа можно задавать в символической и восьмеричной нотациях.

Символическая нотация основана на буквенных обозначениях прав владения и прав доступа, а восьмеричная связана с фактическим представлением этих прав в виде триад бит.

Порядок триад:

1. старшая триада соответствует правам доступа владельца файла (u);
2. средняя триада - правам доступа группы владельцев (g);
3. младшая триада – правам доступа всех остальных пользователей (o).

Порядок битов в триадах:

1. установленный в 1 старший бит в каждой триаде (4 в восьмеричной нотации) обозначает разрешение на чтение данного файла и в символической нотации обозначается r--;
2. установленный в 1 средний бит в каждой триаде (2 в восьмеричной нотации) обозначает разрешение на изменение данного файла: -w-;
3. установленный в 1 младший бит в каждой триаде (1 в восьмеричной нотации) обозначает разрешение на исполнение данного файла: --x.

u			**g**			**o**		
r	**w**	**x**	**r**	**w**	**x**	**r**	**w**	**x**
2^2	2^1	2^0	2^2	2^1	2^0	2^2	2^1	2^0

Пример: запись rxwxr-x--x (751) обозначает, что пользователь файла имеет все права на доступ к нему (rxw или 7), группа пользователей имеет права на чтение и исполнение файла (r-x или 5), все остальные имеют права на исполнение файла (--x или 1).

Символьная и восьмеричная нотация записи прав доступа абсолютно эквивалентны.

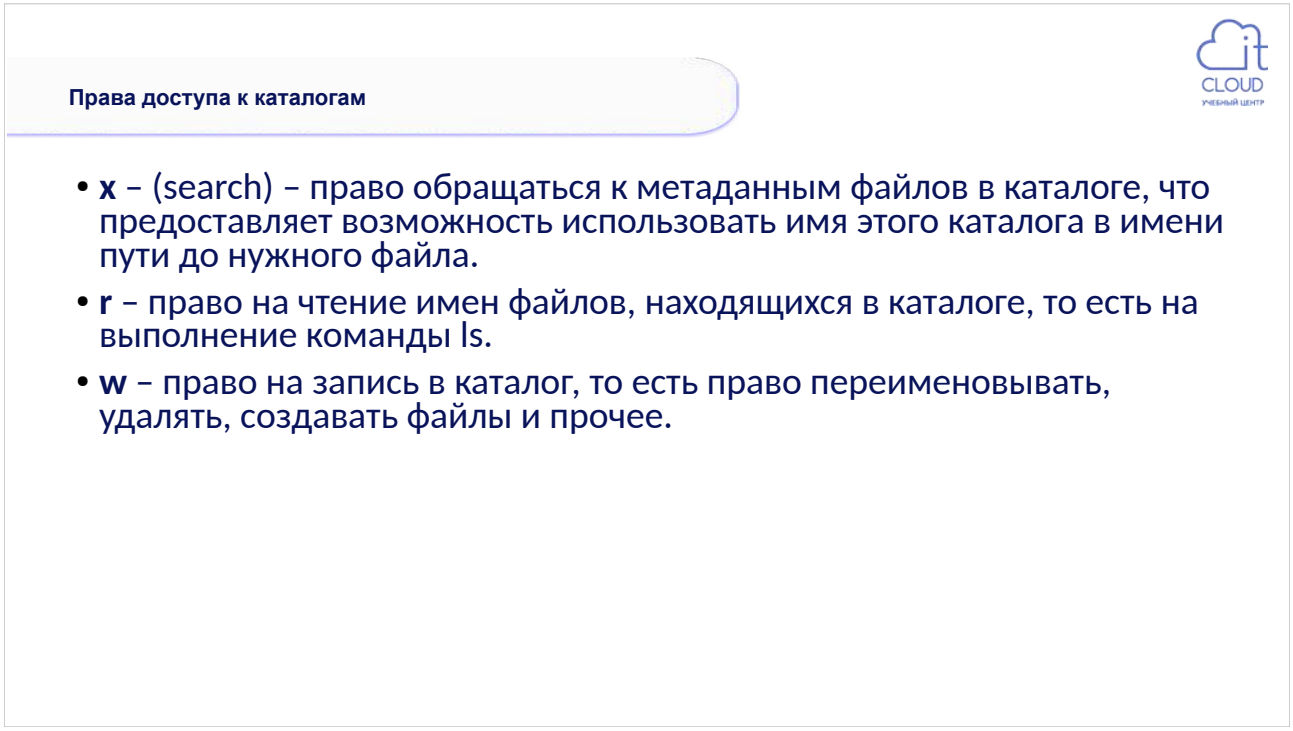
Восьмеричное значение триады бит получается сложением степеней двойки, соответствующих номеру бита в триаде.

Пример: права доступа в символьной нотации rwxr-xr-- в восьмеричной нотации записываются как 754, где $7 = 2^2+2^1+2^0$, $5 = 2^2+2^0$, $4 = 2^2$.

Для того чтобы увидеть права доступа к файлу, достаточно набрать команду `ls -l`, при этом права доступа к файлам выводятся в первой колонке.

Пример:

```
user@s10:~$ ls -l file
-rw-rw-r--. 1 user user 0 Jul 28 09:36 file
```



Права доступа, устанавливаемые на каталоги имеют несколько иной смысл, чем права на файлы.

Для каталогов используются следующие права:

x – (search) – право обращаться к метаданным файлов в каталоге, что предоставляет возможность использовать имя этого каталога в имени пути до нужного файла.

r – право на чтение имен файлов, находящихся в каталоге, то есть на выполнение команды ls.

w – право на запись в каталог, то есть право переименовывать, удалять, создавать файлы и прочее.

Без наличия права на search (x), установленного на каталог, работа с находящимися внутри файлами невозможна. Поэтому, *для каталогов права доступа должны быть либо нечетные, либо они должны отсутствовать*.

Ниже приведены права доступа к каталогам, которые имеют практический смысл:

- 0 (---) – прав нет.
- 1 (--x) – имеется право перехода в каталог, можно обращаться к файлам в нем, однако, нельзя выполнять команду ls и осуществлять какие-либо манипуляции с файлами, например, переименование или удаление.
- 3 (-wx) - в каталог можно переходить, разрешены любые манипуляции с файлами и можно к ним обращаться, однако получить список файлов командой ls невозможно.
- 5 (r-x) - в каталог можно переходить и получать подробную информацию о файлах командой ls -l, разрешено обращаться к файлам, однако, нельзя осуществлять какие-либо манипуляции с файлами, например, переименование или удаление.

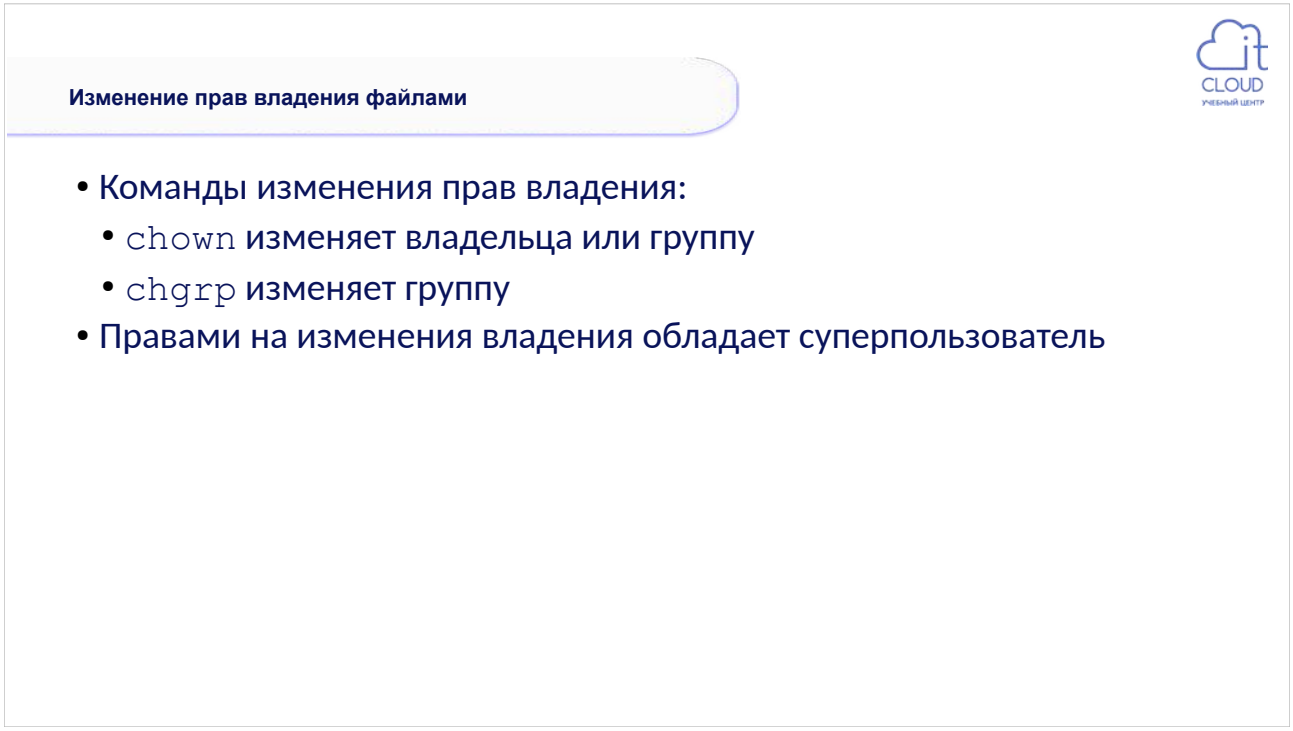
- 7 (rwx) – полные права.

  Для получения прав доступа к каталогу следует использовать команду `ls -ld`.

  Пример:

```
user@s10:~$ ls -ld dir
drwxr-x--x. 2 user adm 6 Jul 28 10:14 dir
```

***Примечание**: В примере, приведенном выше, на каталог dir установлены права 751, то есть владелец этого каталога (user) имеет все права на этот каталог, группа (adm) не имеет прав переименовывать и удалять файлы, поскольку не имеет прав на запись, а для всех остальных каталог является "темным". Все остальные могут переходить в этот каталог и могут обращаться к файлам, находящихся в нем, однако, при этом они должны знать, какие имена имеют файлы, к которым им необходим доступ. Это связано с тем, что командой ls остальные пользоваться не могут, так как прав на чтение каталога нет. Производить какие-либо манипуляции с файлами они также не имеют права, так как права на запись в каталог нет.*



Права владения файлами могут быть изменены с помощью следующих команд:

- `chown` - эта команда позволяет менять как владельца файла или каталога, так и группу пользователей файла;
- `chgrp` - позволяет менять группу пользователей файла.

В Linux этим команды обычно может выполнять только суперпользователь, так как передача прав владения, разрешенная для обычных пользователей, представляет собой существенную угрозу безопасности.

При необходимости разрешить какому-либо уполномоченному пользователю исполнять эти команды, на них необходимо установить специальный бит (например, SUID - бит подмены владельца процесса), о которых будет рассказано в конце этой главы. Однако, даже не смотря на возможность для обычного пользователя с помощью такой манипуляции изменять права владения файлами, обычный пользователь может менять владельца или группу только у тех файлов, которыми он владеет.

Пример: Приведенная ниже команда меняет владельца файла:

```
user@s10:~$ sudo chown tempuser file
[sudo] password for user:
user@s10:~$ ls -l file
-rw-rw-r--. 1 tempuser user 0 Jul 28 09:36 file
```

Пример: Ниже приведен пример смены группы пользователей файлов f1 и text.c :

```
user@s10:~$ ls -l f1 text.c
-rw-rw-r--. 1 user user 0 Jul 28 10:32 f1
-rw-rw-r--. 1 user user 5 Jul 28 10:32 text.c
user@s10:~$ sudo chgrp adm f1 text.c
user@s10:~$ ls -l f1 text.c
```

```
-rw-rw-r--. 1 user adm 0 Jul 28 10:32 f1
-rw-rw-r--. 1 user adm 5 Jul 28 10:32 text.c
```

С помощью команды `chown` можно одновременно изменить владельца и группу (через двоеточие или точку) одновременно, причем новый владелец вовсе не обязан быть членом той группы, которая будет установлена на файл

Пример:

```
user@s10:~$ sudo chown root.disk f1
user@s10:~$ ls -l f1
-rw-rw-r--. 1 root disk 0 Jul 28 10:32 f1
```

Пример:

```
user@s10:~$ sudo chown :adm f1
user@s10:~$ ls -l f1
-rw-rw-r--. 1 root adm 0 Jul 28 10:32 f1
```

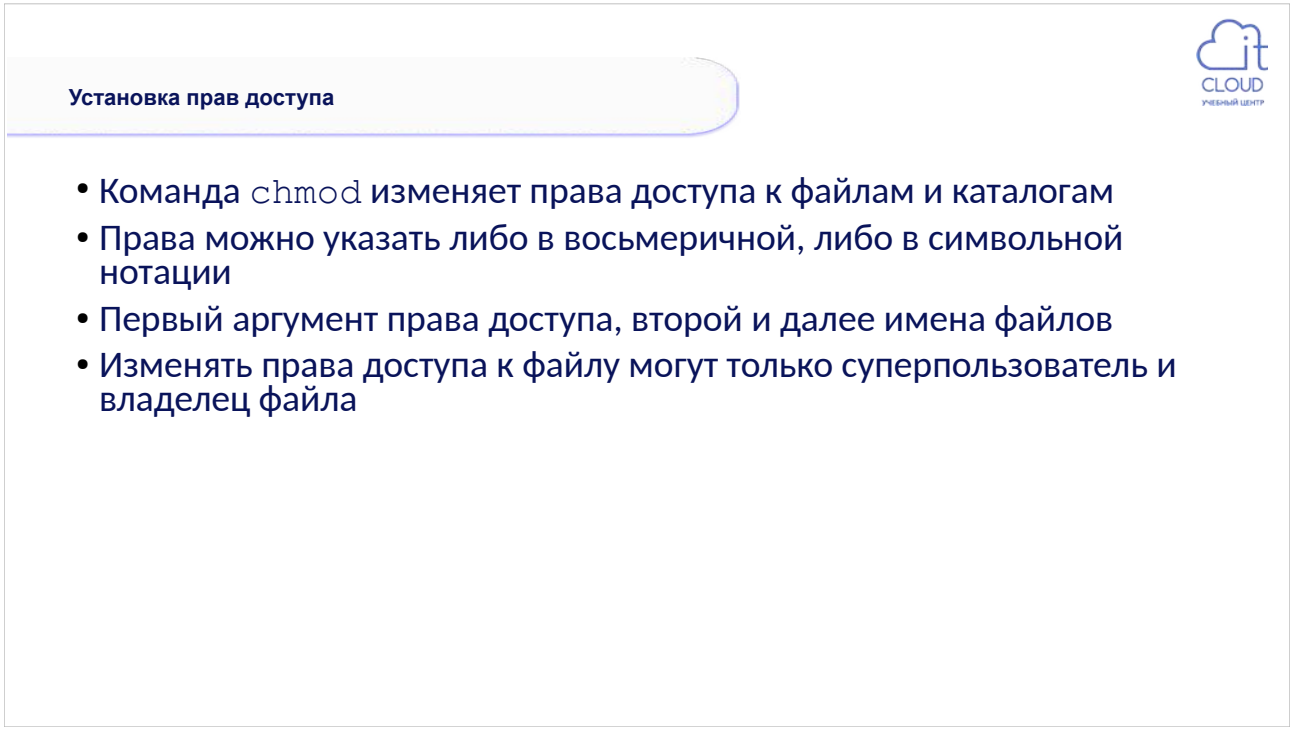
Опция `-c` GNU версий команд `chown` и `chgrp` позволяет получать подробную информацию об изменяемых правах владения. Тоже делает и опция `-v`.

Обе команды `chown` и `chgrp` имеют опцию `-R`, позволяющую рекурсивно изменять права владения на каталоги и их содержимое.

Пример:

```
user@s10:~$ ls -Rl dir/
dir/:
total 0
-rw-rw-r--. 1 user user 0 Jul 28 10:36 f1
-rw-rw-r--. 1 user user 0 Jul 28 10:37 f2
user@s10:~$ sudo chown -cR root dir
changed ownership of ‘dir/f1’ from user to root
changed ownership of ‘dir/f2’ from user to root
changed ownership of ‘dir’ from user to root
user@s10:~$ ls -Rl dir/
ls: cannot open directory dir/: Permission denied
user@s10:~$ sudo ls -Rl dir/
dir/:
total 0
-rw-rw-r--. 1 root user 0 Jul 28 10:36 f1
-rw-rw-r--. 1 root user 0 Jul 28 10:37 f2
```

Внимание! *Неосторожное использование команд `chown` и `chgrp`, особенно с опцией `-R`, может привести к выводу всей системы из строя!*



Команда `chmod` предназначена для изменения прав доступа к файлам и каталогам, указанным в качестве аргументов.

Права доступа должны быть указаны либо в восьмеричной, либо в символьной нотации.

Права указывают в качестве первого аргумента команды. Второй и последующие имена файлов.

Изменять права доступа к файлу могут только суперпользователь и владелец файла.

Пример: Ниже приведен пример использования команды chmod для изменения прав доступа к файлу в восьмеричной нотации:

```
user@s10:~$ ls -l text.c
-rw-rw-r--. 1 user adm 5 Jul 28 10:32 text.c
user@s10:~$ chmod 640 text.c
user@s10:~$ ls -l text.c
-rw-r-----. 1 user adm 5 Jul 28 10:32 text.c
```

Команда `chmod` позволяет также устанавливать права на доступ к файлу, указывая их в символической нотации. Для этого применяется следующая форма команды

`chmod` *`класс_изменение_права файлы`*

- класс может принимать следующие значения:
  - u – доступ владельца;
  - g – доступ группы владельцев;
  - o – доступ всех остальных;
  - a – доступ всех групп пользователей.

- изменение может принимать следующие значения:
  - + - разрешить;
  - - - запретить;
  - = - установить.
- Права может принимать следующие значения:
  - r – чтение;
  - w – запись;
  - x – выполнение.

Если используются операции разрешения (+) или запрета (-) прав на файл, то они не изменяют те биты прав доступа, которые не относятся к требуемому изменению.

Пример: если для файла f1 требуется удалить право на изменение для группы, и добавить право на чтение для всех трех категорий пользователей (владелец, группа, все остальные), то выполним следующую команду:

```
user@s10:~$ ls -l f1
-rw-rw----. 1 user adm 0 Jul 28 10:32 f1
user@s10:~$ chmod g-w,o+r f1
user@s10:~$ ls -l f1
-rw-r--r--. 1 user adm 0 Jul 28 10:32 f1
```

Использование операции назначения (=) стирает те права, которые были установлены ранее и назначает новые.

Пример: установим на файл f1 права на чтение и запись для владельца и группы, и запретим всем остальным какой-либо доступ к файлу:

```
user@s10:~$ chmod ug=rw,o= f1
user@s10:~$ ls -l f1
-rw-rw----. 1 user adm 0 Jul 28 10:32 f1
```

Аналогично командам `chown` и `chgrp` , команда `chmod` способна рекурсивно изменять права доступа к каталогам и всему их содержимому, если она вызвана с опцией `-R`.

Этой возможностью следует пользоваться с особой осторожностью, принимая во внимание концептуальные отличия прав на файлы от прав на каталоги - на файлы в большинстве случаев устанавливаются четные права (отсутствие прав на исполнение), а на каталоги наоборот - нечетные (без права на search каталоги не позволят обращаться к метаданным файлов внутри них).

Пример: Приведенная ниже команда снимает права на запись для каталога dir :

```
user@s10:~$ ls -lR dir
dir:
total 0
-rw-rw-r--. 1 user adm 0 Jul 28 10:36 f1
-rw-rw-r--. 1 user adm 0 Jul 28 10:37 f2
user@s10:~$ chmod -R g-w dir
user@s10:~$ ls -lR dir
```

```
dir:
total 0
-rw-r--r--. 1 user adm 0 Jul 28 10:36 f1
-rw-r--r--. 1 user adm 0 Jul 28 10:37 f2
```

В обычной практике права на каталоги и на файлы устанавливаются отдельно. При необходимости рекурсивного изменения прав на каталог и его содержимое опцию `-R` команды `chmod` обычно не используют. Вместо этого пользуются командой `find` с установкой `-exec chmod` (или `xargs chmod`).

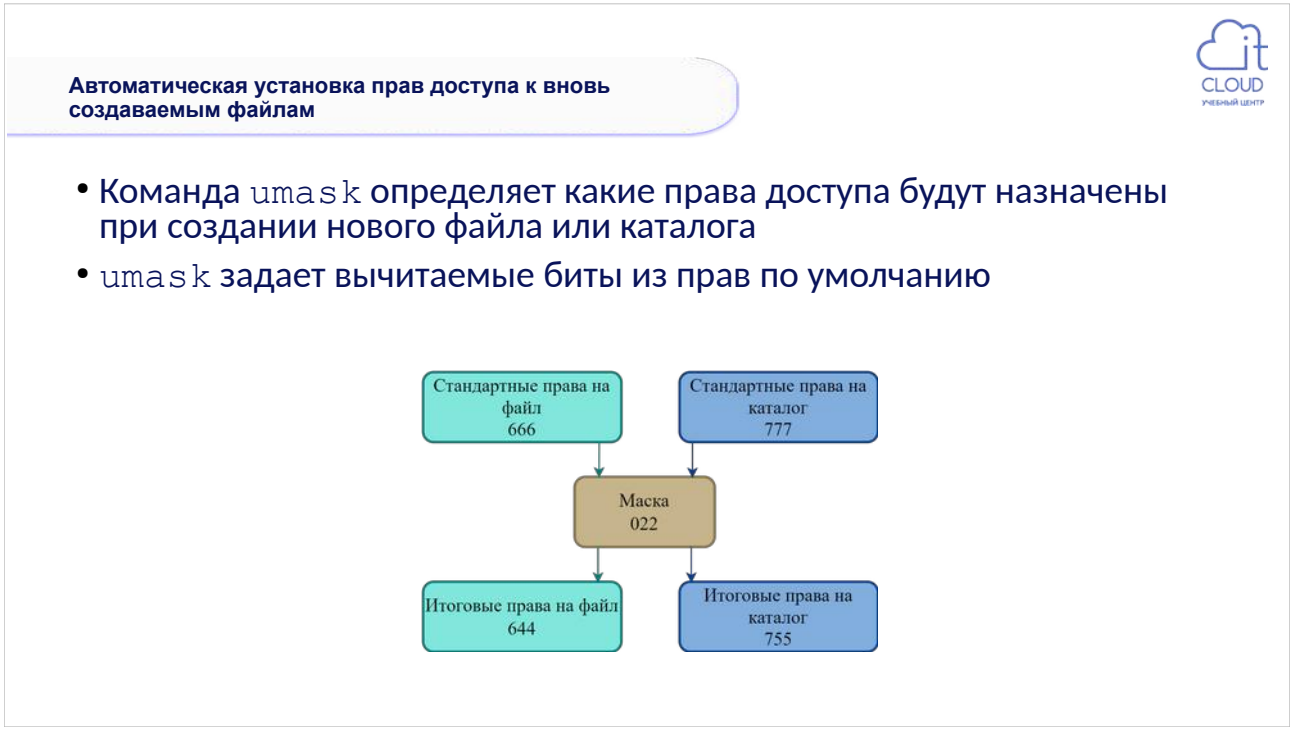
Пример: в каталоге dir требуется установить для файлов права 644, не затрагивая при этом права на каталоги:

```
user@s10:~$ find dir -type f -exec chmod -c 664 {} \;
mode of ‘dir/f1’ changed from 0644 (rw-r--r--) to 0664 (rw-rw-r--)
mode of ‘dir/f2’ changed from 0644 (rw-r--r--) to 0664 (rw-rw-r--)
```

GNU версия команды chmod позволяет использовать опцию -v для получения информации о файлах, права доступа к которым изменяются, и опцию -c для получения подробностей изменения прав.

Опция -v выдает подробную информацию всегда, а -c только тогда, когда права действительно изменяются.

```
user@s10:~$ chmod -v 664 f1
mode of ‘f1’ changed from 0660 (rw-rw----) to 0664 (rw-rw-r--)
user@s10:~$ chmod -v 664 f1
mode of ‘f1’ retained as 0664 (rw-rw-r--)
user@s10:~$ chmod -c 664 f1
user@s10:~$
```



Команда `umask` предназначена для автоматической установки прав доступа к вновь создаваемым файлам и каталогам.

Команда `umask` позволяет задавать значение битовой маски, которая будет “вычитаться” из прав 777 для каталогов и 666 для файлов.

При вызове этой команды без аргумента она возвратит текущее значение маски

Пример:

```
user@s10:~$ umask
0002
```

Установка другого значения `umask` никоим образом не отразится на уже существующих файлах и каталогах, она участвует только в процессе определения прав на вновь создаваемые файлы и каталоги.

Пример:

```
user@s10:~$ umask
0002
user@s10:~$ mkdir dir1
user@s10:~$ touch file1
user@s10:~$ ls -ld dir1 file1
drwxrwxr-x. 2 user user 6 Jul 28 20:21 dir1
-rw-rw-r--. 1 user user 0 Jul 28 20:21 file1
user@s10:~$ umask 077
user@s10:~$ mkdir dir2
user@s10:~$ touch file2
user@s10:~$ ls -ld dir2 file2
drwx------. 2 user user 6 Jul 28 20:22 dir2
-rw-------. 1 user user 0 Jul 28 20:22 file2
```

Ниже приведена таблица наиболее часто применяемых значений `umask`.

umask	002	007	022	027	077
Каталоги	775	770	755	750	700
Файлы	664	660	644	640	600

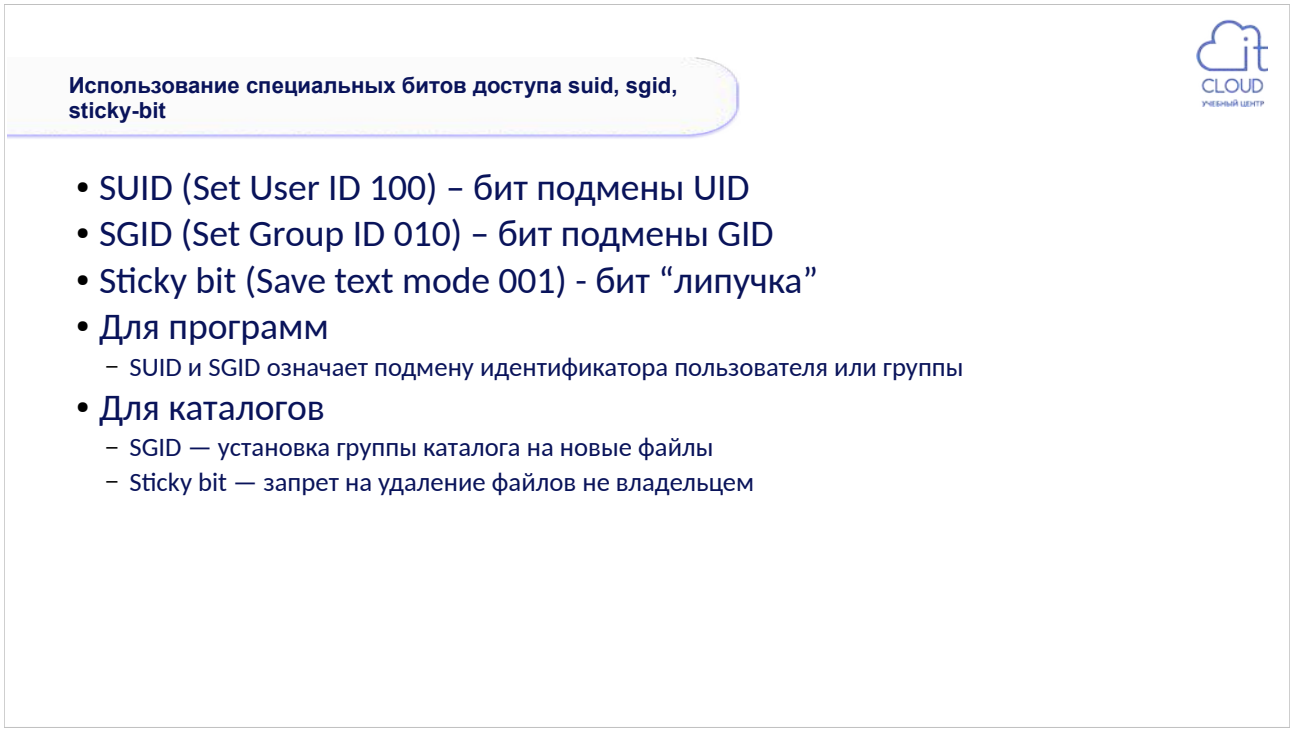
Значение `umask` можно задавать также и в символьной нотации

Пример:

```
user@s10:~$ umask u=rwx,g=rx,o=
user@s10:~$ umask
0027
```

При задании значения `umask` в символьной нотации требуется указать в качестве аргумента права, которые должны будут иметь новые каталоги.

5.3 Использование специальных битов доступа suid, sgid, sticky-bit.



Помимо битов, устанавливающих разрешения на доступ к файлу, существуют специальные атрибуты, которых образуют еще одна триаду битов:

- Sticky bit (Save text mode) - бит “липучка”;
- SUID (Set User ID) – бит подмены UID;
- SGID (Set Group ID) – бит подмены GID.

Sticky bit кодируется восьмеричной 1 (двоичная 001), SGID кодируется восьмеричной 2 (010), а SUID - 4 (100).

В символьной нотации применяются символы T для Sticky bit, S для SUID и SGID. Эти символы всегда выводятся в позиции, где должен находиться флаг разрешения на исполнение (x).

Если одновременно установлены и бит x и бит S, то отображается символ s

Биты T и x всегда устанавливаются вместе, поскольку бит T используется только для каталогов. Поэтому в символьном отображении прав доступа должен встречается только символ t

- SUID отображается в виде буквы s или S в старшей триаде бит, отображающей права владельца
- SGID отображается в виде буквы s или S в средней триаде бит, отображающей права группы

- Sticky bit отображается в виде буквы t в младшей триаде бит, отображающей права для всех остальных.

Пример:

```
user@s10:~$ ls -ld /bin/passwd /tmp
-rwsr-xr-x. 1 root root 27832 Jun 10  2014 /bin/passwd
drwxrwxrwt. 7 root root  4096 Jul 28 20:35 /tmp
```

Приведенный выше пример демонстрирует, что на файл системной команды `passwd` установлен бит SUID (символ **`s`** в старшей триаде вместо прав на исполнение), а на каталог `/tmp` установлен Sticky bit (символ **`t`** в триаде бит для прав всех остальных).

Атрибут Sticky bit для файлов не используется, в ранних версиях UNIX он был предназначен для того, чтобы оставить в памяти образ программы (Save text mode).

Процесс наследует права доступа к системным ресурсам от пользователя (UID), запустившего процесс, и его первичной группы (GID), если для исполняемых файлов, не установлены биты SUID и/или SGID

При установленном на исполняемый файл бите SUID процесс исполняется не от имени пользователя, запустившего его, а от имени владельца исполняемого файла команды.

При установленном бите SGID процесс исполняется не от имени первичной группы пользователя, запустившего процесс, а от имени группы пользователей файла.

У каждого процесса имеется четыре идентификатора:

1. RUID - Real UID, который всегда равен UID пользователя, выполнившего команду.
2. RGID - Real GID, который всегда равен GID пользователя, выполнившего команду.
3. EUID - Effective UID, который либо равен RUID, либо если на исполняемый файл установлен бит SUID, то UID владельца файла.
4. EGID - Effective GID, который либо равен RGID, либо если на исполняемый файл установлен бит SGID, то GID владельца файла.

В подавляющем большинстве случаев подмена владельца или группы осуществляется на root или какого-либо высоко привилегированного пользователя или группу. Например, при выполнении команды `passwd` (см. пример выше), несмотря на то, что ее запустил обычный пользователь, она будет исполняться от имени root, так как он владеет ее исполняемым файлом. Программа passwd требует временного предоставления доступа обычному пользователю к тем ресурсам, к которым он не имеет доступа. Естественно, такие программы требуют особого подхода к разработке, так как предоставляют серьезную угрозу для безопасности системы.

На файлы скриптов Shell биты SUID и SGID устанавливать можно, но они действовать не будут.

Установка Sticky bit на каталог, в отношении которого пользователь имеет права на чтение и на запись, позволяет запретить удалять и изменять пользователю чужие файлы в этом каталоге.

Это используется при установке прав доступа к каталогу `/tmp`, открытому на запись всем, поскольку иначе пользователь может удалить чужие временные файлы, находящиеся в этом каталоге, что может повлечь плачевные последствия.

При установке атрибута SGID на каталог, вновь созданные файлы в этом каталоге будут наследовать группу владельцев по группе владельцев каталога (так называемый “стиль BSD”), вместо RGID процесса, создающего файл по версии System V.

Пример:

```
user@s10:~$ ls -ld dir
drwxr-s--x. 2 user adm 24 Jul 28 10:37 dir
user@s10:~$ id
uid=1000(user) gid=1000(user) groups=1000(user),10(wheel),100(users)
context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
user@s10:~$ cd dir
[user@sl0 dir]$ touch file1
[user@sl0 dir]$ ls -l file1
-rw-rw-r--. 1 user adm 0 Jul 31 15:50 file1
[user@sl0 dir]$ mkdir subdir
[user@sl0 dir]$ ls -ld subdir/
drwxrwsr-x. 2 user adm 6 Jul 31 15:52 subdir/
```

***Примечание**: В каталоге, на который установлен бит SGID, был создан файл. При этом группа владельцев файла была назначена не по первичной группе пользователя, создавшего файл, а по группе пользователей каталога, в котором файл был создан. Обратите внимание, что для вновь создаваемых каталогов бит SGID наследуется.*

Команда `chmod` позволяет установить особые биты доступа на файлы и каталоги.

Для установки специальных битов в символьном режиме команда chmod должна быть выполнена со следующими аргументами:

- **`u+s`** - для установки на файл бита SUID;
- **`g+s`** - для установки на файл или каталог бита SGID;
- **`o+t`** - для установки на каталог бита Sticky bit.

Для установки специальных битов в числовом режиме команде chmod в качестве первого аргумента передается число, состоящее из 4 цифр, где левая цифра представляет собой сумму специальных битов.

Пример: Приведенная ниже команда chmod 2775 d1 устанавливает бит SGID на каталог:

```
user@s10:~$ mkdir d1
user@s10:~$ ls -ld d1
drwxrwxr-x. 2 user user 6 Jul 31 15:55 d1
user@s10:~$ chmod 2775 d1
user@s10:~$ ls -ld d1
drwxrwsr-x. 2 user user 6 Jul 31 15:55 d1
```

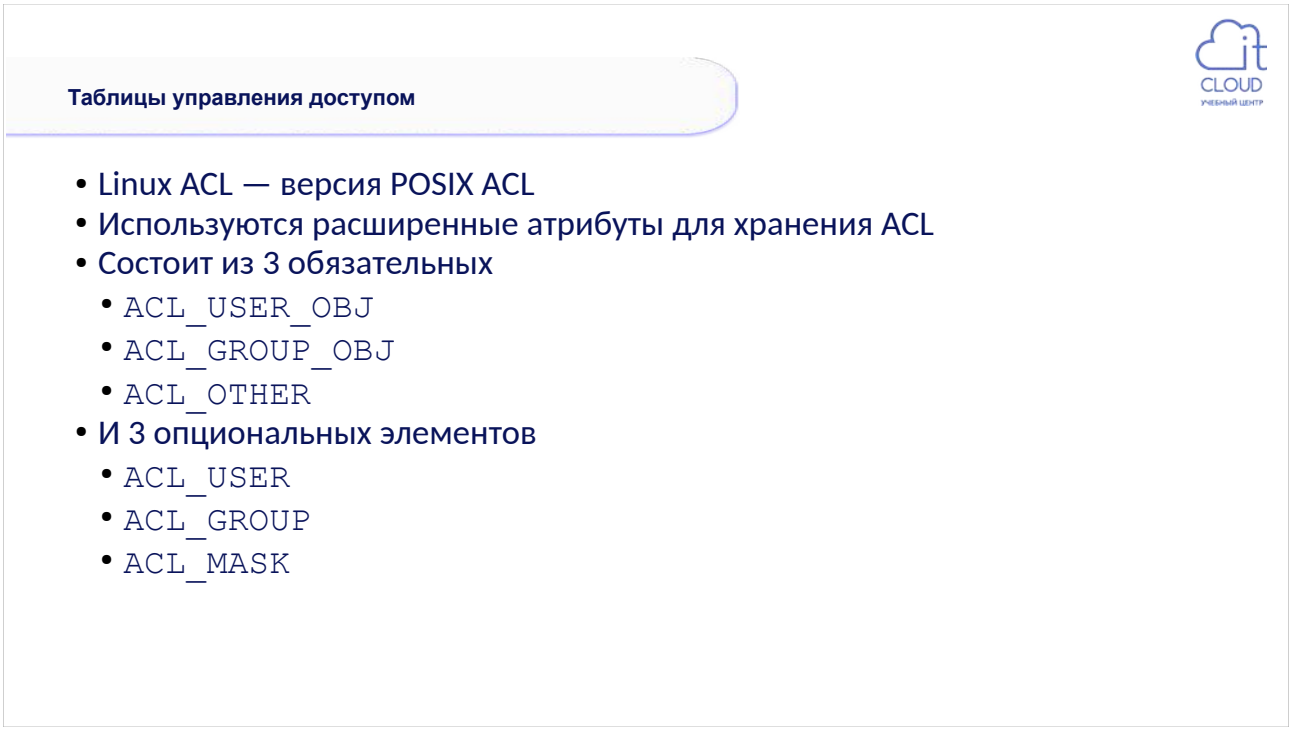
Эффекты специальных битов

Права	Эффект для каталогов	Эффект для файлов
-rws--x--x	-	Команда исполняется от имени владельца файла
-rwx--s--x	-	Команда исполняется от имени группы пользователей файла
drwxrws---	На файлы, создаваемые в каталоге, будет установлена такая же группа, как у каталога	-
drwxrwxrwt	В каталоге можно удалять или переименовывать только собственные файлы	-

Ниже приведена таблица, в которой указаны результаты установки различных специальных прав доступа на файлы и каталоги.

Права	Эффект для каталогов	Эффект для файлов
-rws--x--x	-	Команда исполняется от имени владельца файла
-rwx--s--x	-	Команда исполняется от имени группы пользователей файла
drwxrws---	На файлы, создаваемые в каталоге, будет установлена такая же группа, как у каталога	-
drwxrwxrwt	В каталоге можно удалять или переименовывать только собственные файлы	-

5.4 Списки управления доступом (ACL)



Стандартная система доступа к файлам Linux-систем предусматривает 12 бит, которые определяют вид доступа для трех категорий пользователей: владельца, группы владельцев и всех остальных.

Кроме этого в такой системе нет наследования прав доступа.

Такая система хорошо работает в простых ситуациях, но в сложных случаях она создает проблемы для администраторов.

Для решения данных проблем можно воспользоваться Linux ACL. (Access Control Lists - списки контроля доступа) - версией POSIX ACL для Linux.

Linux ACL — это набор патчей для ядра операционной системы и программ для работы с файловой системой и несколько утилит, дающих возможность устанавливать права доступа к файлам не только для пользователя-владельца и группы-владельца файла, но и для любого пользователя или группы.

В версии ядра 2.6 ACL включен в стандартную поставку.

Linux ACL использует расширенные атрибуты для хранения данных о правах доступа к файлам пользователей и групп.

Расширенные атрибуты — это пара “имя/значение’, привязанная к определенному файлу.

Список расширенного контроля доступа существует для каждого inode и состоит из шести компонентов.

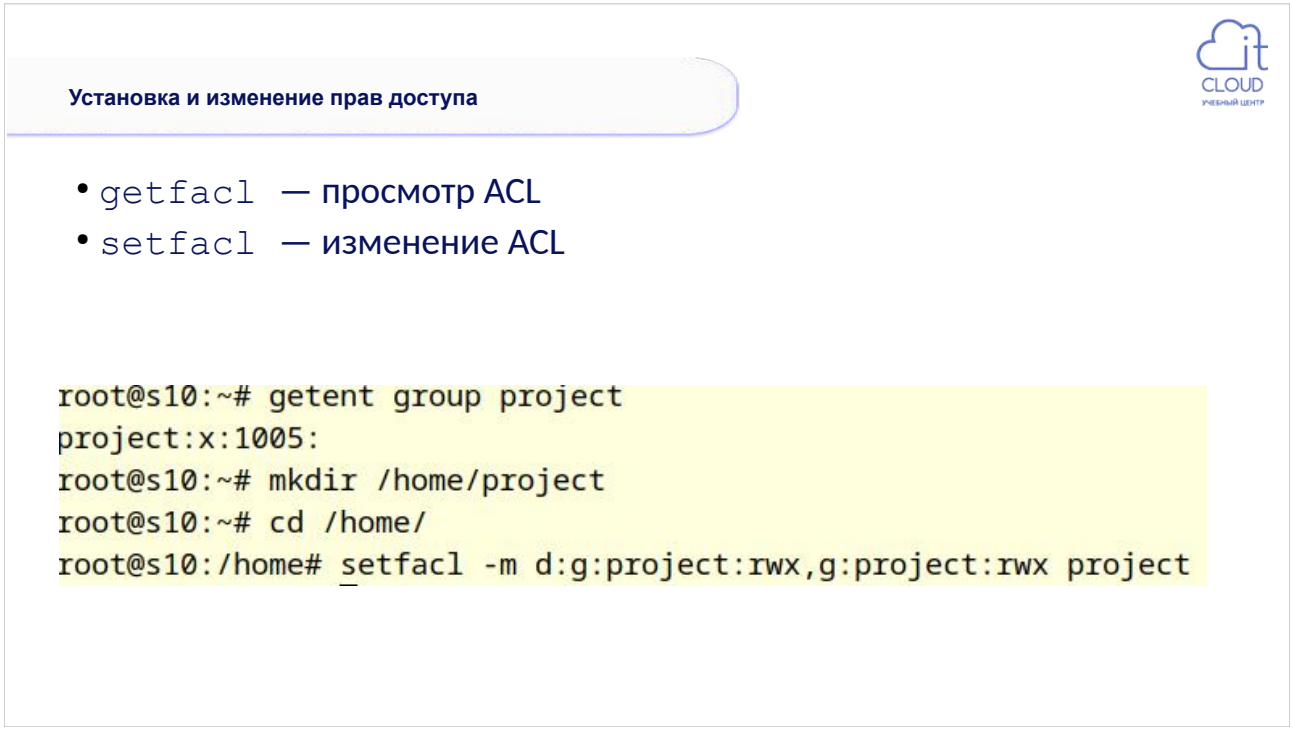
1. Первые три являются копией стандартных прав доступа к файлу. Они содержатся в единственном экземпляре в ACL и есть у каждого файла в системе:
    1. ACL_USER_OBJ — режим доступа к файлу пользователя-владельца;
    2. ACL_GROUP_OBJ — режим доступа к файлу группы-владельца;
    3. ACL_OTHER — режим доступа к файлу остальных пользователей.
2. Следующие компоненты устанавливаются для каждого файла в отдельности и могут присутствовать в ACL в нескольких экземплярах:
    1. ACL_USER — содержит UID и режим доступа к файлу пользователя, которому установлены права, отличные от основных.
    2. ACL_GROUP — то же самое, что и ACL_USER, но для группы пользователей;
    3. ACL_MASK — маска действующих прав доступа для расширенного режима.

На каждого пользователя со своими правами на данный файл хранится отдельная запись. Не может существовать более одной записи на одного и того же пользователя.

При установке дополнительных прав доступа присваивается значение и элементу ACL_MASK.

Каталоги также могут иметь список контроля доступа по умолчанию. В отличие от основного ACL, он действует на создаваемые внутри данного каталога файлы и каталоги.

При создании файла внутри такого каталога файл получает ACL, равный ACL. по умолчанию этого каталога - наследование.



Управление списками контроля доступа производится при помощи двух утилит: `getfacl` и `setfacl`.

С помощью `getfacl` можно просмотреть текущие параметры доступа любого файла.

Пример: при вызове `getfacl` для каталога `dir1` мы получим следующее:

```
user@s10:~$ getfacl dir1
# file: dir1
# owner: user
# group: user
user::rwx
group::rwx
mask::rwx
other::r-x
```

***Примечание**: Как можно видеть, каталог dir1 принадлежит пользователю user, группе user и значение прав доступа к каталогу — 0775. Каталог имеет только основные параметры доступа, поскольку изначально дополнительные права не устанавливаются.*

Дополнительные права доступа к файлу устанавливаются и изменяются при помощи утилиты `setfacl`.

Для этого используется следующий формат вызова:

`setfacl -<опции> <ACL_структура>, <ACL_структура>,…, <ACL_структура> <имя_файла> <имя_файла>`

ACL_структура представляет собой одну из следующих конструкций:

1. `[d[efault]:] [u[ser]:] <пользователь> [:<режимы_доступа>]` — определяет режим доступа к файлу или каталогу пользователя. Если пользователь не указан, определяет режим доступа пользователя-владельца;
2. `[d[efault]:] g[roup]: <группа> [:<режимы_доступа>]` — то же, что и предыдущая конструкция, но для группы;
3. `[d[efault]:] m[ask][:] [:<режимы_доступа>]` — определяет действующие права доступа;
4. `[d[efault]:] o[ther][:] [:<режимы_доступа>]` — определяет режим доступа для остальных пользователей.

Параметр `default` устанавливается для режима наследования прав доступа.

Для установки и изменения ACL. используются следующие опции:

1. `--set` — заменяет полностью ACL-файл, на указанный в командной строке;
2. `-m` — изменяет режимы доступа к файлу (каталогу);
3. `-x` — убирает правила доступа из ACL.

Пример: вот что мы получим, применив `setfacl` к каталогу `dir1`:

```
user@s10:~$ ls -ld dir1
drwxrwxr-x. 2 user user 6 Jul 28 20:21 dir1
user@s10:~$ setfacl \
>-m d:u:tempuser:rwx,u:tempuser:rwx,d:g:adm:rwx,g:adm:rwx,d:o::---,o::--- \
>dir1
user@s10:~$ ls -ld dir1
drwxrwx---+ 2 user user 6 Jul 28 20:21 dir1

user@s10:~$ getfacl dir1
# file: dir1
# owner: user
# group: user
user::rwx
user:tempuser:rwx
group::rwx
group:adm:rwx
mask::rwx
other::---
default:user::rwx
default:user:tempuser:rwx
default:group::rwx
default:group:adm:rwx
default:mask::rwx
default:other::---

user@s10:~$ touch dir1/test.file
user@s10:~$ mkdir dir1/test.dir
user@s10:~$ getfacl dir1/*
# file: dir1/test.dir
# owner: user
# group: user
user::rwx
user:tempuser:rwx
```

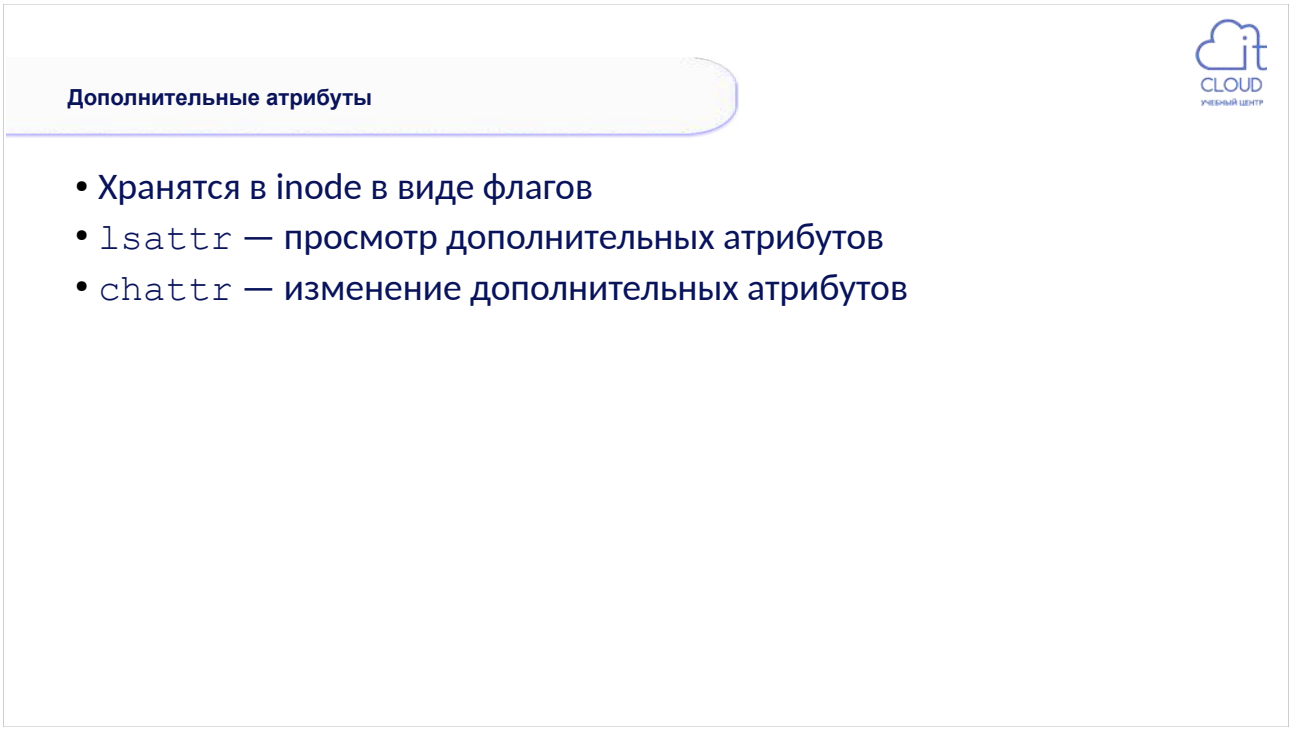
```
group::rwx
group:adm:rwx
mask::rwx
other::---
default:user::rwx
default:user:tempuser:rwx
default:group::rwx
default:group:adm:rwx
default:mask::rwx
default:other::---

# file: dir1/test.file
# owner: user
# group: user
user::rw-
user:tempuser:rwx       #effective:rw-
group::rwx              #effective:rw-
group:adm:rwx           #effective:rw-
mask::rw-
other::---
```

При использовании опции `-x` права доступа не указываются.

Пример:

```
user@s10:~$ setfacl -x u:tempuser dir1/test.file
user@s10:~$ getfacl dir1/test.file
# file: dir1/test.file
# owner: user
# group: user
user::rw-
group::rwx
group:adm:rwx
mask::rwx
other::---
```

5.5 Дополнительные атрибуты файлов.



В нескольких файловых системах (ext2/ext3/ext4, btrfs, xfs и др.) имеются дополнительные атрибуты файлов.

По умолчанию эти атрибуты при создании нового файла не устанавливаются, и их активизация производится вручную.

Команда `lsattr` показывает расширенные атрибуты.

Команда `chattr` изменяет расширенные атрибуты.

Атрибуты хранятся в блоке информационного дескриптора (inode) под заголовком flag (флаг) в виде шестнадцатеричного числа:

1. (a) только добавление
2. (A) не обновлять время доступа
3. (c) сжимать
4. (C) не применять copy on write
5. (d) не архивировать с помощью dump
6. (D) синхронное обновление каталогов
7. (e) хранить данные в виде экстентов
8. (i) не изменяемый
9. (j) журналирование данных
10. (s) защищенное удаление
11. (S) синхронные обновления

12. (t) запрет упаковывания хвостов
13. (T) на верху иерархии блоков
14. (u) защита от удаления (не запрет удаления, но сохранение данных)

Некоторые атрибуты не могут быть изменены `chattr`, но отображаются `lsattr`

1. (E) ошибка сжатия
2. (h) огромный (huge) файл
3. (I) индексированный каталог
4. (N) встроенные (в inode) данные
5. (X) сырой (raw) доступ к сжатым данным, and
6. (Z) грязный (dirty) сжатый файл.

Как и для команды `chmod`, требуемая операция в команде `chattr` указывается при помощи оператора, за которым следуют символьные обозначения атрибутов.

- «+» предписывает команде установить указанные атрибуты;
- «–» – сбросить атрибуты;
- «=» – установить указанные атрибуты и сбросить все остальные.

Параметр `-R`, обозначает, что команда будет применена рекурсивно в отношении всех подкаталогов.

Параметр `-V`, обозначает вывод версии программы `chattr`, а также дополнительных сообщений во время ее работы.

Параметр `-v`. При использовании этого параметра после него следует указать число, которое будет установлено в качестве номера версии индексного дескриптора. Никакого смысла, кроме того, который вы сами ему дадите, в номере версии нет — это просто число, которое можно записать в индексный дескриптор. Это число никак не связано с самим файлом. При создании файла ему выделяется индексный дескриптор из числа свободных, номер версии этого индексного дескриптора устанавливается равным единице.

Команда `lsattr` выводит список файлов и их расширенных-атрибутов.

Подобно команде `chattr`, команда `lsattr` поддерживает параметр `-R`, при наличии которого она рекурсивно обрабатывает все подкаталоги.

Параметр `-a` означает отображение информации обо всех файлах, включая файлы, имена которых начинаются с точки.

При использовании опции `-d` выводятся сведения только о каталогах, но не о файлах в них.

Параметр `-l` позволяет получить сведения о файлах в расширенном формате, где каждый атрибут отображается не с помощью буквы, а с помощью слов.

Параметр `-v` включает вывод версии файлов.

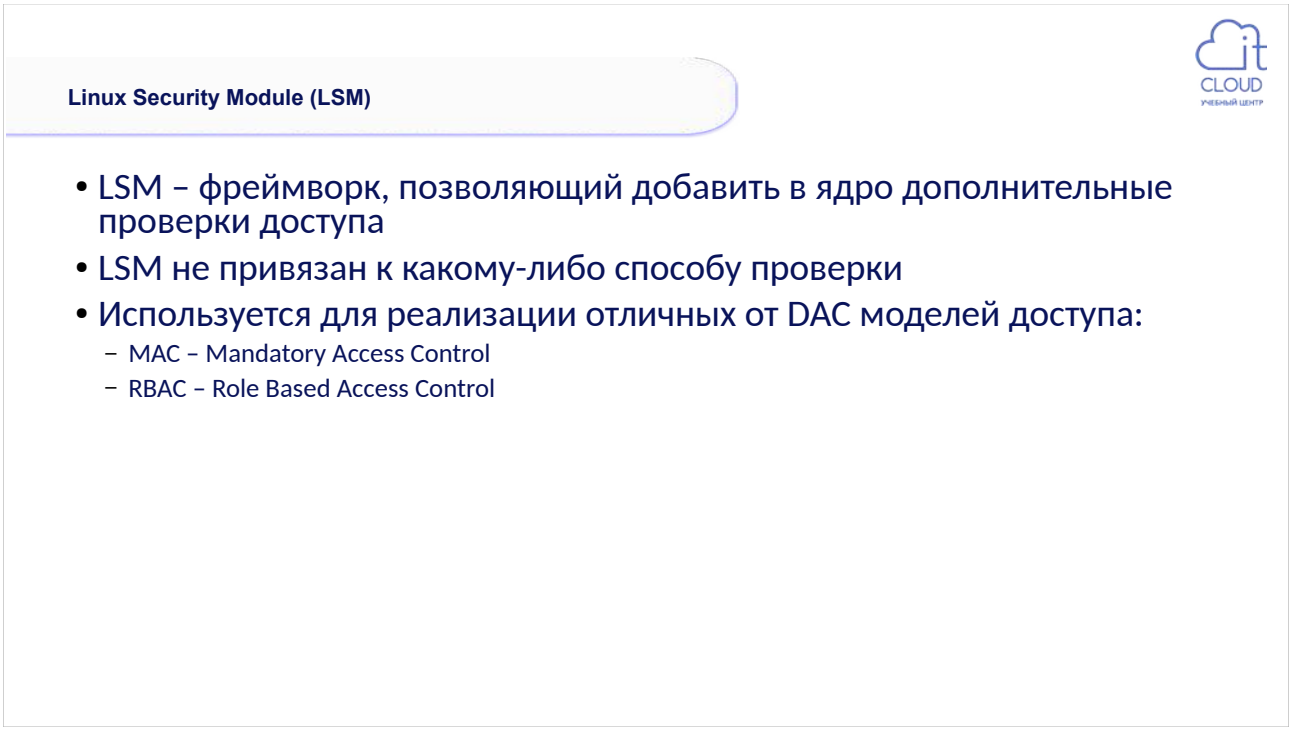
Пример: установка и проверка дополнительных атрибутов

```
root@s10:~# echo 123 > file
root@s10:~# echo abc >> file
root@s10:~# cat file
123
abc
root@s10:~# lsattr file
---------------- file
root@s10:~# chattr +a file
root@s10:~# echo xyz >> file
root@s10:~# echo 098 > file
-bash: file: Operation not permitted
root@s10:~# chattr +i file
root@s10:~# echo cde >> file
-bash: file: Permission denied
root@s10:~# echo 098 > file
-bash: file: Permission denied
root@s10:~# rm file
rm: remove regular file ‘file’? y
rm: cannot remove ‘file’: Operation not permitted
root@s10:~# lsattr file
----ia---------- file
```

***Примечание**: обратите внимание, что после установки атрибута append only (a) root не смог перезаписать файл, а после immutable (i) root не смог ничего сделать с файлом.*

Глава 6. Модули безопасности Linux.

6.1 Linux Security Module (LSM).



В большинстве операционных систем имеются средства управления доступом, которые определяют, может ли определенный объект (пользователь или программа) получить доступ к определенному ресурсу.

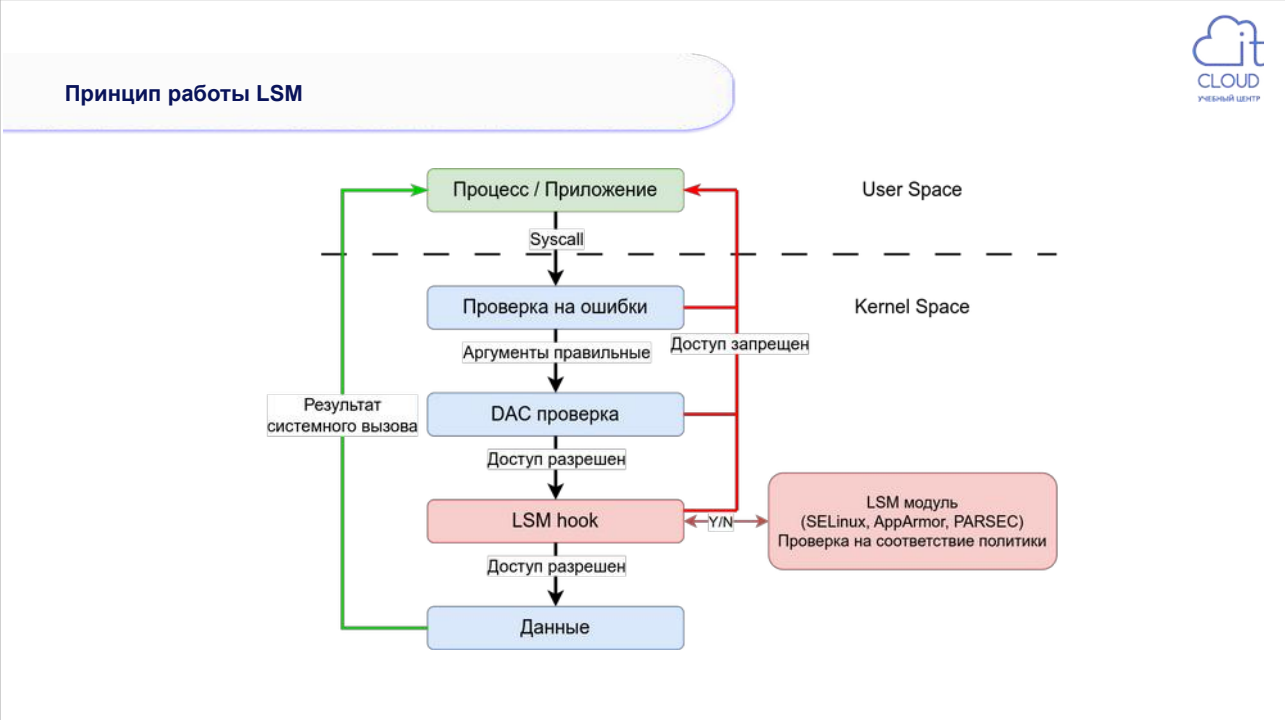
В системах UNIX® применяется разграничительный контроль доступа (discretionary access control, DAC). Этот метод позволяет ограничить доступ к объектам на основе групп, к которым они принадлежат.

Такое разграничение прав доступа может привести к возникновению ряда проблем из-за того, что программа, в которой может быть обнаружена уязвимость, наследует все права доступа пользователя. Следовательно, она может выполнять действия с тем же уровнем привилегий, какой есть у пользователя (что нежелательно).

Вместо того чтобы определять ограничения подобным образом, более безопасно использовать принцип наименьшего уровня привилегий (principle of least privilege), согласно которому программы могут делать только то, что им необходимо для выполнения своих задач, и не более того.

Разработчики ядра Linux не стали реализовывать в своем проекте конкретную модель доступа основанную на принципе наименьших привилегий, вместо этого создали фреймворк LSM (Linux Security Module), который позволяет применять любые существующие и будущие модели управления доступом.

LSM позволяет добавить в систему поддержку таких модулей управления доступом как MAC (Mandatory Access Control) или RBAC (Role Based Access Control).



В 2001 году было предложено добавить в ядро Линукс новый механизм обеспечения безопасности — SELinux, но идея была отвергнута, т. к. существовали и альтернативные проекты в области обеспечения безопасности и не было консенсуса, какая модель лучше.

Поэтому в конце 2003 была реализована концепция LSM, которая позволяла добавлять дополнительные модули ядра, которые реализуют мандатный контроль доступа.

Все LSM выполняют дополнительные проверки внутри соответствующего модуля, а запрос на доступ от приложения в виде системного вызова передается модулю через LSM hook. При чем вызывается LSM hook после того, как было получено разрешение от дискреционного механизма контроля доступа. Таки образом LSM не подменяет, но дополняет традиционные механизмы управления доступом.

Как при этом оценивается запрос от приложения внутри модуля не важно. Модуль может вернуть только два ответа: Да — разрешить действие или Нет — запретить действие.

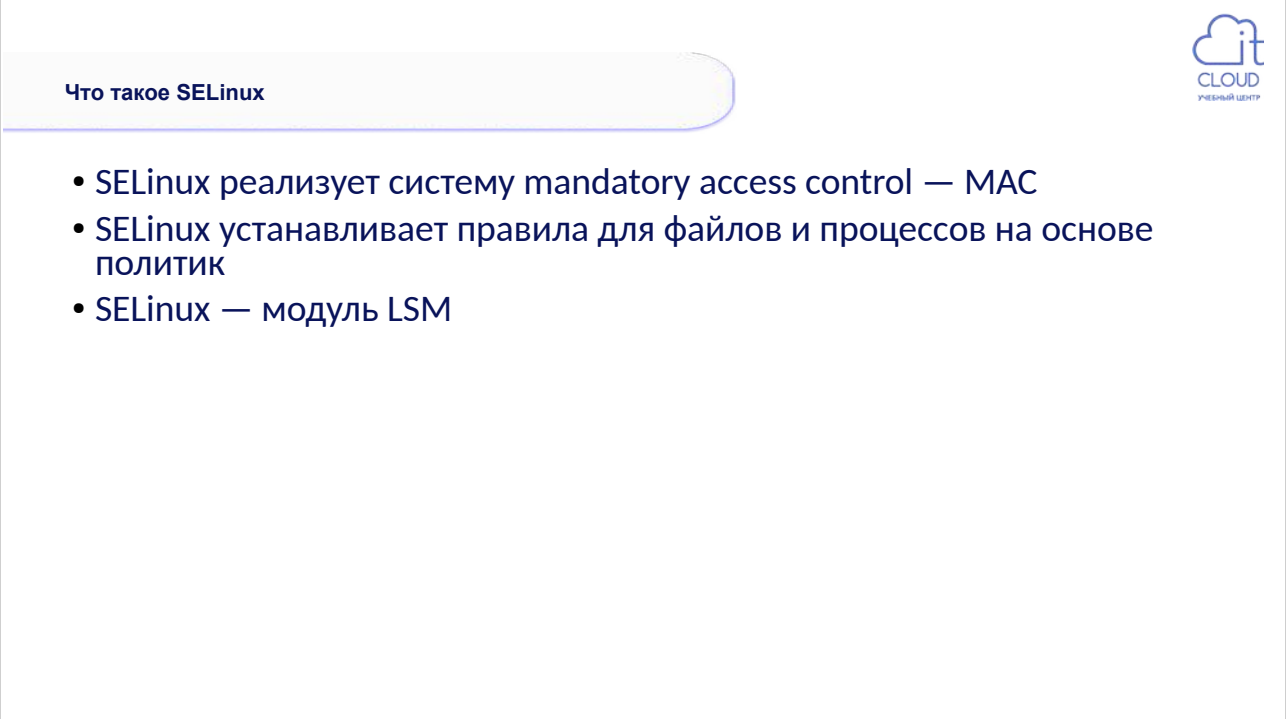
Как правило внутри модулей имеется некоторая политика, в соответствии с которой, модуль принимает решение о разрешении доступа.

Существуют несколько реализаций LSM модулей:

- SELinux (Security Enhanced Linux)
- AppArmor (Application Armor)
- PARSEC
- Smack
- TOMOYO
- Landlock

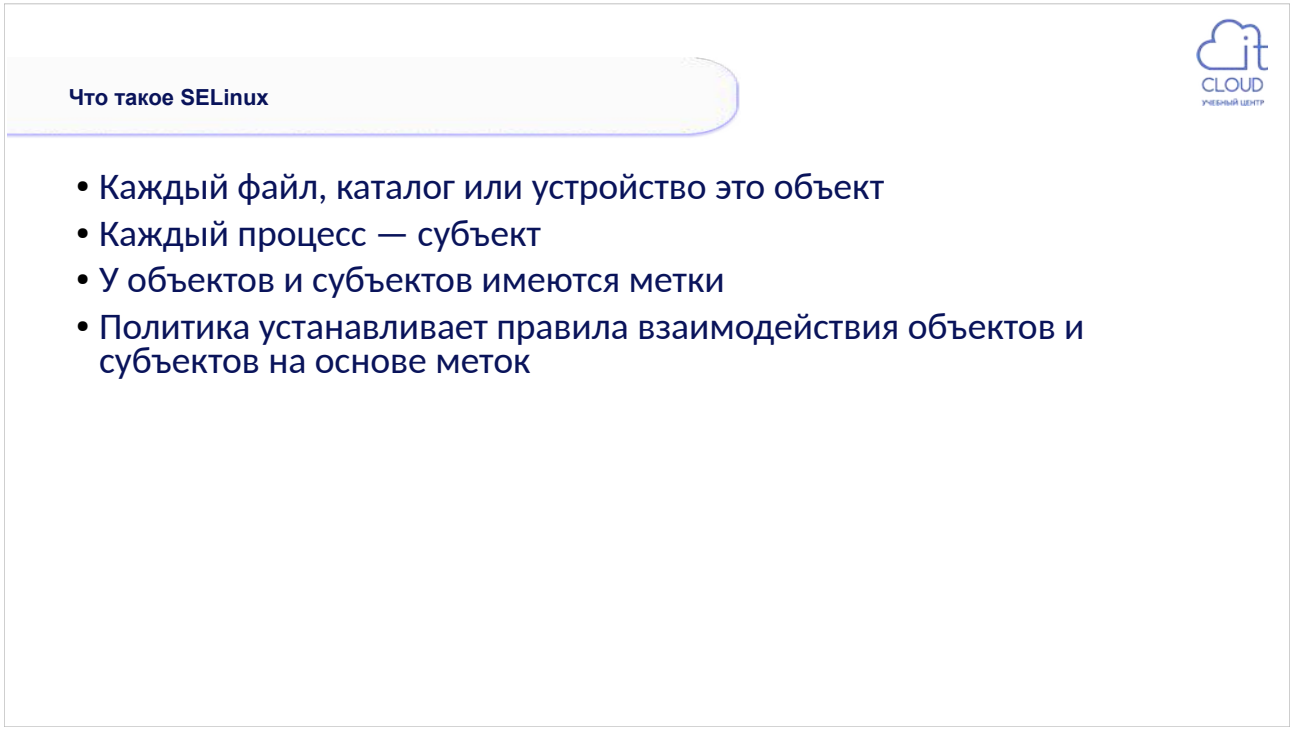
6.2 SELinux.

6.2.1 Механизм работы политик SELinux.



Linux с улучшенной безопасностью (SELinux) - это реализация принудительного управления доступом (mandatory access control — MAC) в ядре Linux, проверяющего разрешение на выполнение операций после проверки стандартного разграничительного управления доступом DAC.

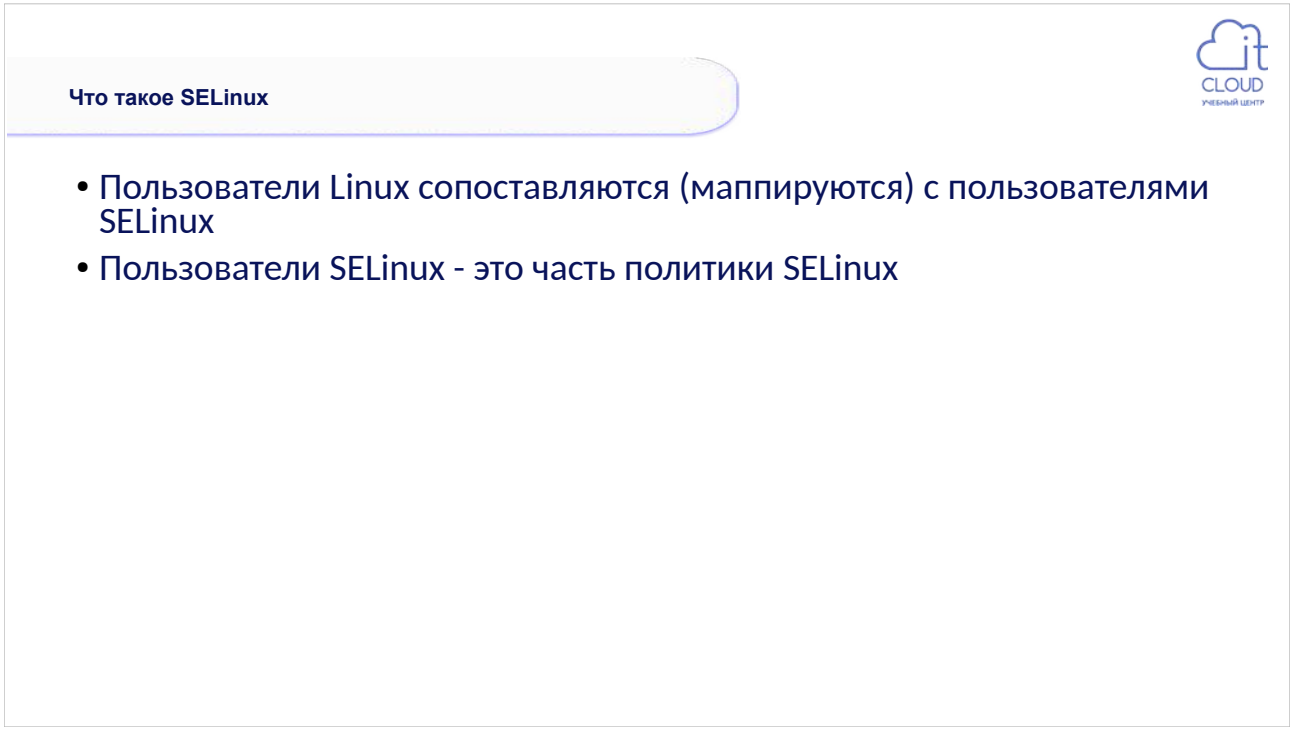
SELinux создан Агенством Национальной Безопасности и вводит в действие правила для файлов и процессов в системе Linux, для совершаемых над ними действий, основываясь на установленной политике.



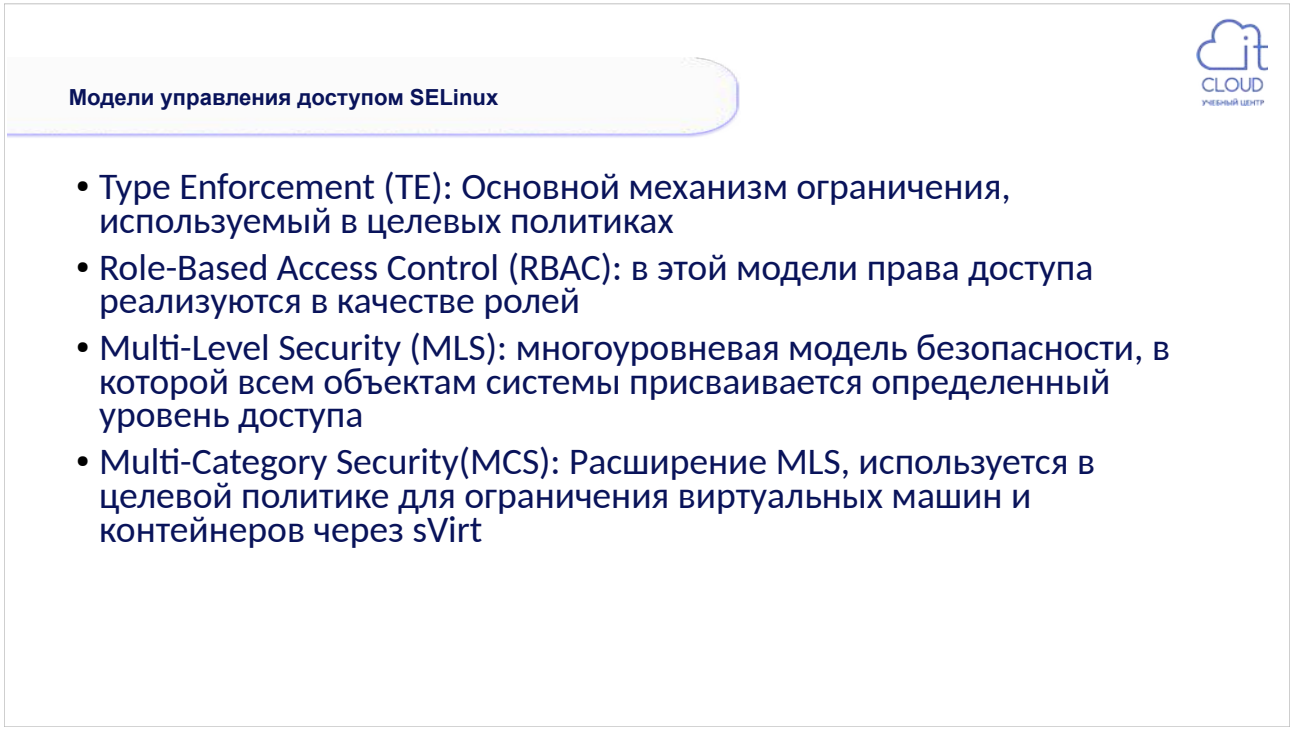
При использовании SELinux, файлы, включая директории и устройства являются объектами. Процессы, такие как, выполнение команды пользователем или приложение Mozilla® Firefox®, являются субъектами.

Основное назначение архитектуры MAC - это возможность принудительного назначения административно-установленной политики безопасности над всеми процессами и файлами системы, при этом решение основывается на метках, содержащих множество значимой информации по безопасности. Когда механизм SELinux реализован, он переводит систему в состоянии достаточной защищенности и предоставляет критичную поддержку приложениям, защищая приложения от взлома или обхода безопасности.

MAC предоставляет строгое разделение приложений и позволяет безопасное исполнение не доверенных приложений. Обладая способностью ограничивать привилегии, связанные с исполнением процессов, MAC ограничивает рамки потенциальной угрозы, таким образом ограничивая взлом уязвимостей в приложениях и системных службах. MAC включает защиту информации от пользователей корректно авторизованных в системе с ограниченными правами также как и от авторизованных пользователей, которые неосознанно исполняют вредоносный код.

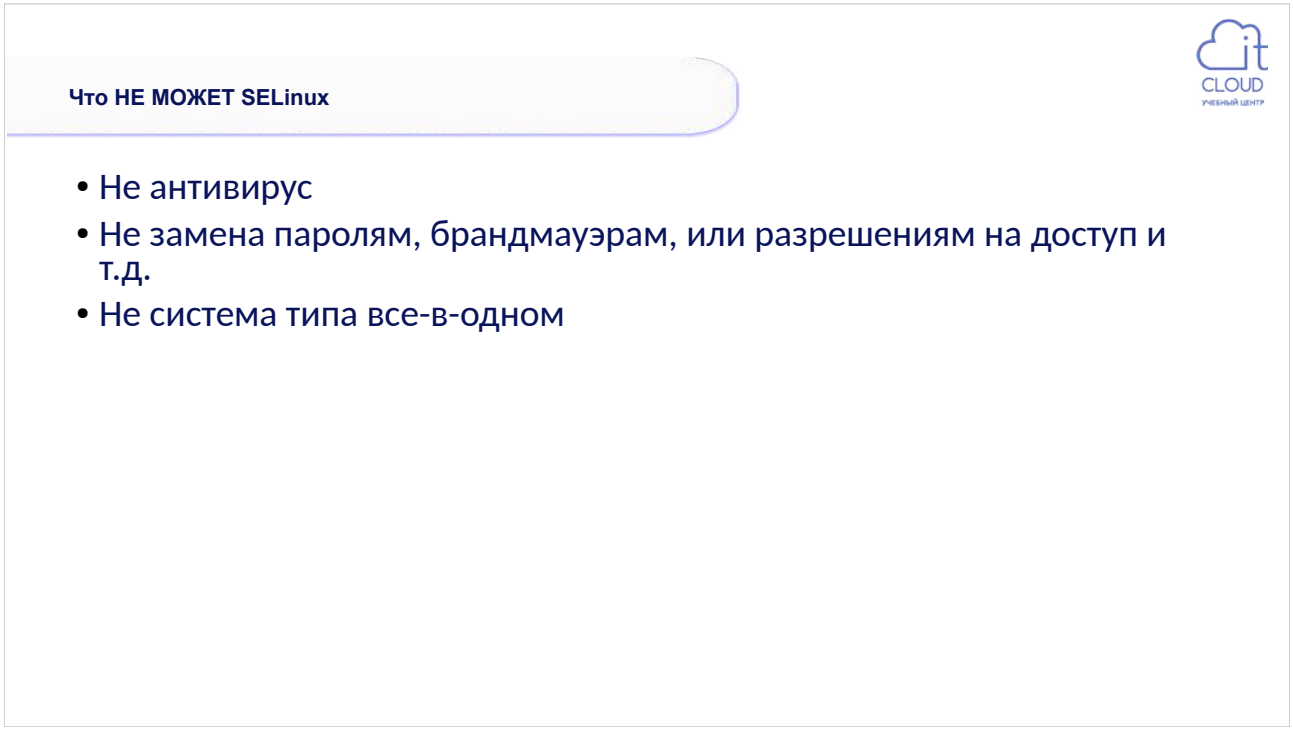


В операционных системах Linux с запущенным SELinux, существуют пользователи Linux и пользователи SELinux. Пользователи SELinux - это часть политики SELinux. Пользователи Linux сопоставляются (маппируются) с пользователями SELinux. Для того, чтобы избежать путаницы, в данном руководстве используются два термина "пользователь Linux" и "пользователь SELinux" для различия двух разных понятий.



В дополнение к DAC SELinux (Security Enhanced Linux) предлагает несколько вспомогательных моделей управления доступом:

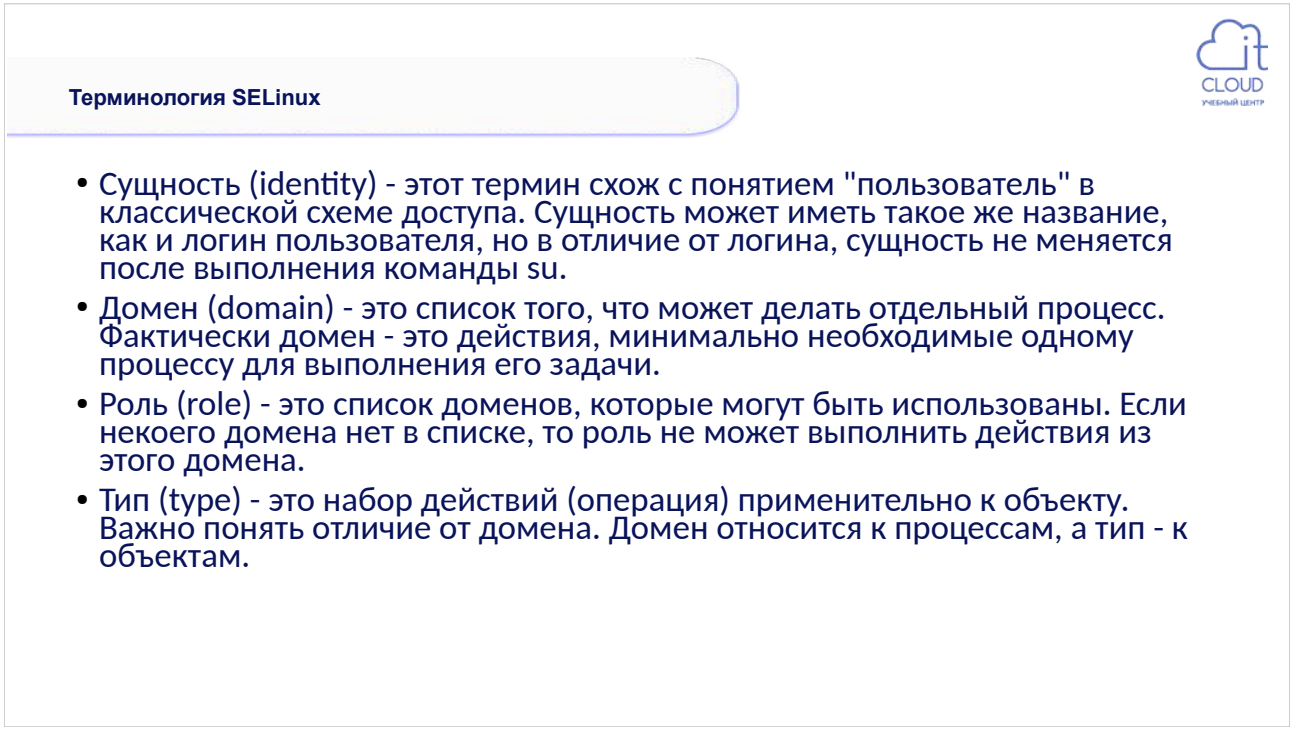
- **Type Enforcement (TE)**: Основной механизм ограничения, используемый в целевых политиках. Позволяет детально, на самом низком уровне управлять разрешениями. Самый гибкий, но и самый трудоемкий для системного администратора механизм.
- **Role-Based Access Control (RBAC)**: в этой модели права доступа реализуются в качестве ролей. Ролью называется разрешения на выполнение определенных действий одним или несколькими элементами системы над другими частями системы. По-сути, RBAC является дальнейшим развитием TE.
- **Multi-Level Security (MLS)**: многоуровневая модель безопасности, в которой всем объектам системы присваивается определенный уровень доступа. Разрешение или запрет доступа определяется только соотношением этих уровней.
- **Multi-Category Security(MCS)**: Расширение MLS, используется в целевой политике для ограничения виртуальных машин и контейнеров через sVirt.



SELinux не является:

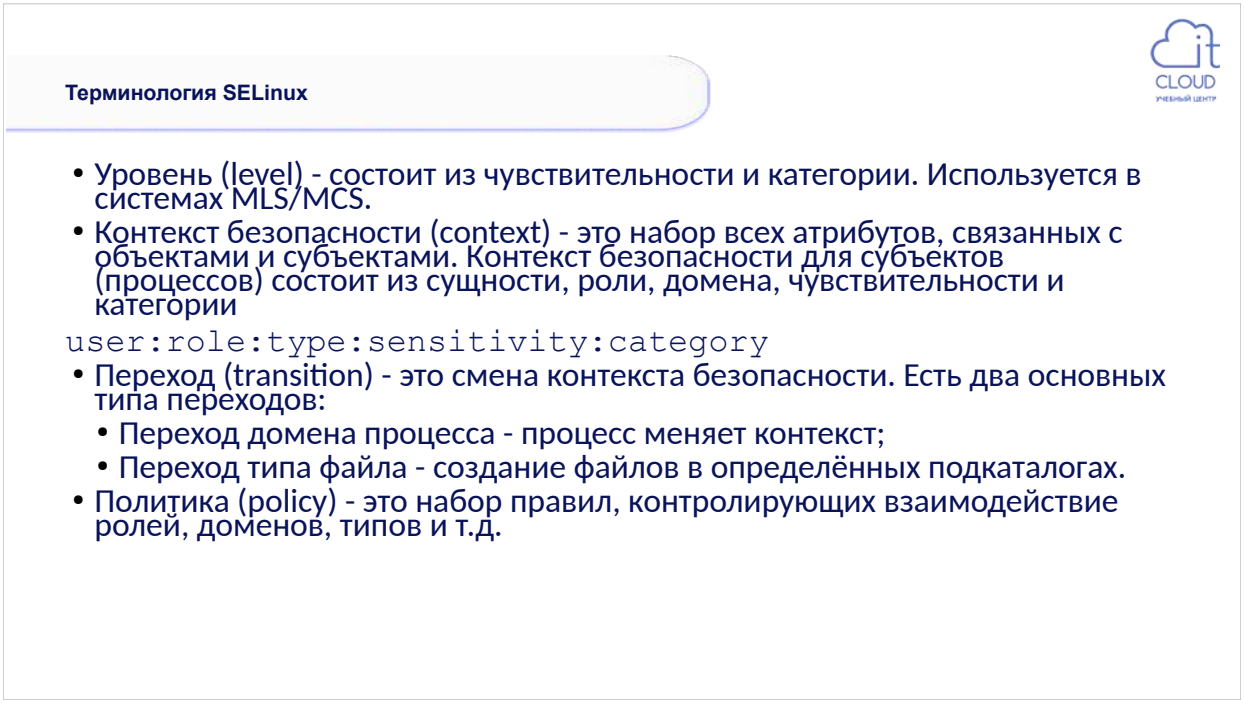
- антивирусным программным обеспечением.
- заменой паролям, межсетевым экранам или другим системам безопасности.
- решением безопасности "всё в одном".

SELinux разработан, для усовершенствования существующих решений по безопасности. Даже с запущенным SELinux, необходимо использование практик по безопасности, таких как обновление программного обеспечения последними обновлениями, использование сложных паролей, межсетевых экранов и прочего.



Основные понятия SELinux

- **Сущность (identity)** — этот термин схож с понятием "пользователь" в классической схеме доступа. Сущность может иметь такое же название, как и логин пользователя, но в отличие от логина, сущность не меняется после выполнения команды su. Если провести аналогию, то сущность - это конкретный человек, Вася Пупкин, Петя Смирнов и т.д.
- **Домен (domain)** — это список того, что может делать отдельный процесс. Фактически домен - это действия, минимально необходимые одному процессу для выполнения его задачи. По аналогии из реальной жизни, доменом можно назвать набор действий для совершения какой-либо операции.
- **Роль (role)** — это список доменов, которые могут быть использованы. Если некоего домена нет в списке, то роль не может выполнить действия из этого домена. В данном случае можно провести аналогию с должностью. То есть роль - это фактически должность (или должностная инструкция), которая может выполнять определённые наборы операций, или, в понятии SELinux, домены.
- **Тип (type)** — это набор действий (операция) применительно к объекту. Важно понять отличие от домена. Домен относится к процессам, а тип - к объектам, таким как файлы, каталоги, пайпы(pipes), сокеты и т. д.



