SMART technologies

SMART Linux

Операционная система SMART Linux

Руководство администратора

Данный документ является руководством администратора операционной системы SMART Linux версии 1.55 (далее SMART Linux) и описывает действия по установке, настройке, запуску и использованию операционной системы, выполняемые администратором в процессе эксплуатации операционной системы. Так же руководство администратора содержит описания:

- функций безопасного управления операционной системой;
- функций и интерфейсов, которые доступны администраторам операционной системы, связанным с администрированием;
- применения доступных администраторам функций безопасности, предоставляемых операционной системой.

Руководство администратора содержит предупреждения относительно доступных для администраторов функций и привилегий, которые следует контролировать в безопасной среде обработки информации.

Первая публикация, декабрь 2023

SMART technologies

Оглавление

1. Описание системы контроля версий

1.1. Описание и область применения операционной системы

SMART Linux является многопользовательской, многозадачной ОС, которая предоставляет платформу унифицированной функциональной универсальной доверенной среды для выполнения прикладного программного обеспечения.

SMART Linux на уровне драйверов поддерживает широкий перечень оборудования актуальных версий, доступного на рынке СВТ, а также оборудования, снятого с производства, но поддерживаемого производителями.

В SMART Linux поддерживается инсталляция с оптических носителей информации, флеш-накопителей, разделов локального жёсткого диска, а также установка по сети передачи данных.

1.2. Основные функции SMART Linux

SMART Linux может обеспечивать обслуживание от одного до нескольких пользователей одновременно. После успешного входа в систему пользователи имеют доступ в главную вычислительную среду, позволяющую запускать пользовательские приложения, создавать и получать доступ к файлам, задавать директивы пользователя на уровне оболочки командного процессора. SMART Linux предоставляет адекватные механизмы для разграничения пользователей и защиты их данных. Использование привилегированных команд ограничено и доступно только административным пользователям.

SMART Linux конфигурируется по умолчанию для работы в режиме дискреционного управления доступом (DAC).

Любой пользователь с ролью, которая позволяет выполнять административные действия, считается административным пользователем. Кроме того, SMART Linux поддерживает типы, которые могут быть связаны с объектами, и доменами, которые могут быть связаны с процессами.

С вышеизложенным определением ролей (в линуксе нету понятия ролей есть группы и пользователи) и прав доступа, подразумеваемых индивидуальными ролями, SMART Linux выполняет требования ролевого доступа.

SMART Linux предназначена для работы в сетевом окружении с другими экземплярами SMART Linux, а также с иными совместимыми серверными и клиентскими системами одного и того же управляемого домена. Все эти системы должны конфигурироваться в соответствии с определённой общей политикой безопасности.

SMART Linux разрешает использование многими пользователями одного или более процессоров, присоединённых внешних и запоминающих устройств для выполнения разнообразных функций, требующих управляемого распределенного доступа к данным, хранимым в системе. Такие инсталляции типичны для вычислительных систем рабочих групп

или предприятий, к которым обращаются локальные пользователи, или компьютерных систем с иначе защищённым доступом.

Предполагается, что ответственность за сохранение данных, защищаемых SMART Linux, может делегироваться пользователям SMART Linux. Все данные находятся под управлением механизмов безопасности SMART Linux. Данные сохраняются в поименованных объектах, и SMART Linux может связать с каждым поименованным объектом описание прав доступа к этому объекту. Всем пользователям назначаются уникальные идентификаторы. Этот идентификатор пользователя используется вместе с атрибутами и ролями, назначенными пользователю, как основание для решений по управлению доступом. SMART Linux подтверждает подлинность предъявленного идентификатора пользователя до того, как разрешать ему выполнять дальнейшие действия. SMART Linux внутри себя сопровождает ряд идентификаторов, связанных с процессами, которые получаются из уникального идентификатора пользователя, предъявляемого при входе в систему. Некоторые из этих идентификаторов могут изменяться во время выполнения процесса согласно политике, реализуемой SMART Linux.

SMART Linux предоставляет такие меры безопасности, при которых доступ к объектам данных осуществляется только в соответствии с ограничениями на доступ, наложенными на этот объект его владельцем, административными пользователями и типом объекта. Права владения на поименованные объекты могут передаваться под контролем политики управления доступом.

На объекты данных могут быть назначены дискреционные права доступа (например, чтение, запись, выполнение) субъектов (пользователей). Как только субъекту предоставляется доступ к объекту, его содержание может быть свободно использовано для воздействия на другие доступные этому субъекту объекты.

Развёрнутую ОС применяют в качестве программной платформы для разработок защищённых систем, требования к безопасности которых не превышают указанных показателей. В частности, такие требования предъявляются к защищённым программным системам, работающим с конфиденциальной информацией и персональными данными.

1.3 Состав SMART Linux

SMART Linux состоит из набора компонентов, предназначенных для реализации функциональных задач, необходимых пользователям (должностным лицам) для выполнения определенных должностными инструкциями повседневных действий, и поставляется в виде дистрибутива и комплекта эксплуатационной документации.

В структуре SMART Linux можно выделить следующие функциональные элементы:

- ядро ОС;
- системные библиотеки;
- системные приложения;
- программные серверы;
- прочие серверные программы;
- командные интерпретаторы.

Комплекс встроенных средств защиты информации, является принадлежностью операционной среды SMART Linux и неотъемлемой частью ядра ОС и системных библиотек.

Ядро ОС — программа (набор программ), выполняющая функции управления ОС и взаимодействия ОС с аппаратными средствами.

Системные библиотеки — наборы программ (пакетов программ), выполняющие различные функциональные задачи и предназначенные для их динамического подключения к работающим программам, которым необходимо выполнение этих задач.

Встроенные средства защиты информации — специальные пакеты программ ОС, входящие в состав ядра ОС и системных библиотек, предназначенные для защиты ОС от несанкционированного доступа к обрабатываемой (хранящейся) информации на ЭВМ.

Системные приложения — это приложения (программы, набор программ), предназначенные для выполнения (оказания) системных услуг пользователю при решении им определенных функциональных задач в работе с операционной средой и обеспечивающие их выполнение.

Программные серверы — специальные приложения, предназначенные для предоставления пользователю определенных услуг и обеспечивающие их выполнение.

К прочим серверным программам относятся программы, предоставляющие пользователю различные услуги по обработке, передаче, хранению информации (серверы протоколов, почтовые серверы, серверы приложений, серверы печати и прочие).

Командные интерпретаторы — специальные программы (терминалы), предназначенные для выполнения различных команд, подаваемых пользователем при работе с SMART Linux.

Прочие системные приложения — приложения (программы), оказывающие пользователю дополнительные системные услуги при работе с ОС.

В состав SMART Linux включены следующие дополнительные системные приложения:

- архиваторы;
- приложения для управления RPM-пакетами;
- приложения мониторинга системы;
- приложения для работы с файлами;
- приложения для настройки системы;
- настройка параметров загрузки;
- настройка оборудования;
- настройка сети.

1.4 Документация в составе

В составе SMART Linux представлена следующая документация:

- электронная, контекстно-зависимая справочная система.

1.5 Требования к персоналу (администратору)

1.5.1 Общие положения

Администратор SMART Linux должен иметь:

- базовые навыки администрирования ОС семейства Linux;
- навыки конфигурирования и настройки программных продуктов и ОС;
- навыки поддержания в работоспособном состоянии технических средств.

1.5.2 Обязанности пользователей, определяемые предположениями безопасности

Предопределённое использование SMART Linux

Доступ администраторов к SMART Linux должен осуществляться только из санкционированных точек доступа — рабочих мест, размещённых в контролируемой зоне, оборудованной средствами и системами физической защиты и охраны (контроля и наблюдения) и исключающей возможность бесконтрольного пребывания посторонних лиц.

Для предотвращения несанкционированного доступа к системным компонентам пользователей SMART Linux администраторы обязаны предотвращать и выявлять установку и запуск встроенных программ отладки.

Администраторы обязаны производить установку только штатных программных средств, не позволяющих осуществить несанкционированную модификацию ОС.

При взаимодействии с внешними информационными системами администраторы, при помощи средств SMART Linux, должны осуществлять настройку взаимодействия только с доверенными системами, ПБ которых скоординированы с ПБ рассматриваемой SMART Linux.

При возникновении сбоев и отказов СВТ или SMART Linux администраторы обязаны предпринимать меры, направленные на восстановление безопасного состояния программного и аппаратного обеспечения SMART Linux, в соответствии с данным Руководством.

Установка, конфигурирование и управление SMART Linux должны осуществляться администратором SMART Linux, в соответствии с настоящим документом. Самостоятельные действия по установке, конфигурированию и управлению пользователям не доступны и ограничены правилами разграничения доступа SMART Linux.

Порядок обеспечения среды функционирования SMART Linux

Администраторы должны использовать функции, предоставляемые SMART Linux, в рамках выполнения своих должностных обязанностей, определенных в должностной инструкции соответствующих категорий пользователей.

Администраторы обязаны производить настройку оборудования СВТ и предотвращать несанкционированную физическую модификацию аппаратного обеспечения, на котором выполняется SMART Linux.

Права пользователей для получения доступа и выполнения обработки информации в SMART Linux основываются на одной или более ролях и назначаются администратором SMART Linux. Роли пользователей в SMART Linux отражают производственную функцию, обязанности, квалификацию и/или компетентность пользователей в рамках организации.

По всем вопросам администрирования SMART Linux пользователь обязан обращаться к администраторам SMART Linux, которые являются компетентными, хорошо обученными и заслуживающими доверия.

Предполагается наличие (одного или более) компетентных лиц (администраторов), которые назначаются для управления безопасностью SMART Linux и информации в нем.

Кроме того, эти лица (в качестве владельцев всех корпоративных данных), наряду с владельцами объекта, должны иметь возможность назначать и отменять права доступа ролей к объектам.

Пользователи, в соответствии с назначенными в SMART Linux полномочиями и ролями, имеют права создавать новые объекты данных, владельцами которых они становятся.

Персонал, ответственный за выполнение администрирования SMART Linux, должен пройти проверку на благонадёжность и в своей деятельности должен руководствоваться документацией на SMART Linux.

Уполномоченные пользователи обладают необходимым разрешением на доступ в SMART Linux, по крайней мере, к части информации, управляемой SMART Linux, и согласованно действуют в благоприятной среде.

Администраторы в обязательном порядке должны быть ознакомлены с настоящим руководством и должны быть обучены применению функциональных возможностей безопасности, предоставляемых операционной системой.

Администраторы должны выполнять группы задач, связанных со своими служебными полномочиями, в безопасной ИТ-среде с применением полного управления своими данными.

Администраторы должны осуществлять регулярный контроль полноты и достаточности мер по обеспечению информационной безопасности на объектах, использующих SMART Linux.

1.6 Приёмка поставленного средства

Для контроля качества и приёмки SMART Linux силами потребителя производятся испытания полученного изделия. При этом производится проверка контрольных сумм дистрибутива SMART Linux. Снятие КС должно осуществляться с использованием программы фиксации и контроля исходного состояния программного комплекса ФИКС-UNIX версии 1.0, по алгоритму ГОСТ 34.11-2012 (256 бит). Полученные КС должны соответствовать эталонным КС SMART Linux, приведённым в формуляре.

Состав и последовательность испытаний:

1. проверка общих требований;
2. проверка комплектности;
3. проверка маркировки и упаковки;
4. проверка документации;
5. проверка носителей данных.

Для проверки комплектности необходимо осуществить сверку состава предъявленного на испытания изделия с комплектностью:

- SMART Linux. Дистрибутив;
- SMART Linux. Формуляр,

представленных на физических носителях или в виде электронных файлов.

Изделие считается соответствующим требованиям, если состав предъявленного на испытания изделия соответствует указанной комплектности.

В качестве носителей дистрибутивного комплекта SMART Linux на физических носителях должны использоваться цифровые многоцелевые диски DVD, не имеющие видимых механических повреждений, отрицательно влияющих на процессы воспроизведения информации. Носители SMART Linux считаются прошедшими проверку, если установлено, что они соответствуют указанным требованиям.

В состав документации SMART Linux должны входить документы:

- Операционная система SMART Linux. Формуляр. RU.29926343.02.01-01 30;
- Операционная система SMART Linux. Руководство администратора. RU.29926343.02.01-01 32 1-1.

Документация считается прошедшей проверку, если ее состав соответствует указанным требованиям.

Цифровые многоцелевые диски DVD с размещённой на них SMART Linux должны быть промаркированы и упакованы. Маркировка цифровых многоцелевых дисков с размещённой на них SMART Linux должна наноситься на внешнюю (нерабочую) поверхность диска и обязательно содержать наименование изделия и серийный номер, позволяющие однозначно идентифицировать SMART Linux.

№ РОСС RU. 0001.01БИ00, наносимым в разделе 8 Формуляра RU.29926343.02.01- 01 30. Для проверки маркировки и упаковки SMART Linux необходимо удостовериться в их соответствии указанным требованиям.

Если в процессе испытаний будет обнаружено несоответствие хотя бы одному требованию, то принимаемый экземпляр SMART Linux не считается выдержавшим испытания и возвращается для выявления причин дефектов.

SMART Linux считается окончательно принятой, если она соответствует требованиям и успешно прошла испытания.

2. Разновидности файловых систем

2.1 Поддерживаемые файловые системы

Основной файловой системой в SMART Linux является журналируемая файловая ext4. Также ядром поддерживаются все стандартные файловые системы xfs, btrfs т.д.

2.1.1 Утилиты для работы с файловыми системами

Общее назначение утилит приведено в таблице.

Утилита	Назначение
mkfs	Создание новой файловой системы.
fsck	Проверка файловой системы на ошибки.
df	Формирует отчёт о доступном и использованном дисковом пространстве на файловых системах. Без аргументов df выдаёт отчёт по доступному и использованному пространству для всех файловых систем (всех типов), которые смонтированы в данный момент. В противном случае, df на каждый файл, заданный как аргумент, выдаётся отчёт по файловой системе, которая его содержит.
du	Формирует отчёт об использовании дискового пространства заданными файлами, а также каждым каталогом иерархии подкаталогов каждого указанного каталога. Здесь под использованным дисковым пространством понимается пространство, используемое для всей иерархии подкаталогов указанного каталога. Запущенная без аргументов команда du выдаёт отчёт о дисковом пространстве для текущего каталога.

Справочник наиболее часто используемых утилит для работы с файловой системой приведён в таблице.

Системный вызов	Назначение
umount	Размонтирование файловой системы.
find	Поиск файлов в директориях.
which	Поиск файла, который будет запущен при выполнении данной команды.
cd	Смена текущего каталога.
pwd	Показать текущий каталог.
mkdir	Создание каталога.

ls	Выдача информации о файлах или каталогах.
cp	Копирование файлов.
mv	Перемещение/переименование файлов.
rm	Удаление файлов.
cat	Вывод содержимого заданных файлов на стандартный вывод.
less	Программа постраничного просмотра файлов.
ln	Создание ссылок (альтернативных имён) для файлов.
file	Определение типа файла.
chmod	Изменение прав доступа к файлам.
chown	Смена прав владения (пользовательских и групповых) для файлов.
umask	Установка маски прав доступа для вновь создаваемых файлов.
chattr	Изменение атрибутов файлов для файловой системы (append-only, immutable, safe deletion, no atime modified, no backup...).
lsattr	Просмотр атрибутов файлов для файловой системы.

3. Общие принципы работы SMART Linux

Работа с операционной средой заключается во вводе определённых команд (запросов) к операционной среде и получению на них ответов в виде текстового отображения.

Диалог с ОС осуществляется посредством команд. Каждая системная библиотека представляет собой набор программ, динамически вызываемых операционной системой.

В самом центре ОС SMART Linux находится управляющая программа, называемая ядром. Ядро взаимодействует с компьютером и периферией (дисками, принтерами и т.д.), распределяет ресурсы и выполняет фоновое планирование заданий. Другими словами, ядро ОС изолирует пользователя от сложностей аппаратуры компьютера, командный интерпретатор от ядра.

3.1 Процессы и файлы

SMART Linux является многопользовательской интегрированной системой. Это значит, что она разработана с расчётом на одновременную работу нескольких пользователей.

Пользователь может либо сам работать в системе, выполняя некоторую последовательность команд, либо от его имени могут выполняться прикладные процессы.

Пользователь взаимодействует с системой через командный интерпретатор, который представляет собой, как было сказано выше, прикладную программу, которая принимает от пользователя команды или набор команд и транслирует их в системные вызовы к ядру системы. Интерпретатор позволяет пользователю просматривать файлы, передвигаться по дереву файловой системы, запускать прикладные процессы. Все командные интерпретаторы имеют развитый командный язык и позволяют писать достаточно сложные программы, упрощающие процесс администрирования системы и работы с ней.

3.1.1 Процессы функционирования SMART Linux

Все программы, которые выполняются в текущий момент времени, называются процессами. Процессы можно разделить на два основных класса: системные процессы и пользовательские процессы.

Системные процессы — программы, решающие внутренние задачи SMART Linux, например организацию виртуальной памяти на диске или предоставляющие пользователям те или иные сервисы (процессы-службы).

Пользовательские процессы — процессы, запускаемые пользователем из командного интерпретатора для решения задач пользователя или управления системными процессами.

Фоновый режим работы процесса — режим, когда программа может работать без взаимодействия с пользователем. В случае необходимости интерактивной работы с пользователем (в общем случае) процесс будет «остановлен ядром» и работа его продолжится только после переведения его в «нормальный режим работы».