- **Уровень (level)** — это атрибут многоуровневого управления доступом MLS и MCS. Пространство MLS - это пара уровней, записанных в виде `lowlevel-highlevel`, если уровни в данной паре отличаются или, если не отличаются, то `lowlevel`. То есть (`s0-s0` то же самое, что и `s0`). Если дополнительно определены категории, то уровень записывается как `sensitivity:category-set`. Если категории не определены, то запись выглядит как `sensitivity`.
- **Контекст безопасности (context)** — это набор всех атрибутов, связанных с объектами и субъектами. Контекст безопасности для субъектов (процессов) состоит из сущности, роли, домена, чувствительности и категории. Обычно используется только сущность-роль-домен(или тип), а, например, целевая политика от Fedora использует только домены и типы.
- **Переход (transition)** — это смена контекста безопасности. Есть два основных типа переходов:
  - Переход домена процесса — процесс меняет контекст; Например, запускается из-под пользователя некий демон. Selinux, на основе метки исполняемого файла, меняет его контекст.
  - Переход типа файла — создание файлов в определённых подкаталогах. Например, пользователь создаёт html-страничку в каталоге WEB-сервера. Чтобы WEB-сервер получил доступ к этой страничке, необходимо сменить контекст безопасности файла (WEB-сервер не имеет доступа к контексту пользователя).
- **Политика (policy)** — это набор правил, контролирующих взаимодействие ролей, доменов, типов и т. д. Политики работают на уровне системных вызовов и

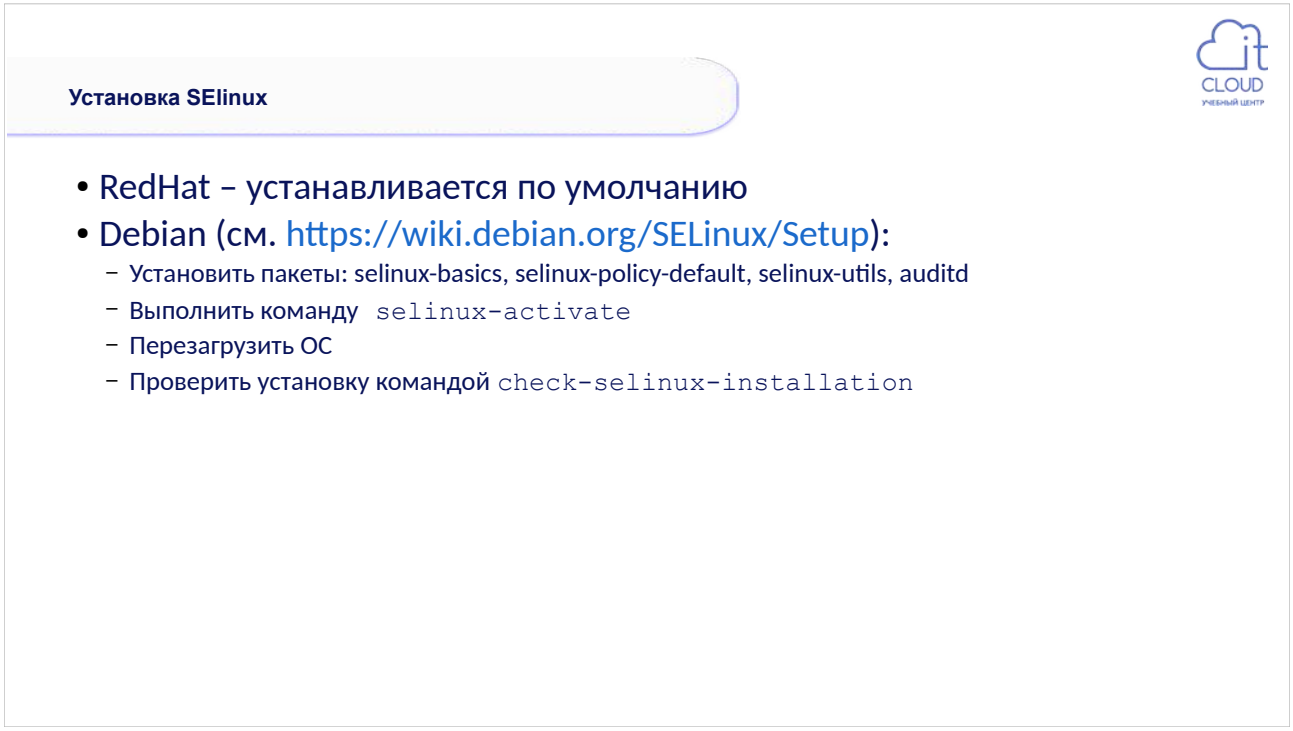
обрабатываются ядром, но можно реализовать и на уровне приложения. Политики описываются при помощи специального языка описания правил доступа.

В настоящий момент уже разработано несколько готовых политик безопасности, которые можно использовать по умолчанию на серверах и на домашних компьютерах. Всё, что требуется от системного администратора - выбрать используемую политику и перезагрузить компьютер с включённым SELinux.

В среднем, политика безопасности SELinux для всей системы содержит более ста тысяч правил, так что её создание и отладка занимает значительное время.

Наиболее распространены следующие политики:

1. **Целевая (targered)**. Эта политика разработана компанией Red Hat и является наиболее используемой;
2. **Минимальная (minimum)**. Является модификацией целевой политики, в которой только выбранные процессы защищаются.
3. **Многоуровневая (MLS)**. Позволяет обеспечивать уровни безопасности и может использоваться госструктурами для хранения информации различных уровней секретности;
4. **Строгая (strict)**. Этот вариант политики подразумевает правило "Что не разрешено, то запрещено".



В RedHat подобных дистрибутивах SELinux устанавливается и включается по умолчанию. Никаких специальных действий по установке не требуется.

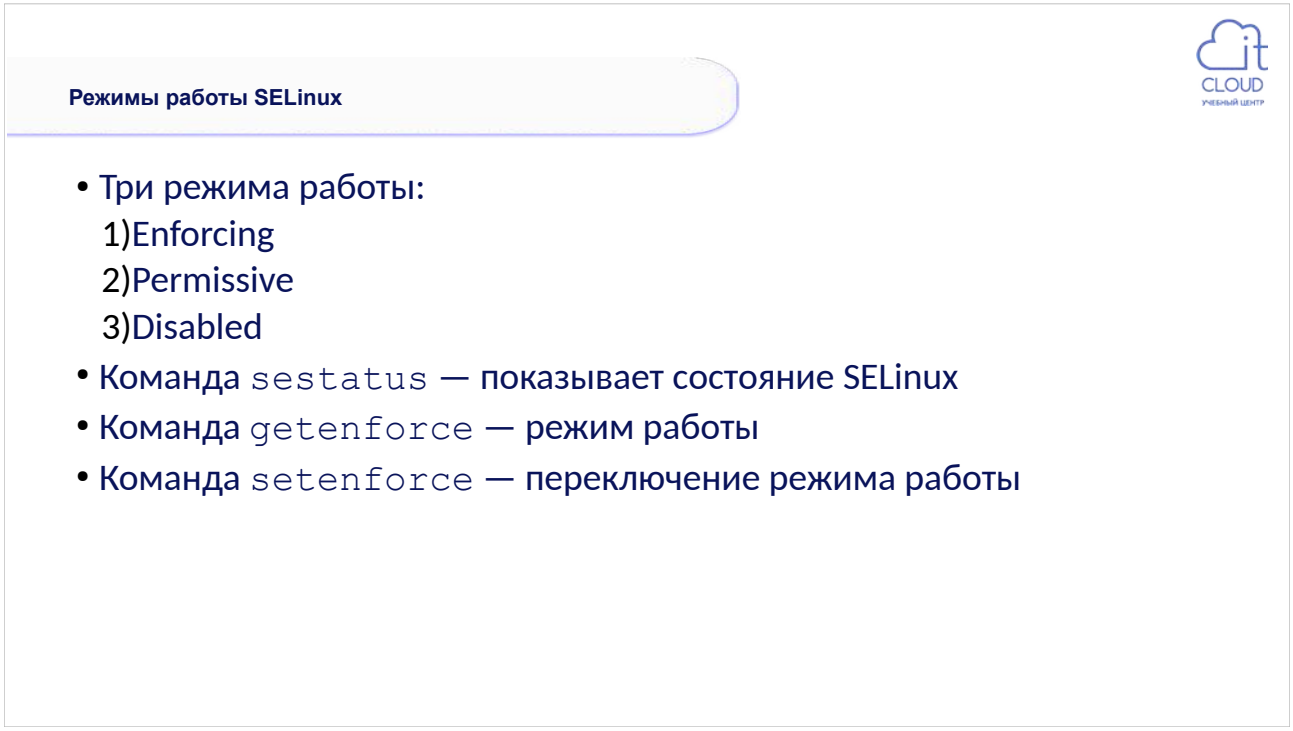
В Debian подобных дистрибутивах для использования SELinux его сначала требуется установить.

Пример: Установка SELinux на Debian, см. подробности тут: https://wiki.debian.org/SELinux/Setup

1. Проверьте предварительные требования: файловые системы должны поддерживать расширенные атрибуты, ядро скомпилировано с поддержкой аудита и собственно SELinux.
2. Установите нужные пакеты:

```
root@s10:~# apt install selinux-basics selinux-policy-default \
  selinux-utils auditd
```

3. Выполнить команду `selinux-activate`, которая настроит GRUB, PAM и создаст файл `/.autorelabel`.
4. Перезагрузите ОС.
5. Проверьте установку командой `check-selinux-installation`.



SELinux имеет три основных режим работы, при этом иногда по умолчанию установлен режим Enforcing. Это довольно жесткий режим, и в случае необходимости он может быть изменен на более удобный для конечного пользователя.

1. **Enforcing**: Режим по-умолчанию. При выборе этого режима все действия, которые каким-то образом нарушают текущую политику безопасности, будут блокироваться, а попытка нарушения будет зафиксирована в журнале.
2. **Permissive**: В случае использования этого режима, информация о всех действиях, которые нарушают текущую политику безопасности, будут зафиксированы в журнале, но сами действия не будут заблокированы.
3. **Disabled**: Полное отключение системы принудительного контроля доступа.

Команда `sestatus` показывает состояние SELinux.

Пример:

```
root@s10:~# sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             default
Current mode:                   permissive
Mode from config file:          permissive
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Memory protection checking:     actual (secure)
Max kernel policy version:      33
```

Также вы можете узнать статус SELinux при помощи команды `getenforce`.

Пример:

```
root@s10:~# getenforce
```

```
Permissive
```

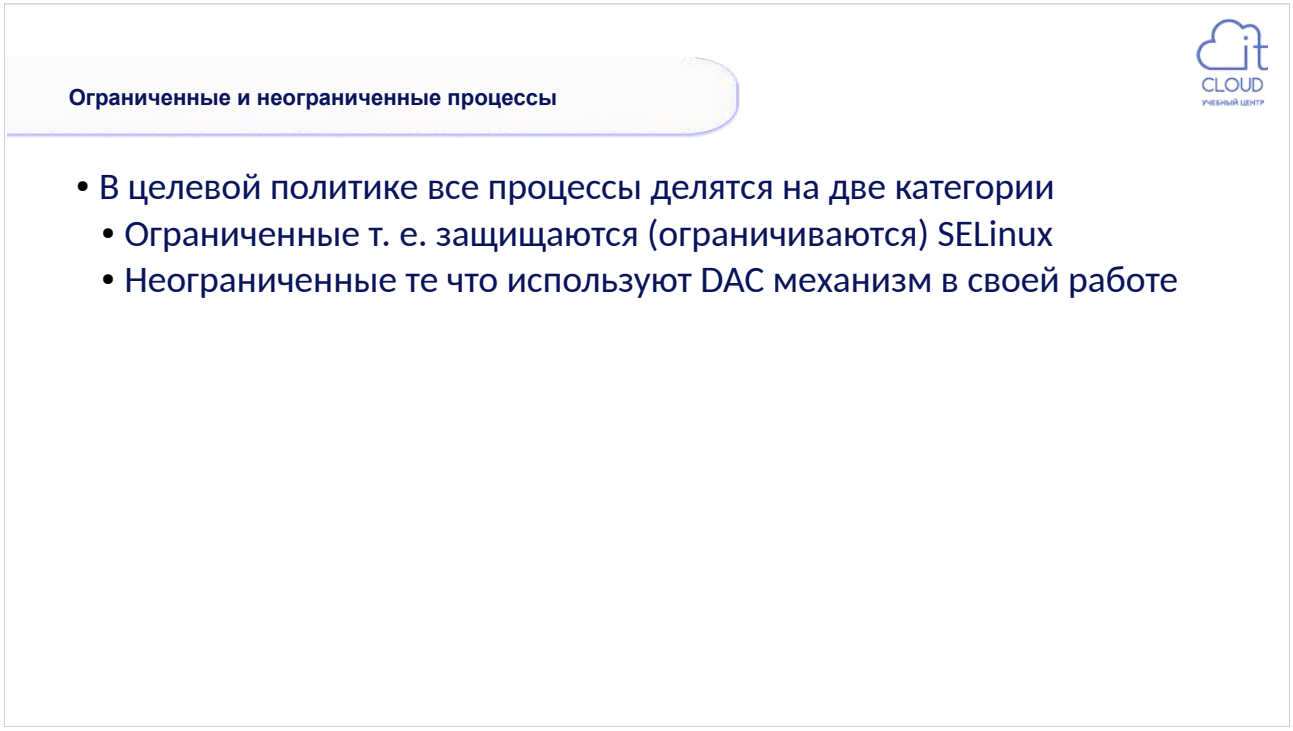
Команда `setenforce` позволяет быстро переключаться между режимами *Enforcing* и *Permissive*, изменения вступают в силу без перезагрузки. Но если вы включаете или отключаете SELinux, требуется перезагрузка, ведь нужно заново устанавливать метки безопасности в файловой системе.

Для установки режима по-умолчанию, который будет применяться при каждой загрузке системы, задайте значение строки `SELINUX=` в файле `/etc/selinux/config`, задав один из режимов — `enforcing`, `permissive` или `disabled`.

Пример:

```
root@s10:~# grep SELINUX=[a-z] /etc/selinux/config
SELINUX=permissive

root@s10:~# apt-cache show selinux-policy-default | grep -A5 Description-ru
Description-ru: ограниченный и специализированный варианты правил SELinux
 В пакете содержится образцовый набор правил для SE Linux. В стандартной
 конфигурации он предоставляет набор правил ранее известный как
 "специализированный" (targeted). Если удалить модуль unconfined, то будет
 предоставляться набор правил ранее известный как "ограниченный" (strict).
 .
```

Параметр `SELINUXTYPE=` в файле `/etc/selinux/config` указывает тип применяемой политики.



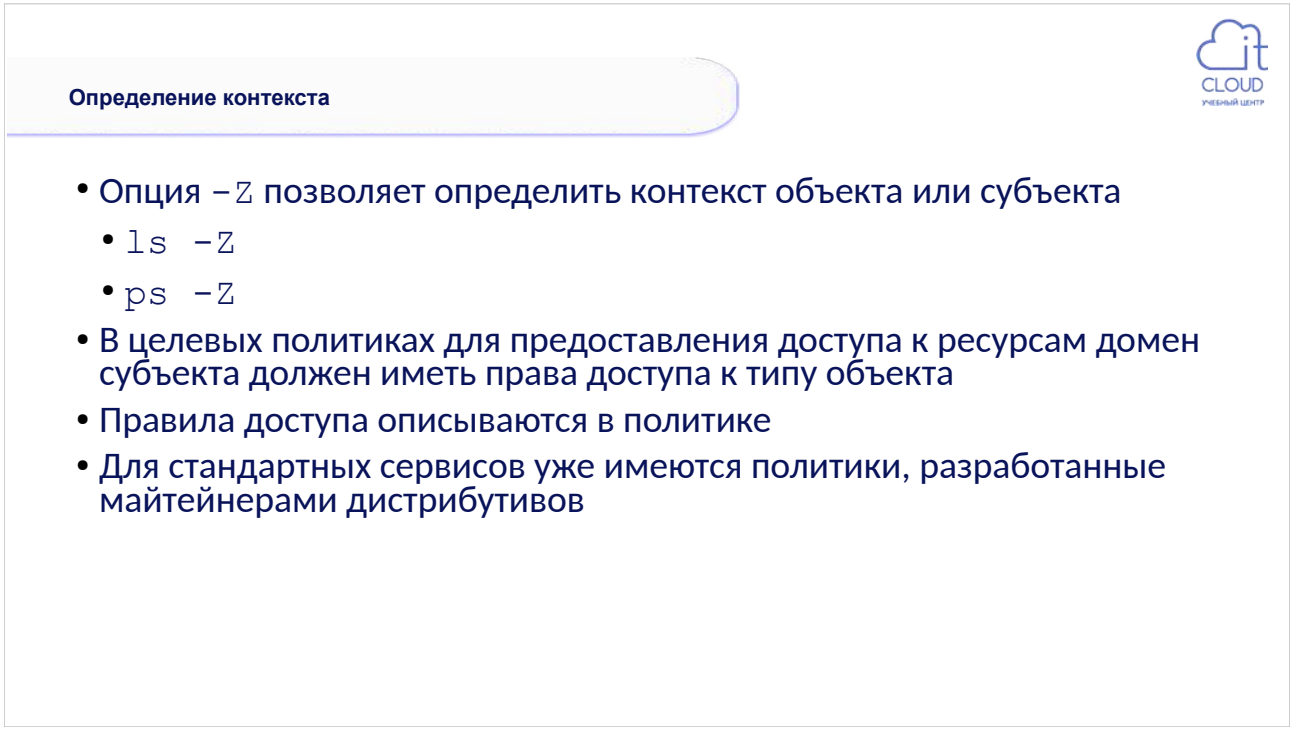
Когда целевая политика targeted используется, процессы, которые являются целевыми, запускаются в ограниченном домене, остальные процессы запускаются в неограниченном домене. Например, по умолчанию пользователи, прошедшие авторизацию, работают в домене `unconfined_t` и системные процессы запущенные init-ом запускаются в домене `initrc_t` - оба домена неограниченные.

Неограниченные домены (так же, как и ограниченные) - это субъекты для операций выполнения и записи в память. По умолчанию, субъекты запущенные в неограниченном домене не могут выделить память для записи и запустить ее. Это уменьшает степень угрозы атаки переполнения буфера buffer overflow attacks. Эти проверки памяти отключаются установкой Булевых переключателей, что позволяет изменять политику SELinux "на ходу". Настройка Булевых значений рассматривается позже.

Почти каждая сетевая служба ограничена. Также большинство процессов, которые запускаются в Linux с привилегиями пользователя root и выполняют задачи для пользователей, такие как приложение passwd, ограничены. Когда процесс ограничен, он запускается в своём собственном домене, например процесс `httpd` запускается в домене `httpd_t`. Если ограниченный процесс скомпрометирован атакующим, в зависимости от конфигурации SELinux, доступ атакующего к ресурсам и вред, который он может нанести ограничен.

Неограниченные (unconfined) процессы выполняются в неограниченных (unconfined) доменах, программы запускаемые init выполняются в неограниченном unconfined `initrc_t` домене, неограниченные процессы ядра запускаются в домене `kernel_t`. Для неограниченных процессов правила политики SELinux также применяются, но правила политики существуют для разрешения практически всех доступов для процессов,

запущенных в неограниченных доменах. Процессы запущенные в неограниченных доменах откатываются к использованию только правил DAC. Если неограниченный процесс скомпрометирован, SELinux не ограничивает атакующего от получения доступа к системным ресурсам и информации, но, конечно, правила DAC всё равно используются. SELinux это улучшение механизмов безопасности над DAC, но SELinux не заменяет его.



В целевых (targeted) политиках предоставление доступа основано на анализе меток. Политика проверяет может ли домен процесса (субъекта) получить доступ к ресурсу (объекту). SELinux перехватывает системные вызовы и разрешает доступ, если политика это позволяет.

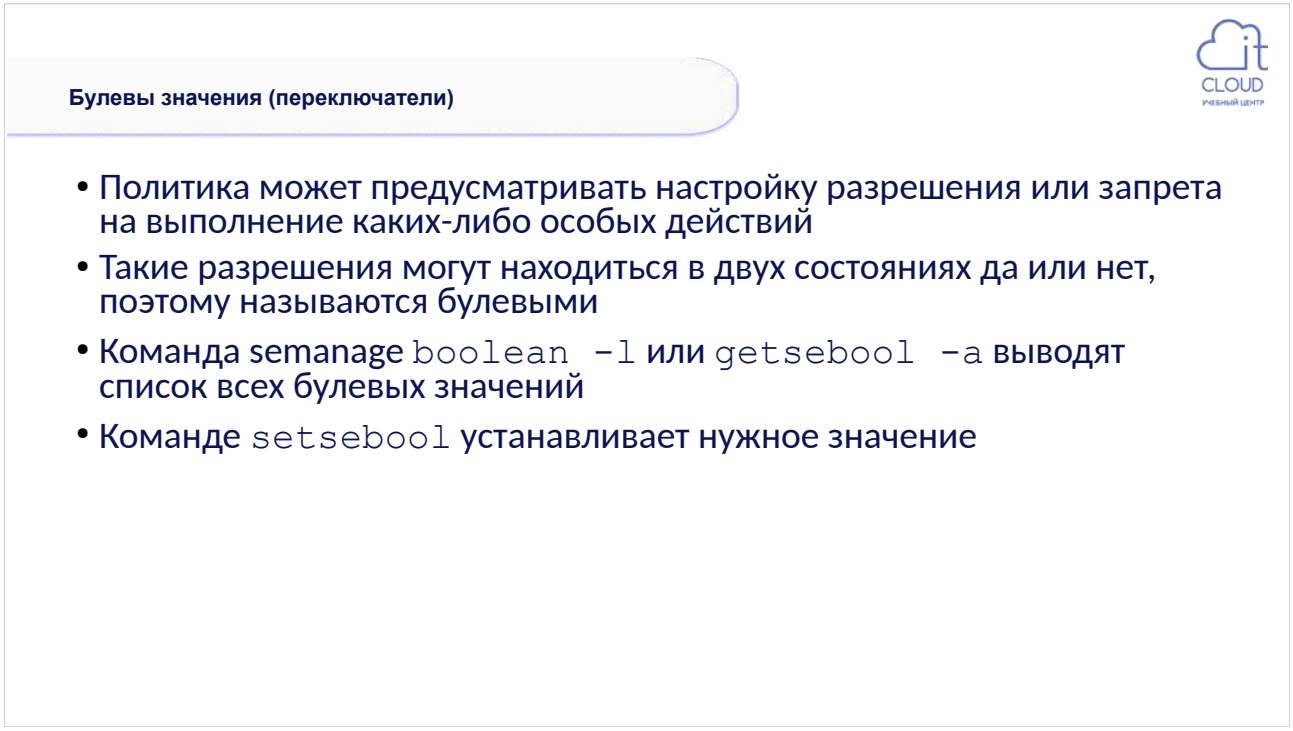
Информация о метках находится в контексте.

Основная опция получения информации о контекстах `-Z`.

Пример: Мы проверим контекст файлов, которые использует демон apache2. И контекст процесса веб сервера. В политиках имеется разрешение для домена `httpd_t` получать доступ к типам `httpd_sys_content_t`. Чтобы процесс получил домен `httpd_t` программа имеет тип `httpd_exec_t`.

```
root@s10:~# apt install apache2

root@s10:~# ls -Z /var/www/html/index.html
unconfined_u:object_r:httpd_sys_content_t:s0 /var/www/html/index.html

root@s10:~# ps -eZ | grep apache2
system_u:system_r:httpd_t:s0       1609 ?        00:00:00 apache2
system_u:system_r:httpd_t:s0       1610 ?        00:00:00 apache2
system_u:system_r:httpd_t:s0       1639 ?        00:00:00 apache2

root@s10:~# ls -Z /usr/sbin/apache2
system_u:object_r:httpd_exec_t:s0 /usr/sbin/apache2
```

В примере выше мы не видим самой политики. Политика представляет из себя бинарный файл, который во время старта системы загружается в ядро. Правила написания политик мы обсудим ниже.



Переключатели позволяют изменять части политики SELinux во время работы (без перезапуска и остановки), не обладая глубоким пониманием создания политики SELinux. Это позволяет вносить изменения, такие как: разрешение доступа службам к файловым системам NFS, без перезагрузки или рекомпиляции политики SELinux.

Для получения списка переключателей, объяснения, за что отвечает каждый переключатель, включен или выключен, необходимо выполнить команду `semanage boolean -l` от имени пользователя root.

Пример:

```
root@s10:~# semanage boolean -l | egrep '(Переключатель|httpd.*connect_db)'
Переключатель SELinux          Состояние  По умолчанию Описание
httpd_can_network_connect_db   (выкл.,выкл.)  Determine whether scripts and
modules can connect to databases over the network.
```

Команда `getsebool -a` выводит список переключателей, показывает выключены они или нет, но не даёт описания, за что они отвечают.

Пример:

```
root@s10:~# getsebool -a | grep 'httpd.*connect_db'
httpd_can_network_connect_db --> off
```

Для получения статуса одного конкретного Булева значения (переключателя) *`boolean-name`* используется команда `getsebool` *`boolean-name`*

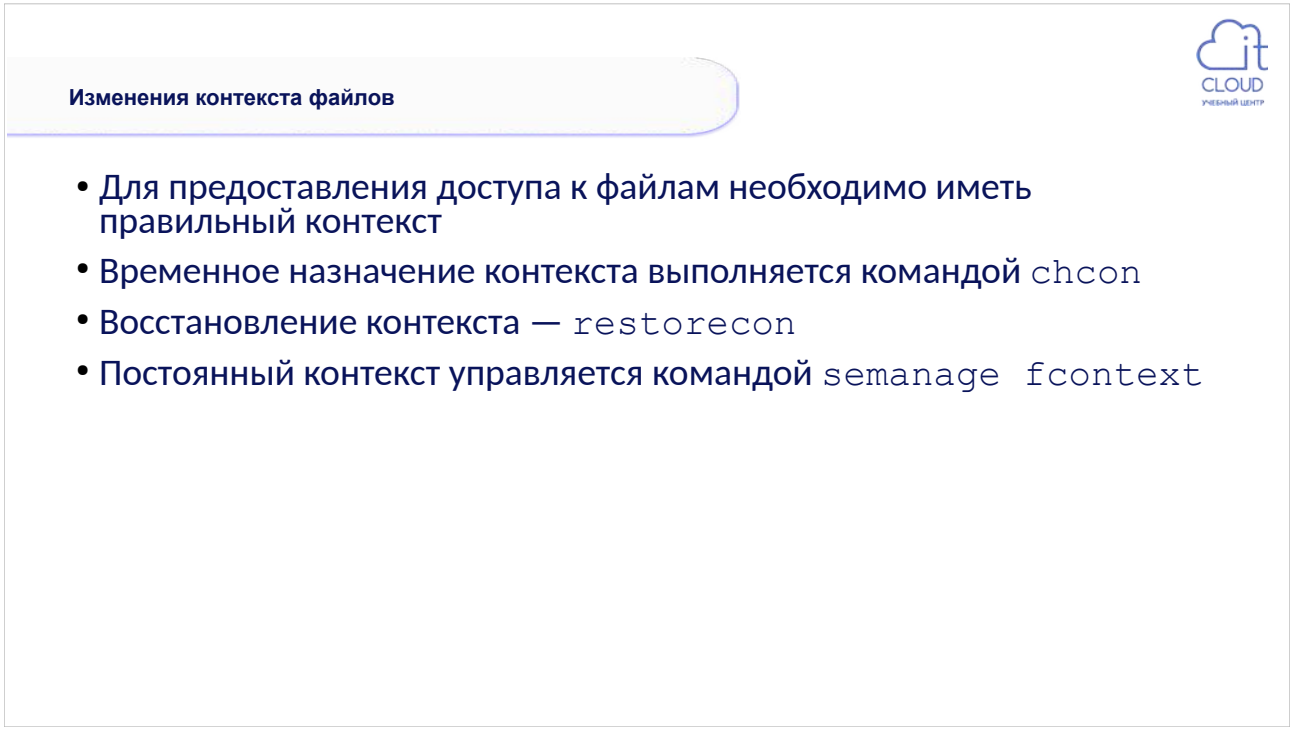
Пример:

```
root@s10:~# getsebool httpd_can_network_connect_db
httpd_can_network_connect_db --> off
```

Команда `setsebool` *`boolean-name x`* переводит переключатели в состояние включено или выключено, где *`boolean-name`* - название переключателя, а *`x`* - `on` для включения или `off` для выключения.

Пример:

```
root@s10:~# setsebool httpd_can_network_connect_db on
root@s10:~# getsebool httpd_can_network_connect_db
httpd_can_network_connect_db --> on
```



Чтобы некоторый ограниченный процесс получил доступ к файлу, последний, в свою очередь, должен иметь нужный тип.

Команда `chcon` устанавливает временный контекст.

Пример: Переключаем в режим Enforcing. Создаем собственный ресурс и получаем запрет доступа. После установки правильного контекста доступ появляется.

```
root@s10:~# setenforce 1
root@s10:~# getenforce
Enforcing

root@s10:~# mkdir /home/myweb
root@s10:~# echo 'My test Web Page' > /home/myweb/index.html
root@s10:~# ls -Z /home/myweb/
unconfined_u:object_r:home_root_t:s0 index.html

root@s10:~# cat /etc/apache2/sites-enabled/myweb.conf
Alias /myweb /home/myweb
<Directory /home/myweb>
 AllowOverride none
 Require all granted
</Directory>

root@s10:~# curl http://127.0.0.1/myweb/index.html
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>403 Forbidden</title>
</head><body>
<h1>Forbidden</h1>
<p>You don't have permission to access this resource.</p>
<hr>
<address>Apache/2.4.62 (Debian) Server at 127.0.0.1 Port 80</address>
</body></html>
```

```
root@s10:~# setenforce 0
root@s10:~# curl http://127.0.0.1/myweb/index.html
My test Web Page

root@s10:~# setenforce 1
root@s10:~# curl http://127.0.0.1/myweb/index.html
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>403 Forbidden</title>
</head><body>
<h1>Forbidden</h1>
<p>You don't have permission to access this resource.</p>
<hr>
<address>Apache/2.4.62 (Debian) Server at 127.0.0.1 Port 80</address>
</body></html>

root@s10:~# chcon -t httpd_sys_content_t /home/myweb/index.html
root@s10:~# curl http://127.0.0.1/myweb/index.html
My test Web Page
```

Команда `restorecon` восстанавливает контекст по умолчанию.

Пример: Восстановление контекста приводит к запрету доступа.

```
root@s10:~# restorecon -R /home/
root@s10:~# curl http://127.0.0.1/myweb/index.html
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>403 Forbidden</title>
</head><body>
<h1>Forbidden</h1>
<p>You don't have permission to access this resource.</p>
<hr>
<address>Apache/2.4.62 (Debian) Server at 127.0.0.1 Port 80</address>
</body></html>

root@s10:~# ls -Z /home/myweb/index.html
unconfined_u:object_r:user_home_t:s0 /home/myweb/index.html
```

Команда `semanage fcontext` изменяет контекст SELinux для файлов. При использовании целевой политики targeted, изменение вносимые данной командой, добавляются в файл `/etc/selinux/targeted/contexts/files/file_contexts`, если изменения вносятся для существующих файлов, то они добавляются в файл `file_contexts`, или добавляются файл `file_contexts.local` для новых файлов и каталогов, например при создании каталога `/web/.setfiles`, использующаяся при маркирование файловой системы и `restorecon`, использующаяся для восстановления контекста SELinux по умолчанию, читают эти файлы. Это значит, что изменения вносимые командой `semanage fcontext` постоянно, даже если файловая система будет перемаркирована. Политика SELinux контролирует возможность пользователей изменять контекст файлов.

Для внесения изменений в контекст SELinux изменений, которые сохранятся при перемаркировании файловой системы надо:

1. Выполнить команду `semanage fcontext -a options file-name|directory-name` Помните, что необходимо использовать полные пути к файлам и каталогам.
2. Выполнить команду `restorecon -v file-name|directory-name` для применения изменений контекста.

Пример: Применение контекста к каталогу.

```
root@s10:~# semanage fcontext -a -t httpd_sys_content_t /home/myweb
libsemanage.add_user: user sddm not in password file
root@s10:~# restorecon -R /home/
root@s10:~# ls -Z /home/myweb/index.html
unconfined_u:object_r:user_home_t:s0 /home/myweb/index.html

root@s10:~# ls -dZ /home/myweb
unconfined_u:object_r:httpd_sys_content_t:s0 /home/myweb

root@s10:~# semanage fcontext -a -t httpd_sys_content_t '/home/myweb(/.*)?'
libsemanage.add_user: user sddm not in password file
root@s10:~# restorecon -vR /home/myweb/
Relabeled /home/myweb/index.html from unconfined_u:object_r:user_home_t:s0 to
unconfined_u:object_r:httpd_sys_content_t:s0

root@s10:~# !curl
curl http://127.0.0.1/myweb/index.html
My test Web Page

root@s10:~# tail -2 /etc/selinux/default/contexts/files/file_contexts.local
/home/myweb    system_u:object_r:httpd_sys_content_t:s0
/home/myweb(/.*)?    system_u:object_r:httpd_sys_content_t:s0
```

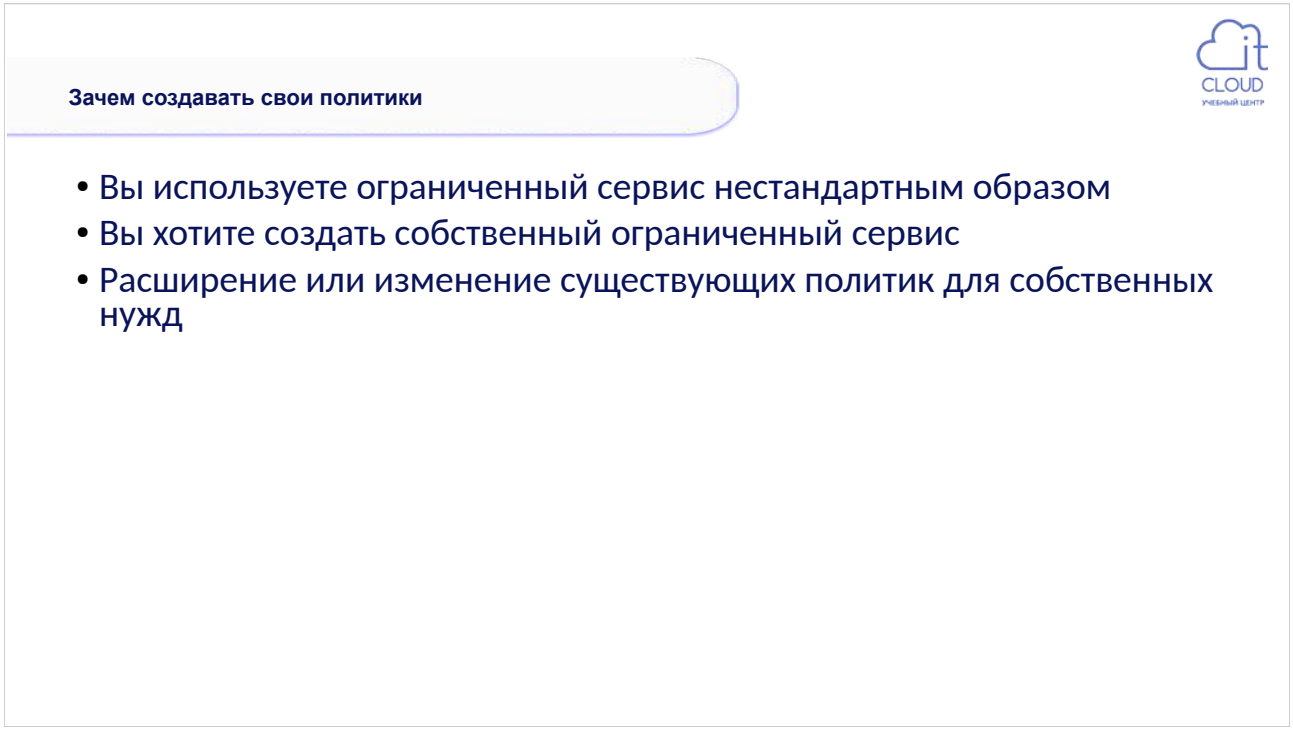
Первое из показанных правил маркировки лишнее, т. к. второе правило (/home/myweb(/.)?) применяется как к каталогу /home/myweb так и ко всему его содержимому.*

Удаление перманентной маркировки файлов производится командой `semanage fcontext -d options file-name|directory-name`

Пример: Удаление лишнего правила маркировки.

```
root@s10:~# semanage fcontext -d -t httpd_sys_content_t /home/myweb
libsemanage.add_user: user sddm not in password file
root@s10:~# tail -2 /etc/selinux/default/contexts/files/file_contexts.local
/usr/bin/VBoxClient    system_u:object_r:unconfined_execmem_exec_t:s0
/home/myweb(/.*)?    system_u:object_r:httpd_sys_content_t:s0
```

6.2.2 Язык описания правил доступа.



SELinux это гибкий и мощный инструмент обеспечения безопасности. Основой для принятия решений в SELinux служит политика. Политику по умолчанию создают люди, которые поддерживают дистрибутив Линукса. Основу для этой политики создают разработчики программного обеспечения.

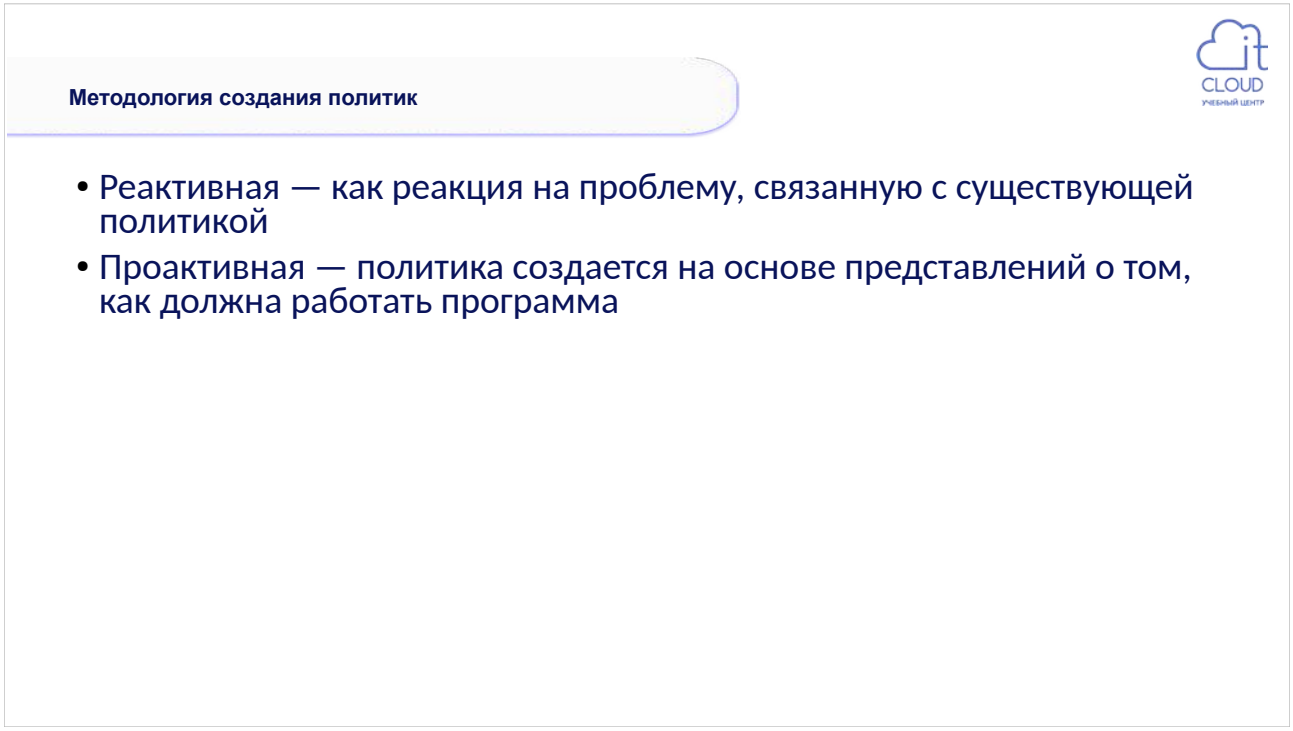
Разработчики софта и дистрибутивов, как правило, придерживаются следующих целей:

- Программы должны работать. Пользы от программы, которая абсолютна защищена и при этом не может работать, нет никакой.
- Обеспечение безопасности основано на общих принципах и соображениях. Это означает, что в конкретно вашей установленной системе, не учитываются все нюансы функционирования даже стандартных пакетов.
- В каждом дистрибутиве имеется набор предусмотренного этим дистрибутивом пакетов. Если вы используете какой-то пакет не предусмотренный разработчиками, то для него не будет политики. В этом случае пакет или не работает или попадает в категорию неограниченных, т. е. не защищается с помощью SELinux.

Исходя из выше изложенного можем сделать вывод, что рано или поздно у вас может возникнуть потребность в создании собственной политики. Основными причинами для этого могут служить следующие соображения:

- Вы используете стандартный ограниченный сервис не так как это предусматривал разработчик. В этом случае вам придется расширять стандартную политику своими правилами.

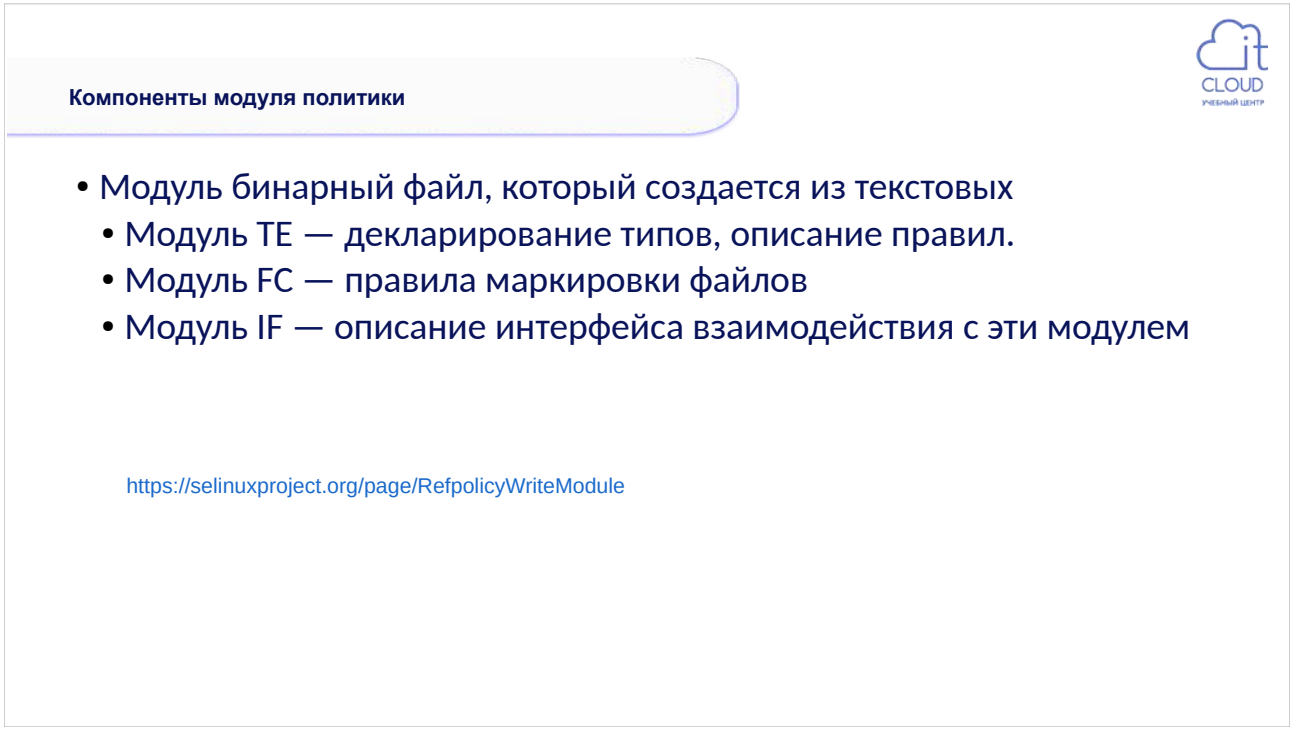
- Вы используете пакет, которого нет в стандартной поставке, или создаете собственный пакет. В таком случае вам потребуется создать свой модуль политики SELinux для данного программного обеспечения.
- Вы, после оценки, существующих дефолтных политик пришли к выводу, что ограничения в ней слишком либеральные и необходимо ужесточение политик.



Перед созданием собственных политик, нужно понимать, что имеется два подхода для создания политик.

- Реактивный. В этом случае вы решаете какую-то конкретную проблему, связанную с неправильной работой ограниченной программы. Как результат решения проблемы вы создаете политику SELinux и загружаете ее.
- Проактивный. Вы создаете собственный ограниченный сервис и у вас имеются представления о том, что и как этот сервис должен делать. Результатом этих представлений будет созданная вами политика.

Конечно же имеется и гибридный подход, в котором, например, сначала создается политика, применяется, тестируется, на основе тестов модифицируется и снова применяется, анализируется и т.д.

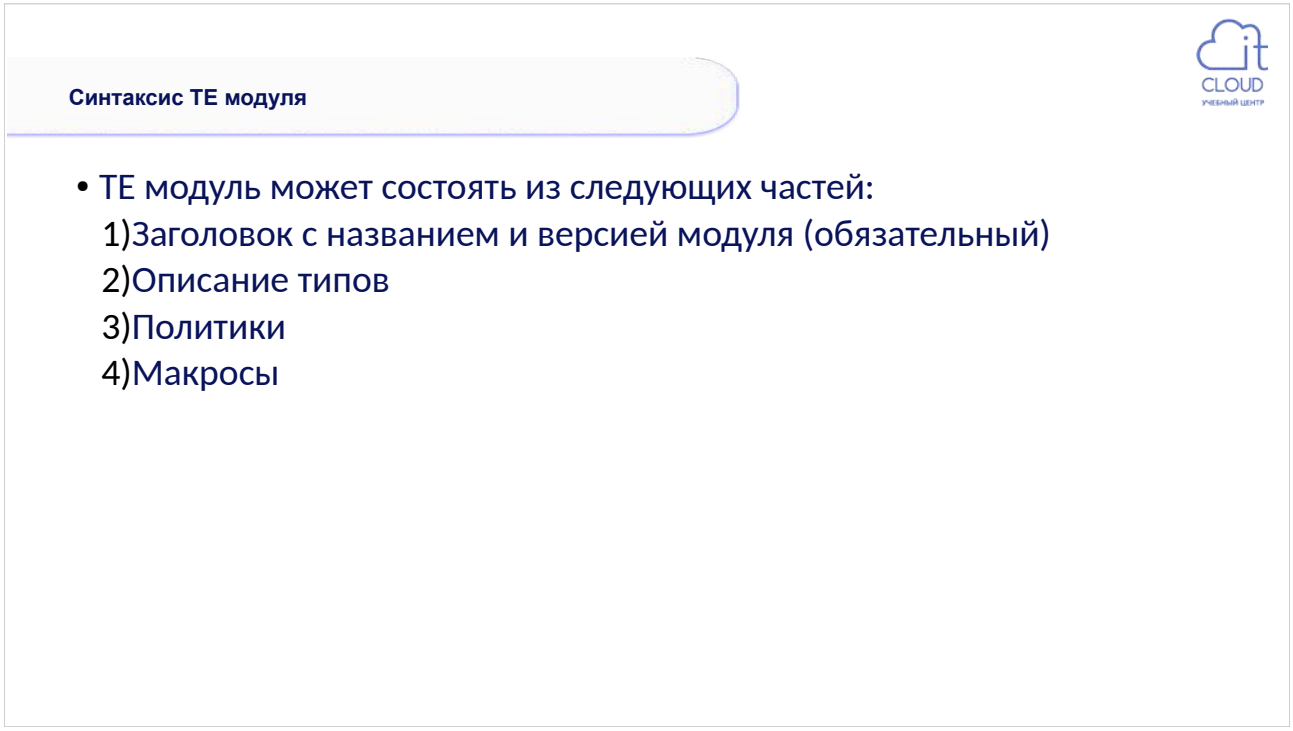


Модуль SELinux это бинарный файл. Создание этого файла основано на использовании трех текстовых файлов. Не все три файла требуется создавать для модуля.

1. Файл `.te` предназначен для описания используемых типов и описания правил.
2. Файл `.fc` описывает маркировку файлом этим модулем.
3. Файл `.if` создает интерфейс взаимодействия с модулем, на который могут ссылаться другие модули. В файле описываются макросы.

На сайте https://selinuxproject.org/page/RefpolicyWriteModule можно посмотреть принципы создания модулей.

Для создания собственных модулей вам потребуется пакеты selinux-policy-dev (в Debian) или selinux-policy-devel (для RedHat). Так же полезно будет установить пакет и selinux-policy-doc.



Каждый модуль `.te` может состоять из четырех составных частей:

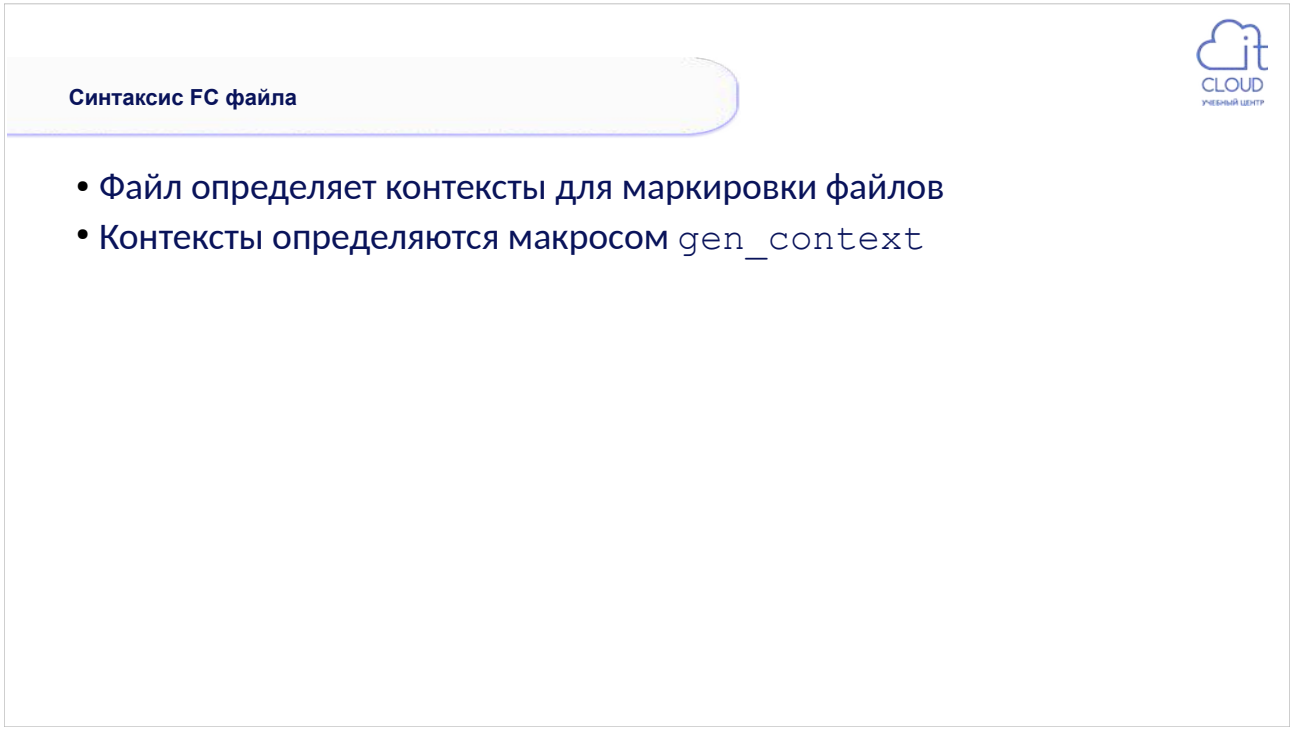
1. Заголовок. Обязательная часть, которая задает уникальное имя модуля и версию. Версия позволяет отслеживать изменения в модуле, а так же ядру понимать, что модуль изменился и надо загрузить его новую версию.
2. Описание типов. Эти строки описывают все возможные типы или атрибуты нашего ограниченного сервиса. Необязательный компонент, если у сервиса нет собственных типов.
3. Политики. Описание правил работы сервиса. Необязательный если нет особых правил.
4. Макросы. Используют политики определенные в других модулях (через файл `.if`). Необязательный.

Пример:Модуль некого приложения с названием myapp. В файле определяется название модуля — myapp и его версия — 1.0.0. Затем идет декларирование типов, как напрямую строками типа type, так и макросами. Последняя часть — политика. Политика описывается напрямую и через макросы.

```
root@s10:~# cat /usr/share/doc/selinux-policy/example.te

policy_module(myapp,1.0.0)

########################################
#
# Declarations
#

type myapp_t;
type myapp_exec_t;
```

```
domain_type(myapp_t)
domain_entry_file(myapp_t, myapp_exec_t)

type myapp_log_t;
logging_log_file(myapp_log_t)

type myapp_tmp_t;
files_tmp_file(myapp_tmp_t)

########################################
#
# Myapp local policy
#

allow myapp_t myapp_log_t:file { read_file_perms append_file_perms };

allow myapp_t myapp_tmp_t:file manage_file_perms;
files_tmp_filetrans(myapp_t,myapp_tmp_t,file)
```

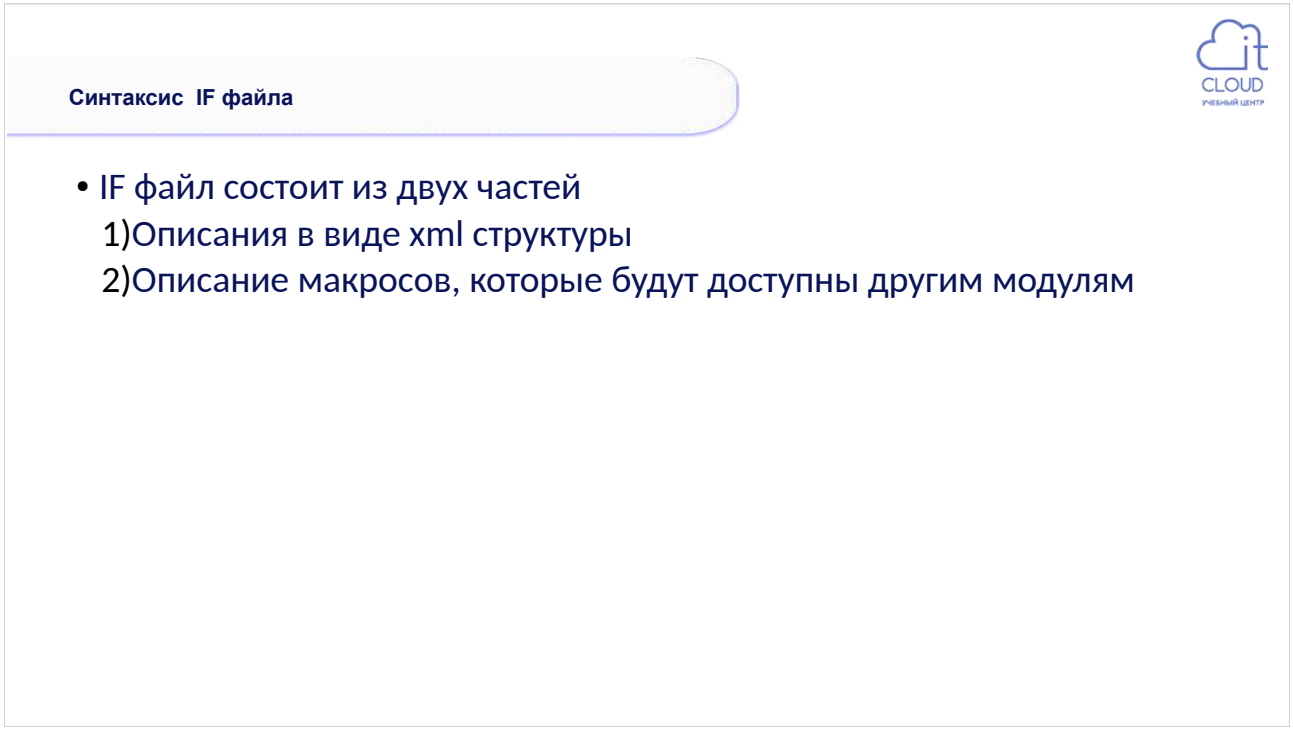
Описание макросов можно посмотреть в каталоге `/usr/share/selinux/devel/include/`, если установлен пакет `selinux-policy-devel`



Файл `.fc` определяет контексты, которые будут использованы для маркировки файлов, необходимых для работы сервиса.

Пример: Назначение контекста на файл программы.

```
root@s10:~# cat /usr/share/doc/selinux-policy/example.fc
# myapp executable will have:
# label: system_u:object_r:myapp_exec_t
# MLS sensitivity: s0
# MCS categories: <none>

/usr/sbin/myapp         --    gen_context(system_u:object_r:myapp_exec_t,s0)
```

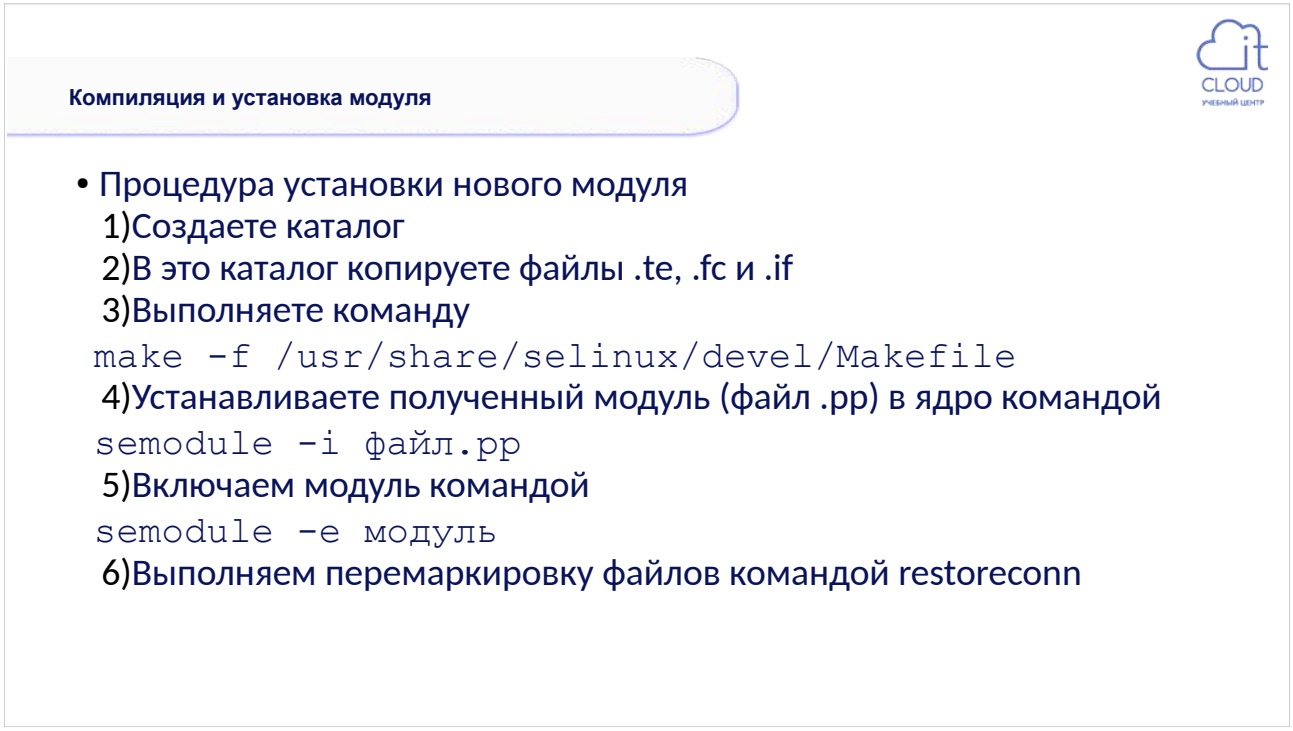


В файле `.if` описываются макросы, которые другие модули могут использовать для получения доступа к сервису.

Пример: Описывается два макроса: `myapp_domtrans`, который дает право переходить в домен `myapp_t` и `myapp_read_log`, который дает возможность просматривать журналы приложения. Обратите внимание, что в каждом макросе явно описываются используемые типы.

```
root@s10:~# cat /usr/share/doc/selinux-policy/example.if
## <summary>Myapp example policy</summary>
## <desc>
##     <p>
##           More descriptive text about myapp.  The desc
##           tag can also use p, ul, and ol
##           html tags for formatting.
##     </p>
##     <p>
##           This policy supports the following myapp features:
##           <ul>
##           <li>Feature A</li>
##           <li>Feature B</li>
##           <li>Feature C</li>
##           </ul>
##     </p>
## </desc>
#

########################################
## <summary>
##     Execute a domain transition to run myapp.
## </summary>
## <param name="domain">
##     <summary>
```

```
##    Domain allowed to transition.
##    </summary>
## </param>
#
interface(`myapp_domtrans',`
      gen_require(`
            type myapp_t, myapp_exec_t;
      ')

      domtrans_pattern($1,myapp_exec_t,myapp_t)
')

########################################
## <summary>
##    Read myapp log files.
## </summary>
## <param name="domain">
##    <summary>
##    Domain allowed to read the log files.
##    </summary>
## </param>
#
interface(`myapp_read_log',`
      gen_require(`
            type myapp_log_t;
      ')

      logging_search_logs($1)
      allow $1 myapp_log_t:file read_file_perms;
')
```



После создания модуля. Его необходимо скомпилировать и установить. Процедура установки модуля следующая:

1. Создаете каталог.
2. В это каталог копируете файлы `.te`, `.fc` и `.if`
3. Выполняете команду
4. `make -f /usr/share/selinux/devel/Makefile`
5. Устанавливаете полученный модуль (файл .pp) в ядро командой `semodule -i` *`файл.pp`*
6. Включаем модуль командой `semodule -e` *`модуль`*
7. Выполняем перемаркировку файлов командой `restoreconn`

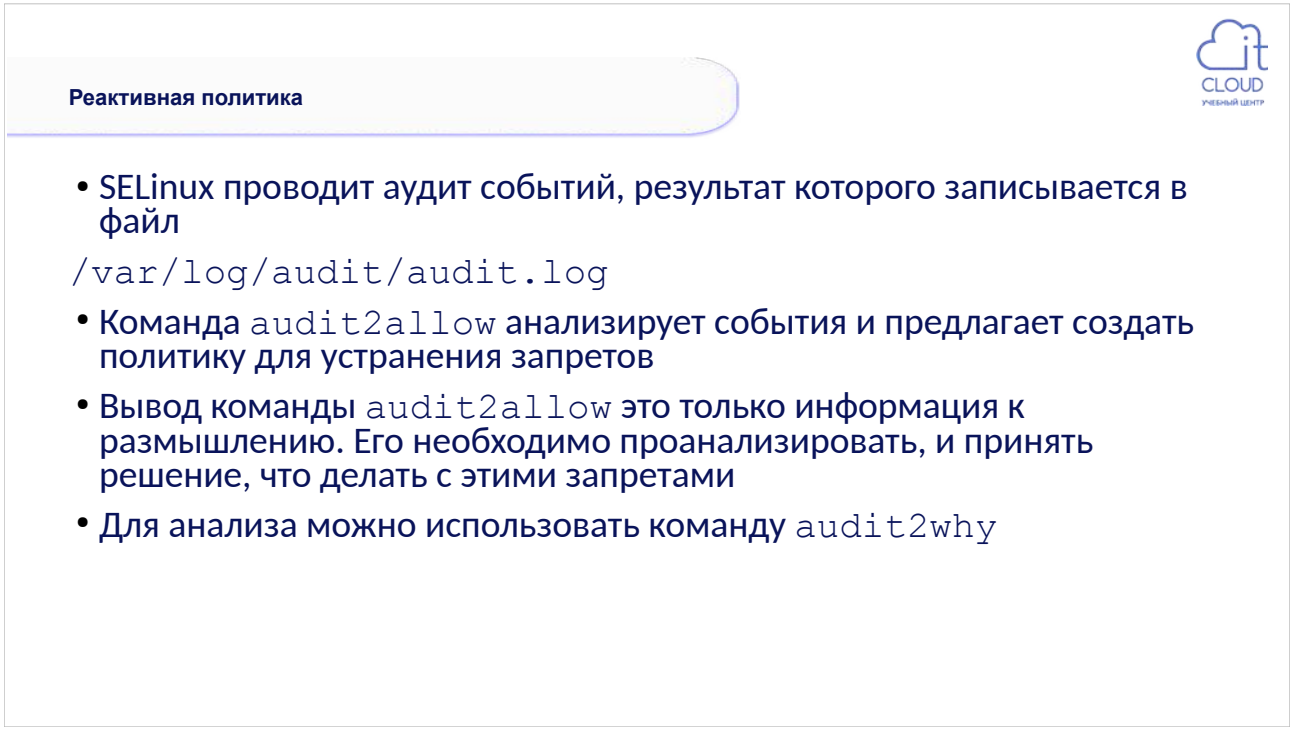
Пример:Установка нового модуля.

```
root@s10:~# mkdir myapp
root@s10:~# cp /usr/share/doc/selinux-policy/example.* myapp/
root@s10:~# cd myapp/
root@s10:myapp# rename example myapp example.*
root@s10:myapp# ls
myapp.fc  myapp.if  myapp.te
root@s10:myapp# make -f /usr/share/selinux/devel/Makefile
Compiling targeted myapp module
/usr/bin/checkmodule:  loading policy configuration from tmp/myapp.tmp
/usr/bin/checkmodule:  policy configuration loaded
/usr/bin/checkmodule:  writing binary representation (version 17) to
tmp/myapp.mod
Creating targeted myapp.pp policy package
rm tmp/myapp.mod.fc tmp/myapp.mod
root@s10:myapp# semodule -i myapp.pp
root@s10:myapp# semodule -e myapp
```

Удаление ненужного модуля производится командой `semodule -d` *модуль*

Пример:

```
root@s10:myapp# semodule -l | grep myapp
myapp 1.0.0
root@s10:myapp# semodule -d myapp
root@s10:myapp# semodule -l | grep myapp
```



При использовании SELinux производится аудит событий. В результате вы можете увидеть, как блокируется доступ к тем или иным функциям ОС. События записываются в журнал `/var/log/audit/audit.log`.

Выбрав из журнала интересующие вас события мы можем сформировать политику, которая устранит блокировки. Для этого можно воспользоваться командой `audit2allow`, которая анализирует события типа denied и на их основе формирует предлагаемую политику.

Пример: Формирование политики на основе анализа событий по метке httpd_t.

```
root@s10:~# grep httpd_t /var/log/audit/audit.log | audit2allow


#============= httpd_t ==============
allow httpd_t home_root_t:file { getattr map open read };

#!!!! This avc can be allowed using the boolean 'httpd_read_user_content'
allow httpd_t user_home_t:file { getattr map open read };
```

В общем случае применять такие политики **ПЛОХАЯ ИДЕЯ**!!!

Лучше проанализировать этот вывод и принять решение что необходимо сделать. Для подробного анализа событий можно воспользоваться командой `audit2why`, которая показывает почему возникла ошибка и предлагает возможные решения.

Пример: Разбор событий командой `audit2why`

```
root@s10:~# grep httpd_t /var/log/audit/audit.log | audit2why | tail -9
type=AVC msg=audit(1738673843.372:305): avc:  denied  { getattr } for  pid=2137
comm="apache2" path="/home/myweb/index.html" dev="sda1" ino=131308
scontext=system_u:system_r:httpd_t:s0
tcontext=unconfined_u:object_r:user_home_t:s0 tclass=file permissive=0
```

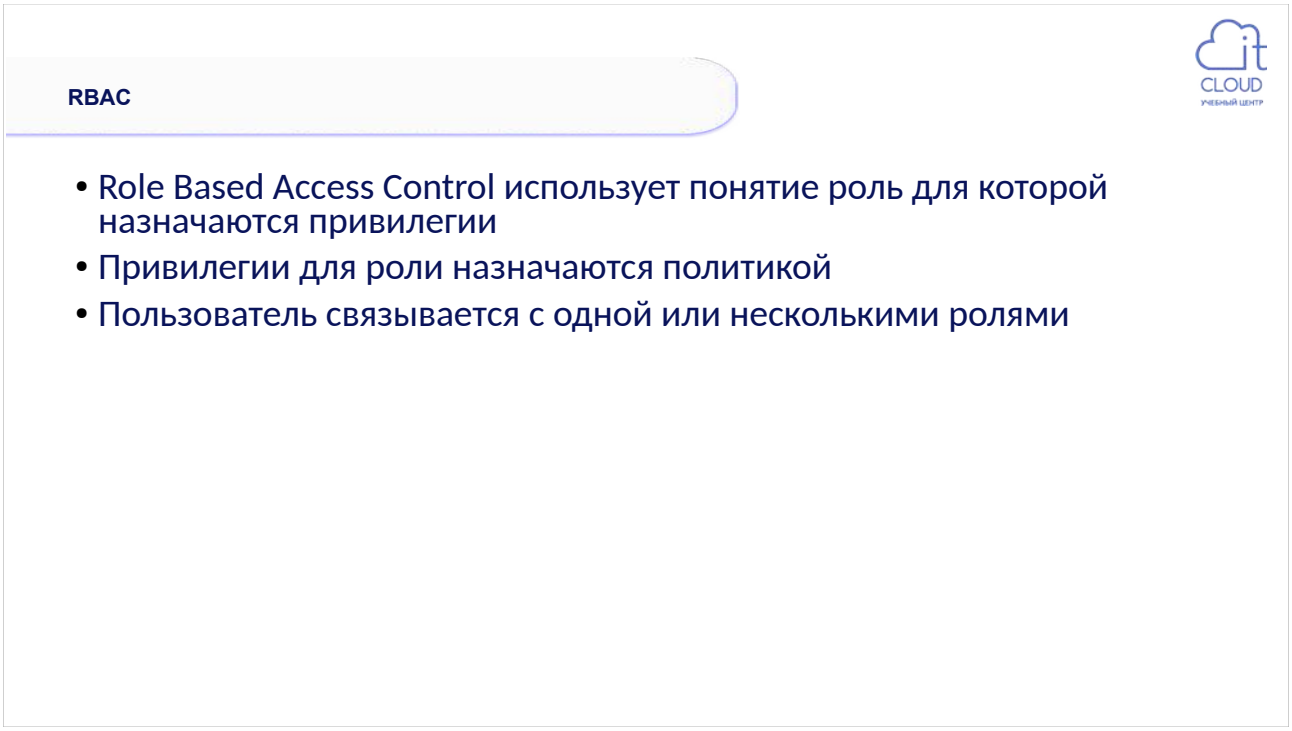
Анализ подразумевает что вы отвечаете на следующие вопросы:

1. Мешает ли блокировка правильной работе ограниченного сервиса?
2. Если сервис работает некорректно, то в чем причина?
    1. Неправильная политика?
    2. Неправильная маркировка файлов?
    3. Некорректные действия пользователей?

По результатам анализа мы можем принять следующие решения:

1. Принять события и ничего не делать. Это означает, что вы не просто блокируете доступ, но и отслеживаете блокировку. Например, чтобы отслеживать атаки на службу.
2. Модифицировать существующую политику, если решили, что политика не верна.
3. Создать новый модуль и установить его. Не создавайте модули автоматически, это может привести к бреши в системе безопасности. В примере выше создавать политику с правилом `allow httpd_t user_home_t:file { getattr map open read };` не лучшее решение. Правильным решением будет установить нужную маркировку файлов.
4. Запретить аудит блокировки таких событий. Тогда вместо `allow` необходимо создать модуль с параметром `dontaudit` или использовать опцию `-D` в команде `audit2allow`

6.2.3 Реализация других форм контроля доступа с помощью SELinux.

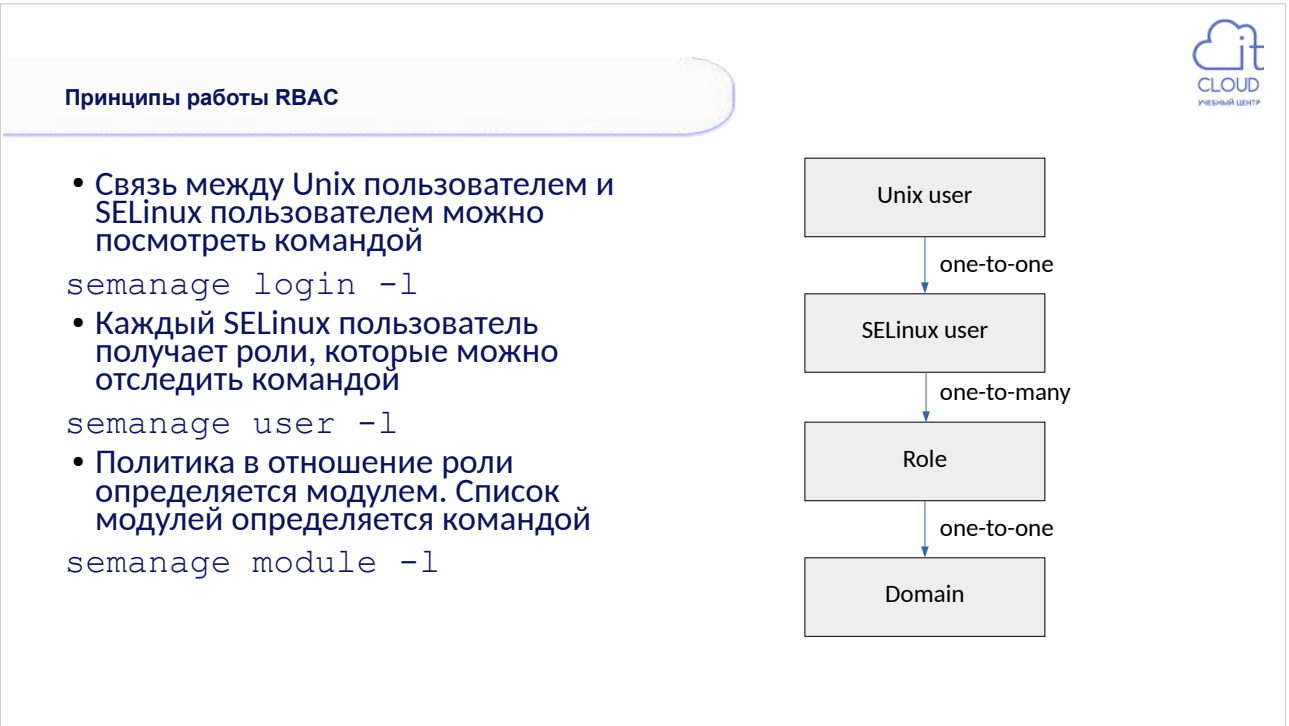


При работе с пользователями использование только меток для процессов и файлов является недостаточным. Причин несколько:

1. Как правило мы не можем заранее предусмотреть какие программы и файлы будет использовать конкретный пользователь.
2. Если мы знаем какую-то программу, которую может использовать пользователь, например `nano`, то не можем с достаточной долей уверенности сказать, как пользователь будет использовать эту программу. Что он будет редактировать?
3. Запуская одну и ту же программу разным пользователям надо разрешать разный тип доступа. Задачи у каждого пользователя свои.
4. В целевой политике процесс получает метку от файла программы, но это совершенно не подходит под принцип разделения доступа на основе пользовательской информации.

Для решения задач по управлению доступом пользователей мы можем воспользоваться системой RBAC (Role Based Access Control). В этой системе каждый пользователь Unix связывается с SELinux пользователем. SELinux пользователь получает список своих ролей.

Роль определяет полномочия ее обладателя. Управление ролями настраивается посредством политики.



По умолчанию система RBAC не используется. Все пользователи являются неограниченными, или, другими словами используют систему DAC.

Пример: Пользователю root назначен пользователь `unconfined_u` (неограниченный) у которого назначена роль `unconfined_r`. Метка домена пользователя `unconfined_t`.

```
root@s10:myapp# id -Z
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
root@s10:myapp# ps -Z
LABEL                                PID TTY          TIME CMD
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 1055 ttyS0 00:00:07 bash
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 3074 ttyS0 00:00:00 ps
```

Связь Unix и SELinux пользователей можно посмотреть командой `semanage login -l`. Эта команда так же показывает SELinux пользователя по умолчанию, это назначение используется для Unix пользователей, которым не привязан SELinux пользователь.

Пример: SELinux пользователь по умолчанию `unconfined_u`

```
root@s10:myapp# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

Назначенные роли SELinux пользователям можно выяснить командой `semanage user -l`.

Пример: В примере пользователю `staff_u` назначены роли `staff_r sysadm_r system_r unconfined_r`.

```
root@s10:myapp# semanage user -l

                Labeling   MLS/       MLS/
SELinux User    Prefix     MCS Level  MCS Range                      SELinux
Roles

guest_u         user       s0         s0                             guest_r
root            user       s0         s0-s0:c0.c1023                 staff_r
sysadm_r system_r unconfined_r
staff_u         user       s0         s0-s0:c0.c1023                 staff_r
sysadm_r system_r unconfined_r
sysadm_u        user       s0         s0-s0:c0.c1023                 sysadm_r
system_u        user       s0         s0-s0:c0.c1023                 system_r
unconfined_r
unconfined_u    user       s0         s0-s0:c0.c1023                 system_r
unconfined_r
user_u          user       s0         s0                             user_r
xguest_u        user       s0         s0                             xguest_r
```

Политика для каждой роли определяется в соответствующем модуле. Команда `semanage module -l` показывает список модуле.

Пример:

```
root@s10:myapp# semanage module -l | grep staff
staff                     100       pp
```

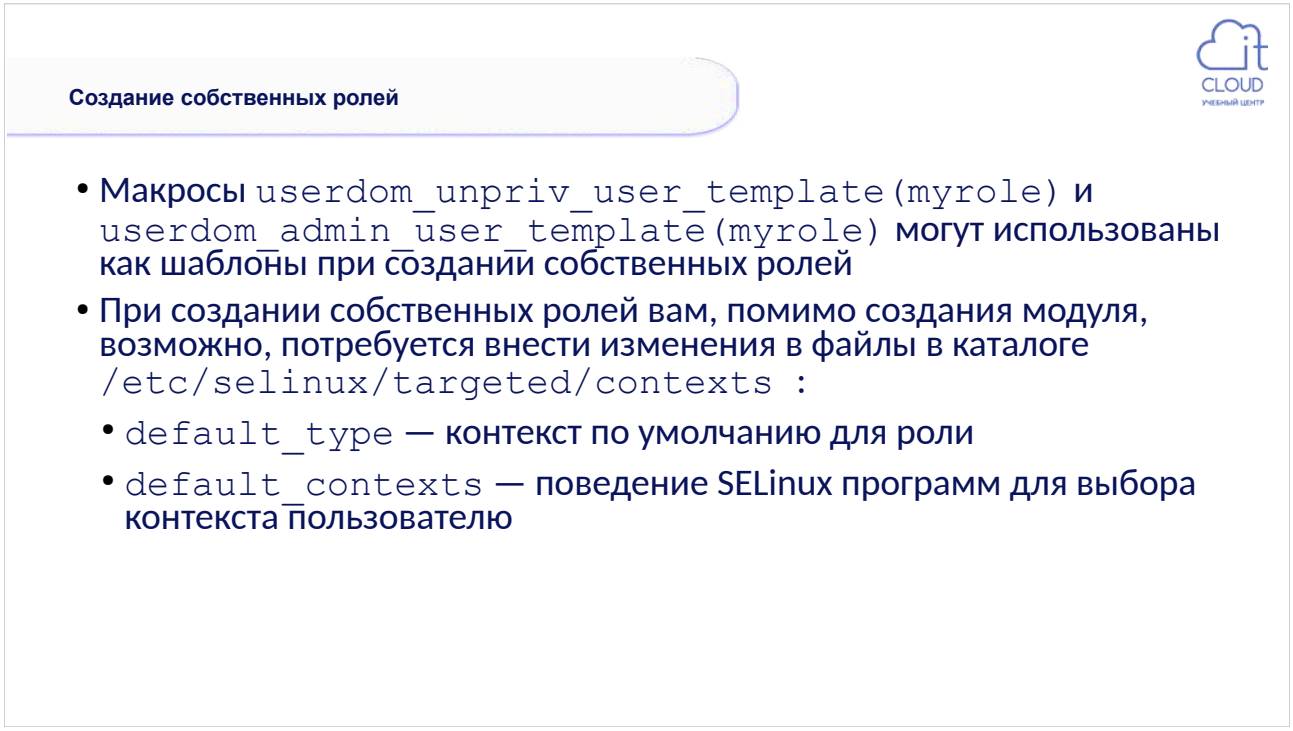
Роли по умолчанию

Роль	Модуль	Описание
user_r	unprivuser	Базовая роль для пользователей. Может выполнять большинство операций непривилегированного пользователя.
staff_r	staff	Администраторская непривилегированная роль.
sysadm_r	sysadm	Стандартный администратор.
secadm_r	secadm	Администратор политик безопасности.
auditadm_r	auditadm	Может настраивать политику аудита и проводить аудит событий.
logadm_r	logadm	Настройка и управление журналами.
webadm_r	webadm	Настройка веб-сервера apache и, опционально, управление контентом веб-сервера.
guest_r	guest	Сильно ограниченный пользователь. Без поддержки GUI.
xguest_r	xguest	Сильно ограниченный пользователь с поддержкой GUI.
unconfined_r	unconfined	Неограниченный пользователь, за исключением защиты памяти.

В SELinux определено несколько стандартных ролей.

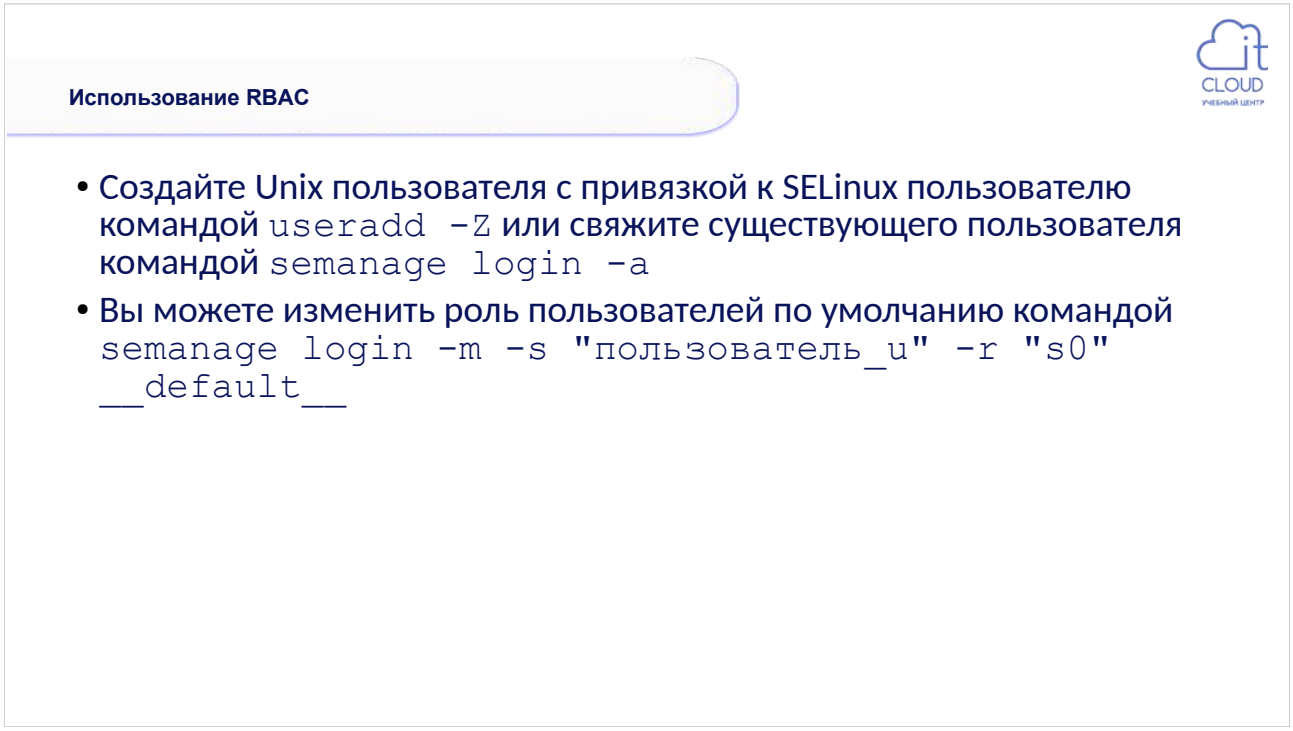
Каждая роль имеет индивидуальный модуль.

Вы можете использовать эти роли для выполнения стандартных системных действий.



Если существующие роли не удовлетворяют ваших потребностей, то вы можете создать свою собственную роль. Для этого вам потребуется:

1. Создать модуль SELinux с описанием политики для вашей роли. При создании модуля вы можете воспользоваться макросами-шаблонами
    1. `userdom_unpriv_user_template(`*`myrole`*`)`: макрос для определения роли похожей на `user_r` и `staff_r`.
    2. `userdom_admin_user_template(`*`myrole`*`)`: макрос для определения роли похожей на `sysadm_r`.
2. Внести изменения в файлы в каталоге `/etc/selinux/targeted/contexts`:
    1. `default_type` — контекст по умолчанию для роли
    2. `default_contexts` — поведение SELinux программ для выбора контекста пользователю.



Создать Unix пользователя с желаемой привязкой к SELinux пользователю можно командой `useeradd` с опцией `-Z`.

Если вы хотите сопоставить существующих пользователей, то нужно использовать команду `semanage login -a`.

Пример: Создание Unix пользователя с привязкой к SELinux пользователю.

```
root@s10:myapp# useradd -Z staff_u staffuser
root@s10:myapp# passwd staffuser
Changing password for user staffuser.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
root@s10:myapp# gpasswd -a staffuser wheel
Adding user staffuser to group wheel
root@s10:myapp# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
staffuser            staff_u              s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

Пример: Проверка возможностей ограниченного пользователя.

```
root@s10:myapp# ssh staffuser@127.0.0.1
staffuser@127.0.0.1's password:
[staffuser@sl0 ~]$ id -Z
staff_u:staff_r:staff_t:s0-s0:c0.c1023
[staffuser@sl0 ~]$ ps -Z
LABEL                             PID TTY          TIME CMD
staff_u:staff_r:staff_t:s0-s0:c0.c1023 4059 pts/0 00:00:00 bash
staff_u:staff_r:staff_t:s0-s0:c0.c1023 4080 pts/0 00:00:00 ps
```

```
[staffuser@sl0 ~]$ su -
Password: Правильный_пароль
su: Authentication failure
[staffuser@sl0 ~]$ sudo su -
[sudo] password for staffuser:
su: avc.c:74: avc_context_to_sid_raw: Assertion `avc_running' failed.
[staffuser@sl0 ~]$ sudo -i
-bash: /root/.bash_profile: Permission denied
-bash-4.2# id -Z
staff_u:staff_r:staff_t:s0-s0:c0.c1023
bash-4.2# passwd root
passwd: SELinux denying access due to security policy.
bash-4.2# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=52 time=30.5 ms
#
--- 8.8.8.8 ping statistics ---
2 packets transmitted, 1 received, 50% packet loss, time 1001ms
rtt min/avg/max/mdev = 30.590/30.590/30.590/0.000 ms
bash-4.2# wget http://www.google.com -O /dev/null
--2017-08-02 20:53:56--  http://www.google.com/
Resolving www.google.com (www.google.com)... 173.194.221.103, 173.194.221.105,
173.194.221.106, ...
Connecting to www.google.com (www.google.com)|173.194.221.103|:80... connected.
HTTP request sent, awaiting response... 302 Found
Location: http://www.google.ru/?gfe_rd=cr&ei=lfWBWbe4DsyG7gTYn5u4Aw [following]
--2017-08-02 20:53:56--  http://www.google.ru/?
gfe_rd=cr&ei=lfWBWbe4DsyG7gTYn5u4Aw
Resolving www.google.ru (www.google.ru)... 173.194.222.94,
2a00:1450:4010:c0b::5e
Connecting to www.google.ru (www.google.ru)|173.194.222.94|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/html]
Saving to: ‘/dev/null’

    [ <=>                                   ] 10,960      --.-K/s   in 0.001s

2017-08-02 20:53:56 (7.81 MB/s) - ‘/dev/null’ saved [10960]
-bash-4.2# logout
```

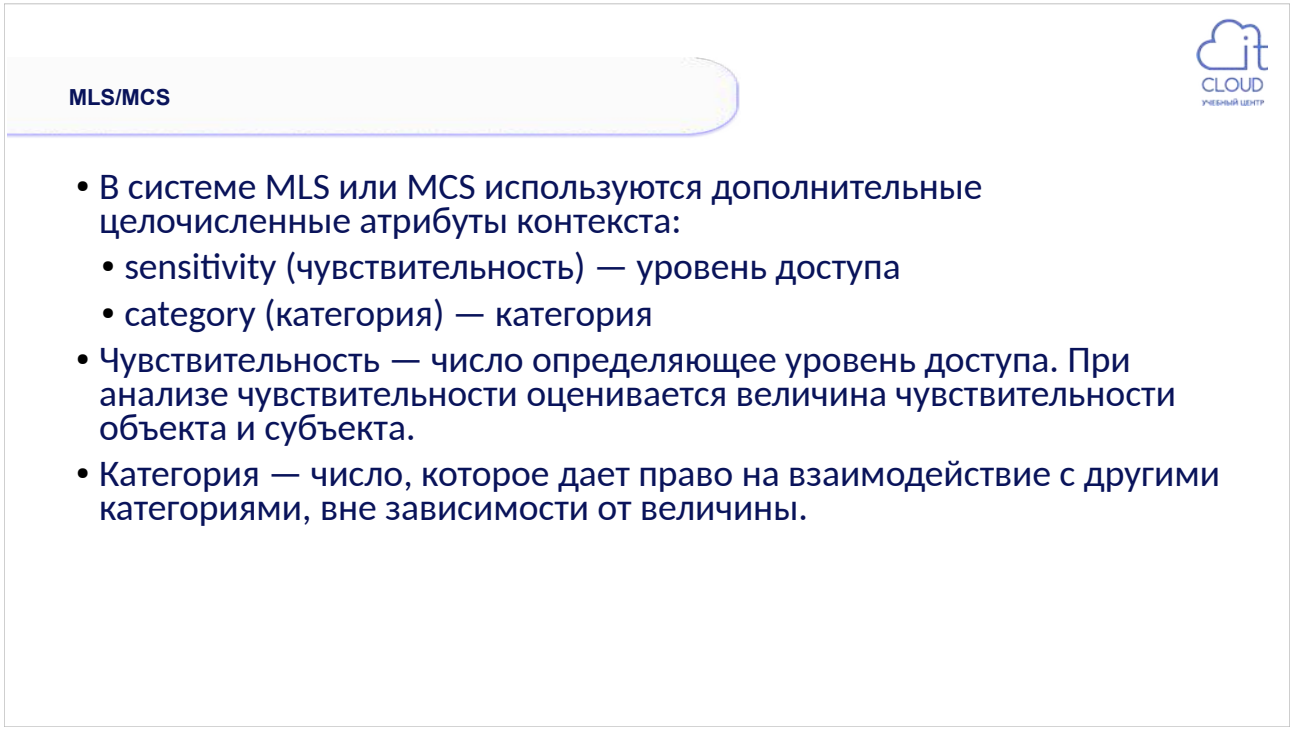
Переключение на новую роль

```
[staffuser@sl0 ~]$ sudo -r sysadm_r -i
root@s10:~# id -Z
staff_u:sysadm_r:sysadm_t:s0-s0:c0.c1023
root@s10:~# passwd root
Changing password for user root.
New password:
BAD PASSWORD: The password is shorter than 7 characters
Retype new password:
passwd: all authentication tokens updated successfully.
root@s10:~# ping 8.8.8.8 -c1
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=52 time=30.8 ms

--- 8.8.8.8 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
```

```
rtt min/avg/max/mdev = 30.802/30.802/30.802/0.000 ms
root@s10:~# wget http://www.google.com -O /dev/null
--2017-08-02 20:53:31--  http://www.google.com/
Resolving www.google.com (www.google.com)... 173.194.222.105, 173.194.222.103,
173.194.222.147, ...
Connecting to www.google.com (www.google.com)|173.194.222.105|:80... connected.
HTTP request sent, awaiting response... 302 Found
Location: http://www.google.ru/?gfe_rd=cr&ei=fPWBWZPSE6rG7gTF0re4Dg [following]
--2017-08-02 20:53:31--  http://www.google.ru/?
gfe_rd=cr&ei=fPWBWZPSE6rG7gTF0re4Dg
Resolving www.google.ru (www.google.ru)... 173.194.221.94,
2a00:1450:4010:c0b::5e
Connecting to www.google.ru (www.google.ru)|173.194.221.94|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/html]
Saving to: ‘/dev/null’

    [ <=>                                   ] 10,944      --.-K/s   in 0s

2017-08-02 20:53:31 (98.7 MB/s) - ‘/dev/null’ saved [10944]

root@s10:~# logout
```

Для сопоставления существующего Unix пользователя с SELinux пользователем можно использовать команду `semanage login -m`:

```
root@s10:myapp# semanage login -m -s "user_u" -r "s0" __default__
root@s10:myapp# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          user_u               s0                   *
root                 unconfined_u         s0-s0:c0.c1023       *
staffuser            staff_u              s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

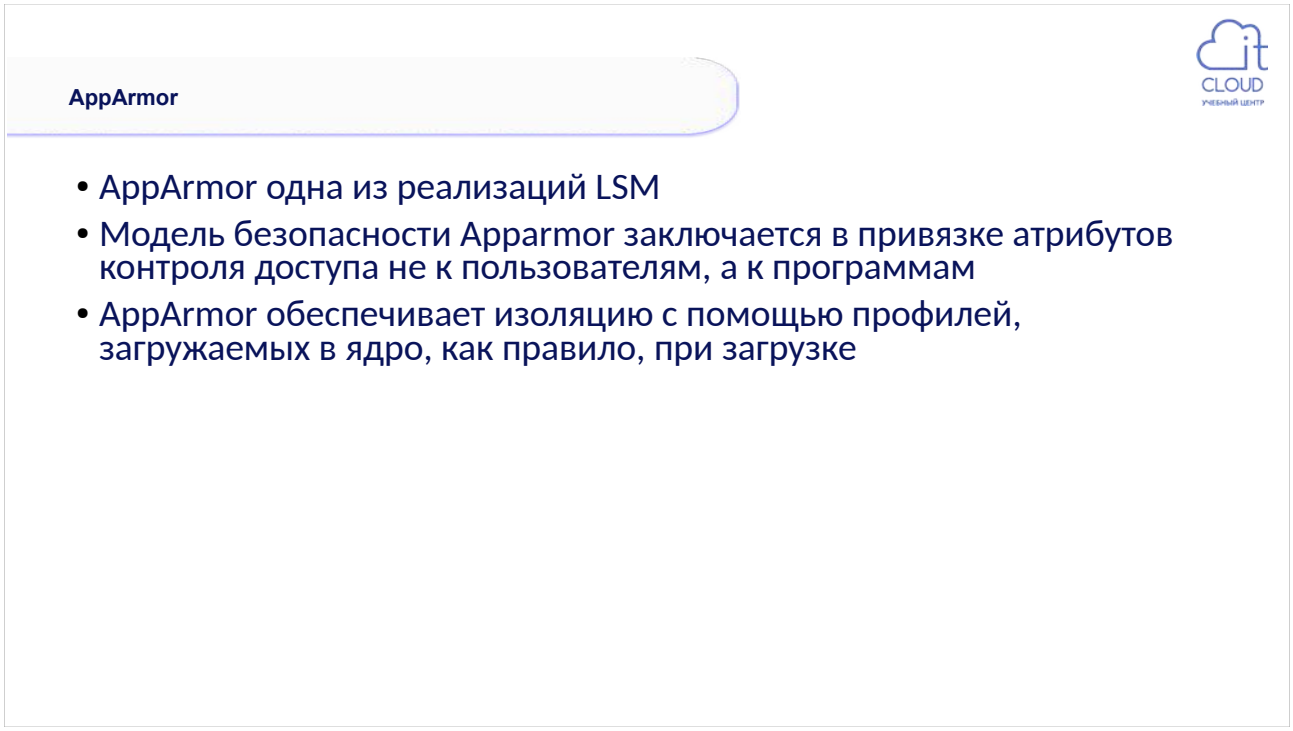


Системы MLS или MCS используют, в дополнение к меткам, еще и числовой анализ при предоставлении доступа.

Метка чувствительность (sensitivity) задает уровень доступа. Общая идея при использовании чувствительности: некий уровень может прочитать информацию уровнем ниже своего, но не может ее изменить; на своем уровне пользователь может читать и изменять данные; на уровень выше своего может только изменять (предоставлять) информацию, но не читать.

Использование категорий показывает какой диапазон категорий может получить доступ к другим категориям.

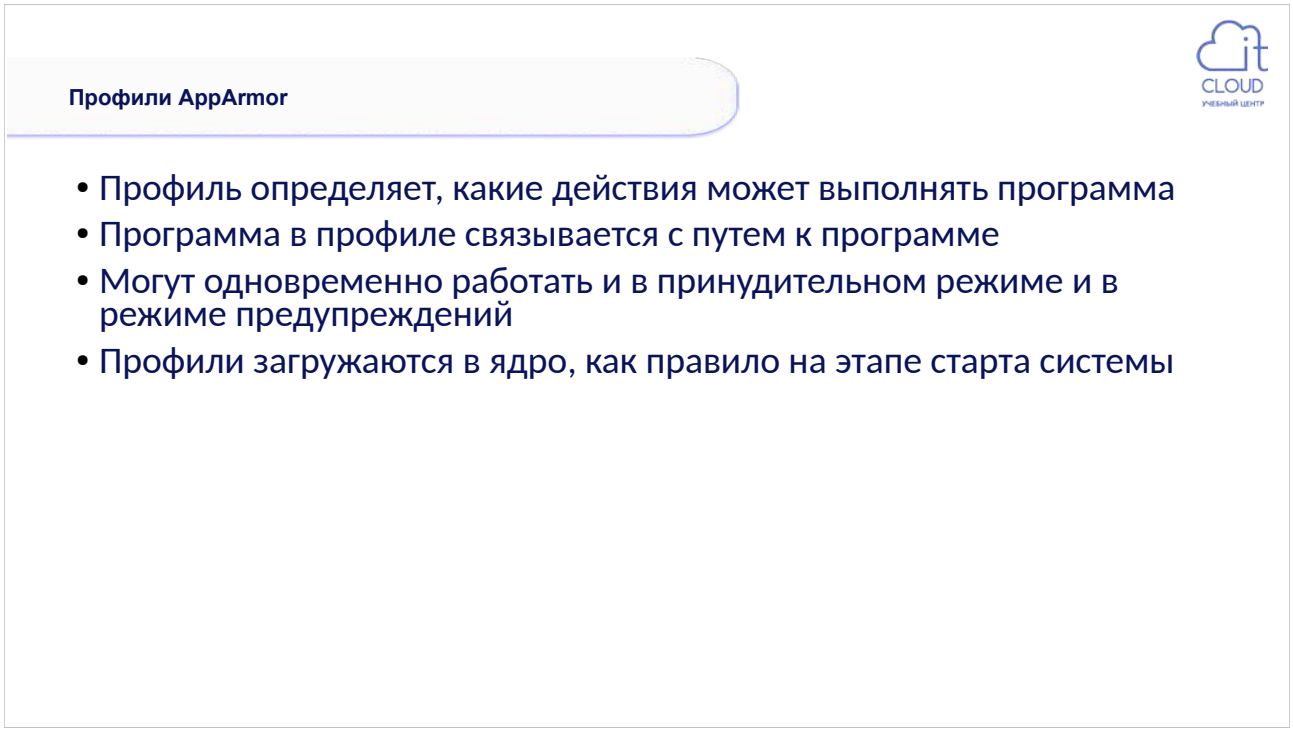
6.3 AppArmor.



Другой популярной реализацией LSM являет AppArmor.

Модель безопасности Apparmor заключается в привязке атрибутов контроля доступа не к пользователям, а к программам.

AppArmor обеспечивает изоляцию с помощью профилей, загружаемых в ядро, как правило, при загрузке.



Основой политики AppArmor является профиль, который определяет какие действия может выполнять программа. В профиле указывается по каким путям может быть расположена данная программа.

Также профиль определяет режим работы: принудительный (Enforce) или режим предупреждений (Complain). При этом одновременно некоторые профили могут работать в принудительном режиме, а другие в режиме предупреждений.

Профиль описывается в файле, который располагается в каталоге `/usr/share/apparmor/extra-profiles/`, что бы профиль сделать активным его следует перенести в каталог `/etc/apparmor.d/`.

Пример:

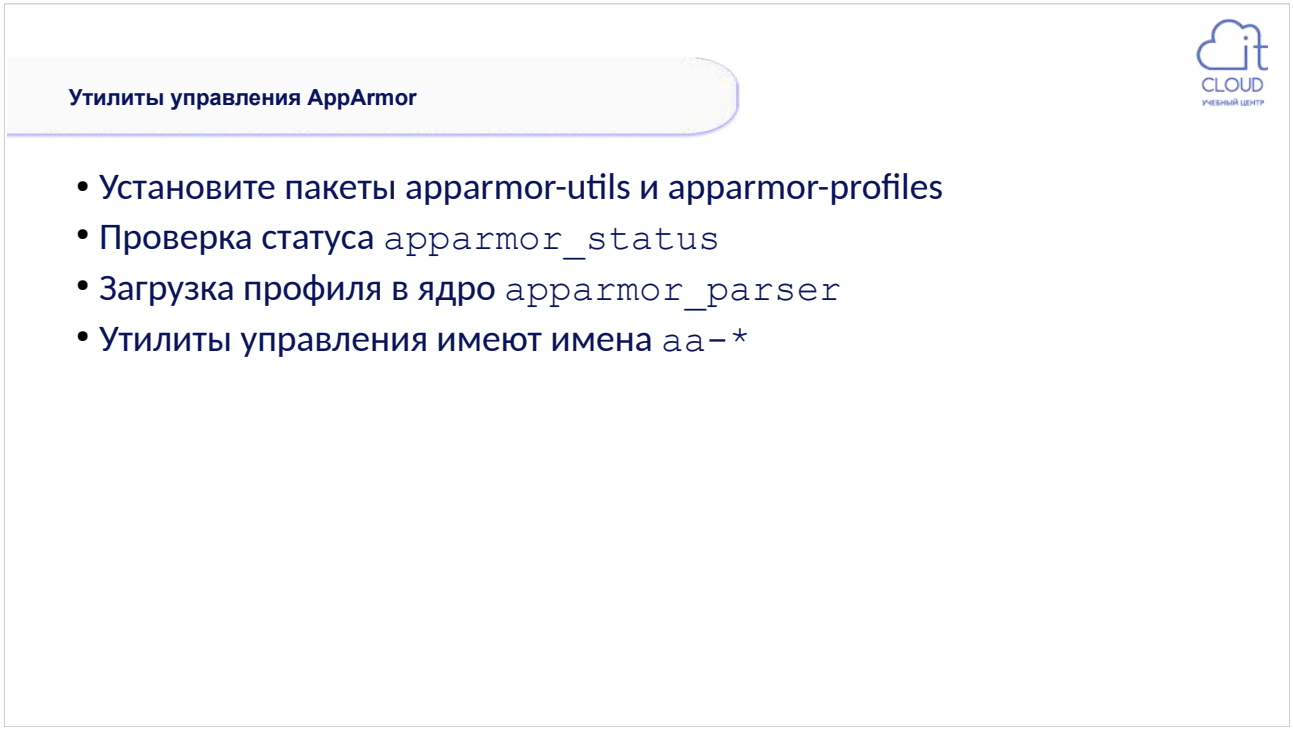
```
# ls /usr/share/apparmor/extra-profiles/ | wc -l
118

# ls /etc/apparmor.d/ | wc -l
28

# cat /etc/apparmor.d/bin.ping
# ------------------------------------------------------------------
#
#    Copyright (C) 2002-2009 Novell/SUSE
#    Copyright (C) 2010 Canonical Ltd.
#
#    This program is free software; you can redistribute it and/or
#    modify it under the terms of version 2 of the GNU General Public
#    License published by the Free Software Foundation.
#
# ------------------------------------------------------------------
```

```
#include <tunables/global>
profile ping /{usr/,}bin/{,iputils-}ping flags=(complain) {
  #include <abstractions/base>
  #include <abstractions/consoles>
  #include <abstractions/nameservice>

  capability net_raw,
  capability setuid,
  network inet raw,
  network inet6 raw,

  /{,usr/}bin/{,iputils-}ping mixr,
  /etc/modules.conf r,

  # Site-specific additions and overrides. See local/README for details.
  #include <local/bin.ping>
}
```

В профиле описываются разрешения для программы, capabilities и флаги.

Значения флагов:

- r — чтение;
- w — запись
- a — инкрементальная запись в конец файла, от английского append;
- k — блокировка файлов;
- l — создание символических ссылок на исполняемые файлы;
- m — загрузка исполняемых файлов в память;
- cx — переход в профиль нижнего уровня при выполнении;
- Cx — переход в профиль нижнего уровня при выполнении с очисткой переменных окружения;
- ix — наследование исполнения;
- px — требуется определение дискретного профиля безопасности для ресурса;
- Px — требуется определение дискретного профиля безопасности для ресурса, производится очистка переменных окружения;
- ux — не проверять запуск новых процессов;
- Ux — не проверять запуск новых процессов и производить очистку переменных окружения;



Команда `apparmor_status` (`aa-status`) показывает текущее состояние AppArmor.

Для загрузки профиля используется команда `apparmor_parser`.

Команды с именами начинающимися с aa- предназначены для управления работой AppArmor.

Пример:

```
# aa-enabled
Yes

# aa-status
apparmor module is loaded.
43 profiles are loaded.
25 profiles are in enforce mode.
   /usr/bin/evince
   /usr/bin/evince-previewer
   /usr/bin/evince-previewer//sanitized_helper
   /usr/bin/evince-thumbnailer
   /usr/bin/evince//sanitized_helper
   /usr/bin/man
   /usr/bin/pidgin
   /usr/bin/pidgin//sanitized_helper
   /usr/bin/totem
   /usr/bin/totem-audio-preview
   /usr/bin/totem-video-thumbnailer
   /usr/bin/totem//sanitized_helper
   /usr/lib/cups/backend/cups-pdf
   /usr/sbin/cups-browsed
   /usr/sbin/cupsd
   /usr/sbin/cupsd//third_party
   apt-cacher-ng
```

```
   libreoffice-senddoc
   libreoffice-soffice//gpg
   libreoffice-xpdfimport
   lsb_release
   man_filter
   man_groff
   nvidia_modprobe
   nvidia_modprobe//kmod
18 profiles are in complain mode.
   /usr/bin/irssi
   /usr/sbin/dnsmasq
   /usr/sbin/dnsmasq//libvirt_leaseshelper
   avahi-daemon
   identd
   klogd
   libreoffice-oopslash
   libreoffice-soffice
   mdnsd
   nmbd
   nscd
   ping
   smbd
   smbldap-useradd
   smbldap-useradd///etc/init.d/nscd
   syslog-ng
   syslogd
   traceroute
3 processes have profiles defined.
3 processes are in enforce mode.
   /usr/sbin/cups-browsed (16403)
   /usr/sbin/cupsd (16397)
   /usr/lib/cups/notifier/dbus (16424) /usr/sbin/cupsd
0 processes are in complain mode.
0 processes are unconfined but have a profile defined.

# aa-enforce ping
Setting /usr/bin/ping to enforce mode.
Warning: profile ping represents multiple programs

$ ping -c2 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=109 time=41.8 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=109 time=41.5 ms

--- 8.8.8.8 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 41.508/41.673/41.838/0.165 ms

$ sudo aa-status
[sudo] password for admuser:
apparmor module is loaded.
43 profiles are loaded.
26 profiles are in enforce mode.
   /usr/bin/evince
   /usr/bin/evince-previewer
   /usr/bin/evince-previewer//sanitized_helper
   /usr/bin/evince-thumbnailer
   /usr/bin/evince//sanitized_helper
   /usr/bin/man
```

```
   /usr/bin/pidgin
   /usr/bin/pidgin//sanitized_helper
   /usr/bin/totem
   /usr/bin/totem-audio-preview
   /usr/bin/totem-video-thumbnailer
   /usr/bin/totem//sanitized_helper
   /usr/lib/cups/backend/cups-pdf
   /usr/sbin/cups-browsed
   /usr/sbin/cupsd
   /usr/sbin/cupsd//third_party
   apt-cacher-ng
   libreoffice-senddoc
   libreoffice-soffice//gpg
   libreoffice-xpdfimport
   lsb_release
   man_filter
   man_groff
   nvidia_modprobe
   nvidia_modprobe//kmod
   ping
17 profiles are in complain mode.
<...>
```

Для создания профилей можно использовать команды `aa-autodep` и `aa-genprof`. См. https://gitlab.com/apparmor/apparmor/-/wikis/Profiles.

Пример:

```
$ cat ~/catsyslog.sh
#!/bin/bash

/bin/cat /var/log/syslog

$ ~/catsyslog.sh
/bin/cat: /var/log/syslog: Permission denied

$ sudo aa-autodep ./catsyslog.sh
[sudo] password for admuser:
Writing updated profile for /home/admuser/catsyslog.sh.

$ sudo cat /etc/apparmor.d/home.admuser.catsyslog.sh
# Last Modified: Sun May  1 19:30:05 2022
#include <tunables/global>

/home/admuser/catsyslog.sh flags=(complain) {
  #include <abstractions/base>
  #include <abstractions/bash>

  /home/admuser/catsyslog.sh r,
  /usr/bin/bash ix,

}

$ sudo aa-genprof /home/admuser/catsyslog.sh

Before you begin, you may wish to check if a
profile already exists for the application you
wish to confine. See the following wiki page for
more information:
```

```
https://gitlab.com/apparmor/apparmor/wikis/Profiles

Profiling: /home/admuser/catsyslog.sh

Please start the application to be profiled in
another window and exercise its functionality now.

Once completed, select the "Scan" option below in
order to scan the system logs for AppArmor events.

For each AppArmor event, you will be given the
opportunity to choose whether the access should be
allowed or denied.

[(S)can system log for AppArmor events] / (F)inish
Reading log entries from /var/log/audit/audit.log.
Updating AppArmor profiles in /etc/apparmor.d.

Profile:  /home/admuser/catsyslog.sh
Execute:  /usr/bin/cat
Severity: unknown

(I)nherit / (C)hild / (N)amed / (X) ix On / (D)eny / Abo(r)t / (F)inish
Complain-mode changes:

Profile:  /home/admuser/catsyslog.sh
Path:     /dev/tty
New Mode: rw
Severity: 9

 [1 - #include <abstractions/consoles>]
  2 - /dev/tty rw,
(A)llow / [(D)eny] / (I)gnore / (G)lob / Glob with (E)xtension / (N)ew / Audi(t)
/ Abo(r)t / (F)inish
Adding #include <abstractions/consoles> to profile.

= Changed Local Profiles =

The following local profiles were changed. Would you like to save them?

 [1 - /home/admuser/catsyslog.sh]
(S)ave Changes / Save Selec(t)ed Profile / [(V)iew Changes] / View Changes b/w
(C)lean profiles / Abo(r)t
Writing updated profile for /home/admuser/catsyslog.sh.

Profiling: /home/admuser/catsyslog.sh

Please start the application to be profiled in
another window and exercise its functionality now.

Once completed, select the "Scan" option below in
order to scan the system logs for AppArmor events.

For each AppArmor event, you will be given the
opportunity to choose whether the access should be
allowed or denied.

[(S)can system log for AppArmor events] / (F)inish
```

```
Reloaded AppArmor profiles in enforce mode.

Please consider contributing your new profile!
See the following wiki page for more information:
https://gitlab.com/apparmor/apparmor/wikis/Profiles

Finished generating profile for /home/admuser/catsyslog.sh.

$ sudo cat /etc/apparmor.d/home.admuser.catsyslog.sh
# Last Modified: Sun May  1 19:31:55 2022
#include <tunables/global>

/home/admuser/catsyslog.sh flags=(complain) {
  #include <abstractions/base>
  #include <abstractions/bash>
  #include <abstractions/consoles>

  /home/admuser/catsyslog.sh r,
  /usr/bin/bash ix,
  /usr/bin/cat mrix,

}

$ ls -l /var/log/syslog
-rw-r----- 1 root adm 57543 May  1 19:55 /var/log/syslog
```

Стоит заметить следующее AppArmor не отменят необходимости иметь дискреционные права доступа. AppArmor может наложить дополнительные ограничения, но не переопределить разрешения DAC.

Пример: Рассмотрим другой пример использования AppArmor. Заметьте, что для программ, для которых нет профиля действует DAC. AppArmor начинает работу, только после того как будет создан профиль программы.

```
$ sudo cp -a /bin/ping ping
$ sudo getcap ping
ping cap_net_raw=ep

$ ./ping -c1 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.065 ms

--- 127.0.0.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.065/0.065/0.065/0.000 ms

$ sudo aa-autodep /home/admuser/ping
Writing updated profile for /home/admuser/ping.

$ sudo cat /etc/apparmor.d/home.admuser.ping
# Last Modified: Tue May  3 11:36:43 2022
#include <tunables/global>

/home/admuser/ping flags=(complain) {
  #include <abstractions/base>

  /home/admuser/ping mr,
```

```
}

$ ./ping -c1 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.051 ms

--- 127.0.0.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.051/0.051/0.051/0.000 ms
```

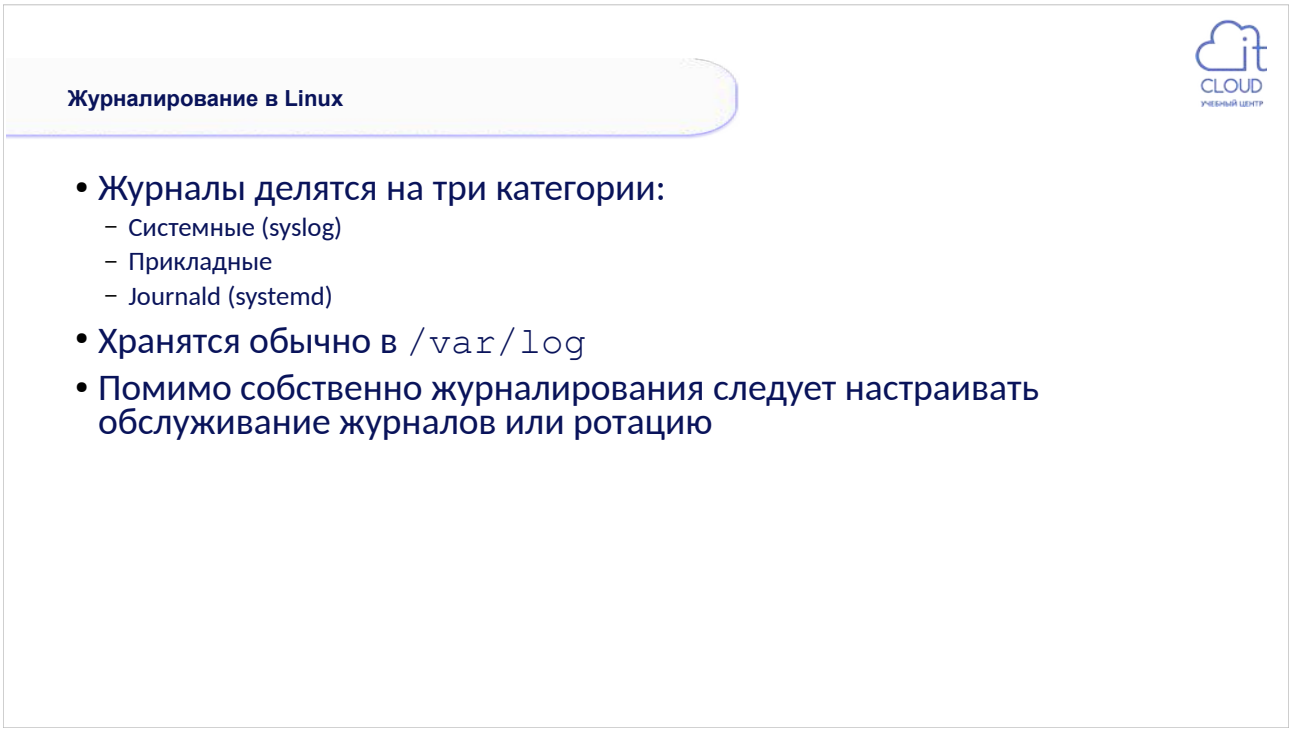
Пока AppArmor не запрещает использование копии программы пинг. Переключим в принудительный режим и проверим, что получится.

```
$ sudo aa-enforce /etc/apparmor.d/home.admuser.ping
Setting /etc/apparmor.d/home.admuser.ping to enforce mode.

$ ./ping -c1 127.0.0.1
./ping: socket: Operation not permitted

$ sudo aa-complain /etc/apparmor.d/home.admuser.ping
Setting /etc/apparmor.d/home.admuser.ping to complain mode.

 sudo aa-genprof /home/admuser/ping

Before you begin, you may wish to check if a
profile already exists for the application you
wish to confine. See the following wiki page for
more information:
https://gitlab.com/apparmor/apparmor/wikis/Profiles

Profiling: /home/admuser/ping

Please start the application to be profiled in
another window and exercise its functionality now.

Once completed, select the "Scan" option below in
order to scan the system logs for AppArmor events.

For each AppArmor event, you will be given the
opportunity to choose whether the access should be
allowed or denied.

[(S)can system log for AppArmor events] / (F)inish
Reading log entries from /var/log/audit/audit.log.
Updating AppArmor profiles in /etc/apparmor.d.
Complain-mode changes:

Profile:    /home/admuser/ping
Capability: net_raw
Severity:   8

 [1 - capability net_raw,]
(A)llow / [(D)eny] / (I)gnore / Audi(t) / Abo(r)t / (F)inish
Adding capability net_raw, to profile.

= Changed Local Profiles =

The following local profiles were changed. Would you like to save them?
```

```
 [1 - /home/admuser/ping]
(S)ave Changes / Save Selec(t)ed Profile / [(V)iew Changes] / View Changes b/w
(C)lean profiles / Abo(r)t
Writing updated profile for /home/admuser/ping.

Profiling: /home/admuser/ping

Please start the application to be profiled in
another window and exercise its functionality now.

Once completed, select the "Scan" option below in
order to scan the system logs for AppArmor events.

For each AppArmor event, you will be given the
opportunity to choose whether the access should be
allowed or denied.

[(S)can system log for AppArmor events] / (F)inish

Reloaded AppArmor profiles in enforce mode.

Please consider contributing your new profile!
See the following wiki page for more information:
https://gitlab.com/apparmor/apparmor/wikis/Profiles

Finished generating profile for /home/admuser/ping.

$ sudo aa-enforce /etc/apparmor.d/home.admuser.ping
Setting /etc/apparmor.d/home.admuser.ping to enforce mode.

$ sudo cat /etc/apparmor.d/home.admuser.ping
# Last Modified: Tue May  3 11:41:33 2022
#include <tunables/global>

/home/admuser/ping {
  #include <abstractions/base>

  capability net_raw,

  /home/admuser/ping mr,

}

$ ./ping -c1 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.051 ms

--- 127.0.0.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.051/0.051/0.051/0.000 ms
```

Глава 7. Мониторинг событий безопасности средствами ОС Linux.

7.1 Журналирование в Linux



В GNU/Linux принято сохранять информацию разной степени детализации о процессе работы программ в специальных текстовых файлах, называемых журналами. Все журнальные файлы принадлежат к одной из трех категорий:

- Системные журналы
- Журналы прикладных программ
- Журналы systemd.

Стандартное место расположения журналов - это каталог /var/log. Но могут быть и исключения.

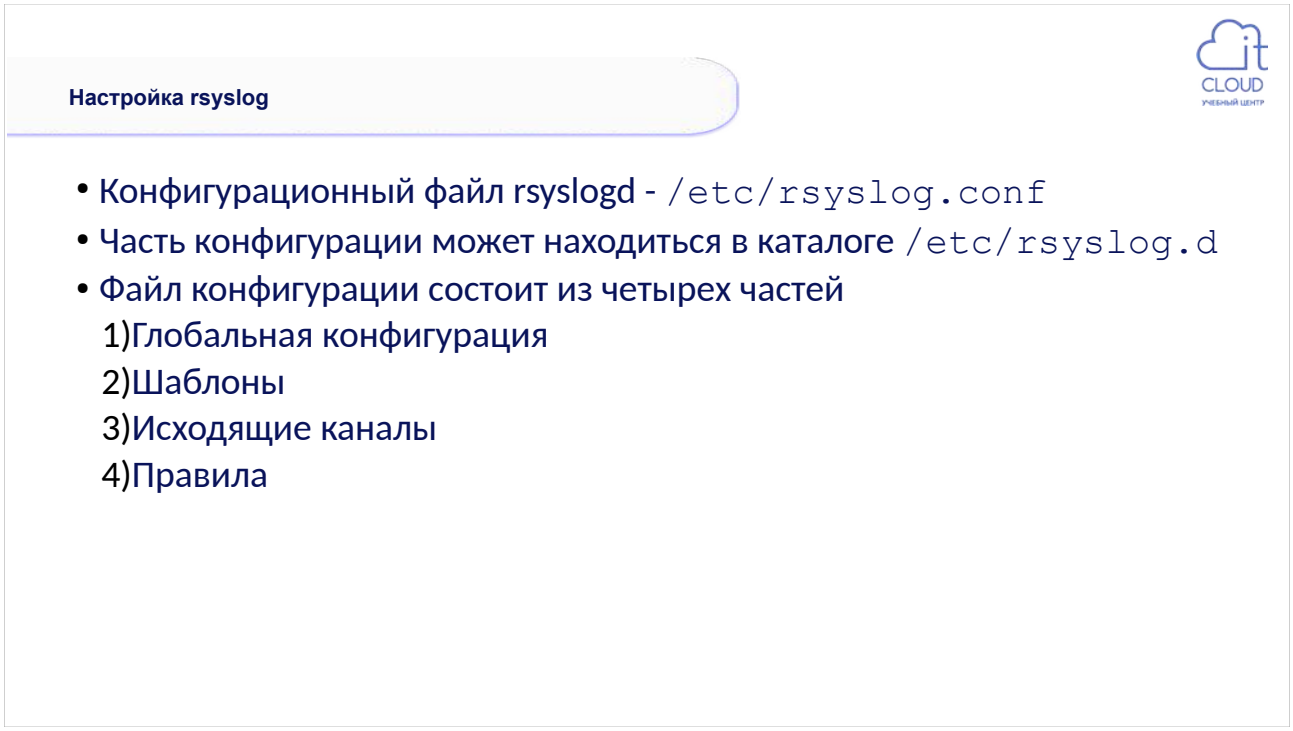
Не все журналы, находящиеся в /var/log , обслуживаются службой syslog. Так, например, бинарный файл базы данных с информацией о последних входах в сеанс wtmp также находится в этом каталоге (имеется аналогичный файл /var/run/utmp с информацией о пользователях, находящихся в сеансе – см. команды who и last).

Служба syslog (или rsyslog, или syslog-ng) предназначена для обеспечения сохранения информации, поступающей от различных системных служб. Некоторые службы ведут собственные журналы, которые не зависят от syslog, например веб сервер Apache. Для таких служб принято создавать отдельные подкаталоги в /var/log.

В GNU/Linux обычно сообщения, поступающие от различных служб, не записываются в один единственный журнал. Наоборот, принято называть файлы журналов так, чтобы по их названию можно было судить об источнике сообщений.

Задачей системы регистрации событий является сохранение информации в нужном журнале, обслуживание журналов это отдельная задача, которая называется ротацией журналов.

7.2 Настройка rsyslog.



Конфигурационным файлом демона `rsyslogd` является `/etc/rsyslog.conf` .

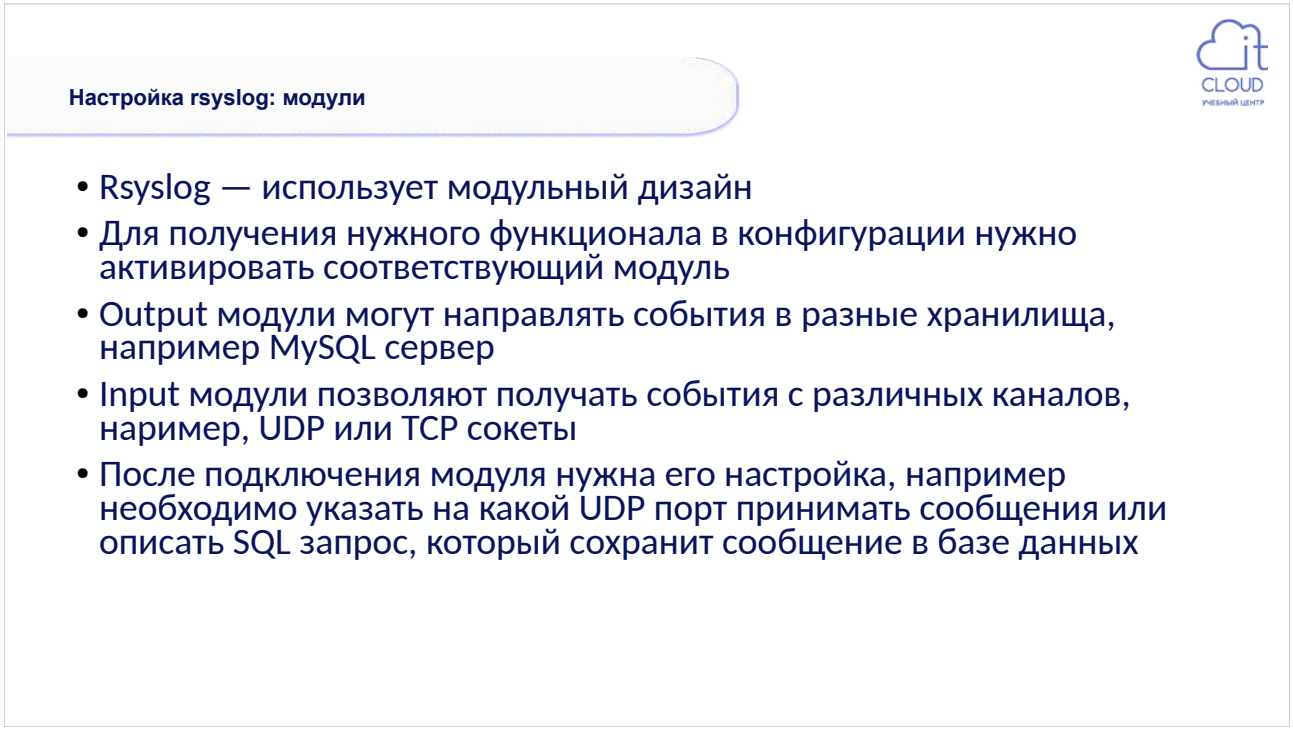
Часть конфигурации может находиться в каталоге `/etc/rsyslog.d`.

Строки этого файла, начинающиеся с решетки # являются комментариями.

Конфигурационный файл состоит из следующих частей:

1. Глобальные параметры: общие настройки службы, например лимиты, загрузка модулей и т. д. Все глобальные директивы начинаются со знака $
2. Шаблоны: используются для определения форматов сообщений и имен журналов.
3. Исходящие каналы: описывают возможные каналы для направления событий.
4. Правила: описывают как сохранять сообщения.

Особенностью настройки rsyslog является то, что в файле конфигурации может сосуществовать два синтаксиса описания конфигурации, условно старый и новый. Это может привести к путанице, поэтому будьте внимательны, когда читаете и применяете различные инструкции по настройке rsyslog.



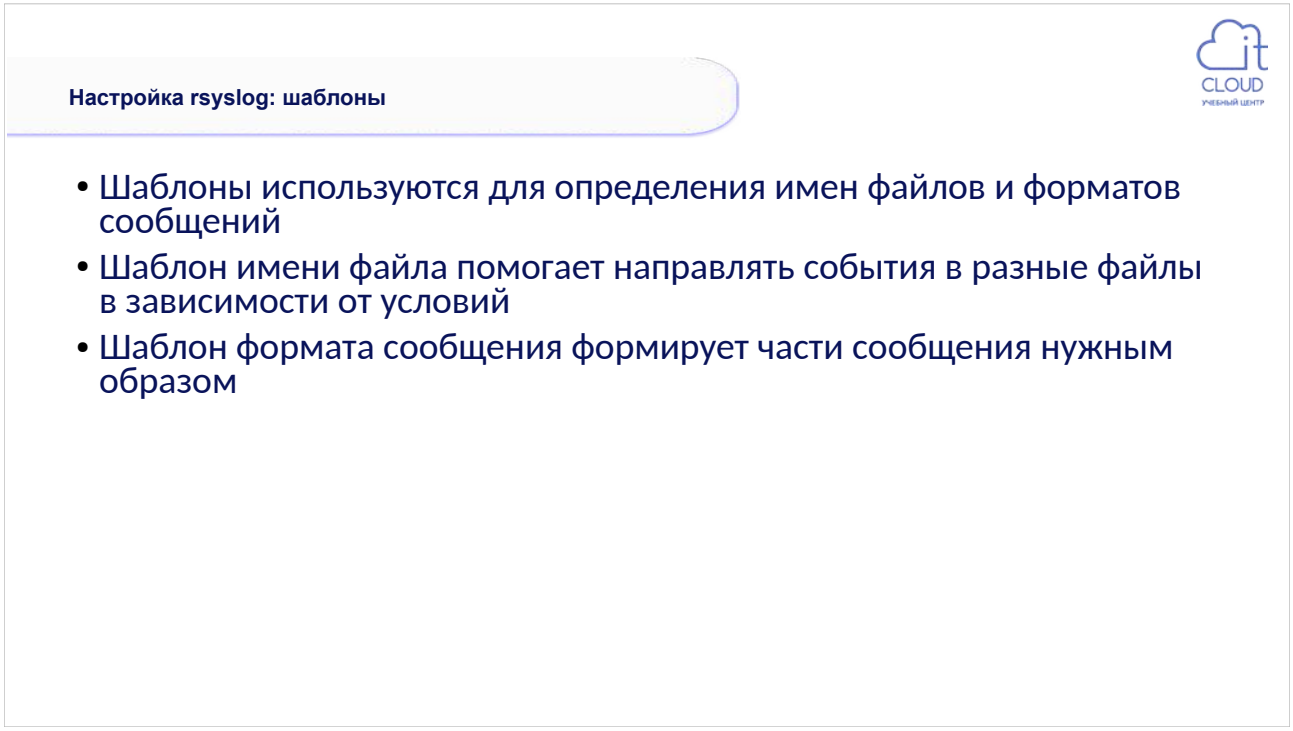
В разработке rsyslogd использовался модульный подход. Это позволило существенно расширить функциональность системы журналирования по сравнению с классической службой syslog.

Модули либо определяют канал и формат сохранения события, тогда такой модуль называется output (в названии используется префикс `om`). Либо модуль определяет канал поступления события — input (название начинается с `im`).

Чтобы получить необходимый функционал в системе регистрации событий вам нужно активировать модуль и настроить параметры его работы.

Пример: Подключение модуля для принятия сообщений через UDP порт и настройка этого модуля на использование 514 порта. Для разбора событий используется набор правил с названием «RemoteUDP514».

```
module(load="imudp")
input(type="imudp" port="514" ruleset="RemoteUDP514")
```



Rsyslog может работать с разными форматами сообщений отличных от стандарта syslog. Чтобы определить формат сообщения вы должны для него создать шаблон.

Шаблоны так же используются для определения формата имени файла, в который будет производиться запись сообщения, т.о. вам не придется создавать множество однотипных строк для фильтрации сообщений.

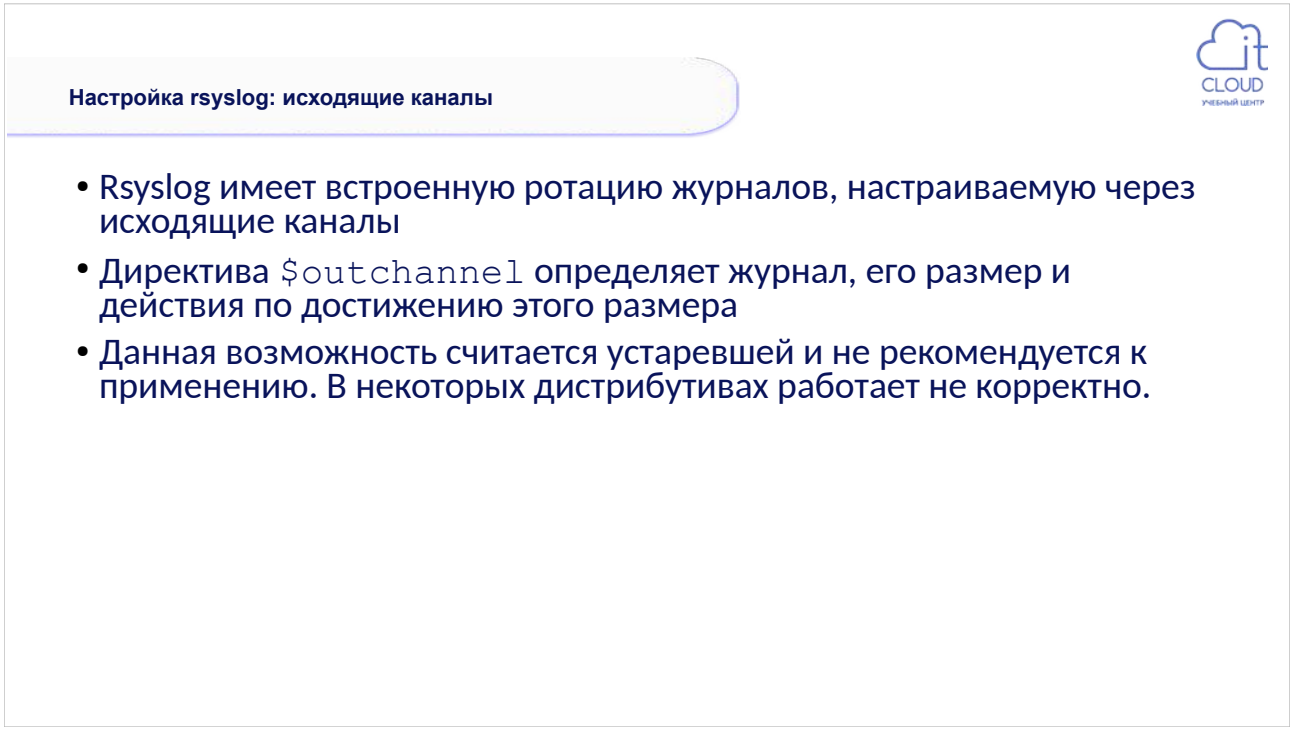
Директива шаблона начинается с ключевого слова `$template`.

Пример:

```
# grep RemHost /etc/rsyslog.conf
$template RemHost,"/var/log/network/%HOSTNAME%.log"
*.* ?RemHost
```

Пример: То же самое, но в новом синтаксисе

```
template(name="RemHost",type="string", string="/var/log/network/%HOSTNAME%.log"
```



Если вы создаете высоко нагруженную систему журналирования, то стандартная система ротации журналов может не удовлетворить ваши потребности. Например, стандартная настройка ротации подразумевает запуск ротации один раз в день, что может быть не достаточно для большого потока событий.

Вместо того, чтобы настраивать ротацию на более частую работу, можно заставить rsyslog запускать скрипты ротации по достижении нужного размера журналов. Сам rsyslog ротацию не производит, но может ее вызвать.

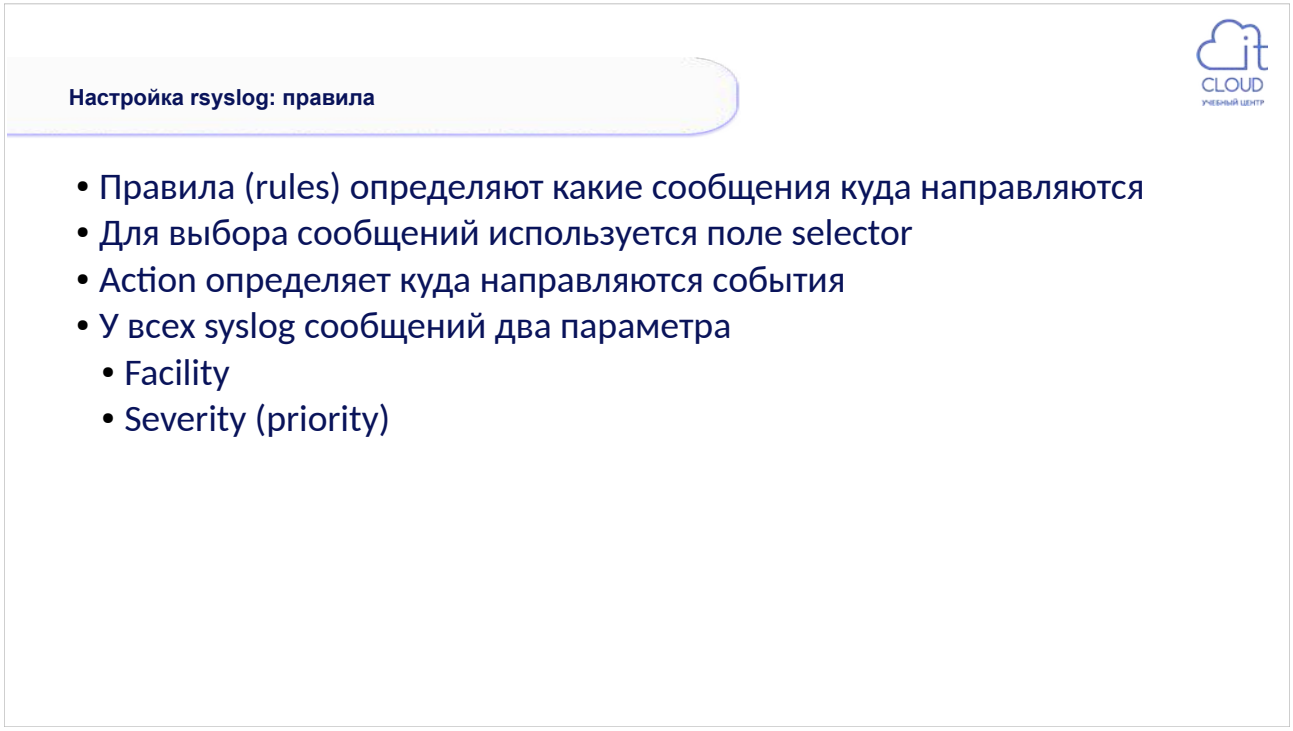
Опция `$outchannel name,file-name,max-size,action-on-max-size` задает:

1. Имя (name) канала
2. Имя файла (file-name) куда будет вестись запись
3. Максимальный размер (max-size) файла
4. Путь к программе или сценарию (action-on-max-size), который будет запущен по достижению этого размера.

После описания канала вы его должны применить.

Пример: Запуск сценария ротации по достижении 50MB

```
# outchannel definition
$outchannel biglog,/var/log/biglog.log,52428800,/usr/local/sbin/log_rotation_script
#  activate the channel and log everything to it
*.* :omfile:$biglog
# end log rotation via outchannel
```



Структура строк, каждая из которых направляет некоторый поток сообщений в заданный файл (или на удаленный компьютер - сервер ведения журналов), представлена двумя полями:

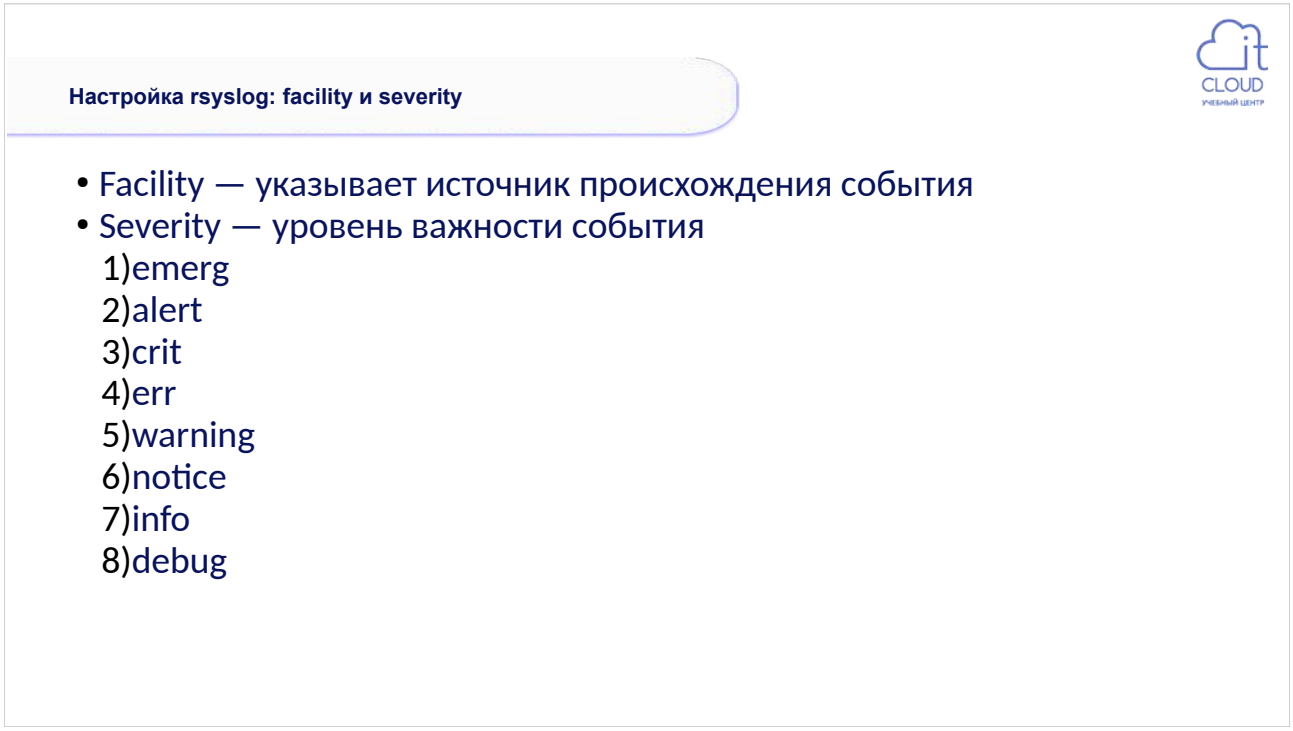
1. Определение сообщения (selector) - поле, в котором указывается от каких классов программ должны собираться сообщения в данный поток. И какие именно сообщения.
2. Поле действия (action), указывающее куда должен быть записан поток сообщений. Чаще всего - это имя файла журнала в `/var/log`.

Пример:

```
root@s10:~# grep log/syslog /etc/rsyslog.conf
*.*;auth,authpriv.none          -/var/log/syslog
```

У каждого сообщения две части. Эти две части уникально определяют все возможные сообщения, обрабатываемые rsyslog.

1. Источник сообщения (facility).
2. Уровень важности (priority или severity) сообщения.



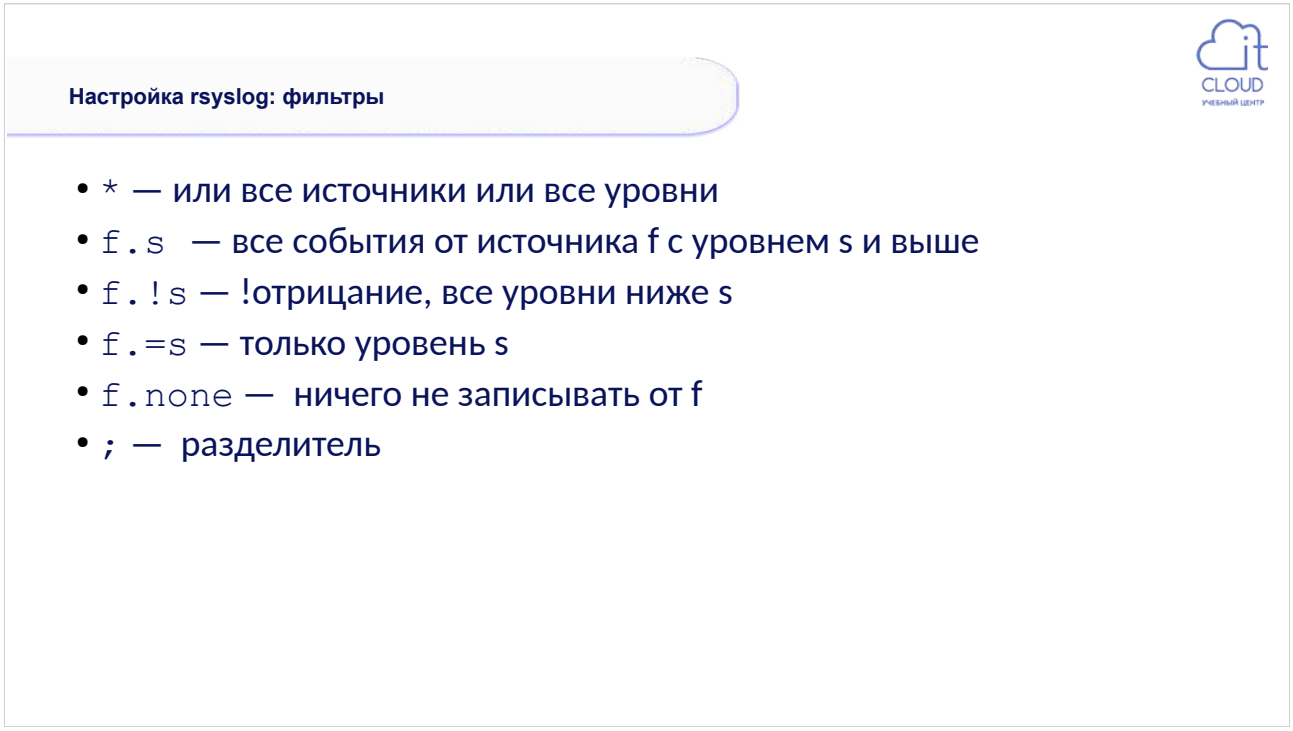
В каждой операционной системе имеется собственный набор источников событий. Источник события один из критериев для направления события в журнал.

Стандартные источники сообщений (facility, см. man 3 syslog):

- auth – сообщения служб авторизации и безопасности (этот источник не рекомендуется использовать, вместо него необходимо использовать authpriv);
- authpriv – сообщения служб авторизации и безопасности;
- cron – сообщения служб at и cron ;
- daemon – сообщения различных демонов;
- kern – сообщения ядра;
- lpr – сообщения службы печати;
- mail – сообщения, поступающие от служб электронной почты;
- mark – источник, зарезервированный для внутреннего использования (в syslog), его не следует использовать в приложениях;
- news – сообщения службы новостей;
- syslog – собственные сообщения syslog;
- user – источник сообщений, зарезервированный для пользовательских программ;
- uucp – сообщения службы uucp;
- local0 ... local7 – источники сообщений, доступные для использования на локальной системе, но не являющиеся стандартными.

Все сообщения разделены также по следующим уровням важности (priority), приведенным в порядке убывания важности:

1. emerg (panic устаревший синоним emerg не рекомендуется использовать) – система не работоспособна;
2. alert – требуется немедленное вмешательство;
3. crit – критическое событие;
4. err (или устаревший error)– ошибка;
5. warning (или устаревший warn)– предупреждение;
6. notice – нормальное, но значимое событие;
7. info – информационное сообщение;
8. debug – отладочная информация.



При настройке rsyslog может использоваться знак звездочка, обозначающий либо все источники, если он указан перед точкой – разделителем, либо все уровни важности, если этот метасимвол установлен после точки.

При указании источника сообщения и уровня важности, разделенных точкой, определяется, что сообщения, поступающие от этого источника и имеющие указанный и вышележащие уровни важности, будут записаны в данный канал.

Пример:все сообщения службы печати с уровнями важности info и выше, будут записаны в файл `/var/log/lpr`

```
lpr.info                        /var/log/lpr
```

При необходимости запретить запись в какой-либо канал сообщений с заданным уровнем важности и выше, можно использовать знак восклицания перед уровнем важности.

Пример: записывать от источника daemon события уровней от info до warning

```
daemon.info;daemon.!err         /var/log/daemons
```

Если же необходимо записывать в журнал сообщения только с определенным уровнем важности и ни с какими другими более, то перед требуемым уровнем важности следует поставить знак равно.

Пример:

```
daemon.=err                 /var/log/daemons.err
```

Для исключения из потока сообщений те из них, которые имеют заданный уровень важности используют восклицательный знак и знак равенства.

Пример:в журнал будут записываться все сообщения от ядра, кроме информационных.

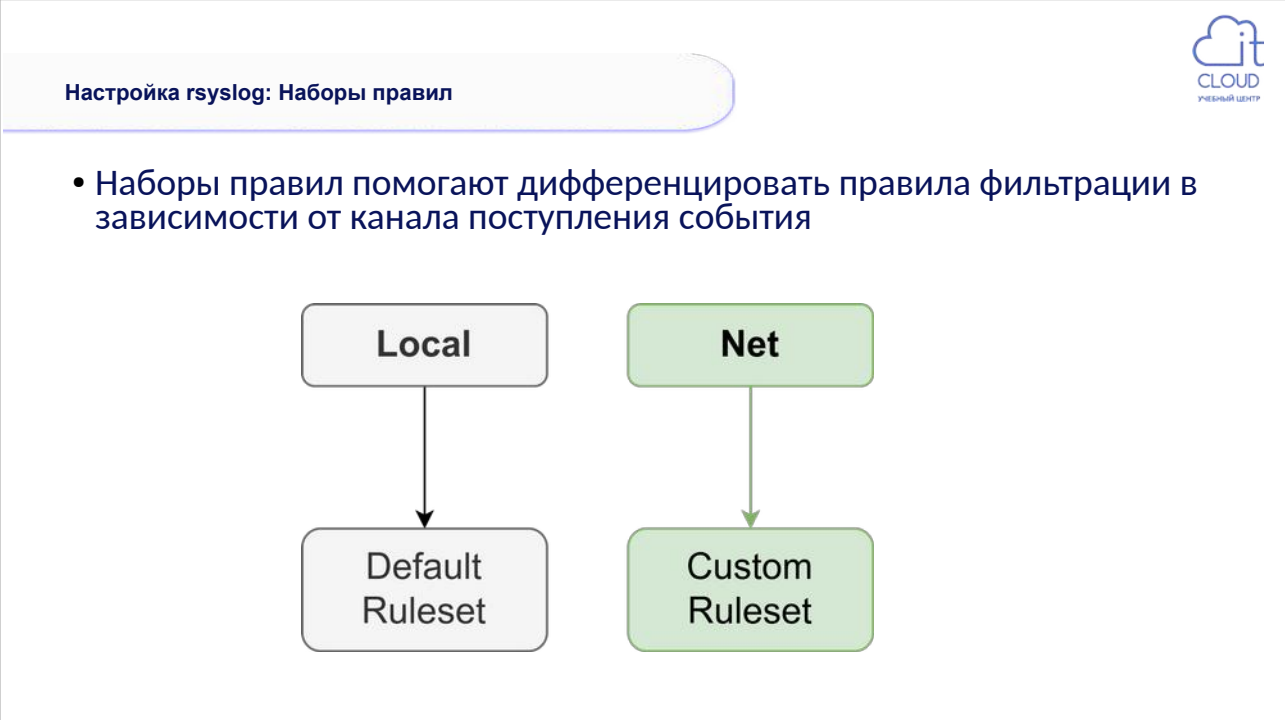
```
kern.*;kern.!=info                  /var/log/kernel
```

Если в канал не должны быть записаны любые сообщения от каких-либо источников, то удобно использовать директиву `none` :

Пример:

```
*.crit;lpr,cron,mail.none     /var/log/critical
```

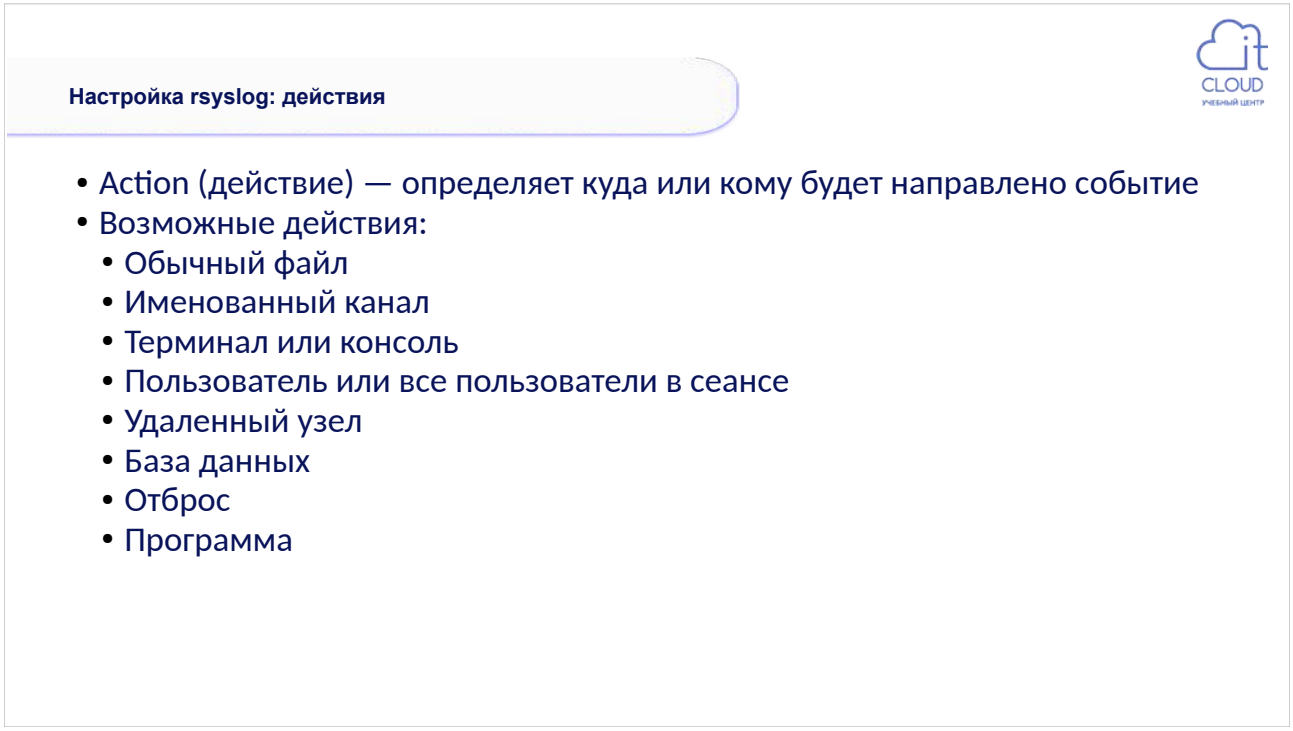
Помимо традиционных фильтров вы можете применять фильтры на основе свойств и на основе выражений.



Наборы правил позволяют создать несколько наборов с правилами фильтрации и применять их для различных каналов поступления событий. Это позволяет, например, создать отдельные правила для событий поступающих из сети.

Пример:

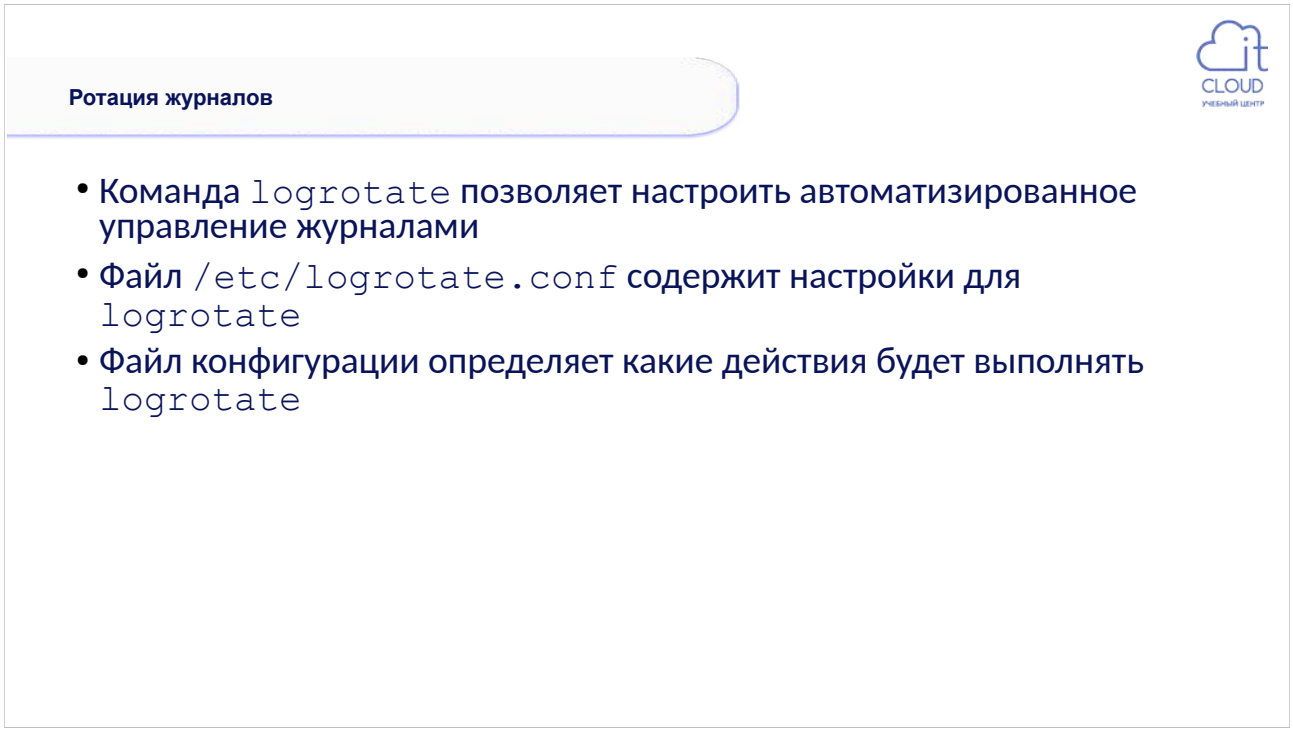
```
input(type="imudp" port="514" ruleset="RemoteUDP514")

template(name="NetworkLog" type="list") {
    constant(value="/var/log/network/")
    constant(value="/")
    property(name="hostname")
    constant(value=".log")
}

ruleset(name="RemoteUDP514") {
        action(type="omfile" dynaFileCacheSize="1024" dynaFile="NetworkLog"
FileOwner="root" FileGroup="root" dirOwner="root" dirGroup="root"
FileCreateMode="0640" DirCreateMode="0755" flushOnTXEnd="off" asyncWriting="on"
flushInterval="10" ioBufferSize="64k")
}
```



После фильтрации над событием выполняется действие. Это могут быть:

- Обычный файл: указывается путь к файлу или имя шаблона через (?) знак.
- Именованный канал: именованный канал может асинхронно передавать события другим процессам.
- Терминал или консоль: если вы укажите путь к специальному файлу устройства типа tty.
- Пользователь или все пользователи в сеансе: либо список пользователей либо все пользователи (*) в сеансе.
- Удаленный узел: направление серверу централизованного журналирования.
- База данных: запись в базу данных, через соответствующий модуль.
- Отброс: игнорировать.
- Программа: запустить программу и передать как аргумент сообщение.

7.3 Управление журналами (хранение, ротация, архивирование).



С течением времени накапливающиеся сообщения в файлах журналов могут переполнить файловую систему.

Для предотвращения этого предназначена программа `logrotate` , обеспечивающая ротацию журналов. Стандартный путь ее вызова – использование ее, как ежедневного задания cron .

Файл `/etc/logrotate.conf` содержит настройки для этой утилиты.

Утилита logrotate способна производить следующие действия с файлами журналов:

- Удалять.
- Переименовывать.
- Сжимать с помощью программ – компрессоров.
- Создавать новые пустые файлы журналов.
- Посылать ротируемые файлы журналов по электронной почте.

Пример: Ротации файла `/var/log/messages` :

```
# ls -w 1 /var/log/messages*

/var/log/messages
/var/log/messages.1.bz2
/var/log/messages.2.bz2
/var/log/messages.3.bz2
/var/log/messages.4.bz2
```

Примечание: *При наступлении момента времени, когда необходимо осуществить первую ротацию файл messages переименовывается в messages.1.bz2 (в данном примере используется компрессия журналов утилитой bzip2).*

При второй ротации файл messages.1.bz2 переименовывается в messages.2.bz2 , а файл messages переименовывается в messages.1.bz2 .

При третьей ротации файл messages.2.bz2 переименовывается в messages.3.bz2 и так далее ...

Утилита logrotate удаляет архивные копии старых журналов по достижении заданного количества копий. В этом примере ротация первой архивной копии журнала (файл messages.1.bz2) осуществляется четыре раза (до messages.4.bz2).

Настройки, находящиеся в начале файла `/etc/logrotate.conf` и не связанные с именами файлов журналов, являются глобальными.

Для каждого конкретного файла журнала можно указывать отдельные настройки.

Обычно применяются следующие настройки:

- `daily`, `weekly` и `monthly` определяют периодичность ротации равной, соответственно, одному дню, неделе или месяцу.
- `rotate` определяет количество ротаций первой архивной копии журнала до ее удаления.
- `create` заставляет создавать пустой файл журнала после его ротации. Причем, настройка create позволяет указывать права доступа и владения создаваемых журнальных файлов.
- `compress` сжатие ротированных файлов. Если не надо сжимать файла архивных копий, то следует использовать настройку nocompress .
- `copy` копировать файлы оставляя при этом оригинальные файлы журналов нетронутыми.
- `notifempty` позволяет не осуществлять ротацию пустых файлов.
- `include` позволяет включать в файл конфигурации дополнительные настройки, указанные в файле – аргументе этой директивы. Если аргументом является каталог, то в основной файл конфигурации включается содержимое всех конфигурационных файлов, находящихся в этом каталоге.
- `mail` позволяет получать копии ротируемых журнальных файлов по электронной почте.
- `prerotate` и `postrotate` позволяют указывать скрипты, которые будут исполнены, соответственно, до и после ротации.
- `size` можно указывать размер файла журнала, по превышении которого должна осуществляться его ротация.

Пример:

```
weekly
```

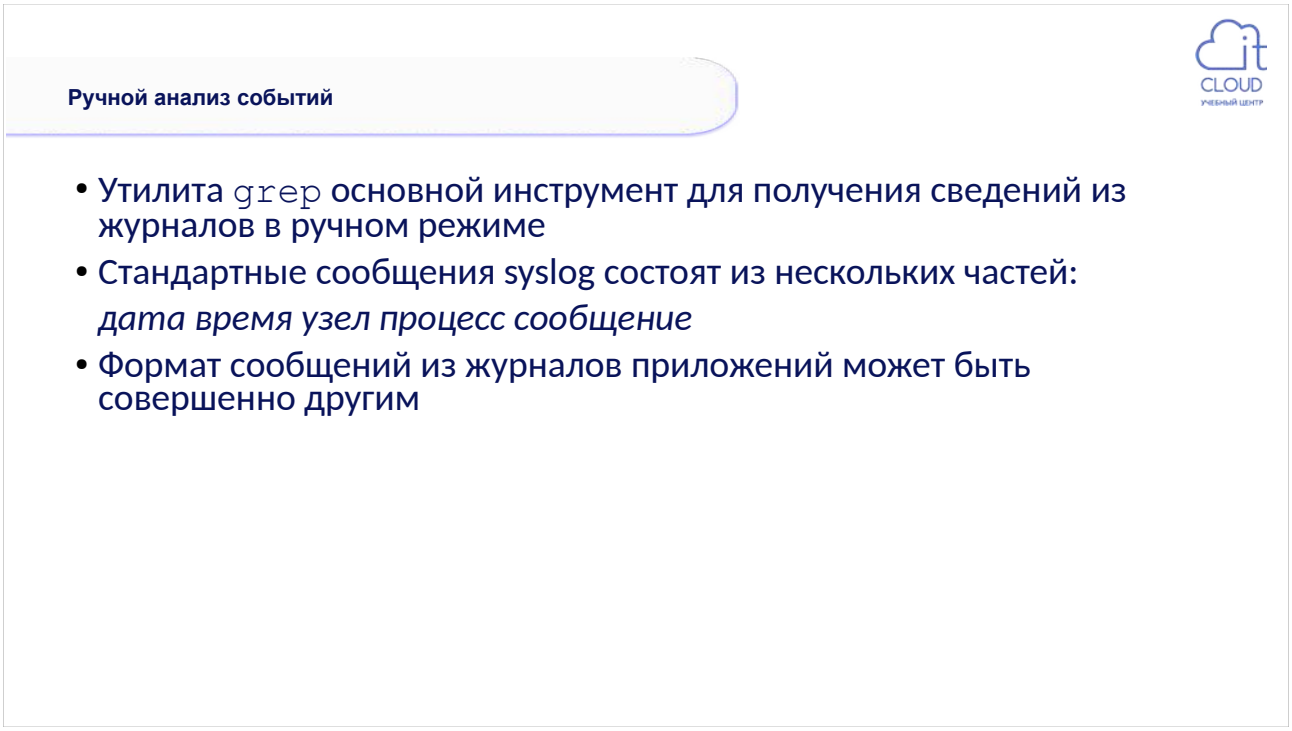
```
rotate 4
create
compress
notifempty
include /etc/logrotate.d
/var/log/wtmp {
    monthly
    create 0664 root utmp
    rotate 4
}
```

Как правило `logrotate` удаляет журналы после нескольких ротаций. Эту опцию можно отключить, но тогда вам нужно внимательно оценить объем дискового пространства, который будет занят журналами.

В некоторых случаях требуется длительное хранение журналов. Рассмотрите возможность по архивации журналов и сохранении их в надежном, защищенном от посторонних месте. Носители информации куда будет производиться архивация должны обеспечивать нужную длительность хранения.

Возможно потребуется оффсайт хранение архивированных журналов.

7.4 Ручной и автоматический анализ событий.



В ручном режиме для анализа журналов обычно используют утилиту grep, которая производит поиск текста на основе регулярных выражений.

Стандартное сообщение syslog состоит из пяти частей:

дата время узел процесс сообщение

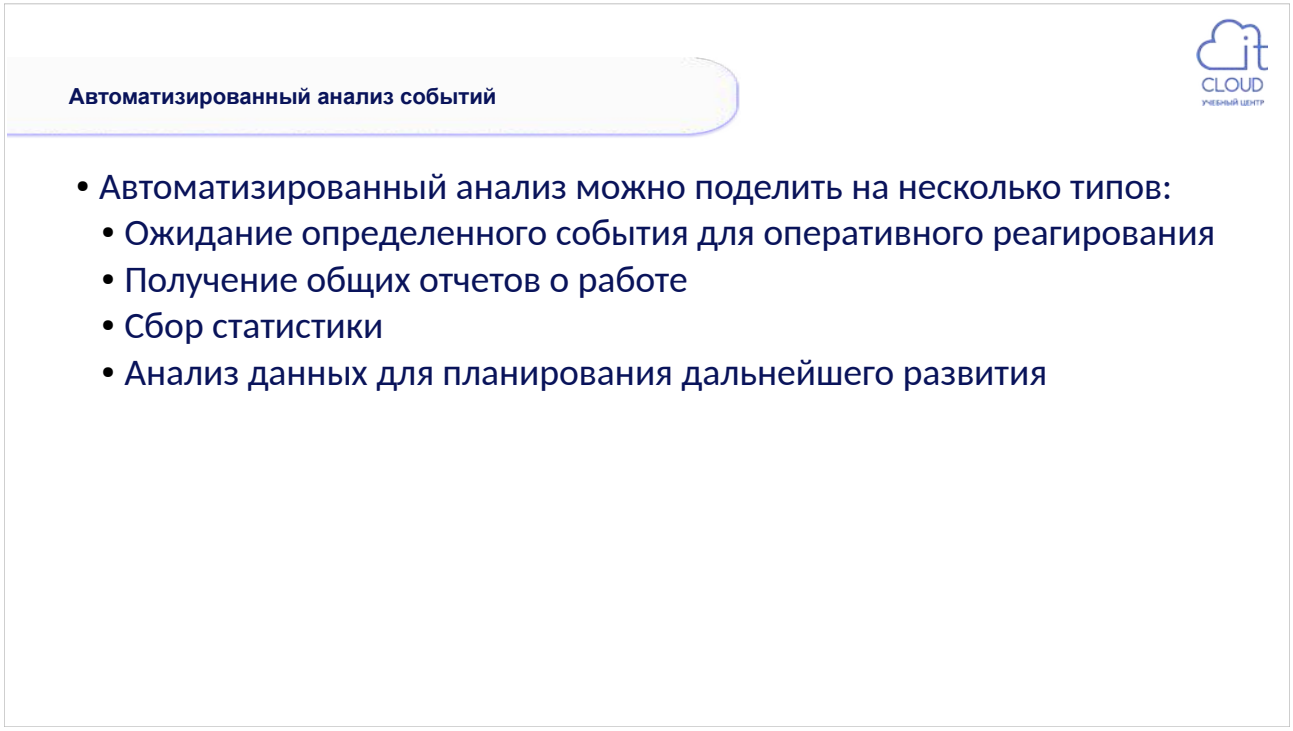
Пример:

```
root@s10:~# grep "Aug  1 18:03" /var/log/messages | tail -1
Aug  1 18:03:40 sl0 systemd: Started The Apache HTTP Server.
```

Журналы приложений могут иметь совершенно другой формат. И даже у одного приложения разные журналы могут иметь разный формат.

Пример:

```
root@s10:~# grep "Aug 01 18:03" /var/log/httpd/error_log | head -1
[Tue Aug 01 18:03:40.527891 2017] [core:notice] [pid 1100] SELinux policy
enabled; httpd running as context system_u:system_r:httpd_t:s0
root@s10:~# grep "01/Aug" /var/log/httpd/access_log | head -1
127.0.0.1 - - [01/Aug/2017:20:21:21 +0500] "GET /sedocs/index.html HTTP/1.1" 200
115877 "-" "Wget/1.14 (linux-gnu)"
```



Не существует единого средства для автоматизированного анализа событий. Все средства можно поделить на несколько категорий.

- Ожидание определенного события. Вы точно знаете что должно произойти, как будет выглядеть сообщение и что вам необходимо сделать при его наступлении, но вы не знаете когда оно произойдет. В таком случае вам нужно использовать программу или скрипт, которые будут получать нужные события от системы журналирования и оперативно на них реагировать. Например: вы получаете сообщение о попытке входа пользователя root по протоколу ssh на сервер, но этого не должно быть, только обычные пользователи могут входить удаленно. Получив такое сообщение важно оперативно заблокировать доступ к порту ssh для узла злоумышленника.

Пример:

```
root@s10:~# grep programname /etc/rsyslog.conf
if $programname == 'sshd' and $msg contains 'Failed password for root from' then
^/usr/local/sbin/mylogparser

root@s10:~# cat /usr/local/sbin/mylogparser
#!/bin/bash
date >> /tmp/myaction
echo -n "My action for message: $1" >> /tmp/myaction
remIP=$(echo $1 | sed -r 's/.*from ([0-9]+\.[0-9]+\.[0-9]+\.[0-9]+).*/\1/')
echo "IS iptables -I INPUT -s $remIP -p tcp --dport 22 -j DROP" >> /tmp/myaction

root@s10:~# tail -3 /tmp/myaction
Thu Aug  3 19:24:33 +05 2017
My action for message: Aug  3 19:24:33 s10 sshd[8444]: Failed password for root
from 10.255.255.254 port 34902 ssh2
IS iptables -I INPUT -s 10.255.255.254 -p tcp --dport 22 -j DROP
```

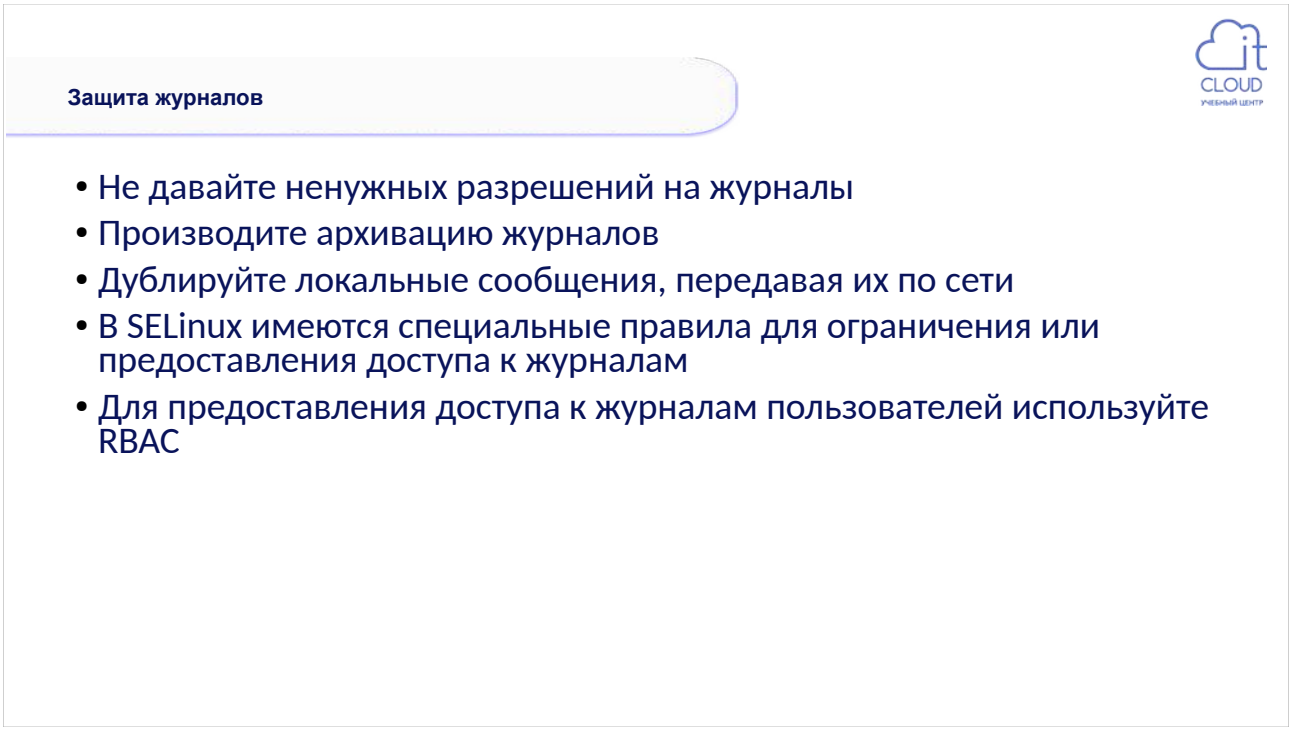
- Получение отчетов о работе. Когда в системе происходит множество событий, то понять что происходило в системе довольно затруднительно. Имеются средства сбора и анализа отчетов о событиях в системе в понятном человеку виде. Таких программ большое количество от наиболее простых до мощнейших средств анализа.

Пример: использование простой программы для составления отчетов по журналам.