3.1.2 Файловая система SMART Linux

В ОС использована файловая система, которая является единым деревом. Корень этого дерева — каталог, называемый root (рут), и обозначаемый «/». Части дерева файловой системы могут физически располагаться в разных разделах разных дисков или вообще на других компьютерах, — для пользователя это прозрачно. Процесс присоединения файловой системы раздела к дереву называется монтированием, удаление — размонтированием.

3.1.3 Организация файловой структуры

Система домашних каталогов пользователей помогает организовывать безопасную работу пользователей в многопользовательской системе. Вне своего домашнего каталога пользователь обладает минимальными правами (обычно чтение и выполнение файлов) и не может нанести ущерб системе, например, удалив или изменив файл.

Кроме файлов, созданных пользователем, в его домашнем каталоге обычно содержатся персональные конфигурационные файлы некоторых программ.

Путь — это последовательность имён каталогов, представляющий собой путь в файловой системе к данному файлу, где каждое следующее имя отделяется от предыдущего наклонной чертой (/). Если название маршрута начинается со слэша, то путь в искомый файл начинается от корневого каталога всего дерева системы. В обратном случае, если название маршрута начинается непосредственно с имени файла, то путь к искомому файлу должен начинаться от текущего каталога (рабочего каталога).

Имя файла может содержать любые символы за исключением косой черты (/). Однако следует избегать применения в именах файлов большинства знаков препинания и непечатаемых символов. При выборе имен файлов рекомендуем ограничиться следующими символам:

- строчные и ПРОПИСНЫЕ буквы. Следует обратить внимание на то, что регистр всегда имеет значение;
- символ подчёркивания (_);
- точка (.).

Для удобства работы можно использовать знак «.» (точка) для отделения имени файла от расширения файла. Данная возможность может быть необходима пользователям или некоторым программам, но не имеет значения для shell.

3.1.4 Иерархическая организация файловой системы

Содержимое корневого каталога «/» представлено в таблице.

Каталог	Описание
/boot	Место, где хранятся файлы, необходимые для загрузки ядра системы.
/lib	Ссылка на /usr/lib
/lib64	Ссылка на /usr/lib
/bin	Ссылка на /usr/bin
/sbin	Ссылка на /usr/bin
/home	Здесь располагаются домашние каталоги пользователей.
/etc	Здесь хранятся общесистемные конфигурационные файлы для большинства программ в системе.
/etc/passwd	База данных пользователей, в которой содержится информация об имени пользователя, его настоящем имени, личном каталоге и другие данные.
/etc/shadow	Теневая база данных пользователей. При этом информация из файла /etc/passwd перемещается в /etc/shadow, который недоступен по чтению всем, кроме пользователя root.
/dev	В этом каталоге находятся файлы устройств. Файлы в /dev создаются сервисом systemd-udevd.
/usr	Обычно файловая система /usr достаточно большая по объёму, так как все программы установлены именно здесь. Вся информация в каталоге /usr помещается туда во время установки системы. Некоторые подкаталоги системы /usr рассмотрены ниже.
/usr/lib	Здесь располагаются файлы динамических библиотек, необходимых для работы большей части приложений и подгружаемые модули ядра.
/usr/bin	Программы, необходимые для работы в системе.
/usr/sbin	Ссылка на /usr/bin.
/usr/share	Каталог для размещения общедоступных файлов большей части приложений.
/var/log	Место, где хранятся файлы журналов работы системы и приложений.

/var/spool	Каталог для хранения файлов, находящихся в очереди на обработку для того или иного процесса (очередь на печать, отправку почты и т.д.).
/tmp	Временный каталог, необходимый некоторым приложениям.
/proc	Файловая система /proc является виртуальной и в действительности не существует на диске. Ядро создаёт её в памяти компьютера. Система /proc предоставляет информацию о системе.
/sys	Виртуальная файловая система, предоставляющая информацию об устройствах.

3.1.5 Имена дисков и разделов

Все физические устройства компьютера отображаются в каталоге /dev файловой системы SMART Linux.

Диски (DASD) имеют имена:

- **/dev/dasda** — первый диск;
- **/dev/dasdb** — второй диск и т.д.

Диски обозначаются /dev/dasdX, где X — a,b,c,d,e,... в зависимости от порядкового номера диска на шине. Раздел диска обозначается числом после его имени.

Например, /dev/dasdb4 — четвёртый раздел второго диска.

3.2 Работа с наиболее часто используемыми компонентами

3.2.1 Командная оболочка Bash

Как было сказано выше, для управления ОС используется командный интерпретатор (shell).

Зайдя в систему, пользователь увидит приглашение — строку, содержащую символ «**$**» (далее этот символ будет обозначать командную строку). Программа ожидает ввода команд. Роль командного интерпретатора — передавать команды операционной системе. При помощи командных интерпретаторов можно писать небольшие программы — сценарии (скрипты). В SMART Linux используется командная оболочка Bash (Bourne Again Shell). Она ведёт историю команд и предоставляет возможность их редактирования.

Командная оболочка Bash

В bash имеется несколько приёмов для работы со строкой команд. Например, используя клавиатуру, можно:

Ctrl + A — перейти на начало строки.

Ctrl + U — удалить текущую строку.

Ctrl + C — остановить текущую задачу.

Можно использовать «;» для того, чтобы ввести несколько команд одной строкой. Клавиши «вверх» и «вниз», позволяют перемещаться по истории команд. Для того, чтобы найти конкретную команду в списке набранных, не пролистывая всю историю, нужно набрать:

Ctrl + R

Команда history показывает историю команд. Команды, присутствующие в истории, отображаются в списке пронумерованными. Для того, чтобы запустить конкретную команду, нужно набрать:

! <номер_команды>

если будет введено:

!!

запустится последняя из набранных команд.

Иногда имена программ и команд слишком длинны, но Bash сам может завершать имена. Нажав клавишу «TAB», можно завершить имя команды, программы или каталога. Предположим, необходимо использовать программу декомпрессии **bunzip2**. Для этого нужно набрать:

bu

затем нажать «**TAB**». Если ничего не происходит, то, вероятно, существует несколько возможных вариантов завершения команды.

Нажав клавишу «**TAB**» ещё раз, можно получить список имён, начинающихся с «**bu**».

Например:

builtin bunzip2 busctl

Если далее добавить «**n**» (**bunzip2** — это единственное имя, третьей буквой которого является «**n**»), а затем нажать клавишу «**TAB**», оболочка дополнит имя и остаётся лишь нажать «Enter», чтобы запустить команду.

Заметим, что программу, вызываемую из командной строки, Bash ищет в каталогах, определяемых в системной переменной **PATH**. По умолчанию, в этот перечень каталогов не входит текущий каталог, обозначаемый «./» (точка слэш). Поэтому, для запуска программы prog из текущего каталога, надо дать команду:

3.2.2 Базовые команды оболочки Bash

Команда «**su**» позволяет получить права администратора. Когда пользователь набирает «**su**», оболочка запрашивает пароль суперпользователя (root). Необходимо ввести пароль и нажать «Enter». Чтобы вернуться к правам основного пользователя, необходимо набрать «exit».

Команда «**cd**» позволяет сменить каталог. Она работает как с абсолютными, так и с относительными путями. Предположим, что, находясь в своём домашнем каталоге, пользователь хочет перейти в его подкаталог docs/. Для этого нужно ввести относительный путь:

cd docs

Чтобы перейти в каталог /usr/bin, нужно набрать (абсолютный путь):

cd /usr/bin

Некоторые варианты команды приведены в таблице:

Команда	Описание
cd ..	Позволяет сделать текущим родительский каталог.
cd -	Позволяет вернуться в предыдущий каталог.
cd	Переводит в домашний каталог.

Команда «ls» (list) выдаёт список файлов в текущем каталоге. Две основные опции:

- **-A** — просмотр всех файлов, включая скрытые;
- **-l** — отображение более подробной информации.

Команда «**rm**» используется для удаления файлов.

rm <имя_файла>

У данной программы существует ряд параметров. Самые часто используемые:

- **-i** — запрос на удаление файла;
- **-r** — рекурсивное удаление (т.е. удаление, включая подкаталоги и скрытые файлы).

Пример использования команды:

rm -i ~/html/*.html

Удаляет все файлы html, в каталоге html.

Команды «**mkdir**» и «**rmdir**». Команда «**mkdir**» позволяет создать каталог, тогда как «**rmdir**» удаляет каталог, при условии, что он пуст.

mkdir <имя_каталога>

rmdir <имя_каталога>

Команда «**rmdir**» часто заменяется командой «**rm -rf**», которая позволяет удалять каталоги, даже если они не пусты.

Команда «**less**» позволяет постранично просматривать текст.

less <имя_файла>

Для выхода из less нужно нажать «**q**».

Команда «**grep**» имеет много опций и предоставляет возможности поиска символьной строки в файле.

grep <шаблон_поиска> <файл>

Команда «**ps**» отображает список текущих процессов.

ps <аргументы>

Колонка команд указывает имя процесса, колонка PID (идентификатор процесса) — номер процесса (этот номер используется, для операций с процессом, например, чтобы «убить» его командой «**kill**»).

Аргументы:

- **-u** — предоставляет больше информации;
- **-x** — позволяет просмотреть те процессы, которые не принадлежат пользователю (такие как те, что были запущены во время процесса загрузки).

Команда «**kill**» используется, если программа перестала отвечать или зависла, чтобы её завершить.

kill <PID>

Иногда необходимо будет использовать «**kill -9 <PID>**» (когда обычная команда «**kill**» не даёт желательного эффекта). Номер PID выясняется при помощи команды «**ps**».

Команда «**cat**» — утилита, выводящая последовательно указанные файлы (или устройства), таким образом объединяя их в единый поток. Если вместо имени файла указывается «-», то читается стандартный ввод.

cat <имя_файла>

Команда «**sort**» — утилита для вывода текстовых строк в определённом порядке.

sort <опции> <файл>

4. Установка SMART Linux

Для установки SMART Linux пользователь должен знать ответы на следующее:

- идентификатор диска DASD, на который будет ставиться ОС;
- нужно ли делать низкоуровневое форматирование DASD диска (dasdfmt);
- идентификаторы сетевых устройств, которые нужны для создания сетевого устройства(eth0) в операционной системе.

Содержимое архива с SMART Linux необходимо сделать доступным по FTP в консоли HMC. В корне находится файл linux.ins. Его необходимо выбрать в HMC и начать загрузку образа с программой установки. После загрузки будет предложено войти систему. Необходимо ввести имя пользователя root, без пароля. После входа необходимо запустить установку из корня файловой системы, введя команду:

/InstallOs

И далее отвечать на вопросы программы установки. Ниже приводятся рисунки с примером установки и ответом на все запрашиваемые вопросы.

```
localhost:~# /InstallOs
Enter dasd id: 3000  1
Run low level format(dasdfmt) on 3000? y  2
Configuring devices in the active configuration only
ECKD DASD 0.0.3000 configured
Create new boot,root partitions? y  3
Enter root fs size(examples: 500m,10g,max): max  4
Enter 1 qeth id (examples: 1003 a000): 1000  5
Enter 2 qeth id (examples: 1003 a000): 1001  6
Enter 3 qeth id (examples: 1003 a000): 1002  7
Starting dry-run, configuration will not be changed
Configuring devices in the active configuration only
QETH device 0.0.1000:0.0.1001:0.0.1002 already configured
Setup network device for using DHCP? n 8
Enter static ip address (example 10.10.10.10/24): 10.10.10.4/24 9
Enter gateway address: 10.10.10.5 10
Enter DNS address: 11

Summary: 12
dasd id: 3000
Run low level format: y
Create new boot partition.
Create new root partition: max
Configure qeth device using 1000 1001 1002
Network configuration:
Address: 10.10.10.4/24
Gateway: 10.10.10.5
DNS:
Continue? y     13
```

Рисунок 4.1. Пример ответов установки ОС.

Пояснение к рисунку 1:

1. запрос идентификатора dasd диска;
2. запрос надо ли делать низкоуровневое форматирование диска;
3. запрос создавания новых boot и root разделов;
4. запрос размера нового root раздела. раздел boot будет создан размером 300 м;

5. запросы идентификаторов для создания сетевого устройства;
6. запросы идентификаторов для создания сетевого устройства;
7. запросы идентификаторов для создания сетевого устройства;
8. запрос конфигурации сетевых настроек сетевого устройства eth0. динамический(dhcp) или статический адрес. если адрес статический, то потом заданы вопросы 9, 10 и 11;
9. запрос адреса;
10. запрос сетевого шлюза. Может быть пустым;
11. запрос dns адреса. Может быть пустым;
12. показываются введённые ответы;
13. запрос на запись изменений на диск и начало установки. До этого момента на диск ничего не пишется, если ответить отрицательно, то вопросы будут задаваться заново.

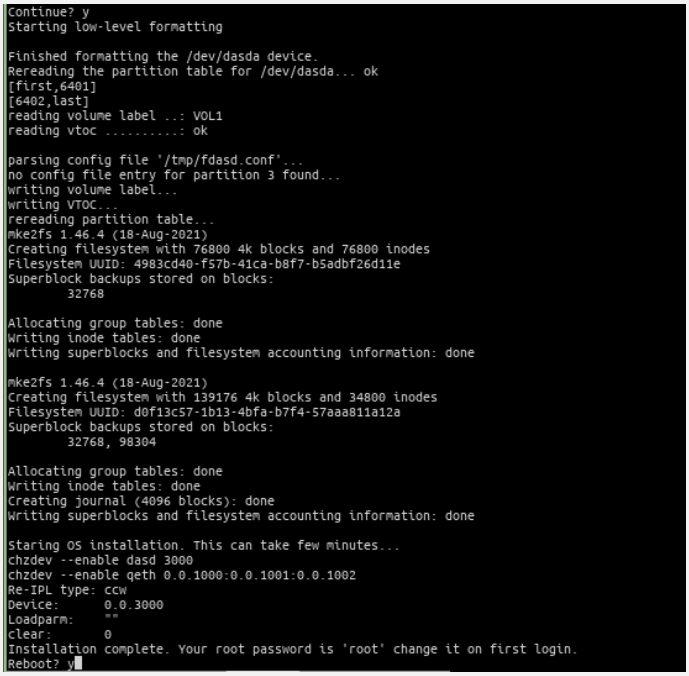

Рисунок 4.2. Процесс установки ОС.

Когда установка завершена, в установленной системе пароль для пользователя root будет установлен root, его необходимо сменить позднее, что видно на рисунке 4.2. В конце задан вопрос на перезагрузку системы, после этого будет загружена установленная ОС. Необходимо иметь ввиду, что во вновь установленной системе после перезарузки может измениться MAC-адрес сетевого устройства. И если система недоступна по сети, то из неё надо сделать ping шлюза, это необходимо сделать только один раз после установки.

5. Управление ПО

5.1 Введение: пакеты, зависимости и репозитории

Для установки, удаления и обновления программ и поддержания целостности системы используются менеджеры пакетов. С точки зрения менеджера пакетов программное обеспечение представляет собой набор компонентов - программных пакетов. Такие компоненты содержат в себе набор исполняемых программ и вспомогательных файлов, необходимых для корректной работы программного обеспечения. Менеджеры пакетов облегчают установку программ: они позволяют проверить наличие необходимых для работы устанавливаемой программы компонент подходящей версии непосредственно в момент установки, а также производят необходимые процедуры для регистрации программы во всех операционных средах пользователя: сразу после установки программа может быть доступна пользователю из командной строки.

Часто компоненты, используемые различными программами, выделяют в отдельные пакеты и помечают, что для работы ПО, предоставляемого пакетом A, необходимо установить пакет B. В таком случае говорят, что пакет A зависит от пакета B или что между пакетами A и B существует зависимость. Отслеживание зависимостей между такими пакетами представляет собой серьёзную задачу — некоторые компоненты могут быть взаимозаменяемыми.

Может обнаружиться несколько пакетов, предлагающих затребованный ресурс. Задача контроля целостности и непротиворечивости установленного в системе ПО ещё сложнее. Представим, что некие программы A и B требуют наличия в системе компоненты C версии 1.0. Обновление версии пакета A, требующее обновления компоненты C до новой, использующей новый интерфейс доступа, версии (скажем, до версии 2.0), влечёт за собой обязательное обновление и программы B. Для решения вышеперечисленных проблем в дистрибутиве SMART Linux используется пакетный менеджер zypper.

5.2 Основы работы с RPM

RPM (Redhat Package Manager) служит для работы с пакетами — установка, удаление, проверка и т.д. При установке пакета rpm записывает информацию о нем в свою базу данных, что и позволяет в дальнейшем удалять пакет, просматривать информацию о нем и т.д.

Такой подход к установке ПО имеет несколько достоинств, в частности:

- унифицированная работа с разными пакетами;
- отслеживание зависимостей между пакетами выполняется автоматически;
- непротиворечивость между разными пакетами.

Если вызвать rpm без параметров, то он покажет краткий список ключей.

Обычно же формат вызова rpm такой:

rpm -<ключ_режима> <дополнительные_ключи> <параметры>

где <ключ_режима>, указываемый первым, определяет режим работы.

Основные варианты вызова rpm приведены в таблице.

rpm -i <файл_пакета>.rpm	Установка пакета (install).
rpm -U <файл-пакета>.rpm	Обновление пакета (upgrade).
rpm -e <пакет>	Удаление пакета (erase).
rpm -q <пакет>	Получение информации (query).
rpm -v <пакет>	Проверка пакета (verify).
rpm -b	Создание пакета .rpm из .src.rpm (build).