```
root@s10:~# logwatch  | head -15

 ################### Logwatch 7.4.0 (03/01/11) ####################
        Processing Initiated: Thu Aug  3 19:35:29 2017
        Date Range Processed: yesterday
                              ( 2017-Aug-02 )
                              Period is day.
        Detail Level of Output: 0
        Type of Output/Format: stdout / text
        Logfiles for Host: sl0
 ##################################################################

 --------------------- Cron Begin ------------------------

 **Unmatched Entries**
 INFO (RANDOM_DELAY will be scaled with factor 24% if used.)
```

- Сбор статистики. В этом случае, как правило, нужна статистика по работе какого-то сервиса. Нас может интересовать разная информация, например общее количество обращений клиентов за период времени, распределение по временным интервалам и и.д. Здесь потребуются специализированные программы или целые комплексы для получения таких сведений. Примером может служить простейший анализатор журналов веб-сервера webalizer.
- Анализ данных. Предполагается вы ведете журналирование сильно нагруженного сервиса, который генерирует огромное количество событий. И вам необходимо иметь информацию как это работает, что делают клиенты, что вы им еще можете предложить, прогноз будущего роста, предсказание проблем и многое другое. Этими вопросами занимается направление больших данных и искусственного интеллекта.

7.5 Защита журналов.



В системных журналах и журналах приложений может содержаться важная информация, которая может быть использована злоумышленниками. Поэтому в первую очередь оцените установленные разрешения на доступ к файлам журналов. Разрешения должны максимально ограничивать доступ без потери работоспособности.

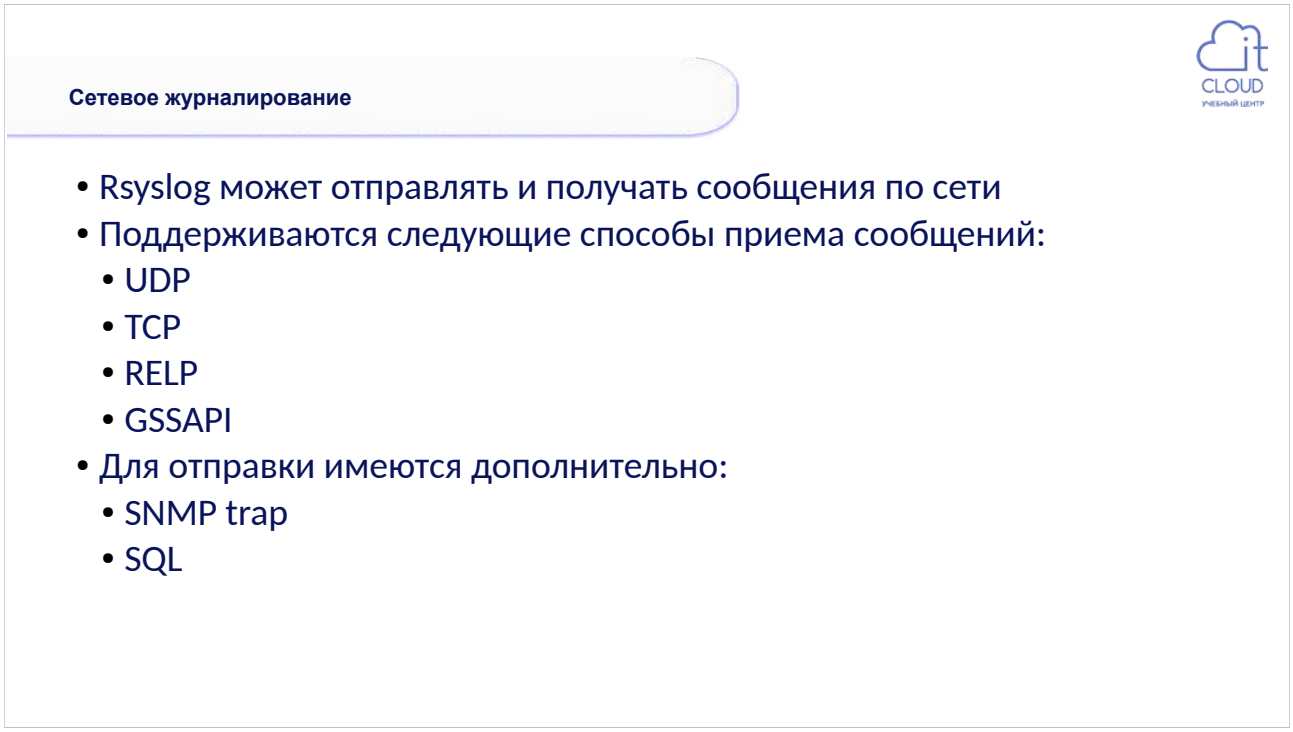
Оцените как вы архивируете журналы. Сохранение архивов с журналами один из важнейших принципов обеспечения безопасности.

Если в вашу систему проник злоумышленник, то он может попытаться подчистить сведения в журналах о своей деятельности. В этом случае дополнительной мерой будет дублирование событий по сети на другой узел.

SELinux имеет специальные политики для предоставления доступа к журналам. Не стоит отключать эту возможность.

Если вы хотите предоставить доступ к журналам пользователям, то рассмотрите использование системы RBAC для ограничения возможных последствий этого.

7.6 Построение системы централизованного управления событиями безопасности на основе rsyslog.



Rsyslog ориентирован на построение или использование централизованной системы регистрации сообщений. Он может как отправлять сообщения по сети так и получать сообщения из сети.

Для настройки сетевых функций вам необходимо активировать соответствующие модули и настроить их.

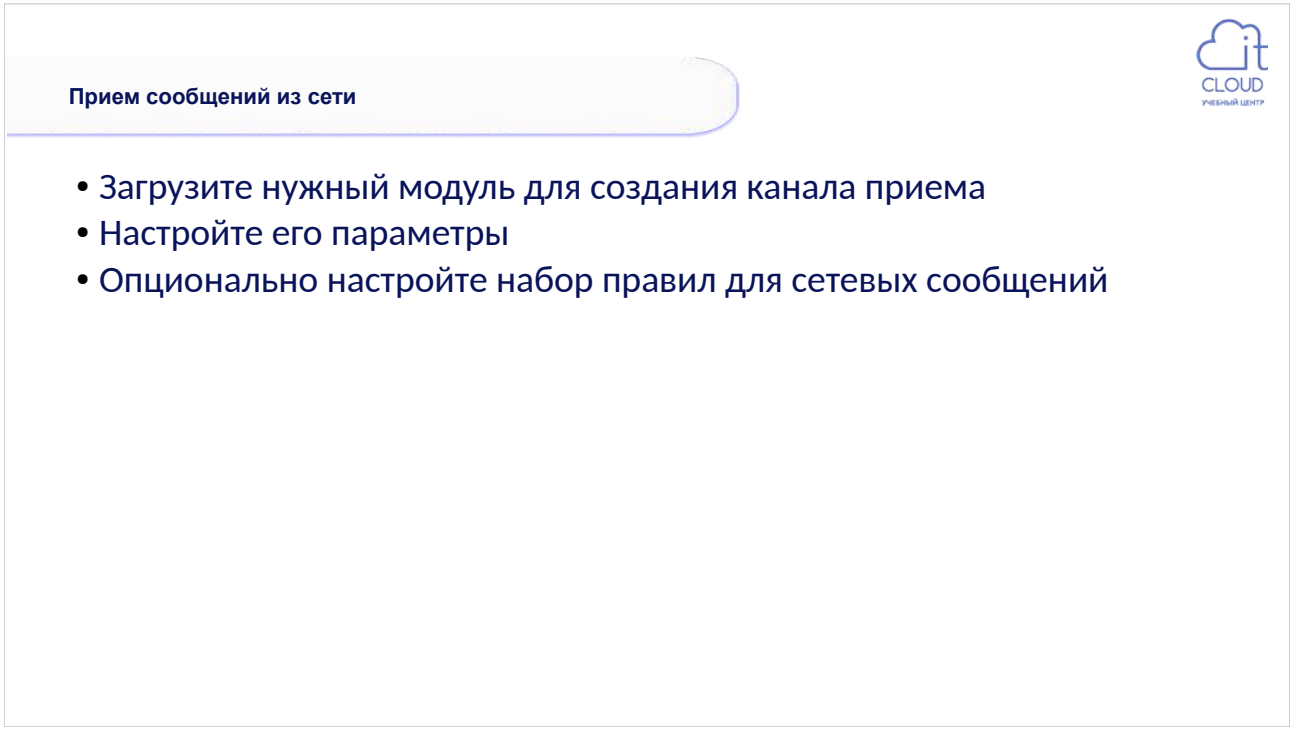
Стандартным средством передачи сообщения по сети является протокол UDP порт 514. UDP не обеспечивает надежной доставки данных.

Протокол TCP обеспечивает больше гарантий по доставке сообщений, но и он не может полностью ее гарантировать.

Протокол RELP (Reliable Event Logging Protocol) обеспечивает дополнительную гарантию доставки на уровне приложений.

GSSAPI позволяет защитить процесс передачи сообщений посредством Kerberos 5.

В дополнение к вышеперечисленным способам, для отправки сообщений, имеется модули для передачи в виде SNMP Trap и модули для сохранения сообщений на различные SQL серверы.

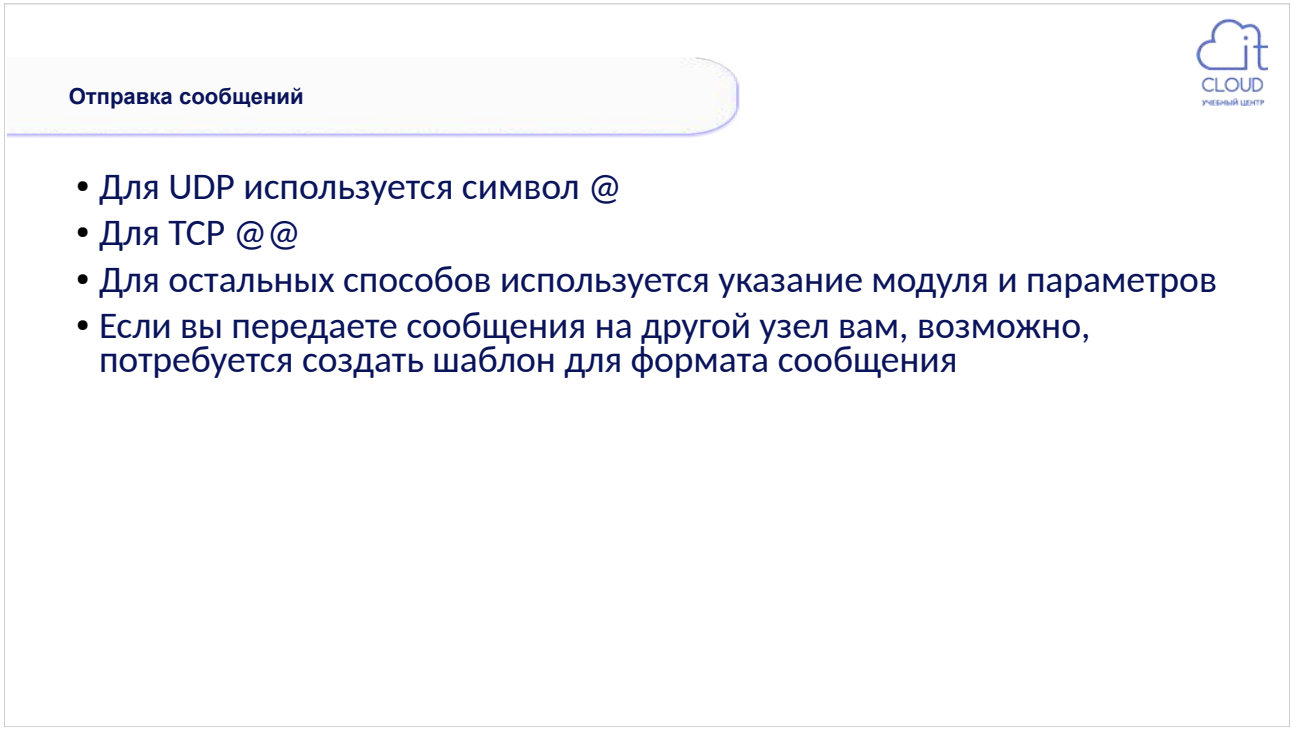


Настройка приема сообщений по сети состоит из:

- Подключения нужного модуля.
- Определения его свойств.
- Дополнительно вы можете настроить отдельные правила фильтрации сообщений или отдельные наборы правил. Последние предпочтительней с точки зрения производительности.

Пример: Используется прием сообщений по сети по протоколу UDP порт 514. Все сетевые сообщения попадают в каталог `/var/log/network` и для каждого узла отправившего сообщения создается отдельный файл в нем.

```
root@s10:~# cat /etc/rsyslog.d/udp514.conf
module(load="imudp")
input(type="imudp" port="514" ruleset="RemoteUDP514")

template(name="NetworkLog" type="list") {
    constant(value="/var/log/network/")
    constant(value="/")
    property(name="hostname")
    constant(value=".log")
}

ruleset(name="RemoteUDP514") {
        action(type="omfile" dynaFileCacheSize="1024" dynaFile="NetworkLog"
FileOwner="root" FileGroup="root" dirOwner="root" dirGroup="root"
FileCreateMode="0640" DirCreateMode="0755" flushOnTXEnd="off" asyncWriting="on"
flushInterval="10" ioBufferSize="64k")
}
```



Отправка сообщений по средством UDP достигается установкой символа @ перед именем узла.

Пример: передача сообщений на нестандартный порт UDP

```
*.* @192.168.0.1:1514
```

Для TCP протокола используется два символа @@

Пример: то же но по протоколу TCP

```
*.* @@192.168.0.1:1514
```

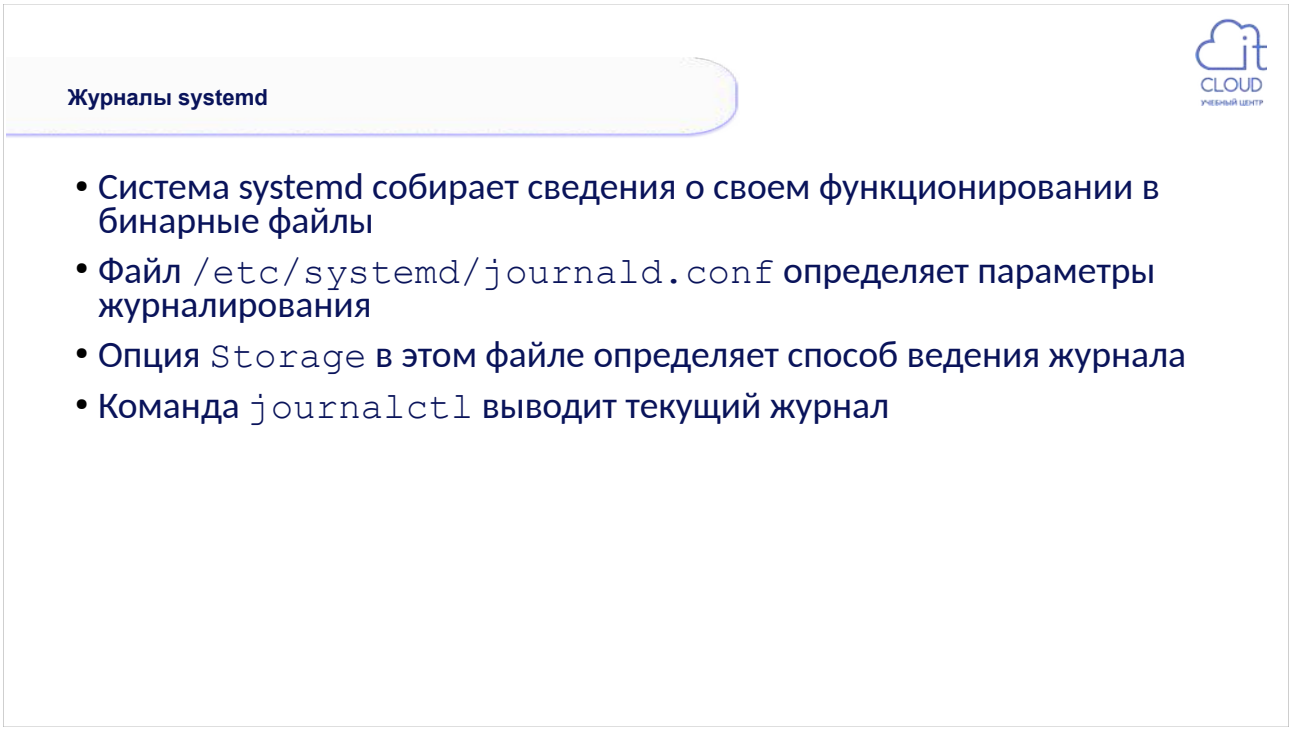
Если использовать другой протокол, например RELP, то необходимо указывать название модуля.

Пример: то же но по протоколу RELP

```
*.* :omrelp:192.168.0.1:1514
```

Возможно сообщения будут передаваться серверу, который не использует формат syslog для сообщений, тогда необходимо создать шаблон для таких сообщений.

7.7 Журналы systemd, демон journald



Одной из проблем классической системы инициализации Linux систем, была потеря сведений, о том как стартовали различные службы. Другая проблема это отсутствие обратной связи с инициализируемыми службами, в том числе информации о функционировании. Для решения этих проблем система `systemd` собирает сведения о своем функционировании в бинарные файлы. Журналирование контролирует демон `systemd-journald`.

Файл `/etc/systemd/journald.conf` определяет параметры журналирования.

Важнейшая опция `Storage` в этом файле определяет способ ведения журнала постоянный (`persistent`), непостоянный (`volatile`) или никакой (`none`).

Команда `journalctl --header` показывает в том числе название файла в который в данный момент записывается информация.

Каждый раз при старте системы создается новый журнал.

Команда journalctl выводит текущий журнал с начала с самого раннего события.

- Опция `-e` показывает журнал с конца.
- Опция `-S` с какого времени выводить сообщения.
- Опция `-U` до какого времени выводить сообщения.
- Опция `-u` сообщения какого юнита выводить.

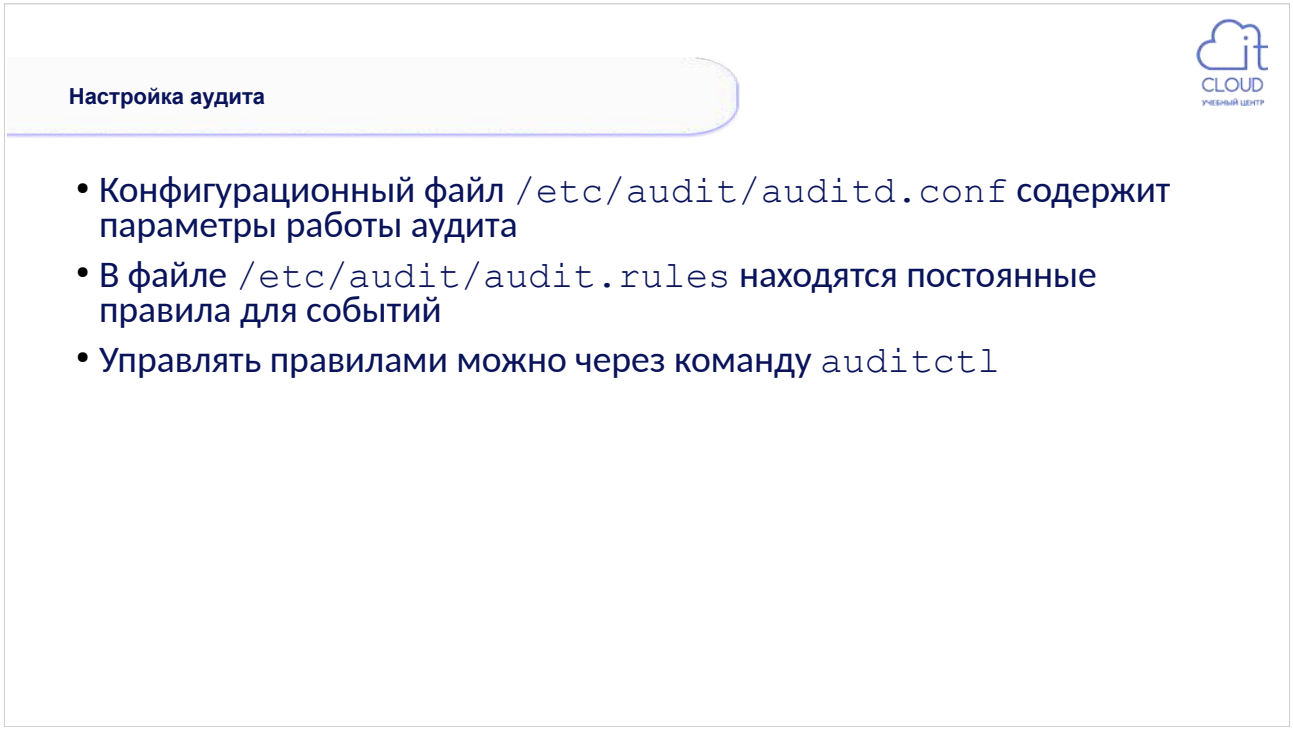
7.8 Система аудита на основе демона auditd

Одним из инструментов, позволяющих повысить уровень безопасности в Linux, является подсистема аудита. С её помощью можно получить подробную информацию обо всех системных событиях.

Подсистема аудита была добавлена в ядро Linux начиная с версии 2.6. Она предназначена для отслеживания критичных с точки зрения безопасности системных событий.

Аудит перехватывает системные вызовы и записывает о этих вызовах информацию:

Получив вызов от приложения в пространстве пользователя, подсистема аудита пропускает его через один из следующих фильтров: user, task или exit (более подробно о них речь пойдёт ниже). После этого вызов пропускается через фильтр exclude, который исходя из правил аудита передаёт его демону auditd для дальнейшей обработки.



В конфигурационном файле `/etc/audit/auditd.conf` содержатся параметры работы аудита. Важнейшие настройки в нем

- `log_file` — файл, в котором будут храниться логи подсистемы аудита;
- `log_format` — формат, в котором будет сохранены логи;
- `freq` — максимальное число записей протокола, которые могут храниться в буфере;
- `flush` — режим синхронизации буфера с диском (`none` — ничего не делать, `incremental` — переносить данные из буфера на диск с частотой, указанной в значении параметра `freq`; `data` — синхронизировать немедленно, `sync` — синхронизировать как данные, так и метаданные файла при записи на диск);
- `max_log_file` — максимальный размер файла лога в мегабайтах;
- `max_log_file_action` — действие при превышении максимального размера файла лога;
- `space_left` — минимум свободного пространства в мегабайтах, по достижении которого должно быть осуществлено действие, указанное в следующем параметре;
- `space_left_admin` — указывает, что делать, когда на диске недостаточно свободного места (`ignore` — ничего не делать; `syslog` — отправлять в syslog, `email` — отправлять уведомление по почте; `suspend` — прекратить

запись логов на диск; `single` — перейти в однопользовательский режим; `halt` — выключить машину);

- `disk_full_action` — действие, которое нужно осуществить при переполнении диска (этот параметр может принимать те же значения, что и `space_left_admin`).

Для добавления и настройки правил используется команда `auditctl`. Вот список её опций:

Чтобы создать новое правило, нужно выполнить команду вида:

```
auditctl -a <список>,<действие> -S <имя системного вызова> -F <фильтры> -k <ключ>
```

Сначала после опции `-a` указывается список, в который нужно добавить правило. Всего существует пять таких списков:

1. `task` — события, связанные с созданием новых процессов;
2. `entry` — события, которые имеют место при входе в системный вызов;
3. `exit` — события, которые имеют место при выходе из системного вызова;
4. `user` — события, использующие параметры пользовательского пространства;
5. `exclude` — используется для исключения событий.

Затем указывается, что нужно делать после наступления события. Здесь возможны два варианта:

1. `always` события будут записываться в журнал
2. `never` не будут записываться

После опции `-S` идёт имя системного вызова, при котором событие нужно перехватить (open, close и т.п.).

Опция `-k` задает ключ, который будет очень полезен в разборе событий аудита.

Пример:

```
root@s10:~# auditctl -a always,exit -F arch=b64 -S adjtimex -S settimeofday -k time_change
```

После опции `-F` указываются дополнительные параметры фильтрации. Например, если нам требуется вести аудит обращений к файлам из каталога `/etc`, правило будет выглядеть так:

Пример:

```
root@s10:~# auditctl -a always,exit -F arch=b64  -F path=/etc -k etc_access
```

При настройке слежения за отдельными файлами можно опустить опцию `-S`.

Для отслеживания доступа к файлам имеется отдельные опции `-w` и `-p`.

Пример:

```
root@s10:~# auditctl -w /etc/passwd -p wa -k passwd_access
```

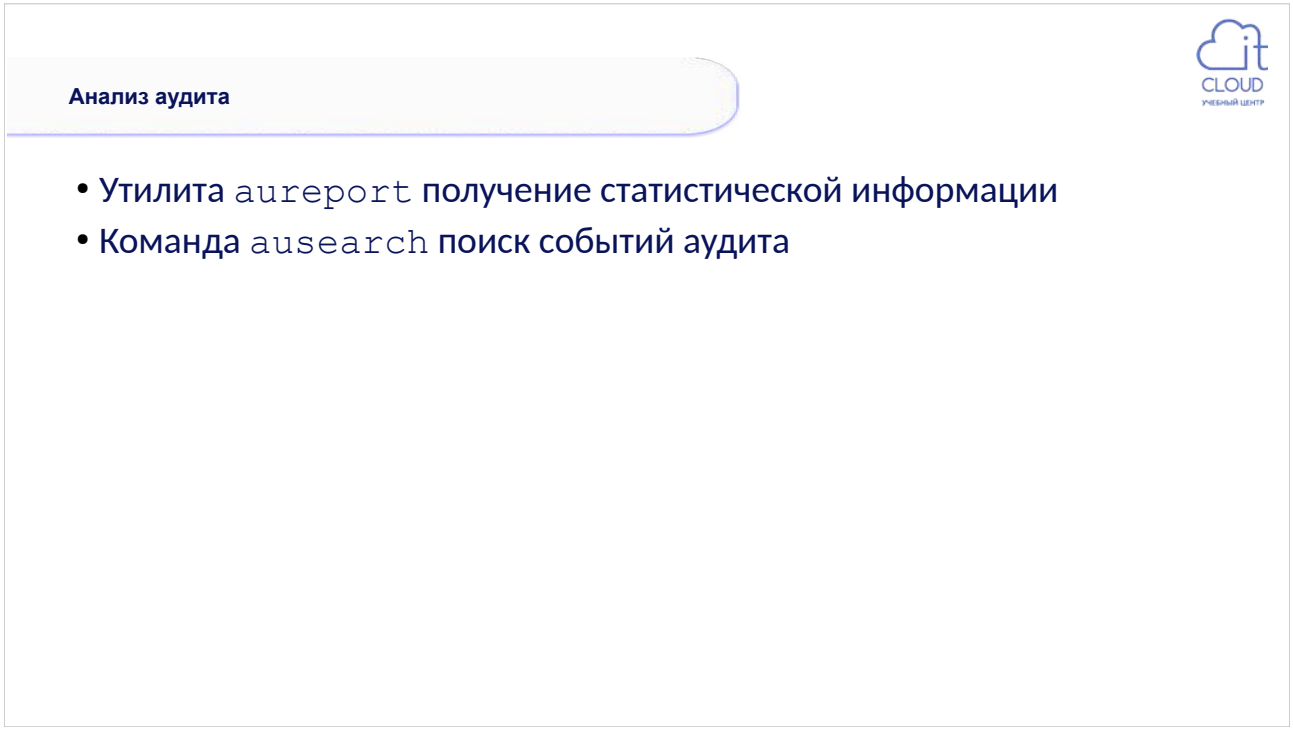
Правила можно не только задавать через командную строку, но и прописывать в файле `/etc/audit/audit.rules`.

Начинается этот файл с так называемых метаправил, в которых задаются общие настройки журналирования:

Пример:

```
root@s10:~# cat /etc/audit/audit.rules
## This file is automatically generated from /etc/audit/rules.d
-D
-b 320
```

Изменения конфигурации вступят в силу после перезапуска демона `auditd`:



Журналы аудита сохраняются в бинарном виде. Их можно сделать человеко понятными с помощью утилиты `aureport`.

Если ввести команду `aureport` без аргументов, мы увидим общую системную статистику (количество пользователей системы, общее количество системных вызовов, число открытых терминалов и т.п.):

Опции команды aureport позволяют получить отчеты по различным типам событий.

Команда `ausearch` производит поиск событий по указанным критериям.

Пример:

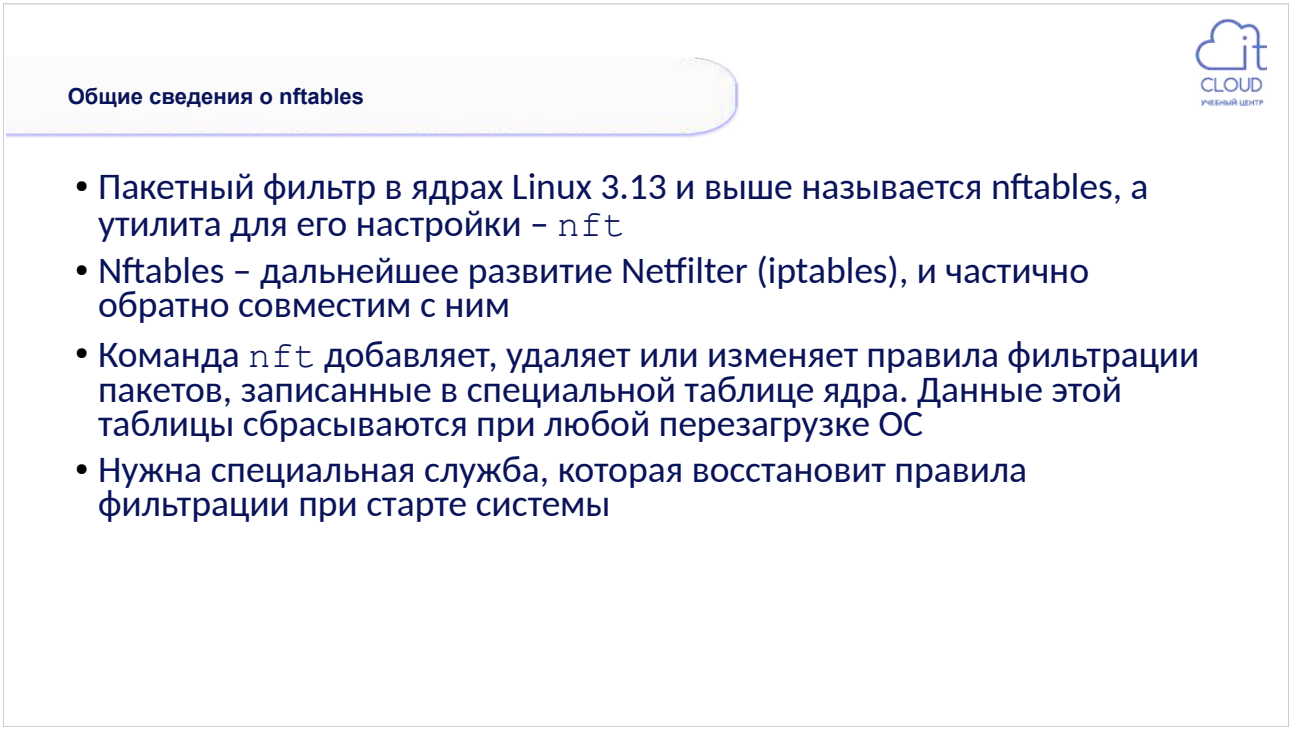
```
root@s10:~# aureport  | head

Summary Report
======================
Range of time in logs: 07/24/2017 20:44:48.597 - 08/04/2017 01:02:04.543
Selected time for report: 07/24/2017 20:44:48 - 08/04/2017 01:02:04.543
Number of changes in configuration: 52
Number of changes to accounts, groups, or roles: 97
Number of logins: 76
Number of failed logins: 40
Number of authentications: 103


root@s10:~# aureport -au | head

Authentication Report
============================================
# date time acct host term exe success event
============================================
1. 07/24/2017 20:45:02 root ? tty1 /usr/bin/login yes 38
2. 07/24/2017 20:47:59 root ? tty1 /usr/bin/login yes 37
```

```
3. 07/24/2017 21:40:06 root ? tty1 /usr/bin/login yes 37
4. 07/24/2017 21:41:54 root ? ttyS0 /usr/bin/login yes 37
5. 07/24/2017 21:42:41 root ? tty1 /usr/bin/login yes 38

root@s10:~# useradd test
root@s10:~# ausearch -k passwd_access -i
----
type=CONFIG_CHANGE msg=audit(08/04/2017 00:43:26.211:336) : auid=root ses=2
op="add_rule" key=passwd_access list=exit res=yes
----
type=CONFIG_CHANGE msg=audit(08/04/2017 00:44:09.270:337) : auid=root ses=2
op="add_rule" key=passwd_access list=exit res=no
----
type=CONFIG_CHANGE msg=audit(08/04/2017 00:54:35.709:347) : auid=root ses=2
op="add_rule" key=passwd_access list=exit res=no
----
type=PATH msg=audit(08/04/2017 01:13:15.071:406) : item=0 name=/etc/passwd
inode=34691654 dev=fd:00 mode=file,644 ouid=root ogid=root rdev=00:00
objtype=NORMAL
type=CWD msg=audit(08/04/2017 01:13:15.071:406) :  cwd=/root
type=SYSCALL msg=audit(08/04/2017 01:13:15.071:406) : arch=x86_64 syscall=open
success=yes exit=5 a0=0x7fbe70f63ce0 a1=O_RDWR|O_NOCTTY|O_NONBLOCK|O_NOFOLLOW
a2=0x0 a3=0x0 items=1 ppid=1940 pid=9019 auid=root uid=root gid=root euid=root
suid=root fsuid=root egid=root sgid=root fsgid=root tty=ttyS0 ses=2 comm=useradd
exe=/usr/sbin/useradd key=passwd_access
----
type=CONFIG_CHANGE msg=audit(08/04/2017 01:13:15.603:420) : auid=root ses=2
op="updated_rules" path=/etc/passwd key=passwd_access list=exit res=yes
```

Глава 8. Защита сетевых взаимодействий.

8.1 Фильтрация трафика.

8.1.1 Базовые концепции

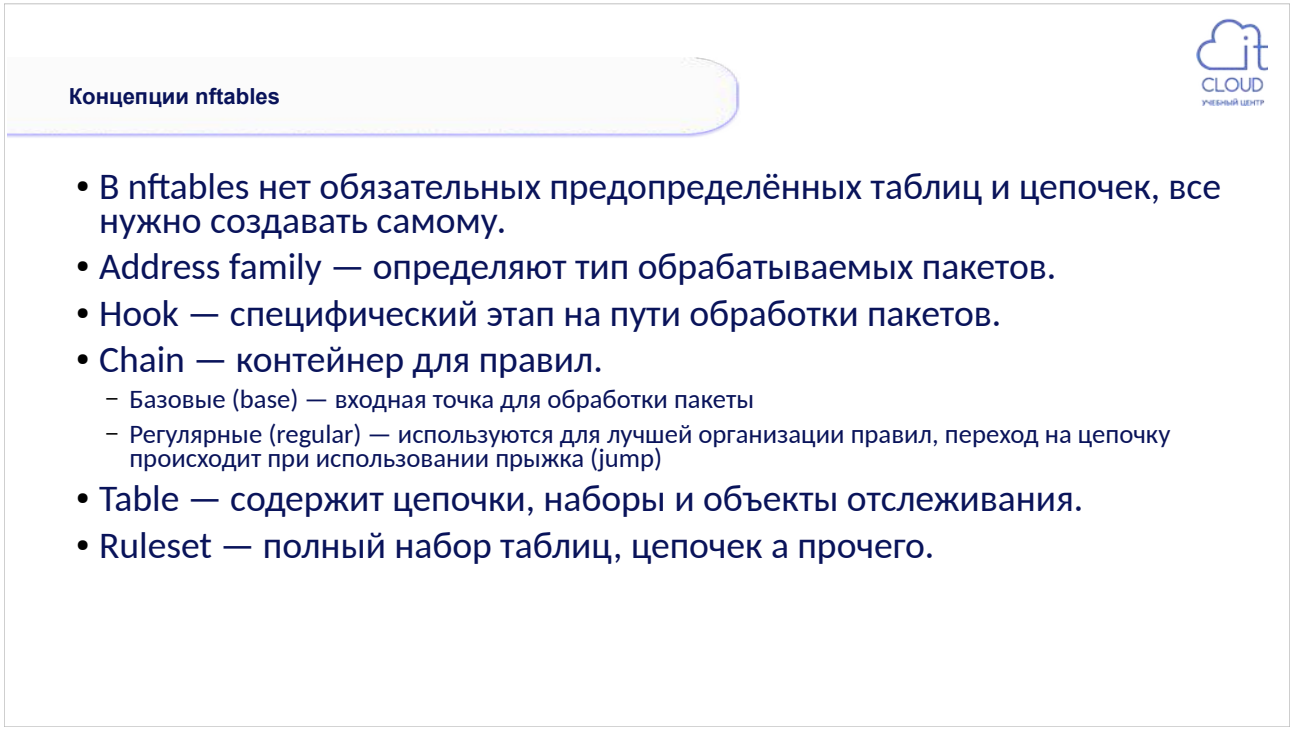


Пакетный фильтр в ядрах Linux называется nftables, а утилита для его настройки – `nft` (ссылка на документацию: https://wiki.nftables.org/wiki-nftables/index.php/Main_Page).

Nftables – пакетный фильтр для Linux, который относится к классу фильтров с проверкой состояния (stateful filter). Фильтры данного класса запоминают информации о текущем состоянии сессии и производят анализ всех входящих пакетов для проверки их корректности. Nftables расширяет возможности предыдущего проекта по фильтрации пакетов Netfilter. При этом обладает ограниченной обратной совместимостью, в том смысле, что все созданные правила командой `iptables` будут видны в `nft`, но не наоборот. Проект Netfilter считается устаревшим с 2021 года.

Команда `nft` добавляет, удаляет или изменяет правила фильтрации пакетов, записанные в специальной таблице ядра. Данные этой таблицы сбрасываются при любой перезагрузке ОС.

Для того, чтобы правила фильтрации применялись во время старта системы необходима какая-нибудь служба, например: firewalld, ufw, netfilter-persistent и др.



В nftables имеется несколько концептуальных моментов.

Во-первых изначально нет никаких правил, цепочек, таблиц и т. д., в отличие от iptables.

Во-вторых названия таблицам и цепочкам вы можете давать какие угодно.

Два параметра являются заданными условиями окружающей среды, а именно тип и состояние пакета:

- Address family — тип обрабатываемого пакета, например ip или arp.
- Hook — специфический этап на пути обработки пакета. Когда пакеты проходят сквозь машину, то возникают несколько стадий обработки. Эти стадии зависят от типа пакета (address family) и направления его движения: входящий и предназначенный для этой машины, транзитный, исходящий от этой машины.

Все известные системы фильтрации пакетов так или иначе состоят из индивидуальных правил. Каждое отдельное правило описывает описывает некоторые признаки пакета и что с этим пакетом делать. Другими словами используется парадигма match-statement.

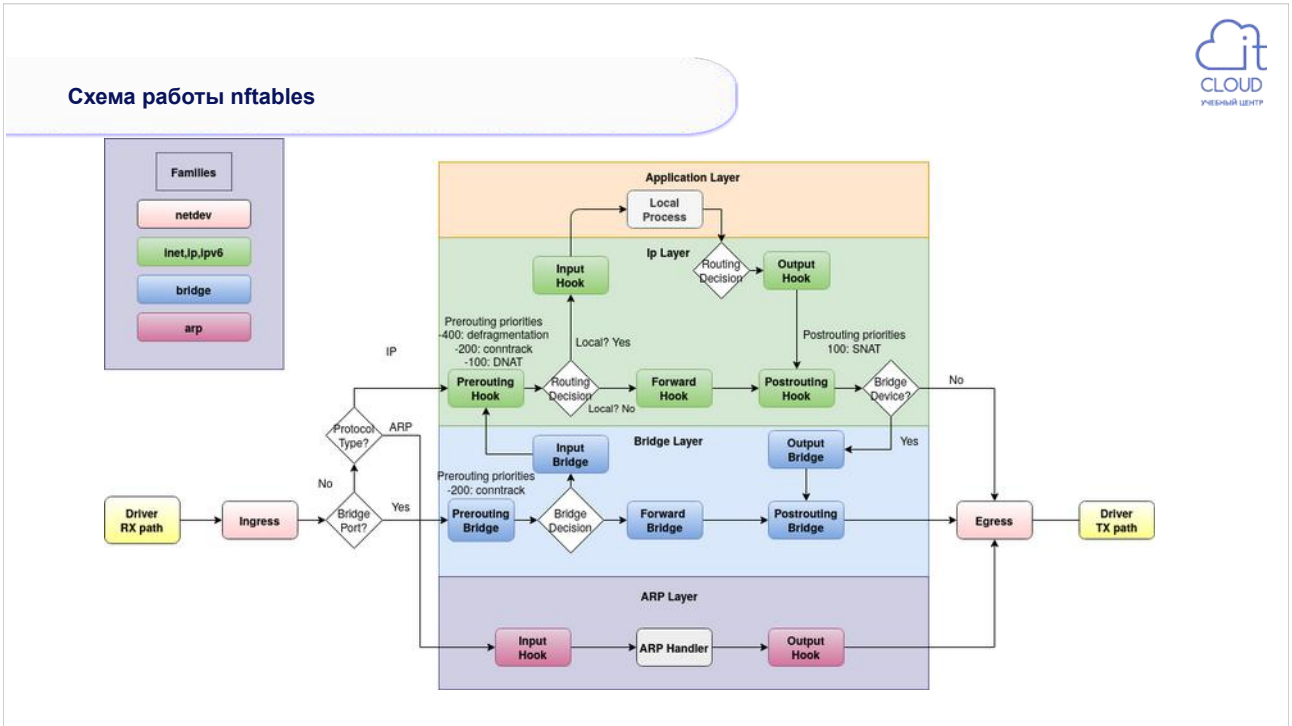
В nftables правила объединяются в цепочки (chain). Задачи цепочек:

1. Определить входную точку для обработки пакетов. Для этого создаются базовые (base) цепочки. В базовых цепочках помимо правил существует еще и политика (policy) — что надо сделать с пакетом который прошел всю цепочку и не соответствует ни одному из правил в этой цепочке.
2. Группировка правил для удобства управления. Для задач группировки создаются регулярные (regular) цепочки.

Ни одно из правил не может быть вне цепочки.

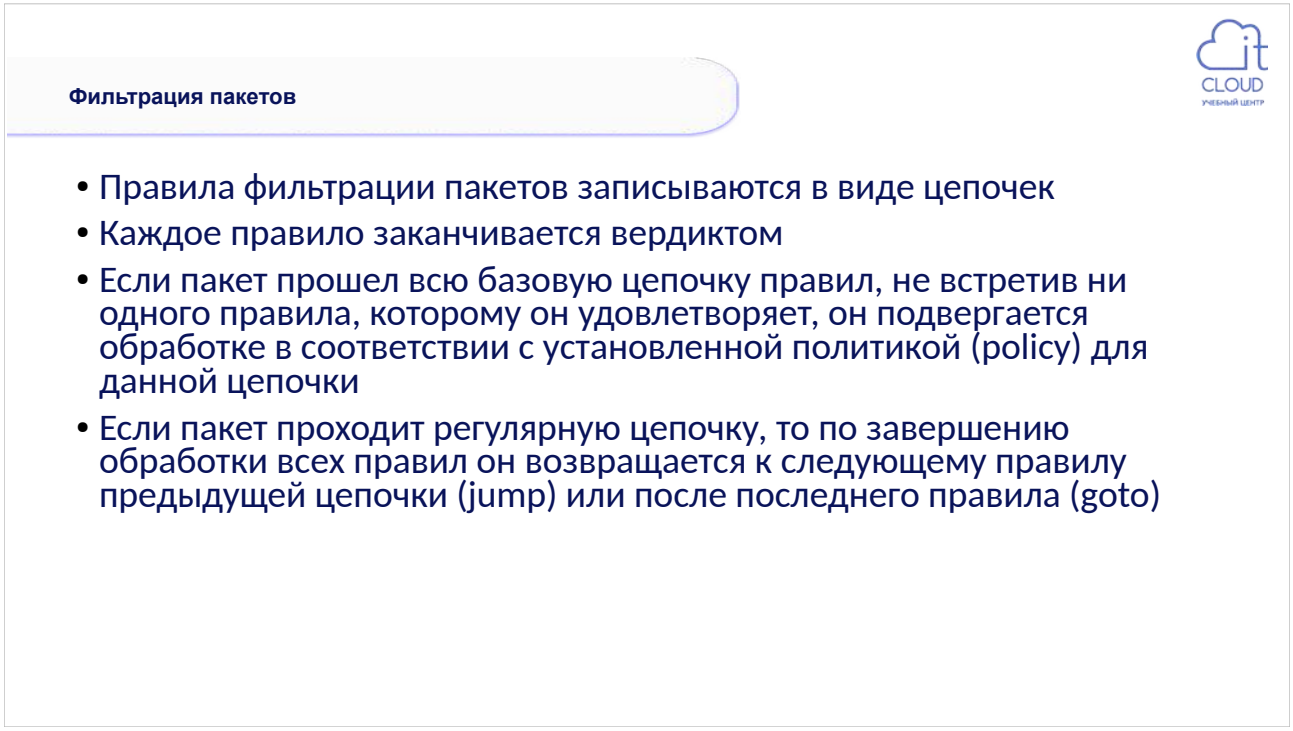
Цепочку в свою очередь объединяются в таблицы, в который помимо цепочек могут быть наборы (set) и объекты отслеживания (stateful objects).

Все вместе взятые таблицы составляют набор правил (ruleset).



На схеме изображена концептуальная схема работы nftables.

Когда пакет попадает на обработку сначала определяется к каким семействам он принадлежит, далее по хукам определяется в каких цепочках нужно обработать этот пакет. Если во всех цепочках пакет будет разрешен, то он будет доставлен в то место, в которое направлялся. Если хотя бы в одной цепочке будет заблокирован, то пакет доставлен не будет.



Правила фильтрации пакетов записываются в виде цепочек (как и в iptables).

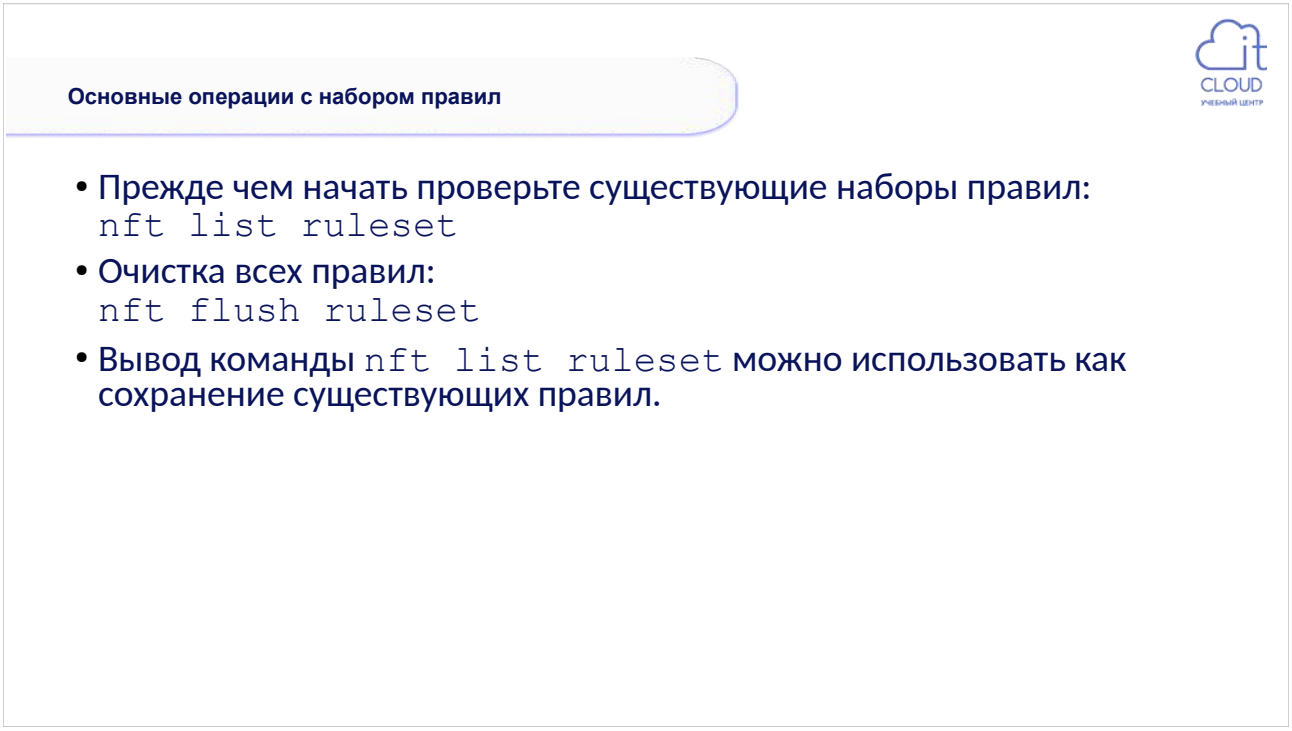
Каждое правило заканчивается вердиктом, обычно, `accept` (пропустить) или `drop` (отбросить)

Заранее определенных цепочек нет.

Если пакет прошел всю базовую цепочку правил, не встретив ни одного правила, которому он удовлетворяет, он подвергается обработке в соответствии с установленной политикой (policy) для данной цепочки: пропустить — `accept` или отклонить — `drop`.

Для регулярных цепочек важно то, как пакет попал на обработку в цепочку. Если в некоторой цепочке используется правило с вердиктом `jump`, и пакет пройдя всю цепочку не совпал ни с одним из правил, то он проверяется на соответствие следующему правилу в цепочке, в которой был прыжок. Если переход был сделан посредством `goto`, то мы продолжаем как-будто прошли все правила в вызывающей цепочки, соответственно будет применена или политика или принято решение как продолжать обработку в вышестоящей цепочке.

8.1.2 Создание таблиц, цепочек и правил.



Все вместе таблицы, цепочки, правила в них и т. д. составляют набор правил (ruleset). Для просмотра набора можно воспользоваться командой `nft list ruleset`:

Пример:

```
root@s10:~# nft list ruleset
table ip filter {
      chain INPUT {
            type filter hook input priority filter; policy accept;
            iifname "lo" counter packets 17 bytes 1241 accept
      }

      chain TEST {
            type filter hook input priority filter; policy accept;
            iifname "enp0s3" tcp dport 22 counter packets 2290 bytes 174448
      }
}
```

У каждого элемента имеется параметр handle, посредством которого можно управлять правилами. Опция `-a` показывает эти хэндлы:

```
root@s10:~# nft -a list ruleset
table ip filter { # handle 2
      chain INPUT { # handle 1
            type filter hook input priority filter; policy accept;
            iifname "lo" counter packets 17 bytes 1241 accept # handle 3
      }

      chain TEST { # handle 2
            type filter hook input priority filter; policy accept;
            iifname "enp0s3" tcp dport 22 counter packets 2318 bytes 176728 #
handle 5
      }
```

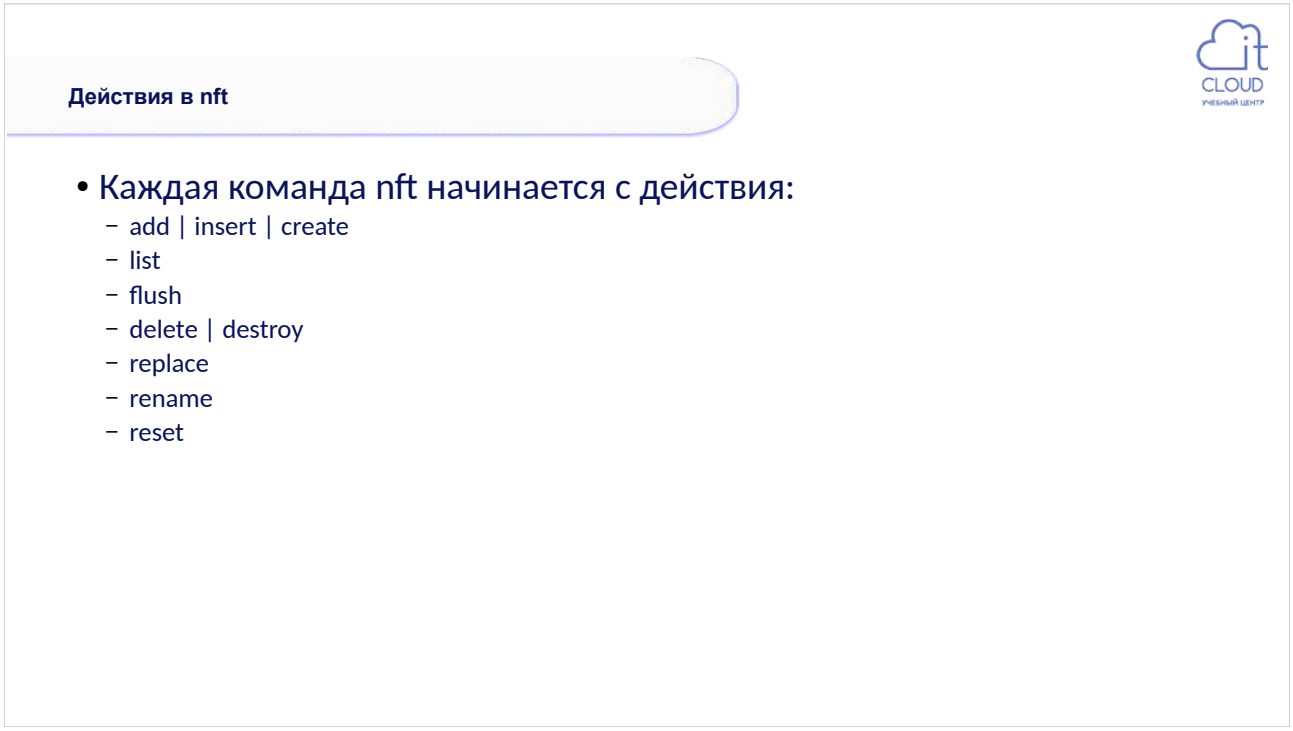
```
}
```

Вы можете редактировать существующие правила или очистить их все:

```
root@s10:~# nft flush ruleset
```

Пример: Сохраним и восстановим правила.

```
root@s10:~# nft list ruleset
table ip filter {
      chain INPUT {
            type filter hook input priority filter; policy accept;
            iifname "lo" counter packets 14 bytes 1022 accept
      }

      chain TEST {
            type filter hook input priority filter; policy accept;
            iifname "enp0s3" tcp dport 22 counter packets 459 bytes 32912
      }
}

root@s10:~# echo flush ruleset > nft.rules
root@s10:~# nft list ruleset >> nft.rules ← Сохранение правил

root@s10:~# nft flush ruleset
root@s10:~# nft list ruleset

root@s10:~# nft -f nft.rules ← Восстановление правил
root@s10:~# nft list ruleset
table ip filter {
      chain INPUT {
            type filter hook input priority filter; policy accept;
            iifname "lo" counter packets 14 bytes 1022 accept
      }

      chain TEST {
            type filter hook input priority filter; policy accept;
            iifname "enp0s3" tcp dport 22 counter packets 600 bytes 43876
      }
}
```



Все команды nft начинаются с действия, т. е. того, что вы хотите сделать.

Действие create это то же самое, что и add, но если объект существует вы получите ошибку. Add для правил означает поставить правило в конец, insert — в начало.

List получить список указанных объектов.

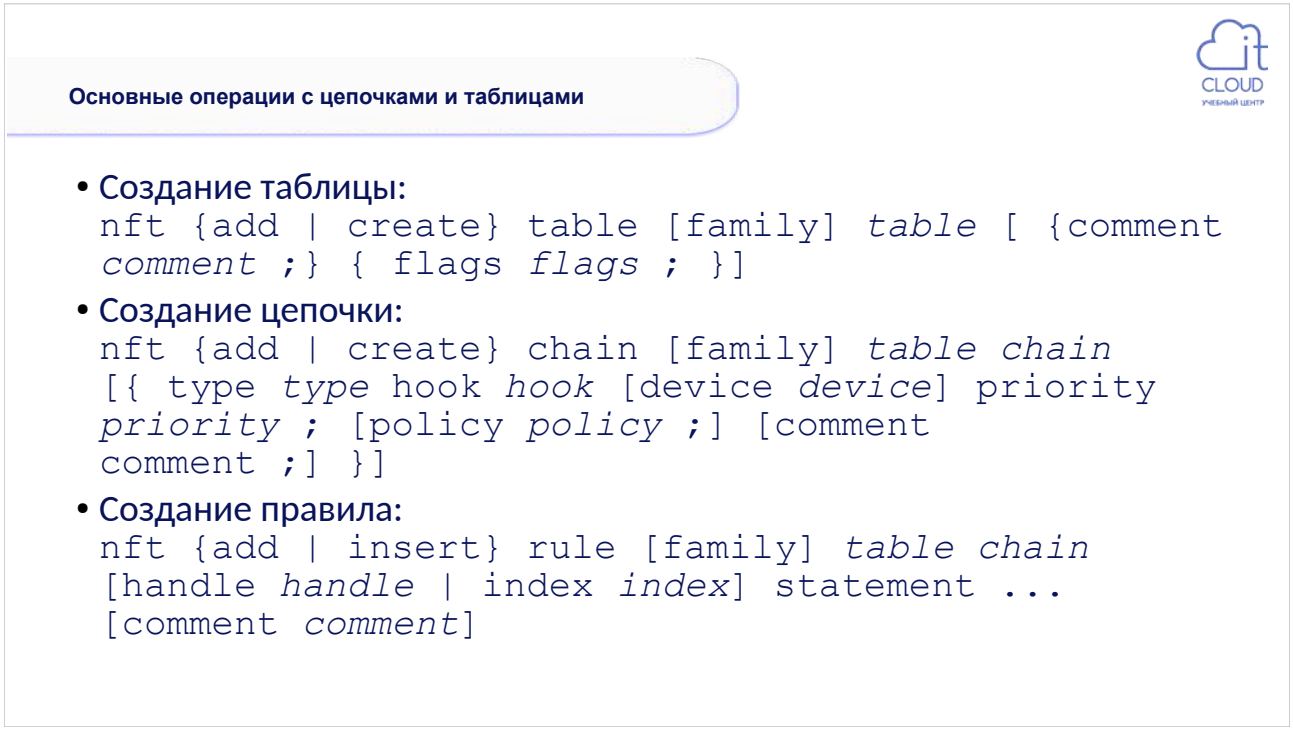
Flush очищает наборы правил, таблицы, цепочки, наборы и т. д.

Delete и destroy удаляют объект, но destroy не вызывает ошибку, если объект не существует.

Replace заменяет указанный объект.

Rename — переименовывает.

Reset — сбрасывает счетчики.



Парадигма настройки в общем виде такая: создаем таблицу, в таблице цепочки, а в цепочках правила.

Пример:

Создаем таблицу:

```
root@s10:~# nft flush ruleset
root@s10:~# nft create table ip MyTable '{ comment "My first table"; }'
root@s10:~# nft list ruleset
table ip MyTable {
      comment "My first table"
}
```

В таблице создаем цепочки:

```
root@s10:~# nft add chain MyTable input '{ type filter hook input priority
filter; policy accept; comment "Base chain for INPUT packets"; }'

root@s10:~# nft add chain MyTable SSH-IN '{  comment "Regular chain for SSH
input control"; }'

root@s10:~# nft list ruleset
table ip MyTable {
      comment "My first table"
      chain input {
            comment "Base chain for INPUT packets"
            type filter hook input priority filter; policy accept;
      }

      chain SSH-IN {
            comment "Regular chain for SSH input control"
      }
}
```

Добавляем правила в цепочки:

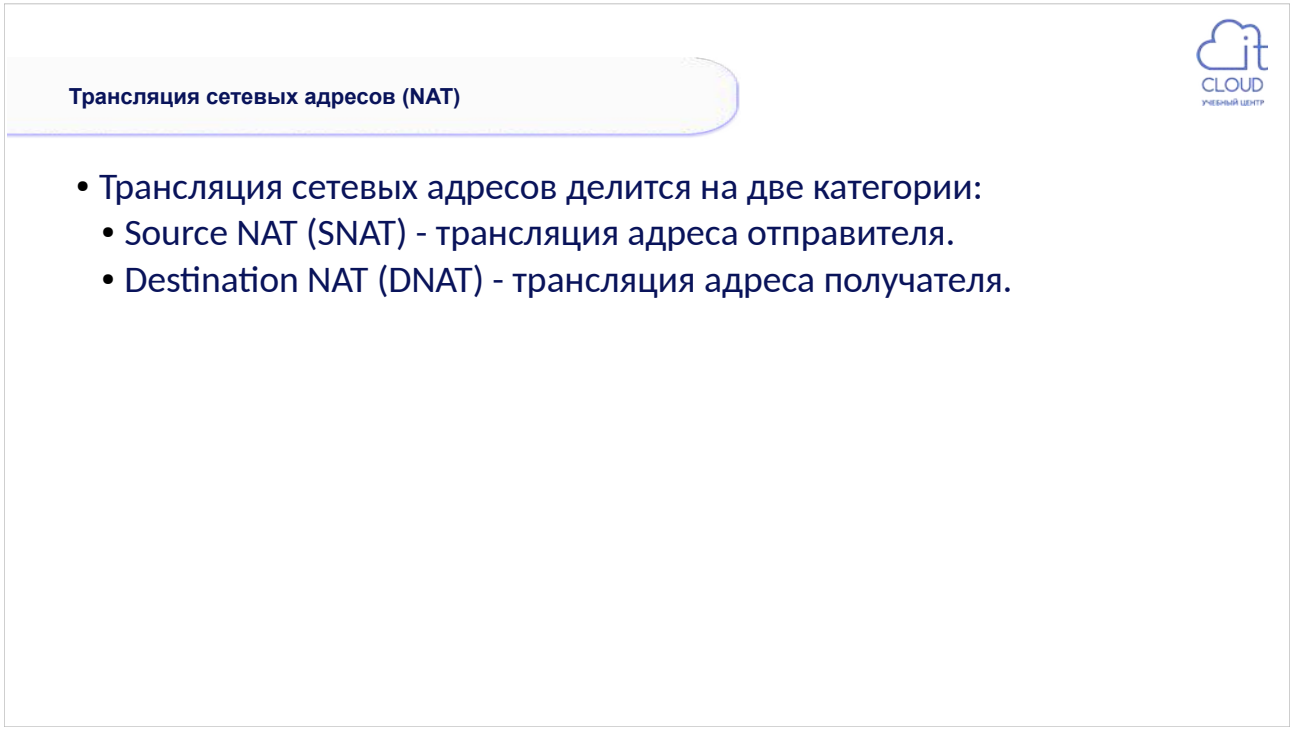
```
root@s10:~# nft add rule MyTable input ct state established,related accept
```

```
root@s10:~# nft add rule MyTable input iif "lo" ip saddr 127.0.0.0/8 accept
root@s10:~# nft add rule MyTable input iif "lo" ip saddr != 127.0.0.0/8 drop
root@s10:~# nft add rule MyTable input tcp dport 22 jump SSH-IN

root@s10:~# nft add rule MyTable SSH-IN ip saddr 127.0.0.0/8 accept
root@s10:~# ip -4 -br ad
lo               UNKNOWN        127.0.0.1/8
enp0s3           UP             10.1.1.4/24
root@s10:~# nft add rule MyTable SSH-IN ip saddr 10.1.1.0/24 accept
root@s10:~# nft add rule MyTable SSH-IN drop

root@s10:~# nft list ruleset
table ip MyTable {
      comment "My first table"
      chain input {
            comment "Base chain for INPUT packets"
            type filter hook input priority filter; policy accept;
            ct state established,related accept
            iif "lo" ip saddr 127.0.0.0/8 accept
            iif "lo" ip saddr != 127.0.0.0/8 drop
            tcp dport 22 jump SSH-IN
      }

      chain SSH-IN {
            comment "Regular chain for SSH input control"
            ip saddr 127.0.0.0/8 accept
            ip saddr 10.1.1.0/24 accept
            drop
      }
}
```

Заменим некоторые правила:

```
root@s10:~# nft -a list ruleset
table ip MyTable { # handle 4
      comment "My first table"
      chain input { # handle 4
            comment "Base chain for INPUT packets"
            type filter hook input priority filter; policy accept;
            ct state established,related accept # handle 8
            iif "lo" ip saddr 127.0.0.0/8 accept # handle 9
            iif "lo" ip saddr != 127.0.0.0/8 drop # handle 10
            tcp dport 22 jump SSH-IN # handle 11
      }

      chain SSH-IN { # handle 5
            comment "Regular chain for SSH input control"
            ip saddr 127.0.0.0/8 accept # handle 12
            ip saddr 10.1.1.0/24 accept # handle 13
            drop # handle 14
      }
}
root@s10:~# nft replace rule MyTable SSH-IN handle 14 counter drop
root@s10:~# nft -a list ruleset
table ip MyTable { # handle 4
      comment "My first table"
      chain input { # handle 4
            comment "Base chain for INPUT packets"
            type filter hook input priority filter; policy accept;
            ct state established,related accept # handle 8
```

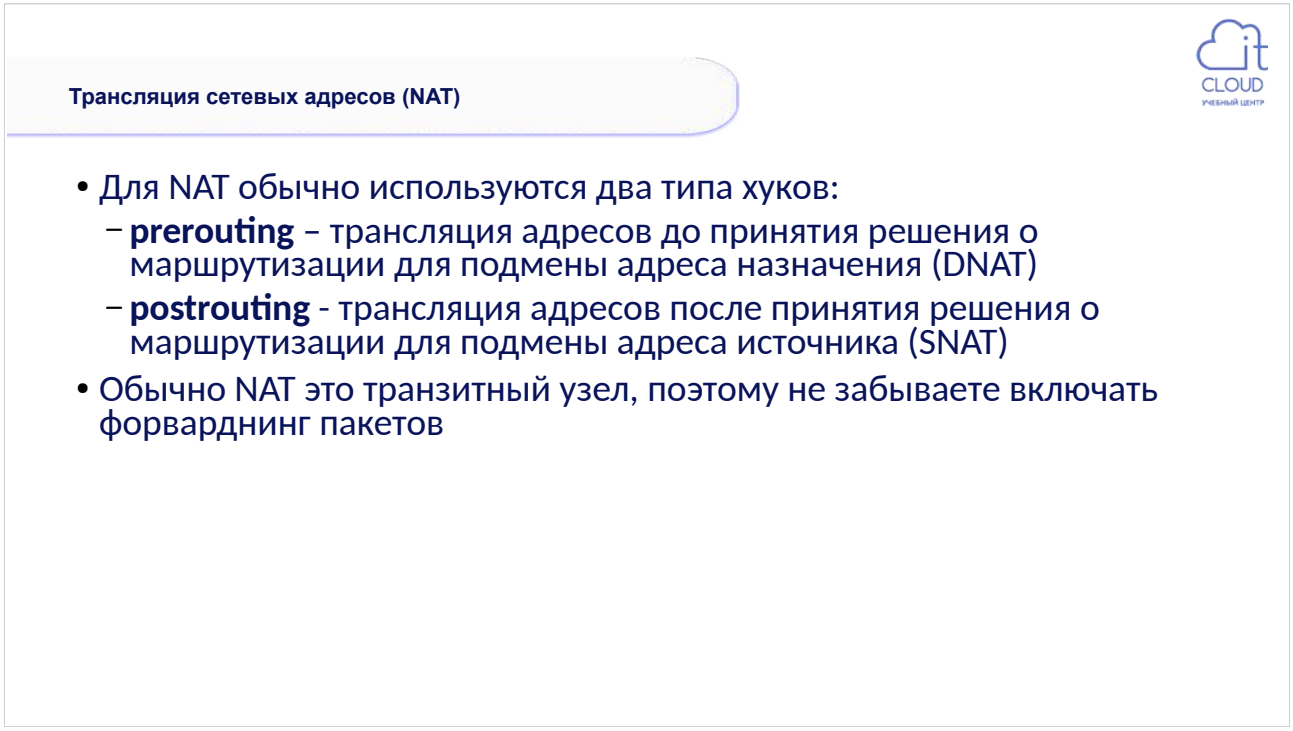
```
            iif "lo" ip saddr 127.0.0.0/8 accept # handle 9
            iif "lo" ip saddr != 127.0.0.0/8 drop # handle 10
            tcp dport 22 jump SSH-IN # handle 11
      }

      chain SSH-IN { # handle 5
            comment "Regular chain for SSH input control"
            ip saddr 127.0.0.0/8 accept # handle 12
            ip saddr 10.1.1.0/24 accept # handle 13
            counter packets 0 bytes 0 drop # handle 14
      }
}
```



Применение трансляции сетевых адресов вызывает изменения в адресах отправителя или получателя пакетов. При этом происходит запоминание информации об измененном пакете так, чтобы для ответного пакета можно было провести обратное преобразование.

Трансляцию сетевых адресов принято разделять на две категории:

1. Source NAT (`SNAT`) - трансляция адреса отправителя.
2. Destination NAT (`DNAT`) - трансляция адреса получателя.



Для выполнения трансляции адресов разумно будет создать отдельную таблицу, но это не является обязательным.

Так же следует включить форвардинг пакетов, т. е. разрешить пересылку транзитного трафика:

```
root@s10:~# grep ip_forward /etc/sysctl.conf
net.ipv4.ip_forward=1
root@s10:~# sysctl -p /etc/sysctl.conf
net.ipv4.ip_forward = 1
```

Всего имеется четыре хука для трансляции адресов prerouting, postrouting, input и output. На практике обычно применяются первые два.

Prerouting выполняется перед принятием решения о маршрутизации и предназначен для перенаправления пакета к другому место назначению или, другими словами, для подмены destination адреса (DNAT).

Postrouting выполняется после принятие решения об отправке и предназначен для скрытия адреса источника, т. е. подменяет source адрес (SNAT).

Пример: Простой NAT (masquerade), когда у всех исходящих пакетов с интерфейса подменяются адрес источника на адрес этого интерфейса.

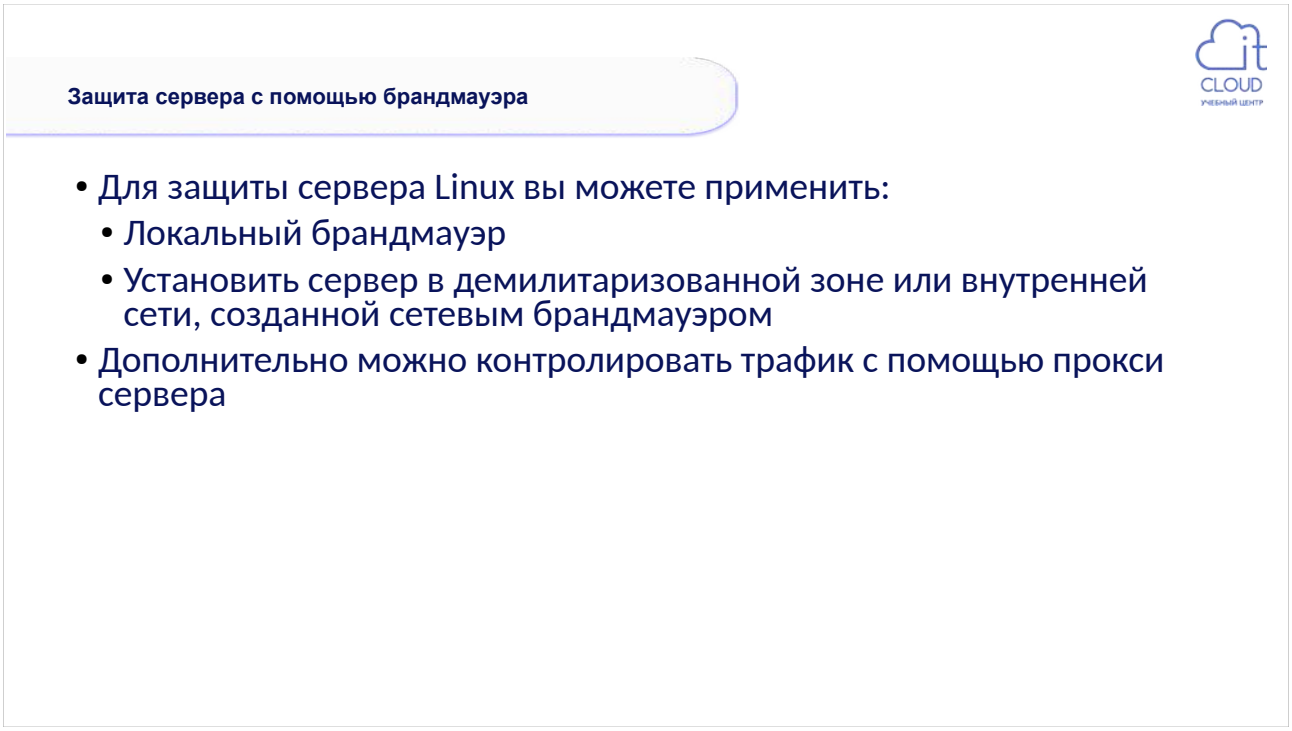
```
root@s10:~# nft add table ip NAT
root@s10:~# nft create chain NAT s-nat '{ type nat hook postrouting priority 0 ;
comment "My SNAT chain" ; }'
root@s10:~# nft add rule NAT s-nat oif enp0s3 counter masquerade

root@s10:~# nft -a list table NAT
table ip NAT { # handle 4
      chain s-nat { # handle 1
            comment "My SNAT chain"
            type nat hook postrouting priority filter; policy accept;
```

```
            oif "enp0s3" counter packets 2 bytes 168 masquerade # handle 3
      }
}
```

Пример: публикация веб сервера с адресом 10.2.2.2, который находится за NAT-ом, у которого адрес 10.1.1.5.

```
root@s10:~# nft insert rule NAT s-nat oif enp0s3 ip saddr 10.2.2.2 tcp sport
8080 snat to 10.1.1.5:80
root@s10:~# nft insert rule NAT d-nat iif enp0s3 ip daddr 10.1.1.5 tcp dport 80
dnat to 10.2.2.2:8080
root@s10:~# nft -a list table NAT
table ip NAT { # handle 4
      chain s-nat { # handle 1
            comment "My SNAT chain"
            type nat hook postrouting priority filter; policy accept;
            oif "enp0s3" ip saddr 10.2.2.2 tcp sport 8080 snat to 10.1.1.5:80 #
handle 7
            oif "enp0s3" counter packets 6 bytes 973 masquerade # handle 3
      }

      chain d-nat { # handle 4
            comment "My DNAT chain"
            type nat hook prerouting priority filter; policy accept;
            iif "enp0s3" ip daddr 10.1.1.5 tcp dport 80 dnat to 10.2.2.2:8080 #
handle 8
      }
}
```

8.2 Защита сервера Linux с помощью межсетевого экрана.



Устанавливая сервер для обслуживания клиентских запросов обдумайте вариант защиты сетевых соединений к этому серверу.

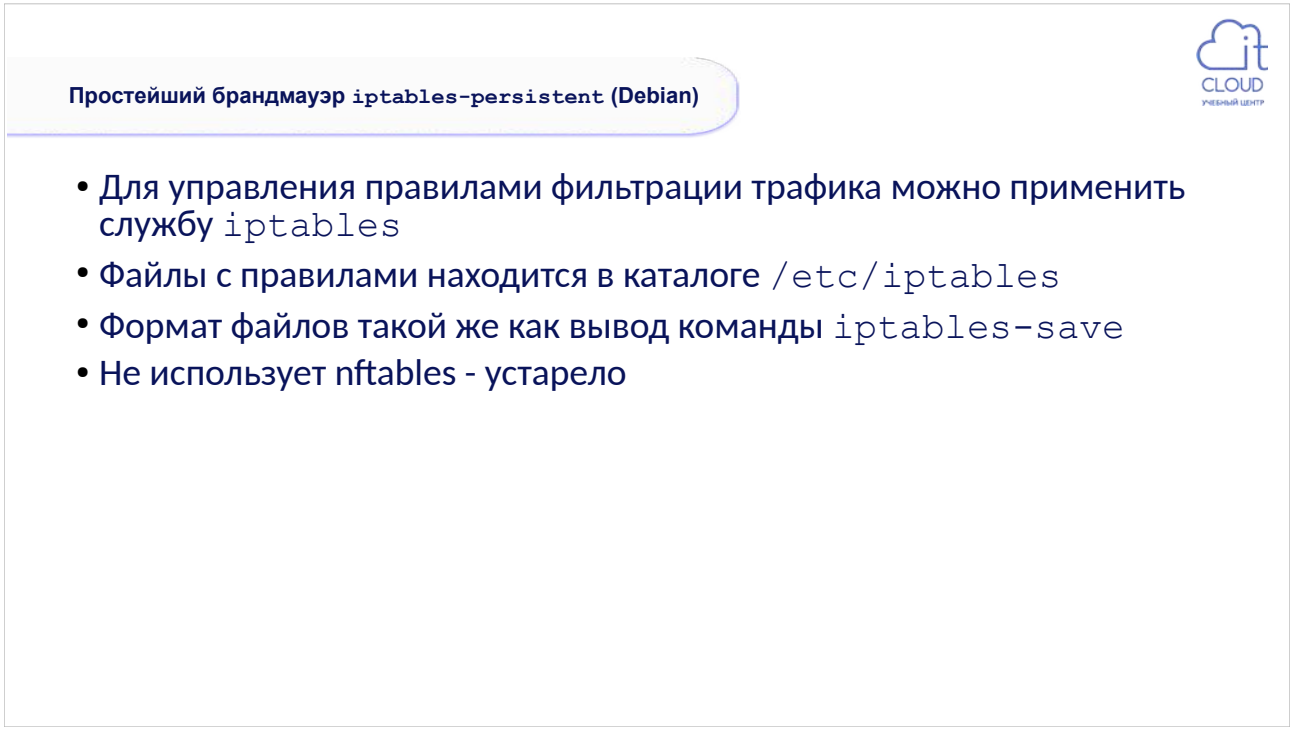
Вы можете установить сервер напрямую подключенным к интернету, тогда для его защиты вам необходимо применять локальный брандмауэр. И только локальный брандмауэр будет защищать ваш сервер от внешних угроз. При этом возможности хостовых брандмауэров как правило ограничены.

Лучшим, с точки зрения безопасности, будет установить на границе вашей сети специализированный брандмауэр для контроля сетевого трафика всей сети. Такие системы могут, в том числе, использовать в качестве основы Linux. При этом установка централизованного решения не отменяет настройку локальной защиты.

Преимущества специализированных решений:

- Поддержка продвинутых механизмов маршрутизации.
- Настройка ориентированная на обеспечение безопасности.
- Продвинутая поддержка VPN.
- В некоторых случаях контентная фильтрация.

Одной из мер защиты может рассматриваться применение прокси серверов для фильтрации и оптимизации трафика.



В Debian подобных системах в качестве базового решения для фильтрации трафика можно применить службу iptables (или netfilter-persistent). Для использования службы нужно установить пакет iptables-persistent. Это простая служба, которая при своем старте загружает правила фильтрации трафика из файла.

Конфигурационные файлы в каталоге `/etc/iptables` содержит правила фильтрации.

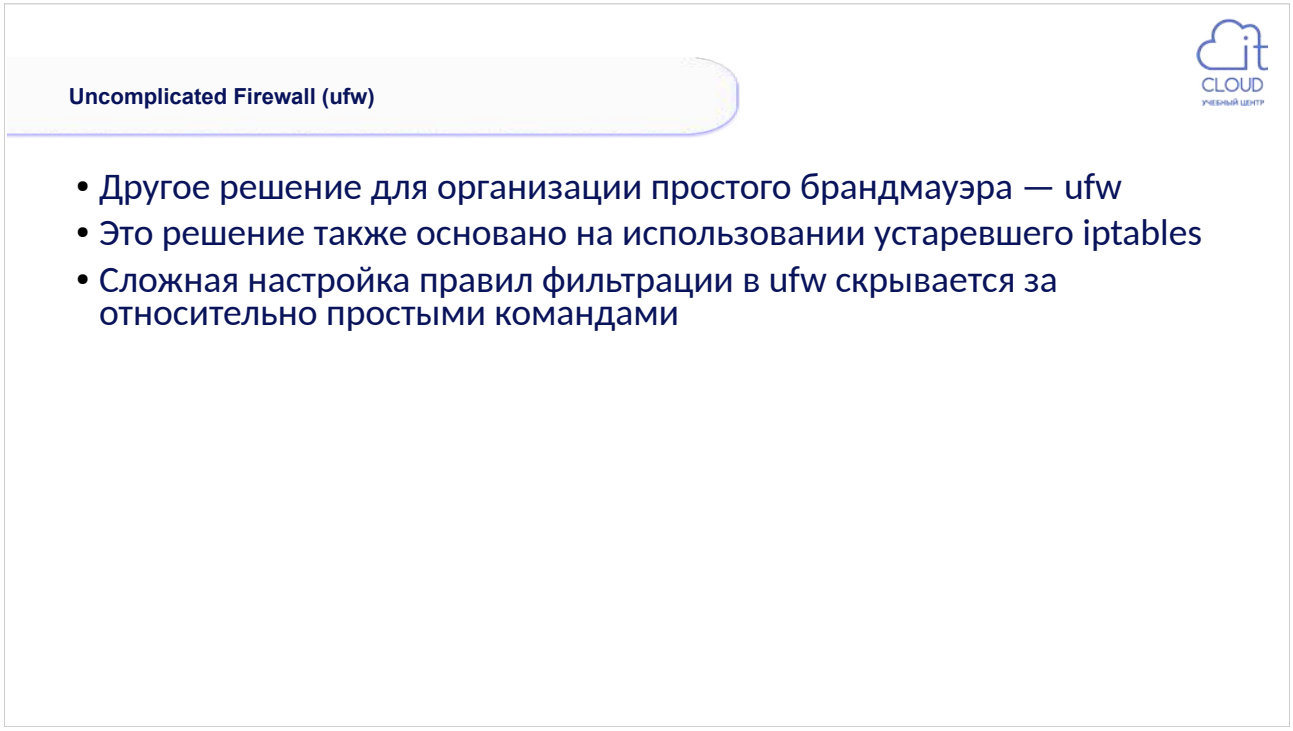
Формат конфигурационного файла такой же как и вывод команды `iptables-save`.

Так как служба использует устаревший способ фильтрации, то в будущем планируется отказаться от нее. Вместо нее применяется служба firewalld.

Пример:

```
root@s10:~# iptables -I INPUT -i lo -j ACCEPT
root@s10:~# iptables -t nat -I POSTROUTING -o enp0s3 -j MASQUERADE
root@s10:~# /etc/init.d/netfilter-persistent save
Saving netfilter rules...run-parts: executing
/usr/share/netfilter-persistent/plugins.d/15-ip4tables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/25-ip6tables save
done.
root@s10:~# cat /etc/iptables/rules.v4
# Generated by iptables-save v1.8.9 (nf_tables) on Thu Feb  6 14:39:43 2025
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -i lo -j ACCEPT
COMMIT
# Completed on Thu Feb  6 14:39:43 2025
# Generated by iptables-save v1.8.9 (nf_tables) on Thu Feb  6 14:39:43 2025
*nat
:PREROUTING ACCEPT [0:0]
:INPUT ACCEPT [0:0]
```

```
:OUTPUT ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
-A POSTROUTING -o enp0s3 -j MASQUERADE
COMMIT
# Completed on Thu Feb  6 14:39:43 2025
```



Другое, более продвинутое, решение для локальной защиты — Uncomplicated Firewall (UFW). Внутреннее устройство ufw чем-то похоже на netfilter-persistent, так же использует iptables, правила тоже в текстовых файлах. И правила считываются и применяются при старте системы.

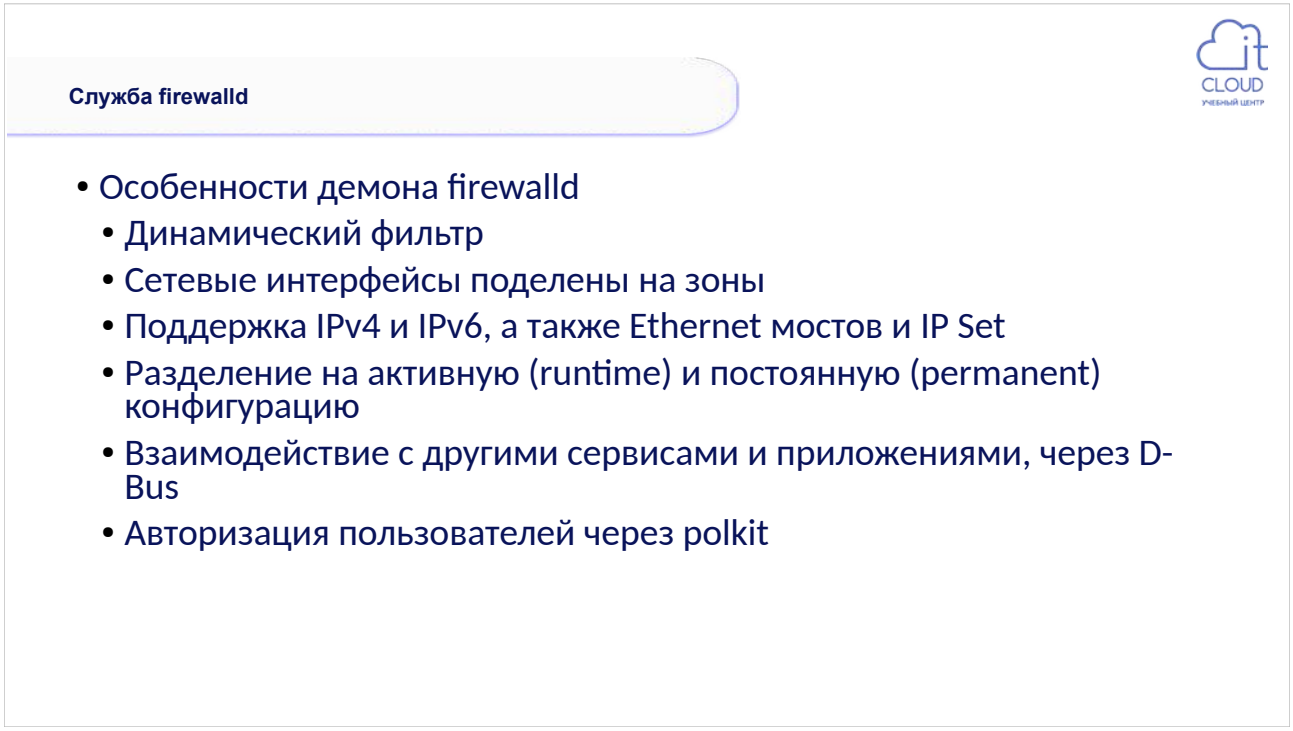
Преимуществом ufw является его относительная простота при решении стандартных задач: открыть или закрыть порт, предоставить доступ приложению и пр.

В силу того, что используется устаревший движок, то же стоит рассмотреть другие, более современные решения.

Пример:

```
root@s10:~# ufw status
Status: inactive

root@s10:~# ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup

root@s10:~# ufw status
Status: active

root@s10:~# ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), deny (routed)
New profiles: skip

root@s10:~# ufw show added
Added user rules (see 'ufw status' for running firewall):
(None)
root@s10:~# ufw allow ssh
Rule added
```

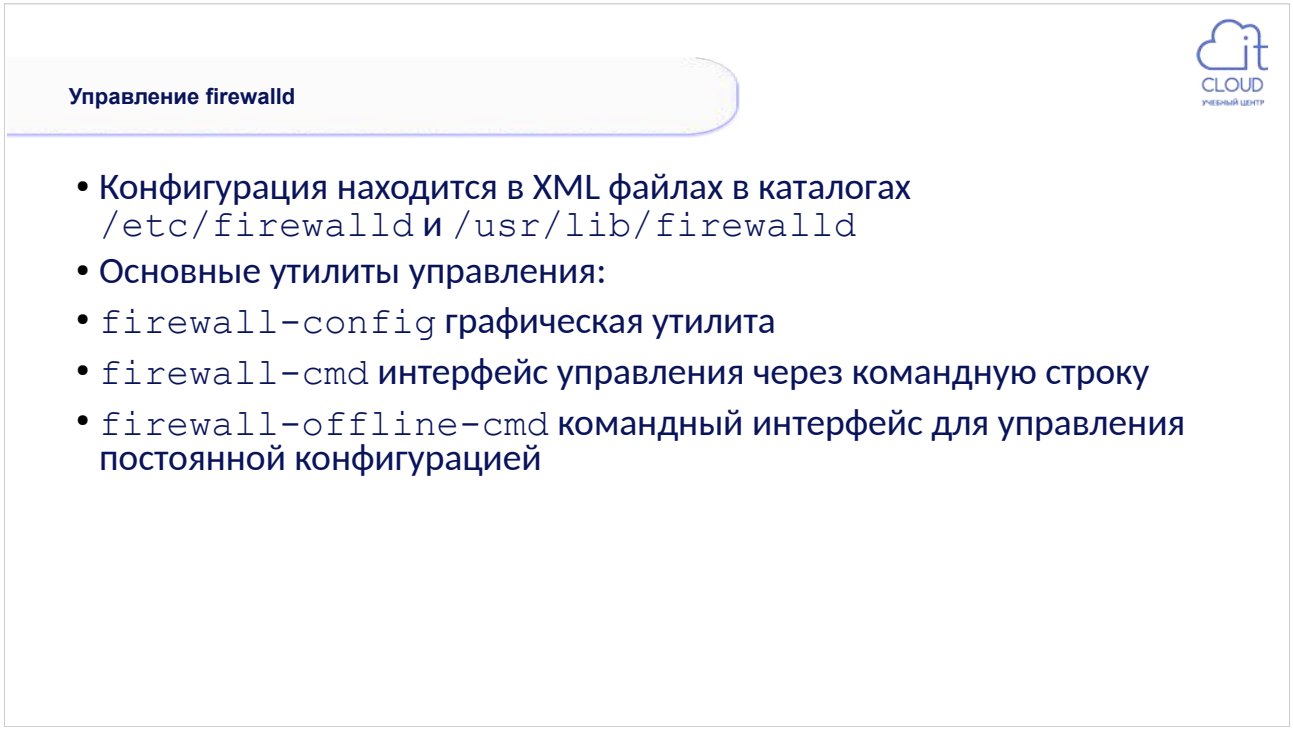
```
Rule added (v6)
root@s10:~# ufw show added
Added user rules (see 'ufw status' for running firewall):
ufw allow 22/tcp

root@s10:~# ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)

root@s10:~# ufw allow CIFS
Rule added
Rule added (v6)

root@s10:~# ufw app info CIFS
Profile: CIFS
Title: SMB/CIFS server
Description: SMB/CIFS server

Ports:
  137,138/udp
  139,445/tcp

root@s10:~# ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
CIFS                       ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
CIFS (v6)                  ALLOW       Anywhere (v6)
```



Имеются несколько сценариев, в которых применение служб iptables или ufw является нецелесообразным. Например: вы хотите применять разные правила фильтрации трафика для разных интерфейсов или разных сетей. Или возможно вы захотите делегировать полномочия на управление брандмауэром. Для решения этих задач предназначена служба firewalld, которая является надстройкой над nftables.

Основные характеристики службы firewalld:

- Динамический управление межсетевым экраном.
- Сетевые интерфейсы поделены на зоны. Зоны определяют наборы применяемых правил фильтрации.
- Поддержка IPv4 и IPv6, а также Ethernet мостов и IP Set.
- Разделение на активную (runtime) и постоянную (permanent) конфигурацию.
- Взаимодействие с другими сервисами и приложениями, через D-Bus.
- Авторизация пользователей через polkit.



Настройки демона firewalld находятся в XML файлах, за исключением основного конфигурационного файла `/etc/firewalld/firewalld.conf`.

Пример: Просмотр состояния демона `firewalld`, и его запуск.

```
root@s10:~# firewall-cmd --state
not running
root@s10:~# systemctl enable firewalld
root@s10:~# systemctl start firewalld
root@s10:~# firewall-cmd --state
running
```

Пример: получение информации о зоне. И определение интерфейса в зону.

```
root@s10:~# firewall-cmd --get-default-zone
public
root@s10:~# firewall-cmd --zone=public --list-services
ssh dhcpv6-client
root@s10:~# firewall-cmd --get-active-zones
public
  interfaces: enp0s8
root@s10:~# ip -4 -br ad
lo               UNKNOWN        127.0.0.1/8
enp0s3           UP             10.1.1.5/24
enp0s8           UP             10.2.2.1/24
root@s10:~# firewall-cmd --get-zone-of-interface=enp0s3
no zone
root@s10:~# firewall-cmd --add-interface=enp0s3 --zone=external
success
root@s10:~# firewall-cmd --get-zone-of-interface=enp0s3
external
root@s10:~# reboot
root@s10:~# firewall-cmd --get-zone-of-interface=enp0s3
no zone
root@s10:~# firewall-cmd --add-interface=enp0s3 --permanent --zone=external
```

```
success
root@s10:~# firewall-cmd --add-interface=enp0s3  --zone=external
success
root@s10:~# systemctl restart firewalld.service
root@s10:~# firewall-cmd --get-zone-of-interface=enp0s3
external
```

Пример: добавление служб в нужную зону.

```
root@s10:~# firewall-cmd --list-services --zone=external
ssh
root@s10:~# firewall-cmd --add-service=http --add-service=https --zone=external
success
root@s10:~# firewall-cmd --add-service=http --add-service=https --permanent --
zone=external
success
root@s10:~# firewall-cmd --list-services --zone=external
http https ssh
root@s10:~# firewall-cmd --list-services --permanent --zone=external
http https ssh
```

Пример: добавление порта TCP в нужную зону.

```
root@s10:~#  firewall-cmd --add-port=22222/tcp --permanent --zone=work
success
root@s10:~#  firewall-cmd --list-ports --zone=work

root@s10:~#  firewall-cmd --add-port=22222/tcp  --zone=work
success
root@s10:~#  firewall-cmd --list-ports --zone=work
22222/tcp
root@s10:~#  firewall-cmd --list-ports --zone=work --permanent
22222/tcp
```

Пример: Настройка NAT.

```
root@s10:~# firewall-cmd --get-active-zones
external
  interfaces: enp0s3
home
  interfaces: enp0s8

root@s10:~# firewall-cmd --zone=external --list-all
external (active)
  target: default
  icmp-block-inversion: no
  interfaces: enp0s3
  sources:
  services: http https ssh
  ports:
  protocols:
  forward: yes
  masquerade: yes
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:

root@s10:~# firewall-cmd --zone=home --list-all
```

```
home (active)
  target: default
  icmp-block-inversion: no
  interfaces: enp0s8
  sources:
  services: dhcpv6-client mdns samba-client ssh
  ports:
  protocols:
  forward: yes
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:

sa@client:~$ ping -c1 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
From 10.2.2.1 icmp_seq=1 Packet filtered

--- 8.8.8.8 ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms

root@s10:~# firewall-cmd --remove-interface=enp0s8
You're performing an operation over default zone ('public'),
but your connections/interfaces are in zone 'external,home' (see --get-active-
zones)
You most likely need to use --zone=external option.

success
root@s10:~# firewall-cmd --add-interface=enp0s8 --zone=external
success

sa@client:~$ ping -c1 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=107 time=48.9 ms

--- 8.8.8.8 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 48.947/48.947/48.947/0.000 ms
```

Видим, что трафик между зонами блокируется, но когда мы перевели оба интерфейса в одну зону, то все заработало. Делаем вывод, что любой трафик между зонами по умолчанию запрещен и нужно что-то сделать, чтобы его разрешить.

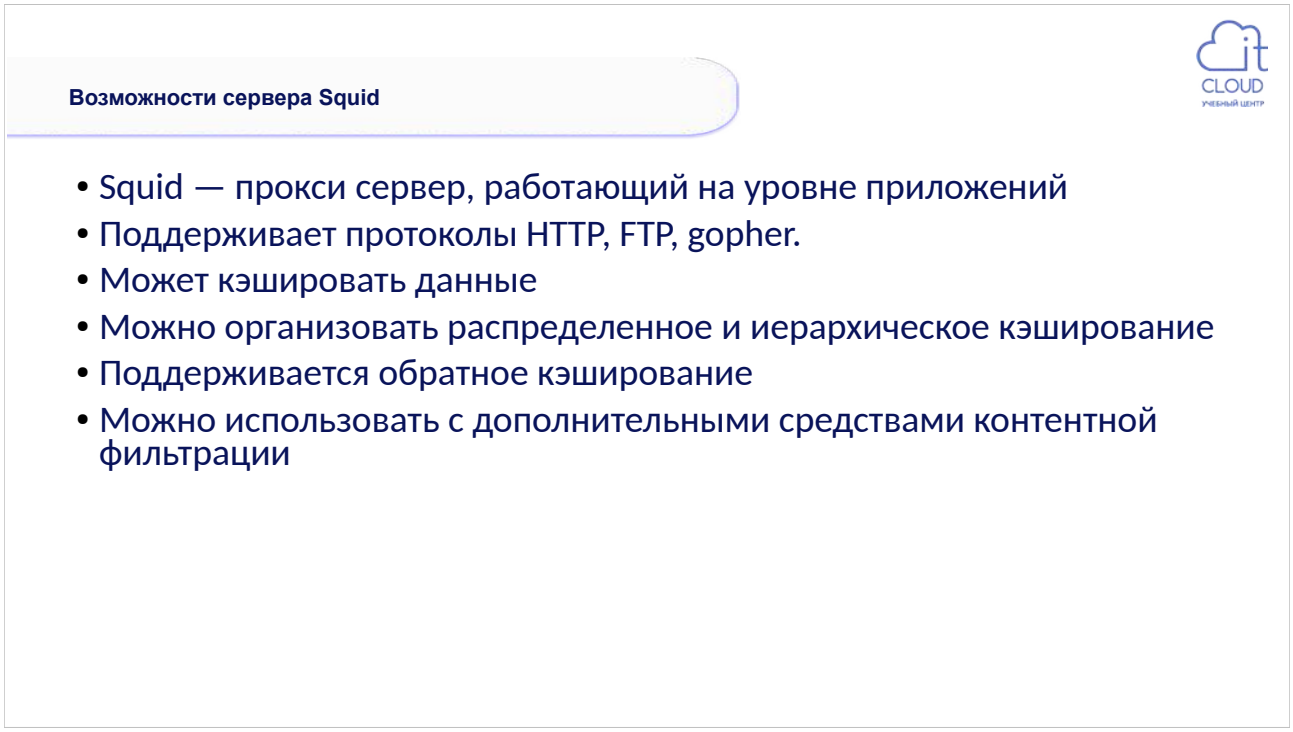
Снова вернем enp0s8 интерфейс в зону home и попробуем другой способ.

```
root@s10:~# firewall-cmd --remove-interface=enp0s8
You're performing an operation over default zone ('public'),
but your connections/interfaces are in zone 'external' (see --get-active-zones)
You most likely need to use --zone=external option.

success
root@s10:~# firewall-cmd --add-interface=enp0s8 --zone=home
success

root@s10:~# firewall-cmd --permanent --new-policy FromHomeToExternal
success
```

```
root@s10:~# firewall-cmd --permanent --policy FromHomeToExternal --add-ingress-
zone home
success
root@s10:~# firewall-cmd --permanent --policy FromHomeToExternal --add-egress-
zone external
success
root@s10:~# firewall-cmd --permanent  --policy=FromHomeToExternal --add-
protocol=icmp
success
root@s10:~# firewall-cmd --reload
success
root@s10:~# firewall-cmd --info-policy=FromHomeToExternal
FromHomeToExternal (active)
  priority: -1
  target: CONTINUE
  ingress-zones: home
  egress-zones: external
  services:
  ports:
  protocols: icmp
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:

sa@client:~$ ping -c1 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=107 time=47.3 ms

--- 8.8.8.8 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 47.312/47.312/47.312/0.000 ms
```

В примере выше мы создали политику, которая разрешает протокол ICMP и переменили ее к пакетом, которые идут из зоны home в зону external. После этого пинги заработали. По умолчанию трафик между зонами запрещен, чтобы его разрешить необходимо создавать политики.

8.3 Защита сервера Linux с помощью прокси-сервера squid.



Программа Squid является прокси сервером. Прокси серверы работают, как посредники между клиентом и сервером.

С точки зрения клиента прокси – это сервер, а с точки зрения сервера – клиент.

В отличие от фильтров пакетов, работающих на сетевом уровне, прокси серверы работают на прикладном уровне (application layer).

Прокси серверы способны фильтровать трафик на основе анализа “сути” передаваемых данных. Squid поддерживает следующие протоколы: HTTP, FTP, Gopher. Также поддерживается шифрованная передача данных посредством туннелирования SSL.

По умолчанию Squid прослушивает порт 3128 TCP. Браузеры должны быть настроены на использование порта, прослушиваемого Squid. Порт, прослушиваемый Squid может быть изменен с помощью настройки `http_port` в файле конфигурации.

Squid кэширует передаваемые для клиента объекты, позволяя при повторном обращении к этим объектам извлекать их из кэша. Это значительно ускоряет повторное обращение к требуемым объектам.

Кэш представляет собой набор каталогов.

Имеется специализированный режим работы Squid, называемый режимом акселерации (accelerator mode). В этом режиме Squid снимает часть нагрузки с HTTP сервера, отвечая на внешние запросы, поступающие из Internet.

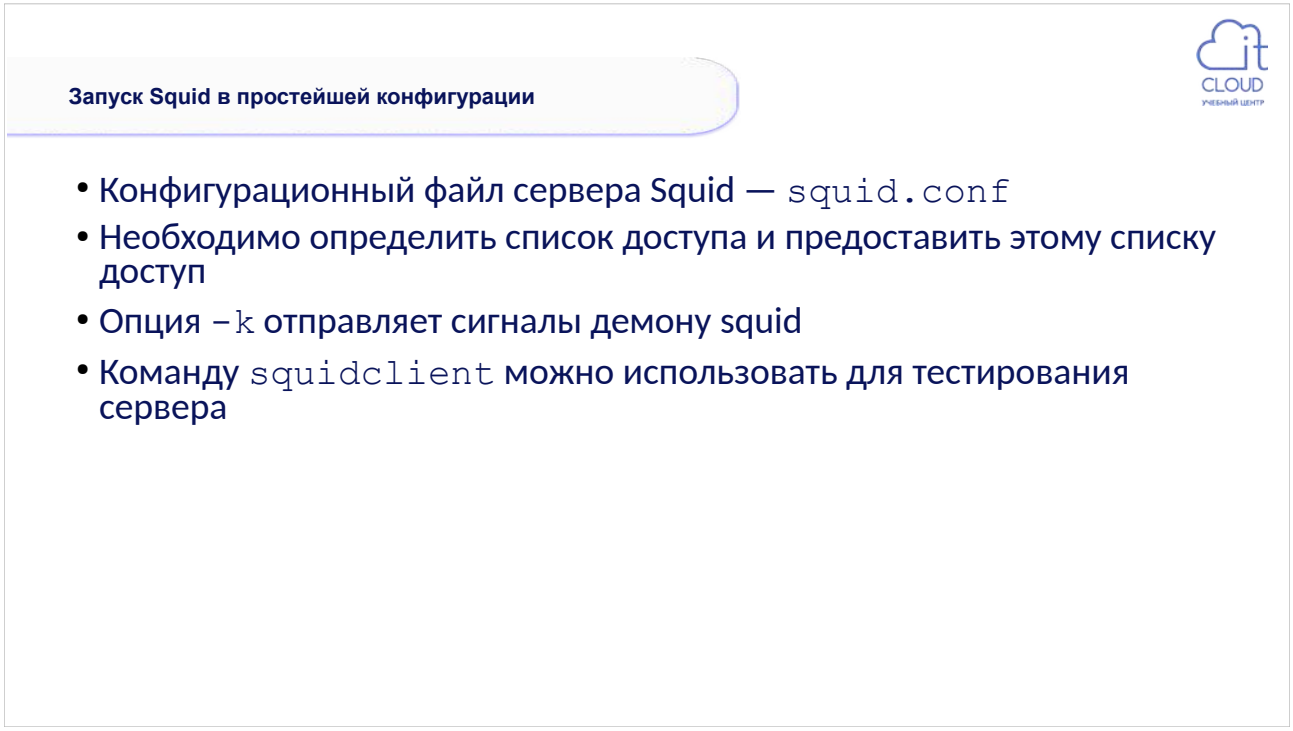
Обычный кэширующий режим предназначен для ускорения получения информации внутренними клиентами, а режим акселерации, напротив, внешними.

Squid предоставляет средства контроля доступа к нему с помощью ACL (Access Control Lists).

При необходимости можно выстраивать иерархические структуры из взаимодействующих серверов Squid. Имеется специальный протокол, обеспечивающий такое взаимодействие – ICP (Internet Cache Protocol).

При старте Squid запускается демон squid и демон dnsserver. Первый демон является собственно прокси сервером, а второй выполняет запросы к DNS.

Squid может быть настроен на использование дополнительных средств контентной фильтрации трафика. Например squidguard.



Конфигурационный файл сервера Squid - `squid.conf` обычно в GNU/Linux располагается в каталоге `/etc/squid`.

В случае установки Squid не из пакета, а в ручную, он располагается в подкаталоге squid базового каталога установки, указанного опцией `--prefix`.

Для запуска сервера Squid в простейшей конфигурации достаточно определить список доступа (ACL - Access Control List) по некоторому критерию, например, принадлежности клиентов к требуемому блоку IP адресов, и разрешить им доступ.

Списки доступа определяются с помощью ключевого слова `acl`.

Определить список доступа по критерию IP адресов источников (source addresses), то есть, адресов клиентов можно, используя конструкцию:

```
acl aclname src ip-address/netmask
```

Пример: Список доступа определен методом указания подсети и ее маски.

```
acl my_net src 192.168.111.0/255.255.255.192
```

После определения списка доступа необходимо разрешить или, наоборот, запретить доступ для объектов, входящих в этот список. Это достигается с помощью ключевого слова `http_access`.

Синтаксис: `http_access allow|deny [!]aclname`.

Доступ может быть либо разрешен (`allow`), либо запрещен (`deny`). Если перед именем ACL стоит знак восклицания, то это обозначает “кроме”.

Доступ для определенного ACL должен быть разрешен до строки конфигурации, запрещающей доступ для всех, которая по умолчанию имеется в файле конфигурации.

Пример:

```
http_access allow my_net
http_access deny all
```

Примечание: *Здесь для списка доступа my_net разрешен доступ к Squid. Следующей строкой запрещен доступ для списка доступа all. Он определен в файле конфигурации Squid, как acl all src 0.0.0.0/0.0.0.0.*

Перед запуском Squid полезно проверить его конфигурацию. Это достигается с помощью команды:

```
squid -k parse
```

В распространенных дистрибутивах GNU/Linux сервер Squid, установленный из пакета, сразу обладает каталогом кэша. Однако при самостоятельной сборке Squid этот каталог автоматически создан не будет. Для его создания используйте команду `squid -z`

При запущенном сервере Squid проверить его можно с помощью команды

```
squid -k check
```

Если необходимо изменить конфигурацию Squid, то после внесения изменений в файл конфигурации следует послать сигнал реконфигурации серверу Squid. Это достигается с помощью команды `squid -k reconfigure`.

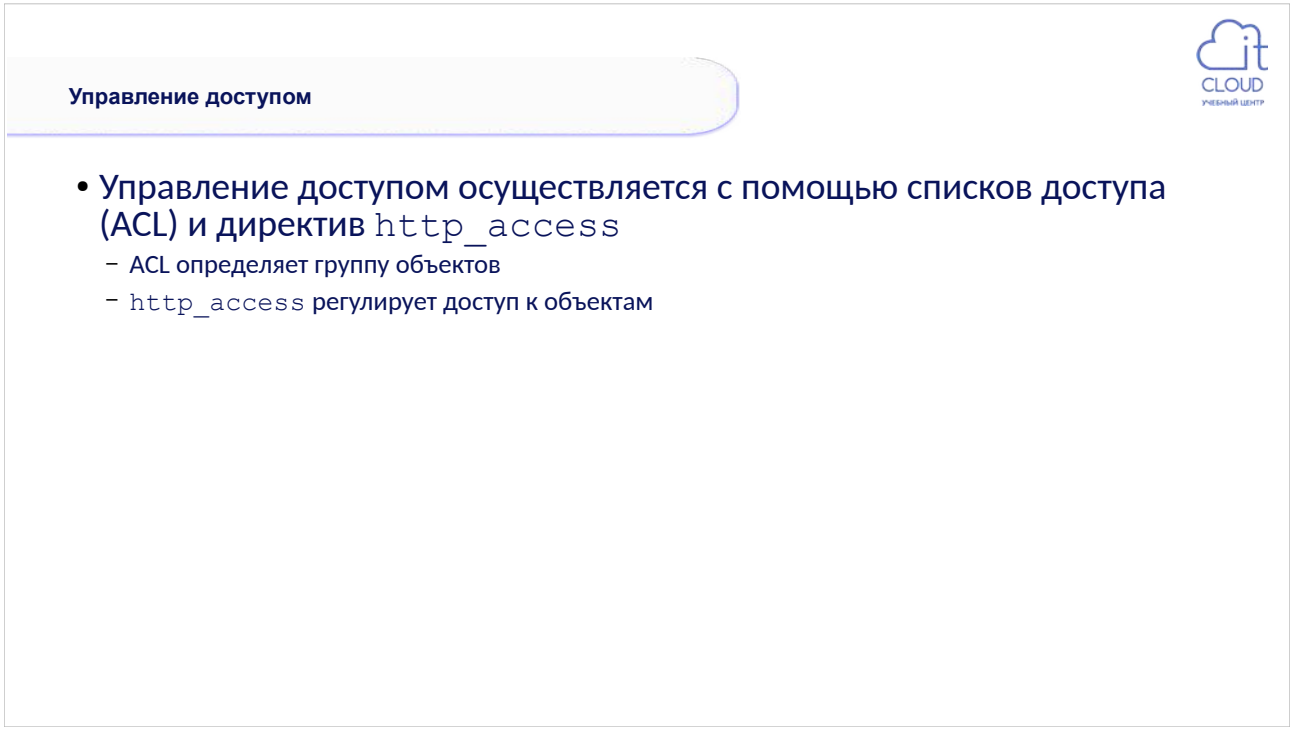
Проверить работоспособность Squid можно с помощью программы `squidclient`.

```
$ /usr/sbin/squidclient -l localhost http://www.squid-cache.org
```

В результате выполнения этой команды в стандартный поток вывода будет передано содержимое стартовой WEB страницы запрошенного URL. После опции `-l` указан адрес тестируемого сервера Squid.

При необходимости использования альтернативного имени узла, на котором запущен Squid (например, для изменения реального имени узла в сообщениях об ошибках), его необходимо указать в файле конфигурации с помощью директивы `visible_hostname`.

```
visible_hostname myalias.domain.ru
```



Элементы списков доступа (ACL elements) фактически определяют принадлежность объектов, с которыми работает Squid, к тому или иному множеству. То есть, они выполняют функцию группировки объектов.

Директивы `http_access` устанавливают разрешения и запреты на обслуживание сервером Squid для клиентов, определенных с помощью элементов ACL. Эти директивы, собственно, и устанавливают списки доступа (ACL).

Существуют иные, чем `http_access` виды ACL. Например, директива `always_direct` определяет ACL запросов, направляемых напрямую серверу.

Проверка прав обслуживания сервером осуществляется последовательным перебором директив `http_access`. Как только в одной из таких директив в качестве аргумента будет установлено имя ACL, в который входит проверяемый объект (входит в смысле критерия определения ACL), никакие дальнейшие проверки не проводятся и доступ либо разрешается, либо запрещается.

Пример:

```
acl comp1 src 192.168.111.1/255.255.255.255
acl comp2 src 192.168.111.2/32
http_access allow comp1
http_access deny !comp2
```

***Примечание**: В этом примере определены два ACL по критерию IP адресов источников запросов (клиентов). В ACL comp1 входит единственный компьютер с адресом 192.168.111.1. Маска 255.255.255.255 указывает, что это - единственный узел, а не сеть. ACL comp2 определен аналогично с помощью нотации CIDR. Первая директива http_access разрешает доступ узлу comp1, а вторая запрещает доступ со всех узлов, кроме comp2. Если в этом примере директивы http_access поменять местами, то узел comp1 доступа не получит.*

Если при определении ACL указан целый ряд значений (элементов), то при проверке доступа к этим значениям будет применена логическая операция ИЛИ.

Пример: список доступа two_comps определен указанием двух адресов клиентов. Только этим двум узлам разрешен доступ (ИЛИ первому, ИЛИ второму).

```
acl two_comps src 192.168.111.1/32 192.168.111.2/32
http_access allow two_comps
http_access deny all
```

Напротив, при указании в качестве аргументов директивы `http_access` нескольких имен ACL, будет использована операция И. То есть, проверяемый объект должен одновременно удовлетворять обоим критериям.

Пример:

```
acl two_comps src 192.168.111.1/32 192.168.111.2/32
acl ext_net dst 212.193.90.0/24
http_access allow two_comps ext_net
http_access deny all
```

***Примечание**: В этом примере список доступа two_comps определен указанием двух адресов клиентов, как и в предыдущем примере. Однако они получат доступ только тогда, когда будут обращаться по адресам сети 212.193.90.0/224, определяющей ACL ext_net. И первое И второе условие должны быть выполнены.*

Если несколько элементов ACL с одним именем описаны в разных строках конфигурации, то Squid преобразует эти директивы `acl` в одну строку, устанавливая их значения через пробел. То есть используется логика ИЛИ.

```
acl comps src 192.168.111.1/32
acl comps src 192.168.111.2/32
```

Эквивалентно

```
acl comps src 192.168.111.1/32 192.168.111.2/32
```

При определении ACL бывает удобно описывать их элементы в отдельных файлах, но не в squid.conf. Это достигается с помощью указания в директиве `acl` вместо списка элементов имени файла, взятого в кавычки.

```
acl trasted_nets src "/etc/squid/acl/trasted_nets.src"

# cat /etc/squid/acl/trasted_nets.src
192.168.111.0/24
192.168.112.0/24
192.168.113.0/24
```

Имеется возможность создания HTML страниц с сообщением об ошибке доступа для объектов, которым запрещен доступ с помощью http_access deny. Эти страницы можно

отображать в случае попытки доступа объектов из заданных ACL. Указать страницу сообщения для ACL можно директивой deny_info.

Пример:

```
http_access deny black_list
http_access deny ohrannyk_list
deny_info ERR_CUSTOM_ACCESS_DENIED black_list
deny_info http://www.zyx.edu/nepushu.html ohrannyk_list
```

***Примечание**: В этом примере для ACL black_list и ohrannyk_list доступ запрещен. При попытке доступа элемента ACL black_list будет выведена HTML страница ERR_CUSTOM_ACCESS_DENIED, а для ohrannyk_list страница сообщения будет взята по адресу http://www.zyx.edu/nepushu.html ohrannyk_list.*

Squid предоставляет заранее созданные страницы с сообщением об ошибках доступа. По умолчанию они находятся в каталоге `/etc/squid/errors`. Указать другой каталог можно с помощью директивы error_directory.

Пример:

```
error_directory /etc/squid/err.ru
```

Для управления доступом на основе адресов и имен компьютеров применяются следующие типы ACL:

Тип	Критерий
`src`	По принадлежности клиента к определенной сети или блоку адресов.
`dst`	Принадлежность адреса назначения к заданной сети или блоку адресов.
`srcdomain`	Проверка доменного имени клиента с помощью обратного преобразования IP адреса.
`dstdomain`	Имя сервера назначения из URL.
`srcdom_regex`	Соответствие доменного имени клиента регулярному выражению.
`dstdom_regex`	Соответствие доменного имени сервера назначения регулярному выражению.
`url_regex`	Соответствие URL регулярному выражению.

В ACL, основанных на регулярных выражениях, можно применять опцию `-i`, отключающую чувствительность к регистру.

Пример: к ACL web_mail будут подходить любые доменные имена серверов назначения, содержащие строку mail без учета регистра.

```
acl web_mail dstdom_regex -i mail
```

Ограничить доступ к WEB ресурсам определенными днями недели и часами суток можно с помощью ACL типа `time`.

Пример: При такой конфигурации доступ будет возможен в рабочие дни с 9 до 18 (кроме пятницы - до 17). Для двух адресов разрешен доступ в любое время, кроме выходных дней.

```
acl mynet src 192.168.111.1-2/255.255.255.255

acl w_days time M 09:00-18:00 # Понедельник
acl w_days time T 09:00-18:00 # Вторник
acl w_days time W 09:00-18:00 # Среда
acl w_days time H 09:00-18:00 # Четверг
acl w_days time F 09:00-17:00 # Пятница
acl h_days time A S           # Суббота и Воскресенье

http_access allow w_days
http_access allow my_net !h_days
http_access deny all
```

Для ограничения максимального количества подключений одного и того же клиента предназначен ACL типа `maxconn`.

Пример:

```
acl lammer maxconn 2
```

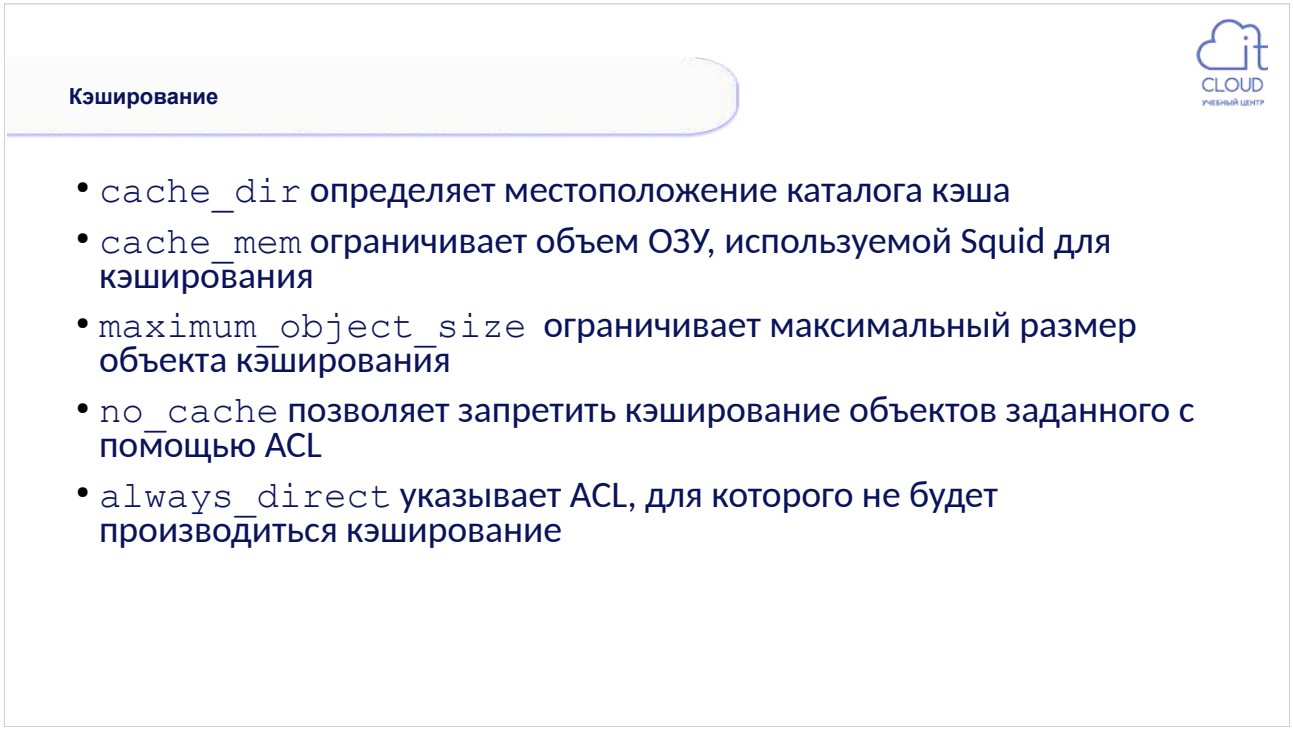
Если необходимо управлять выгрузкой (то есть передачей от клиента) заданных типов mime данных, то используется ACL типа `req_mime_type`.

Напротив, при необходимости ограничения загрузки заданных mime типов данных следует использовать ACL типа `rep_mime_type`.

Если Squid был скомпилирован с опцией `--enable-arp-acl`, то это позволит ограничивать доступ на основе MAC (Ethernet) адресов.

Пример: Такая настройка запретит доступ для компьютера с заданным MAC адресом сетевого адаптера.

```
acl bubuka arp 0F:12:A3:57:01:0E
http_access deny bubuka
```



Клиенты запрашивают у Squid некоторый объект, например, WEB страницу, по заданному URL. Если этот объект находится в кэше, то он передается клиенту. В противном случае, он предварительно извлекается с сервера назначения, а затем передается клиенту.

Директива `cache_dir` определяет местоположение каталога кэша, а также ограничивает максимальный размер его содержимого. Кроме того она определяет формата хранения данных в кэше.

По умолчанию используется формат ufs. Синтаксис директивы `cache_dir` для этого формата:

```
cache_dir ufs каталог объем уровень1 уровень2
```

Где:

каталог - имя каталога кэша;

объем - ограничение объема кэша в Мб;

уровень1 - количество подкаталогов первого уровня в кэше;

уровень2 - количество подкаталогов второго уровня в кэше (это подкаталоги каталогов первого уровня).

Пример: размер кэша ограничен 1 Гб. При этом количество подкаталогов первого уровня будет 32, и в каждом из них будет создано 512 подкаталогов второго уровня.

```
cache_dir ufs /var/spool/squid 1000 32 512
```

Для ограничения объема ОЗУ, используемой Squid для кэширования, предназначена директива `cache_mem`.

Она не ограничивает объем памяти, используемый процессом Squid. Эта настройка ограничивает лишь объем памяти для кэширования.

При возникновении необходимости ограничить максимальный размер объектов, помещаемых в кэш, следует использовать настройку `maximum_object_size` (по умолчанию - 4 Мб).

Пример:

```
maximum_object_size 8192 KB
```

Директива `no_cache` позволяет запретить кэширование объектов заданного с помощью ACL.

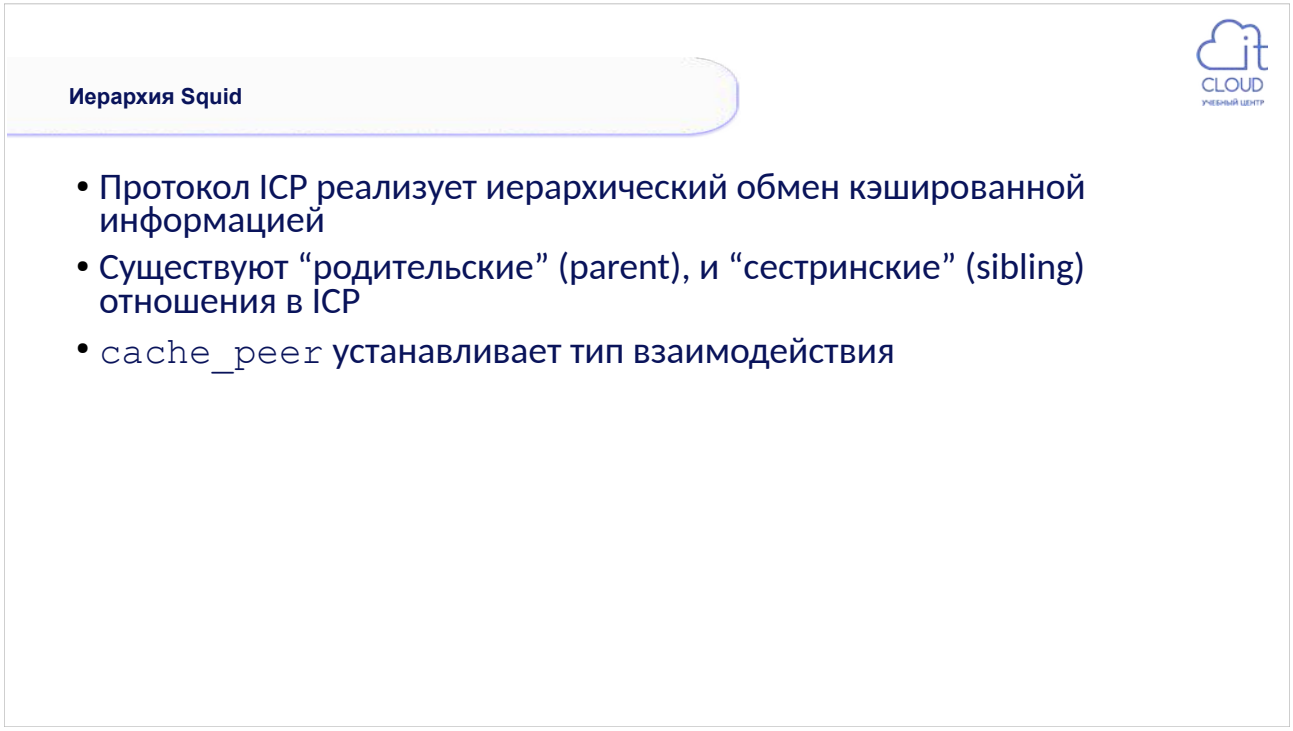
Пример: Рекомендуется использовать настройку, которая запрещает кэширование динамически генерируемых страниц.

```
acl QUERY urlpath_regex cgi-bin \?
no_cache deny QUERY
```

Можно определить ACL, кэширование для которых производится не будет.

Пример: для прямого доступа к заданному URL без кэширования можно воспользоваться конструкцией:

```
acl dir_dom dstdomain bamboo.net
always_direct allow dir_dom
```



Серверы Squid способны к взаимодействию, направленному на обмен кэшированной информацией по специальному протоколу ICP (порт 3130 UDP).

Взаимодействие Squid серверов выстраивается на иерархической основе. По критерию уровней, на которых находятся взаимодействующие серверы, различают связи между серверами, находящимися на разных уровнях иерархии, то есть, “родительские” (parent), и “сестринские” (sibling), которые устанавливаются между серверами одного уровня.

Существует также особый вид взаимодействия серверов Squid, использующих групповое вещания - multicast.

Определить тип взаимодействия серверов Squid можно с помощью директивы cache_peer.

```
cache_peer узел тип http_порт icp_порт опции
```

Где узел указывает партнерский сервер Squid, тип - уровень взаимодействия (`parent`, `sibling` или `multicast`), - `http_порт` порт для обмена данными, `icp_порт` - порт для запросов кэшированных объектов от партнерского сервера.

Можно отключить или изменить номер порта UDP, используемый для ICP запросов. Это можно сделать, использую директиву `icp_port`.

Пример: запрет использования ICP на сервере.

```
icp_port 0
```

Пример: Для установления отношений между серверами Squid, принадлежащими одному уровню иерархии, можно использовать настройку:

```
cache_peer peer1.example.ru sibling 3128 3130
```

Пример: Связать дочерний сервер с родительским можно следующим образом:

```
cache_peer parent.example.ru parent 3128 3130
```

Часто бывает необходимо установить связь между Squid, находящимся в защищенной файерволлом внутренней сети, и внешним сервером Squid. При этом из внутренней сети во внешнюю разрешен лишь трафик по порту 3128 TCP. В таком случае подходит настройка:

```
cache_peer external.examle.ru parent 3128 0 no-query default
```

Внутренний сервер, на котором находятся такие настройки, будет работать, фактически, как клиент внешнего external.examle.ru. Обмена по ICP не будет.

Можно разделить целевые домены по разным партнерским серверам. Для этого используется директива `cache_host_domain`.

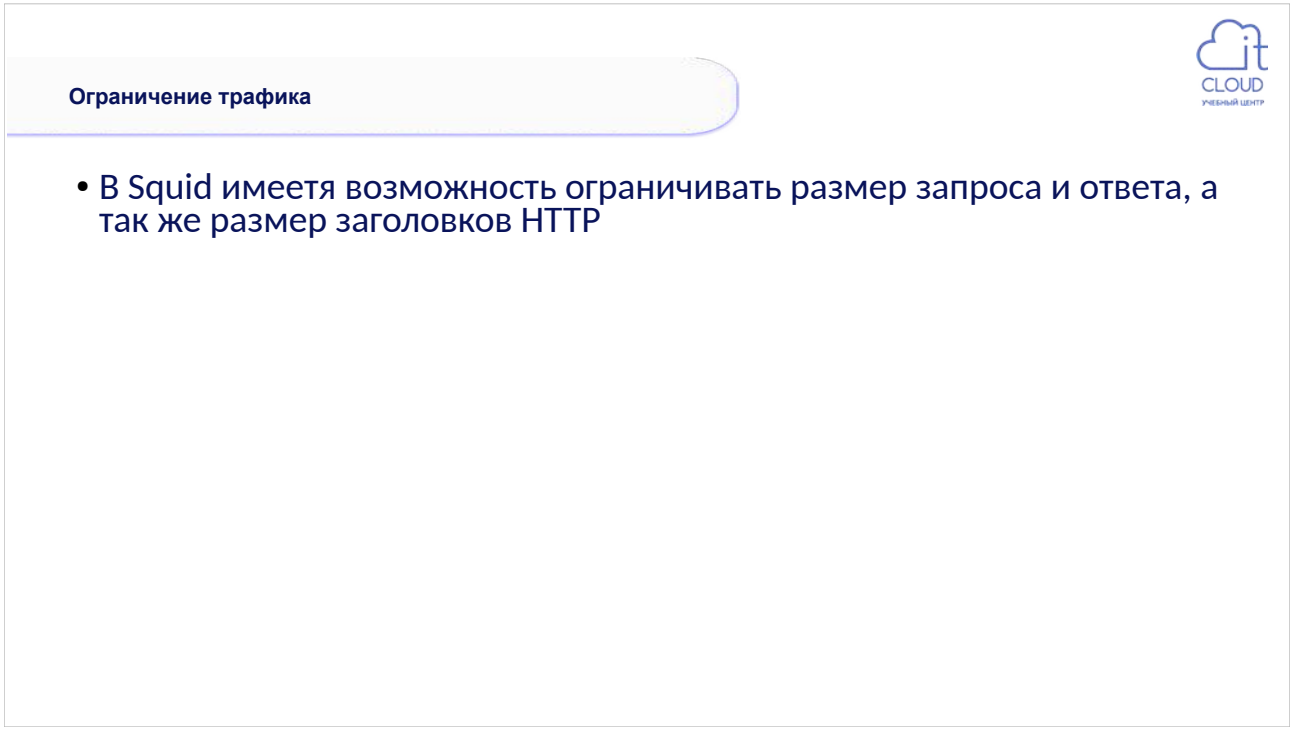
Пример: все запросы клиентов, направленные в поддомены .ru, будут порождать обмен сервера по ICP с родительским сервером parent.example.ru, а для запросов, направленных в .ch сервер будет взаимодействовать с родительским сервером parent.example.ch.

```
cache_peer parent.example.ru parent 3128 3130
cache_peer parent.example.ch parent 3128 3130
cache_peer_domain parent.example.ru .ru
cache_peer_domain parent.example.ch .ch
```

Имеется возможность изменять тип взаимодействия серверов Squid в зависимости от целевого домена клиентского запроса. Это обеспечивает директива `neighbor_type_domain`.

Пример: сервер parent.example.su используется, как родительский, за исключением случаев, когда запросы клиентов направлены в поддомены домена .su

```
cache_peer parent parent.example.su 3128 3130
neighbor_type_domain parent.example.su sibling .su
```



Ограничение размера клиентского запроса (то есть информации, передаваемой клиентом серверу с помощью запросов PUT/POST), достигается использованием директивы `request_body_max_size`.

Пример: максимальный размер клиентского запроса будет ограничен 10 Кб.

```
request_body_max_size 10 KB
```

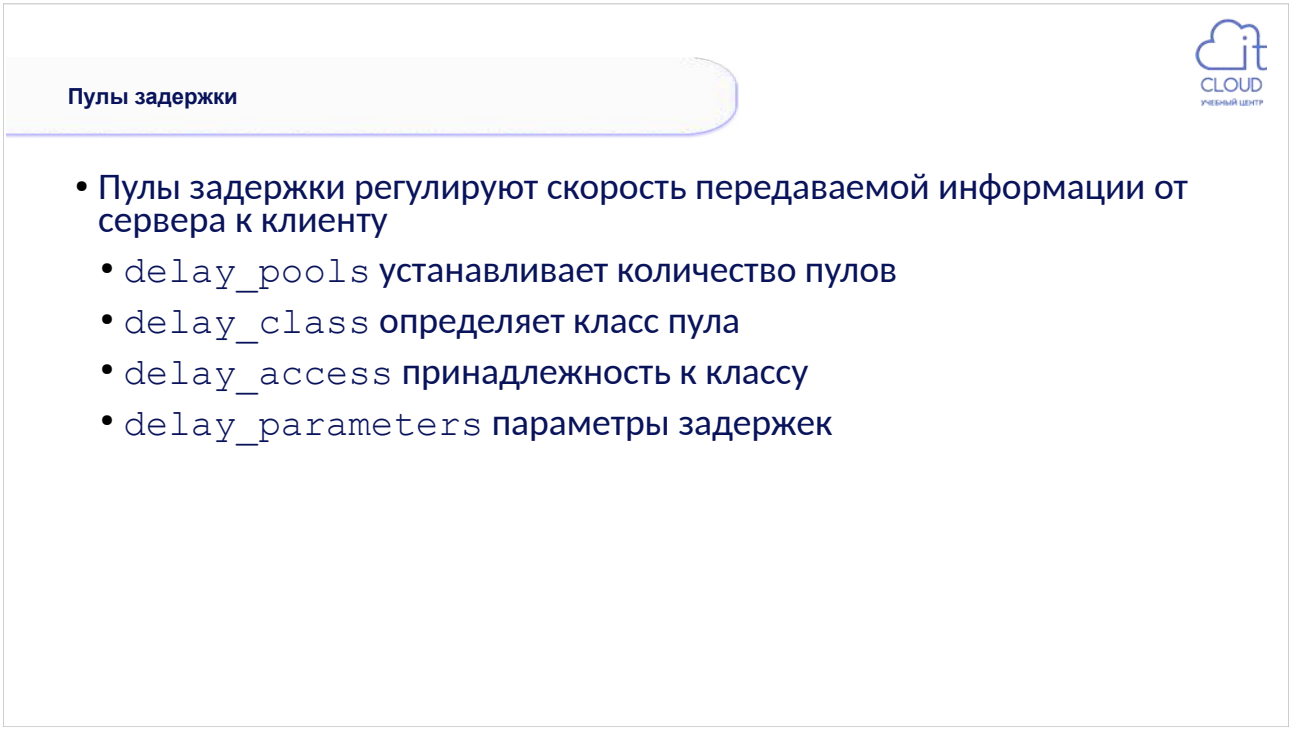
Для предотвращения получения клиентом слишком больших файлов используется директива `reply_body_max_size`. С ее помощью можно ограничить максимальный размер получаемой информации для заданных ACL.

Пример: максимальный размер получаемой информации ограничен 1 Мб для всех клиентов.

```
reply_body_max_size 1048576 allow all
```

Если в качестве ограничения в директиве `reply_body_max_size` указан 0, то это обозначает отсутствие ограничения.

Некоторые виды атак, связанные с особенностями протокола HTTP, относящиеся к DoS атакам, либо к атакам, направленным на переполнение буферов, используют длинные заголовки HTTP. Обычные заголовки HTTP не превышают по размеру 512 байт. Для ограничения максимального размера заголовков HTTP применяется настройка `request_header_max_size`.



Squid предоставляет возможность регулировать “полосу пропускания” канала для разных клиентов. Это достигается с помощью использования пулов задержки, в которые помещаются передаваемые клиенту пакеты. Использование пулов задержки с точки зрения клиента будет обозначать ограничение скорости передачи информации.

Squid должен быть скомпилирован с опцией `DELAY_POOLS` при необходимости использования пулов задержки. Эта опция компиляции устанавливается с помощью опции `--enable-delay-pools` команды `./configure`.

Может быть использовано сразу несколько различных пулов задержки, например, для разных категорий клиентов.

Количество пулов задержки устанавливается с помощью директивы `delay_pools`.

Пример:

```
delay_pools 2
```

Пулы задержки идентифицируются по номерам, начиная с единицы.

Класс пула определяется директивой `delay_class`. Различаются три класса пулов задержки:

Класс	Клиенты
1	Позволяет ограничивать пропускную способность всех пакетов, относящихся к клиентам, заданным с помощью некоторого ACL. В таком пуле все пакеты находятся в единственной очереди задержки. Squid позволяет определять несколько пулов задержки первого класса.
2	В пуле второго класса имеется общая очередь задержки и 255 индивидуальных

	очередей. Индивидуальные очереди создаются Squid автоматически для каждого адреса из соответствующей пулу сети класса C. Это позволяет определять индивидуальные задержки для узлов сети класса C.
3	В пуле второго класса имеется общая очередь задержки для сети класса B, 255 индивидуальных очередей для подсетей сети класса B и индивидуальные очереди для узлов сети.

ACL, члены которого будут пользоваться пулом задержки, указывается с помощью директивы `delay_access`.

Пропускная способность канала, определяемая пулом задержки, задается с помощью директивы `delay_parameters`. Для различных классов пулов задержки у этой директивы разное количество параметров, однако первый – всегда номер пула.

При создании пула задержки первого класса в качестве аргументов `delay_parameters` должны быть указаны два аргумента: номер пула и параметры задержки.

Пример:

```
acl all src 0.0.0.0/0.0.0.0
delay_pools 1
delay_class 1 1
delay_access 1 allow all
delay_parameters 1 8000/16000
```

***Примечание**: Здесь определен единый пул задержки, имеющий номер 1. Директива delay_class относит этот пул (номер пула – первый аргумент), к первому классу (класс – второй аргумент). Директива delay_access устанавливает, что субъекты ACL (здесь ACL all – все клиенты) попадают в сферу действия пула задержки номер 1 (первый параметр delay_access). Последняя настройка – delay_parameters определяет параметры задержки для пула номер 1. Параметры задержки 8000/16000 – устанавливают пропускную способность пула в байтах (см. ниже).*

Пропускную способность канала в директиве `delay_parameters` указывают с помощью двух значений через косую черту – `restore/maximum`. Параметр `restore` определяет количество байт в секунду, помещаемое в очередь задержки. Параметр `maximum` – максимальное количество байт в очереди.

Пул задержки второго класса требует указать в качестве аргументов `delay_parameters` три аргумента: номер пула, параметры задержки общей очереди задержки (aggregate) и индивидуальные параметры задержки.

Пример:

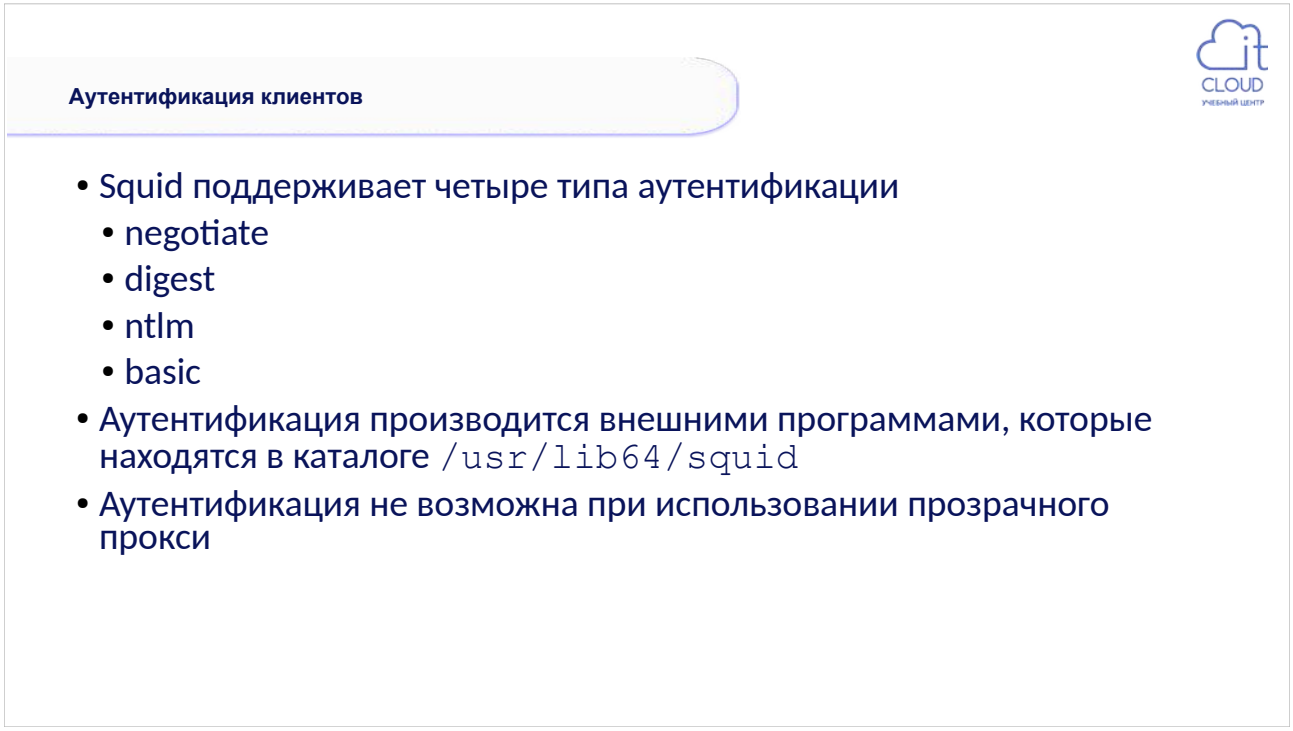
```
acl mynet src 192.168.111.0/24
delay_pools 1
delay_class 1 2
delay_access 1 allow mynet
delay_access 1 deny all
delay_parameters 1 -1/-1 8000/16000
```

***Примечание**: Пул задержки второго класса здесь создан для клиентов из сети класса C. Директива delay_parameters здесь имеет три аргумента: номер пула - 1, параметры задержки для всех клиентов - -1/-1 (отсутствие ограничений) и индивидуальные параметры задержки для клиентов – 8000/16000 (8000 – количество байт в секунду, разрешенное для постановки в очередь, 16000 – общее количество байт в очереди).*

Параметры задержки для пулов третьего класса указывают с помощью четырех параметров:

```
delay_parameters pool aggregate network individual
```

где *pool* – номер пула, *aggregate* – общие параметры задержки, *network* – параметры задержки для подсетей, *individual* – индивидуальные параметры.



Squid поддерживает несколько типов аутентификации, которые определяются настройкой `auth_param`.

- negotiate — прозрачная аутентификация Kerberos в домене MS Windows;
- digest — аутентификация, использующая метод challenge/response;
- ntlm — использование протокола NTLM для аутентификации в домене MS Windows;
- basic — простая имя и пароль аутентификация.

Аутентификация в Squid осуществляется с помощью вспомогательных программ, обычно размещаемой в GNU/Linux в каталоге `/usr/lib64/squid`.

Пример: настройка basic аутентификации в файле с паролями.

1. Создаем файл паролей пользователей с помощью программы `htpasswd`.

```
htpasswd -c /etc/squid/squid.passwd squiduser
```

В этом примере создается новый файл паролей `/etc/squid/squid.conf`. При этом в файл заносится пароль пользователя `squiduser`. Файл будет создан, так как здесь установлена опция `-c` команды `htpasswd`. Если файл паролей уже создан, использование этой опции сотрет его содержимое.

2. Для определения группы клиентов, для которых требуется аутентификация, используется специальный тип ACL – `proxy_auth`. В качестве аргументов этого ACL задают список имен пользователей, либо параметр REQUIRED, обозначающий любые имена пользователей, имеющих право доступа.

3. Программе `basic_ncsa_auth` необходимо передать параметр, указывающий местонахождение файла паролей и указать тип аутентификации.

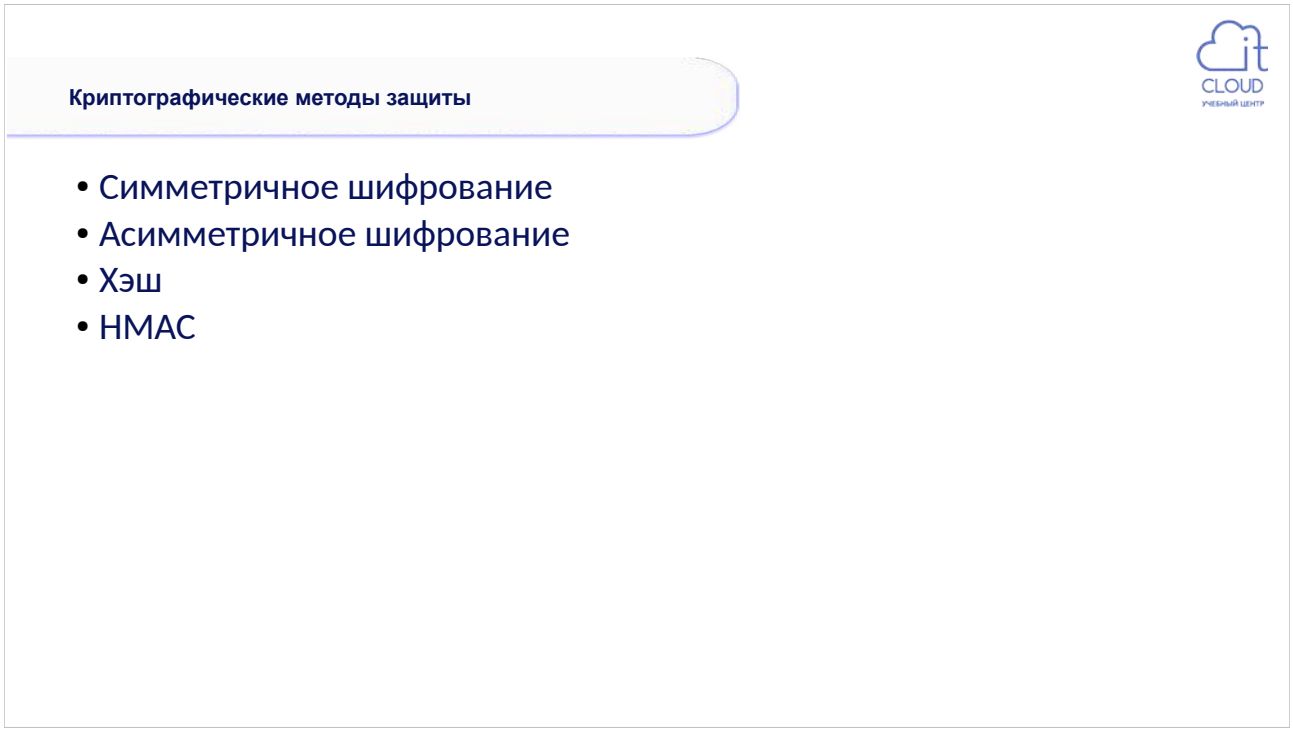
```
auth_param basic program /usr/lib64/squid/basic_ncsa_auth /etc/squid/squid.passwd
```

4. Определенный выше с помощью ACL список доступа должен иметь разрешение на доступ:

```
acl password proxy_auth REQUIRED
auth_param basic program /usr/lib64/squid/basic_ncsa_auth /etc/squid/squid.passwd
http_access allow password
```

В этом примере указан список доступа `password`. Он формируется из содержимого файла `/etc/squid/squid.passwd`.

8.4 Краткое введение в криптографические механизмы защиты.

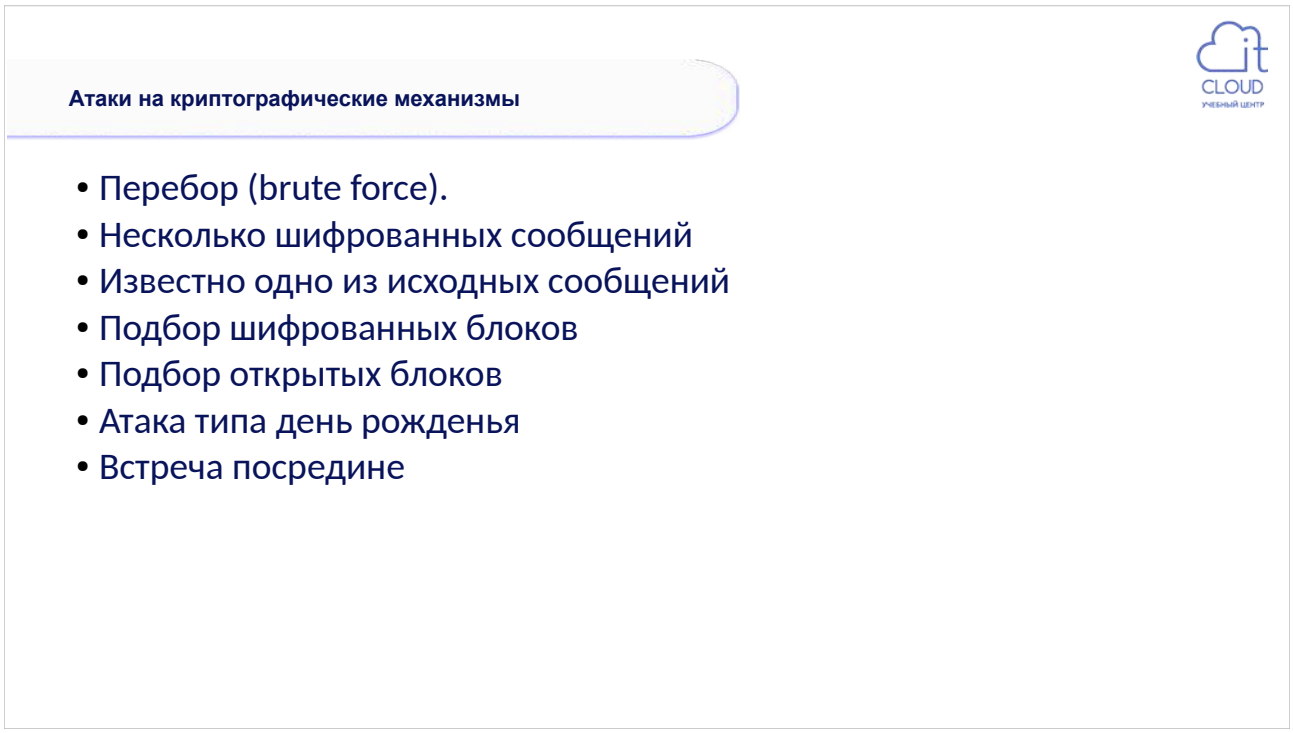


Для защиты передаваемых по сети данных применяются несколько способов защиты:

- Симметричное шифрование. Для шифрования и расшифровки данных применяется один и тот же ключ. При использовании этого метода необходимо обеспечить защиту ключа шифрования. Применяется для массового шифрования. Примеры: DES, 3DES, AES.
- Асимметричное шифрование. Для шифрования данных и их расшифровки применяются разные ключи. Ключи создаются парой один из ключей называется публичным, а второй частным. Зашифрованное публичным ключом может расшифровано только соответствующим ему частным ключом, и наоборот зашифрованное частным ключом может быть расшифровано только соответствующим ему публичным. Публичный ключ может быть передан другим лицам для того, чтобы получить от них зашифрованные данные. Шифрованные данные частным ключом могут быть использованы для подтверждения получения данных от держателя частного ключа. Поскольку асимметричные алгоритмы существенно сложней симметричных, этот метод не используется для шифрования большого количества данных. Примеры: RSA. Elgamal, ECDH.
- Хэш. Алгоритм в котором данные любого размера превращаются в строку фиксированной длины. Это однонаправленный процесс, в котором после хэширования не возможно восстановить исходные данные. Используется для подтверждения неизменности передаваемых данных. Примеры: MD5, SHA1, SHA2

- HMAC. Алгоритмы хэширования, которые используют секретный ключ для аутентификации передаваемых данных. Применяется двойное хэширование данных с секретным ключом. Примеры: HMAC-MD5, HMAC-SHA2.

8.4.1 Атаки на криптографические механизмы.



Существую несколько типов атак на криптографические алгоритмы.

- Перебор (brute force). Атакующий перебирает все возможные ключи для дешифровки данных. Все криптоалгоритмы подвержены этой атаке.
- Несколько шифрованных сообщений. Атакующий сравнивает шифрованные сообщения для предсказания ключа шифрования. Поскольку современные алгоритмы выдают псевдослучайные данные, то данный метод не очень практичен.
- Известно одно из исходных сообщений. Атакующий имея одно из известных ему сообщений и знания о алгоритмах генерации ключей пытается подобрать ключи шифрования, чтобы дешифровать остальные сообщения.
- Подбор шифрованных блоков. Атакующий разбирает шифрованные данные на блоки и пытается определить способ формирования ключей.
- Подбор открытых блоков. Атакующий подбирает открытые блоки и ключи и сравнивает их с шифрованным сообщением, пытаясь предсказать ключи.
- Атака типа день рождения. В группе из 23 человек вероятность дня рождения в одну дату более 50%. Статистический анализ ключей шифрования пытается предсказать возможные ключи шифрования.
- Встреча посредине. Атакующий знает часть шифрованного и открытого сообщения. Открытая часть шифруется всеми возможными ключами, а затем расшифровывается закрытая часть пока не будет найден ключ.

Помимо применения криптографических механизмов необходимо еще обеспечить аутентификацию участвующих в процессе партнеров. Непосредственно методами криптографии эта проблема не решается.

Пример: атаки посередине.

Предположим, что Алиса хочет передать Бобу некоторую информацию. Мэлори хочет перехватить сообщение и, возможно, изменить его так, что Боб получит неверную информацию.

Мэлори начинает свою атаку с того, что устанавливает соединение с Бобом и Алисой, при этом они не могут догадаться о том, что кто-то третий присутствует в их канале связи. Все сообщения, которые посылают Боб и Алиса, проходят Мэлори.

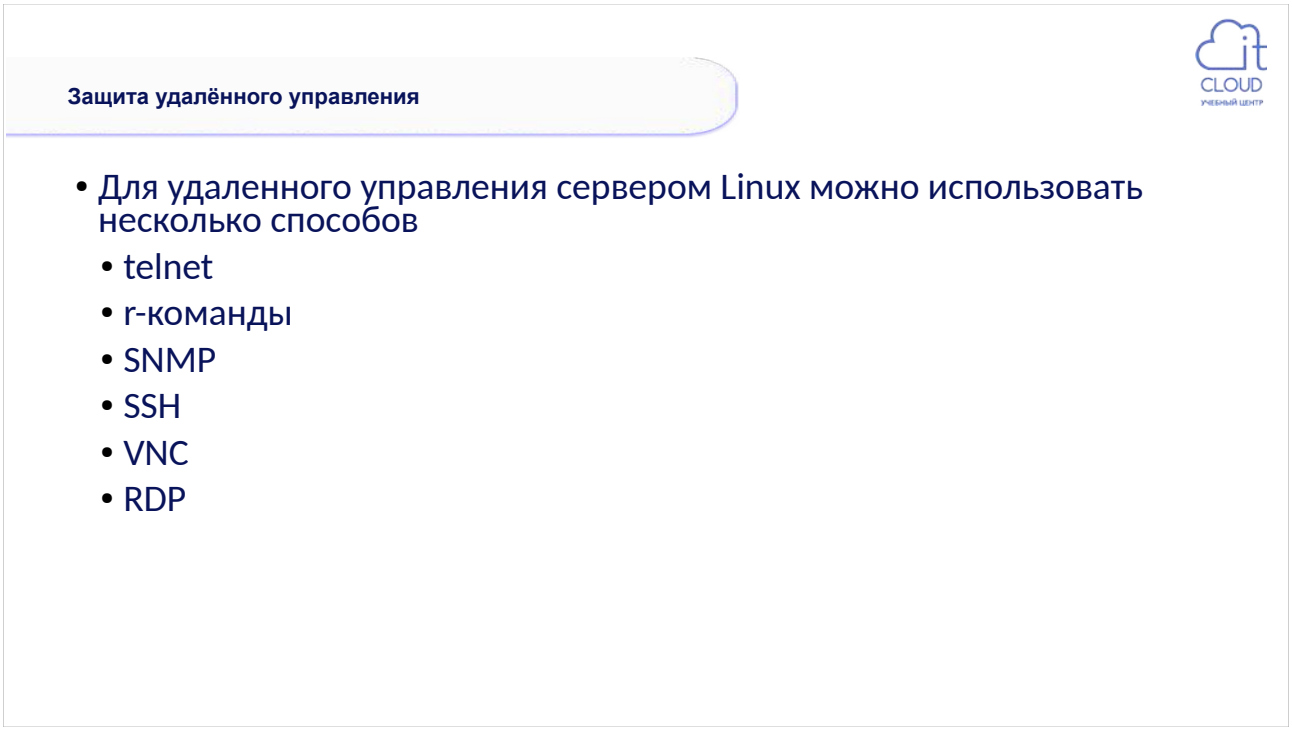
Алиса просит у Боба его открытый ключ. Мэлори представляется Алисе Бобом и отправляет ей свой открытый ключ. Алиса, считая, что это ключ Боба, шифрует им сообщение и отправляет его Бобу. Мэлори получает сообщение, расшифровывает, затем изменяет его, если нужно, шифрует его открытым ключом Боба и отправляет его ему. Боб получает сообщение и думает, что оно пришло от Алисы:

1. **Алиса** отправляет Бобу сообщение, которое перехватывает Мэлори:
   Алиса «Привет, Боб, это Алиса. Пришли мне свой открытый ключ.» → Мэлори Боб
2. **Мэлори** пересылает сообщение Бобу; Боб не может догадаться, что это сообщение не от Алисы:
   Алиса Мэлори «Привет, Боб, это Алиса. Пришли мне свой открытый ключ.» → Боб
3. **Боб** посылает свой ключ:
   Алиса Мэлори ← [ключ Боба] Боб

4. **Мэлори** подменяет ключ Боба своим и пересылает сообщение Алисе:
   Алиса ← [ключ Мэлори] Мэлори Боб
5. **Алиса** шифрует сообщение ключом Мэлори, считая, что это ключ Боба, и только он может расшифровать его:
   Алиса *«Встречаемся на автобусной остановке!»* [зашифровано ключом Мэлори] → Мэлори Боб
6. **Мэлори** расшифровывает сообщение, читает его, модифицирует его, шифрует ключом Боба и отправляет его:
   Алиса Мэлори *«Жди меня у входа в музей в 18:00.»* [зашифровано ключом Боба] → Боб
7. **Боб** считает, что это сообщение Алисы.

Этот пример демонстрирует необходимость использования методов для подтверждения того, что обе стороны используют правильные открытые ключи, то есть что у стороны А открытый ключ стороны Б, а у стороны Б — открытый ключ стороны А. В противном случае, канал может быть подвержен атаке «человек посередине».

8.5 Защита удалённого управления. Протокол SSH.

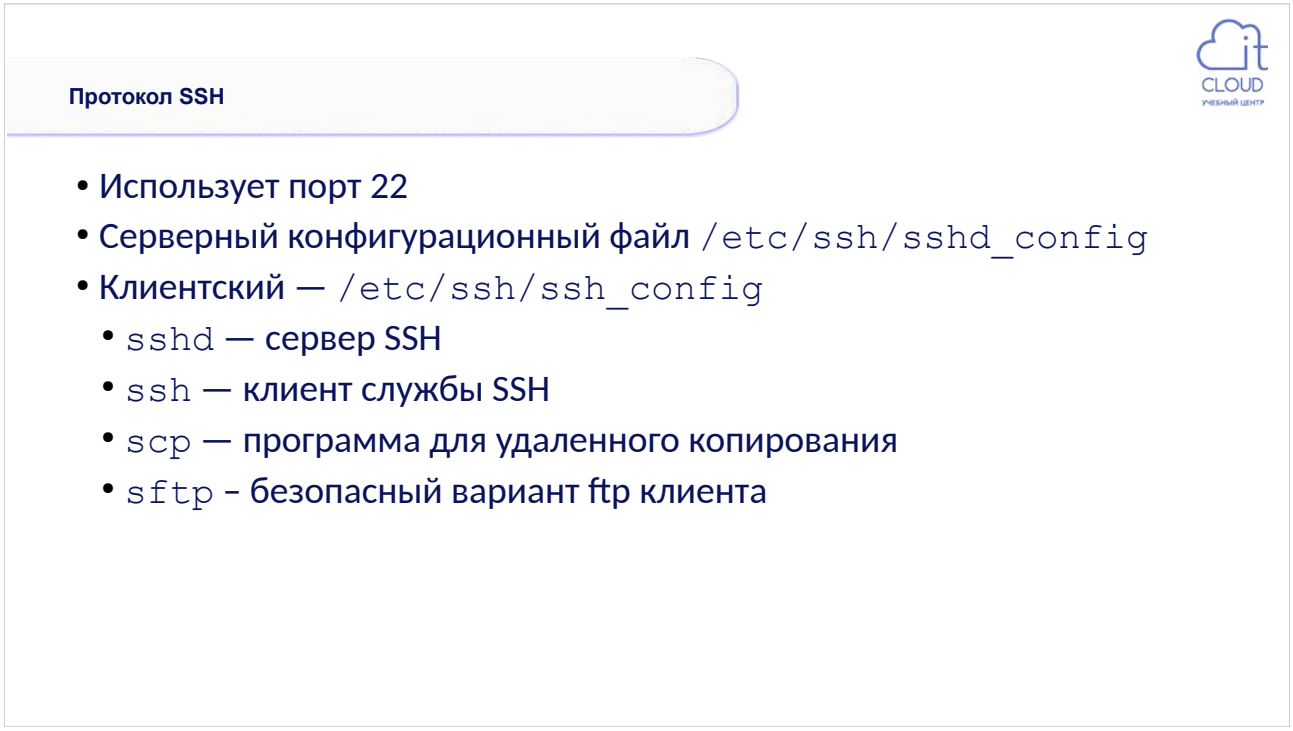


Для удаленного управления Unix подобными системами существует множество протоколов. Часть этих протоколов работают открытым текстом, например: telnet, r-команды или SNMP.

Система SSH (Secure Shell) предоставляет защищенную криптографически надежную альтернативу r-командам и службе telnet для удаленного управления и туннелирования TCP-соединений (например, для передачи файлов).

SSH допускает выбор различных алгоритмов шифрования.

В процессе взаимодействия открывается шифрованный канал связи между удаленными узлами. Также предоставляется возможность аутентификации с использованием публичных и частных ключей (несимметричное шифрование).



В GNU/Linux часто используется версия системы SSH - OpenSSH, распространяемая на свободной основе. Реже может быть использован drobear, в основном на малых системах типа домашнего маршрутизатора.

Пакет OpenSSH представлен четырьмя основными программами:

1. `sshd` - сервер SSH, прослушивающий 22 порт TCP.
2. `ssh` - клиент службы SSH, позволяющий инициировать удаленный сеанс.
3. `scp` – клиентская программа для удаленного копирования.
4. `sftp` – безопасный вариант ftp клиента

Серверная и клиентская части системы OpenSSH обладают разными файлами конфигурации:

- сервер `sshd` имеет конфигурационный файл `/etc/ssh/sshd_config`.
- клиенты `ssh` и `scp` - `/etc/ssh/ssh_config`.

Сервер OpenSSH запускается самостоятельно (stand-alone) с помощью демона systemd, поэтому для автоматического старта сервера OpenSSH при переходе в многопользовательский режим следует надлежащим образом настроить систему инициализации.

Пример:

```
root@s10:~# systemctl enable sshd
Created symlink from /etc/systemd/system/multi-user.target.wants/sshd.service to
/usr/lib/systemd/system/sshd.service.
root@s10:~# systemctl start sshd
```

Для инициирования сеанса на удаленной машине с запущенным сервером OpenSSH достаточно на клиентском узле просто выполнить команду `ssh`, указав ей в качестве аргумента имя или IP адрес узла назначения.

Пример:

```
[vlesk@vlesk-nb ~]$ ssh root@10.255.255.100
The authenticity of host '10.255.255.100 (10.255.255.100)' can't be established.
ECDSA key fingerprint is SHA256:2aIobNR2QmFy3PHcFC9F2nEa7nMe9KhqYTsCxGAIDy8.
ECDSA key fingerprint is MD5:42:25:61:78:71:ff:2e:93:1b:68:20:bb:9f:86:18:17.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.255.255.100' (ECDSA) to the list of known hosts.
root@10.255.255.100's password:
Last login: Sat Aug 26 19:15:19 2017
root@s10:~#
```

Одна из наиболее часто используемых опций команды `ssh` является `-l`, с помощью которой можно указать имя пользователя для входа в удаленный сеанс.

Пример:

```
[vlesk@vlesk-nb ~]$ ssh -l root 10.255.255.100
root@10.255.255.100's password:
Last login: Sat Aug 26 19:24:06 2017 from gateway
root@s10:~#
```

Команда `scp` позволяет копировать файлы на удаленную машину и с нее.

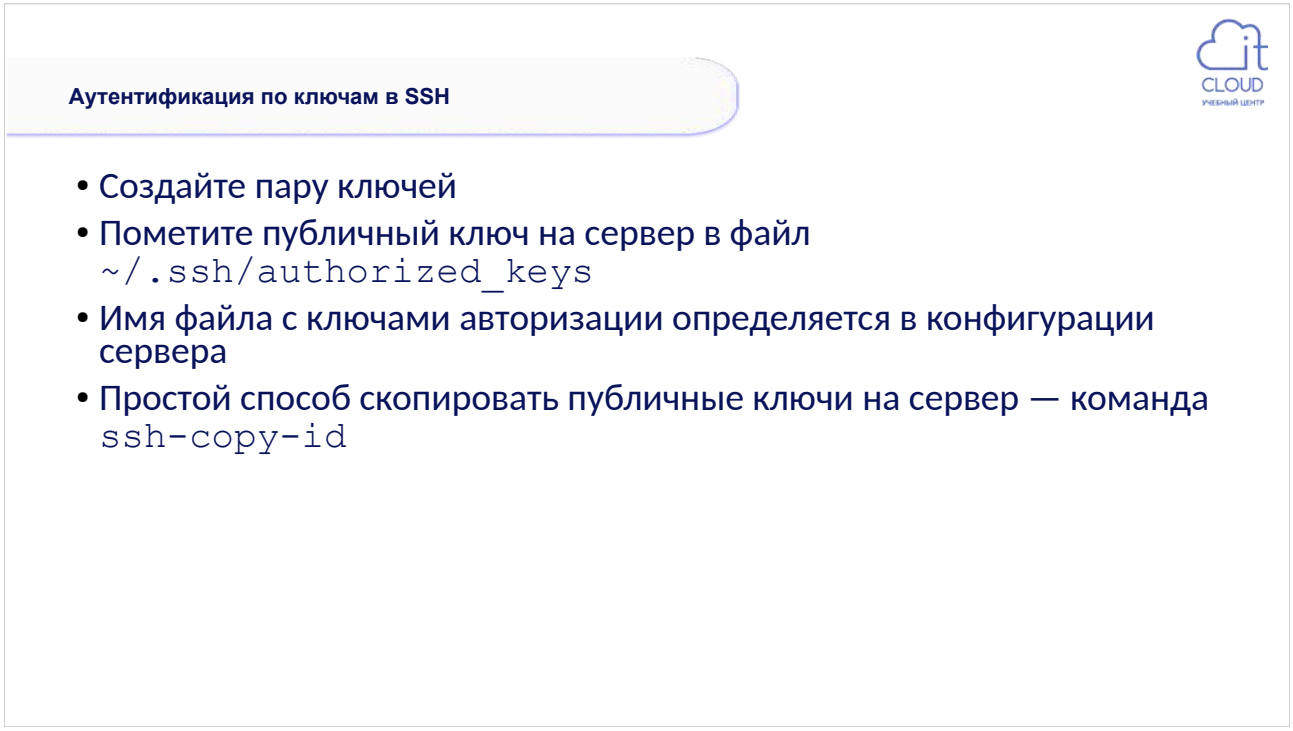
Пример:

```
[vlesk@vlesk-nb ~]$ scp root@10.255.255.100:/etc/hosts hosts_from_s10
root@10.255.255.100's password:
hosts                                         100%  158   128.7KB/s   00:00
[vlesk@vlesk-nb ~]$ cat hosts_from_s10
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
```

Также можно в явном виде указать имя пользователя на удаленной машине.

Пример:

```
[vlesk@vlesk-nb ~]$ scp root@10.255.255.100:/etc/hosts hosts_from_s10
root@10.255.255.100's password:
hosts                                         100%  158   128.7KB/s   00:00
[vlesk@vlesk-nb ~]$ cat hosts_from_s10
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
```



Не смотря на то, что команды `ssh` и `scp` используют шифрованный канал, во многих случаях аутентификацию с помощью пароля нельзя признать безопасной.

В таких случаях можно использовать криптографическую аутентификацию.

SSH предоставляет возможность использовать аутентификацию по протоколам RSA и DSA.

Несимметричное шифрование используется лишь на стадии аутентификации. После подтверждения аутентичности пользователя дальнейшая связь осуществляется с применением симметричного шифрования, так как оно обеспечивает приемлемый уровень быстродействия системы, в отличии от несимметричного.

Первое, что необходимо сделать для обеспечения возможности аутентификации с помощью RSA - это создать командой `ssh-keygen` пару ключей несимметричного шифрования.

После создания пары ключей несимметричного шифрования, необходимых для аутентификации пользователя, требуется поместить каким-либо путем публичный ключ на удаленный хост, с которым требуется обеспечить связь. Для этого можно скопировать с помощью `scp` публичный ключ с собственного узла на удаленный узел.

Пример:

```
[user@sl1 ~]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/user/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user/.ssh/id_rsa.
Your public key has been saved in /home/user/.ssh/id_rsa.pub.
The key fingerprint is:
```

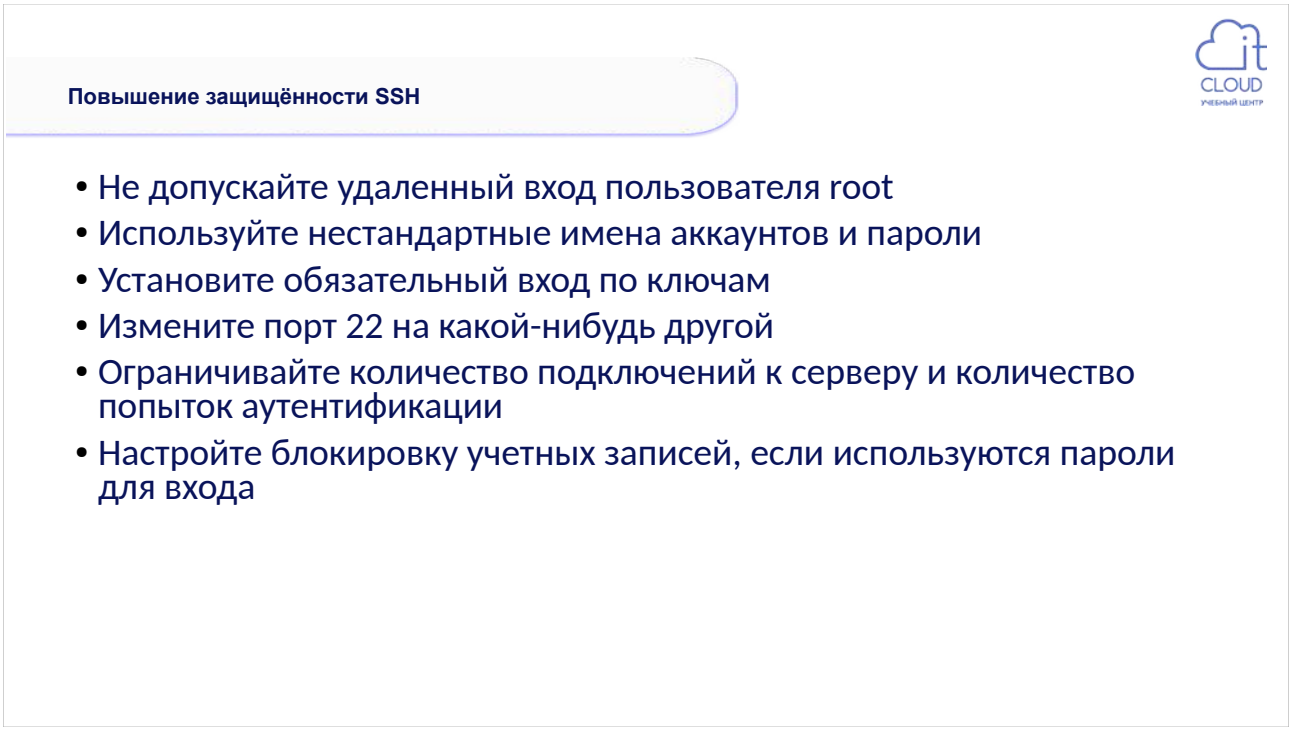
```
0f:a5:5f:36:a3:cd:9b:c7:95:aa:a6:37:51:b3:f5:fd user@sl1
The key's randomart image is:
+--[ RSA 2048]----+
|                 |
|                 |
|          .      |
|         o   o . |
|        S   * + +|
|         + B + .+|
|          + +....|
|           + +o E|
|         .+.=.   |
+-----------------+
[user@sl1 ~]$ scp /home/user/.ssh/id_rsa.pub 10.255.255.100:~/sl1.pub
The authenticity of host '10.255.255.100 (10.255.255.100)' can't be established.
ECDSA key fingerprint is 42:25:61:78:71:ff:2e:93:1b:68:20:bb:9f:86:18:17.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.255.255.100' (ECDSA) to the list of known hosts.
user@10.255.255.100's password:
id_rsa.pub                                    100%  390     0.4KB/s   00:00
[user@sl1 ~]$ ssh 10.255.255.100
user@10.255.255.100's password:
Last login: Wed Aug  2 21:37:36 2017
user@s10:~$ ssh 127.0.0.1
The authenticity of host '127.0.0.1 (127.0.0.1)' can't be established.
ECDSA key fingerprint is 42:25:61:78:71:ff:2e:93:1b:68:20:bb:9f:86:18:17.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '127.0.0.1' (ECDSA) to the list of known hosts.
user@127.0.0.1's password:

user@s10:~$ cat sl1.pub >> .ssh/authorized_keys
user@s10:~$ chmod 600 .ssh/authorized_keys
user@s10:~$ rm sl1.pub
user@s10:~$ logout
Connection to 10.255.255.100 closed.
[user@sl1 ~]$ ssh 10.255.255.100
Enter passphrase for key '/home/user/.ssh/id_rsa':
Last login: Sat Aug 26 20:54:50 2017 from 10.255.255.101
user@s10:~$ logout
Connection to 10.255.255.100 closed.
[user@sl1 ~]$
```

***Примечание**: Эта команда создает пару ключей RSA. Они помещаются в подкаталог .ssh домашнего каталога пользователя, вызвавшего команду. Файл id_rsa содержит частный ключ, доступ к которому должен быть предоставлен лишь его владельцу. Файл id_rsa.pub должен быть помещен на удаленный узел, с которым необходимо обеспечить связь по шифрованному каналу. Команда ssh 127.0.0.1 нужна для создания каталога .ssh*

Парольная фраза, которую требуется ввести здесь - это “пропуск” к частному ключу. Допускается не вводить ее, но в таком случае частный ключ не будет защищен.

Вместо копирования ключа вручную вы можете воспользоваться командой `ssh-copy-id`, которая скопирует все публичные ключи на сервер. С опцией `-i` можно указать, какой ключ надо копировать.

8.6 Повышение защищённости SSH.

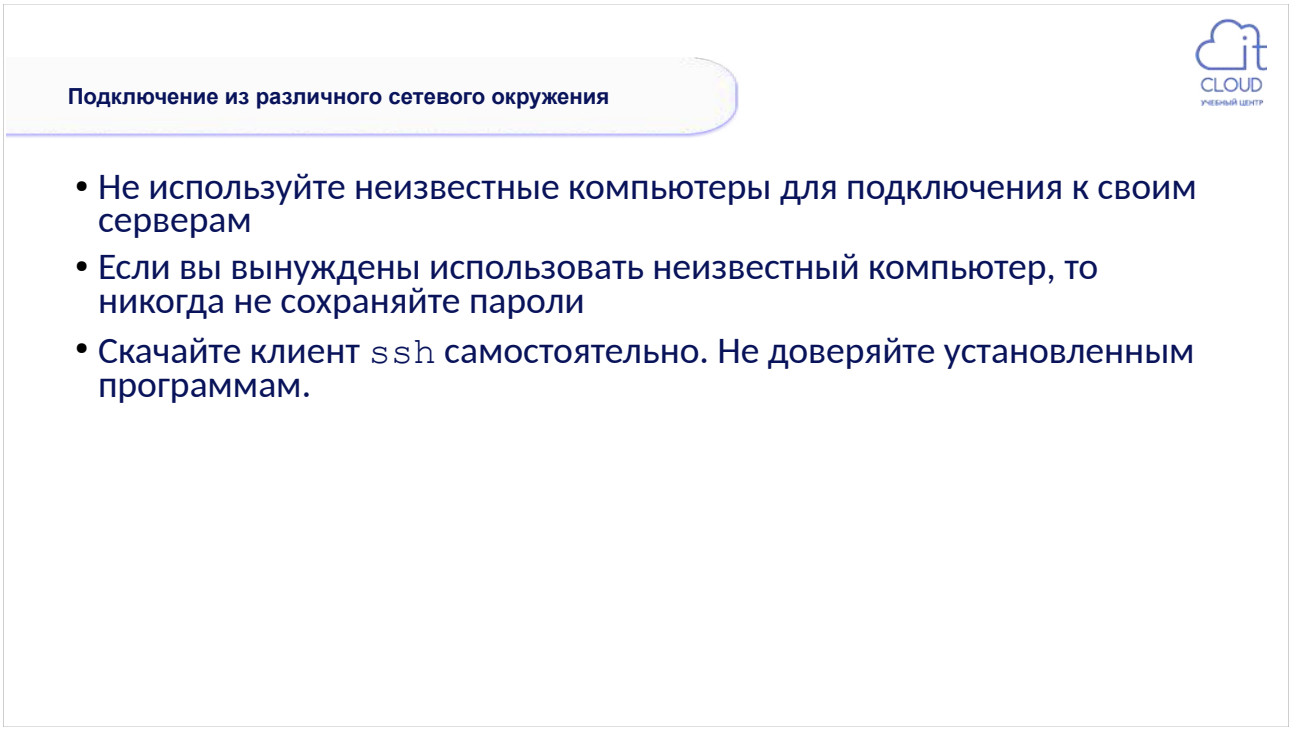


Протокол SSH обеспечивает богатые возможности по удаленному управлению и подключению, но, в то же время, является источником опасности. Имеется несколько рекомендаций по уменьшению рисков, связанных с использованием SSH.

- Не допускайте удаленный вход пользователя root. Настройка `PermitRootLogin no` или `PermitRootLogin without-password` или `PermitRootLogin forced-commands-only`
- Используйте нестандартные имена аккаунтов и пароли. Использование стандартных имен типа user, admin, administrator или root наиболее часто используются для подбора паролей.
- Установите обязательный вход по ключам. Блокировка входа всех пользователей может существенно повысить безопасность. Опция `PasswordAuthentication`.
- Измените порт 22 на какой-нибудь другой. Опция `Port` определяет порт, который прослушивает демон `sshd`.
- Ограничивайте количество подключений к серверу и количество попыток аутентификации. Опции MaxAuthTries, MaxSessions и MaxStartups ограничивают соответственно количество попыток аутентификации, количество сессий и количество попыток подключения.
- Nftables или iptables может быть использован для ограничения количества соединений в единицу времени с одного адреса.

- Вместо прямой настройки nftables удобней будет установить специальную службу sshguard или fail2ban, которая будет блокировать адреса, с которых осуществляются попытки атаки на ssh сервер.
- Настройте блокировку учетных записей, если используются пароли для входа. Блокировка учетных записей позволит эффективно бороться с атаками по подбору паролей.

8.7 Подключение из различного сетевого окружения.

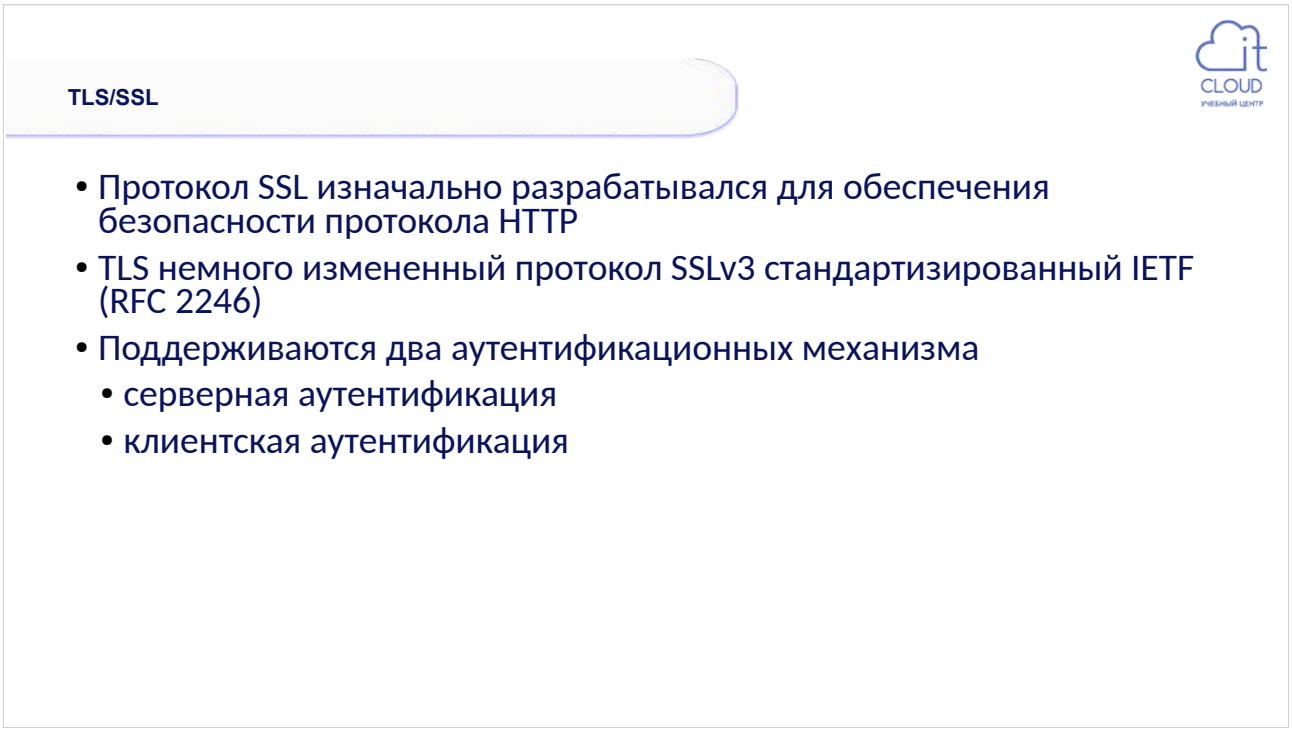


Вы никогда не сможете доверять не своим компьютерам. Поэтому помните несколько простых рекомендаций.

- По мере возможностей не используйте неизвестные компьютеры для подключения к своим серверам. Вы не можете полностью знать что и как работает на таком компьютере.
- Если вы вынуждены использовать неизвестный компьютер, то никогда не сохраняйте пароли. Сохранение паролей или имен пользователей упрощает жизнь пользователю, но при этом являются серьезной угрозой безопасности.
- Скачайте клиента `ssh` самостоятельно. Не доверяйте установленным программам. Клиента `ssh` не обязательно устанавливать можно найти несколько вариантов, которые работают без установки.

Глава 9. Инфраструктура открытых ключей на основе openssl.

9.1 TLS/SSL. Терминология и основные принципы.



Протокол SSL (Secure Sockets Layer) разрабатывался в компании Netscape Communications изначально для организации безопасного канала между двумя хостами, использующих HTTP протокол

Всего имеются две версии SSL: SSLv2 и SSLv3. Первая версия не была опубликована.

Протокол SSLv2 был выпущен в 1994 году и имел главной задачей организацию безопасного канала в WWW окружении, хотя и предполагалась возможность использования SSLv2 и для других протоколов

Протокол SSLv3 выпущен в 1995 году в него были включены аутентификация, поддержка большого количества криптографических алгоритмов.

В 1997 году комитет IETF(Internet Engineering Task Force) стандартизовал SSL подобный протокол под названием TLS (Transport Layer Security, RFC 2246), который является, по сути, немного измененным протоколом SSLv3. Текущая версия протокола – первая (TLSv1). Дата ее публикации январь 1999 года. Термины SSL и TLS очень часто используются взаимозаменяемо.

TLSv1 можно использовать как при взаимодействии между клиентом и сервером, так и между двумя серверами (например, при репликации данных каталогов)

TLSv1 можно использовать, как с простой аутентификацией (характерное имя и пароль), так и с аутентификацией на базе сертификатов.

Различают два аутентификационных механизма, которые может осуществлять TLSv1

1. серверная аутентификация
2. клиентская аутентификация

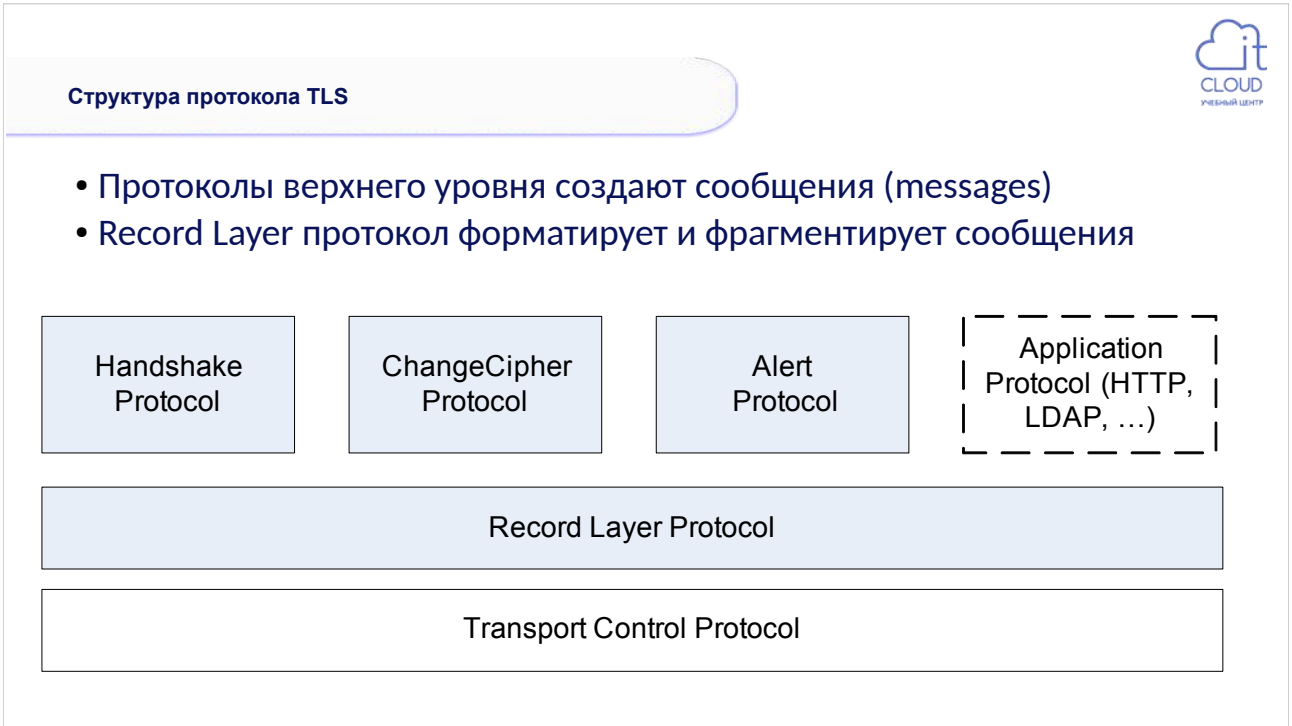
При использовании серверной аутентификации клиент решает, доверяет ли он сертификату, который прислал сервер

Во время клиентской аутентификации клиент решает, доверяет ли он серверному сертификату, а сервер решает, доверяет ли он сертификату, предоставленному клиентом.

Серверная аутентификация обеспечивает шифрование данных и их целостность. Аутентификация клиента обеспечивается другими механизмами, например, простым (simple) механизмом (DN+пароль) или SASL механизмом.

Серверная аутентификация не требует специального клиентского обеспечения и не требует создания и хранения сертификата на клиентском хосте.

Клиентская аутентификация обеспечивает и шифрование и целостность данных, а также, собственно, аутентификацию клиента на базе сертификата.



Протокол TLS состоит из двух уровней:

1. На первом уровне находятся Handshaking Protocol, ChangeCipherSpec Protocol, Alert Protocol
2. Record Layer Protocol

Четыре протокола верхнего слоя, включая и один из протоколов прикладного уровня, который не принадлежит к TLS, создают TLS сообщения (messages) и направляют их в протокол нижнего уровня – Record Layer Protocol.

Общение клиента и сервера происходит с помощью SSL сообщений (message). Протокол TLS определяет 13 типов сообщений.

Record Layer протокол форматирует и фрагментирует сообщения, а затем отправляет полученные кадры (фрагменты) транспортному уровню (TCP), тем самым обеспечивается единый формат данных, которые TLS поставляет на транспортный уровень.

Record Layer инкапсулирует все сообщения в кадр (frame) и, если нужно, добавляет MAC (message authentication code) для обеспечения целостности данных и шифрует данные вместе с MAC.

MAC – это либо MD5, либо SHA хеш данных, которые помещаются в кадры протокола Record Layer.

Для эффективности TCP может помещать несколько кадров протокола Record Layer в один сегмент

Протокол ChangeCipherSpec обеспечивает выбор механизмов предназначенных для шифрования и гарантирования целостности данных и определяет один тип сообщений

Протокол Alert обеспечивает информирование одной из сторон, участвующих в соединении, об ошибках произошедших на другой стороне.

Протокол Handshake наиболее сложен и важен. Именно он обеспечивает инициализацию безопасного соединения между двумя хостами.

TLS использует 9 сообщений для установки безопасного (шифрованного) соединения.

В сообщении ClientHello содержатся следующие параметры

1. Version -версия TLS/SSL
2. RandomNumber – случайное число, которое вместе с таким же параметром из сообщения ServerHello, будет использоваться в качестве "зерна" (seed) в криптографических вычислениях.
3. SessionID - в данном случае пустое поле.
4. CipherSuites – перечисляются криптографические службы, которые поддерживаются клиентом
5. CompressionMethods - перечисляются поддерживаемые клиентом методы сжатия.

Сообщение ServerHello состоит из тех же пяти параметров. Сервер в этом сообщении делает выбор на основе предложений, содержащихся в сообщении ClientHello. Поле SessionID содержит уникальный номер TLS сессии.

Сообщение ServerKeyExchange дополняет параметр CipherSuites из сообщения ServerHello. Если в параметре CipherSuites содержатся имя криптографического алгоритма и размер ключей, то здесь передается сам публичный ключ.

В сообщении ClientKeyExchange клиент сообщает серверу разделяемый (shared) ключ, который в дальнейшем используется для симметричного шифрования. При этом симметричный ключ шифруется публичным ключом сервера.

При обмене сообщениями ChangeCipherSpec утверждается (или изменяется) выбор криптографических алгоритмов, который был сделан при обмене сообщениями ClientHello/ServerHello.

При этом утверждаются алгоритм для симметричного шифрования (например, DES) и метод проверки целостности данных (например, MD5) с помощью, которого создается MAC. Алгоритмы могут различаться в разных направлениях.

С помощью сообщений Finished стороны подтверждают успешное завершение стадии установления безопасного соединения.

При завершении соединения стороны обмениваются сообщениями типа ClosureAlert

Изложенная выше схема показывает, как устанавливается шифрованное соединение, но не описывает аутентификацию сторон, участвующих в соединении.

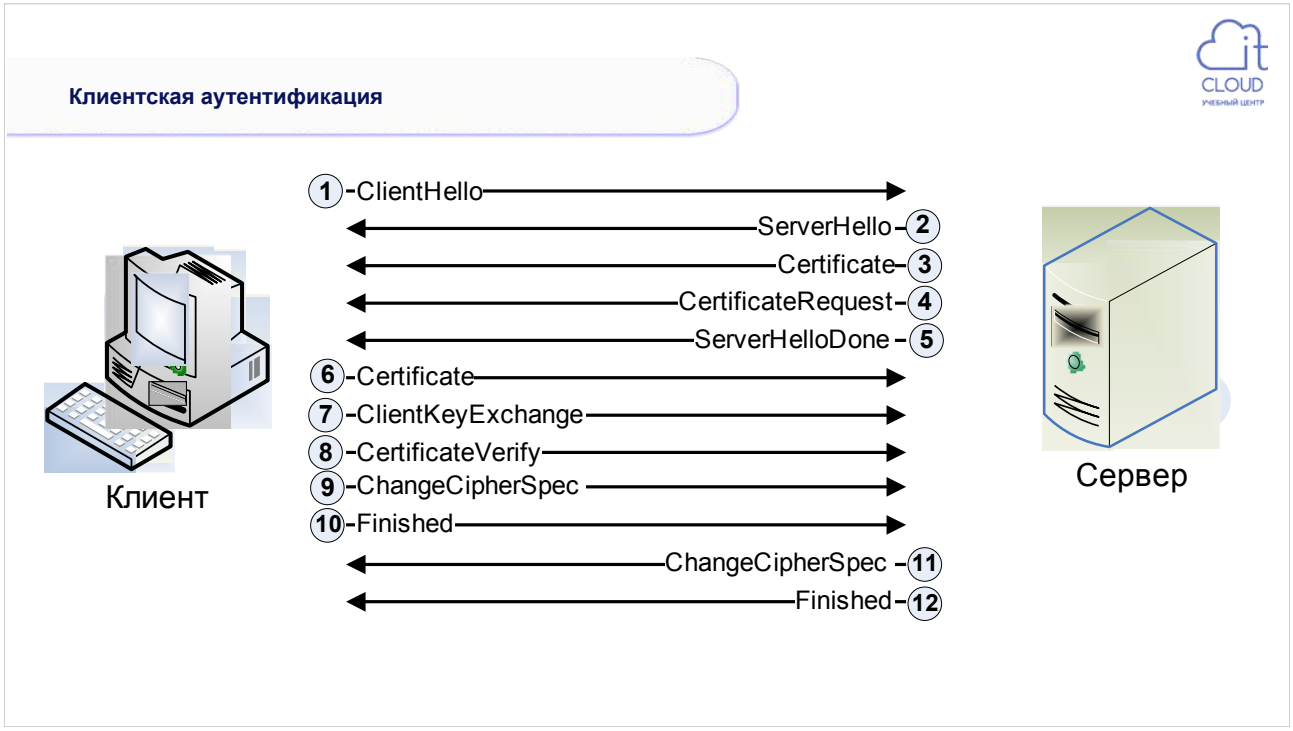
Серверная аутентификация происходит практически по той же схеме, что и установление шифрованного канала.

Изменения заключаются в следующем:

- Вместо сообщения № 3 ServerKeyExchange сервер направляет сообщение Certificate. В этом сообщении содержится цепочка сертификатов от сертификата сервера с его публичным ключом до сертификата корневого центра сертификации (root authority)
- При формировании сообщения №5 ClientKeyExchange клиент шифрует симметричный общий ключ публичным ключом, хранящимся в сертификате сервера, тем самым подтверждая принятие серверного сертификата.

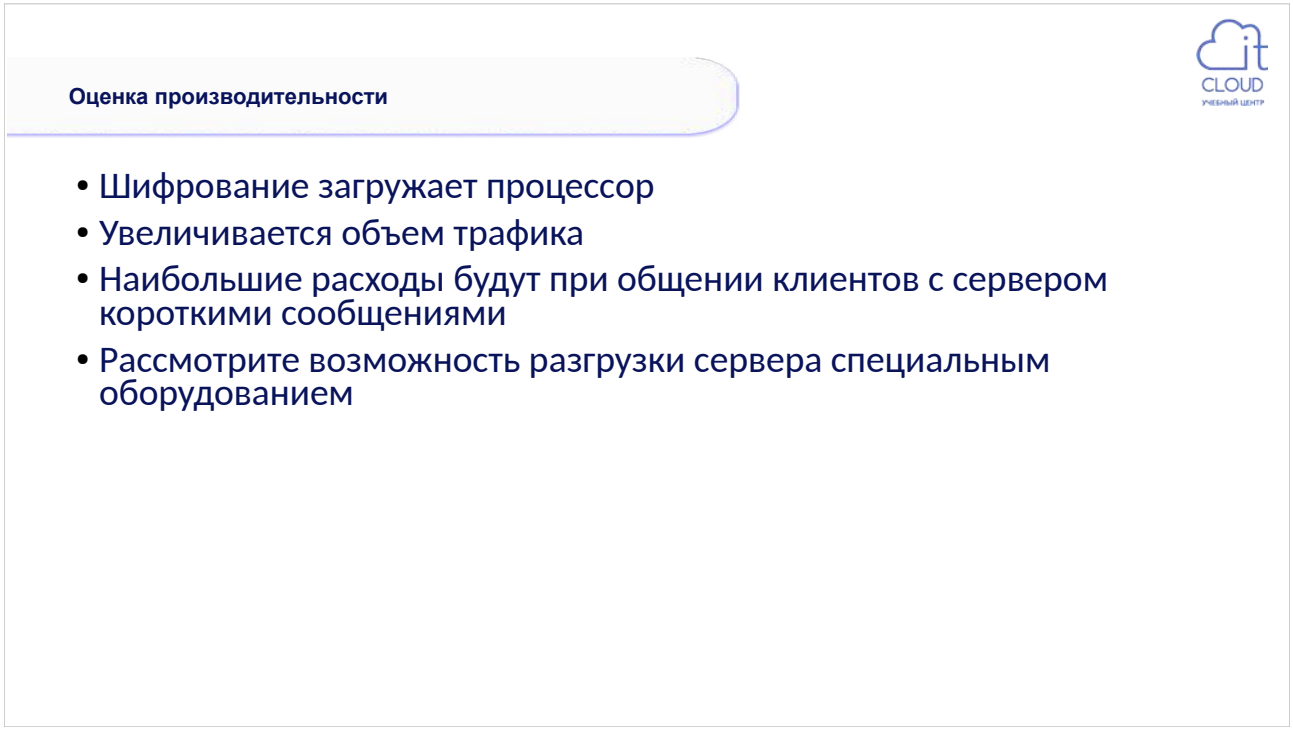
При установлении безопасного соединения с серверной аутентификацией можно разделить процессы шифрования и аутентификации. В этом случае после посылки сервером сообщения Certificate сервер направляет клиенту сообщение ServerKeyExchange. Общее число сообщений увеличивается до 10.

Это делается, например, в тех случаях, когда нельзя в силу каких-то причин использовать публичный ключ сервера. Одна из возможных причин – запреты местного законодательства.



Клиентская аутентификация требует обмена 12 TLS сообщениями. Добавляются

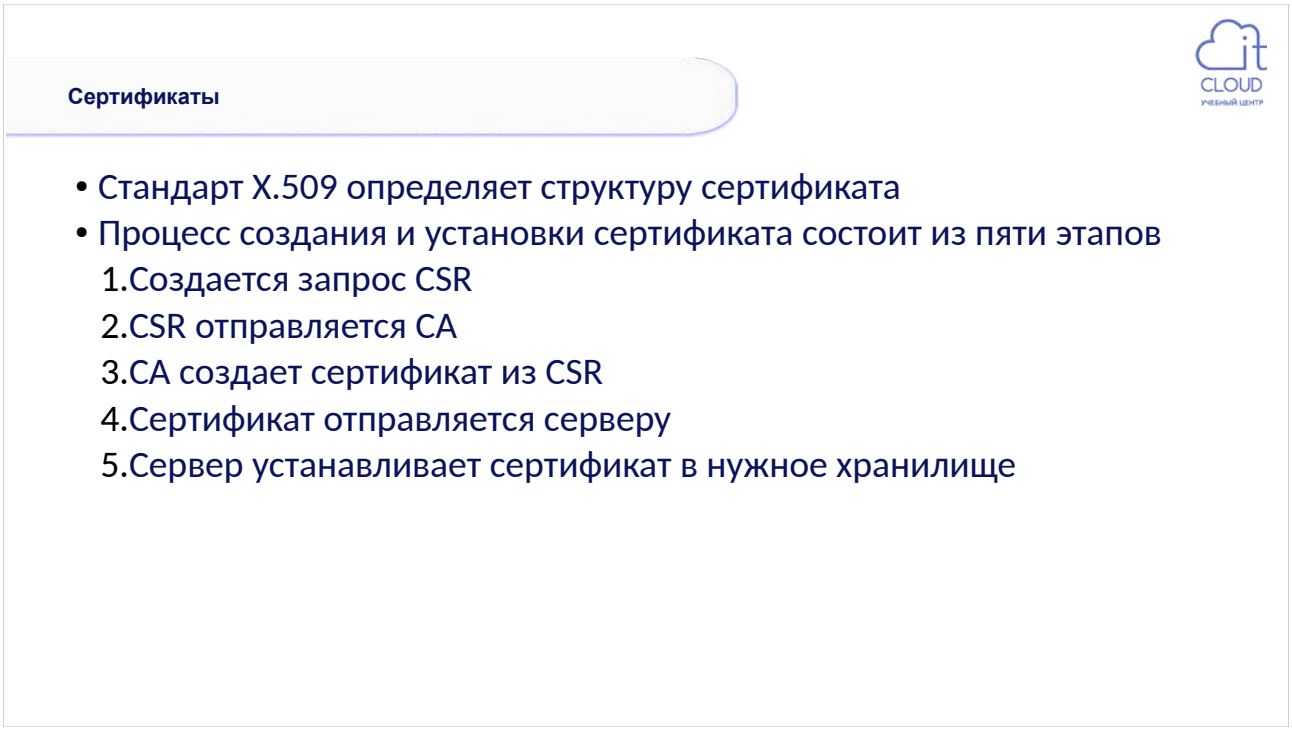
1. сообщение сервера CertificateRequest
2. два сообщения клиента Certificate и CertificateVerify



При принятии решения об использовании TLS нужно имет в виду следующие факторы.

- Шифрование накладывает требует повышенной производительности хоста, особенно процессора.
- Увеличивается объем трафика.
- Поскольку установка TLS соединения влечет много накладных расходов, то наиболее неприятной является ситуация, когда серверу поступает много коротких запросов от разных хостов.
- Вы можете использовать специализированное оборудование для шифрования, например специальные сетевые карты.

9.2 Сертификаты.



Сертификаты с публичным ключом (public key certificate) – это цифровой аналог паспорта (водительских прав, удостоверения личности)

Структура сертификата с публичным ключом закреплена в протоколе X.509

Типовой сертификат с публичным ключом содержит 11 полей:

1. Version
2. Serial Number
3. Algorithm Identifier (в стандарте X.509 имеет название Signature)
4. Issuer
5. Period of Validity
6. Subject
7. Subject's Public Key
8. Issuer Unique ID
9. Subject Unique ID
10. Extension
11. Signature (в стандарте X.509 имеет название Encrypted)

Наибольший интерес представляют поля Issuer, Period of Validity, Subject, Subject's Public Key и Signature

- Поле Issuer содержит характерное (DN) имя организации-нотариуса (CA, Certificate Authority), которая создала и подписала сертификат

- Поле Period of Validity содержит начало и конец срока действия сертификата
- Поле Subject – характерное имя (DN) субъекта, чей приватный ключ (private key) удостоверяет данный сертификат. Приватный ключ не хранится в сертификате.
- Поле Subject's Public Key содержит публичный ключ, который соответствует приватному ключу субъекта, указанной в поле Subject. Кроме того, здесь хранится имя алгоритма публичных ключей.
- Поле Signature – цифровая подпись содержимого сертификата. Создается путем получения хеша из содержимого сертификата с помощью приватного ключа организации из поля Subject. Тем самым гарантируется целостность данных сертификата при его передаче.

Процесс создания и установки сертификата состоит из пяти этапов

1. Создание запроса на подпись сертификата (CSR, Certificate Signing Request)
2. Отправка CSR CA. Это может быть осуществлено по почте, через веб и т. д.
3. CA создает сертификат из CSR и подписывает его своим частным ключом
4. Получение подписанного сертификата от CA. Это также производится с помощью электронной почты, веб-доступа или простой передачей файла.
5. Импорт сертификата в хранилище на сервере. Формат хранилища определяется в соответствии с требованиями прикладного программного обеспечения.

Представление сертификата в шифрованном виде определяется Правилами Шифрования Различимых имен (Distinguised Encoding Rules - DER). Эти правила являются подмножеством Основных Правил Шифрования (Basic Encoding Rules - BER).

Правила DER определяют представление сертификата в бинарном виде.

В случаях, когда бинарное представление сертификата использовано быть не может, сертификат представляется в ASCII виде с помощью Base64 Encodings. Такое представление называется PEM (Privacy Enhanced Mail).

Если сертификат не может быть подписан в агентстве сертификации более высокого уровня, то он является подписанным самостоятельно (self-signed). То есть агентство сертификации и субъект сертификации являются одним и тем же.

Уровень доверия к таким сертификатам должен быть ниже и можно ли им доверять - это решение должен принимать сам пользователь.

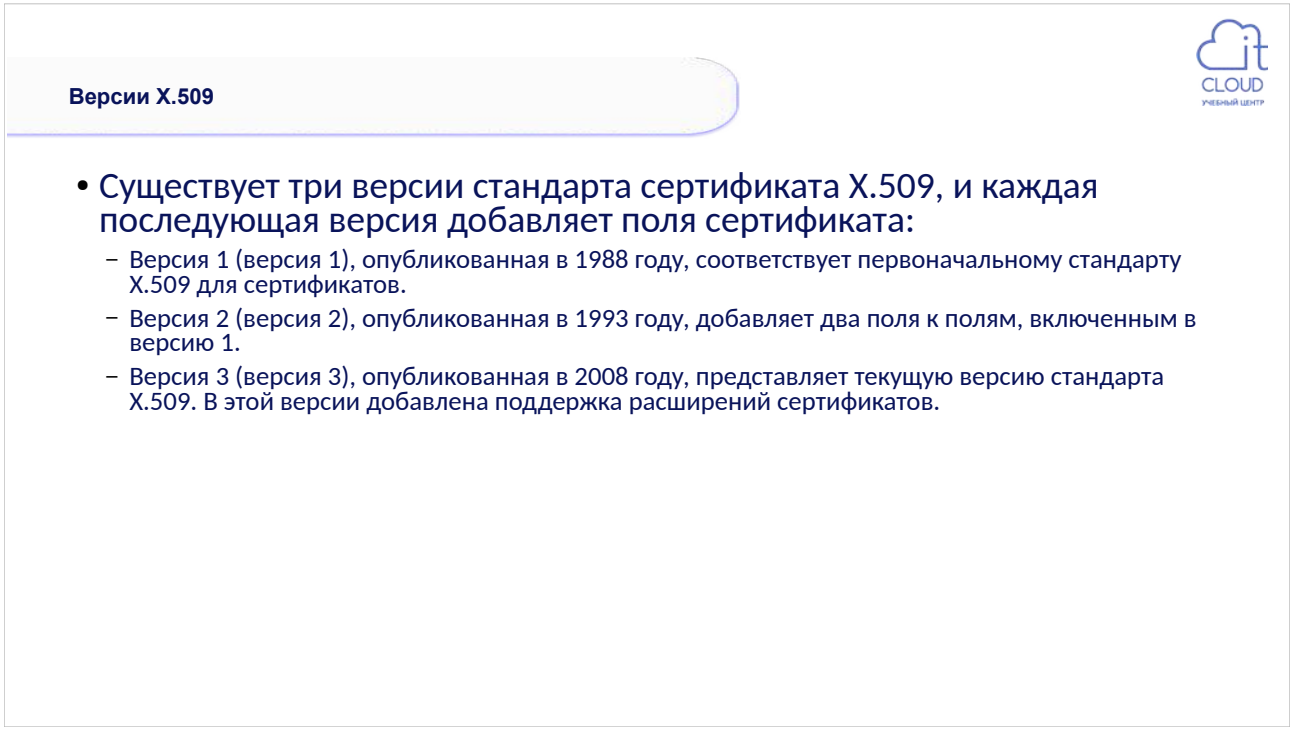
Для организаций, которым требуется создание защищенных корпоративных информационных служб, имеется возможность создания собственных агентств сертификации.

В таком случае клиентское ПО корпоративных пользователей должно содержать публичный ключ собственного агентства сертификации.

Управление сертификатами заключается в определение сроков действия сертификатов, отзыве их в следствие каких-либо событий, обновление сертификатов.

Один из аспектов управления сертификатами - поддержка списков отозванных сертификатов CRL (Certificate Revocation Lists). Отозванный сертификат это такой сертификат, который стал недействительным раньше срока действия сертификатов.

ПО работающие с сертификатами может использовать CRL для проверки действительности сертификатов. Чтобы проверить сертификат в CRL необходимо знать где его публикует CA.

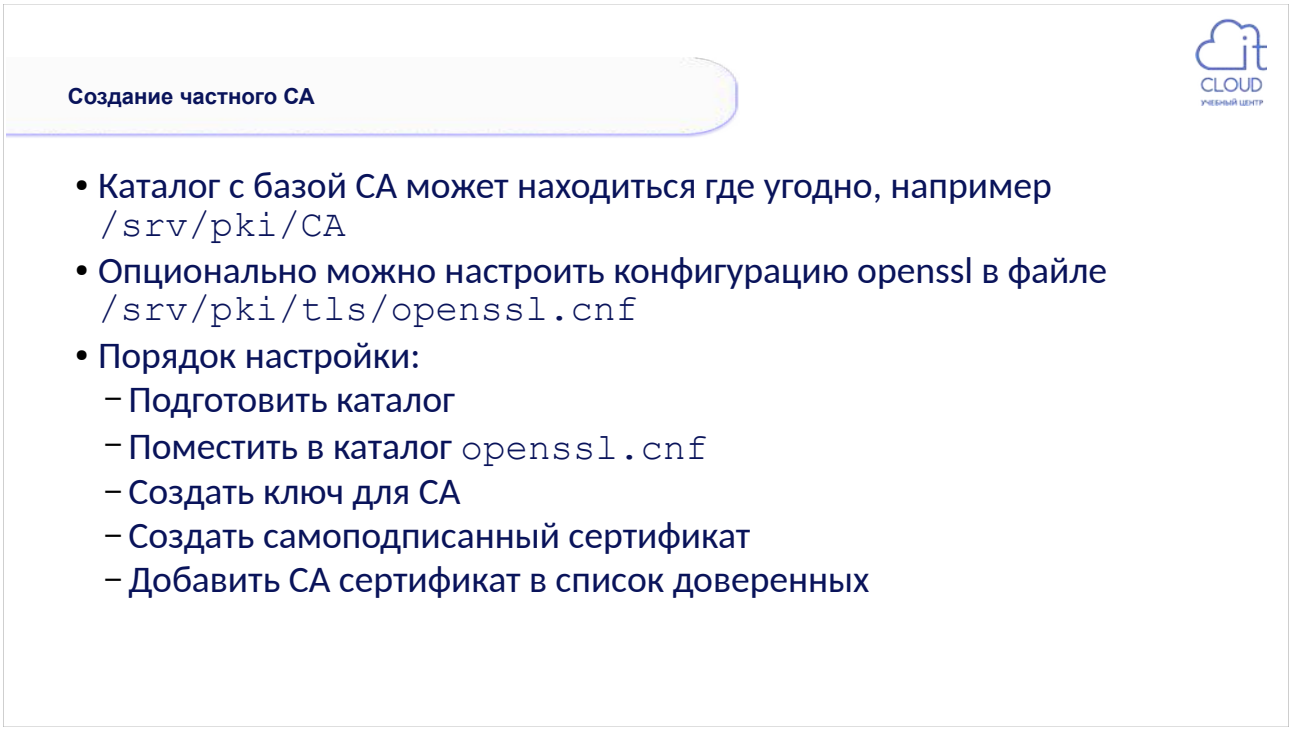


Существует три версий сертификатов. Актуальная версия 3.

Третья версия содержит поле Extensions, которое используется для связывания большего числа атрибутов с пользователями или открытыми ключами, а также для управления связями между центрами сертификации. Наиболее значимые расширения: Subject Alternative Name (SAN), Key Usage, Basic Constraints, CRL Distribution Points.

Хорошая статья о версиях сертификатов опубликована у Microsoft (https://learn.microsoft.com/en-us/azure/iot-hub/reference-x509-certificates). Статья фактически кратко разбирает и объясняет документ RFC5280 (https://www.rfc-editor.org/rfc/rfc5280).

9.3 Создание частного агентства сертификации.



Если организация планирует использовать корпоративную информационную систему без необходимости допуска сторонних клиентов извне может быть создано собственное агентство сертификации.

Для работы с CA в RedHat подобных линуксах имеется каталог `/etc/pki/CA`, но вы можете использовать любой другой. В других системах специального каталога может и не быть.

Создаем в любом месте файловой системы (например, `/srv/pki/CA`) структуру каталогов и нужные файлы

```
root@s10:~# mkdir -p /srv/pki/CA && cd /srv/pki/CA
root@s10:/srv/pki/CA# mkdir certs crl newcerts private
root@s10:/srv/pki/CA# echo "01" > serial
root@s10:/srv/pki/CA# > index.txt
root@s10:/srv/pki/CA# cd ..
```

Копируем в каталог CA конфигурационный файл `openssl – openssl.cnf`

```
root@s10:/srv/pki# cp /etc/ssl/openssl.cnf .
```

Вносим изменения в `openssl.cnf` (например, исправляем в секции `[CA_default]` переменную `dir – dir = ./CA`). Обратите внимание и на следующие опции: `countryName_default`, `stateOrProvinceName_default`, `localityName_default`, `0.organizationName_default`, `organizationalUnitName_default`.

Так же настройте опции в файле `openssl.cnf` следующим образом:

```
[ CA_default ]

dir             = /srv/pki/CA           # Where everything is kept
certs           = $dir/certs            # Where the issued certs are kept
crl_dir         = $dir/crl              # Where the issued crl are kept
database        = $dir/index.txt        # database index file.
#unique_subject = no                    # Set to 'no' to allow creation of
                                        # several certs with same subject.
new_certs_dir   = $dir/newcerts         # default place for new certs.

certificate     = $dir/certs/ca.crt             # The CA certificate
serial          = $dir/serial           # The current serial number
crlnumber       = $dir/crlnumber        # the current crl number
                                        # must be commented out to leave a V1
CRL
crl             = $dir/crl.pem          # The current CRL
private_key     = $dir/private/ca.key   # The private key

x509_extensions = usr_cert              # The extensions to add to the cert
copy_extensions = copyall               # ОЧЕНЬ ОПАСНО!!!

# Пропущено несколько строк
[ policy_match ]
countryName             = match
stateOrProvinceName     = optional
organizationName        = optional
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

# Пропущено несколько строк
[ req_distinguished_name ]
countryName                     = Country Name (2 letter code)
countryName_default             = RU
countryName_min                 = 2
countryName_max                 = 2

stateOrProvinceName             = State or Province Name (full name)
stateOrProvinceName_default     = Sverdlovsk

localityName                    = Locality Name (eg, city)
й
0.organizationName              = Organization Name (eg, company)
0.organizationName_default      = IT Cloud

# we can do this but it is not needed normally :-)
#1.organizationName             = Second Organization Name (eg, company)
#1.organizationName_default     = World Wide Web Pty Ltd

organizationalUnitName          = Organizational Unit Name (eg, section)
organizationalUnitName_default  = Class SecurL
```

Вначале создается приватный ключ агентства:

```
root@s10:/srv/pki# openssl genrsa -aes256 -out CA/private/ca.key 2048
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
```

В файл `ca.key` будет помещен приватный ключ агентства, зашифрованный AES256 в формате PEM.

Проверить созданный приватный ключ можно с помощью команды:

```
root@s10:/srv/pki# openssl rsa -noout -text -in CA/private/ca.key
Enter pass phrase for CA/private/ca.key:
Private-Key: (2048 bit, 2 primes)
modulus:
    00:c8:3d:01:cc:9d:c2:02:c2:11:23:9d:6e:2e:c3:
---
```

Создать расшифрованную версию приватного ключа (что опасно!) можно так:

```
root@s10:/srv/pki# openssl rsa -in CA/private/ca.key -out ca.key.unsecure
```

Далее необходимо создать самоподписанный сертификат агентства. Так как мы не используем каталог по умолчанию определенный для данной версии, то нам стоит определить переменную OPENSSL_CONF, которая укажет на собственный конфигурационный файл:

```
root@s10:/srv/pki# openssl version -d
OPENSSLDIR: "/usr/lib/ssl"
root@s10:/srv/pki# ls -l /usr/lib/ssl/openssl.cnf
lrwxrwxrwx. 1 root root 20 окт 27 19:16 /usr/lib/ssl/openssl.cnf ->
/etc/ssl/openssl.cnf

root@s10:/srv/pki# export OPENSSL_CONF=/srv/pki/openssl.cnf

root@s10:/srv/pki# openssl req -new -x509 -days 36525 -key CA/private/ca.key
-out CA/certs/ca.crt
Enter pass phrase for CA/private/ca.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [RU]:
State or Province Name (full name) [Sverdlovsk]:
Locality Name (eg, city) [Ykaterinburg]:
Organization Name (eg, company) [IT Cloud]:
Organizational Unit Name (eg, section) [Class SecurL]:
Common Name (e.g. server FQDN or YOUR name) []:My CA cert
Email Address []:
```

В данном случае срок действия сертификата указан как 100 лет.

Проверить сертификат можно с помощью:

```
root@s10:/srv/pki# openssl x509 -in CA/certs/ca.crt -text -noout
Certificate:
    Data:
        Version: 3 (0x2)
```

```
        Serial Number:
            37:b8:96:9c:cc:be:7b:df:0d:6c:c8:5e:0e:ca:eb:6b:42:da:a4:09
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = RU, ST = Sverdlovsk, L = Ykaterinburg, O = IT Cloud, OU =
Class SecurL, CN = My CA cert
        Validity
            Not Before: Feb  6 17:02:51 2025 GMT
            Not After : Feb  7 17:02:51 2125 GMT
        Subject: C = RU, ST = Sverdlovsk, L = Ykaterinburg, O = IT Cloud, OU =
Class SecurL, CN = My CA cert
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:c8:3d:01:cc:9d:c2:02:c2:11:23:9d:6e:2e:c3:
                    f3:5d:b9:01:48:ff:1a:80:03:d2:0b:42:a0:54:f8:
                    c3:6a:32:e2:8d:5c:bd:f4:e7:17:56:7d:1d:5b:09:
                    73:60:6d:99:fc:b3:d5:3e:82:3a:50:fb:dd:64:09:
                    94:65:21:31:81:1c:af:32:9f:90:b8:9d:57:4d:28:
                    91:df:67:2a:df:89:c7:60:ca:7e:79:66:f6:ed:5c:
                    49:1e:6f:f0:d1:14:08:44:fd:bf:6d:d4:02:5e:54:
                    9f:78:7d:61:2e:45:be:18:be:24:17:d8:0e:0d:3d:
                    a6:ae:f4:97:b9:91:9d:16:84:ef:50:8a:d6:ad:83:
                    b4:df:af:29:47:bf:32:c2:fe:bf:46:ce:81:eb:d3:
                    18:79:38:3a:83:72:58:05:c8:82:09:8a:ab:47:d5:
                    19:22:93:8a:84:be:fd:b1:e5:c4:a2:e2:5f:e9:ea:
                    55:36:2e:f1:c1:88:de:4f:d3:f0:83:34:d9:a7:0e:
                    a2:6a:56:a4:16:97:05:5f:ea:c2:3b:91:77:c0:f8:
                    f5:c5:2a:a6:d6:a8:b9:0e:3b:71:cc:d9:99:1d:1e:
                    fd:fb:b9:ef:b4:4d:76:68:f6:ad:41:7c:51:6d:8c:
                    1a:93:90:1d:fa:cf:a1:2f:f0:ae:8a:b8:1b:ad:3b:
                    09:f1
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                FD:A4:25:32:14:F0:0A:3A:E9:69:B1:2F:F3:3A:86:DC:1C:09:38:1F
            X509v3 Authority Key Identifier:
                FD:A4:25:32:14:F0:0A:3A:E9:69:B1:2F:F3:3A:86:DC:1C:09:38:1F
            X509v3 Basic Constraints: critical
                CA:TRUE
    Signature Algorithm: sha256WithRSAEncryption
    Signature Value:
        c2:f7:b4:65:30:94:64:c6:bd:fa:f1:54:cc:4c:9a:b1:11:87:
        7b:39:87:9a:f3:1d:8a:b4:79:9b:14:23:88:32:ce:28:7d:a4:
        17:fb:f2:dc:49:c3:cf:0c:8a:dd:93:16:3d:df:1c:f2:e7:f3:
        8b:14:24:b5:09:9f:36:60:03:ef:18:41:08:aa:e4:29:0a:6b:
        5a:f0:40:de:61:fa:f1:7f:b3:f0:eb:c4:25:2d:e2:e9:c9:49:
        7b:69:26:03:68:88:62:f8:74:bf:06:00:4c:4a:43:62:06:81:
        f5:7a:d4:59:24:cc:08:6f:a8:55:dc:4d:f5:5d:e3:d0:92:64:
        f7:80:ec:55:eb:6a:d6:d8:89:27:ed:23:c7:f9:57:ce:61:42:
        ef:df:e1:70:01:27:78:c5:e7:dc:9f:d4:6b:7a:2f:cf:a6:e7:
        cc:d0:4f:6f:7e:a7:68:13:f5:f4:dd:5d:ef:f6:8d:73:4b:ed:
        ab:19:f9:f2:22:80:09:28:98:e1:ef:56:a9:a8:12:04:2f:39:
        8b:32:ae:44:35:7d:a5:7f:6a:d4:79:ff:a7:51:35:b2:01:70:
        b3:9b:47:88:07:4e:f2:ad:9a:70:7c:b1:57:94:27:88:32:86:
        b6:2c:bd:1a:6d:9f:70:7f:6e:92:5b:3e:df:47:fb:ab:10:c0:
        f9:68:83:58
```

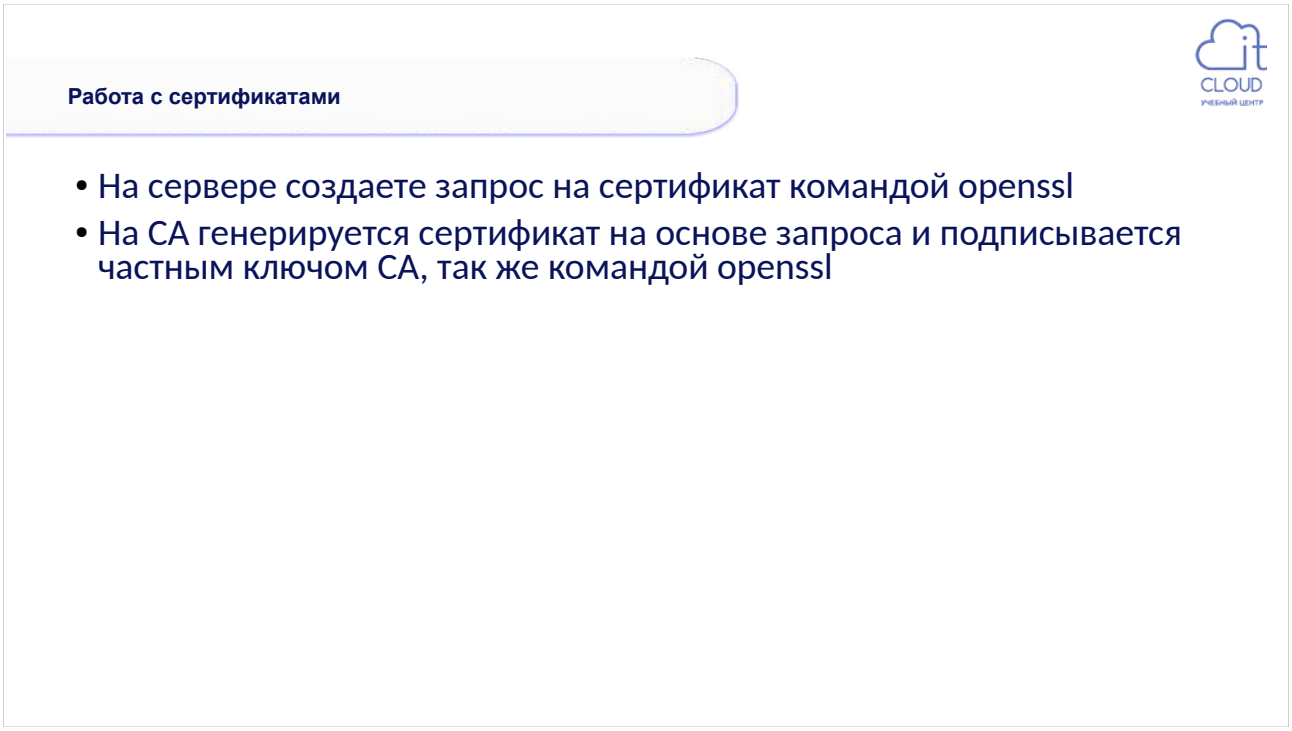
Далее нам необходимо добавить сертификат CA в список доверенных:

```
root@s10:/srv/pki# mkdir /usr/local/share/ca-certificates/myca
root@s10:/srv/pki# cp CA/certs/ca.crt /usr/local/share/ca-certificates/myca
root@s10:/srv/pki# update-ca-certificates
Updating certificates in /etc/ssl/certs...
rehash: warning: skipping ca-certificates.crt,it does not contain exactly one
certificate or CRL
1 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
```

Проверим попал ли сертификат в список доверенных:

```
root@s10:/srv/pki# openssl crl2pkcs7 -nocrl -certfile /etc/ssl/certs/ca-
certificates.crt | openssl pkcs7 -print_certs -noout | grep subject | grep 'My
CA cert'
subject=C = RU, ST = Sverdlovsk, L = Ykaterinburg, O = IT Cloud, OU = Class
SecurL, CN = My CA cert
```

9.4 Работа с сертификатами.



Вначале требуется создать приватный ключ сервера. Для этого следует выполнить команду:

```
root@s10:~# cd /etc/ssl/
root@s10:/etc/ssl# unset OPENSSL_CONF
root@s10:/etc/ssl# openssl genrsa -out private/s10.key 2048
```

В этом примере приватный ключ сервера будет записан в файл `s10.key`. Ключ в данном случае создается не шифрованный.

Теперь можно создать запрос на подпись сертификата (Certificate Signing Request):

```
root@s10:/etc/ssl# openssl req -new -key private/s10.key -out s10.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:RU
State or Province Name (full name) [Some-State]:Свердловская область
Locality Name (eg, city) []:Екатеринбург
Organization Name (eg, company) [Internet Widgits Pty Ltd]:АйТи Клауд
Organizational Unit Name (eg, section) []:Класс SecurL
Common Name (e.g. server FQDN or YOUR name) []:s10.class.itcloud
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
```

```
An optional company name []:
```

Обратите внимание, что мы в данном примере используем конфигурацию openssl.cnf по умолчанию, поэтому все данные для запрашиваемого сертификата мы указали вручную.

Данная команда создает в файле `s10.csr` запрос на подпись сертификата в формате PEM, которую можно направить в агентство сертификации.

Содержимое запроса на подпись сертификата может быть проверено с помощью команды:

```
root@s10:/etc/ssl# openssl req -noout -text -in s10.csr
Certificate Request:
    Data:
        Version: 1 (0x0)
        Subject: C = RU, ST =
\C3\90\C2\A1\C3\90\C2\B2\C3\90\C2\B5\C3\91\C2\80\C3\90\C2\B4\C3\90\C2\BB\C3\90\C
2\BE\C3\90\C2\B2\C3\91\C2\81\C3\90\C2\BA\C3\90\C2\B0\C3\91\C2\8F
\C3\90\C2\BE\C3\90\C2\B1\C3\90\C2\BB\C3\90\C2\B0\C3\91\C2\81\C3\91\C2\82\C3\91\C
2\8C, L =
\C3\90\C2\95\C3\90\C2\BA\C3\90\C2\B0\C3\91\C2\82\C3\90\C2\B5\C3\91\C2\80\C3\90\C
2\B8\C3\90\C2\BD\C3\90\C2\B1\C3\91\C2\83\C3\91\C2\80\C3\90\C2\B3, O =
\C3\90\C2\90\C3\90\C2\B9\C3\90\C2\A2\C3\90\C2\B8
\C3\90\C2\9A\C3\90\C2\BB\C3\90\C2\B0\C3\91\C2\83\C3\90\C2\B4, OU =
\C3\90\C2\9A\C3\90\C2\BB\C3\90\C2\B0\C3\91\C2\81\C3\91\C2\81 SecurL, CN =
s10.class.itcloud
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:ae:35:d5:4d:98:29:66:69:a7:da:09:2b:ea:2f:
                    0e:86:08:6a:56:8c:f6:d6:d9:66:76:92:59:81:fd:
                    29:a0:7b:98:74:69:91:36:33:3f:1b:73:a8:a5:e5:
                    e9:3b:2a:12:b6:e4:7c:1a:82:14:de:32:1a:c3:d3:
                    8a:f5:4e:92:3e:68:b8:c0:f0:8f:4b:23:4a:95:d9:
                    2e:61:0b:8f:40:94:4c:1f:2e:95:8b:63:e7:fd:1b:
                    49:c6:cf:0d:62:ef:42:0d:65:5e:c4:4d:d5:85:c5:
                    f0:86:6a:12:bc:7c:bd:aa:10:ae:ec:6f:00:64:84:
                    ea:f0:e0:ab:b4:f2:fc:d3:8a:c0:c1:9b:a0:8b:cf:
                    b2:7b:14:c9:6f:b0:ab:22:ae:ab:0b:a3:6d:c0:af:
                    f8:cc:00:e8:66:aa:aa:ed:6d:b5:d7:66:42:08:4c:
                    bd:33:72:b5:fb:51:cf:0c:78:88:bf:3b:a4:90:4e:
                    cf:a6:4b:10:18:e7:51:5f:17:80:13:e9:f8:cd:77:
                    44:78:09:ef:f6:1d:d6:29:1e:d8:a2:df:ea:20:e5:
                    96:57:37:ea:f6:54:3b:01:b6:a1:18:93:a3:81:9b:
                    f4:ca:72:59:5d:5a:69:d8:34:88:4b:14:ab:07:49:
                    2a:b2:dd:97:79:b5:26:ce:5f:6f:59:02:db:f0:99:
                    75:f9
                Exponent: 65537 (0x10001)
        Attributes:
            (none)
            Requested Extensions:
    Signature Algorithm: sha256WithRSAEncryption
    Signature Value:
        95:c0:50:10:ce:ed:7c:cb:98:8b:33:3e:95:0e:be:b0:1c:92:
        78:ab:d2:e8:df:d3:a8:e4:c4:24:0d:2f:04:d0:e4:f4:f5:32:
        78:c4:ad:6f:9c:52:92:5e:91:ee:4c:41:ae:d0:89:46:f9:35:
        a6:09:b1:11:2e:25:7a:9f:2a:c4:b6:47:3e:6c:c2:42:b4:0c:
        b5:18:d5:01:96:df:20:a7:db:44:9d:81:01:08:85:a4:ac:31:
```

```
        3a:b7:c3:0e:46:7f:91:8e:11:16:80:da:9d:15:47:87:25:ca:
        96:ae:c5:aa:6f:30:81:39:a7:f1:fa:35:cf:4c:d7:7e:0d:e2:
        e9:6e:49:6b:b3:47:aa:2e:66:b9:df:81:d8:70:7b:53:5d:43:
        da:2f:e7:c4:13:7a:8c:d1:c6:09:14:39:1c:df:fa:e3:98:2a:
        8e:d9:64:cc:ac:7a:bd:48:56:3e:54:55:55:5e:61:71:5a:c3:
        ef:ab:df:c4:5d:af:bd:5c:61:32:fb:4e:2f:73:b9:bd:41:b7:
        05:dd:05:c3:d0:09:67:f5:00:65:c8:b9:a6:b1:18:da:b6:f8:
        d4:2d:be:1b:0f:0b:7f:76:3a:d7:dd:bc:37:ed:dc:a4:46:02:
        26:c0:2d:06:55:16:6f:ff:4a:d4:de:d5:9e:8e:94:9f:e9:50:
        6d:97:ba:b2
```

По результатам проверки мы видим, что запрос получился первой версии, без дополнительных расширений X509v3, но, очень часто, требуются сертификаты третьей версии, хотя бы для того, чтобы указать альтернативные имена или адреса машины.

Пример: Создадим запрос на сертификат 3 версии:

1. Нужно создать дополнительный файл конфигурации:

```
root@s10:/etc/ssl# cat s10_v3.ext
[ v3_req ]
subjectAltName = @alt_names
[alt_names]
IP.1=10.1.1.5
IP.2=10.2.2.1
DNS.1=s10.class.itcloud
DNS.2=www.class.itcloud
DNS.3=s10
```

2. Желательно поправить конфигурацию в `/etc/ssl/openssl.cnf`:

```
root@s10:/etc/ssl# grep Name_default /etc/ssl/openssl.cnf
countryName_default             = RU
stateOrProvinceName_default     = Свердловская область
localityName_default            = Екатеринбург
0.organizationName_default      = АйТи Клауд
#1.organizationName_default     = World Wide Web Pty Ltd
organizationalUnitName_default      = Класс SecurL
```

3. Сделать новый запрос:

```
root@s10:/etc/ssl# openssl req -new -key private/s10.key -out s10.csr
-extensions v3_req -reqexts v3_req -config <(cat openssl.cnf s10_v3.ext )
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [RU]:
State or Province Name (full name) [Свердловская область]:
Locality Name (eg, city) [Екатеринбург]:
Organization Name (eg, company) [АйТи Клауд]:
Organizational Unit Name (eg, section) [Класс SecurL]:
Common Name (e.g. server FQDN or YOUR name) []:s10.class.itcloud
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

```
root@s10:/etc/ssl# openssl req -noout -text -in s10.csr
Certificate Request:
    Data:
        Version: 1 (0x0)
        Subject: C = RU, ST =
\C3\90\C2\A1\C3\90\C2\B2\C3\90\C2\B5\C3\91\C2\80\C3\90\C2\B4\C3\90\C2\BB\C3\90\C
2\BE\C3\90\C2\B2\C3\91\C2\81\C3\90\C2\BA\C3\90\C2\B0\C3\91\C2\8F
\C3\90\C2\BE\C3\90\C2\B1\C3\90\C2\BB\C3\90\C2\B0\C3\91\C2\81\C3\91\C2\82\C3\91\C
2\8C, L =
\C3\90\C2\95\C3\90\C2\BA\C3\90\C2\B0\C3\91\C2\82\C3\90\C2\B5\C3\91\C2\80\C3\90\C
2\B8\C3\90\C2\BD\C3\90\C2\B1\C3\91\C2\83\C3\91\C2\80\C3\90\C2\B3, O =
\C3\90\C2\90\C3\90\C2\B9\C3\90\C2\A2\C3\90\C2\B8
\C3\90\C2\9A\C3\90\C2\BB\C3\90\C2\B0\C3\91\C2\83\C3\90\C2\B4, OU =
\C3\90\C2\9A\C3\90\C2\BB\C3\90\C2\B0\C3\91\C2\81\C3\91\C2\81 SecurL, CN =
s10.class.itcloud
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:ae:35:d5:4d:98:29:66:69:a7:da:09:2b:ea:2f:
                    0e:86:08:6a:56:8c:f6:d6:d9:66:76:92:59:81:fd:
                    29:a0:7b:98:74:69:91:36:33:3f:1b:73:a8:a5:e5:
                    e9:3b:2a:12:b6:e4:7c:1a:82:14:de:32:1a:c3:d3:
                    8a:f5:4e:92:3e:68:b8:c0:f0:8f:4b:23:4a:95:d9:
                    2e:61:0b:8f:40:94:4c:1f:2e:95:8b:63:e7:fd:1b:
                    49:c6:cf:0d:62:ef:42:0d:65:5e:c4:4d:d5:85:c5:
                    f0:86:6a:12:bc:7c:bd:aa:10:ae:ec:6f:00:64:84:
                    ea:f0:e0:ab:b4:f2:fc:d3:8a:c0:c1:9b:a0:8b:cf:
                    b2:7b:14:c9:6f:b0:ab:22:ae:ab:0b:a3:6d:c0:af:
                    f8:cc:00:e8:66:aa:aa:ed:6d:b5:d7:66:42:08:4c:
                    bd:33:72:b5:fb:51:cf:0c:78:88:bf:3b:a4:90:4e:
                    cf:a6:4b:10:18:e7:51:5f:17:80:13:e9:f8:cd:77:
                    44:78:09:ef:f6:1d:d6:29:1e:d8:a2:df:ea:20:e5:
                    96:57:37:ea:f6:54:3b:01:b6:a1:18:93:a3:81:9b:
                    f4:ca:72:59:5d:5a:69:d8:34:88:4b:14:ab:07:49:
                    2a:b2:dd:97:79:b5:26:ce:5f:6f:59:02:db:f0:99:
                    75:f9
                Exponent: 65537 (0x10001)
        Attributes:
            Requested Extensions:
                X509v3 Basic Constraints:
                    CA:FALSE
                X509v3 Key Usage:
                    Digital Signature, Non Repudiation, Key Encipherment
                X509v3 Subject Alternative Name:
                    IP Address:10.1.1.5, IP Address:10.2.2.1,
DNS:s10.class.itcloud, DNS:www.class.itcloud, DNS:s10
    Signature Algorithm: sha256WithRSAEncryption
    Signature Value:
        04:56:7a:c2:3e:a8:32:cb:7d:8c:0d:45:88:3a:85:3e:55:85:
        d4:cf:35:81:94:d8:5b:be:4d:26:9e:0b:a6:2e:4c:02:23:e7:
        35:d7:9e:5a:57:25:d6:b1:8d:6d:1f:93:19:65:6c:f2:7c:85:
        52:cf:c4:54:49:77:da:14:09:e0:af:26:6f:18:83:0b:d0:8a:
        be:33:d7:9c:5f:dd:f2:9c:c8:ef:b3:fa:18:b8:14:46:c3:30:
        e1:4b:12:d6:b5:a6:2f:f9:e0:e1:12:60:d5:f9:c1:5b:4b:43:
        bb:17:ff:4b:1b:96:33:9d:13:d8:54:bf:4b:68:a3:73:57:4a:
        64:b9:a3:75:51:e8:61:ec:3c:e2:0e:dc:5b:38:03:09:20:ea:
        5a:d0:fb:8b:75:c4:d8:c6:67:85:09:ff:e2:76:e8:4d:c6:3d:
        47:21:eb:4a:ee:59:56:57:16:5d:c8:90:78:c5:56:11:a4:bc:
```

```
        54:77:cf:00:a2:4e:d0:01:9f:1a:99:f0:bc:6c:33:ef:18:95:
        17:b9:6e:58:bd:cd:52:06:1a:81:0c:55:9a:14:df:ac:5c:ea:
        2e:77:75:41:a6:ff:19:db:b5:6a:ae:08:b6:a0:46:65:18:47:
        1c:21:c0:7a:13:e3:84:90:9e:7e:f7:09:a3:7a:07:2d:39:ea:
        e5:ca:4f:5b
```

В качестве альтернативы созданию дополнительного файла конфигурации, можно использовать следующую команду:

```
root@s10:/srv/pki# SAN='DNS:s10,DNS:s10.class.itcloud,IP:10.1.1.5,IP:10.2.2.1'
openssl req -new -key /etc/ssl/private/s10.key -out /tmp/s10.csr -extensions
v3_req -reqexts v3_req -config <(cat /etc/ssl/openssl.cnf; echo -e '[ v3_req ]\n
subjectAltName = ${ENV::SAN}\n' )
```

4. Копируем запрос в CA:

```
root@s10:/etc/ssl# cp s10.csr /srv/pki/
```

Агентство сертификации проверяет информацию об источнике запроса на подпись сертификата и подписывает (в случае позитивного результата проверки) сертификат.

Команда, используемая для подписи сертификата:

```
root@s10:/srv/pki# openssl ca -in s10.csr -out CA/certs/s10.crt -create_serial
-days 3653 -config /srv/pki/openssl.cnf
Using configuration from /srv/pki/openssl.cnf
Enter pass phrase for /srv/pki/CA/private/ca.key:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 3 (0x3)
        Validity
            Not Before: Feb  7 06:07:13 2025 GMT
            Not After : Feb  8 06:07:13 2035 GMT
        Subject:
            countryName               = RU
            stateOrProvinceName       =
\D0\A1\D0\B2\D0\B5\D1\80\D0\B4\D0\BB\D0\BE\D0\B2\D1\81\D0\BA\D0\B0\D1\8F
\D0\BE\D0\B1\D0\BB\D0\B0\D1\81\D1\82\D1\8C
            organizationName          = \D0\90\D0\B9\D0\A2\D0\B8
\D0\9A\D0\BB\D0\B0\D1\83\D0\B4
            organizationalUnitName    = \D0\9A\D0\BB\D0\B0\D1\81\D1\81 SecurL
            commonName                = s10.class.itcloud
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                02:E8:52:40:07:2D:1A:36:CC:E5:53:4B:D9:FE:79:F5:AD:A4:7F:DC
            X509v3 Authority Key Identifier:
                FD:A4:25:32:14:F0:0A:3A:E9:69:B1:2F:F3:3A:86:DC:1C:09:38:1F
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Key Usage:
                Digital Signature, Non Repudiation, Key Encipherment
            X509v3 Subject Alternative Name:
                IP Address:10.1.1.5, IP Address:10.2.2.1, DNS:s10.class.itcloud,
DNS:www.class.itcloud, DNS:s10
Certificate is to be certified until Feb  8 06:07:13 2035 GMT (3653 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
```

```
Write out database with 1 new entries
Database updated
```

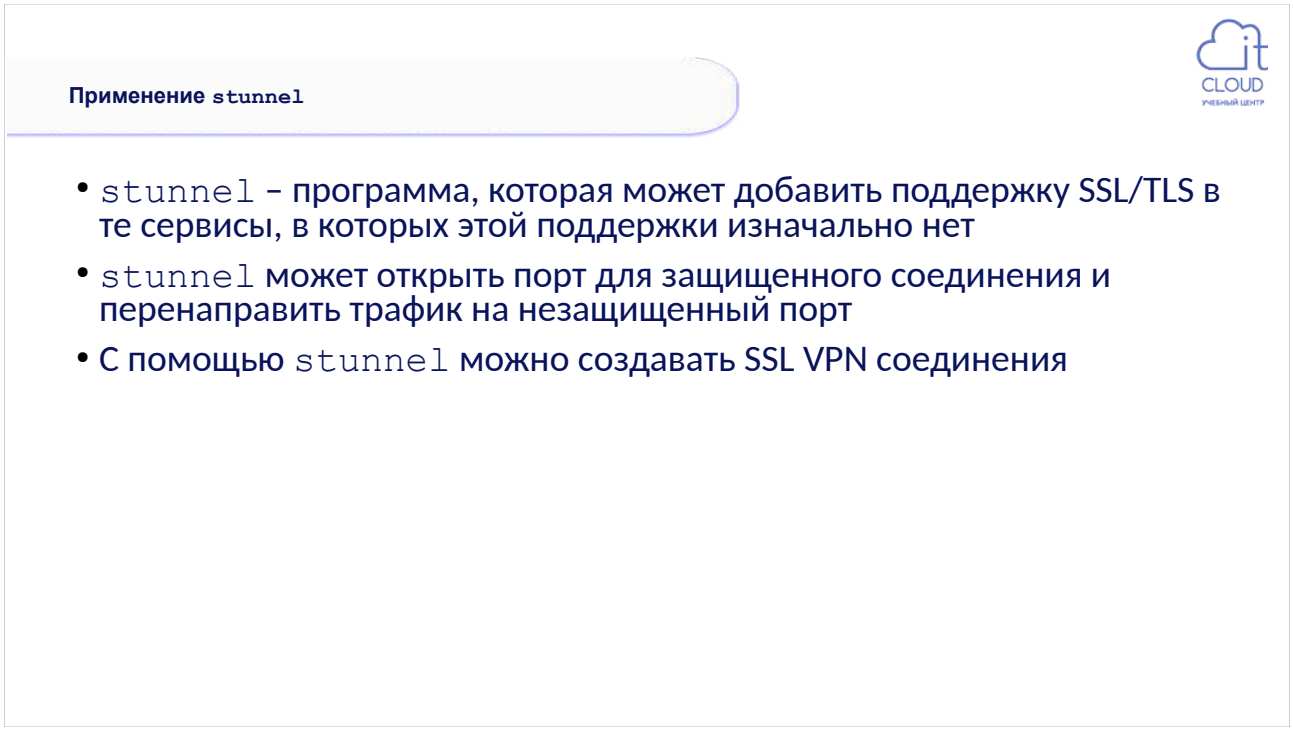
Внимательно проверяйте подписываемый сертификат. Особое внимание уделяйте разделу `X509v3 Basic Constraints: CA:FALSE`.

Проверить результаты полученного подписанного сертификата можно с помощью команды:

```
openssl x509 -noout -text -in  CA/certs/s10.crt
```

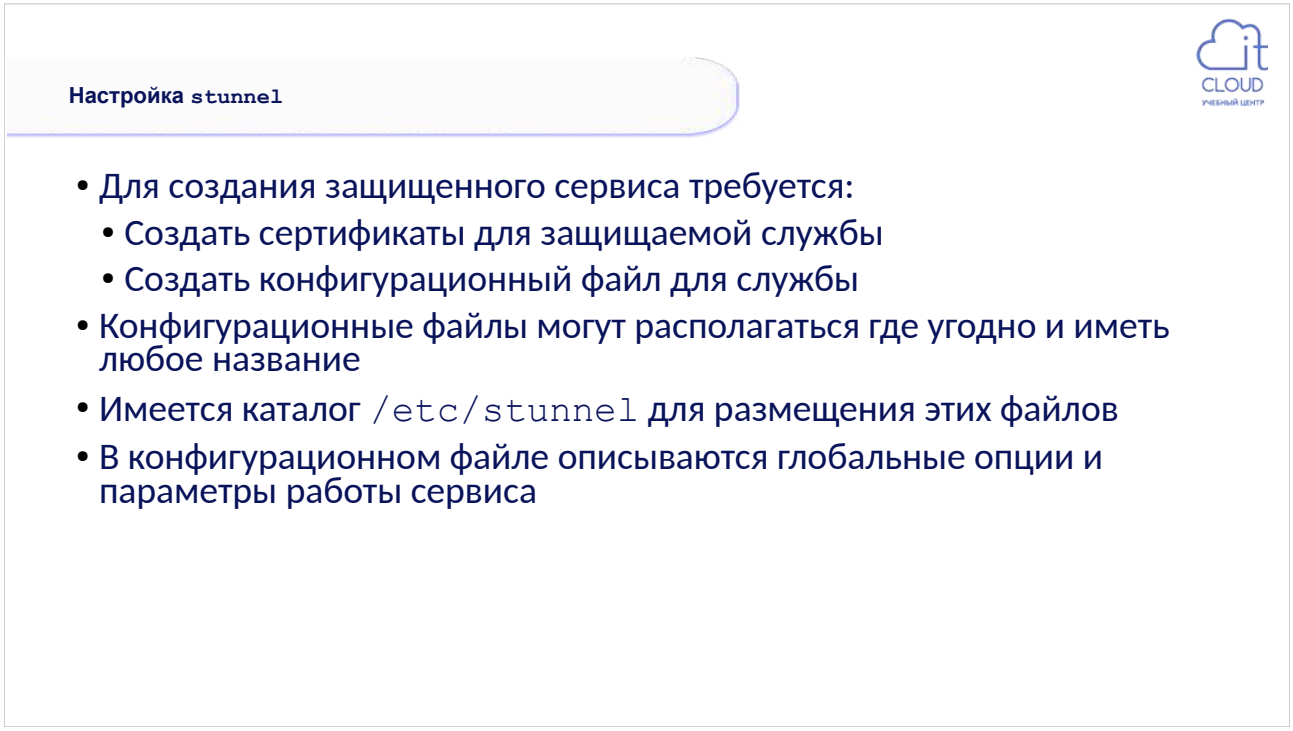
Далее подписанный сертификат передается тому, кто его запросил.

9.5 Использование stunnel.



Stunnel предназначена для защиты сетевого трафика тех приложений, в которых такой защиты не предусмотрено. Обычно это службы типа POP или IMAP, либо туннелирование PPP через сеть.

Stunnel программа врапер, которая слушает порт для защищенного взаимодействия, а затем перенаправляет трафик на обычный порт службы, которую stunnel защищает.



Первое что необходимо для настройки защищаемого сервиса это установить пакет stunnel4 и создать сертификаты для службы и, возможно, клиента. Как создавать сертификаты описано выше.

Следующий этап создание конфигурационного файла защищенной службы.

Пример:

Файл конфигурации сервера.

```
root@sl0:stunnel# pwd
/etc/stunnel
root@sl0:stunnel# cat pppsrv.conf
cert = /etc/ssl/certs/ppp_server.crt
key = /etc/ssl/private/ppp_server.key
pid = /tmp/pppstunnel.pid
socket = l:TCP_NODELAY=1
socket = r:TCP_NODELAY=1
debug = 7
output = /var/log/pppstunnel.log
client = no
verify = 2
CAfile = /usr/local/share/ca-certificates/myca/ca.crt

[ppp]
 client = no
 accept = 31234
 exec = /usr/sbin/pppd
 execargs = local noauth 10.1.2.1:10.1.2.2
 pty = yes
```

Сервер запускается службой stunnel4:

```
root@s10:/etc/ssl# systemctl restart stunnel4.service
```

```
root@s10:/etc/ssl# systemctl status stunnel4.service
● stunnel4.service - LSB: Start or stop stunnel 4.x (TLS tunnel for network
daemons)
     Loaded: loaded (/etc/init.d/stunnel4; generated)
     Active: active (running) since Fri 2025-02-07 11:45:45 +05; 7s ago
       Docs: man:systemd-sysv-generator(8)
    Process: 9502 ExecStart=/etc/init.d/stunnel4 start (code=exited,
status=0/SUCCE>
      Tasks: 2 (limit: 2284)
     Memory: 2.2M
        CPU: 133ms
     CGroup: /system.slice/stunnel4.service
             └─9518 /usr/bin/stunnel4 /etc/stunnel/pppsrv.conf

фев 07 11:45:45 s10 stunnel[9516]: LOG7[ui]: Listening file descriptor created
(FD=>
фев 07 11:45:45 s10 stunnel[9516]: LOG7[ui]: Setting accept socket options
(FD=10)
фев 07 11:45:45 s10 stunnel[9516]: LOG7[ui]: Option SO_REUSEADDR set on accept
sock>
фев 07 11:45:45 s10 stunnel[9516]: LOG5[ui]: Binding service [ppp] to :::31234:
Add>
фев 07 11:45:45 s10 stunnel[9518]: LOG7[main]: Created pid file
/tmp/pppstunnel.pid
фев 07 11:45:45 s10 stunnel[9518]: LOG6[main]: Accepting new connections
фев 07 11:45:45 s10 stunnel[9518]: LOG7[cron]: Cron thread initialized
фев 07 11:45:45 s10 stunnel[9518]: LOG6[cron]: Executing cron jobs
фев 07 11:45:45 s10 stunnel[9518]: LOG6[cron]: Cron jobs completed in 0 seconds
фев 07 11:45:45 s10 stunnel[9518]: LOG7[cron]: Waiting 86400 seconds
root@s10:/etc/ssl# ss -ltnp sport 31234
State    Recv-Q   Send-Q     Local Address:Port       Peer Address:Port
Process
LISTEN   0        4096             0.0.0.0:31234           0.0.0.0:*
users:(("stunnel4",pid=9518,fd=9))
```

Файл конфигурации клиента.

```
root@sl0:~# pwd
/root
root@sl0:~# cat pppclnt.conf
cert = /etc/ssl/certs/client.crt
key = /etc/ssl/private/client.key
pid = /tmp/pppstunnel.pid
socket = l:TCP_NODELAY=1
socket = r:TCP_NODELAY=1
debug = 0
output = /var/log/pppstunnel.log
client = yes
verify = 2
CAfile = /usr/local/share/ca-certificates/myca/ca.crt
foreground = no
connect = 10.2.2.1:31234
```

Клиент запускается вручную командой `pppd`:

```
root@client:~# pppd passive updetach noauth pty "stunnel pppclnt.conf"
Using interface ppp0
Connect: ppp0 <--> /dev/pts/3
```

```
Deflate (15) compression enabled
local  LL address fe80::a03f:1684:789b:66de
remote LL address fe80::b945:05fd:f776:8c26

root@client:~# ip -4 -br ad show dev ppp0
ppp0             UNKNOWN        192.168.111.2 peer 192.168.111.1/32

root@client:~# ping -c1 192.168.111.1
PING 192.168.111.1 (192.168.111.1) 56(84) bytes of data.
64 bytes from 192.168.111.1: icmp_seq=1 ttl=64 time=11.1 ms

--- 192.168.111.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 11.051/11.051/11.051/0.000 ms
```

Это то что передается по сети при выполнении на клиенте команды `ping -c1 192.168.111.1`

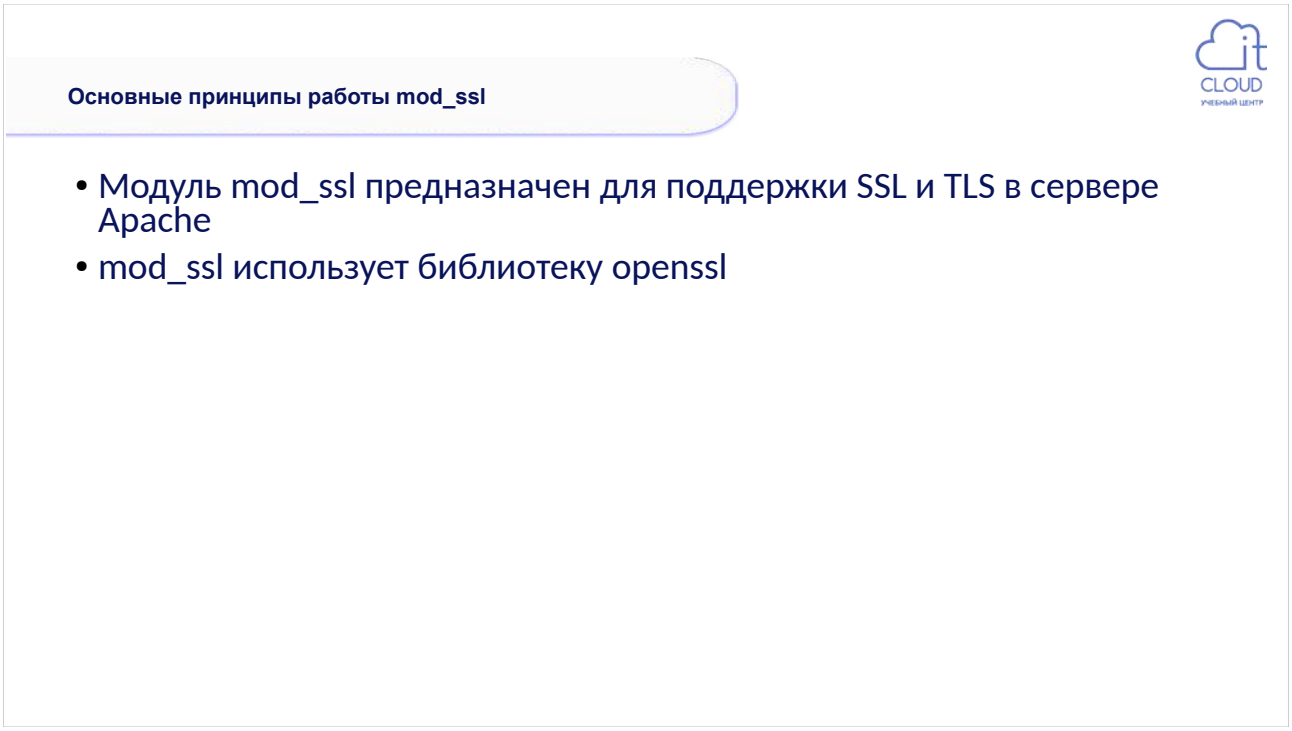
```
root@s10:~# tcpdump -xxx -i enp0s8 tcp port 31234
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s8, link-type EN10MB (Ethernet), snapshot length 262144 bytes
12:26:37.108288 IP 10.2.2.2.35972 > 10.2.2.1.31234: Flags [P.], seq
2775096202:2775096259, ack 3808170654, win 501, options [nop,nop,TS val
2708951398 ecr 1985291697], length 57
      0x0000:  0800 27ff 6d87 0800 2746 dcff 0800 4500
      0x0010:  006d c8d4 4000 4006 59b0 0a02 0202 0a02
      0x0020:  0201 8c84 7a02 a568 9b8a e2fc 129e 8018
      0x0030:  01f5 136d 0000 0101 080a a177 5166 7655
      0x0040:  25b1 1703 0300 347d 297f b462 8e65 1e14
      0x0050:  c2ac 033d aafa b76e 7cb6 b3c5 d663 eb84
      0x0060:  ca55 4d51 ab30 cb71 378e 23c3 c00e 2c5a
      0x0070:  1fcf 9bae 5530 001d 6eb7 34
12:26:37.112053 IP 10.2.2.1.31234 > 10.2.2.2.35972: Flags [P.], seq 1:58, ack
57, win 501, options [nop,nop,TS val 1985303859 ecr 2708951398], length 57
      0x0000:  0800 2746 dcff 0800 27ff 6d87 0800 4500
      0x0010:  006d 5e6d 4000 4006 c417 0a02 0201 0a02
      0x0020:  0202 7a02 8c84 e2fc 129e a568 9bc3 8018
      0x0030:  01f5 1866 0000 0101 080a 7655 5533 a177
      0x0040:  5166 1703 0300 3494 81f8 17ea b7c9 f5df
      0x0050:  19c1 51db 21e7 7a76 bd6b 010d 80fe 5264
      0x0060:  1955 5381 aab1 c89d 7fa0 9af6 03ce 4be1
      0x0070:  7e6c 504c ebc2 dac2 f610 af
12:26:37.114950 IP 10.2.2.2.35972 > 10.2.2.1.31234: Flags [.], ack 58, win 501,
options [nop,nop,TS val 2708951405 ecr 1985303859], length 0
      0x0000:  0800 27ff 6d87 0800 2746 dcff 0800 4500
      0x0010:  0034 c8d5 4000 4006 59e8 0a02 0202 0a02
      0x0020:  0201 8c84 7a02 a568 9bc3 e2fc 12d7 8010
      0x0030:  01f5 60cd 0000 0101 080a a177 516d 7655
      0x0040:  5533
```

А это то что внутри туннеля:

```
root@s10:~# tcpdump -xxx -i ppp0 icmp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on ppp0, link-type LINUX_SLL (Linux cooked v1), snapshot length 262144
bytes
```

```
12:27:34.994365 IP 192.168.111.2 > 192.168.111.1: ICMP echo request, id 23305,
seq 1, length 64
      0x0000:  0000 0200 0000 0000 0000 0000 0000 0800
      0x0010:  4500 0054 10fe 4000 4001 ca56 c0a8 6f02
      0x0020:  c0a8 6f01 0800 2768 5b09 0001 e6b5 a567
      0x0030:  0000 0000 1c9d 0e00 0000 0000 1011 1213
      0x0040:  1415 1617 1819 1a1b 1c1d 1e1f 2021 2223
      0x0050:  2425 2627 2829 2a2b 2c2d 2e2f 3031 3233
      0x0060:  3435 3637
12:27:34.994468 IP 192.168.111.1 > 192.168.111.2: ICMP echo reply, id 23305, seq
1, length 64
      0x0000:  0004 0200 0000 0000 0000 0000 0000 0800
      0x0010:  4500 0054 393c 0000 4001 e218 c0a8 6f01
      0x0020:  c0a8 6f02 0000 2f68 5b09 0001 e6b5 a567
      0x0030:  0000 0000 1c9d 0e00 0000 0000 1011 1213
      0x0040:  1415 1617 1819 1a1b 1c1d 1e1f 2021 2223
      0x0050:  2425 2627 2829 2a2b 2c2d 2e2f 3031 3233
      0x0060:  3435 3637
```

9.6 Использование сертификатов на примере Apache.



Модуль mod_ssl предназначен для использования совместно с сервером Apache. Он обеспечивает шифрование информации посредством протоколов SSL (Secure Socket Layer v2/v3) и TLS (Transport Layer Security v1), используя реализующие эти протоколы библиотеку OpenSSL.

Библиотека OpenSSL предоставляет возможности несимметричной криптографии, то есть информация шифруется приватным ключом, а расшифровывается публичным и наоборот.

В силу невысокой скорости работы несимметричных алгоритмов шифрования, при передаче информации используются симметричные алгоритмы шифрования, а несимметричные используются при аутентификации.

Для исключения возможности подделки сообщения или его искажения можно применять дайджесты сообщений (message digest), представляющие собой аналог контрольных сумм файлов, вычисляемые на основе односторонних хеш-функций.

Дайджест сообщения является числом с фиксированным числом разрядов, зависящим от каждого байта исходного сообщения.

Дайджест должен обеспечивать крайнюю затруднительность подбора иного сообщения, чем оригинальное, с таким же дайджестом.

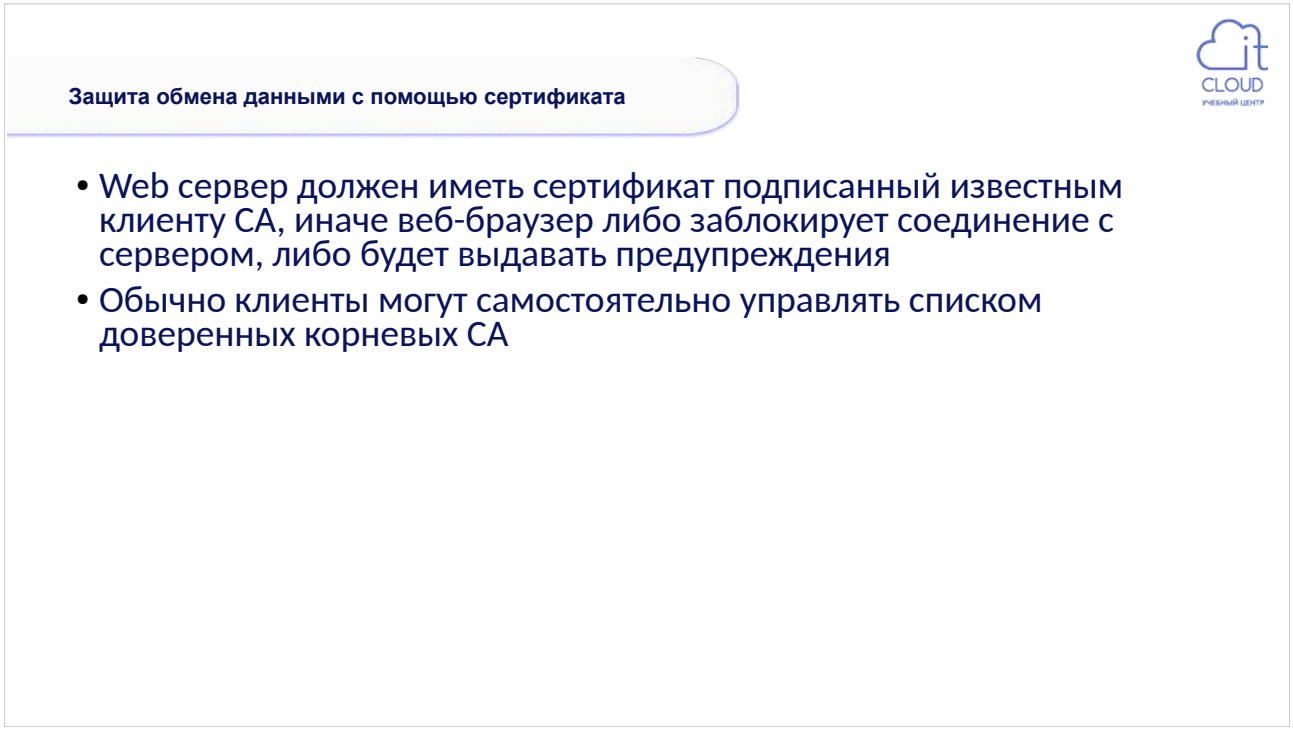
Дайджест сообщения должен быть послан получателю либо по защищенному каналу, либо он должен быть защищен сам. Один из способов защиты дайджеста от подделки заключается во вложении его в цифровую подпись (digital signature).