для режимов «**-i**» и «**-U**» — это полное (с директорией) Имя <файла-пакета>.rpm имя файла. Пример команды выглядит следующим образом:

~/RPMS/grep-3.11-1.s390x.rpm

Пакет — это имя уже установленного пакета для режимов «**-e**», «**-q**» и «**-y**».

Оно может указываться как с номером версии, так и без него.

Примеры:

grep-3.11-1

grep

Имя <файла-пакета>.rpm для режимов «**-i**» и «**-U**» — это полное (с директорией) имя файла. Пример команды выглядит следующим образом:

~/RPMS/grep-3.11-1.s390x.rpm

Если вместо списка пакетов указать ключ «**-a**» (all), то это будет означать «все пакеты». Кроме того, ключ «**-f**» позволяет вместо имени пакета указать какой-либо файл, принадлежащий этому пакету.

Можно указывать не один файл-пакета или пакет, а сразу несколько, разделяя их пробелами.

Команда «**rpm -q**» позволяет получать следующую информацию о пакете:

- версию пакета;
- список файлов;
- чего требует пакет;
- можно узнать, какому пакету принадлежит указанный файл.

Просто «**rpm -q <имя-пакета>**» выдаёт полное название пакета, вместе с версией. Но чаще всего команда «**rpm -q**» используется для получения списка файлов пакета.

Команда «**rpm -qi**» (info) выдаёт сводку информации о пакете — название, версию, объем и т.д., плюс краткую аннотацию. Для получения списка файлов используется ключ «**-l**» (list).

Для получения «полной» информации о пакете (аннотации и списка файлов) можно указать ключи «**-i**» и «**-l**» одновременно. Часто возникает необходимость узнать, какому пакету принадлежит какой-то файл (например, чтобы знать, где искать к нему документацию). Для этого можно воспользоваться ключом «**-f**» (file). При этом надо указывать полное имя файла — с директорией. Кроме того, если к файлу есть «несколько путей» (из-за символьных линков на директории), то следует указывать «основной» (обычно тот, который без символьных линков), иначе rpm не сможет дать ответ.

Ключ «**-R**» (Requirements) позволяет узнать, какие пакеты и библиотеки требуются пакету.

Пример:

rpm -qp -R perl-5.26.1-150300.17.11.1.s390x.rpm

5.3 Назначение zypper

Zypper — это менеджер пакетов в SMART Linux. Он предоставляет пользователю интерфейс командной строки для управления установкой, удалением, обновлением и поиском программного обеспечения в системе. Основные команды zypper:

zypper install <package_name>	Установка пакетов
zypper remove <package_name>	Удалить пакеты
zypper update	Обновить систему и все установленные пакеты
zypper search <package_name>	Поиск пакета в репозитории
zypper info <package_name>	Получить информацию о пакете
zypper repos	Показать список всех доступных репозиториев
zypper removerepo <repository_name>	Удалить репозиторий
zypper addrepo <repository_url> <name>	Добавить новый репозиторий
zypper patch	Показать доступные обновления без их применения.
zypper list-patches	Показать список установленных обновлений
zypper clean	Очистить кеш загруженных пакетов
zypper clean --all	Очистить кеш и журналы установки
zypper verify	Проверить целостность файлов пакетов

Zypper предоставляет много возможностей для управления системой и пакетами в ней через командную строку, делая управление пакетами и обновлениями на системе довольно простым.

5.4 Источники программ (репозитории)

Для просмотра списка доступных репозиториев используйте команду:

zypper repos

Дополнительные аргументы:

-e, --export <FILE.repo> — Экспортировать все определенные репозитории в один локальный файл .repo.

-a, --alias — Показывать также псевдоним репозитория. По умолчанию: false.

-n, --name — Показывать также имя репозитория. По умолчанию: false.

-r, --refresh — Показывать также флаг автообновления. По умолчанию: false.

-u, --uri — Показывать также базовый URI репозиториев. По умолчанию: false.

-p, --priority — Показывать также приоритет репозитория. По умолчанию: false.

-d, --details — Показывать больше информации, такую как URI, приоритет, тип. По умолчанию: false.

-s, --servic — Показывать также псевдоним родительской службы. По умолчанию: false.

-E, --show-enabled-only — Показывать только включенные репозитории. По умолчанию: false.

-U, --sort-by-uri — Сортировать список по URI. По умолчанию: false.

-N, --sort-by-name — Сортировать список по имени. По умолчанию: false.

-P, --sort-by-priority — Сортировать список по приоритету репозитория. По умолчанию: false.

-A, --sort-by-alias — Показывать также псевдоним родительской службы. По умолчанию: false.

Чтобы добавить репозиторий нужно выполнить команду:

zypper addrepo <repository_url> <repository_name>

Чтобы удалить репозиторий:

zypper removerepo <repository_name>

Пример добавления репозитория:

zypper addrepo https://smartlinux.ru/repository/smartlinux/ smartlinux

zypper refresh smartlinux

Примечание

После внесения изменений в репозитории или их установки следует выполнить команду zypper refresh для обновления данных репозитория. При указании refresh без дополнительного аргумента произойдет обновление всех репозиториев.

5.5 Поиск пакетов

Если пользователь не знает точного названия пакета, для его поиска можно воспользоваться утилитой «**zypper search**», которая позволяет искать пакет по части его названия.

Команда:

zypper search <подстрока>

Пример вывода команды zypper search glib:

S	Name	Summary	Type
i+	glib2-devel	Development files for glib, a general-purpose utility library	package
i+	glib2-tools	Tools from glib2, a general-purpose utility library	package
i+	glibc	Standard Shared Libraries (from the GNU C Library)	package
i	glibc-devel	Include Files and Libraries Mandatory for Development	package

Здесь «**i**» означает что пакет был автоматически установлен (возможно в качестве зависимости другого пакета),

«**i+**» установлен вручную

Для того чтобы подробнее узнать о каждом из найденных пакетов и прочитать его описание, можно воспользоваться командой «**zypper info**», которая покажет информацию о пакете из репозитория:

zypper info glib2-devel

5.6 Установка или обновление пакета

Установка пакета с помощью zypper выполняется командой:

zypper install <имя_пакета>

Zypper позволяет устанавливать в систему пакеты, требующие для работы другие, пока ещё не установленные. В этом случае он определяет, какие пакеты необходимо установить, и устанавливает их, пользуясь всеми доступными репозиториями.

При помощи zypper можно установить и отдельный бинарный rpm-пакет, не входящий ни в один из репозиториев. Для этого достаточно выполнить команду:

zypper install <путь_к_файлу.rpm>

При этом zypper проведёт стандартную процедуру проверки зависимостей и конфликтов с уже установленными пакетами.

5.7 Удаление установленного пакета

Для удаления пакета используется команда:

zypper remove <имя_пакета>

Для того, чтобы не нарушать целостность системы, будут удалены и все пакеты, зависящие от удаляемого: если отсутствует необходимый для работы приложения компонент (например, библиотека), то само приложение становится бесполезным.

5.8 Обновление всех установленных пакетов

Чтобы проверить доступные обновления для всех системных пакетов, необходимо выполнить:

zypper list-updates

Когда все пакеты, установленные на ОС, должны быть обновлены, необходимо использовать команду:

zypper update

5.9 Работа с блокировкой пакетов

Zypper предоставляет возможность блокировать пакеты, чтобы предотвратить их обновление или удаление.

Чтобы заблокировать пакет, используйте команду zypper addlock с указанием имени пакета:

zypper addlock <package_name>

Для разблокировки пакета используйте команду zypper removelock с указанием имени пакета:

zypper removelock <package_name>

Чтобы просмотреть список заблокированных пакетов, выполните команду:

zypper locks

6. Система безопасности

6.1 Программа sudo

Sudo — это программа, разработанная в помощь системному администратору и позволяющая делегировать те или иные привилегированные ресурсы пользователям с ведением протокола работы. Основная идея — дать пользователям как можно меньше прав, но при этом ровно столько, сколько необходимо для решения поставленных задач.

Команда sudo предоставляет возможность пользователям выполнять команды от имени root либо других пользователей. Правила, используемые sudo для принятия решения о предоставлении доступа, находятся в файле /etc/sudoers. Кроме того, пример правил, предоставляющих пользователям, являющимися членами группы rpm, возможность устанавливать, обновлять и удалять пакеты в системе, приведён в файле /usr/share/doc/packages/sudo/examples/sudoersq

Для редактирования файла /etc/sudoers следует использовать программу visudo, которая проверяет синтаксис и тем самым позволяет избежать ошибок в правилах.

В большинстве случаев грамотная настройка sudo делает работу от имени суперпользователя ненужной.

6.2 IP-фильтр iptables: архитектура и синтаксис

Назначение IP-фильтра — обработка потока данных, проходящих через стек сетевых протоколов ядра ОС по заданным критериям.

Фильтры состоят из правил. Каждое правило — это строка, содержащая в себе критерии, определяющие, подпадает ли пакет под заданное правило, и действие, которое необходимо выполнить в случае удовлетворения критерия.

6.2.1 Устройство фильтра iptables

Для iptables в общем виде правила выглядят так:

Необязательно ставить описание действия <target/jump> последним в строке, но лучше придерживаться именно такой нотации для удобочитаемости правил.

Если в правило не включается спецификатор <-t table>, то по умолчанию предполагается использование таблицы «filter», если же предполагается использование другой таблицы, то это требуется указать явно. Спецификатор таблицы так же можно указывать в любом месте строки правила, однако более или менее стандартом считается указание таблицы в начале правила.

Далее, непосредственно за именем таблицы должна стоять команда управления фильтром. Если спецификатора таблицы нет, то команда всегда должна стоять первой. Команда определяет действие iptables, например: вставить правило, или добавить правило в конец цепочки, или удалить правило и т.п. Тело команды в общем виде выглядит так:

<команда> <цепочка>

Ключ команды указывает на то, что нужно сделать с правилом, например, команда «-A» указывает на то, что правило нужно добавить в конец указанной цепочки.

Цепочка указывает в какую цепочку нужно добавить правило. Стандартные цепочки — INPUT, OUTPUT, FORWARD, PREROUTING и POSTROUTING. Они находятся в таблицах фильтра. Не все таблицы содержат все стандартные цепочки.

Раздел <match> задаёт критерии проверки, по которым определяется, попадает ли пакет под действие этого правила или нет. Здесь можно указать самые разные критерии - IP-адрес источника пакета или сети, сетевой интерфейс и т.д.

<Target> указывает, какое действие должно быть выполнено при условии выполнения критериев в правиле. Здесь можно передать пакет в другую цепочку правил, «сбросить» пакет и забыть про него, выдать на источник сообщение об ошибке и т.д.

Когда пакет приходит на сетевое устройство, он обрабатывается соответствующим драйвером и далее передается в фильтр в ядре ОС. Далее пакет проходит ряд таблиц и затем передаётся либо локальному приложению, либо переправляется на другую машину.

6.2.2 Встроенные таблицы фильтра iptables

По умолчанию используется таблица «filter». Опция «-t» в правиле указывает на используемую таблицу. С ключом «-t» можно указывать следующие таблицы:

«nat», «mangle», «filter», «raw».

Таблица nat

Таблица «nat» используется, главным образом, для преобразования сетевых адресов Network Address Translation. Через эту таблицу проходит только первый пакет из потока. Преобразования адресов автоматически применяется ко всем последующим пакетам. Это один из факторов, исходя из которых, не нужно осуществлять какую-либо фильтрацию в этой таблице.

Цепочка **PREROUTING** используется для внесения изменений в пакеты на входе в фильтр.

Цепочка **OUTPUT** используется для преобразования пакетов, созданных приложениями внутри компьютера, на котором установлен фильтр, перед принятием решения о маршрутизации.

Цепочка **POSTROUTING** используется для преобразования пакетов перед выдачей их в сеть.

Таблица mangle

Таблица «mangle» используется для внесения изменений в заголовки пакетов. Примером может служить изменение поля TTL, TOS или MARK.

Цепочка **PREROUTING** используется для внесения изменений на входе в фильтр перед принятием решения о маршрутизации.

Цепочка **OUTPUT** — для внесения изменений в пакеты, поступающие от внутренних приложений. Таблица «mangle» не должна использоваться для преобразования сетевых адресов (Network Address Translation) или маскарадинга (masquerading), поскольку для этих целей имеется таблица «nat».

Таблица filter

Таблица filter используется главным образом для фильтрации пакетов. Для примера, здесь мы можем выполнить DROP, LOG, ACCEPT или REJECT без каких-либо сложностей, как в других таблицах. Имеется три встроенных цепочки FORWARD, INPUT, OUTPUT.

Цепочка **FORWARD** используется для фильтрации пакетов, идущих транзитом через фильтрующий компьютер.

Цепочка **INPUT** предназначена для обработки входящих пакетов, направляемых локальным приложениям фильтрующего компьютера.

Цепочка **OUTPUT** используется для фильтрации исходящих пакетов, сгенерированных локальными приложениями фильтрующего компьютера.

6.2.3 Команды утилиты iptables

Ниже приводится список команд и правила их использования. Посредством команд сообщается iptables, что предполагается сделать. Обычно предполагается одно из двух действий — это добавление нового правила в цепочку или удаление существующего правила из той или иной таблицы. Далее приведены команды, которые используются в iptables.

Команда	Формат вызова, результат
-A, -append	iptables -A Добавляет новое правило в конец заданной INPUT цепочки.
-D, -delete	Удаление правила из цепочки. Команда имеет два формата записи, первый — когда задаётся iptables -D критерий сравнения с опцией -D (см. первый INPUT, -dport 80 пример), второй — порядковый номер правила. -j DROP, iptables -D INPUT 1 Если задаётся критерий сравнения, то удаляется правило, которое имеет в себе этот критерий, если задаётся номер правила, то будет удалено правило с заданным номером. Счёт правил в цепочках начинается с 1.
-R, -replace	iptables -R INPUT 1 -s 192.168.0.1 -j DROP Данная команда заменяет одно правило другим. В основном она используется во время отладки новых правил.

-I, -insert	iptables -I INPUT 1 -dport 80 -j ACCEPT Вставляет новое правило в цепочку. Число, следующее за именем цепочки, указывает номер правила, перед которым нужно вставить новое правило, другими словами, число задаёт номер для вставляемого правила. В примере, указывается, что данное правило должно быть 1-м в цепочке INPUT.
-L, -list	iptables -L INPUT Вывод списка правил в заданной цепочке, в данном примере предполагается вывод правил из цепочки INPUT. Если имя цепочки не указывается, то выводится список правил для всех цепочек. Формат вывода зависит от наличия дополнительных ключей в команде, например, -n, -v, и пр.
-F, -flush	Удаление всех правил из заданной цепочки iptables -F (таблицы). Если имя цепочки и таблицы не INPUT указывается, то удаляются все правила, во всех цепочках.
-Z, -zero	iptables -Z INPUT Обнуление всех счётчиков в заданной цепочке. Если имя цепочки не указывается, то подразумеваются все цепочки. При использовании ключа -v совместно с командой -L, на вывод будут поданы и состояния счётчиков пакетов, попавших под действие каждого правила. Допускается совместное использование команд -L и -Z. В этом случае будет выдан сначала список правил со счётчиками, а затем произойдёт обнуление счётчиков.
-N, -new-chain	iptables -N allowed Создаётся новая цепочка с заданным именем в заданной таблице В выше приведённом примере создаётся новая цепочка с именем allowed. Имя цепочки должно быть уникальным и не должно совпадать с зарезервированными именами цепочек и действий (DROP, REJECT и т.п.)
-X, -delete -chain	iptables -X allowed Удаление заданной цепочки из заданной таблицы. Удаляемая цепочка не должна иметь правил и не должно быть ссылок из других цепочек на удаляемую цепочку. Если имя цепочки не указано, то будут удалены все цепочки, определённые командой — N в заданной таблице.

-P, -policy	iptables -P INPUT DROP Определяет политику по умолчанию для заданной цепочки. Политика по умолчанию определяет действие, применяемое к пакетам не попавшим под действие ни одного из правил в цепочке. В качестве политики по умолчанию допускается использовать DROP, ACCEPT и REJECT.
-E, -rename -chain	iptables -E allowed disallowed Команда -E выполняет переименование пользовательской цепочки. В примере цепочка allowed будет переименована в цепочку disallowed. Эти переименования не изменяют порядок работы, а носят только косметический характер.

Команда должна быть указана всегда. Список доступных команд можно просмотреть с помощью команды:

Iptables -h

Некоторые команды могут использоваться совместно с дополнительными ключами. Ниже приводится список дополнительных ключей, и описывается результат их действия.

6.2.4 Ключи утилиты iptables

Команда	Пример Пояснения
-v, --verbose	-list, -list, -insert, -delete, -replace Данный ключ используется для повышения информативности вывода и, как правило, используется совместно с командой -list. В случае использования с командой -list, в вывод этой команды включаются так же имя интерфейса, счётчики пакетов и байт для каждого правила. Формат вывода счётчиков предполагает вывод кроме цифр числа ещё и символьные множители K (x1000), M (x1,000,000) и G (x1,000,000,000). Для того чтобы заставить команду -list выводить полное число (без употребления множителей), требуется применять ключ -x, который описан ниже. Если ключ -v, -verbose используется с командами -append, - insert, - delete или -replace, то на вывод будет выдан подробный отчёт о произведённой операции.

-x, --exact	Для всех чисел в выходных данных выводятся -list их точные значения без округления и без применения множителей K, M, G.
-n, --numeric	Iptables выводит IP-адреса и номера портов в -list числовом виде, предотвращая попытки преобразовать их в символические имена.
-line-number	s -list Ключ -line-numbers включает режим вывода номеров строк при отображении списка правил.
-c, =-set **-counters**	-insert, -append, -replace Этот ключ используется при создании нового правила для установки счётчиков пакетов и байт в заданное значение. Например, ключ -set-counters 20 4000 установит счётчик пакетов = 20, а счётчик байт = 4000.
--modprobe	Любая команда Ключ -modprobe определяет команду загрузки модуля ядра.

6.2.5 Основные действия над пакетами в фильтре iptables

Действие	Пояснения
ACCEPT	Пакет прекращает движение по цепочке (и всем вызвавшим цепочкам, если текущая цепочка была вложенной) и считается принятым; тем не менее, пакет продолжит движение по цепочкам в других таблицах и может быть отвергнут там.
DROP	Отбрасывает пакет и iptables «забывает» о его существовании. Отброшенные пакеты прекращают своё движение полностью.
RETURN	Прекращает движение пакета по текущей цепочке правил и производит возврат в вызывающую цепочку, если текущая цепочка была вложенной, или, если текущая цепочка лежит на самом верхнем уровне (например INPUT), то к пакету будет применена политика по умолчанию
LOG	Служит для журналирования отдельных пакетов и событий. В системный журнал могут заноситься заголовки IP-пакетов и другая интересующая вас информация.
REJECT	Используется, как правило, в тех же самых ситуациях, что и DROP, но в отличие от DROP, команда REJECT выдаёт сообщение об ошибке на хост, передавший пакет.

SNAT	Используется для преобразования сетевых адресов (Source Network Address Translation), т.е. изменение исходящего IP- адреса в IP-заголовке пакета.
DNAT	Destination Network Address Translation используется для преобразования адреса места назначения в IP-заголовке пакета.
MASQUERADE	В основе своей представляет то же самое, что и SNAT только не имеет ключа -to -source. Причиной служит то, что маскарадинг может работать, например, с dialup подключением или DHCP, т.е. в тех случаях, когда IP-адрес присваивается устройству динамически. Если используется динамическое подключение, то нужно использовать маскарадинг, если же используется статическое IP- подключение, то лучшим выходом будет использование действия SNAT.
REDIRECT	Выполняет перенаправление пакетов и потоков на другой порт той же самой машины. К примеру, можно пакеты, поступающие на HTTP порт перенаправить на порт HTTP proxy. Действие REDIRECT очень удобно для выполнения «прозрачного» проксирования (transparent proxy), когда компьютеры в локальной сети даже не подозревают о существовании прокси.
TTL	Используется для изменения содержимого поля «время жизни» (Time To Live) в IP-заголовке. Один из вариантов применения этого действия — это устанавливать значение поля Time To Live во всех исходящих пакетах в одно и то же значение. Если установить на все пакеты одно и то же значение TTL, то тем самым можно лишить провайдера одного из критериев определения того, что подключение к Интернету разделяется между несколькими компьютерами. Для примера можно привести число TTL = 64, которое является стандартным для ядра ОС.

6.2.6 Основные критерии пакетов в фильтре iptables

Критерий	Пояснения
-p, -protocol	Используется для указания типа протокола. Примерами протоколов могут быть TCP, UDP и ICMP. Список протоколов можно посмотреть в файле /etc/protocols. Прежде всего, в качестве имени протокола в данный критерий можно передавать три вышеупомянутых протокола, а также ключевое слово ALL. В качестве протокола допускается передавать число — номер протокола

-s, -src, -source	IP-адрес (-a) источника пакета. Адрес источника может указываться так - 192.168.1.1, тогда подразумевается единственный IP-адрес. А можно указать адрес в виде address/mask, например, как 192.168.0.0/255.255.255.0, или более современным способом 192.168.0.0/24, т.е. фактически определяя диапазон адресов. Символ «!», установленный перед адресом, означает логическое отрицание, т.е. -source ! 192.168.0.0/24 означает любой адрес, кроме адресов 192.168.0.x.
-d, -dst, -destination	IP-адрес (-a) получателя. Имеет синтаксис, схожий с критерием — source, за исключением того, что подразумевает единственный IP-адрес, так и диапазон адресов. Символ «!» используется для логической инверсии критерия.
-i, --in-interface	Интерфейс, с которого был получен пакет. Использование этого критерия допускается только в цепочках INPUT, FORWARD и PREROUTING, в любых других случаях будет вызывать сообщение об ошибке.
-o, --out-interface	Задаёт имя выходного интерфейса. Этот критерий допускается использовать только в цепочках OUTPUT, FORWARD и POSTROUTING, в противном случае будет генерироваться сообщение об ошибке.
-f, -fragment	Правило распространяется на все фрагменты фрагментированного пакета, кроме первого, сделано это потому, что нет возможности определить исходящий/входящий порт для фрагмента пакета, а для ICMP-пакетов определить их тип. С помощью фрагментированных пакетов могут производиться атаки на межсетевой экран, так как фрагменты пакетов могут не отлавливаться другими правилами.
-sport, -source-port	Исходный порт, с которого был отправлен пакет. В качестве параметра может указываться номер порта или название сетевой службы. Соответствие имён сервисов и номеров портов вы сможете найти в файле /etc/services. При указании номеров портов правила отрабатывают несколько быстрее.
-dport, -destination -port	Порт, на который адресован пакет. Аргументы задаются в том же формате, что и для -source-port.

-tcp-flags	SYN, ACK, FIN SYN определяет маску и флаги tcp-пакета. Пакет считается удовлетворяющим критерию, если из перечисленных флагов в первом списке в единичное состояние установлены флаги из второго списка. В качестве аргументов критерия могут выступать флаги SYN, ACK, FIN, RST, URG, PSH, а также зарезервированные идентификаторы ALL и NONE. ALL — значит ВСЕ флаги и NONE — НИ ОДИН флаг. Так, критерий -tcp-flags ALL NONE означает, что все флаги в пакете должны быть сброшены. Как и ранее, символ «!» означает инверсию критерия. Имена флагов в каждом списке должны разделяться запятыми, пробелы служат для разделения списков.
-icmp-type	Тип сообщения ICMP определяется номером или именем. Числовые значения определяются в RFC 792. Чтобы получить список имен ICMP значений выполните команду iptables - protocol icmp ^^.Символ «!» инвертирует критерий, например - icmp-type ! 8.
-state	Для использования данного критерия в правиле перед -state нужно явно указать -m state. Проверяется признак состояния соединения. Можно указывать 4 состояния: INVALID, ESTABLISHED, NEW и RELATED. INVALID подразумевает, что пакет связан с неизвестным потоком или соединением и, возможно содержит ошибку в данных или в заголовке. ESTABLISHED указывает на то, что пакет принадлежит уже установленному соединению, через которое пакеты идут в обоих направлениях. NEW подразумевает, что пакет открывает новое соединение или пакет принадлежит однонаправленному потоку. RELATED указывает на то, что пакет принадлежит уже существующему соединению, но при этом он открывает новое соединение. Примером тому может служить передача данных по FTP, или выдача сообщения ICMP об ошибке, которое связано с существующим TCP или UDP соединением. Признак NEW — это не то же самое, что установленный бит SYN в пакетах TCP, посредством которых открывается новое соединение, и, подобного рода пакеты могут быть потенциально опасны в случае, когда для защиты сети используется один сетевой экран.

6.3 Права доступа к файлам и каталогам

SMART Linux — система многопользовательская, поэтому вопрос об организации разграничения доступа к файлам и каталогам является одним из существенных вопросов, которые должна решать операционная система.

В основе механизмов разграничения доступа лежат имена пользователей и имена групп пользователей. В SMART Linux каждый пользователь имеет уникальное имя, под которым

он входит в систему (логируется). Кроме того, в системе создаётся некоторое число групп пользователей, причём каждый пользователь может быть включён в одну или несколько групп. Создаёт и удаляет группы суперпользователь, он же может изменять состав участников той или иной группы. Члены разных групп могут иметь разные права по доступу к файлам, например, группа администраторов может иметь больше прав, чем группа программистов.

В индексном дескрипторе каждого файла записаны имя так называемого владельца файла и группы, которая имеет права на этот файл. Первоначально, при создании файла его владельцем объявляется тот пользователь, который этот файл создал. Точнее — тот пользователь, от чьего имени запущен процесс, создающий файл. Группа тоже назначается при создании файла — по идентификатору группы процесса, создающего файл. Владельца и группу файла можно поменять в ходе дальнейшей работы с помощью команд «chown» и «chgrp».

Выполним команду «ls -l». Но зададим ей в качестве дополнительного параметра имя конкретного файла, например, файла, задающего саму команду ls.

В данном случае владельцем файла является пользователь root и группа root. Но нас сейчас в выводе этой команды больше интересует первое поле, определяющее тип файла и права доступа к файлу. Это поле в приведённом примере представлено цепочкой символов «-rwxr-xr-x». Эти символы можно условно разделить на 4 группы.

Первая группа, состоящая из единственного символа, определяет тип файла. Этот символ в соответствии с возможными типами файлов может принимать такие значения:

- **-** — обычный файл;
- **d** — каталог;
- **b** — файл блочного устройства;
- **c** — файл символьного устройства;
- **s** — доменное гнездо (socket);
- **p** — именованный канал (pipe);
- **l** — символическая ссылка (link).

Далее следуют три группы по три символа, которые и определяют права доступа к файлу соответственно для владельца файла, для группы пользователей, которая сопоставлена данному файлу, и для всех остальных пользователей системы. В нашем примере права доступа для владельца определены как «rwx», что означает, что владелец (root) имеет право читать файл (r), производить запись в этот файл (w) и запускать файл на выполнение (x). Замена любого из этих символов прочерком будет означать, что пользователь лишается соответствующего права. В том же примере мы видим, что все остальные пользователи (включая и тех, которые вошли в группу root) лишены права записи в этот файл, т. е. не могут файл редактировать и вообще как-то изменять.

Права доступа и информация о типе файла хранятся в индексных дескрипторах в отдельной структуре, состоящей из двух байтов, т. е. из 16 бит. Четыре бита из этих 16-ти отведены для кодированной записи о типе файла. Следующие три бита задают особые свойства исполняемых файлов. И, наконец, оставшиеся 9 бит определяют права доступа

к файлу. Эти 9 бит разделяются на 3 группы по три бита. Первые три бита задают права пользователя, следующие три бита — права группы, последние 3 бита определяют права всех остальных пользователей (т. е. всех пользователей, за исключением владельца файла и группы файла). При этом, если соответствующий бит имеет значение «1», то право предоставляется, а если он равен «0», то право не предоставляется. В символьной форме записи прав единица заменяется соответствующим символом (r, w или x), а 0 представляется прочерком.

Право на чтение (r) файла означает, что пользователь может просматривать содержимое файла с помощью различных команд просмотра, например, командой «more» или с помощью любого текстового редактора. Но, подредактировав содержимое файла в текстовом редакторе, вы не сможете сохранить изменения в файле на диске, если не имеете права на запись (w) в этот файл. Право на выполнение (x) означает, что вы можете загрузить файл в память и попытаться запустить его на выполнение как исполняемую программу. Конечно, если в действительности файл не является программой (или скриптом shell), то запустить этот файл на выполнение не удастся, но, с другой стороны, даже если файл действительно является программой, но право на выполнение для него не установлено, то он тоже не запустится.

Если выполнить ту же команду «ls -l», но в качестве последнего аргумента ей указать не имя файла, а имя каталога, мы увидим, что для каталогов тоже определены права доступа, причём они задаются теми же самыми символами rwx. Например, выполнив команду «ls –l /», мы увидим, что каталогу bin соответствует строка:

drwxr-xr-x 2 root root 2048 Jun 21 21:11 bin

Естественно, что по отношению к каталогам трактовка понятий «право на чтение», «право на запись» и «право на выполнение» несколько изменяется. Право на чтение по отношению к каталогам легко понять, если вспомнить, что каталог — это просто файл, содержащий список файлов в данном каталоге. Следовательно, если вы имеете право на чтение каталога, то вы можете просматривать его содержимое (этот самый список файлов в каталоге). Право на запись тоже понятно — имея такое право, вы сможете создавать и удалять файлы в этом каталоге, т. е. просто добавлять в каталог или удалять из него запись, содержащую имя какого-то файла и соответствующие ссылки. Право на выполнение в данном случае означает право переходить в этот каталог. Если вы, как владелец, хотите дать доступ другим пользователям на просмотр какого-то файла в своём каталоге, вы должны дать им право доступа в каталог, т. е. дать им «право на выполнение каталога». Более того, надо дать пользователю право на выполнение для всех каталогов, стоящих в дереве выше данного каталога. Поэтому в принципе для всех каталогов по умолчанию устанавливается право на выполнение как для владельца и группы, так и для всех остальных пользователей. И, уж если вы хотите закрыть доступ в каталог, то лишите всех пользователей (включая группу) права входить в этот каталог.

После прочтения предыдущего абзаца может показаться, что право на чтение каталога не даёт ничего нового по сравнению с правом на выполнение. Однако разница в этих правах все же есть. Если задать только право на выполнение, вы сможете войти в каталог, но не увидите там ни одного файла (этот эффект особенно наглядно проявляется в том случае, если вы пользуетесь каким-то файловым менеджером, например, программой Midnight Commander). Если

вы имеете право доступа в каком-то из подкаталогов этого каталога, то вы можете перейти в него (командой cd), но, как говорится «вслепую», по памяти, потому что списка файлов и подкаталогов текущего каталога вы не увидите.

Алгоритм проверки прав пользователя при обращении к файлу можно описать следующим образом. Система вначале проверяет, совпадает ли имя пользователя с именем владельца файла. Если эти имена совпадают (т. е. владелец обращается к своему файлу), то проверяется, имеет ли владелец соответствующее право доступа: на чтение, на запись или на выполнение (не удивляйтесь, суперпользователь может лишить некоторых прав и владельца файла). Если право такое есть, то соответствующая операция разрешается. Если же нужного права владелец не имеет, то проверка прав, предоставляемых через группу или через группу атрибутов доступа для остальных пользователей, уже даже не проверяются, а пользователю выдаётся сообщение о невозможности выполнения затребованного действия.

Если имя пользователя, обращающегося к файлу, не совпадает с именем владельца, то система проверяет, принадлежит ли владелец к группе, которая сопоставлена данному файлу (далее будем просто называть ее группой файла). Если принадлежит, то для определения возможности доступа к файлу используются атрибуты, относящиеся к группе, а на атрибуты для владельца и всех остальных пользователей внимания не обращается. Если же пользователь не является владельцем файла и не входит в группу файла, то его права определяются атрибутами для остальных пользователей. Таким образом, третья группа атрибутов, определяющих права доступа к файлу, относится ко всем пользователям, кроме владельца файла и пользователей, входящих в группу файла.

6.3.1 Chmod

Для изменения прав доступа к файлу используется команда chmod. Ее можно использовать в двух вариантах. В первом варианте вы должны явно указать, кому какое право даёте или кого этого права лишаете:

Chmod wXp <имя файла>

Где вместо символа <w> подставляется:

- либо символ «**u**» (т. е. пользователь, который является владельцем);
- либо «**g**» (группа);
- либо «**o**» (все пользователи, не входящие в группу, которой принадлежит данный файл);
- либо «**a**» (все пользователи системы, т. е. и владелец, и группа, и все остальные).

Вместо =<X> ставится:

- либо «**+**» (предоставить право);
- либо «**–**» (лишить соответствующего права);
- либо «**=**» (установить указанные права вместо имеющихся).

Вместо <p> — символ, обозначающий соответствующее право:

- «**r**» (чтение);
- «**w**» (запись);
- «**x**» (выполнение).

Второй вариант задания команды chmod (он используется чаще) основан на цифровом представлении прав. Для этого мы кодируем символ «**r**» цифрой 4, символ «**w**» — цифрой 2, а символ «**x**» — цифрой 1. Для того, чтобы предоставить пользователям какой-то набор прав, надо сложить соответствующие цифры. Получив, таким образом, нужные цифровые значения для владельца файла, для группы файла и для всех остальных пользователей, задаём эти три цифры в качестве аргумента команды chmod (ставим эти цифры после имени команды перед вторым аргументом, который задаёт имя файла). Например, если надо дать все права владельцу (4+2+1=7), право на чтение и запись — группе (4+2=6), и не давать никаких прав остальным, то следует дать такую команду:

chmod 760 <file>

Если вы знакомы с двоичным кодированием восьмеричных цифр, то вы поймёте, что цифры после имени команды в этой форме ее представления есть ни что иное, как восьмеричная запись тех самых 9 бит, которые задают права для владельца файла, группы файла и для всех пользователей.

Выполнять смену прав доступа к файлу с помощью команды chmod может только сам владелец файла или суперпользователь. Для того, чтобы иметь возможность изменить права группы, владелец должен дополнительно быть членом той группы, которой он хочет дать права на данный файл.

Надо рассказать ещё о трёх возможных атрибутах файла, устанавливаемых с помощью той же команды chmod. Это те самые атрибуты для исполняемых файлов, которые в индексном дескрипторе файла в двухбайтовой структуре, определяющей права на файл, занимают позиции 5-7, сразу после кода типа файла.

Первый из этих атрибутов — так называемый «бит смены идентификатора пользователя». Смысл этого бита состоит в следующем.