Цифровая подпись предназначена для аутентификации сообщения, то есть для исключения возможности подмены отправителя. Цифровая подпись составляется из зашифрованного дайджеста сообщения и дополнительной информации.

Дайджест сообщения шифруется приватным ключом отправителя, что и гарантирует аутентичность сообщения, так как приватным ключом обладает только настоящий отправитель.

Дайджест может быть расшифрован любым обладателем публичного ключа отправителя.

Каждая цифровая подпись содержит в себе уникальный последовательный номер для исключения ее повторного использования. Это предотвращает возможность подлога сообщения, а также отказа от признания отправки сообщения.



Для обеспечения аутентификации сервера, предоставляющего клиенту информацию, необходимо, чтобы сертификат сервера, отправляемый клиенту был удостоверен независимым и хорошо известным агентством сертификации CA (Certification Authority).

Сертификат сервера подписан приватным ключом агентства сертификации, поэтому клиент должен иметь публичный ключ агентства для проверки подлинности сертификата сервера.

Если у клиента нет сертификата CA, который подписал сертификат сервера, то клиенту необходимо добавить этот CA в список доверенных CA.

Программы просмотра WEB, поддерживающие SSL, обычно имеют заранее установленный набор публичных ключей широко известных агентств сертификации (например, VeriSign).

Если подлинность сертификата, отправленного сервером клиенту, установлена с помощью имеющегося у клиента публичного ключа агентства сертификации, то клиент извлекает из сертификата публичный ключ сервера и, таким образом, он может дешифровать информацию, отправляемую сервером.

Несмотря на то, что подлинность сертификата установлена, этот факт не гарантирует подлинности отправившего его сервера, так как этот сертификат может быть отправлен с другого сервера.

Для проверки подлинности сервера необходимо проверить обладает ли отправивший сертификат сервер приватным ключом, соответствующим публичному ключу сервера в сертификате, подписанном агентством сертификации.

Проверка подлинности сервера осуществляется с помощью отправки клиенту тестового сообщения. Сервер вычисляет контрольную сумму тестового сообщения, шифрует ее своим приватным ключом и также отправляет ее клиенту.

Таким образом клиент имеет:

- сертификат сервера, подписанный агентством сертификации;
- публичный ключ сервера, извлеченный из сертификата;
- тестовое сообщение от сервера;
- зашифрованную контрольную сумму идентифицирующего сообщения.

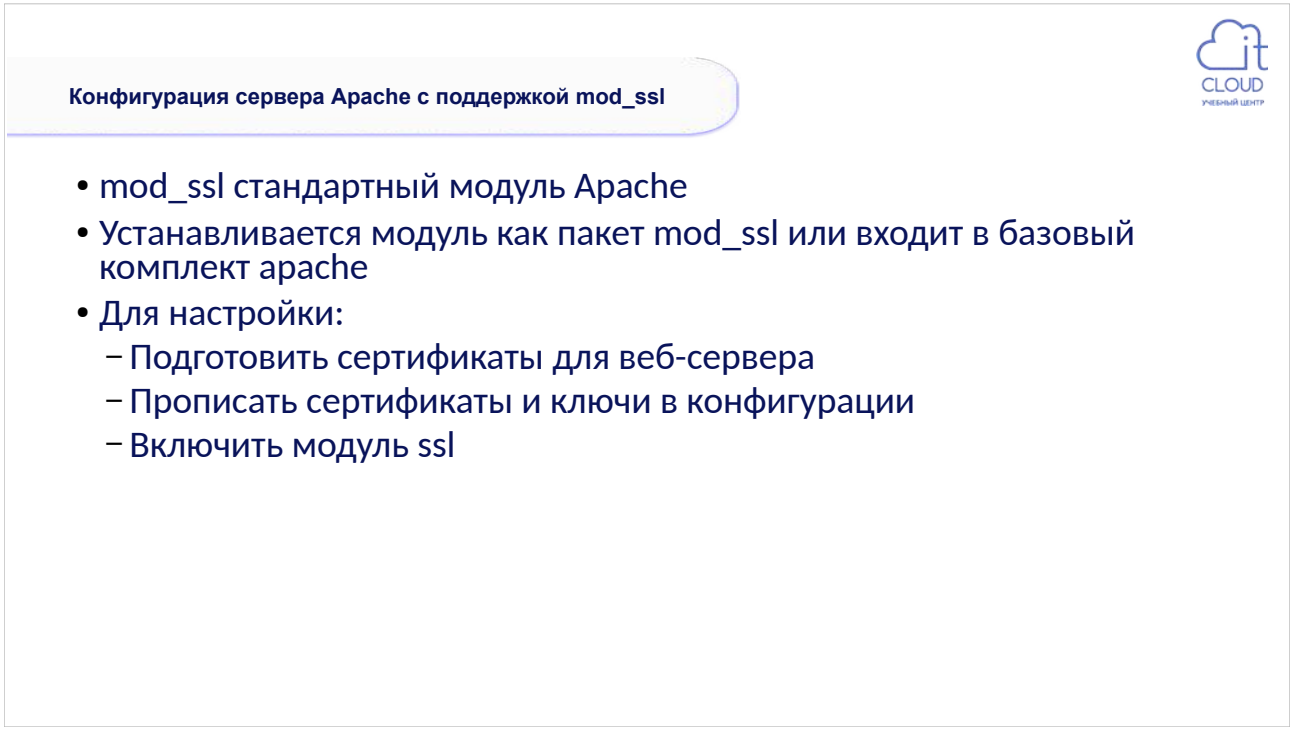
Клиент дешифрует контрольную сумму идентифицирующего сообщения с помощью публичного ключа сервера, извлеченного из сертификата.

Кроме этого, клиент самостоятельно вычисляет контрольную сумму тестового сообщения, которая сравнивается с дешифрованной публичным ключом сервера контрольной суммой. Если полученная и дешифрованная контрольная сумма совпадает с вычисленной, то подлинность сервера считается установленной.

Для установки сеанса шифрованной связи клиент отправляет серверу зашифрованный с помощью публичного ключа сервера ключ симметричного шифрования.

Для защиты сеанса шифрованной связи от искажения передаваемых пакетов, зашифрованных симметричным ключом, используется код аутентификации сообщений (Message Authentication Code - MAC), являющийся вычисляемым с помощью симметричного ключа фрагмент данных.

MAC не может быть вычислен без знания симметричного ключа.



Программа mod_ssl является стандартным модулем сервера Apache. Исходный код его находится в подкаталоге `src/modules/ssl/` дерева файлов исходного кода сервера Apache.

В RedHat подобных системах модуль устанавливается в пакете mod_ssl. В Debian устанавливается по умолчанию, но не включается.

Для настройки SSL рекомендуется создать цепочку сертификатов:

```
root@s10:~# mkdir /etc/apache2/ssl.crt
root@s10:~# cat /etc/ssl/certs/s10.crt
/usr/local/share/ca-certificates/myca/ca.crt  > /etc/apache2/ssl.crt/s10-ca.crt
```

В файле конфигурации `/etc/apache2/sites-available/default-ssl.conf` необходимо указать сертификаты:

```
root@s10:~# grep SSLCertificate /etc/apache2/sites-available/default-ssl.conf |
grep -v '#'
      SSLCertificateFile      /etc/ssl/certs/s10.crt
      SSLCertificateKeyFile   /etc/ssl/private/s10.key
      SSLCertificateChainFile /etc/apache2/ssl.crt/s10-ca.crt
```

Включить модуль ssl:

```
root@s10:~# a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create
self-signed certificates.
To activate the new configuration, you need to run:
  systemctl restart apache2
```

Включить сайт SSL:

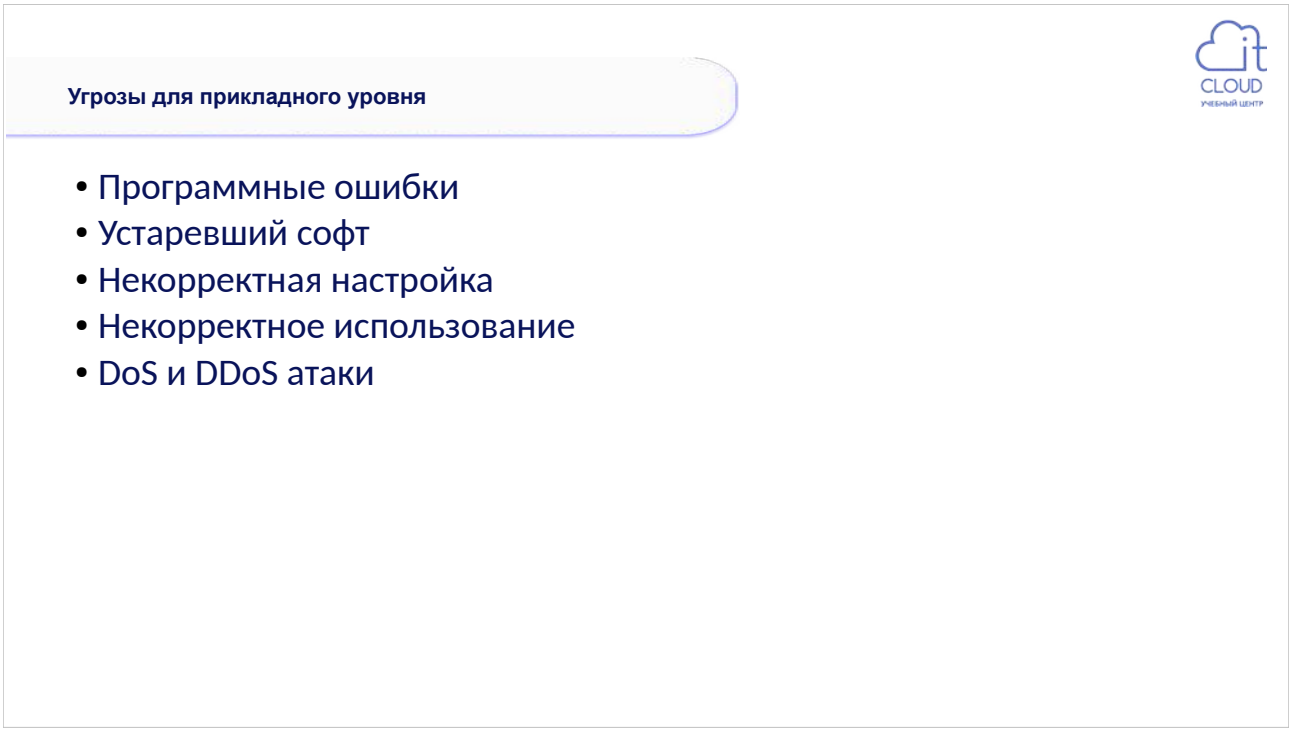
```
root@s10:~# a2ensite default-ssl.conf
Enabling site default-ssl.
To activate the new configuration, you need to run:
  systemctl reload apache2
```

Перезапустить веб-сервер и проверить, что он функционирует после перезапуска.

```
root@s10:~# systemctl restart apache2
root@s10:~# systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; preset:
enabled)
     Active: active (running) since Fri 2025-02-07 12:50:11 +05; 5s ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 10458 ExecStart=/usr/sbin/apachectl start (code=exited,
status=0/SUCCE>
   Main PID: 10463 (apache2)
      Tasks: 55 (limit: 2284)
     Memory: 16.0M
        CPU: 137ms
     CGroup: /system.slice/apache2.service
             ├─10463 /usr/sbin/apache2 -k start
             ├─10464 /usr/sbin/apache2 -k start
             └─10465 /usr/sbin/apache2 -k start

фев 07 12:50:11 s10 systemd[1]: Starting apache2.service - The Apache HTTP
Server...
фев 07 12:50:11 s10 apachectl[10462]: AH00558: apache2: Could not reliably
determin>
фев 07 12:50:11 s10 systemd[1]: Started apache2.service - The Apache HTTP
Server.
root@s10:~# ss -tlnp sport 443
State    Recv-Q   Send-Q     Local Address:Port       Peer Address:Port
Process
LISTEN   0        511                    *:443                   *:*
users:(("apache2",pid=10465,fd=6),("apache2",pid=10464,fd=6),
("apache2",pid=10463,fd=6))
```

Глава 10. Безопасность уровня приложений.

10.1 Особенности защиты прикладных сервисов в UNIX-системах на примере сервера Apache+MySQL+PHP.

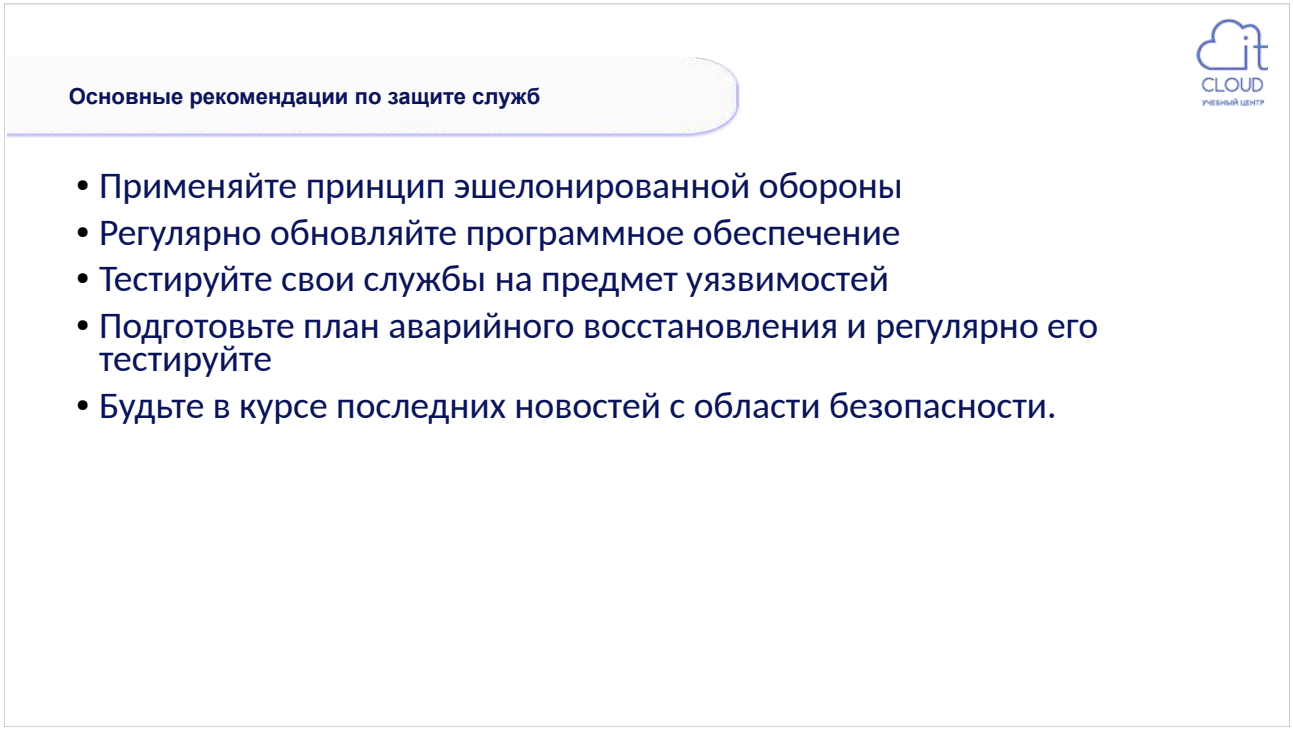


Проблемы с безопасностью прикладных служб открывают бреши в безопасности всей системы.

При использовании программного обеспечения возникает множество угроз. Некоторые из этих угроз описаны ниже.

- Программные ошибки. В идеальном мире программисты создавая свои продукты должны всесторонне проверить программу, выявить все ошибки и их устранить. Но ошибки могут быть не очевидны или у программиста нет времени и еще множество причин почему в программах возникают ошибки. Наличие ошибки еще не означает непосредственной угрозы, ошибку еще необходимо выявить, чтобы иметь возможность целенаправленного ее использования.
- Устаревший софт. С течением времени в софте находят имеющиеся там ошибки и их устраняют. Как правило выявление ошибки означает серьезную угрозу для безопасности, т. к. выявленная ошибка подразумевает возможность ее использования злоумышленником.
- Некорректная настройка. Администратор для использования некоторых программных продуктов должен произвести его настройку. Очень часто настройка происходит по принципу «лишь бы работало». Такой подход совершенно не приемлем с точки зрения обеспечения безопасности системы.

- Некорректное использование. Какой бы безопасной программа ни была, если ее неправильно использовать (случайно или целенаправленно) некорректно, то может возникнуть угроза безопасности.
- DoS (Denial of Servise) или DDoS (Distributed Denial of Service) атаки. Целенаправленно выведение из строя сервисов с целью остановки обслуживания клиентов или получения доступа к системе.



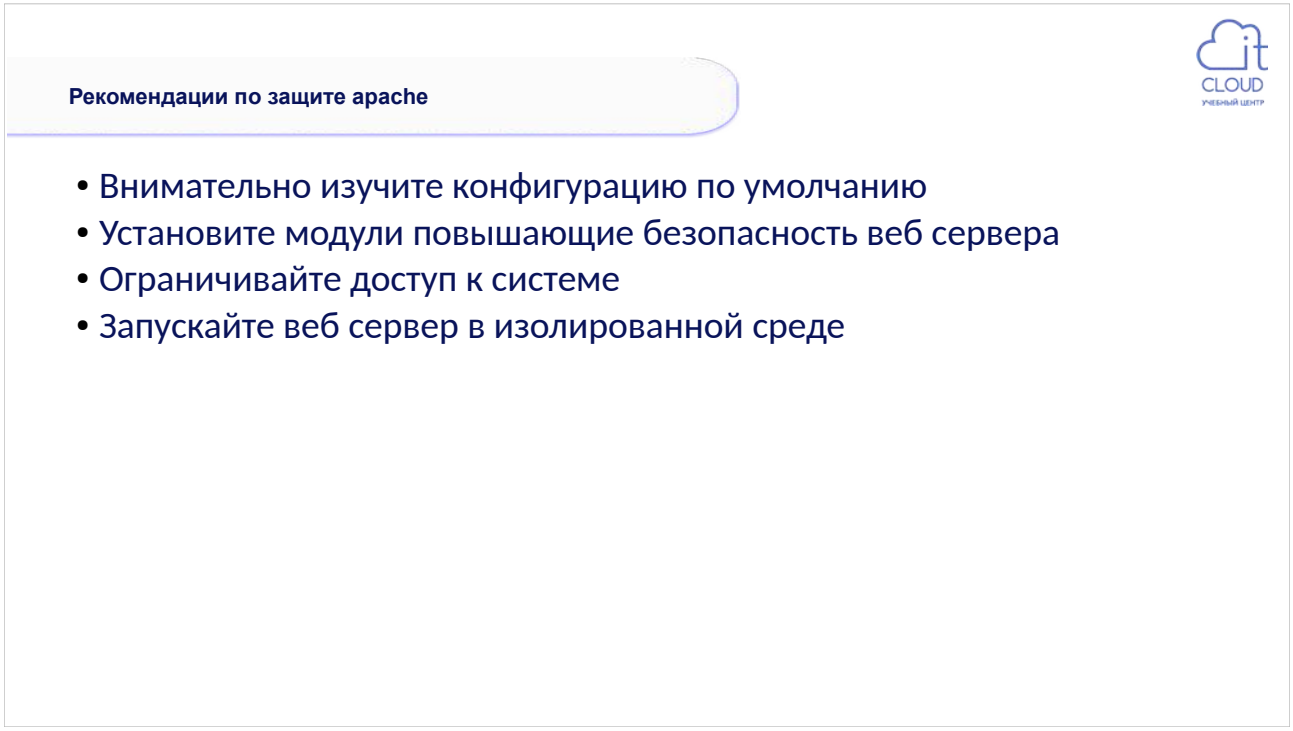
Эшелонированная оборона один из главных принципов построения безопасности. Рассмотрим пример в веб сервером. Вы можете предоставить доступ пользователям непосредственно к серверу apache. Но такой подход может привести к перегрузке сервера. Одно из решений этой проблемы — установить сервер nginx в качестве кэширующего прокси сервера.

Обновление системы одна из главных задач администратора. Но важно не просто ставить все возможные обновления сразу же как они были опубликованы, он еще и протестировать как это обновление повлияет на работу обновляемых служб.

Самостоятельное тестирование работы сервисов даст вам ответ на вопрос: как работают мои программы? Тестирование должно проводиться в отдельной, но репрезентативной среде. Тестирование рабочей службы может вызвать предсказуемые последствия.

Какой бы надежной ни была система, всегда есть вероятность выхода ее из строя или поломки отдельного компонента. Продуманный и протестированный план аварийного восстановления поможет вам восстановить работоспособность системы в кратчайшие сроки. Продумайте так же способ аварийного доступа к системе, в случае нарушения работы обычного способа работы.

Знания и информированность лишними не бывают. Будьте в курсе последних событий и новостей безопасности.



Если вы установили apache из пакетов, то его конфигурация по умолчанию может быть излишне либеральной. Многие программные продукты имеют веб интерфейс, который автоматически активируется в виде дополнительного конфигурационного файла apache. С точки зрения безопасности правильным подходом будет принцип наименьших привилегий или функционала.

- Конфигурация сервера находится в файле `httpd.conf`, который в RedHat подобных системах располагается в каталоге `/etc/httpd/conf`. Помимо основного конфигурационного файла могут быть подключены дополнительные файлы конфигурации, с помощью директив Include

Пример: В данном примере к конфигурации подключены дополнительные файлы из каталогов conf.modules.d и conf.d.

```
$ grep ^Include /etc/httpd/conf/httpd.conf
Include conf.modules.d/*.conf
IncludeOptional conf.d/*.conf
```

- Внимательно изучите все загружаемые модули. Действительно ли они вам требуются.
- Не включайте опций, позволяющих получить сведения о системе.
- Не используйте страницы ошибок по умолчанию.
- Веб сервер должен работать от имени специально созданного непривилегированного пользователя.

Пример:

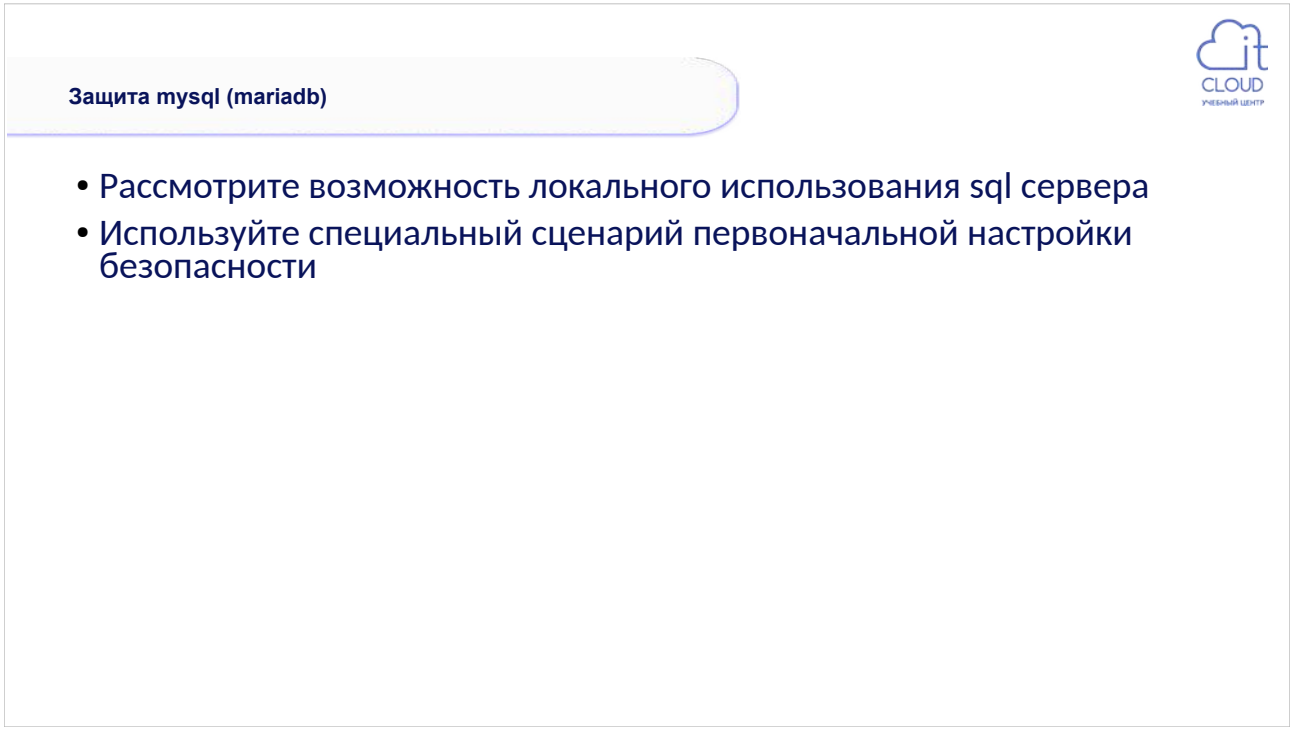
```
$ egrep '^(User|Group)' /etc/httpd/conf/httpd.conf
User apache
Group apache
```

- Изолируйте процессы веб сервера.
- Не устанавливайте в изолированную среду веб сервера оболочки пользователя.

В дополнение к стандартным модулям имеются дополнительные модули, предназначенные для повышения безопасности. Например mod_security.

Операционная система, в которой работает веб сервер так же должна быть защищена от несанкционированного доступа.

В Unix подобных системах поддерживается концепция изоляции процессов, которая будет рассмотрена ниже. Существуют три основных типа изоляции chroot, docker и lxc. Веб сервер, который скомпрометировали в изолированной среде, может повредить только эту среду.



В огромном числе инсталляций веб приложений используются какие-либо разновидности SQL сервера. Mysql или его форк mariadb, одни из самых популярных вариантов.

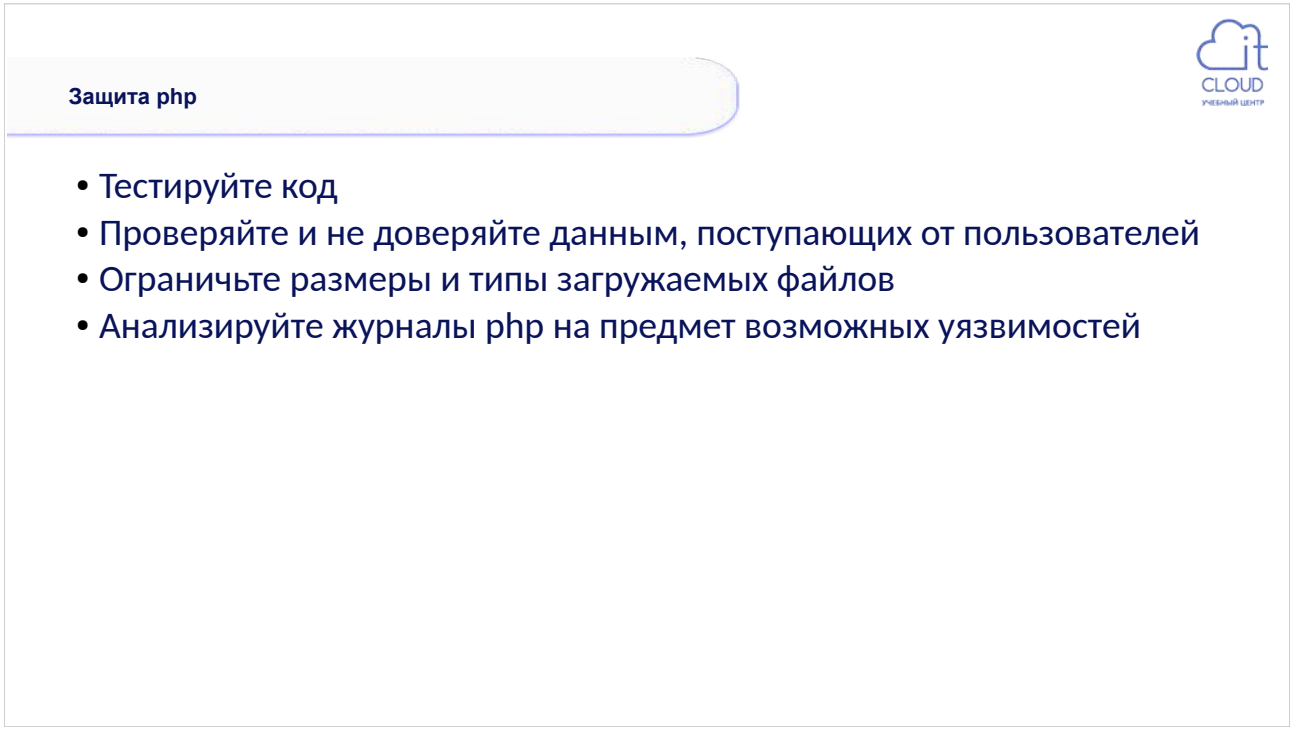
Первый вопрос, который следует решить где будет работать сервер баз данных на локальной машине или на удаленной. В случае локального использования ограничьте запуск SQL сервера только на адресе 127.0.0.1. Если используется удаленное подключение, то настройте правила локального файрволла для подключения только с разрешенных узлов.

В сервере mysql (mariadb) имеется специальный сценарий для первоначальной настройки сервера базы данных `mysql_secure_installation`. Воспользуйтесь им.

Пример:

```
root@s10:~# mysql_secure_installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user.  If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

Set root password? [Y/n]
New password:
Re-enter new password:
```

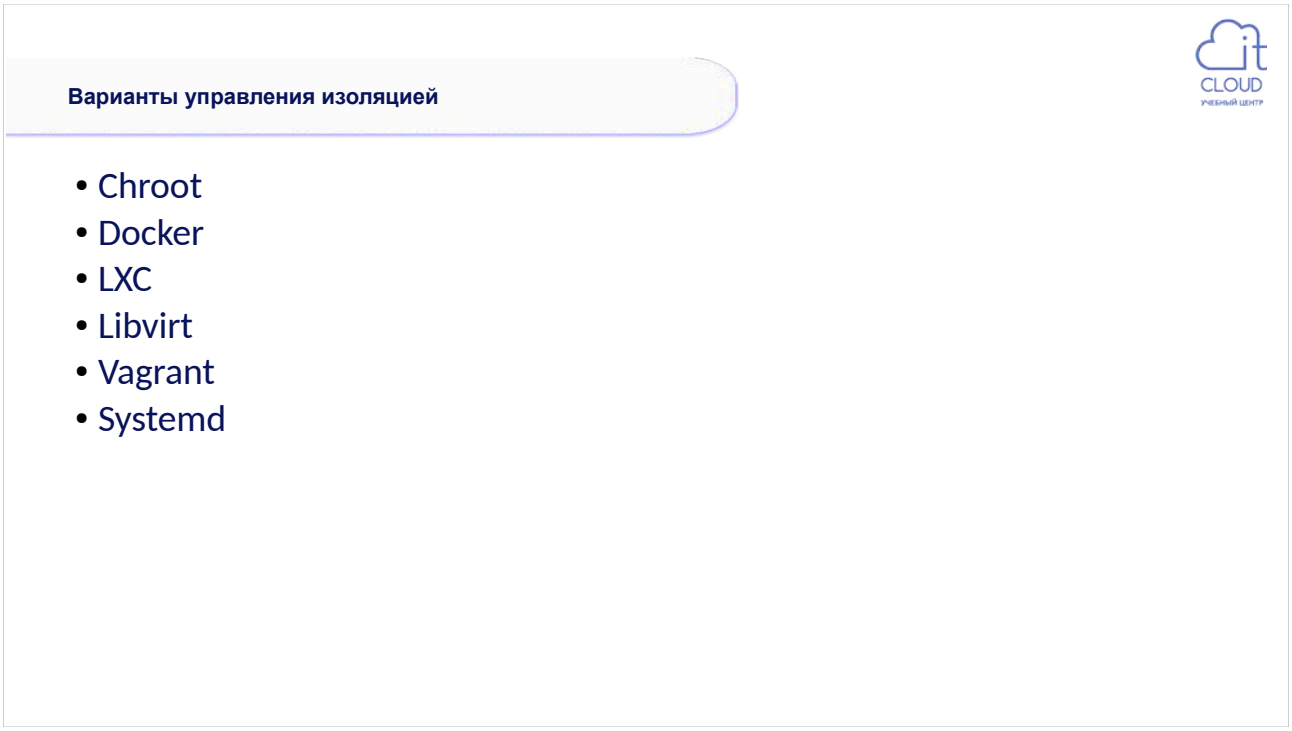
```
Password updated successfully!
Reloading privilege tables..
 ... Success!


By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] y
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] y
 ... Success!

By default, MariaDB comes with a database named 'test' that anyone can
access.  This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n] y
 - Dropping test database...
 ... Success!
 - Removing privileges on test database...
 ... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n] y
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
```



PHP это интерпретируемый язык сценариев. Все сценарии несут потенциальную угрозу, т. к. могу исполнять произвольный код.

- Всегда тестируйте ваш код. Оцените результат работы, получили ли вы, то что от него ожидали.
- Многие веб сайты принимают данные от пользователей: логины и пароли, сообщения в форумах, комментарии и многое другое. В этих данных пользователь может передать код php, который будет интерпретирован как руководство к действию.
- Если ваш сайт позволяет загружать файлы, то всегда проверяйте что загружается и в каком размере.
- Журналирование работы PHP поможет выявить ошибки кода.

10.2 Изоляция процесса.



Изоляция процессов применяется исходя из двух соображений.

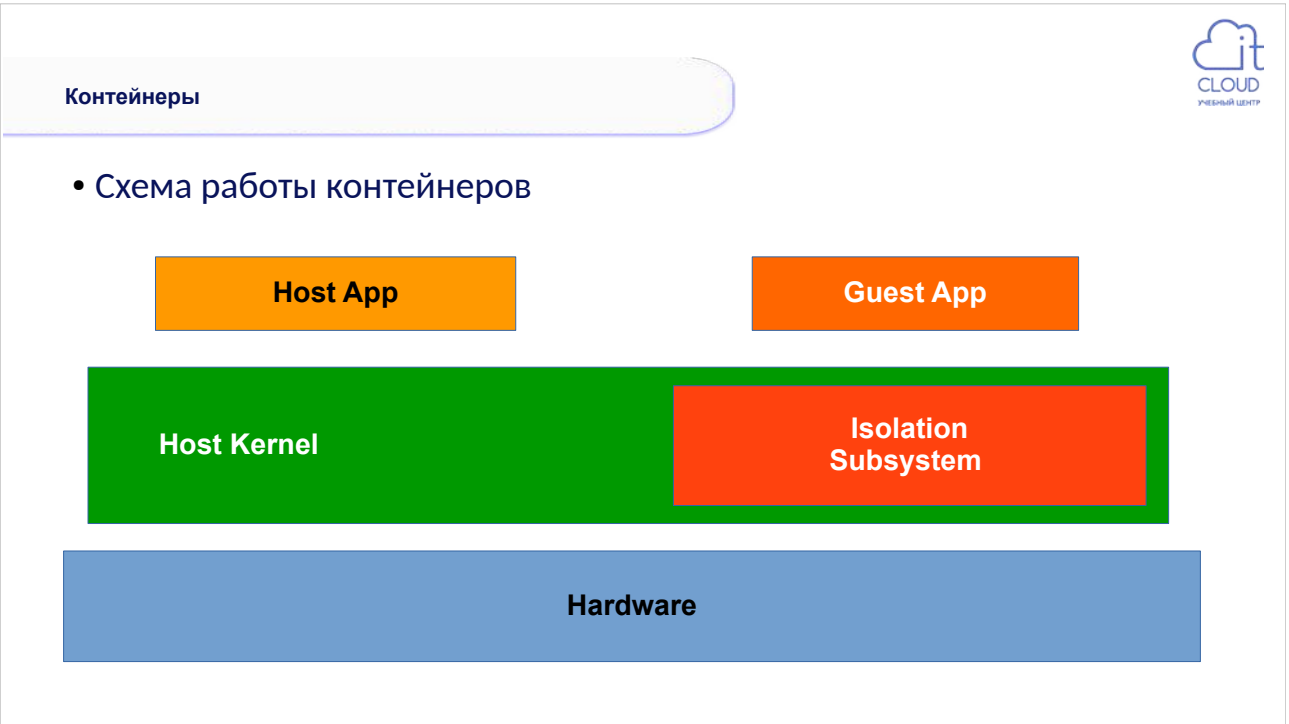
- Ограничение доступных ресурсов.
- Обеспечение безопасности.

Существуют несколько разновидностей изоляции:

- Chroot. При использовании chroot вы подменяете корневой каталог для процесса, т.о. процесс не может выбраться из того каталога, который ему выделен для работы. Иногда говорят, что процесс помещен в песочницу «sandbox». В остальном chroot процесс является обычным процессом.
- Docker. Вариант изоляции с использованием контейнеров и не только линукс. С помощью docker вы можете как настроить полноценный контейнер, так и создать контейнер, в котором работает только одна служба.
- LXC. (Linux upstream containers). Изолированная среда, в которой вы можете запустить отдельный ограниченный экземпляр ОС.

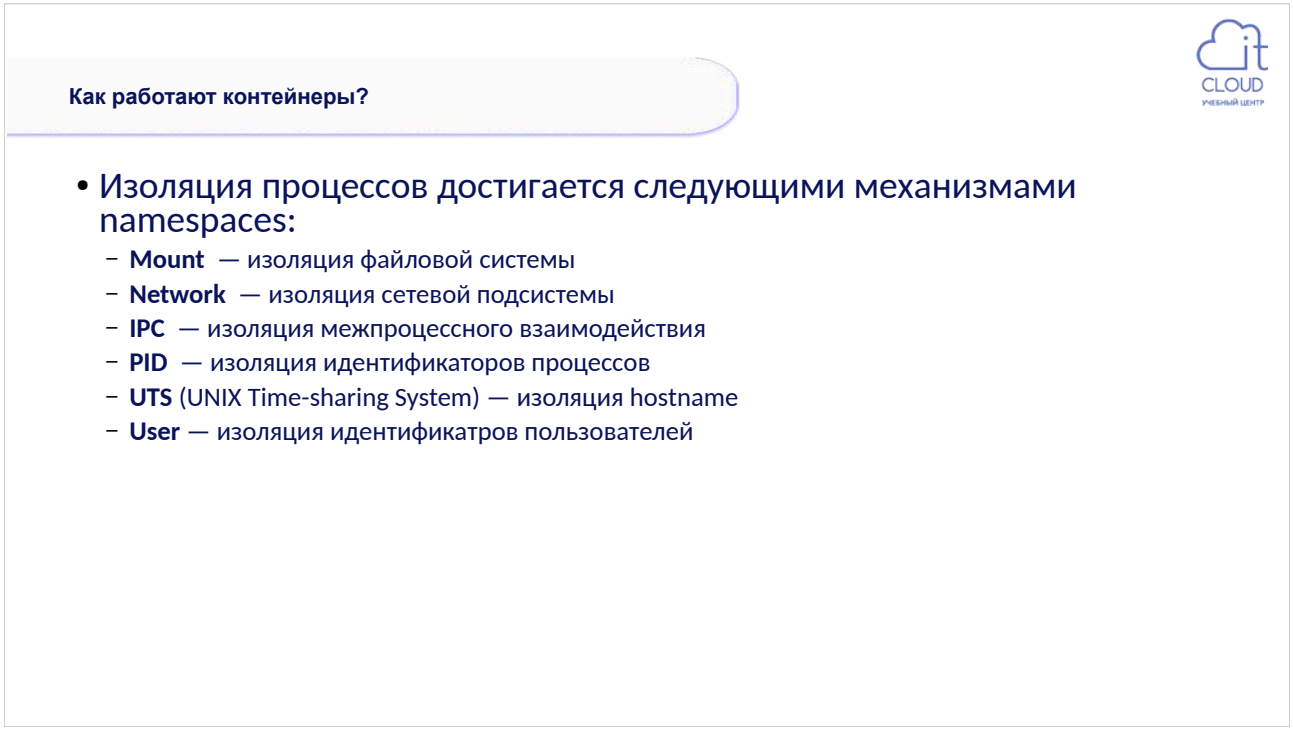
Строго говоря Docker и LXC это не вариант изоляции, а только инструмент посредством, которого производится управление контейнерами. Это только внешняя оболочка управления контейнерами, но сами инструменты контейнерами не являются.

Кроме указанных выше вариантов, существуют и другие инструменты такие, как LXD, libvirt, systemd, Vagrant, …



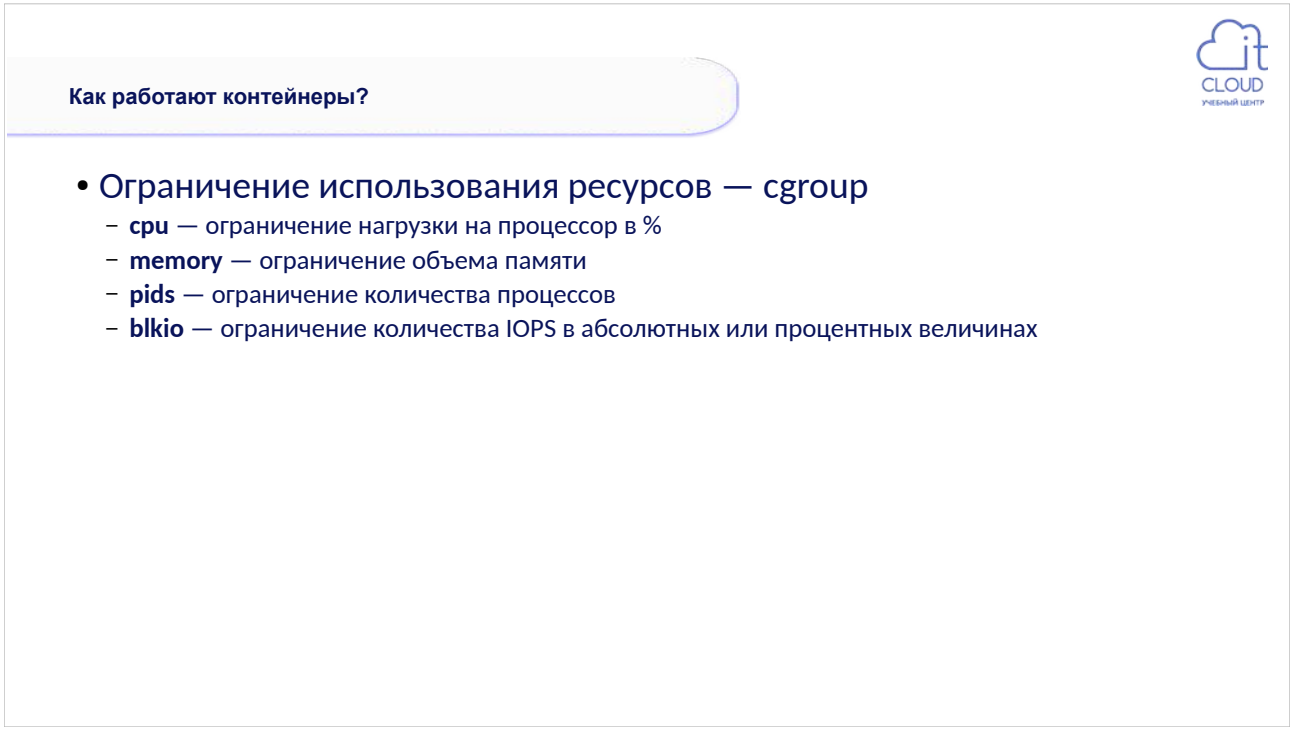
Контейнеры иногда называются виртуальными машинами, но это не совсем верно. Когда используется контейнер, то все его процессы с помощью подсистемы изоляции ограничиваются в отдельной среде.

Для изоляции применяются специальные механизмы под названием namespaces.



Изоляция процессов достигается следующими механизмами namespaces:

- Mount — изоляция файловой системы. Для контейнера создается отдельная корневая папка.
- Network — изоляция сетевой подсистемы. В контейнер вы можете добавить несколько виртуальных сетевых адаптеров.
- IPC — изоляция межпроцессного взаимодействия. Процессы внутри контейнера могут взаимодействовать только друг с дургом.
- PID — изоляция идентификаторов процессов. С точки зрения процессов внутри контейнера процессы имеют отдельную иерархию процессов с отдельным процессом init (PID 1).
- UTS (UNIX Time-sharing System) — изоляция hostname. У каждого контейнера может быть свое уникальное имя узла.
- User — изоляция идентификатров пользователей. Все пользователи контейнера отделены от хостовых пользователей и пользователей других контейнеров. В том числе и суперпользователь.



Помимо изоляции процессов, пользователей и устройств контейнеры ограничивают использование системных ресурсов. Для этого применяется механизм cgroup (control group).

- cpu — ограничение нагрузки на процессор в %.
- memory — ограничение объема памяти.
- pids — ограничение количества процессов.
- blkio — ограничение количества IOPS в абсолютных или процентных величинах.

Механизм ограничений использует для своей работы SELinux. Поэтому при использовании контейнеров нельзя отключать SELinux.

Система управления контейнерами Docker, наверное, самое популярное решение на данный момент. Docker может работать не только в Linux, но и в других операционных системах: Windows и MacOS.

Docker предназначен для решения однотипных задач, когда нужно раз за разом разворачивать однотипные службы. Хорошо сочетается с идеями микросервисов и Devops.

Пример: Установка Docker и запуск в нем контейнера.

1. Установим пакет docker:

```
root@s10:~# apt install docker.io
```

2. Добавим пользователя в группу docker, чтобы разрешить ему управлять контейнерами.

```
root@s10:~# gpasswd -a sa docker
Добавление пользователя sa в группу docker
```

3. Найдем образ с приложением httpd (веб-сервер Apache).

```
sa@s10:~$ docker search --filter is-official=true httpd
NAME      DESCRIPTION                    STARS     OFFICIAL   AUTOMATED
httpd     The Apache HTTP Server Project   4828      [OK]
```

4. Запустим это приложение и проверим работу веб-сервера:

```
sa@s10:~$ docker run -p 8080:80 --rm -d httpd
Unable to find image 'httpd:latest' locally
latest: Pulling from library/httpd
c29f5b76f736: Pull complete
830a84f99cc8: Pull complete
4f4fb700ef54: Pull complete
a1a1b409f475: Pull complete
35b1ecb71608: Pull complete
80350326cd93: Pull complete
Digest: sha256:3195404327ecd95b2fa0a5d4eac1f2206bb12996fb2561393f91254759e422b9
Status: Downloaded newer image for httpd:latest
```

```
3c0097b47d8a7bff026ee5bd8a087b8c38a2b661cd024fd39da35a87fb809c62

sa@s10:~$ docker container ls
CONTAINER ID   IMAGE     COMMAND              CREATED          STATUS
PORTS                                  NAMES
3c0097b47d8a   httpd     "httpd-foreground"   49 seconds ago   Up 46 seconds
0.0.0.0:8080->80/tcp, :::8080->80/tcp   epic_bhaskara

sa@s10:~$ curl http://localhost:8080
<html><body><h1>It works!</h1></body></html>

sa@s10:~$ docker container stop epic_bhaskara
epic_bhaskara
sa@s10:~$ docker container ls
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES

sa@s10:~$ docker image ls
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
httpd        latest    4d98e80840bb   2 weeks ago   148MB
```

Контейнер в докере автоматически загрузил образ и заработал. Контейнер запустился в фоновом режиме (опция `-d`) и после остановки удалился (опция `-rm`). При следующем запуске контейнер начнет работу с чистого листа.

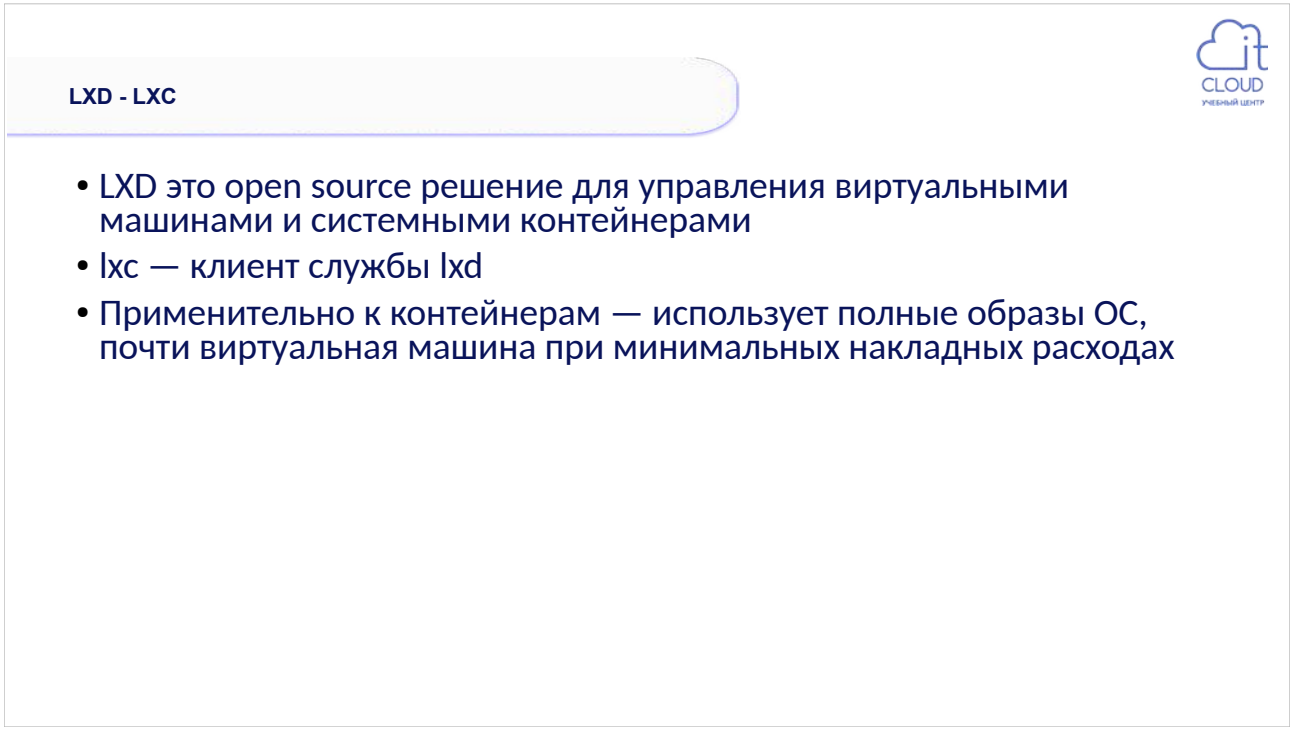
5. Наполним контейнер «смыслом»:

```
sa@s10:~$ mkdir docker_html
sa@s10:~$ echo '<html><body><h1>My content!</h1></body></html>' >
docker_html/index.html

sa@s10:~$ docker run -p 8080:80  -d -h websrv --name websrv-docker -m 1024M -v \
 $(pwd)/docker_html:/usr/local/apache2/htdocs httpd
e1a3101868645458ec1db9368259ab9988604ebb08a8a3c51a118a34932295ae

sa@s10:~$ curl http://localhost:8080
<html><body><h1>My content!</h1></body></html>

sa@s10:~$ cp -r docker_html/ docker8888_html/
sa@s10:~$ echo '<html><body><h1>My 8888 server!</h1></body></html>' >
docker8888_html/index.html

sa@s10:~$ docker run -p 8888:80  -d -h websrv8888 --name websrv8888-docker -m
1024M -v $(pwd)/docker8888_html:/usr/local/apache2/htdocs httpd
ce45a6d9b4ca01a34f2cf747166ec67b6f732e41a3053b1d80c73167ae9a56c9

sa@s10:~$ curl http://localhost:8888
<html><body><h1>My 8888 server!</h1></body></html>

sa@s10:~$ docker container ls
CONTAINER ID   IMAGE     COMMAND              CREATED          STATUS
PORTS                                  NAMES
ce45a6d9b4ca   httpd     "httpd-foreground"   55 seconds ago   Up 53 seconds
0.0.0.0:8888->80/tcp, :::8888->80/tcp   websrv8888-docker
e1a310186864   httpd     "httpd-foreground"   5 minutes ago    Up 5 minutes
0.0.0.0:8080->80/tcp, :::8080->80/tcp   websrv-docker
```

Docker достаточно прост в освоении и использовании, но у этого решения есть и спорные моменты, например, Docker бесцеремонно вмешивается в работу системы фильтрации пакетов на хосте, более того использует до сих пор устаревший iptables.

Глава 10. Безопасность уровня приложений.

В идеале кроме микросервисов докера на хосте ничего не должно быть запущено.



Для управления контейнерами могут использоваться различные средства. Для выбора способа управления контейнерами сначала определите круг решаемых задач. Так, например, при массовом развертывании однотипных приложений удобней воспользоваться Docker. С другой стороны, если вы создаете сложные контейнеры, состоящие из множества элементов, то, возможно, больше подойдет LXD, которым удобно создавать контейнеры в виде экземпляров запущенной ОС (почти виртуальная машина).

Пример: Установка LXD и запуск в нем контейнера.

1. Установим нужные пакеты:

```
root@s10:~# apt install lxd lxd-tools
```

2. Произведем первоначальную настройку:

```
root@s10:~# lxd init
Would you like to use LXD clustering? (yes/no) [default=no]:
Do you want to configure a new storage pool? (yes/no) [default=yes]:
Name of the new storage pool [default=default]:
Would you like to connect to a MAAS server? (yes/no) [default=no]:
Would you like to create a new local network bridge? (yes/no) [default=yes]:
What should the new bridge be called? [default=lxdbr0]:
What IPv4 address should be used? (CIDR subnet notation, “auto” or “none”)
[default=auto]:
What IPv6 address should be used? (CIDR subnet notation, “auto” or “none”)
[default=auto]:
Would you like the LXD server to be available over the network? (yes/no)
[default=no]: yes
Address to bind LXD to (not including port) [default=all]:
Port to bind LXD to [default=8443]:
Trust password for new clients:
Again:
Would you like stale cached images to be updated automatically? (yes/no)
[default=yes]:
Would you like a YAML "lxd init" preseed to be printed? (yes/no) [default=no]: y
```

```
config:
  core.https_address: '[::]:8443'
  core.trust_password: lin123
networks:
- config:
    ipv4.address: auto
    ipv6.address: auto
  description: ""
  name: lxdbr0
  type: ""
  project: default
storage_pools:
- config: {}
  description: ""
  name: default
  driver: dir
profiles:
- config: {}
  description: ""
  devices:
    eth0:
      name: eth0
      network: lxdbr0
      type: nic
    root:
      path: /
      pool: default
      type: disk
  name: default
projects: []
cluster: null
```

3. Проверим источники, с которых мы можем получить образы:

```
root@s10:~# lxc remote list -f csv
images,https://images.linuxcontainers.org,simplestreams,none,YES,NO,NO
local (current),unix://,lxd,file access,NO,YES,NO
ubuntu,https://cloud-images.ubuntu.com/releases,simplestreams,none,YES,YES,NO
ubuntu-daily,https://cloud-images.ubuntu.com/daily,simplestreams,none,YES,YES,NO

root@s10:~# lxc image list images: -f csv
```

Ничего нет. На самом деле источник закрыт создателями для публичного использования. Цитата: «Image server access is being phased out for LXD users, see here for details.»

```
root@s10:~# lxc image list ubuntu: -f csv | head -3
a (5 more),2d53824fdf89,yes,ubuntu 17.10 amd64 (release)
(20180706),x86_64,CONTAINER,169.51MB,"Jul 6, 2018 at 12:00am (UTC)"
a (5 more),34bae4293007,yes,ubuntu 17.10 amd64 (release)
(20180706),x86_64,VIRTUAL-MACHINE,307.06MB,"Jul 6, 2018 at 12:00am (UTC)"
a/arm64 (2 more),9807825fcc6a,yes,ubuntu 17.10 arm64 (release)
(20180706),aarch64,VIRTUAL-MACHINE,286.44MB,"Jul 6, 2018 at 12:00am (UTC)"
```

4. Добавим еще один удаленный репозиторий:

```
root@s10:~# lxc remote add canonical-imgs https://images.lxd.canonical.com --
protocol simplestreams
root@s10:~# lxc image list canonical-imgs: -f csv | wc -l
320
root@s10:~# lxc image list canonical-imgs: -f csv | grep debian | wc -l
```

```
24
```

Для сравнения:

```
root@s10:~# lxc image list ubuntu: -f csv | wc -l
11601
root@s10:~# lxc image list ubuntu: -f csv | grep ubuntu | wc -l
11601
```

5. Запустим контейнер:

```
root@s10:~# lxc launch canonical-imgs:'debian/12' deb12-container
Creating deb12-container
Starting deb12-container

root@s10:~# lxc list -c n4st
+-----------------+----------------------+---------+-----------+
|      NAME       |         IPV4         |  STATE  |   TYPE    |
+-----------------+----------------------+---------+-----------+
| deb12-container | 10.201.160.126 (eth0) | RUNNING | CONTAINER |
+-----------------+----------------------+---------+-----------+
```

6. Войдем в контейнер и выполним какое-нибудь действие:

```
root@s10:~# lxc exec deb12-container /bin/bash
root@deb12-container:~# echo ABC > test.txt
root@deb12-container:~# exit
exit
```

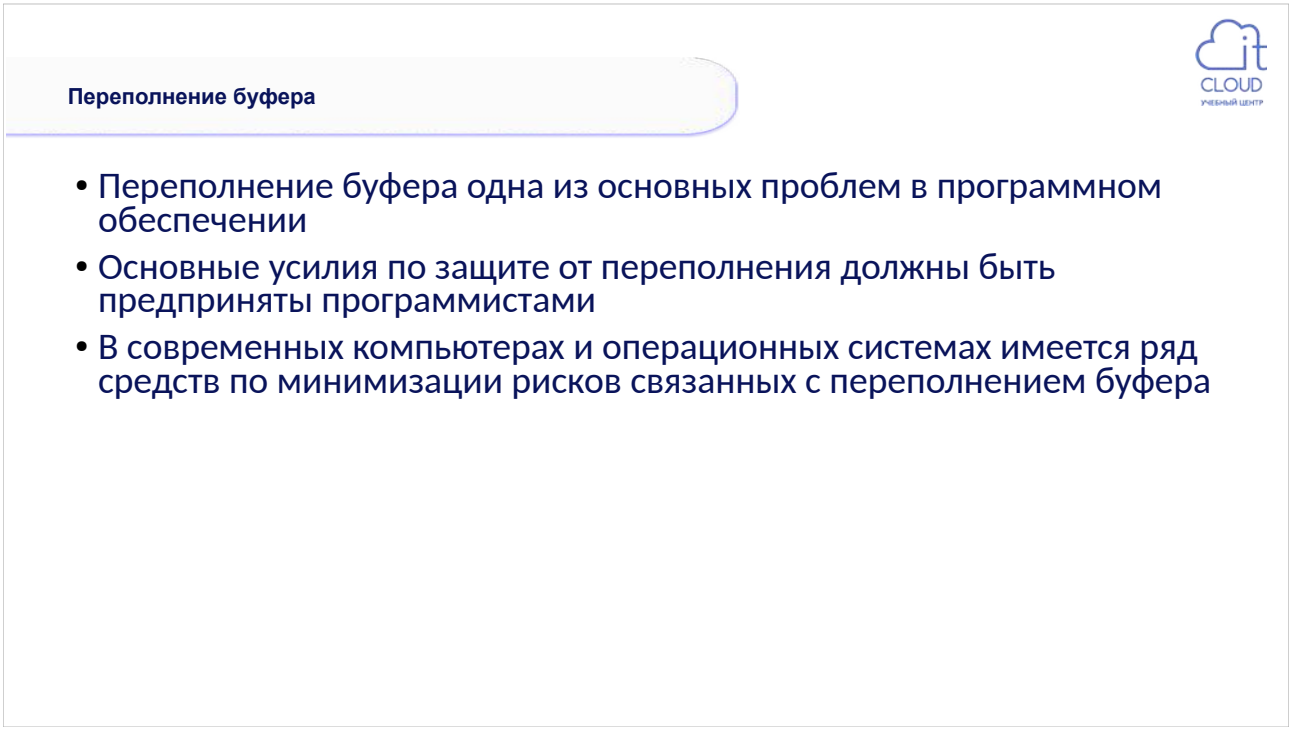
7. Выключим контейнер и получим доступ к файлам внутри контейнера:

```
root@s10:~# lxc stop deb12-container

root@s10:~# lxc list -c n4st
+-----------------+------+---------+-----------+
|      NAME       | IPV4 |  STATE  |   TYPE    |
+-----------------+------+---------+-----------+
| deb12-container |      | STOPPED | CONTAINER |
+-----------------+------+---------+-----------+

root@s10:~# lxc storage list -f csv
default,dir,/var/lib/lxd/storage-pools/default,,2,CREATED

root@s10:~# cat
/var/lib/lxd/storage-pools/default/containers/deb12-container/rootfs/root/
test.txt
ABC

root@s10:~# echo 123 >> /var/lib/lxd/storage-pools/default/containers/deb12-
container/rootfs/root/test.txt
```

8. Вновь запустим контейнер и проверим содержимое файла.

```
root@s10:~# lxc start deb12-container
root@s10:~# lxc exec deb12-container cat test.txt
ABC
123
```

10.3 Защита от переполнения буфера (запрет формирования дампа ядра core dump, запрет выполнения кода в стеке).



Переполнение буфера (Buffer Overflow) — явление, возникающее, когда компьютерная программа записывает данные за пределами выделенного в памяти буфера.

Переполнение буфера обычно возникает из-за неправильной работы с данными, полученными извне, и памятью, при отсутствии жесткой защиты со стороны подсистемы программирования (компилятор или интерпретатор) и операционной системы. В результате переполнения могут быть испорчены данные, расположенные следом за буфером (или перед ним).

Переполнение буфера является одним из наиболее популярных способов взлома компьютерных систем, так как большинство языков высокого уровня используют технологию стекового кадра — размещение данных в стеке процесса, смешивая данные программы с управляющими данными (в том числе адреса начала стекового кадра и адреса возврата из исполняемой функции).

Переполнение буфера может вызывать аварийное завершение или зависание программы, ведущее к отказу обслуживания (denial of service, DoS). Отдельные виды переполнений, например переполнение в стековом кадре, позволяют злоумышленнику загрузить и выполнить произвольный машинный код от имени программы и с правами учетной записи, от которой она выполняется.

Известны примеры, когда переполнение буфера намеренно используется системными программами для обхода ограничений в существующих программных или программно-

аппаратных средствах. Например, операционная система iS-DOS (для компьютеров ZX Spectrum) использовала возможность переполнения буфера встроенной TR-DOS для запуска своего загрузчика в машинных кодах (что штатными средствами в TR-DOS сделать невозможно).

Перед использованием данных полученных извне программа должна проанализировать его, и если данные превышают допустимый размер или содержат такие части, как шелл-коды, то такие данные должны быть отброшены.

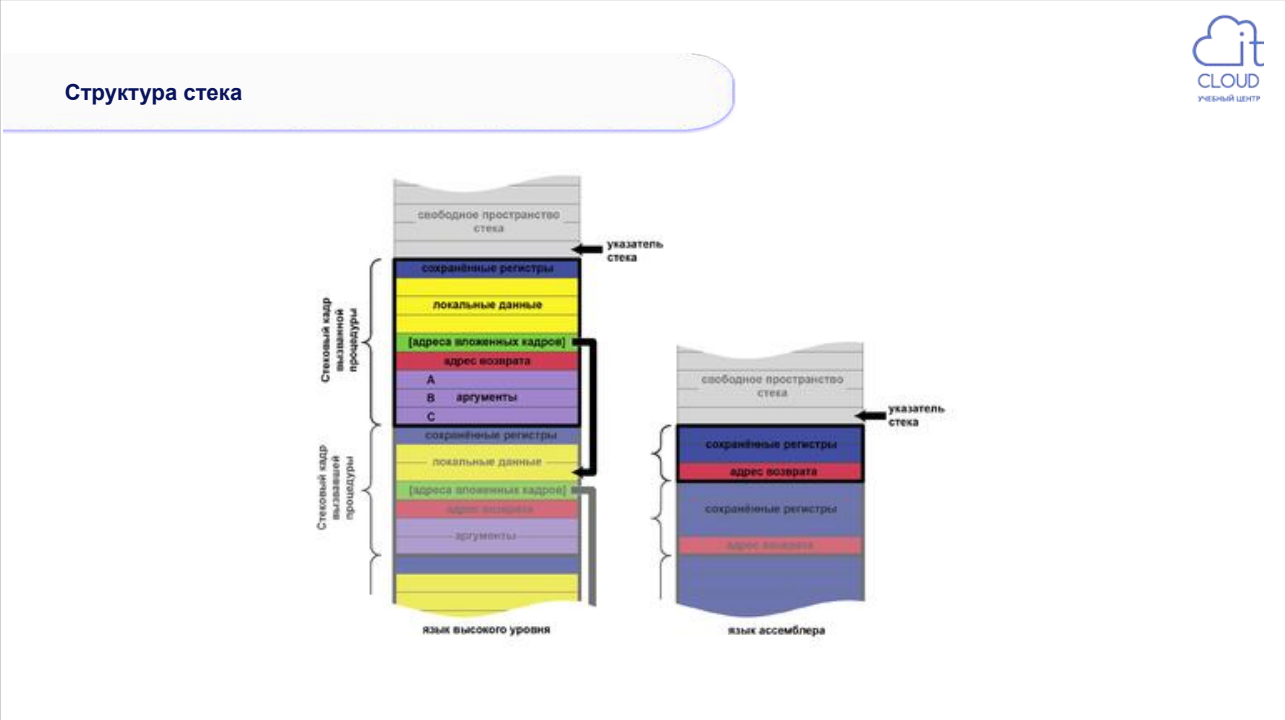
Защита пространства исполняемого кода может смягчить последствия переполнений буфера, делая большинство действий злоумышленников невозможными. Это достигается рандомизацией адресного пространства (**ASLR**) и/или запрещением одновременного доступа к памяти на запись и исполнение. Неисполняемый стек предотвращает большинство эксплойтов кода оболочки.

Существует два исправления для ядра Linux, которые обеспечивают эту защиту — PaX и exec-shield. Ни один из них ещё не включен в основную поставку ядра. OpenBSD с версии 3.3 включает систему, называемую W^X, которая также обеспечивает контроль исполняемого пространства.

Заметим, что этот способ защиты не предотвращает повреждение стека. Однако он часто предотвращает успешное выполнение «полезной нагрузки» эксплойта. Программа не будет способна вставить код оболочки в защищённую от записи память, такую как существующие сегменты исполняемого кода. Также будет невозможно выполнение инструкций в неисполняемой памяти, такой как стек или куча.

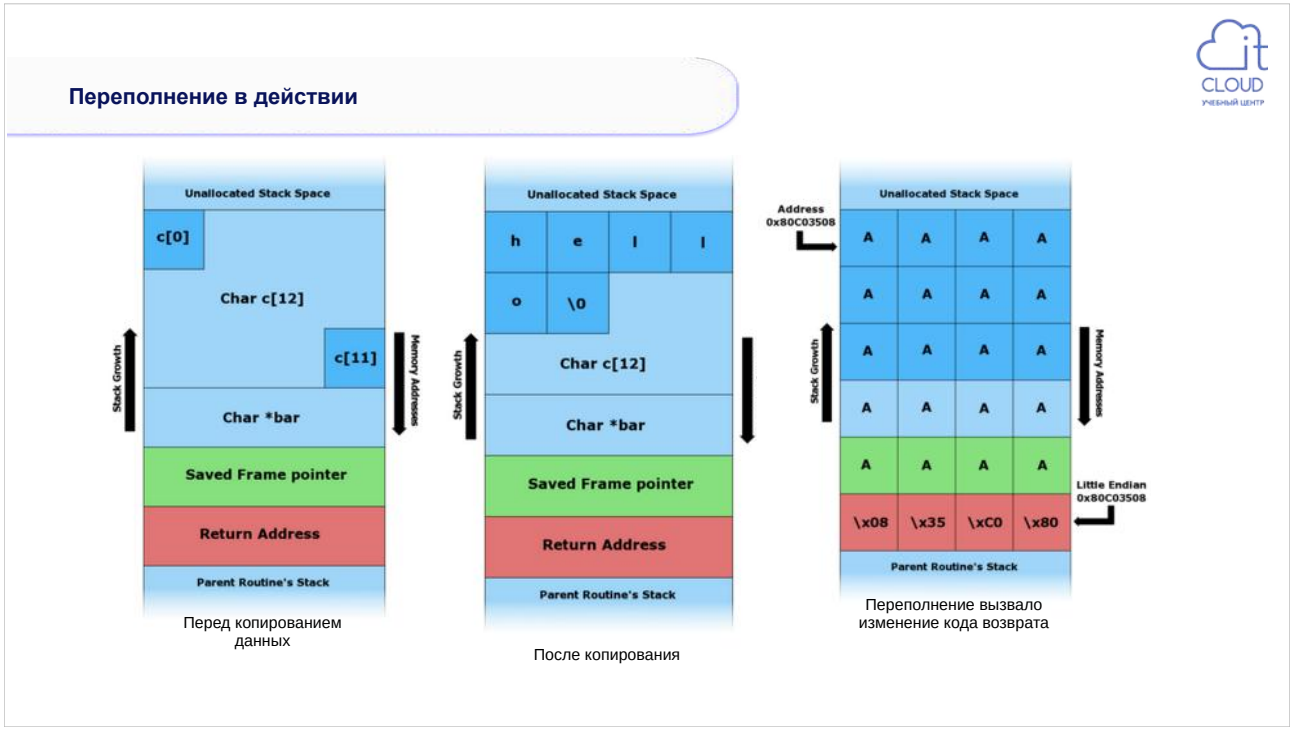
ASLR затрудняет для взломщика определение адресов функций в коде программы, с помощью которых он мог бы осуществить успешную атаку, и делает атаки типа ret2libc очень трудной задачей, хотя они всё ещё возможны в контролируемом окружении, или если атакующий правильно угадает нужный адрес.

Некоторые процессоры, такие как Sparc фирмы Sun, Efficeon фирмы Transmeta, и новейшие 64-битные процессоры фирм AMD и Intel предотвращают выполнение кода, расположенного в областях памяти, помеченных специальным битом NX. AMD называет своё решение NX (от англ. No eXecute), а Intel своё — XD (от англ. eXecute Disabled).

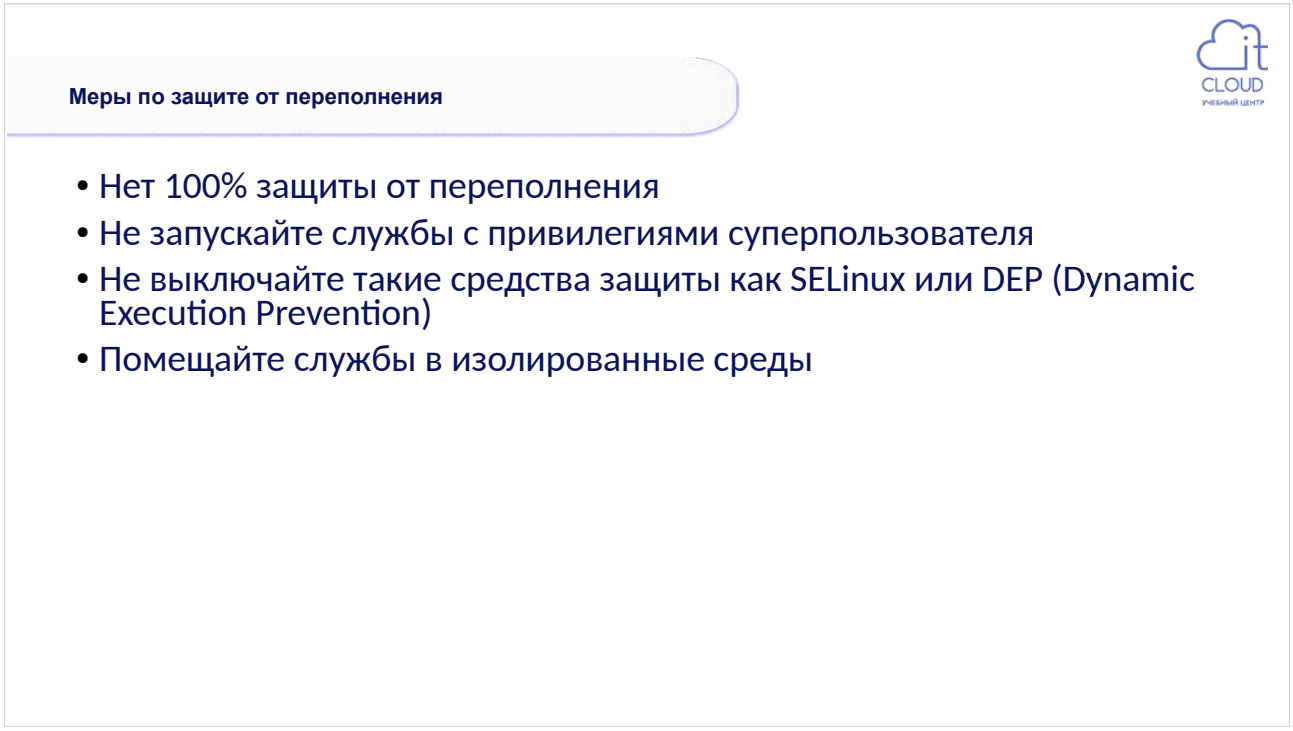


Стек растет от старших адресов к младшим. При этом данные помещаемые в стек могут превысить предоставленное им пространство.

Адрес возврата указывает на какую инструкцию должна вернуться программа после выхода из процедуры.



Если не отслеживать что записывается в стек, то можно подменить адрес возврата на шелл код. И вы получите доступ к системе с правами этой программы.

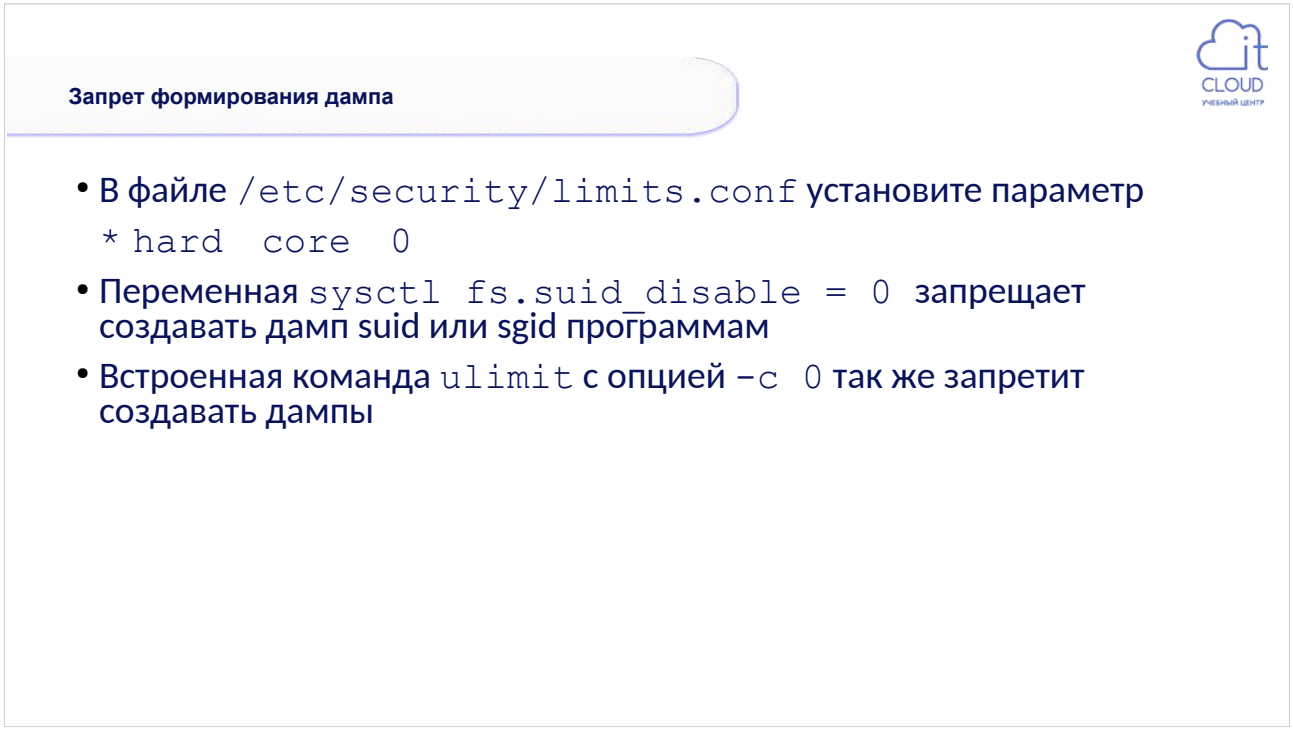


Прежде всего нет 100% гарантированной защиты от переполнения.

Служба, которая работает с привилегиями суперпользователя являются потенциально самыми опасными. Если злоумышленник сможет скомпрометировать такую службу, он сможет получить полный контроль над системой.

Такие средства как SELinux, DEP или ASLR существенно затрудняют или возможность эксплуатации переполнения или уменьшают потенциальный вред от использованной уязвимости.

Изоляция процессов также сможет минимизировать потенциальный вред от эксплойтов.



Дампы программ могут содержать важную и конфиденциальную информацию. В целях усиления безопасности рекомендуется запретить создавать дампы сбойных процессов.

В файле `/etc/security/limits.conf` установите параметр hard core равным 0. Ноль означает нулевой размер файла.

Пример:

```
root@s10:~# grep core /etc/security/limits.conf
#        - core - limits the core file size (KB)
*            hard  core        0
#*               soft    core            0
```

Если вы хотите временно запретить создавать дампы, то можно использовать встроенную команду ulimit для установки размера core файла.

Пример:

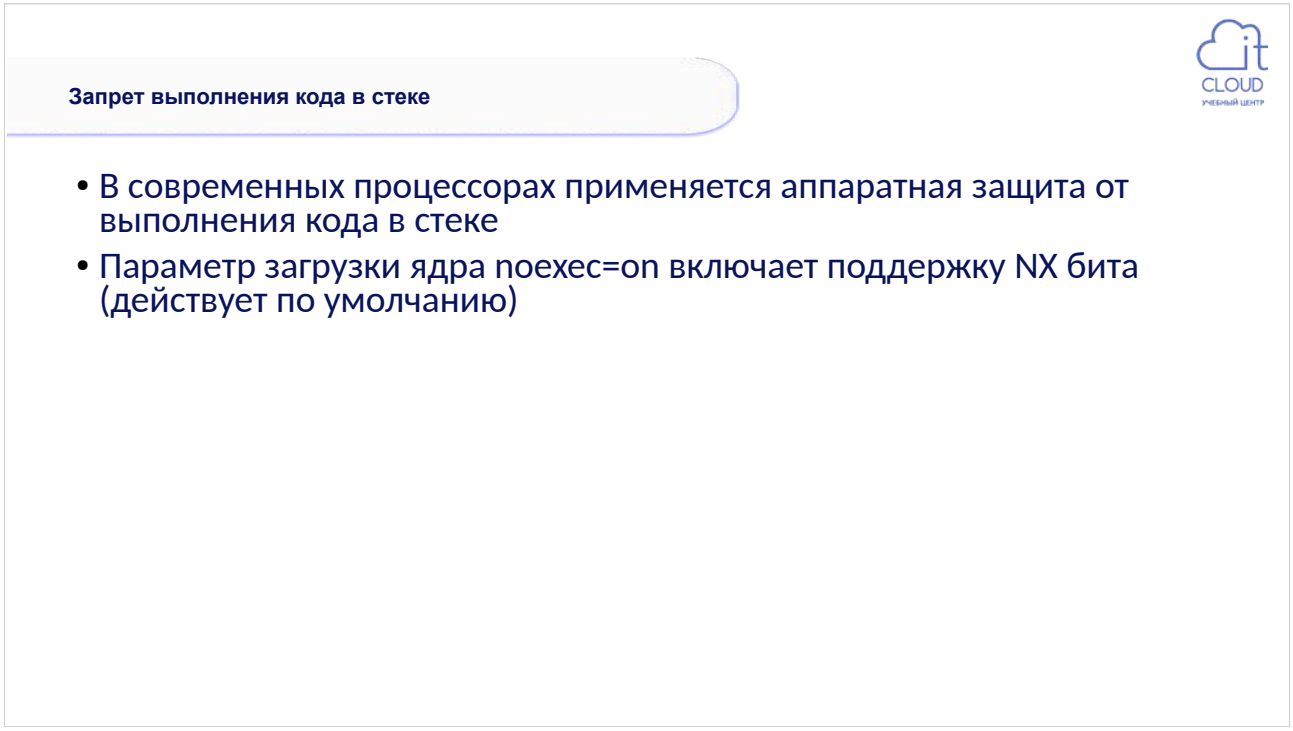
```
root@s10:~# ulimit -c 0
root@s10:~# ulimit -c
0
```

Переменная `sysctl fs.suid_disable` определяет разрешения определяет какие разрешения имеют core файлы suid программ.

0 (применяется по умолчанию) запрещает дампы у программ с измененными привилегиями.

1 всем процессам разрешены дампы.

2 дампы доступны только суперпользователю.



Запрет на выполнения может применяться на аппаратном уровне в виде NX бита (AMD) или XD (Intel).

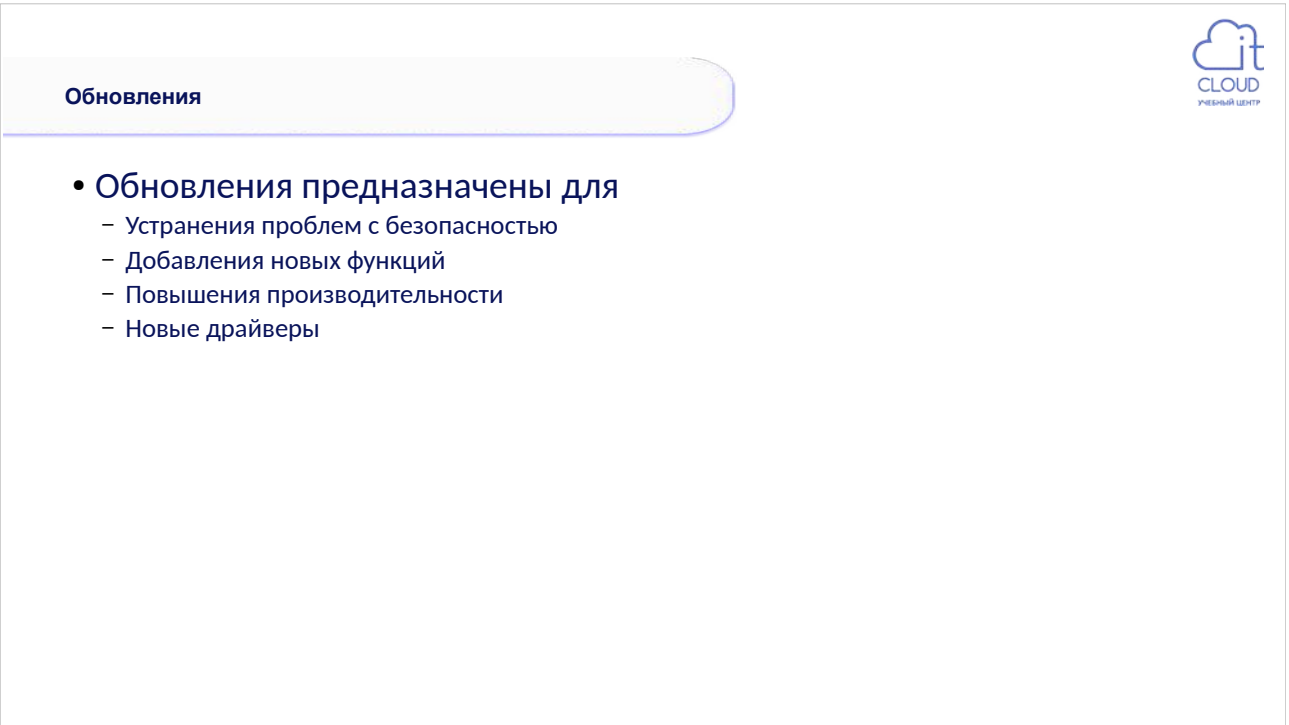
Параметр загрузки ядра noexec управляет включением или отключением этой функции.

Существуют так же альтернативы нативной поддержки NX.

- В RedHat exec-shield.
- Патч ядра PaX, который может эмулировать функционал NX бита.

Глава 11. Поддержание системы в актуальном состоянии.

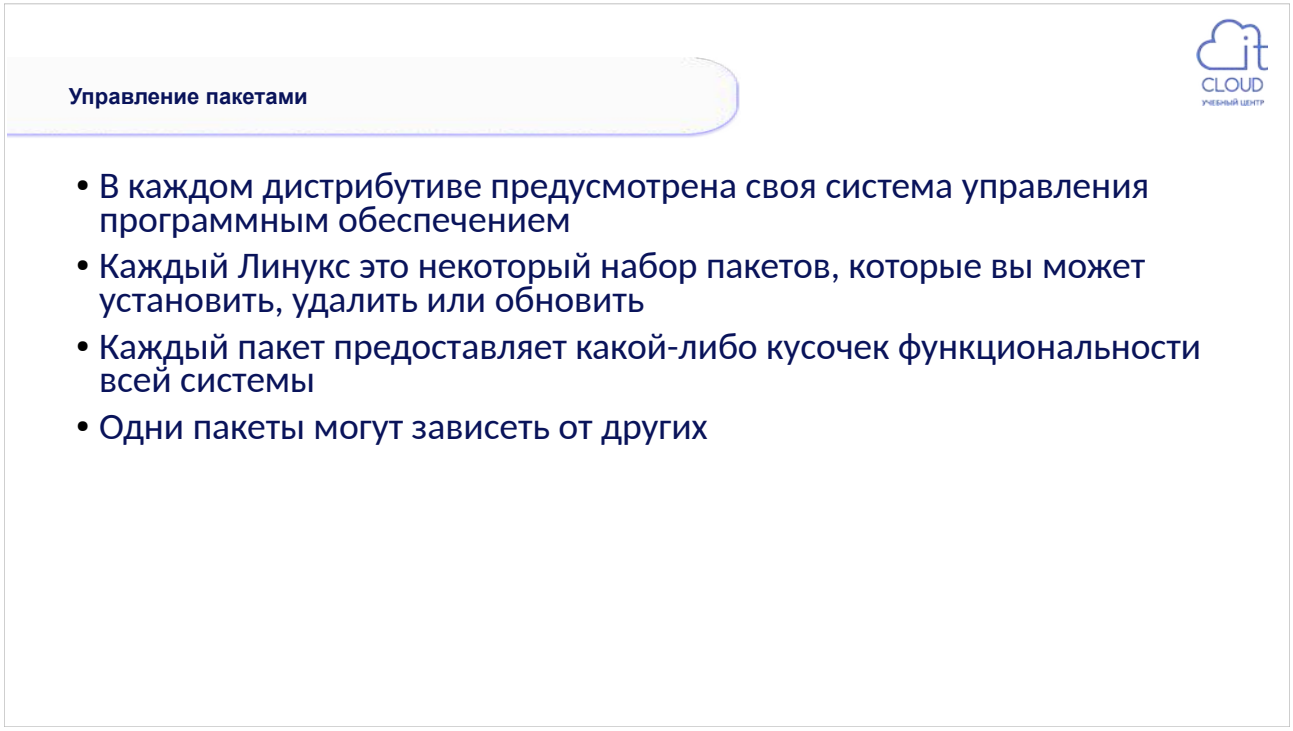
11.1 Обновление системы



С течением времени в любой операционной системе обнаруживаются ошибки. Эти ошибки могут влиять на стабильность работы или использоваться для вторжения в систему.

Программисты постоянно улучшают свои продукты, добавляя в них новые функции или улучшая работу старого функционала.

Постоянно совершенствуется аппаратная часть компьютеров, для которой может потребоваться обновление программной части. Чтобы эти аппаратные компоненты работали более эффективно или вообще работали.