Обычно, когда пользователь запускает некоторую программу на выполнение, эта программа получает те же права доступа к файлам и каталогам, которые имеет пользователь, запустивший программу. Если же установлен «бит смены идентификатора пользователя», то программа получит права доступа к файлам и каталогам, которые имеет владелец файла программы (таким образом, рассматриваемый атрибут лучше называть «битом смены идентификатора владельца»). Это позволяет решать некоторые задачи, которые иначе было бы трудно выполнить. Самый характерный пример — команда смены пароля passwd. Все пароли пользователей хранятся в файле /etc/passwd, владельцем которого является суперпользователь root. Поэтому программы, запущенные обычными пользователями, в том числе команда passwd, не могут производить запись в этот файл. А значит, пользователь как бы не может менять свой собственный пароль. Но для файла /usr/bin/passwd установлен «бит смены идентификатора владельца», каковым является пользователь root. Следовательно, программа смены пароля passwd запускается с правами root и получает право записи в файл

/etc/passwd (уже средствами самой программы обеспечивается то, что пользователь может изменить только одну строку в этом файле).

Установить «бит смены идентификатора владельца» может суперпользователь с помощью команды:

chmod u+s <program>

Аналогичным образом работает «бит смены идентификатора группы».

Еще один возможный атрибут исполняемого файла — это «бит сохранения задачи» или «**sticky bit**» (дословно — «бит прилипчивости»). Этот бит указывает системе, что после завершения программы надо сохранить ее в оперативной памяти. Удобно включить этот бит для задач, которые часто вызываются на выполнение, так как в этом случае экономится время на загрузку программы при каждом новом запуске. Этот атрибут был необходим на старых моделях компьютеров. На современных быстродействующих системах он используется редко.

Если используется цифровой вариант задания атрибутов в команде chmod, то цифровое значение этих атрибутов должно предшествовать цифрам, задающим права пользователя:

При этом веса этих битов для получения нужного суммарного результата задаются следующим образом:

- **4** — «бит смены идентификатора пользователя»;
- **2** — «бит смены идентификатора группы»;
- **1** — «бит сохранения задачи (sticky bit)».

Если какие-то из этих трёх битов установлены в 1, то несколько изменяется вывод команды «**ls -l**» в части отображения установленных атрибутов прав доступа. Если установлен в 1 «бит смены идентификатора пользователя», то символ «**x**» в группе, определяющей права владельца файла, заменяется символом «**s**». Причём, если владелец имеет право на выполнение файла, то символ «**x**» заменяется на маленькое «s», а если владелец не имеет права на выполнение файла (например, файл вообще не исполняемый), то вместо «**x**» ставится «**S**». Аналогичные замены имеют место при задании «бита смены идентификатора группы», но заменяется символ «**x**» в группе атрибутов, задающих права группы. Если равен 1 «бит сохранения задачи (sticky bit)», то заменяется символ «**x**» в группе атрибутов, определяющей права для всех остальных пользователей, причём «**x**» заменяется символом «**t**», если все пользователи могут запускать файл на выполнение, и символом «**T**», если они такого права не имеют.

Таким образом, хотя в выводе команды «**ls –l**» не предусмотрено отдельных позиций для отображения значений битов смены идентификаторов и бита сохранения задачи, соответствующая информация выводится.

6.3.2 Umask

Umask (от англ. user file creation mode mask — маска режима создания пользовательских файлов) — функция среды POSIX, изменяющая права доступа, которые присваиваются новым файлам и директориям по умолчанию. Права доступа файлов, созданных при

конкретном значении umask, вычисляются при помощи следующих побитовых операций (umask обычно устанавливается в восьмеричной системе счисления): побитовое «И» между унарным дополнением аргумента (используя побитовое «НЕ») и режимом полного доступа.

Фактически, umask указывает, какие биты следует сбросить в выставляемых правах на файл — каждый установленный бит umask запрещает выставление соответствующего бита прав. Исключением из этого запрета является бит исполняемости, который для обычных файлов зависит от создающей программы.

(Трансляторы ставят бит исполняемости на создаваемые файлы, другие программы — нет), а для каталогов следует общему правилу. Umask 0 означает, что следует (можно) выставить все биты прав (rwxrwxrwx), umask 777 запрещает выставление любых прав.

Допустим, что значение umask равняется 174, тогда каждый новый файл будет иметь права доступа 602, а каждая новая директория 603.

6.3.3 Chown

Chown (от англ. change owner) — утилита, изменяющая владельца и/или группу для указанных файлов. В качестве имени владельца/группы берётся первый аргумент, не являющийся опцией. Если задано только имя пользователя (или числовой идентификатор пользователя), то данный пользователь становится владельцем каждого из указанных файлов, а группа этих файлов не изменяется. Если за именем пользователя через двоеточие следует имя группы (или числовой идентификатор группы), без пробелов между ними, то изменяется также и группа файла.

Синтаксис:

chown [-cfhvR] [--dereference] [--reference=rfile] <пользователь>[:<группа>] <файл>...

Основные опции, используемые утилитой chown, приведены в таблице.

Опция	Значение опции
-c, --changes	Подробно описывать действие для каждого файла, владелец которого действительно изменяется.
-f, --silent, --quiet	Не выдавать сообщения об ошибках для файлов, чей владелец не может быть изменён.
-h, --no-dereference	Работать с самими символьными ссылками, а не с файлами, на которые они указывают. Данная опция доступна только если имеется системный вызов lchown.
-R, - --recursive.	Рекурсивное изменение владельца каталогов и их содержимого
-v, --verbose	Подробное описание действия (или отсутствия действия) для каждого файла.

--dereference	Изменить владельца файла, на который указывает символьная ссылка, вместо самой символьной ссылки.
--reference= - rfile	Изменить владельца файла на того, который является владельцем файла.

6.4 Systemd — управление компонентами ОС

Systemd — менеджер системы и сервисов в операционной системе SMART Linux.

Systemd реализует концепцию юнитов systemd. Юниты представлены конфигурационными файлами, размещёнными в одной из директорий:

- **/usr/lib/systemd/system/** — юниты из установленных пакетов RPM;
- **/run/systemd/system/** — юниты, созданные в рантайме. Этот каталог приоритетнее каталога с установленными юнитами из пакетов;
- **/etc/systemd/system/** — юниты, созданные и управляемые системным администратором. Этот каталог приоритетнее каталога юнитов, созданных в рантайме.

Юниты содержат информацию о системных сервисах, прослушиваемых сокетах, сохраненных снапшотах состояний системы и других объектах, относящихся к системе инициализации.

Типы юнитов systemd:

- **.service** — системный сервис;
- **.target** — группа юнитов systemd;
- **.automount** — точка автомонтирования файловой системы;
- **.device** — файл устройства, распознанного ядром;
- **.mount** — точка монтирования файловой системы;
- **.path** – файл или директория в файловой системе;
- **.scope** – процесс, созданный извне;
- **.slice** – группа иерархически организованных юнитов, управляющая системными процессами;
- **.snapshot** – сохранённое состояние менеджера systemd;
- **.socket** – сокет межпроцессного взаимодействия;
- **.swap** – свап-устройство или свап-файл (файл подкачки);
- **.timer** – таймер systemd.

Во время загрузки systemd прослушивает сокеты для всех системных сервисов после старта сервисов. Это позволяет systemd не только запускать сервисы параллельно, но также дает возможность перезапускать сервисы без потери любых отправленных им сообщений,

пока сервисы были недоступны. Соответствующий сокет остается доступным и все сообщения выстраиваются в очередь.

Системные сервисы, использующие D–Bus для межпроцессного взаимодействия, могут быть запущены по требованию, когда клиентское приложение пытается связаться с ними.

Системные сервисы, поддерживающие активацию, основанную на устройствах, могут быть запущены, когда определённый тип оборудования подключается или становится доступным.

Системные сервисы могут поддерживать этот вид активации, если изменяется состояние папки или директории.

Система может сохранять состояние всех юнитов и восстанавливать предыдущее состояние системы.

Systemd отслеживает и управляет точками монтирования и автомонтирования.

Агрессивная параллелизация Systemd запускает системные сервисы параллельно из-за использования активации, основанной на сокетах. В комбинации с сервисами, поддерживающими активацию по требованию, параллельная активация значительно уменьшает время загрузки системы.

До активации и деактивации юнитов systemd вычисляет их зависимости, создает временную транзакцию и проверяет целостность этой транзакции. Если транзакция не целостная, systemd автоматически пытается исправить ее и удалить не требующиеся задания из нее до формирования сообщения об ошибке.

SystemD полностью поддерживает скрипты инициализации SysV, как описано в спецификации Linux Standard Base (LSB), что упрощает переход на systemd. По способу использования сервисные юниты service напоминают скрипты инициализации. Для просмотра, старта, остановки, перезагрузки, включения или выключения системных сервисов используется команда systemctl. Команды service и chkconfig по-прежнему включены в систему, но только по соображениям совместимости.При использовании systemctl указывать расширение файла не обязательно. Основные команды systemctl описаны в таблице.

Команда	Описание
systemctl start name.service	Запуск сервиса
systemctl stop name.service	Остановка сервиса
systemctl restart name.service	Перезапуск сервиса
systemctl try-restart name.service	Перезапуск сервиса только, если он запущен
systemctl reload name.service	Перезагрузка конфигурации сервиса
systemctl status - name.service	Проверка, запущен ли сервис с детальным выводом состояния сервиса
systemctl is-active	Проверка, запущен ли сервис с простым ответом: name.service active или inactive

systemctl list-units--type service --all	Отображение статуса всех сервисов
systemctl enable	Активирует сервис (позволяет стартовать во время запуска системы)
systemctl disable name.service	Деактивирует сервис
systemctl reenable name.service	Деактивирует сервис и сразу активирует его
systemctl is–enabled name.service	Проверяет, активирован ли сервис
systemctl list-unit-files --type service	Отображает все сервисы и проверяет, какие из них активированы
systemctl mask name.service	Заменяет файл сервиса симлинком на /dev/null, делая юнит недоступным для systemd
systemctl unmask - name.service	Возвращает файл сервиса, делая юнит доступным для systemd.

Файлы целей systemd.target предназначены для группировки вместе других юнитов systemd через цепочку зависимостей. Например юнит graphical.target, использующийся для старта графической сессии, запускает системные сервисы GNOME Display Manager (gdm.service) и Accounts Service (accounts–daemon.service) и активирует multi–user.target. В свою очередь multi–user.target запускает другие системные сервисы, такие как Network Manager (NetworkManager.service) или D-Bus (dbus.service) и активирует другие целевые юниты basic.target.

В SMART Linux присутствуют предопределённые цели, похожие на стандартный набор уровней запуска. По соображениям совместимости они также имеют псевдонимы на эти цели, которые напрямую отображаются в уровнях запуска SysV.

- **poweroff.target (runlevel0.target)** — завершение работы и отключение системы;
- **rescue.target (runlevel1.target)** — настройка оболочки восстановления;
- **multi–user.target (runlevel2.target, runlevel3.target, runlevel4.target)** — настройка неграфической многопользовательской системы;
- **graphical.target (runlevel5.target)** — настройка графической многопользовательской системы;
- **reboot.target (runlevel6.target)** — выключение и перезагрузка системы.

Команды runlevel и telinit по-прежнему доступны, но оставлены в системе по соображениям совместимости. Рекомендуется использовать systemctl для изменения или настройки системных целей.

Для определения, какой целевой юнит используется по умолчанию, полезна следующая команда:

systemctl get-default

Для просмотра всех загруженных целевых юнитов воспользуйтесь командой:

systemctl list-units --type target

а для просмотра вообще всех целевых юнитов командой:

systemctl list-units --type target --all

Для изменения цели по умолчанию поможет команда:

systemctl get-default

Для изменения текущей цели:

systemctl set-default name.target

Команда запустит целевой юнит и все его зависимости и немедленно остановит все остальные.

В SMART Linux systemctl заменяет значительное количество команд управления питанием. Прежние команды сохранены для совместимости, но рекомендуется использовать systemctl:

systemctl isolate name.target

Systemd позволяет управлять удалённой машиной по SSH. Для управления используйте команду:

systemctl --host <user_name>@<host_name> <command>

где <user_name> — имя пользователя, <host_name> — имя хоста, которым осуществляется удалённое управление, а <command> — выполняемая команда systemd.

Подробная информация о всех параметрах файла .service есть в соответствующем разделе документации по systemd.

Давайте посмотрим на секцию [Unit]. Она содержит общую информацию о сервисе. Такая секция есть не только в сервис-юнитах, но и в других юнитах (например, при управлении устройствами, точками монтирования и т.д.). В примере мы даем описание сервиса и указываем на то, что демон должен быть запущен после Syslog.

В следующей секции [Service] непосредственно содержится информация о нашем сервисе. Используемый параметр ExecStart указывает на исполняемый файл нашего сервиса. В Type мы указываем, как сервис уведомляет systemd об окончании запуска.

Финальная секция [Install] содержит информацию о цели, в которой сервис должен стартовать. В данном случае мы говорим, что сервис должен быть запущен, когда будет активирована цель multi–user.target.

Это минимальный работающий файл сервиса systemd. Написав свой, для тестирования скопируйте его в /etc/systemd/system/<имя_сервиса>.service. Выполните команду:

Systemctl daemon-reload

Systemd узнает о сервисе, и вы сможете его запустить.

6.5 PAM

PAM или Pluggable Authentication Modules (подключаемые модули аутенти- фикации) — это модульный подход к системе аутентификации. Они позволяют сторонним службам предоставлять модуль аутентификации для обеспечения доступа к службе для систем с поддержкой PAM. Службы, использующие PAM для аутентификации, могут использовать их сразу же, без необходимости дополнительной пересборки.

На сервере PAM (Pluggable Authentication Modules) могут использоваться для управления аутентификацией (как часть управления предоставления доступа). При использовании PAM сервисам нет необходимости поддерживать собственную систему аутентификации. Вместо этого они полагаются на модули PAM, доступные в системе.

Любой сервис при необходимости может использовать собственную конфигурацию PAM, хотя в большинстве случаев аутентификация выполняется одинаково во множестве сервисов. Вызывая модули PAM, сервисы могут поддерживать двухфакторную аутентификацию «из коробки», сразу же использовать централизованные хранилища аутентификационных средств и многое другое. PAM предоставляют гибкую модульную архитектуру для следующих сервисов:

- управление аутентификацией — проверяет, существует ли пользователь под которым пытаются зайти;
- управление учётными записями — проверяет, что пароль пользователя не истёк или имеет ли пользователь право обращаться к определённому сервису;
- управление сеансами — выполняет определённые задачи во время входа или выхода пользователя из системы (аудит, монтирование файловых систем и так далее);
- управление паролями — предлагает интерфейс для сброса пароля и тому подобное.

При работе с PAM администраторы очень быстро поймут принципы, по которым функционирует PAM.

Во-первых, это «независимость от бэк-энда». Приложениям, поддерживающим PAM, нет необходимости учитывать низкоуровневую логику, чтобы работать с бэк-эндами. Используя PAM, приложения отделяют логику работы бэк-энда от своей. Всё, что им нужно сделать — это вызвать функцию PAM.

Другим принципом является «независимость от конфигурации». Администраторам не нужно знать, как настраивать десятки различных приложений, чтобы заставить их поддерживать аутентификацию. Вместо этого им достаточно воспользоваться одной конфигурационной структурой, предоставляемой PAM.

Последним принципом, являющимся также частью названия PAM, является «подключаемая архитектура». Когда необходимо интегрировать новый бэк-энд, всё, что нужно сделать администратору — это установить библиотеку для этого бэк-энда (большинство модулей используют один файл настроек). Начиная с этого момента модуль становится доступен для использования приложениями. Администраторы могут настроить аутентификацию для использования этого бэк-энда и просто перезапустить приложение.

Приложения, для которых необходимо использование PAM, линкуются с библиотекой PAM (libpam) и могут вызывать нужные функции работы с указанными выше службами. Кроме этого, в приложении не нужно ничего реализовывать специфичного для работы с этими сервисами, так как эту задачу на себя берёт PAM. И когда пользователь захочет аутентифицироваться, скажем, в веб-приложении, то это приложение вызывает PAM (передавая ему идентификатор и, возможно, пароль или запрос) и проверяет возвращаемые данные, чтобы принять решение, аутентифицировался ли пользователь и имеет ли он доступ к приложению. Внутренней задачей PAM является определение, где необходимо аутентифицировать пользователя.

Сильной стороной PAM является то, что любой желающий может создать модули PAM для интеграции с любым поддерживающим PAM сервисом или приложением. Если какая-нибудь компания выпускает новый сервис для аутентификации, всё, что нужно будет сделать, — это предоставить для взаимодействия с этим сервисом модуль PAM, после чего любое использующее PAM приложение сможет незамедлительно работать с этим сервисом: нет необходимости что-то пересобирать или улучшать.

Другой важной особенностью PAM является то, что они поддерживают объединение в цепочки нескольких модулей. Пример конфигурационного файла PAM для некоего сервиса:

```
# Аутентификация
auth required pam_env.so
auth required pam_ldap.so
# Управление учётными записями
account required pam_ldap.so
# Управление паролями
password required pam_ldap.so
# Управление сеансами
session optional pam_loginuid.so
session required pam_selinux.so close
session required pam_env.so
session required pam_log.so level=audit
session required pam_selinux.so open multiple
session optional pam_mail.so
```

Видно, что конфигурационный файл разделён на четыре области сервисов, которые поддерживают PAM: аутентификация, управление учётными записями, управление паролями и управление сеансами.

Каждый из этих разделов в файле вызывает один или несколько модулей PAM. Например, «pam_env.so» устанавливает переменные среды, которые могут быть использованы последующими модулями. Код, возвращаемый модулем PAM, вместе с управляющими директивами (в данном примере — «required» или «optional») позволяет PAM решать, что делать дальше.

PAM поддерживают следующие управляющие директивы:

- **required** — указанный модуль PAM должен вернуть код успеха для того, чтобы весь сервис (например, аутентификация) была успешен. Если модуль PAM вернёт код неудачи, остальные модули будут всё равно вызваны (хотя уже точно известно, что сам сервис будет недоступен);
- **requisite** — указанный модуль PAM должен вернуть код успеха для того, чтобы весь сервис был доступен. В отличие от «required», если модуль PAM вернёт код неудачи, директива сразу же завершится, и сам сервис будет недоступен;
- **sufficient** — если указанный модуль PAM вернёт код успеха, весь сервис будет разрешён. Оставшиеся модули PAM не будут проверяться. Однако, если модуль PAM вернёт код неудачи, оставшиеся модули пройдут проверку, а неудача данного модуля не будет приниматься во внимание;
- **optional** — код успеха или неудачи указанного модуля PAM будут иметь значение, если это единственный модуль в стеке.

Цепочки модулей позволяют выполнить множественную аутентификацию, выполнить несколько задач в процессе создания сеанса и тому подобное.

Так как конфигурационные файлы PAM управляют процессом аутентифика- ции в приложении, очень важно правильно с ними взаимодействовать. Файлы обычно располагаются в каталоге /etc/pam.d/. В больших организациях или в требовательных к безопасности системах любая модификация конфигурационных файлов PAM должна подвергаться соответствующему аудиту.

Это же относится к тому месту, где располагаются модули PAM (/lib/security или /lib64/security). Помимо файлов-сценариев для некоторых модулей могут использоваться дополнительные файлы конфигурации. Все они расположены в каталоге /etc/security и каждый файл предназначен для конкретной группы настроек.

- **time.conf** — здесь вы сможете ограничить время доступа пользователей с различных терминалов к различным сервисам. Например, запретить входить в систему с первой виртуальной консоли администратору во время выходных. Эти настройки обслуживает модуль «pam_time» и, соответственно, если вы желаете, чтобы ваши ограничения возымели действие, модуль должен быть прописан в соответствующем сценарии.
- **pam_env.conf** — здесь можно ограничить возможности в изменении отдельных переменных среды пользователями. Работает под руководством модуля «pam_env».
- **limits.conf** — здесь можно индивидуально или для группы ограничить: размер core-файла, максимальный допустимый размер файла, максимальное количество открытых файлов, запущенных процессов, сколько раз можно одновременно зайти в систему и т.д. Руководящий модуль «pam_limits».
- **access.conf** — так как PAM имеет средства аутентификации по сети, то подобные настройки являются полезными ибо контролируется не только «кто может зайти» или «не зайти», но и «откуда». Контролируется «pam_access».

- **group.conf** — можно указать, какой группе будет принадлежать служба, запущенная определенным пользователем, в определённое время с определённого терминала. Контролируется «pam_time» и «pam_group».

Список модулей:

- **pam_cracklib**: тип «password». Проверяет ваш пароль на стойкость, не является ли он, например, палиндромом (примечание — это не обязательно при использовании модуля «pam_unix»). Полезен для программ, задающих пароли. Полезные параметры: retry=N — даётся N попыток на исправление ошибки, diffok=N — должно быть изменено минимум N символов при смене пароля, minlen=N — минимальный размер пароля, dcredit=N ucredit=N lcredit=N ocredit=N - в пароле должно присутствовать минимум N цифр, строчных, прописных букв и других символов.
- **pam_deny**: тип любой. Всегда перекрывает доступ.
- **pam_env**: тип «auth». Контролирует сохранность переменных среды. Полезный параметр conffile=S — задаёт альтернативное название файла конфигурации.
- **pam_ftp**: тип «auth». Предназначен для организации анонимного доступа.

То есть получив имя пользователя «anonymous», ждёт в качестве пароля что-то похожее на его почтовый адрес. Полезные параметры: ignore — не обращать внимание похож ли пароль на почтовый адрес, users=XXX,YYY — позволяет анонимный вход для пользователей из этого списка.

- **pam_group**: тип «auth». Предназначение ясно из описания конфигурационного файла «group.conf».
- **pam_lastlog**: тип «auth». Сообщает о времени и месте входа в систему. Обновляет /var/log/wtmp файл.
  Полезные параметры: «nodate», «noterm», «nohost», «silent» — не выводить в сообщении даты, терминала, машины или вообще ничего, «never» — если пользователь никогда ранее не появлялся, то его приветствуют.
- **pam_limits**: тип «session». Предназначение указано выше при описании файла «limits.conf».
  Полезный параметр: conf=S — альтернативное имя конфигурационного файла.
- **pam_listfile**: тип «auth». Предназначен для организации доступа на основе конфигурационных файлов-списков, например, /etc/ftpaccess. Для примера смотрите соответствующие файлы сценариев. Возможные параметры: onerr=succeed|fail; sence=allow|deny; file=filename; item=user|tty|rhost|ruser| group| shell apply=user|@group. Первый параметр задаёт возвращаемое значение в случае неудачного поиска. Второй — в случае удачного поиска. Третий — имя файла со списком. Четвёртый — тип элемента в списке. Последний вносит дополнительные ограничения, если тип объявлен «tty», «rhost» или «shell».
- **pam_mail**: тип «auth». Сообщается о наличии почты, если таковая имеется. Полезные параметры: dir=S — путь к каталогу почтовых очередей, noenv — не устанавливать переменную среды MAIL, close — сообщать если есть почта у пользователей

с аннулированными бюджетами, nopen — не печатать какой-либо почтовой информации, если пользовательский бюджет только что заведён.

- **pam_nologin**: тип «auth». Стандартная реакция на наличие файла /etc/no- login. Когда он присутствует, в систему может зайти только root, а остальным будет выдано на экран содержимое этого файла;
- **pam_permit**: тип любой. Использование данного модуля ОПАСНО и применимо только в критических ситуациях. Всегда даёт допуск.
- **pam_pwdb**: тип любой. Замещает модули серии «pam_unix...» . Использует интерфейс библиотеки «libpwdb» (пользовательские базы данных), что повышает независимость системы аутентификации от способа хранения пользовательских данных (NIS или passwd или shadow). Полезные параметры: nullok — можно использовать пустые пароли, md5, shadow, bigcrypt — различные способы шифрования пароля.
- **pam_radius**: тип «session». Позволяет осуществлять аутентификацию через RADIUS сервер.
- **pam_rhosts_auth**: тип «auth». Механизм работы этого модуля основывается на анализе содержимого файлов hosts.equiv и .rhosts, используемых для аутентификации таких служб как rlogin и rsh. Полезные параметры: no_hosts_equiv — игнорировать содержимое файла hosts.equiv, no_rhosts — игнорировать содержимое файлов. Rhosts, suppres s — охраняет системные журналы от потока малозначимых сообщений, в частности, когда используется контрольный флаг «sufficient».
- **pam_root_ok**: тип auth. Используется в случае, когда администратору получать доступ к сервису без введения пароля. Этот модуль допускает пользователя к сервису только если его uid равен 0.
- **pam_securetty** — включает в проверку файл /etc/securitty. Суперпользова- тель сможет зайти только на указанных там терминалах.
- **pam_time**: тип account. Смотри описание устройства конфигурационного файла time.conf;
- **pam_warn**: тип «auth» и «password». Просто ведёт записи в системных журналах, например, при смене пароля.
- **pam_weel**: тип «auth». Права суперпользователя может использовать только пользователь группы «wheel» (группа root). Полезные параметры: group=XXX — использовать указанную группу вместо стандартной нулевой, deny — инвертирование действия алгоритма, запрещается получать права суперпользователя пользователям указанной группы, используется вместе с предыдущим параметром, trust — избавляет.

В PAM модулях важен порядок следования строк, поэтому структура редактируемых файлов после изменений должна быть следующая:

cat /etc/pam.d/password-auth

```
# Generated by authselect on Mon May 31 07:25:19 2021
# Do not modify this file manually.
auth required pam_env.so
```

auth required pam_faillock.so preauth silent audit deny=4 unlock_time=900 auth
required pam_faildelay.so delay=2000000
auth [default=1 ignore=ignore success=ok] pam_usertype.so isregular auth
[default=1 ignore=ignore success=ok] pam_localuser.so
auth sufficient pam_unix.so nullok try_first_pass
auth [default=die] pam_faillock.so authfail audit deny=4 unlock_time=900 auth
[default=1 ignore=ignore success=ok] pam_usertype.so isregular auth sufficient
pam_sss.so forward_pass
auth required pam_deny.so
account required pam_unix.so
account sufficient pam_localuser.so
account sufficient pam_usertype.so issystem
account [default=bad success=ok user_unknown=ignore] pam_sss.so
account required pam_permit.so
account required pam_faillock.so
password requisite pam_pwquality.so try_first_pass local_users_only
password sufficient pam_unix.so sha512 shadow nullok try_first_pass
use_authtok
password sufficient pam_sss.so use_authtok
password required pam_deny.so
session optional pam_keyinit.so revoke
session required pam_limits.so
-session optional pam_systemd.so
session [success=1 default=ignore] pam_succeed_if.so service in crond
quiet use_uid
session required pam_unix.so
session optional pam_sss.so

cat /etc/pam.d/system-auth

 # Generated by authselect on Mon May 31 07:25:19 2021
 # Do not modify this file manually.
auth required pam_env.so
auth required pam_faillock.so preauth silent audit deny=4 unlock_time=900 auth
required pam_faildelay.so delay=2000000
auth sufficient pam_fprintd.so
auth [default=1 ignore=ignore success=ok] pam_usertype.so isregular auth
[default=1 ignore=ignore success=ok] pam_localuser.so

auth sufficient pam_unix.so nullok try_first_pass
auth [default=die] pam_faillock.so authfail audit deny=4 unlock_time=900 auth [default=1
ignore=ignore success=ok] pam_usertype.so isregular auth sufficient pam_sss.so forward_pass
auth required pam_deny.so
account required pam_unix.so
account sufficient pam_localuser.so
account sufficient pam_usertype.so issystem
account [default=bad success=ok user_unknown=ignore] pam_sss.so account
required pam_permit.so
account required pam_faillock.so
password requisite pam_pwquality.so try_first_pass local_users_only
password sufficient pam_unix.so sha512 shadow nullok try_first_pass
use_authtok
password sufficient pam_sss.so use_authtok
password required pam_deny.so
session optional pam_keyinit.so revoke
session required pam_limits.so
-session optional pam_systemd.so
session [success=1 default=ignore] pam_succeed_if.so service in crond
quiet use_uid
session required pam_unix.so
session optional pam_sss.so

Здесь параметр deny= отвечает за число отслеживаемых неуспешных попыток, после которых произойдёт блокировка, а unlock_time= время в секундах, на которое учётная запись будет заблокирована.

6.6 Изменение приоритета процесса

Утилита nice — программа, предназначенная для запуска процессов с изменённым приоритетом nice. Приоритет nice (целое число) процесса используется планировщиком процессов ядра ОС при распределении процессорного времени между процессами.

Приоритет nice — число, указывающее планировщику процессов ядра ОС приоритет, который пользователь хотел бы назначить процессу.

Утилита nice, запущенная без аргументов, выводит приоритет nice, унаследованный от родительского процесса. nice принимает аргумент «смещение» в диапазоне от -20 (наивысший приоритет) до +19 (низший приоритет). Если указать смещение и путь к исполняемому файлу, утилита nice получит приоритет своего процесса, изменит его на указанное смещение и использует системный вызов семейства exec() для замещения кода своего процесса кодом из указанного исполняемого файла. Команда nice сделает то же, но

сначала выполнит системный вызов семейства fork() для запуска дочернего процесса (англ. sub-shell). Если смещение не указано, будет использовано смещение +10. Привилегированный пользователь (root) может указать отрицательное смещение.

Приоритет nice и приоритет планировщика процессов ядра ОС — разные числа. Число nice — приоритет, который пользователь хотел бы назначить процессу. Приоритет планировщика — действительный приоритет, назначенный процессу планировщиком. Планировщик может стремиться назначить процессу приоритет, близкий к nice, но это не всегда возможно, так как в системе может выполняться множество процессов с разными приоритетами. Приоритет nice является атрибутом процесса и, как и другие атрибуты, наследуется дочерними процессами. В выводе утилит top, ps, htop и др. приоритет nice называется «NI» — сокращение от «nice», а приоритет планировщика — «PRI» — сокращение от «priority». Обычно, NI = PRI - 20, но это верно не всегда. По умолчанию NI=0, соответственно PRI=20.

Планировщик процессов ядра ОС поддерживает приоритеты от 0 (реальное время) до 139 включительно. Приоритеты -20...+19 утилиты или команды nice соответствуют приоритетам 100...139 планировщика процессов. Другие приоритеты планировщика процессов можно установить командой «chrt» из пакета «util-linux».

Для изменения приоритета уже запущенных процессов используется утилита renice.

Синтаксис:

nice [-n <смещение>] [--adjustment=<смещение>]

[<команда> [<аргумент...>]]

Параметры:

- -n <смещение>;
- --adjustment=<смещение>.

Установить приоритет nice, равный сумме текущего приоритета nice и указанного числа <смещение>. Если этот аргумент не указан, будет использовано число 10.

Чтобы запустить процесс со значением nice, отличным от 10, можно использовать ключ -n.

nice -n 15 yes > /dev/null &

или

nice -15 yes > /dev/null

Чтобы установить значение nice ниже нуля, требуются права суперпользователя. В противном случае будет установлено значение «0». Попробуем задать значение nice «-1» без прав root:

nice -n -1 yes > /dev/null &

nice: cannot set niceness: Permission denied

Поэтому, чтобы задать значение nice меньше 0, необходимо запускать программу как root, или использовать sudo.

nice -n -1 yes > /dev/null &

У запущенной программы с помощью команды renice можно изменить назначенный приоритет. Предположим, что есть работающая программа «yes» со значением nice 10. Чтобы изменить его значение, можно использовать команду «renice» со значением nice и PID процесса. Изменим значение nice на 15:

renice -n 15 -p 5645

Согласно правилам, обычный пользователь может только увеличивать значение nice (уменьшать приоритет) любого процесса. Если попробовать изменить значение nice с 15 до 10, мы получим следующее сообщение об ошибке:

renice: failed to set priority for 5645 (process ID):

Permission denied

Также, команда renice позволяет суперпользователю изменять значение nice процессов любого пользователя. Это делается с помощью ключа «-u». Следующая команда изменяет значение приоритета всех процессов пользователя на -19:

renice -n -19 -u lubos

1000 (user ID) old priority 0, new priority -19

6.7 Экспорт данных пользователя

Для экспортирования данных используется утилита cp.

Cp — команда, предназначенная для копирования файлов из одного в другие каталоги (возможно, с другой файловой системой). Исходный файл остаётся неизменным, имя созданного файла может быть таким же, как у исходного, или измениться.

Чтобы скопировать файл:

cp [-f] [-h] [-i] [-p][--] <исходный_файл>

<целевой_файл>

Чтобы скопировать файл или файлы в другой каталог:

cp [-R] [-H | -L | -P] [-f | -i] [-pv]

<исходный_файл> ... <целевой_каталог>

Чтобы скопировать каталог в другой каталог (должен быть использован флаг -r или -R):

cp [-f] [-h] [-i] [-p] [--] { -r | -R }

<исходный_каталог> ... <целевой_каталог>

Чтобы скопировать каталог /media/fff1787/share1/load/ в каталог /media/ beac6e58/, с выводом имени копируемого файла, автоматическим пропуском существующих файлов, рекурсивно для вложенных каталогов.

cp -invR /media/fff1787/share1/load/ /media/beac6e58/

Опции команды описаны в таблице.

Опция	Значение опции
-R, -r, --recursive (recursive)	Копировать каталоги рекурсивно (то есть все подкаталоги и все файлы в подкаталогах);
-f (force)	Разрешает удаление целевого файла, в который производится копирование, если он не может быть открыт для записи;
-H	Используйте этот ключ, чтобы копировать символические ссылки. По умолчанию команда переходит по символическим ссылкам и копирует файлы, на которые те указывают;
-i (interactive)	Команда будет запрашивать, следует ли перезаписывать конечный файл, имя которого совпадает с именем исходного, то есть если в параметре <целевой_каталог> или <целевой_файл> встречается такое же имя файла, какое было задано в параметре <исходный_файл> или <исходный_каталог>, то запрашивается подтверждение. Для того, чтобы перезаписать файл, следует ввести «y» или его эквивалент для данной локали. Ввод любого другого символа приведёт к отмене перезаписи данного файла;
-n, - --no-clobber	Не перезаписывать существующий файл (отменяет предыдущий параметр -i);
-v, --verbose	Выводит имя каждого файла перед его копированием;
-p (preserve)	Повторяет следующие свойства исходного файла или каталога у целевого файла или каталога: • время последнего изменения и последнего доступа; • идентификатор пользователя и группы; • права доступа и биты SUID и SGID.

6.8 Лимиты ресурсов

В состав дистрибутива входит утилита ulimit, позволяющая управлять аппаратными ресурсами. limits.conf — это конфигурационный файл для pam_limits.so модуля. Он определяет ulimit лимиты для пользователей и групп. Конфигурация по умолчанию находится в /etc/security/limits.conf. Также присутствует возможность добавлять отдельные настройки для приложений в /etc/security/limits.d.