В современных дистрибутивах Линукс, как правило, имеется система управления программным обеспечением. Зачастую эти системы управления не совместимы между собой.

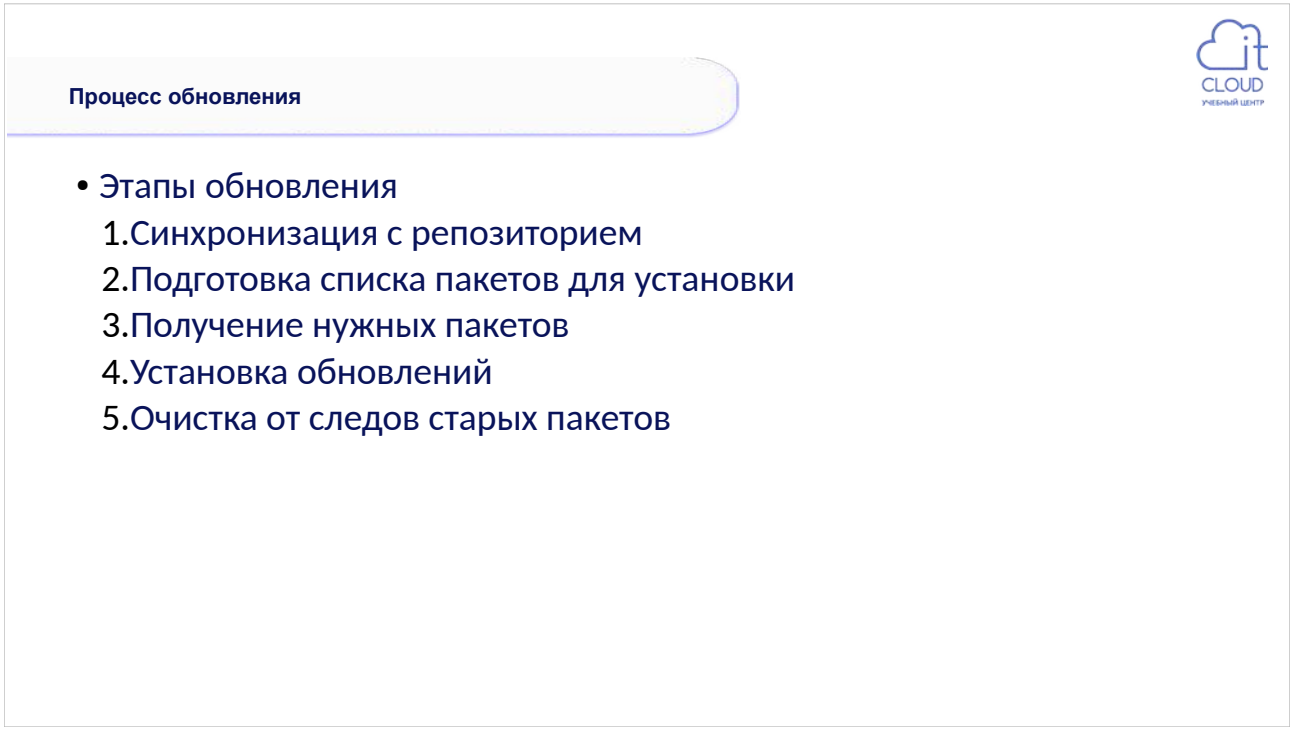
Обычно системы управления пакетами делятся на два уровня.

1. Нижний уровень. Предназначен для создания, установки, удаления, обновления или проверки пакетов. Примеры rpm или dpkg.
2. Верхний уровень. Позволяет найти нужный пакет или автоматически разрешить зависимости пакетов. Примеры apt, yum, dnf, zipper и т.д.

В Линукс используется подход для разделения функциональности на мелкие части, которые находятся в своих пакетах. В связи с этим.

- Пакеты, как правило, не большого размера.
- Пакеты зависят друг от друга.

Как правило, система управления пакетами работает со множеством пакетов, хранящихся в специальном репозитории — хранилище, которое может располагаться как на локальных запоминающих устройствах (оптическом или жёстком диске), так и на удалённой машине (HTTP, FTP или rsync-сервере).



Весь процесс обновления состоит из нескольких шагов.

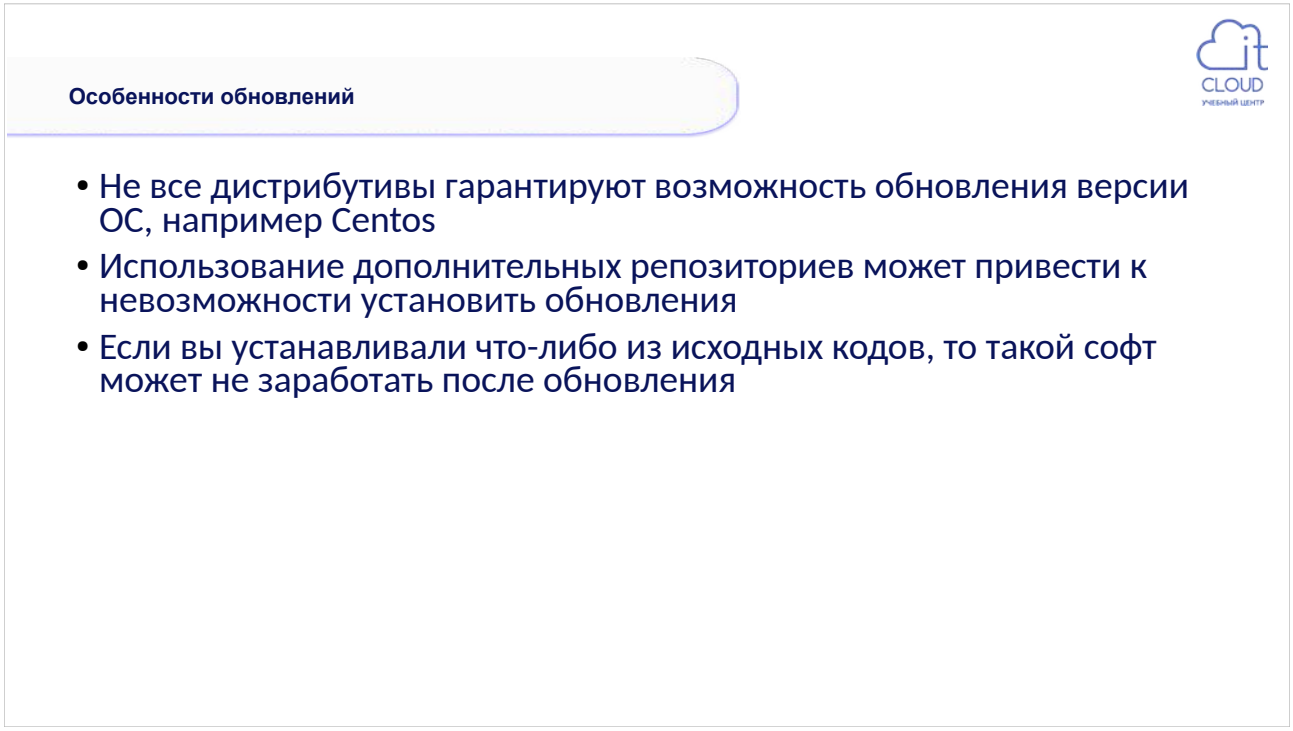
1. Синхронизация с репозиторием. Системы управления пакетами хранят локальные копии базы данных пакетов из репозиторя, поэтому процесс обновления начинается с синхронизации локальной копии с репозиторием. В некоторых случаях для этого выполняется отдельная команда, например в apt, в других случаях это специально делать не требуется, например в yum.
2. Подготовка списка пакетов для установки. После обновления базы пакетов системы управления создают список пакетов, которые имеют обновленную версию.
3. Получение нужных пакетов. Пакеты загружаются из репозитория.
4. Установка обновлений.
5. Очистка от следов старых пакетов. После обновления производится очистка от старых пакетов.

Пример: Обновление системы в Debian.

```
root@s10:~# apt update
Сущ:1 http://deb.debian.org/debian bookworm InRelease
Пол:2 http://security.debian.org/debian-security bookworm-security InRelease
[48,0 kB]
Пол:3 http://deb.debian.org/debian bookworm-updates InRelease [55,4 kB]
Пол:4 http://security.debian.org/debian-security bookworm-security/main amd64
Packages [243 kB]
Получено 346 kB за 1с (239 kB/s)
Чтение списков пакетов… Готово
Построение дерева зависимостей… Готово
Чтение информации о состоянии… Готово
Может быть обновлено 2 пакета. Запустите «apt list --upgradable» для их показа.

root@s10:~# apt list --upgradable
```

```
Вывод списка… Готово
firefox-esr-l10n-ru/stable-security 128.7.0esr-1~deb12u1 all [может быть
обновлён с: 128.6.0esr-1~deb12u1]
firefox-esr/stable-security 128.7.0esr-1~deb12u1 amd64 [может быть обновлён с:
128.6.0esr-1~deb12u1]

root@s10:~# apt upgrade
Чтение списков пакетов… Готово
Построение дерева зависимостей… Готово
Чтение информации о состоянии… Готово
Расчёт обновлений… Готово
Следующие пакеты будут обновлены:
  firefox-esr firefox-esr-l10n-ru
Обновлено 2 пакетов, установлено 0 новых пакетов, для удаления отмечено 0
пакетов, и 0 пакетов не обновлено.
Необходимо скачать 70,4 MB архивов.
После данной операции объём занятого дискового пространства возрастёт на 34,8
kB.
Хотите продолжить? [Д/н]
Пол:1 http://security.debian.org/debian-security bookworm-security/main amd64
firefox-esr-l10n-ru all 128.7.0esr-1~deb12u1 [663 kB]
Пол:2 http://security.debian.org/debian-security bookworm-security/main amd64
firefox-esr amd64 128.7.0esr-1~deb12u1 [69,8 MB]
Получено 70,4 MB за 9c (7 803 kB/s)
Чтение журналов изменений... Выполнено
(Чтение базы данных … на данный момент установлено 224489 файлов и каталогов.)
Подготовка к распаковке …/firefox-esr-l10n-ru_128.7.0esr-1~deb12u1_all.deb …
Распаковывается firefox-esr-l10n-ru (128.7.0esr-1~deb12u1) на замену (128.6.0es
r-1~deb12u1) …
Подготовка к распаковке …/firefox-esr_128.7.0esr-1~deb12u1_amd64.deb …
Оставляется «отклонение /usr/bin/firefox в /usr/bin/firefox.real из-за firefox-
esr»
Распаковывается firefox-esr (128.7.0esr-1~deb12u1) на замену (128.6.0esr-1~deb1
2u1) …
Настраивается пакет firefox-esr (128.7.0esr-1~deb12u1) …
Настраивается пакет firefox-esr-l10n-ru (128.7.0esr-1~deb12u1) …
Обрабатываются триггеры для mailcap (3.70+nmu1) …
Обрабатываются триггеры для desktop-file-utils (0.26-1) …
Обрабатываются триггеры для hicolor-icon-theme (0.17-2) …
Обрабатываются триггеры для mate-menus (1.26.0-3) …
Обрабатываются триггеры для man-db (2.11.2-2) …
Сканирование процессов...
Сканирование образов linux...

Запущено ядро последней версии.

Службы не требуют перезапуска.

Контейнеры не требуют перезапуска.

В сеансах пользователей нет устаревших
 процессов.

No VM guests are running outdated hypervisor (qemu) binaries on this host.

root@s10:~# apt full-upgrade
Чтение списков пакетов… Готово
Построение дерева зависимостей… Готово
Чтение информации о состоянии… Готово
```

```
Расчёт обновлений… Готово
Обновлено 0 пакетов, установлено 0 новых пакетов, для удаления отмечено 0
пакетов, и 0 пакетов не обновлено.
```



В работе с обновлениями могут возникнуть трудности. Которые надо заранее учитывать перед запуском обновления.

Не всегда гарантирована возможность обновить версию ОС. Например Centos не гарантирует, что вы сможете выполнить обновление версии с 6 на 7 или, что после обновления у вас не возникнут трудности. В других ОС процесс обновления версии хорошо отработан и почти всегда проходит без осложнений. Но надо помнить, что изменение версии это всегда повышенный риск того, что у вас что-нибудь после обновления не заработает.

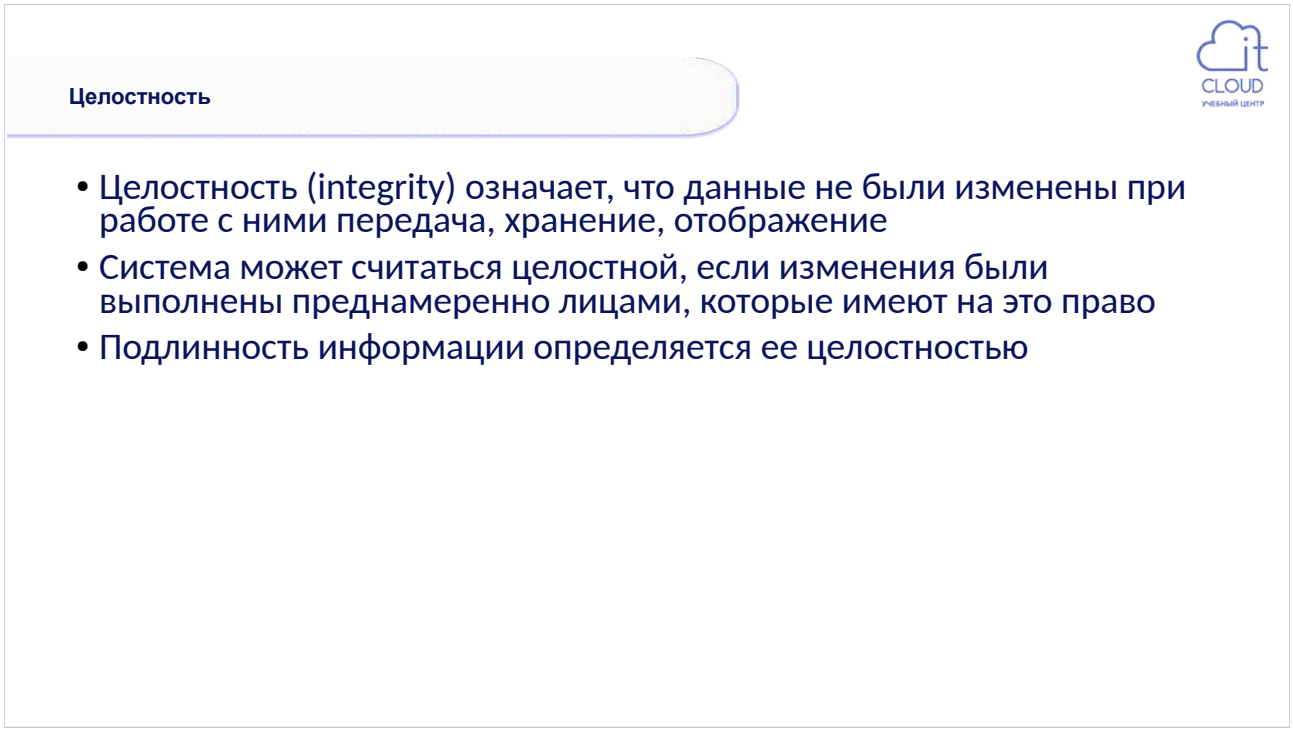
Если вы используете нестандартные репозитории, то нет гарантии что обновления не вызовут проблем. Майнтайнеры этих репозиториев не всегда могут отследить изменения в дистрбутивах, для которых они публикуют свои пакеты.

Еще одна потенциальная проблема это софт, который был установлен из исходных кодов. Обновления могут нарушить нормальное функционирование таких программ.

Для предупреждения таких проблем необходимо тестировать обновления на непроизводственных системах. Необходимо разрабатывать план установки обновления, а так же план аварийного восстановления на случай, если обновление будет неудачным.

Глава 12. Контроль целостности.

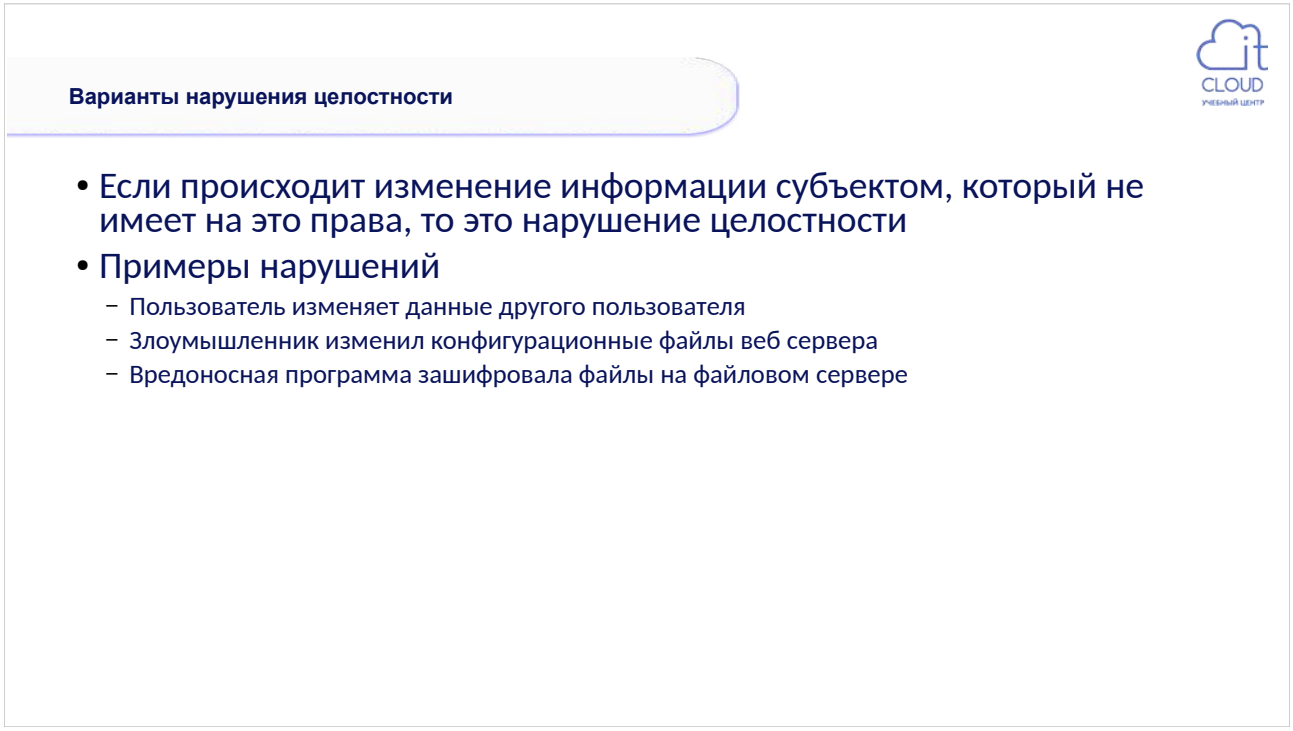
12.1 Возможные варианты нарушения целостности системы.



Целостность (integrity) одно из основополагающих понятий в безопасности. Целостность данных означает, что при работе с информацией ее передача, хранение или отображение данные не были изменены.

Целостность операционной системы подразумевает возможность изменения каких-то данных, но эти изменения могут производиться только уполномоченными на это лицами.

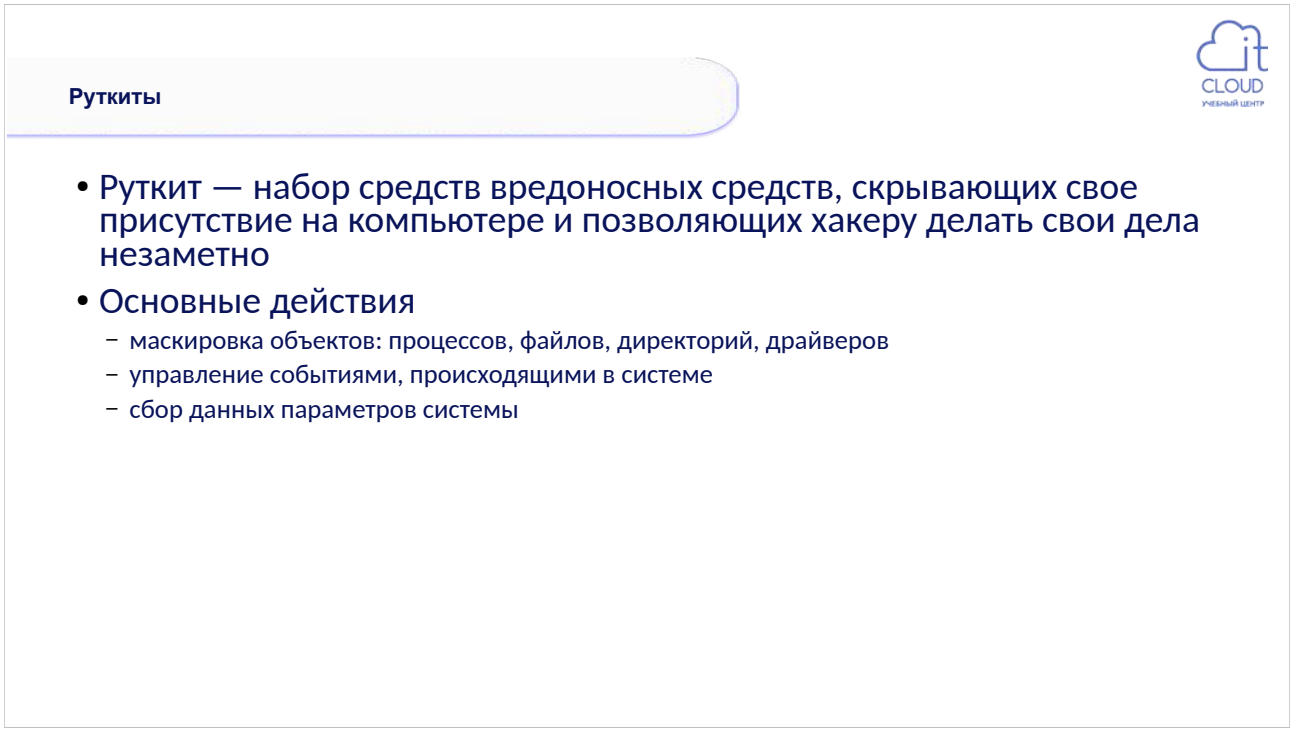
Понятие подлинности тесно связанно с целостностью. Информация может считаться подлинной, если мы знаем источник этой информации. Данные, которые были изменены, фактически имеют новый источник. Если новый источник неизвестен (нет ссылки на источник), вопрос об изменении данных не может быть разрешён. Таким образом, механизмы проверки целостности данных обеспечивают проверку их подлинности и наоборот.



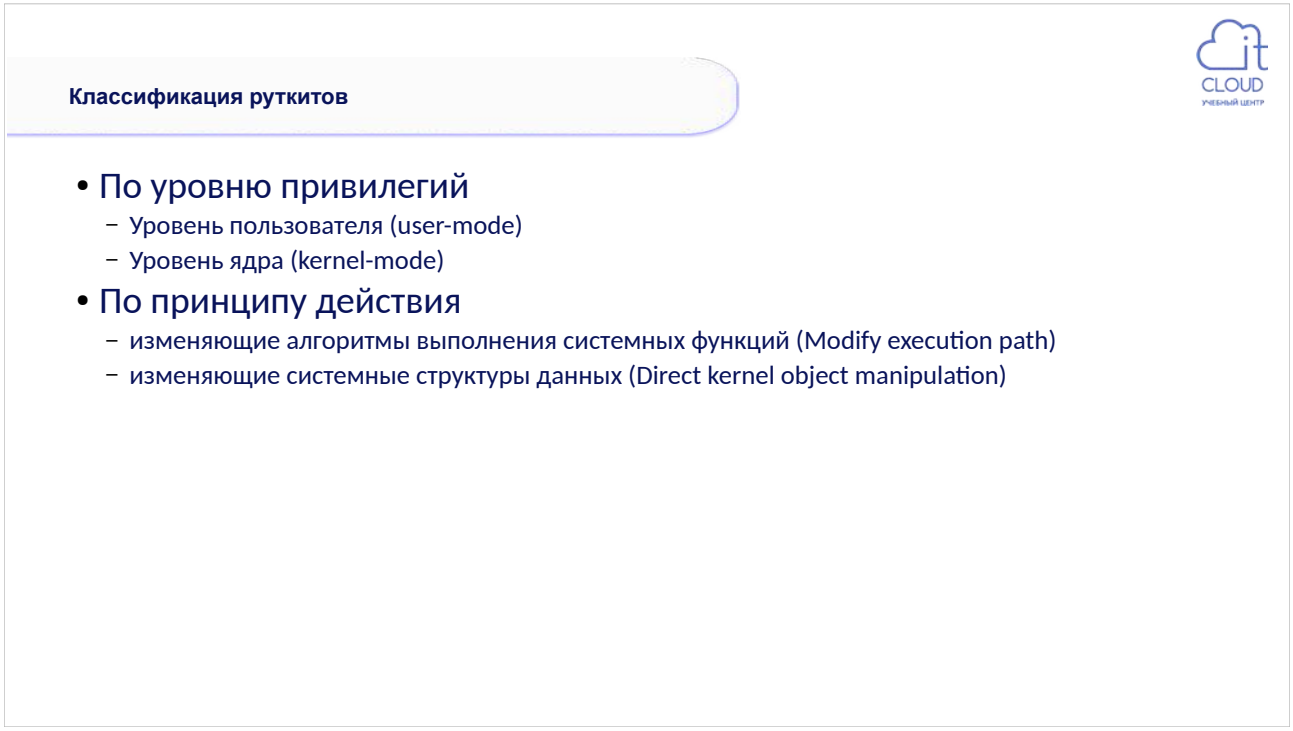
Целостность будет нарушена, если субъект, который не имеет права на изменение информации такое изменение произведет. Вариантов таких нарушений может быть огромное количество. Ниже перечислены некоторые примеры:

- В результате неправильной настройки разрешений на доступ к файлам один пользователь изменяет данные другого пользователя.
- Злоумышленник изменил конфигурационные файлы веб сервера.
- Вредоносная программа на клиентском компьютере зашифровала файлы на файловом сервере.

12.2 “Руткиты”, классификация, способы внедрения в систему.

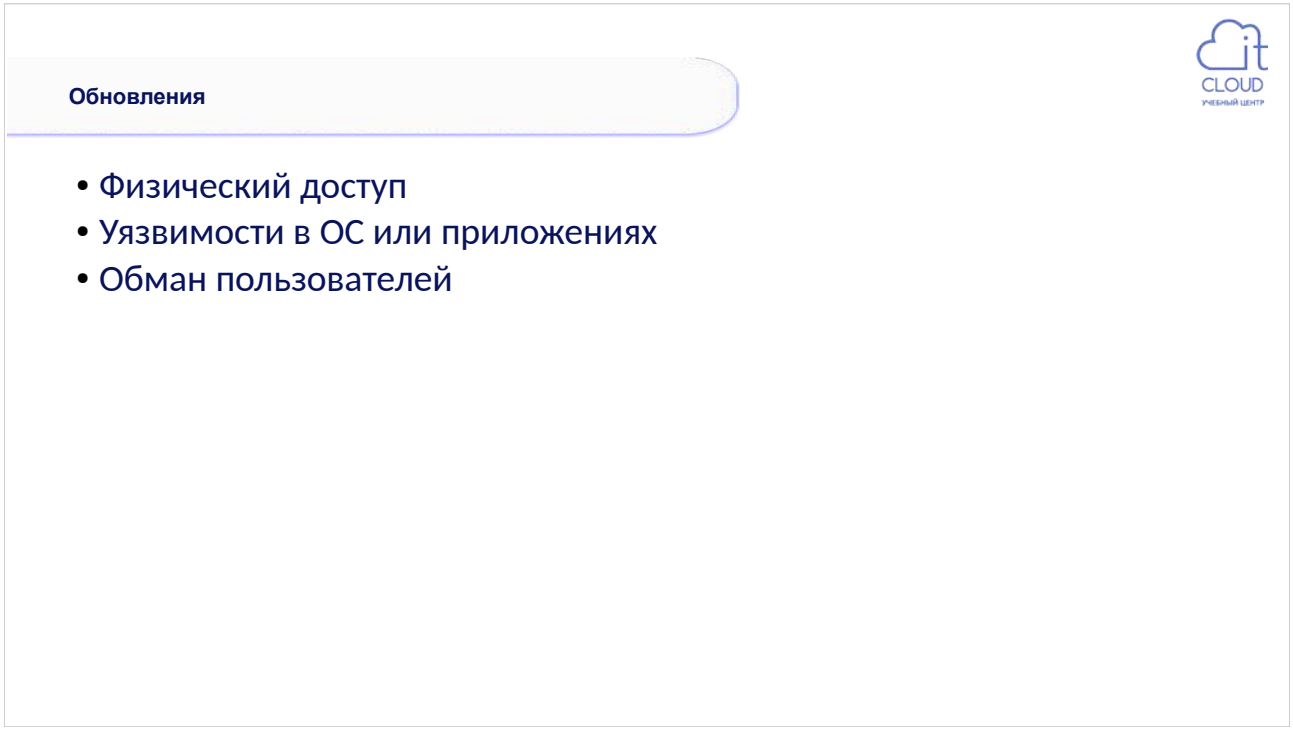


Под этим термином Руткит (Rootkit) понимается набор утилит или специальный модуль ядра, которые злоумышленник устанавливает на взломанной им компьютерной системе сразу после получения прав суперпользователя. Этот набор, как правило, включает в себя разнообразные утилиты для «заметания следов» вторжения в систему, делает незаметными снифферы, сканеры, кейлоггеры, троянские программы, замещающие основные утилиты UNIX (в случае не ядерного руткита). Rootkit позволяет взломщику закрепиться во взломанной системе и скрыть следы своей деятельности путём скрытия файлов, процессов, а также самого присутствия руткита в системе.



Руткиты классифицируются по уровню привилегий и по принципу действия.

- По уровню привилегий
  - Уровень пользователя (user-mode)
  - Уровень ядра (kernel-mode)
- По принципу действия
  - изменяющие алгоритмы выполнения системных функций (Modify execution path)
  - изменяющие системные структуры данных (Direct kernel object manipulation)



Способы проникновения руткитов в систему:

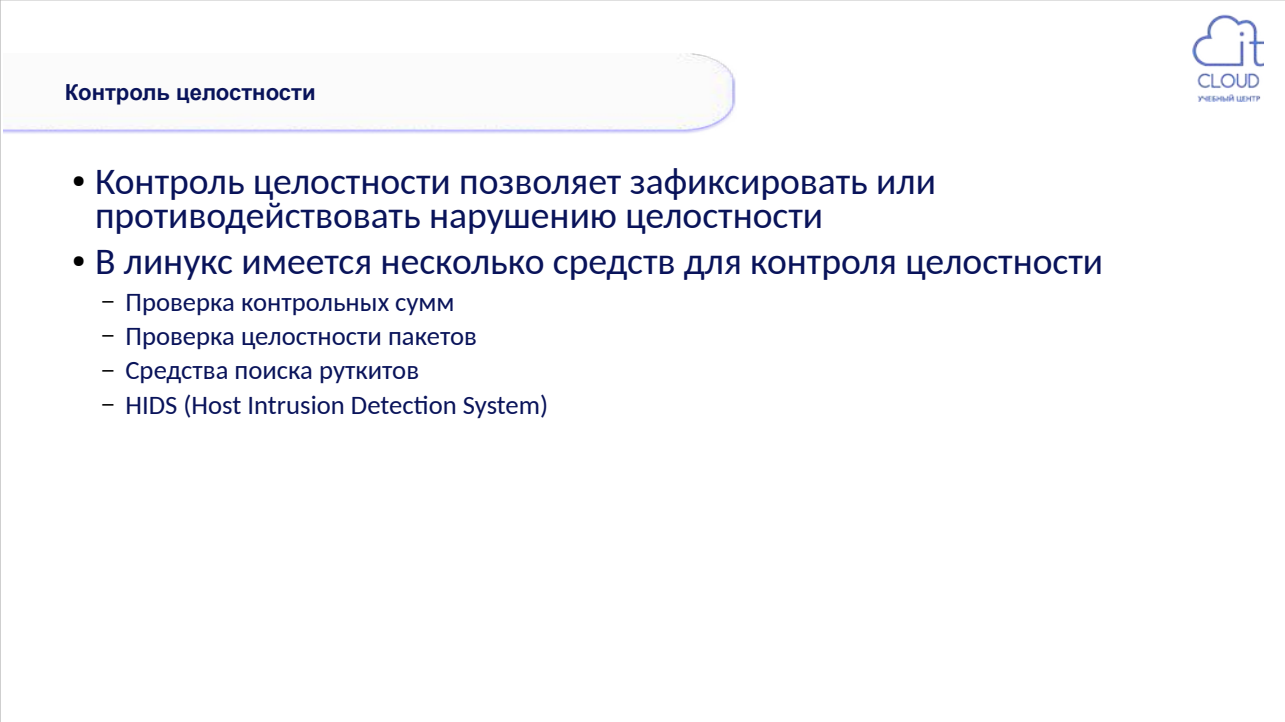
- Физический доступ. Получив физический доступ злоумышленник сможет выполнить любые действия в системе.
- Уязвимости в ОС или приложениях. Если будет обнаружена версия не обновленной ОС или приложений на компьютере, то злоумышленник может использовать известные уязвимости.
- Обман пользователей. Электронная почта, подмена сайтов.

Пример: сканирование удаленного узла.

```
[root@vlesk-nb ~]# nmap -O 10.255.255.100

Starting Nmap 7.40 ( https://nmap.org ) at 2017-08-31 16:25 +05
Nmap scan report for 10.255.255.100
Host is up (0.00049s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
111/tcp  open  rpcbind
3306/tcp open  mysql
MAC Address: 52:54:00:C0:6A:9E (QEMU virtual NIC)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.6
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.70 seconds
```

```
[root@vlesk-nb ~]# nmap -sV 10.255.255.100

Starting Nmap 7.40 ( https://nmap.org ) at 2017-08-31 16:30 +05
Nmap scan report for 10.255.255.100
Host is up (0.000027s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 6.6.1 (protocol 2.0)
111/tcp  open  rpcbind 2-4 (RPC #100000)
3306/tcp open  mysql   MariaDB (unauthorized)
MAC Address: 52:54:00:C0:6A:9E (QEMU virtual NIC)

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.45 seconds
```

12.3 Контроль целостности как механизм защиты.



Контроль целостности позволяет либо зафиксировать либо противодействовать нарушению целостности. В основном для проверки целостности используются хэш функции на основе, которых вычисляются MAC (Message Authentication Code).

Средств противодействия или выявления в ОС Linux огромное количество. Так или иначе почти все средства обеспечения безопасности призваны защищать целостность.

Среди всех средств можно выделить несколько категорий:

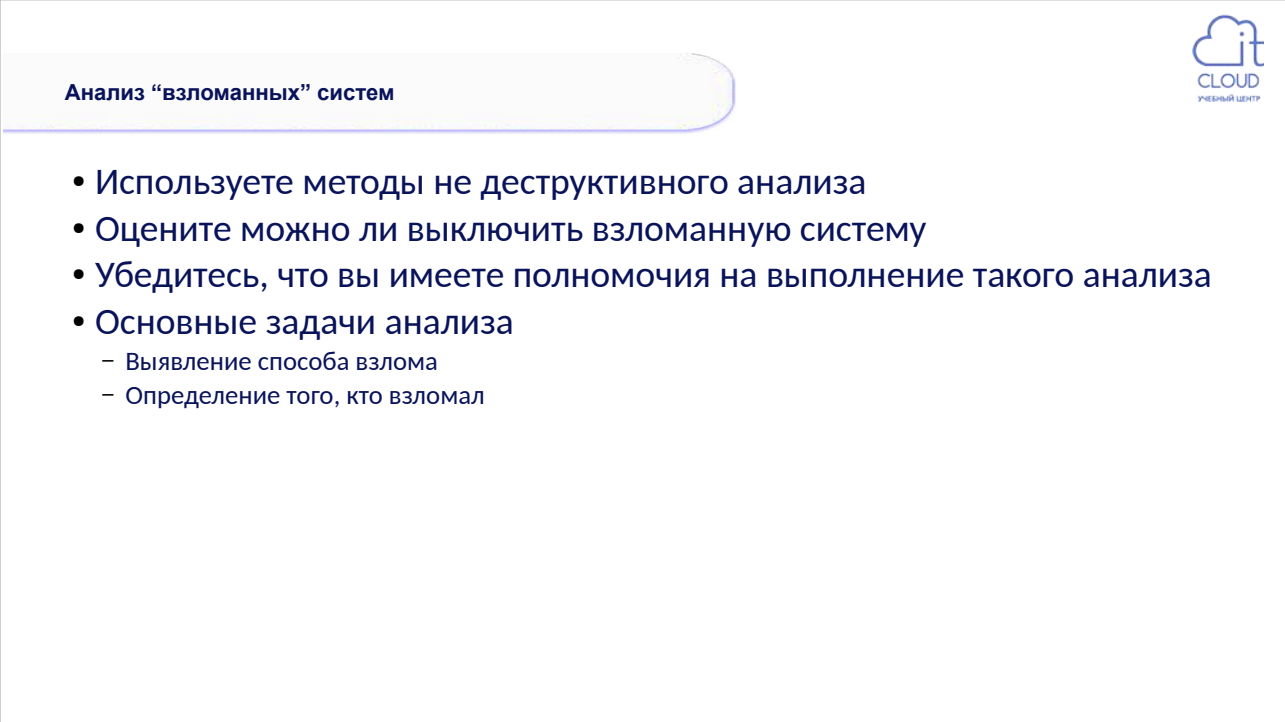
- Средства проверки контрольных сумм md5sum или shasum. Эти утилиты следует применять, когда вы, например, скачиваете исходные коды для проверки полученного архива.
- В некоторые системах пакетов имеется встроенный механизм проверки файлов, установленных из пакетов. Например в RPM.
- Специализированные средства поиска руткитов, такие как rkhunter.
- Системы обнаружения вторжений уровня хоста, например Samhain.

Пример:

```
root@s10:~# rkhunter --check
[ Rootkit Hunter version 1.4.6 ]

Checking system commands...

  Performing 'strings' command checks
    Checking 'strings' command                               [ OK ]
<--->
```

```
System checks summary
=====================

File properties checks...
    Files checked: 147
    Suspect files: 1

Rootkit checks...
    Rootkits checked : 497
    Possible rootkits: 1

Applications checks...
    All checks skipped

The system checks took: 5 minutes and 12 seconds

All results have been written to the log file: /var/log/rkhunter.log

One or more warnings have been found while checking the system.
Please check the log file (/var/log/rkhunter.log)

root@s10:~# grep Warning /var/log/rkhunter.log
[22:25:37]   /usr/bin/lwp-request                              [ Warning ]
[22:25:37] Warning: The command '/usr/bin/lwp-request' has been replaced by a
script: /usr/bin/lwp-request: Perl script text executable
[22:28:10]   Checking for suspicious (large) shared memory segments [ Warning ]
[22:28:10] Warning: The following suspicious (large) shared memory segments have
been found:
[22:29:31]   Checking if SSH root access is allowed          [ Warning ]
[22:29:31] Warning: The SSH configuration option 'PermitRootLogin' has not been
set.

root@s10:~# grep -A5 'shared memory segments' /var/log/rkhunter.log
[22:28:10]   Checking for suspicious (large) shared memory segments [ Warning ]
[22:28:10] Warning: The following suspicious (large) shared memory segments have
been found:
[22:28:10]          Process: /usr/sbin/lightdm-gtk-greeter    PID: 1109
Owner: lightdm    Size: 32MB (configured size allowed: 1,0MB)
[22:28:10]
[22:28:10] Info: Starting test name 'trojans'
[22:28:10] Performing trojan specific checks
[22:28:11]   Checking for enabled inetd services             [ Skipped ]
```

12.4 Анализ “взломанных” систем.



Основные принципа анализа взломов.

Взломанная система содержит в себе следы взлома. Эти следы помогут вам определить:

- Каким образом был произведен взлом.
- Кто взломал систему.

Таким образом анализ не должен повредить эти следы, поэтому используйте методы не деструктивного анализа системы. Анализ производите на копии оригинальных данных. Например создайте образ диска взломанной системы.

- Если взломали виртуальную машину, то вы можете сохранить состояние виртуальной машины для анализа хранилища и памяти.
- Если взломали обычный компьютер, то сохранение состояния памяти может быть проблематичным. Но вы всегда можете сделать образы дисков.

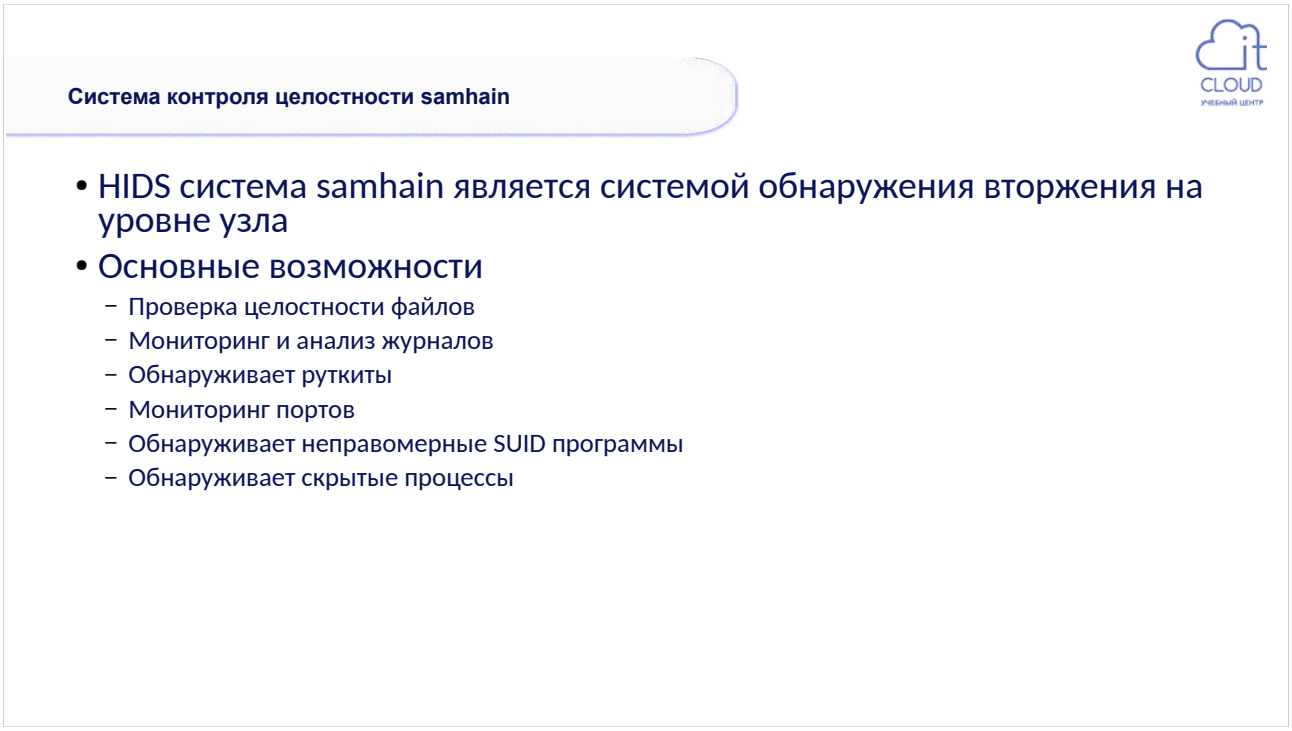
Если вы подозреваете, что ваша система взломана, то оцените возможность ее немедленного выключения. К сожалению, пока мы не знаем что произошло, мы не можем оценить все последствия выключения системы. Желательно заблокировать сетевой доступ скомпрометированной системе, если мы не можем ее выключить.

Оффлайн анализ может не дать полную картину взлома, поэтому может возникнуть необходимость запуска такой системы. Никогда не запускайте оригинальную систему, только ее копию. Здесь вам помогут образы дисков или замороженные состояния виртуальных машин. Использование копий не ограничивает количество попыток анализа.

Для онлайн анализа создайте контролируемую среду.

Еще одним аспектом анализа является полномочия на выполнения этого. Удостоверьтесь, что вы имеете на это юридические полномочия. Анализ может быть использован, в том числе, в расследовании уголовных дел. Для такого анализа требуется специальное разрешение, что бы выводы имели юридическую силу.

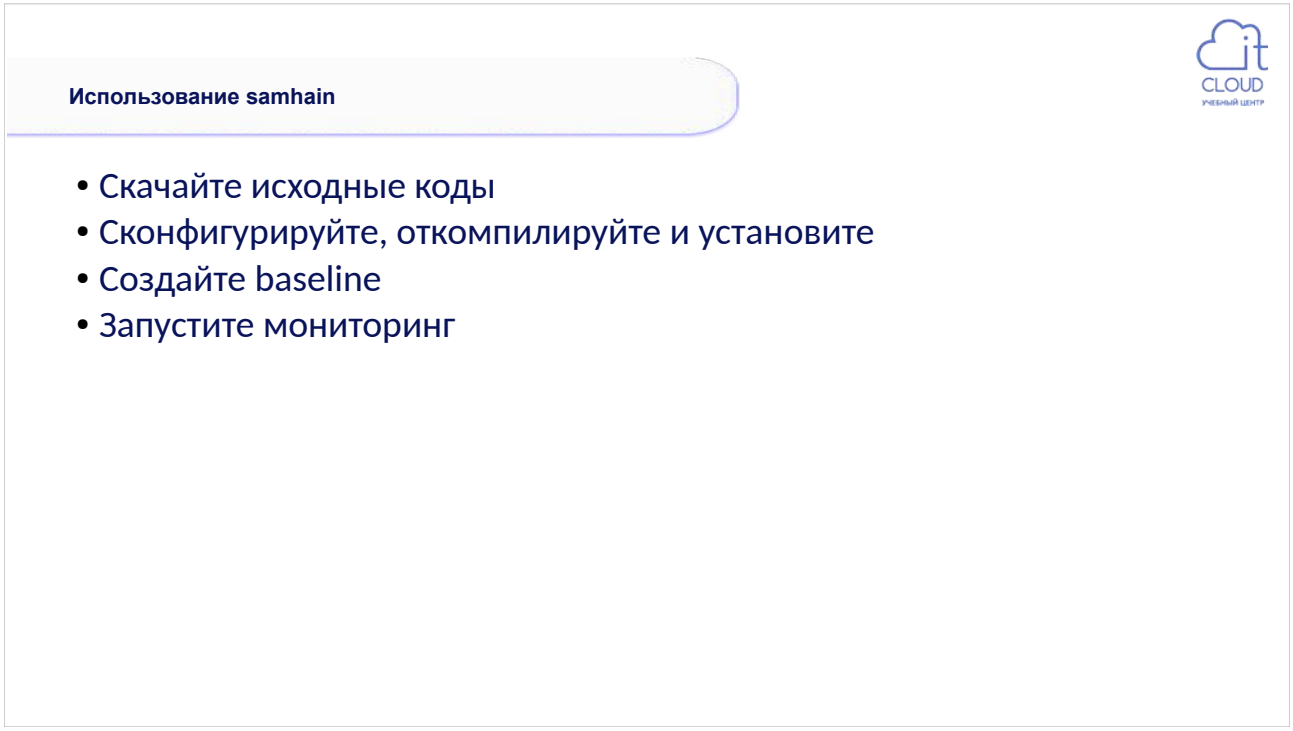
12.5 Система контроля целостности samhain.



Система samhain является системой обнаружения вторжения на уровне узла HIDS (Host-based Intrusion Detection System). Samhain может работать самостоятельно или в режиме клиента-сервера. Имеется так же веб интерфейс взаимодействия с samhain, под названием Beltane. http://www.la-samhna.de

Основные возможности samhain

- Проверка целостности файлов.
- Мониторинг и анализ журналов.
- Обнаруживает руткиты.
- Мониторинг портов.
- Обнаруживает неправомерные SUID программы.
- Обнаруживает скрытые процессы.



Установка samhain из исходных кодов является стандартной и рекомендованной создателями. Надо скачать, распаковать, запустить скрипт конфигурации, скомпилировать и установить.

Пример:

```
sa@s10:~$ wget http://la-samhna.de/samhain/samhain-current.tar.gz
sa@s10:~$ tar xf samhain-current.tar.gz
sa@s10:~$ tar xf samhain-4.5.2.tar.gz
sa@s10:~$ cd samhain-4.5.2/
sa@s10:~/samhain-4.5.2$ ./configure
sa@s10:~/samhain-4.5.2$ make
sa@s10:~/samhain-4.5.2$ sudo make install
sa@s10:~/samhain-4.5.2$ sudo make install-boot
```

Важным этапом является создание baseline, который будет использоваться в качестве основы для анализа системы.

Пример: создание baseline.

```
sa@s10:~/samhain-4.5.2$ sudo samhain -t init
---
NOTICE :  [2025-02-09T14:21:33+0500] msg=<Finished writing baseline database.>
ALERT  :  [2025-02-09T14:21:33+0500] msg=<EXIT>, program=<Samhain>,
status=<exit_success>
```

Для обновления baseline используйте параметр `update` с опцией `-t`.

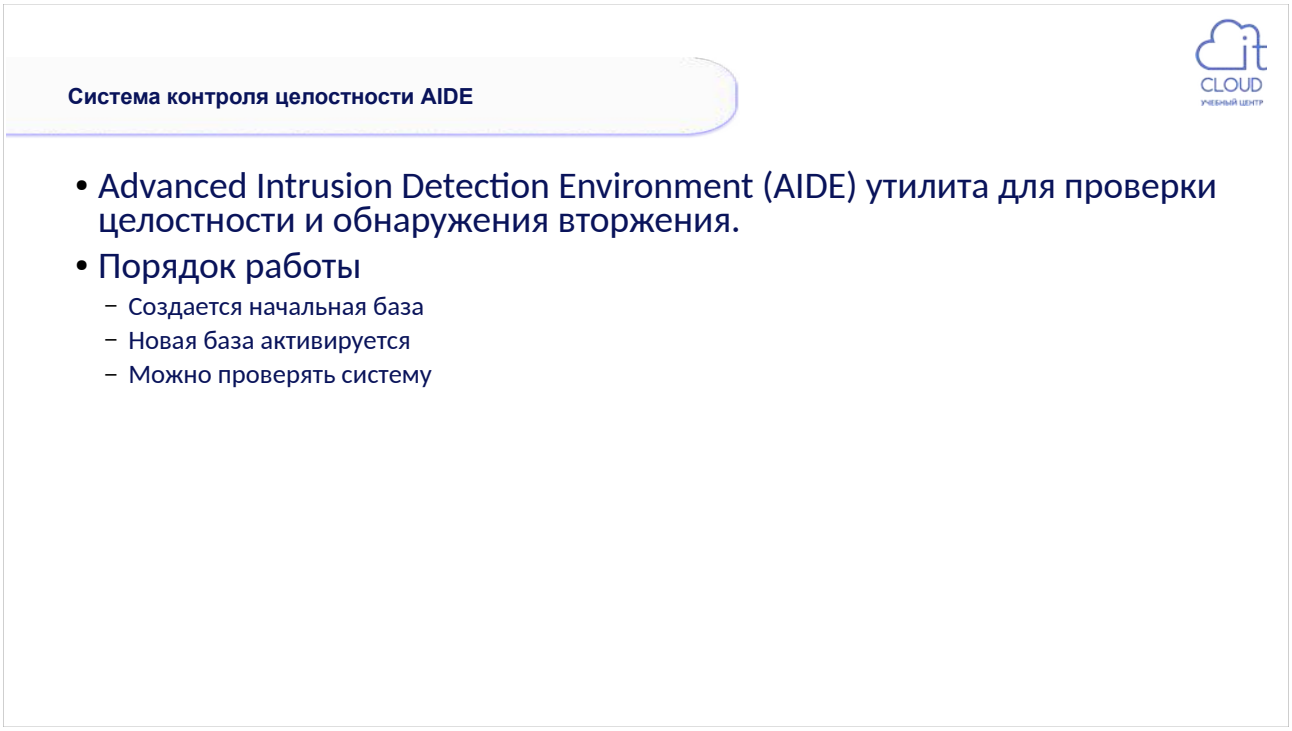
Далее вы можете начать мониторинг системы.

Пример:

```
sa@s10:~/samhain-4.5.2$ sudo systemctl start samhain
sa@s10:~/samhain-4.5.2$ sudo systemctl status samhain
```

В Debian имеется готовый пакет. При установке пакета производится первоначальная инициализация baseline и создается соответствующая служба.

12.6 Система контроля целостности файлов AIDE.



Advanced Intrusion Detection Environment (AIDE) утилита, которая создает базу файлов и позволяет отслеживать изменения с ними, для проверки целостности и обнаружения вторжения.

Принцип ее работы похож на samhain:

1. После установки создается начальная база данных.

```
root@client:~# aide -c /etc/aide/aide.conf -i
Start timestamp: 2025-02-09 13:41:16 +0500 (AIDE 0.18.3)
AIDE successfully initialized database.
New AIDE database written to /var/lib/aide/aide.db.new
Ignored e2fs attributes: EINV

Number of entries:      201528

---------------------------------------------------
The attributes of the (uncompressed) database(s):
---------------------------------------------------

/var/lib/aide/aide.db.new
 MD5       : w/yQU1LCeHIeGX6Ua2tM6A==
 SHA1      : a2j1B/unsOa4cNxCkGN8VOv3eN0=
 SHA256    : gUBo+Y9YEijR+1GD5eV1mF2la9FYhzUc
             n0K29f3kvGE=
 SHA512    : JggqucR0gcEttQ/lrOUnokFMANF1t38Z
             byYMMZJbwoiDV1xBQp6Ywu7hj1T0IMBe
             iA34irzMXsizFTkCleevNA==
 RMD160    : XAYEk07McA3wUoPE43XKc4DMtW8=
 TIGER     : 6D3sps9DJTuKDXRuFyIJH7bX2co9wN1F
 CRC32     : 4ch5Og==
 CRC32B    : nOSDew==
```

```
 HAVAL     : 76ooxy2KJ2BxpugrV2K3yJGaSrivyxMK
             Wj7z5UVRLDI=
 WHIRLPOOL : QgBebDmhO9/DsRt5+mVWPdsJdAXIeQ2/
             ZxLnJ00olGkBT2i2NeCxkMVtvB+guNCm
             874Azv/Gi+cQBXD+sIMZbA==
 GOST      : iItkvSt/w4pap8a2Yiv5kKM0WYDY8ThV
             8a4WqJG3H/Q=


End timestamp: 2025-02-09 13:51:20 +0500 (run time: 10m 4s)
```

- Новая база не используется автоматически необходимо удалить суффикс .new из ее названия.

```
root@client:~# mv /var/lib/aide/aide.db.new /var/lib/aide/aide.db
```

- Затем базу можно использовать для отслеживания изменений.

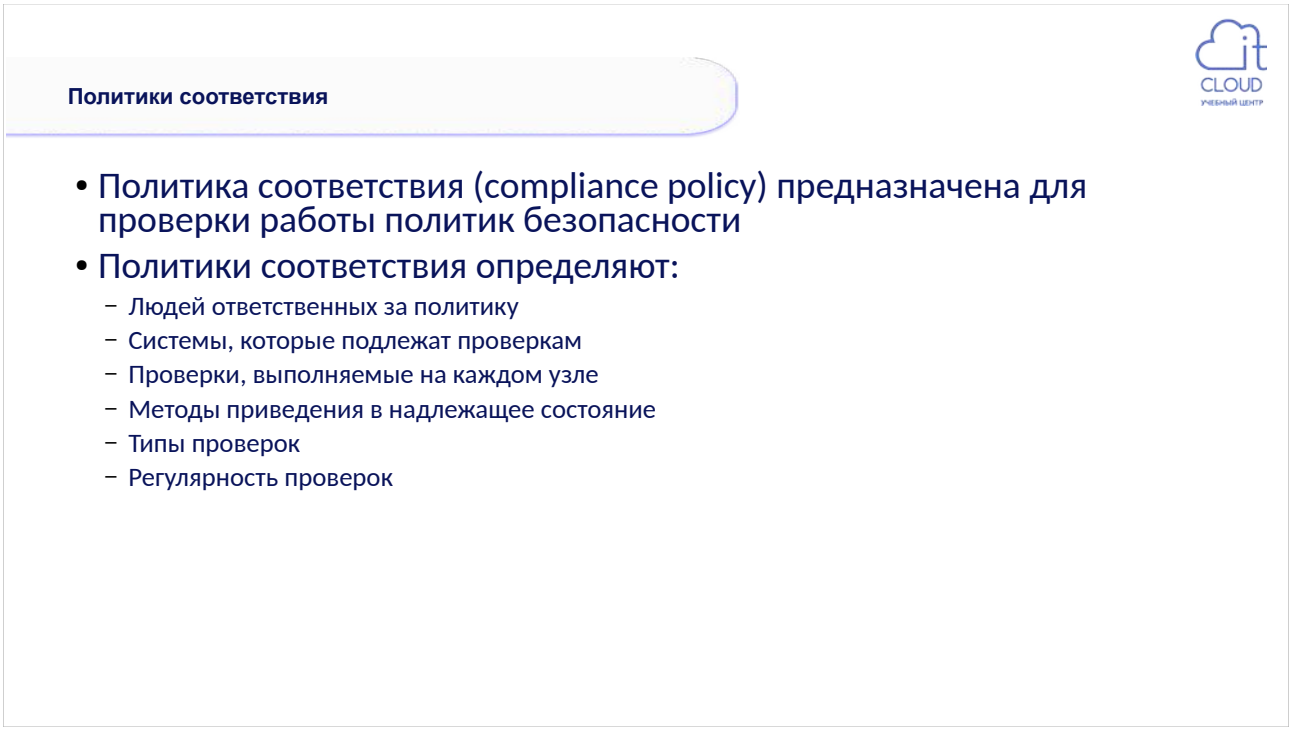
```
root@client:~# echo 1.2.3.4 somehost >> /etc/hosts

root@client:~# aide --check -c /etc/aide/aide.conf
Start timestamp: 2025-02-09 14:13:56 +0500 (AIDE 0.18.3)
AIDE found differences between database and filesystem!!
Ignored e2fs attributes: EINV

Summary:
  Total number of entries:    201528
  Added entries:        0
  Removed entries:            1
  Changed entries:            1

---------------------------------------------------
Removed entries:
---------------------------------------------------

f----------------: /var/lib/aide/aide.db.new

---------------------------------------------------
Changed entries:
---------------------------------------------------

f >.... mc..H.. . : /etc/hosts

---------------------------------------------------
Detailed information about changes:
---------------------------------------------------

File: /etc/hosts
 Size      : 186                              | 203
 Mtime     : 2025-02-06 16:03:11 +0500        | 2025-02-09 14:13:38 +0500
 Ctime     : 2025-02-06 16:03:11 +0500        | 2025-02-09 14:13:38 +0500
 MD5       : f5C2Bk/VyMPhIGPNN3VfFg==          | ZPlgdKyhLCUPFNhZz5SS2w==
 SHA1      : 2CbNXaGO0bkSz7YEQ+9CI+cZMfA=     | 5A5S4a00W6N1mKIbOQjPRo6hlaE=
 SHA256    : GwKfLcztbAZLDeNlcCW+KrNOoVe4pnZc | P9NVFu6SBIJx97T0WTDK9jrvwsbZeCh7
             8NpgPkgQCeo=                     | 5L72o/EELiE=
 SHA512    : w3t7h9+hFpWoFNdwTchRz6cqX36E/lmH | mBQcbPy91X7VzDrHq5vBNpCR/KYFzL3s
             ChNaE7lEgxcNlF1isLFSQiMxRYyd+Ra1 | GYeyo8GLeUIaRFDckNANksZRXU2xWgtz
```

```
             k6wzATK+MtFXEFC8i4qi3Q==          | 2S5cHdtK+499qnk5pcxN6Q==
 RMD160    : JcETkbTHi/KX0hPp5HDz/oet9Ck=      | hLc0QMR/whn6i6uL4qyCFftBKiQ=
 TIGER     : GbrujQdl3qYMETiNd7Zv3fhaEWTdxMKm | khetxwo4sboxETt860mbycdGxARmS1Wl
 CRC32     : gbg1fg==                         | lTc6vA==
 CRC32B    : ofEwwQ==                         | kJAWVg==
 HAVAL     : ZoTeFSgjNXIwCZKTidDfmt6Bjz5rlQxB | YdVUIjHrDPJRltGkPN1x5v8ywhEX7guC
             gBXsvuiLCKg=                     | haiUrhmcUR0=
 WHIRLPOOL : R7ySL9l7xKLjQJicnGUN88W7oywtIhvx | rDXqqcQF0f8yVQq11i0AxSgZOaCGGem3
             IlhblkyNfewbL4KrlacTXbSq5mdKvVQy | mltZinJPk3qrJSPUU4SGRIRqQ+JLDe1X
             +2t5+nj8TZeKVqOIBzqj5w==         | xG6J/tIGY93g7dLQdS1PUQ==
 GOST      : jk8xrNa9PPgbteeX11MVcnLf5uwNAMcN | RFm8HQimmxkSeD81cjDTYi6qYMMh0LtB
             2NW8l8ZtDAY=                     | bQF2tSfXONk=


---------- Summary: ----------
The attributes of the (uncompressed) database(s):
------------------------------

/var/lib/aide/aide.db
 MD5       : w/yQU1LCeHIeGX6Ua2tM6A==
 SHA1      : a2j1B/unsOa4cNxCkGN8VOv3eN0=
 SHA256    : gUBo+Y9YEijR+1GD5eV1mF2la9FYhzUc
             n0K29f3kvGE=
 SHA512    : JggqucR0gcEttQ/lrOUnokFMANF1t38Z
             byYMMZJbwoiDV1xBQp6Ywu7hj1T0IMBe
             iA34irzMXsizFTkCleevNA==
 RMD160    : XAYEk07McA3wUoPE43XKc4DMtW8=
 TIGER     : 6D3sps9DJTuKDXRuFyIJH7bX2co9wN1F
 CRC32     : 4ch5Og==
 CRC32B    : nOSDew==
 HAVAL     : 76ooxy2KJ2BxpugrV2K3yJGaSrivyxMK
             Wj7z5UVRLDI=
 WHIRLPOOL : QgBebDmhO9/DsRt5+mVWPdsJdAXIeQ2/
             ZxLnJ00olGkBT2i2NeCxkMVtvB+guNCm
             874Azv/Gi+cQBXD+sIMZbA==
 GOST      : iItkvSt/w4pap8a2Yiv5kKM0WYDY8ThV
             8a4WqJG3H/Q=


End timestamp: 2025-02-09 14:28:06 +0500 (run time: 14m 10s)
```

Конфигурация `aide` находится в файле `/etc/aide/aide.conf`.

Глава 13. Контроль защищенности Linux – систем.

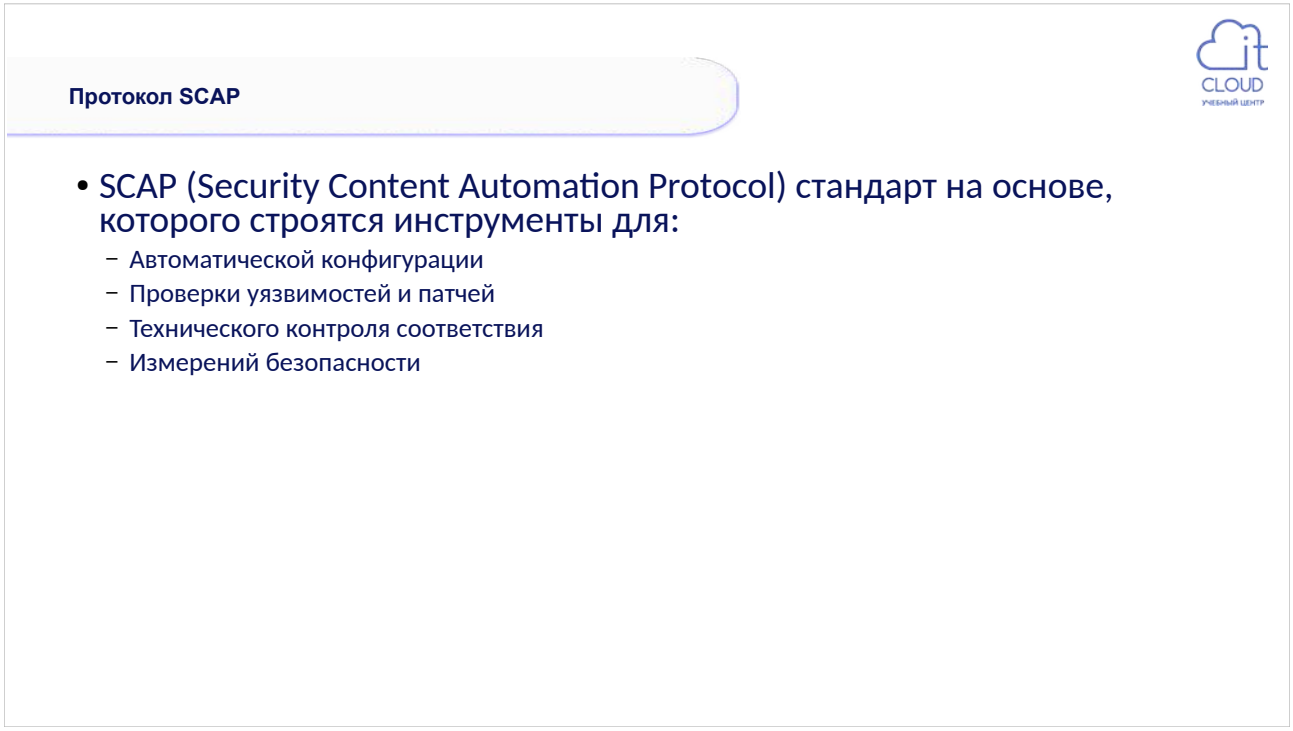
13.1 Контроль соответствия политикам безопасности.



Создать и применить политику безопасности не достаточно, для того, чтобы считать систему защищенной. Важно еще контролировать применение политик безопасности, поэтому необходимо разработать и внедрить политику соответствия политикам безопасности (compliance policy). Вы не можете считать свою политику безопасности успешной, если не знаете как она работает.

Такие политики включают в себя следующие элементы:

- Лица, которые отвечают за разработку, внедрение, применение и анализ работы политик соответствия.
- Системы, которые подлежат аудиту соответствия.
- Какие элементы политик применяются на этих системах.
- Как системы приводятся в надлежащее состояние, если не соответствуют политикам.
- Будут ли использоваться автоматизированные или ручные проверки или и те и другие.
- Как регулярно следует проверять системы.

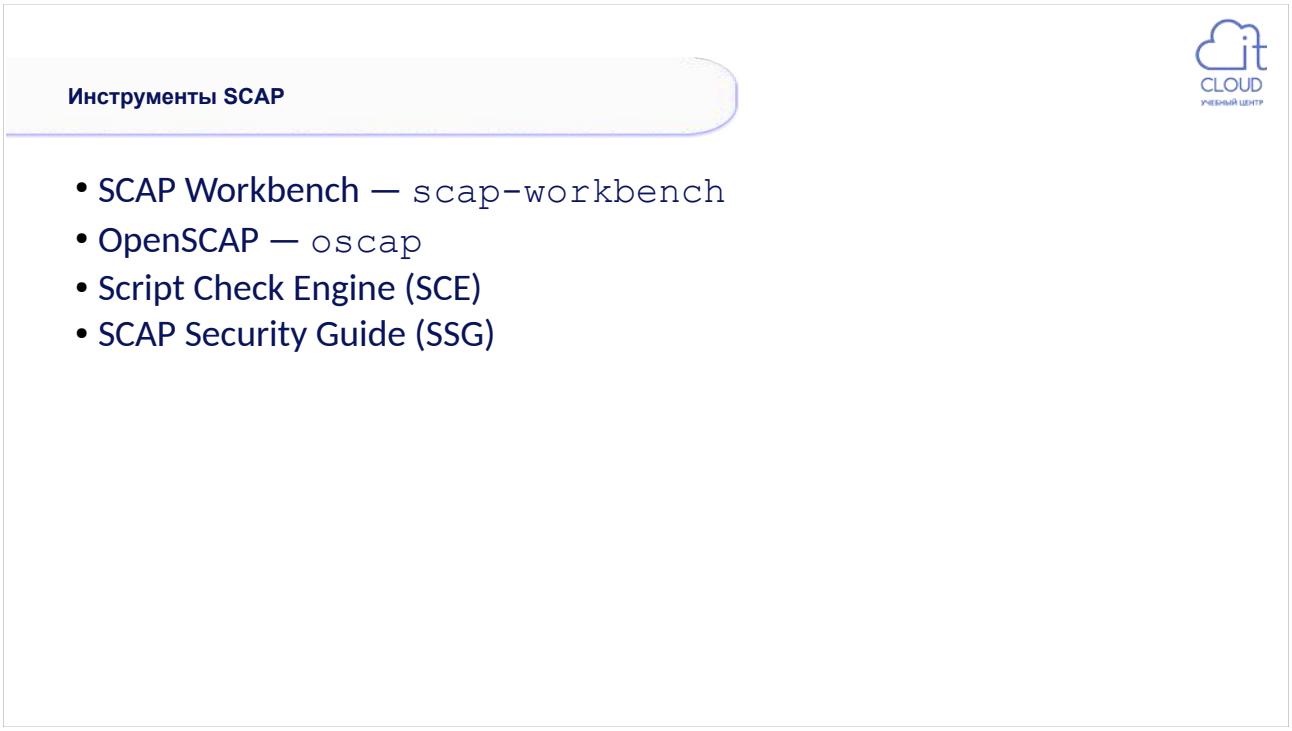


Политики соответствия не берутся из воздуха. Имеется множество стандартов и рекомендаций о том, как необходимо строить политики соответствия. Одним из таких стандартов является SCAP (Security Content Automation Protocol). На основе этого стандарта разработано несколько инструментов для:

- Автоматической конфигурации систем.
- Проверки на наличие уязвимостей и патчей.
- Технического контроля соответствия систем политикам.
- Измерений безопасности.

Сайт проекта https://www.open-scap.org

13.2 Инструментарий для выполнения проверок.



В рамках проекта OpenSCAP разработано несколько инструментов для поддержки политик безопасности.

- SCAP Workbench — `scap-workbench` графическая утилита для проверок настроек и уязвимостей в системе. Работает локально и удаленно. Может так же применяться для корректировки системы. В Debian 12 этой утилиты нет, только экспериментальная версия. В предыдущих версиях была.
- OpenSCAP — oscap основная утилита для выполнения проверок в командной строке.
- Script Check Engine (SCE) — расширение для создания своих политик.
- SCAP Security Guide (SSG) — обеспечивает последний набор политик для Линукс систем.

SCAP Workbench

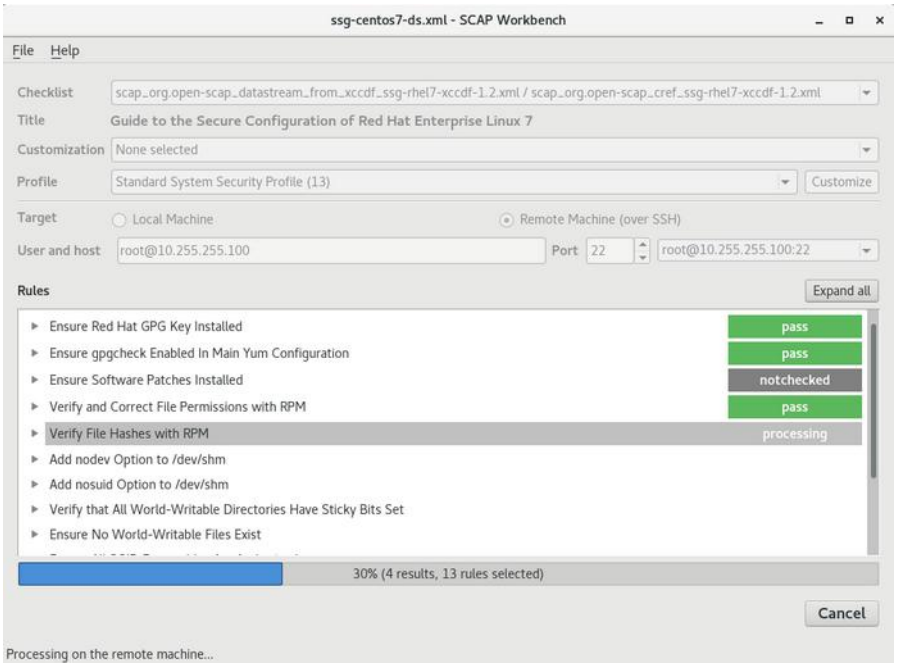
- `scap-workbench` графическая утилита для выполнения проверок
- Для работы `scap-workbench` требуется SCAP Security Guide
- Перед выполнением проверки выберите профиль проверки

Графическая утилита `scap-workbench` работает на Redhat подобных дистрибутивах и выполняет проверки на локальных или удаленных машинах. Может попытаться исправить найденные проблемы. Проверять можно не только RedHat системы.

Для того чтобы выполнить проверку `scap-workbench` загружает профиль проверки. Профили устанавливаются отдельным пакетом.

Пример:

```
# dnf install scap-security-guide scap-workbench
```



Для выполнения проверок на удаленной машине должен быть установлен пакет openscap-scanner.

В актуальных версиях Debian (начиная с 10) пакета scap-workbench нет. Есть только экспериментальная версия. В будущем возможно появится готовый пакет. Поэтому необходимо будет скомпилировать пакет вручную.

Пример:

```
$ wget https://github.com/OpenSCAP/scap-workbench/releases/download/1.2.1/scap-workbench-1.2.1.tar.bz2

$ sudo apt install build-essential openssh-client libopenscap-dev
libqt5xmlpatterns5-dev ssh-askpass pkg-config asciidoc libpolkit-agent-1-0 cmake

$ tar xf scap-workbench-1.2.1.tar.bz2
$ cd scap-workbench-1.2.1/

$ $ sed -i 's/-Wall//' CmakeLists.txt
$ sed -i 's/-Werror//' CMakeLists.txt

$ mkdir build
$ cd build/

$ cmake ../

$ make
$ sudo make install

$ sudo apt install ssg-debian
```

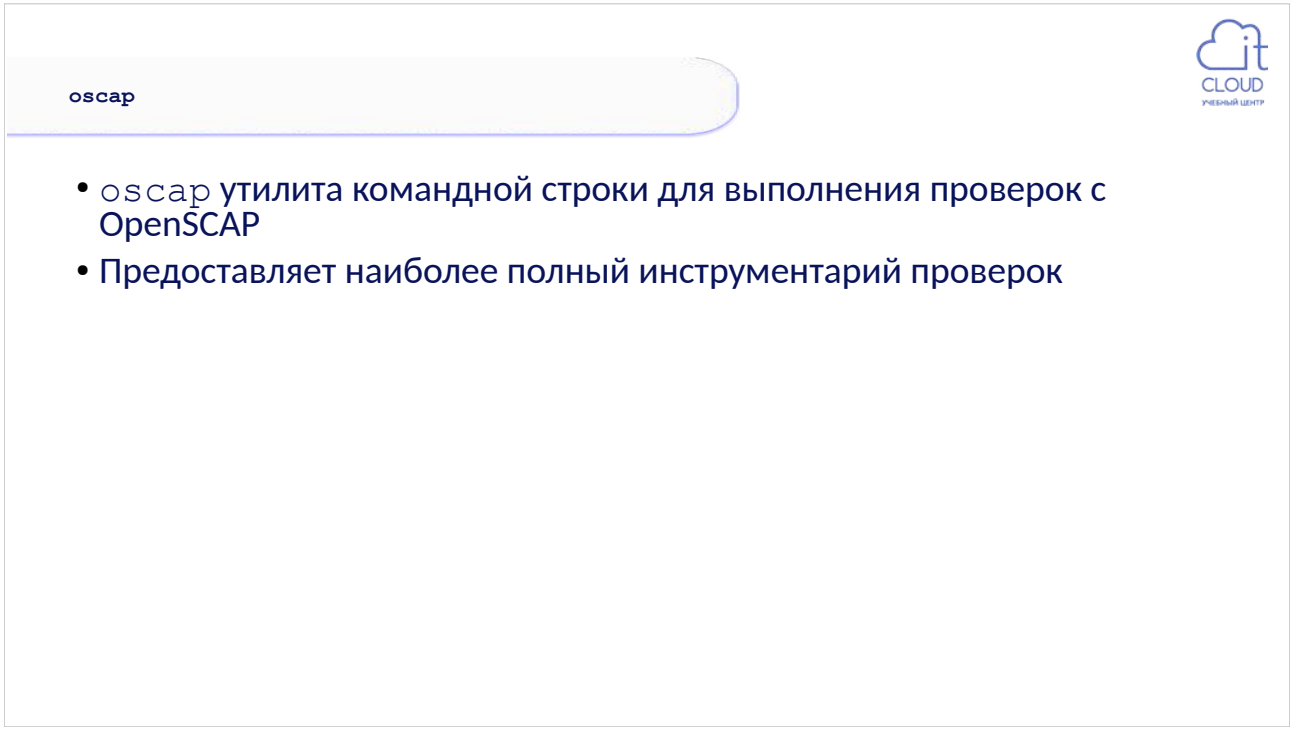
Проделав все выше перечисленное, мы увидим, что в пакете ssg-debian имеется только профили сканирования для Debian10 и Debian11, но у нас Debian12.

```
$ wget https://deb.debian.org/debian/pool/main/s/scap-security-guide/ssg-debian_0.1.74-1_all.deb
$ wget https://deb.debian.org/debian/pool/main/s/scap-security-guide/ssg-base_0.1.74-1_all.deb

$ sudo dpkg -P ssg-base ssg-debian
$ sudo apt install ./ssg-base_0.1.74-1_all.deb ./ssg-debian_0.1.74-1_all.deb
```

Теперь при открытии scap-workbench мы видим профиль сканирования для Debian 12.

Можно было и не заменять пакеты, но вытащить из пакета нужные файлы.

```
$ sudo apt purge ssg-base
$ sudo apt install ssg-debian

$ mkdir ssg-temp
$ cd ssg-temp/
sa@cl1:~/ssg-temp$ ar x ../ssg-debian_0.1.74-1_all.deb
sa@cl1:~/ssg-temp$ ls
control.tar.xz  data.tar.xz  debian-binary

$ sudo tar -tf data.tar.xz --wildcards '*debian12*' | \
 sudo tar xf data.tar.xz -T - -C /
```



oscap основная утилита для работы с OpenSCAP. Работает в командной строке.

Весь функционал OpenSCAP строится на этой утилите.

Пример:

1. Устанавливаем пакеты:

```
root@s10:~# apt -y install openscap-scanner openscap-utils bzip2
```

2. Скачиваем документ с описанием рекомендованных настроек системы, в формате OVAL (Open Vulnerability and Assessment Language) и распаковываем.

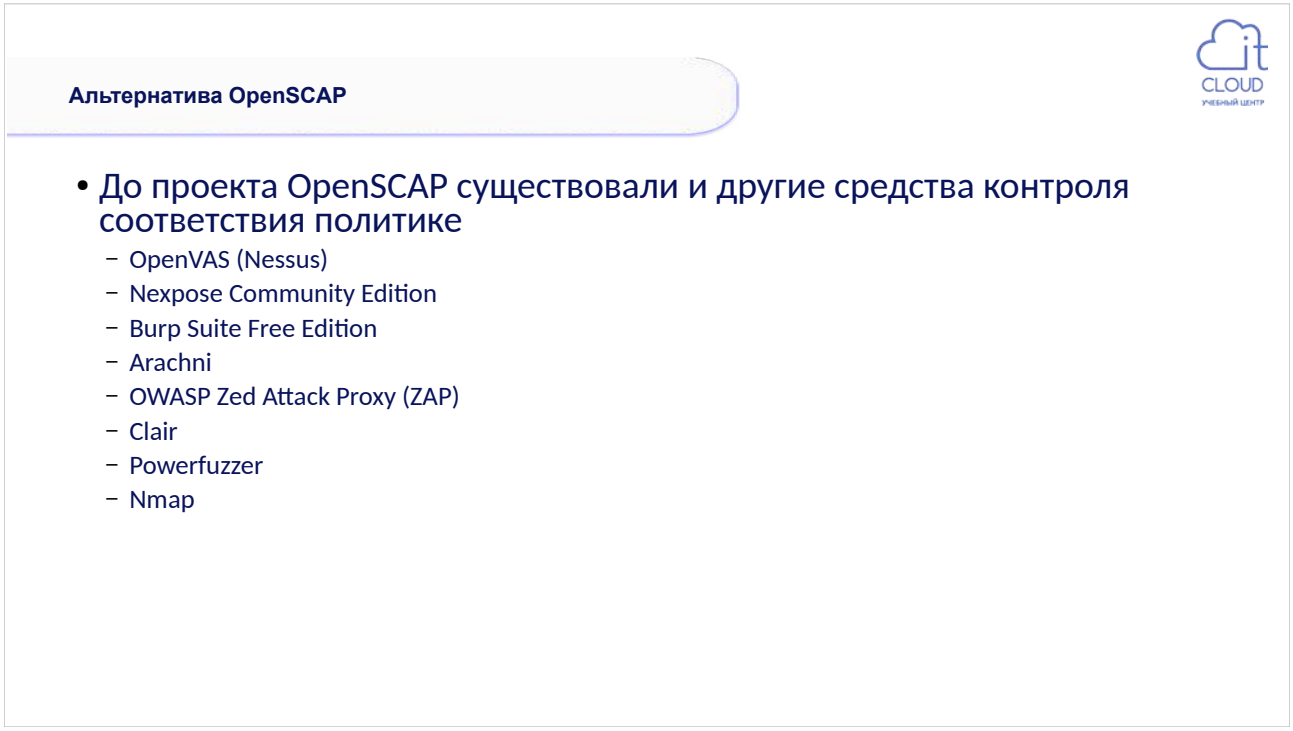
```
root@s10:~# wget wget https://www.debian.org/security/oval/oval-definitions-
bookworm.xml.bz2
root@s10:~# bunzip2 oval-definitions-bookworm.xml.bz2
```

3. Запускаем проверку:

```
root@s10:~# oscap oval eval --report /tmp/eval_result.html oval-definitions-
bookworm.xml
---
Definition oval:org.debian:def:100025136735225569795784532702130753406: false
Definition oval:org.debian:def:100020194192621893181231895146832483613: false
Evaluation done.
```

4. Проверяем результат:

```
sa@s10:~$ firefox /tmp/eval_result.html
```



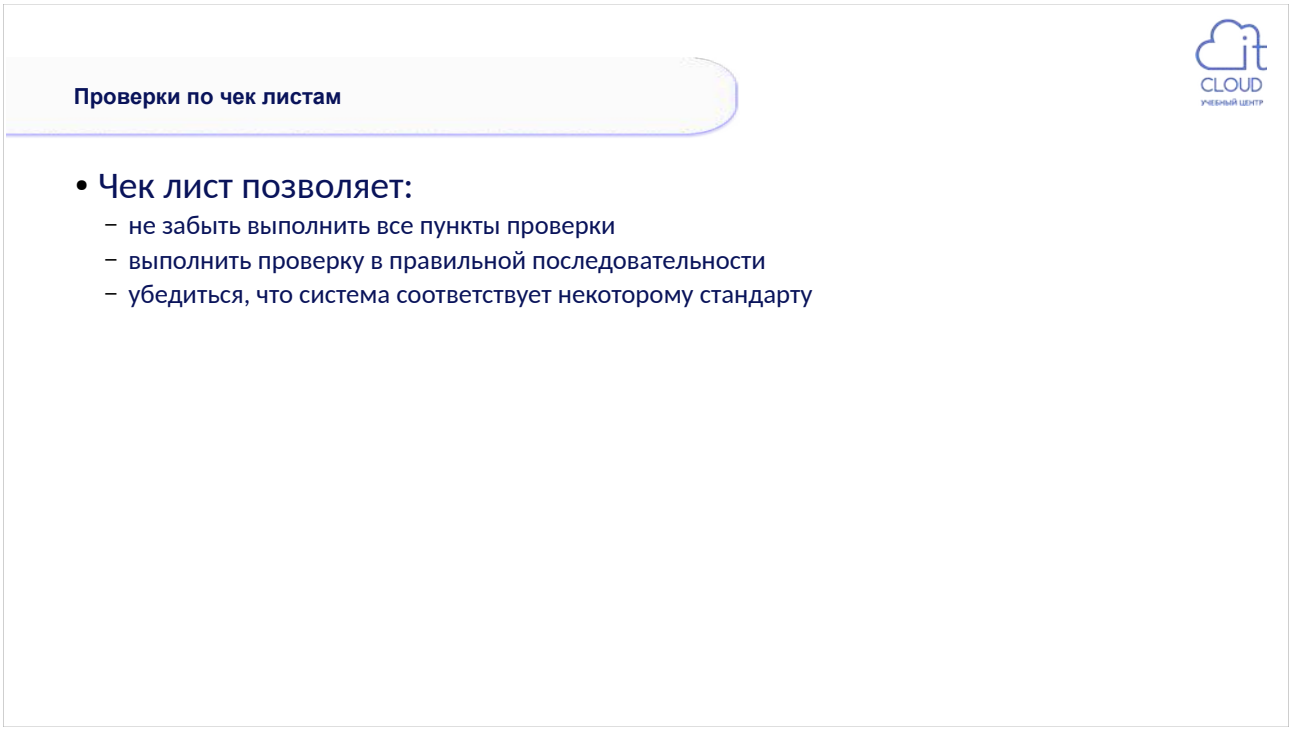
Проект OpenSCAP относительно молодой (начат в 2008). Еще до него было разработано большое количество сканеров безопасности.

- OpenVAS (Nessus) в 2005 разработчики проекта Nessus решили закрыть свой код. И сделали свой сканер платным. На его основе был создан форк проекта с именем OpenVAS.
- Nexpose Community Edition инструмент поиска уязвимостей linux с открытым исходным кодом, разрабатываемый компанией Rapid7, это та же самая компания, которая выпустила Metasploit. Версия Comunity бесплатна, но она имеет ограничение, на одновременное сканирование до 32 IP адресов и только одного пользователя.
- Burp Suite Free Edition сканер веб-уязвимостей, написанный на Java. Программа состоит из прокси-сервера, паука, инструмента для генерации запросов и выполнения стресс тестов. С помощью Burp вы можете выполнять проверку веб-приложений.
- Arachni полнофункциональный фреймворк для тестирования веб-приложений, написанный на Ruby, который распространяется с открытым исходным кодом. Он позволяет оценить безопасность веб-приложений и сайтов, выполняя различные тесты на проникновение.
- OWASP Zed Attack Proxy (ZAP) инструмент для поиска уязвимостей в веб-приложениях. Написана на Java.
- Clair инструмент поиска уязвимостей linux в контейнерах.
- Powerfuzzer полнофункциональный, автоматизированный и очень настраиваемый веб-сканер, позволяющий проверить реакцию веб-приложения на некорректные данные и повторные запросы. Инструмент поддерживает только протокол HTTP и может

обнаруживать такие уязвимости, как XSS, SQL инъекции, LDAP, CRLF и XPATH атаки.

- Nmap не совсем сканер уязвимостей для Linux. Эта программа позволяет просканировать сеть и узнать какие узлы к ней подключены, а также определить какие сервисы на них запущенны. Это не дает исчерпывающей информации про уязвимости, зато вы можете предположить какое из программного обеспечения может быть уязвимым, попытаться перебрать слабые пароли. Также есть возможность выполнять специальные скрипты, которые позволяют определить некоторые уязвимости в определенном программном обеспечении.

13.3 Проверка системы по чек листам.



Проверка разных систем обычно выполняется по множеству пунктов. Зачастую выполнение одной проверки зависит от другой. Во многих случаях необходимо проводить проверки вручную. В этом случае применение чек листов не позволит человеку выполняющему проверку что-либо забыть или пропустить.

Обычно чек лист создается, чтобы проверить систему на соответствие некоторому стандарту.