В данном примере для группы <limit_users> введены жёсткие ограничения.

Формат таков:

<группа>/<пользователь> <лимит>(жёсткий/мягкий) <параметр>

<значение>

Описание параметров:

- **core** — размер core-файлов (KB);
- **data** — максимальный размер данных (KB);
- **fsize** — максимальный размер файла (KB);
- **memlock** — максимальное заблокированное адресное пространство (KB);
- **nofile** — максимальное количество открытых файлов;
- **rss** — максимальный размер памяти для резидент-программ (KB);
- **stack** — максимальный размер стэка (KB);
- **cpu** — максимальное процессорное время (MIN);
- **nproc** — максимальное количество процессов;
- **as** — ограничение адресного пространства (KB);
- **maxlogins** — максимальное число одновременных регистраций в системе;
- **maxsyslogins** — максимальное количество учётных записей;
- **priority** — приоритет запущенных процессов;
- **locks** — максимальное количество блокируемых файлов пользователем;
- **sigpending** — максимальное количество сигналов, которые можно передать процессу;
- **msgqueue** — максимальный размер памяти для очереди POSIX сообщений (bytes);
- **nice** — максимальный приоритет, который можно выставить: [-20, 19];
- **rtprio** — максимальный приоритет времени выполнения;
- **chroot** — изменить директорию root'a (Debian-specific).

Вместо групп/юзера можно использовать групповой символ «*» (для всех) и групповой символ «%» для wildcast'а групп. Параметры вступают в силу сразу после перелогина пользователя. Что бы посмотреть какие действуют ограничения, используйте команду ulimit:

ulimit -aH

6.9 Монтирование файловых систем

mount — утилита командной строки для монтирования файловых систем.

Использование:

mount /dev/cdrom /mnt/cdrom

Устройство /dev/cdrom монтируется в каталог /mnt/cdrom, если он существует. Начиная от момента монтирования и пока пользователь не отмонтирует файловую систему (или туда не будет смонтировано что-то иное), в каталоге

/mnt/cdrom будет содержаться дерево каталогов устройства /dev/cdrom; те файлы и подкаталоги, которые раньше находились в /mnt/cdrom, сохранятся, но будут недоступны до размонтирования устройства /dev/cdrom.

Для размонтирования достаточно указать точку монтирования или имя устройства.

umount /dev/cdrom

В случае необходимости, при выполнении команды mount, можно указать дополнительные параметры монтирования.

-t <Тип файловой системы>

Обычно при монтировании определяется автоматически или берётся из файла конфигурации. Но в отдельных случаях нужно указывать тип файловой системы явно. Например, при монтировании DVD-диска.

mount /dev/cdrom /mnt/dvd -t udf

Если неправильно указать тип файловой системы, то команда mount выдаст сообщение об ошибке:

mount: wrong fs type, bad option, bad superblock on /dev/cdrom,

missing codepage or other error

In some cases useful info is found in syslog - try

dmesg | tail or so

и посоветует посмотреть в конец файла системных сообщений.

Unable to identify CD-ROM format.

В случае успешного монтирования обычно сообщается, что компакт-диск монтируется (по умолчанию) в режиме «только для чтения».

mount: block device /dev/cdrom is write-protected, mounting

read-only

-o <Атрибуты доступа>

- Доступ «только для чтения» (ro) или на «чтение и запись» (rw);
- Разрешение или запрещение запуска программ (noexec).

При запуске команды mount без параметров выводится список смонтированных файловых систем:

/dev/md/5 on / type reiserfs (rw,noatime)

proc on /proc type proc (rw)

sysfs on /sys type sysfs (rw,nosuid,nodev,noexec)

udev on /dev type tmpfs (rw,nosuid)

Чтобы облегчить процедуру монтирования, можно внести в файл конфигурации /etc/fstab соответствующие строки. Примерное содержимое для этого файла:

 # <fs> <mountpoint> <type> <opts> <dump/pass>

 # NOTE: If your BOOT partition is ReiserFS, add the notail option

to opts.

#/dev/BOOT /boot ext3 noauto,noatime 1 2

/dev/sda5 / reiserfs noatime 0 1

/dev/sda1 none swap sw 0 0

 # NOTE: The next line is critical for boot!

proc /proc proc defaults 0 0

В дальнейшем можно будет указывать в команде mount только имя устройства или точку монтирования — все дополнительные параметры будут браться из файла конфигурации.

Другое назначения файла конфигурации — автоматическое монтирование файловых систем при загрузке системы. Если не требуется монтировать определённые файловые системы, то для них в файле конфигурации нужно указать параметр noauto.

7. Управление пользователями

7.1 Общая информация

SMART Linux — многопользовательская операционная система. Также имеются группы пользователей, основное предназначение которых — облегчить управление большим количеством пользователей, а также более точно распределить права доступа к различным объектам системы. Пользователи и группы внутри системы обозначаются цифровыми идентификаторами — UID и GID, соответственно.

Пользователь может входить в одну или несколько групп. По умолчанию он входит в группу, совпадающую с его именем. Чтобы узнать, в какие ещё группы входит пользователь, введите команду id, вывод ее может быть примерно следующим:

uid=500(test) gid=500(test) группы=500(test),16(rpm)

Такая запись означает, что пользователь test (цифровой идентификатор 500) входит в группы test и rpm. Разные группы могут иметь разный уровень доступа к тем или иным каталогам; чем в большее количество групп входит пользователь, тем больше прав он имеет в системе.

7.2 Утилита passwd

Утилита passwd поддерживает традиционные опции «passwd» и утилиту «shadow». Синтаксис:

Passwd <опция> <имя_пользователя>

Основные опции утилиты passwd приведены в таблице.

Опция	Значение опции
-d, --delete	Удалить пароль для указанной записи
-f, --force	Форсировать операцию.
-k, --keep-tokens	Сохранить не устаревшие пароли.
-l, --lock	Блокировать указанную запись.
--stdin	Прочитать новые пароли из стандартного ввода.
-S, --status	Дать отчёт о статусе пароля в указанной записи.
-u, --unlock	Разблокировать указанную запись.
-?, --help	Показать справку и выйти.
--usage	Дать короткую справку по использованию.
-V, --version	Показать версию программы и выйти.

При успешном завершении «passwd» заканчивает работу с кодом выхода «0». Код выхода «1» означает, что произошла ошибка. Текстовое описание ошибки выводится на стандартный поток ошибок.

7.3 Добавления нового пользователя

Для добавления нового пользователя используйте команды «useradd» и «passwd». В примере в качестве первоначально введённого пароля указана последовательность символов «123.

В результате описанных действий в системе появился пользователь test1 с некоторым паролем. Пользователь в дальнейшем может поменять свой пароль при помощи команды «passwd».

Программа «useradd» имеет множество параметров, которые позволяют менять ее поведение по умолчанию. Например, можно принудительно указать, какой будет UID или какой группе будет принадлежать пользователь.

Синтаксис:

useradd [-u идентификатор [-o] [-i]] [-g группа] [-G группа[[,группа] . . .]] [-d каталог] [-s shell] [-c комментарий] [-m [-k skel_dir]] [-f inactive] [-e expire] [-p passgen] [-a событие[, . . .]] рег_имя

Основные опции команды useradd приведены в таблице.

Опция	Значение опции
-u	Идентификационный номер пользователя (UID). Этот номер должен быть неотрицательным целым числом, не превосходящим MAXUID, определённый в sys/param.h. По умолчанию используется следующий доступный (уникальный) не устаревший UID, больший 99. Эта опция игнорируется, если новое регистрационное имя будет администрироваться сетевой информационной службой (NIS).
-o	Эта опция позволяет сдублировать UID (сделать его не уникальным). Поскольку защита системы в целом, а также целостность контрольного журнала (audit trail) и учётной информации (accounting information) в частности, зависит от однозначного соответствия каждого UID определённому лицу, использовать эту опцию не рекомендуется (чтобы обеспечить учёт действий пользователей).
-i	Позволяет использовать устаревший идентификатор UID.
-g	Целочисленный идентификатор или символьное имя существующей группы. Эта опция задаёт основную группу (primary group) для нового пользователя. По умолчанию используется стандартная группа, указанная в файле /etc/default/useradd. Эта опция игнорируется, если новое регистрационное имя будет администрироваться сетевой информационной службой (NIS).

-G [[,] ...]	Один или несколько элементов в списке через запятую, каждый из которых представляет собой целочисленный идентификатор или символьное имя существующей группы. Этот список определяет принадлежность к дополнительным группам (supplementary group membership) для пользователя. Повторения игнорируются. Количество элементов в списке не должно превосходить - 1, поскольку общее количество дополнительных групп для пользователя плюс основная группа не должно превосходить. Эта опция игнорируется, если новое регистрационное имя будет администрироваться сетевой информационной службой (NIS).
-d	Начальный каталог (home directory) нового пользователя. Длина этого поля не должна превосходить 256 символов. По умолчанию используется HOMEDIR/, где HOMEDIR - базовый каталог для начальных каталогов новых пользователей, а <рег_имя> — регистрационное имя нового пользователя.
-s	Полный путь к программе, используемой в качестве начального командного интерпретатора для пользователя сразу после регистрации. По умолчанию это поле - пустое, что заставляет систему использовать стандартный командный интерпретатор /usr/bin/sh. В качестве значения shell должен быть указан существующий выполняемый файл.
-c	Любая текстовая строка. Обычно, это краткое описание регистрационного имени и используется сейчас для указания фамилии и имени реального пользователя. Эта информация хранится в записи пользователя в файле /etc/passwd.
-m	Создаёт начальный каталог нового пользователя, если он ещё не существует. Если каталог уже существует, добавляемый пользователь должен иметь права на доступ к указанному каталогу.
-k	Копирует содержимое каталога в начальный каталог нового пользователя, вместо содержимого стандартного «скелетного» каталога /etc/skel. Каталог должен существовать. Стандартный «скелетный» каталог содержит стандартные файлы, определяющие среду работы пользователя. Заданный администратором каталог может содержать аналогичные файлы и каталоги, созданные для определённой цели.
-f	Число дней, которые должны пройти после устаревания пароля, перед тем как учётная запись будет заблокирована. Вводить значение аргумента expire (представляющего собой дату) можно в любом формате (кроме Julian date). Например, можно ввести 10/6/99 или October 6, 1999.
-p	Шифрованное значение пароля, которое возвращает функция crypt (шифрование пароля). По умолчанию учётная запись заблокирована.

-a	Список типов или классов событий через запятую, образующих маску аудита (audit mask) для пользователя. Сразу после установки системы стандартной маски аудита для пользователя нет, но ее можно задать в файле /etc/default/useradd с помощью команды defadm. Эту опцию можно использовать, только если установлены утилиты аудита (Auditing Utilities).
<рег_имя>	Строка печатных символов, задающая регистрационное имя для нового пользователя. В имени пользователя допускается латиница нижнего и верхнего регистра, цифры, подчёркивание и дефис (короткое тире). Первым символом должна быть или буква или подчёркивание. Длина строки имени пользователя до 32-х символов.

7.4 Модификация уже имеющихся пользовательских записей

Для модификации уже имеющихся пользовательских записей применяется утилита usermod:

usermod -G audio,rpm,test1 test1

Такая команда изменит список групп, в которые входит пользователь test1 — теперь это audio, rpm, test1.

usermod -l test2 test1

Будет произведена смена имени пользователя с test1 на test2. Команды

usermod -L test2

и

usermod -U test2

соответственно временно блокируют возможность входа в систему пользователю (блокируют вход только по паролю) test2 и возвращают все на свои места.

Изменения вступят в силу только при следующем входе пользователя в систему.

При неинтерактивной смене или задании паролей для целой группы пользователей используйте утилиту «chpasswd». На стандартный вход ей следует подавать список, каждая строка которого будет выглядеть как:

<имя>:<пароль>

7.5 Удаление пользователей

Для удаления пользователей используйте userdel.

Команда

userdel test2

удалит пользователя test2 из системы. Если будет дополнительно задан параметр «-d», то будет уничтожен и домашний каталог пользователя. Нельзя удалить пользователя, если в данный момент он ещё работает в системе.

Утилиты «vigr» и «vipw» используются для ручного редактирования файлов /etc/passwd и /etc/group, в которых хранятся основные записи о пользователях и группах в системе.

Не рекомендуется создавать пользователей с правами сверх необходимых. Предпочтительнее создать серию новых групп и включить в них требуемого пользователя. А для данных групп установить соответствующие права на объектах файловой системы (утилиты «chmod» и «chown»).

7.6 Пароли пользователей

/etc/passwd — файл, содержащий в текстовом формате список пользовательских учётных записей (аккаунтов).

Является первым и основным источником информации о правах пользователя операционной системы.

Принцип:

Login : password : UID : GID : GECOS : home : shell

Каждая строка файла описывает одного пользователя и содержит семь полей, разделённых двоеточиями:

- регистрационное имя или логин;
- хеш пароля;
- идентификатор пользователя;
- идентификатор группы по умолчанию;
- информационное поле GECOS;
- начальный (он же домашний) каталог;
- регистрационная оболочка, или shell.

Основным назначением /etc/passwd является сопоставление логина и идентификатора пользователя (UID). Изначально поле пароля содержало хеш пароля и использовалось для аутентификации. Однако, в связи с ростом вычислительных мощностей процессоров появилась серьёзная угроза применения простого перебора для взлома пароля. Поэтому все пароли были перенесены в специальные файлы, такие как /etc/shadow. Эти файлы

недоступны для чтения обычным пользователям. Такой подход называется механизмом скрытых паролей.

Регистрационные имена должны быть уникальными и представлять собой строки не длиннее 32 символов (любые, кроме двоеточия и символа новой строки). По сути, имя пользователя — это его короткий и легко запоминаемый псевдоним, который используется при входе в систему и часто включается в адреса электронной почты.

Идентификатор пользователя — это число от 0 до 232-1. Пользователь с идентификатором «0» (обычно root) называется суперпользователем и имеет право на выполнение любых операций в системе. Принято соглашение о выделении «специальным» пользователям (bin, daemon), назначение которых — только запуск определённых программ, маленьких идентификаторов (меньше 100).

Пользователь может принадлежать к одной или нескольким группам, которые используются для задания прав более чем одного пользователя на тот или иной файл.

Список групп с их участниками задаётся в /etc/group. В файле же /etc/passwd указывается идентификатор группы по умолчанию.

Всем файлам, созданным пользователем после регистрации в системе, будет автоматически присвоен этот номер группы (исключение — если для каталога, в котором создаётся файл, установлен в правах бит SGID, то будет присвоена такая же группа, как у самого каталога).

/etc/group содержит записи обо всех группах в системе. Каждая его строка содержит:

- символьное имя группы;
- пароль группы — устаревшее поле, сейчас не используется. В нём обычно стоит «x»;
- идентификатор группы, или GID;
- список имён участников, разделённых запятыми.

Пример записи:

bin:x:1:root,bin,daemon

Здесь сообщается, что группа bin имеет GID=1, а входят в неё пользователи root, bin и daemon.

Поле GECOS хранит вспомогательную информацию о пользователе (номер телефона, адрес, полное имя и так далее).

- полное имя;
- адрес офиса или домашний адрес;
- рабочий телефон;
- домашний телефон.

С помощью утилиты «chfn» можно изменять эту информацию.

Пример строки с заполненным полем GECOS:

tester:x:210:8:Edward Chernenko,Marx Street 10,4554391,5454221:

/home/ed:/bin/bash

После входа в систему пользователь оказывается в своём домашнем каталоге. Исторически сложилось так, что домашний каталог пользователя root называется /root, а остальные имеют вид /home/.

Если на момент входа в систему домашний каталог отсутствует, то система выдаёт сообщение об ошибке и отказывается допустить пользователя к командной строке. Это можно изменить посредством установки параметра DEFAULT_HOME в файле /etc/login.defs в значение «no».

Следует отметить, что при использовании графического интерфейса пользователь не увидит предупреждения или сообщения об ошибке, но просто будет выведен из системы безо всяких объяснений (так как оконный менеджер не сможет выполнить запись в нужный каталог, такой как ~/.gnome).

В поле регистрационной оболочки задаётся shell, то есть интерпретатор командной строки. Здесь может быть указана любая программа, и пользователь может сам выбирать для себя наиболее подходящую при помощи команды «chsh». Тем не менее некоторые системы в целях безопасности требуют, чтобы суперпользователь явно разрешил использовать приложение в качестве интерпретатора командной строки. Для этого используется специальный файл /etc/shells, содержащий список «допустимых» оболочек.

Vipw — запускает текстовый редактор, указанный в переменной среды EDITOR (или редактор по умолчанию, mcedit), загружая в него копию файла /etc/passwd. После закрытия редактора переносит временную копию в сам файл. Не позволяет двум пользователям выполнять редактирование одновременно.

В файле /etc/shadow хранятся хеши паролей всех пользователей в системе. Процессы суперпользователя могут читать его напрямую, а для остальных создана специальная библиотека PAM. Она позволяет непривилегированным приложениям спрашивать у неё, правильный ли пароль ввёл пользователь, и получать ответ. Библиотека PAM, как правило, действует с привилегиями вызвавшего процесса. Таким образом, хеш не попадает «в чужие руки».

Пароль шифруется с MD5-хешированием или blowfish-хешированием (bcrypt), MD5-хеши всегда записываются после префикса «1».

Перед хешированием к паролю добавляются случайные символы — «salt» (соль, от англ. add salt to something — сделать что-либо более интересным; в русскоязычных источниках иногда используется термин «затравка»). Salt также приписывается к началу полученного хеша. Благодаря salt нельзя при простом просмотре файла обнаружить пользователей с одинаковыми паролями.

Кроме имени (первое поле каждой строки) и хеша (второе поле) в файле /etc/shadow также хранятся:

- дата последнего изменения пароля;
- через сколько дней можно будет поменять пароль;
- через сколько дней пароль устареет;
- за сколько дней до того, как пароль устареет, начать напоминать о необходимости смены пароля;
- через сколько дней после того, как пароль устареет, заблокировать учётную запись пользователя;
- дата, при достижении которой учётная запись блокируется;
- зарезервированное поле.

Даты обозначаются как число дней с 1 января 1970 года.

ОС поддерживает управление качеством используемых паролей. Рассмотрим настройку различных параметров используемых паролей.

Время действия пароля (по истечении указанного времени пользователь должен будет сменить пароль).

Необходимо в конфигурационном файле /etc/login.defs изменить параметр <PASS_MAX_DAYS>

Обратите внимание, что данное требование будет работать только для вновь создаваемых пользователей, для уже существующих нужно использовать команду:

chage –M <days> <users>

Если пользователю необходимо выдавать предупреждение за несколько дней до окончания срока действия пароля, необходимо использовать параметр <PASS_WARN_AGE>

Для изменения срока действия пароля необходимо использовать утилиту «chage». Синтаксис:

chage [-m mindays] [-M maxdays] [-d lastday] [-I inactive]

[-E expiredate] [-W warndays] user

Chage -l user

Chage изменяет количество дней между сменой пароля и датой последнего изменения пароля. Информация используется системой для определения времени, когда пользователь должен сменить свой пароль. Команда chage разрешена только для суперпользователя, за исключением использования ее с параметром «-l», который позволяет непривилегированным пользователям определить время, когда истекает их пароль или учётная запись.

С параметром «-m» изменяется значение на минимальное число дней между сменой пароля. Значение «0» в этом поле обозначает, что пользователь может изменять свой пароль когда угодно. С параметром «-M» изменяется значение на максимальное число дней, в течение которых пароль ещё действителен. Когда сумма и меньше, чем текущий день, у пользователя будет запрошен новый пароль до начала работы в системе. Эта операция может предваряться предупреждением (параметр «-W»).

С параметром «-d» изменяется значение на день, когда был изменён пароль последний раз (число дней с 1 января 1970). Дата также может быть указана в формате <ГГГГ-ММ-ДД> (или формат, используемый в вашем регионе).

Параметр «-E» используется для задания даты, с которой учётная запись пользователя станет недоступной. Параметр есть число дней с 1 января 1970. Пользователь, чья учётная запись была заблокирована, должен сообщить об этом администратору для дальнейшей работы в системе.

Параметр «-I» используется для задания количества дней «неактивности», то есть дней, когда пользователь вообще не входил в систему, после которых его учётная запись будет заблокирована. Пользователь, чья учётная запись была заблокирована, должен сообщить об этом администратору для дальнейшей работы в системе. Параметр есть количество «неактивных» дней. Значение «0» отключает этот режим.

Параметр «-W» используется для задания числа дней, с которых пользователю начнёт выводиться предупреждение об истечении срока действия его пароля и необходимости его изменения. Параметр есть число дней до истечения срока действия пароля, с которых пользователю будет выдаваться предупреждение.

Все вышеперечисленные значения хранятся в виде дней, если используется система теневых паролей, но если используется системы обычных паролей, то значения преобразуются в недели. Из-за этого преобразования могут происходить ошибки округления.

Если параметры не указаны, то chage работает в интерактивном режиме, сообщая пользователю текущие значения полей. Необходимо далее либо ввести новое значение поля, либо оставить его как есть. Текущее значение поля показывается в скобках [ ].

8. Настройка сети

Начиная с версии 209, в systemd имеется служба systemd-networkd, которую можно использовать для настройки сети. Кроме того, начиная с версии 213, разрешение имен DNS можно обрабатывать с помощью systemd-resolved вместо статического файла /etc/resolv.conf. Обе службы включены по умолчанию.

Файлы конфигурации для systemd-networkd (и systemd-resolved) можно разместить в /usr/lib/systemd/network или /etc/systemd/network. Файлы в /etc/systemd/network имеют более высокий приоритет, чем файлы в /usr/lib/systemd/network.

Существует три типа файлов конфигурации: файлы .link, .netdev и .network.

8.1. Именование сетевых устройств.

Udev обычно назначает имена интерфейсов сетевых карт на основе физических характеристик системы, таких как eth0. Если вы не уверены, как называется ваш интерфейс, вы всегда можете запустить ip link после загрузки вашей системы.

Для большинства систем существует только один сетевой интерфейс для каждого типа подключения. Например, классическое имя интерфейса для проводного подключения - eth0. Беспроводное соединение обычно имеет имя wifi0 или wlan0.

Если вы предпочитаете использовать классические или настраиваемые имена сетевых интерфейсов, есть три способа это сделать:

- замаскируйте файл .link udev для политики по умолчанию:

  ln -s /dev/null /etc/systemd/network/99-default.link

- создайте схему именования вручную, например, вроде internet0, dmz0 или lan0. Для этого создайте файлы .link в /etc/systemd/network/, которые выберут явное имя или лучшую схему именования для ваших сетевых интерфейсов. Например:

  cat > /etc/systemd/network/10-ether0.link << "EOF"

  [Match]

  # Change the MAC address as appropriate for your network device

  MACAddress=12:34:45:78:90:AB

  [Link]

  Name=ether0

  EOF

- в файле /boot/grub/grub.cfg передайте опцию net.ifnames=0 в командной строке ядра.

8.2 Настройка DHCP

Приведенная ниже команда создаёт файл базовой конфигурации для настройки IPv4 DHCP:

cat > /etc/systemd/network/10-eth-dhcp.network << "EOF"

[Match]

Name=<network-device-name>

[Network]

DHCP=ipv4

[DHCP]

UseDomains=true

EOF

8.3 Настройка статического IP-адреса

Приведенная ниже команда создает базовый файл конфигурации для настройки статического IP-адреса (с использованием как systemd-networkd, так и systemd-resolved):

cat > /etc/systemd/network/10-eth-static.network << "EOF"

[Match]

Name=<network-device-name>

[Network]

Address=192.168.0.2/24

Gateway=192.168.0.1

DNS=192.168.0.1

Domains=<Your Domain Name>

EOF

Если у вас несколько DNS-серверов, можно добавить несколько записей DNS. (Замените ключ DNS в команде, расположенной выше)

DNS=192.168.0.1 8.8.8.8 8.8.4.4

Не включайте записи DNS или доменов, если вы собираетесь использовать статический файл /etc/resolv.conf.

8.4 Создание файла /etc/resolv.conf

Если система будет подключена к сети Интернет, ей потребуются некоторые средства разрешения доменных имен (DNS) для преобразования доменных имён в IP-адреса и наоборот. Лучше всего это достигается помещением IP-адреса DNS-сервера, доступного у интернет-провайдера или сетевого администратора, в /etc/resolv.conf.

8.5 Настройка systemd-resolved

При использовании методов, несовместимых с systemd-resolved для настройки сетевых интерфейсов (например, ppp и т. д.), или при использовании любого типа локального преобразователя (например, bind, dnsmasq, unbound и т. д.) или любого другого программного обеспечения, которое генерирует /etc/resolv.conf (например, программа resolvconf, отличная от той, которая предоставляется systemd), службу systemd-resolved использовать не следует.

При использовании службы systemd-resolved для настройки DNS, будет создан файл /run/systemd/resolve/resolv.conf.

Создайте символическую ссылку, чтобы использовать этот файл в каталоге /etc

ln -sfv /run/systemd/resolve/resolv.conf /etc/resolv.conf

8.6 Статическая настройка файла resolv.conf

Если требуется статический файл /etc/resolv.conf, создайте его, выполнив следующую команду:

cat > /etc/resolv.conf << "EOF"
 # Begin /etc/resolv.conf
domain <Ваше доменное имя>
nameserver <IP-адрес вашего основного сервера имен>
nameserver <IP-адрес вашего вторичного сервера имен>
 # End /etc/resolv.conf
EOF

Domain — можно можно опустить или заменить на search

Замените <Ваше доменное имя> IP-адресом DNS-сервера. Может присутствовать несколько записей (требования предусматривают наличия вторичных серверов для возможности восстановления). Если вам нужен только один DNS-сервер, удалите вторую строку nameserver из файла. IP-адрес также может быть маршрутизатором в локальной сети.

8.7 Настройка /etc/hostname

Во время процесса загрузки файл /etc/hostname используется для определения имени хоста системы.

Создайте файл /etc/hostname и введите имя хоста, запустив:

echo «MyHostName» > /etc/hostname

«MyHostName» необходимо заменить именем, присвоенным компьютеру. Не вводите здесь полное доменное имя (FQDN). Эта информация помещается в файл /etc/hosts.

8.8 Настройка /etc/hosts

Выберите полное доменное имя (FQDN) и возможные псевдонимы для использования в файле /etc/hosts. Если вы используете статические IP-адреса, вам также необходимо выбрать IP-адрес. Синтаксис записи файла hosts:

IP_address myhost.example.org aliases

Если компьютер не должен быть видимым для Интернета (существует зарегистрированный домен и действительный блок назначенных IP-адресов — у большинства пользователей этого нет), убедитесь, что IP-адрес находится в диапазоне IP-адресов частной сети. Допустимые диапазоны:

Диапазон сети	Префикс
10.0.0.1 - 10.255.255.254	8
172.x.0.1 - 172.x.255.254	16
192.168.y.1 - 192.168.y.254	24

- **x** — может быть любым числом в диапазоне от 16 до 31
- **y** — может быть любым числом в диапазоне от 0 до 255

Создайте файл /etc/hosts

cat > /etc/hosts << "EOF"

 # Begin /etc/hosts

127.0.0.1 localhost.localdomain localhost

127.0.1.1 <FQDN> <HOSTNAME>

<192.168.0.2> <FQDN> <HOSTNAME> [alias1] [alias2] ...

::1 localhost ip6-localhost ip6-loopback

ff02::1 ip6-allnodes

ff02::2 ip6-allrouters

 # End /etc/hosts

EOF

Значения <192.168.0.2>, <FQDN> и <HOSTNAME> необходимо изменить в соответствии с потребностями (если IP-адрес назначен сетевым / системным администратором, и оборудование будет подключено к существующей сети).

Необязательные псевдонимы можно опустить, а строку 192.168.0.2 можно не указывать, если вы используете соединение, настроенное с помощью DHCP или автоконфигурации IPv6.

Запись ::1 является эквивалентом IPv6 127.0.0.1 и представляет loopback интерфейс IPv6.

127.0.1.1 — так называемый, «местный» от англ. local, или «локальный хост», по смыслу — это компьютер, зарезервированный специально для FQDN.

9. Служебные программы

9.1 Crontab

Crontab — таблицы, управляющие работой службы «cron». Файл содержит инструкции службы «cron» в общей форме: запускать указанную команду в заданное время и в заданные дни. На компьютере обычно имеются общесистемный файл (/etc/crontab), и индивидуальные файлы (/var/cron/tabs/<имя-пользователя>) для пользователей системы. Таким образом, команды в файле будут выполняться с правами этих пользователей или, в случае общесистемного файла, с правами пользователя, указанного в командной строке при запуске службы. Хотя «cron», по сути, является обыкновенным текстовым файлом, он не должен редактироваться обычными средствами. Для создания, изменения и удаления следует использоваться специальную утилиту, «crontab».

Пустые строки, ведущие пробелы и символы табуляции игнорируются. Строки, начинающиеся с символа «#», считаются комментариями и игнорируются. Заметьте, что комментарии не допускаются в тех же строках, где расположены команды «cron», так как они будут распознаны как части команды. По этой же причине комментарии не разрешены в строках, задающих переменные среды. Строка-директива представляет собой либо задание переменной среды, либо команду «cron». Можно определять среду (набор переменных среды), в которой будет выполняться команда. Задание переменной среды осуществляется в следующей форме:

<имя_переменной> = <значение>

где пробелы вокруг знака равенства («=») не обязательны, и любые пробелы после значения будут использованы как часть значения переменной <имя_переменной>. Строка <значение> может быть заключена в кавычки (одинарные или двойные) для возможности сохранения пробелов в начале и конце.

Несколько переменных среды устанавливаются автоматически службой «cron». SHELL устанавливается в /bin/sh, а LOGNAME и HOME определяются по файлу /etc/passwd (в соответствии с владельцем crontab).

Формат команд «cron» аналогичен стандарту V7 и является совместимым с ним. Существует две конфигурации «cron»: системная и пользовательская.

Общесистемная настройка «crontab» работает безусловно для всех пользователей. Пользовательская настройка является дополнительной и выполняется в сессиях пользователей.

Каждая стока в системной конфигурации «cron», расположенной в каталоге /etc, состоит из шести полей и команды:

<минута> <час> <число> <месяц> <день_недели> <пользователь>

<команда>

Каждая строка в пользовательской конфигурации «cron», расположенной в домашнем каталоге пользователя состоит из пяти полей и команды:

<минута> <час> <число> <месяц> <день_недели> <команда>

Поля отделяются друг от друга пробелами или символами табуляции. Команда может состоять из нескольких полей. Допустимые значения полей:

Поле	Допустимые значения
<минута>	* или 0-59
<час>	* или 0-23
<число>	* или 1-31
<месяц>	*, 1-12 или имя месяца (см. ниже)
<день_недели>	*, 0-7 или имя дня (воскресенье — это 0 и 7)
<пользователь>	имя существующего пользователя
<команда>	строка

Допустимо указание нескольких значений (и диапазонов через тире) через запятую. Примеры:

"1, 2, 5, 9" "0-4, 8-12"

Диапазон указывается как два числа, разделённых дефисом. Указываемые числа включаются в диапазон. Например, значение поля <час> 8-11 приведёт к выполнению команды в 8, 9, 10 и 11 часов.

При указании диапазона можно пропускать некоторые его значения, указав шаг в форме /<число>.

Например:

"0-23/2"

для поля <час> означает запуск команды через два часа (по стандарту V7 пришлось бы указывать "0,2,4,6,8,10,12,14,16,18,20,22").

Шаг можно указывать также после звёздочки — «каждые два часа» соответствует значению:

"*/2"

Звёздочка («*») без шага соответствует полному диапазону значений. Для задания полей <месяц> и <день_недели> можно использовать имена. Указывайте первые три буквы нужного дня или месяца на английском, регистр букв не имеет значения. Диапазоны или списки имён не разрешены.

Поле <команда> (остаток строки) определяет запускаемую по расписанию команду — вся оставшаяся часть строки до символа перевода строки или символа «%». Будет выполнен

вызов /bin/sh или другой оболочки, определённой в переменной SHELL в «crontab». Знак процента («%») в команде (если он не экранирован обратной косой чертой («\»)), будет соответствовать символу перевода строки и все данные после первого «%» будут посланы для команды на стандартный ввод.

Служба «cron» запускает команды, когда значения полей <минута>, <час>, <месяц> и хотя бы одно из полей <число> и <день_недели>, совпадают с текущим временем (см. замечание ниже). Служба «cron» сверяет директивы с текущим временем раз в минуту.

Примечание.

День выполнения команды может быть задан в двух полях — <число> и <день_недели>. Если оба поля определены (т.е. не равны *), то команда будет запущена, когда любое поле совпадёт с текущим временем.

Например, запись:

30 4 1,15 * 5

приведёт к выполнению команды в 4:30 по полуночи первого и пятнадцатого числа каждого месяца, плюс в каждую пятницу.

Вместо первых пяти полей допустимо указание одного из восьми специальных триггеров:

Строка	Значение
@reboot	Выполнить команду один раз, при запуске cron.
@yearly	Выполнять команду каждое 1 января, «0 0 1 1 *».
@annually	Эквивалентно @yearly.
@monthly	Выполнять команду в начале каждого месяца, «0 0 1 * *».
@weekly	Выполнять команду каждое воскресенье, «0 0 * * 0».
@daily	Выполнять команду в полночь, «0 0 * * *».
@midnight	Эквивалентно @daily.
@hourly	Выполнять команду раз в час, «0 * * * *».

9.2 Midnight Commander

Midnight Commander — это программа, предназначенная для просмотра содержимого каталогов и выполнения основных функций управления файлами в UNIX-подобных операционных системах.

Основные аргументы:

Опция	Значение опции
-b	Запуск программы в черно-белом режиме экрана.
-e	Запустить встроенный редактор. Если параметр файл задан, этот файл будет открыт при старте. Смотрите также mcedit(9.3).
-f	Выводит на экран определенный в процессе компиляции программы путь к файлам программы Midnight Commander.
-s	Включает медленный режим терминала, в котором программа выводит меньше псевдографических символов (в том числе в меню и экранах помощи) и отключается вывод дополнительных (избыточных) сообщений.
-v	Запустить встроенную программу просмотра Midnight Commander-а для просмотра указанного файла. После выхода из режима просмотра вы выходите из Midnight Commander и оказываетесь в shell.
-V	Отображает версию программы.

Главное окно программы Midnight Commander состоит из трех полей. Два поля, называемые "панелями", идентичны по структуре и обычно отображают перечни файлов и подкаталогов каких-то двух каталогов файловой структуры. Эти каталоги в общем случае различны, хотя, в частности, могут и совпасть. Каждая панель состоит из заголовка, списка файлов и информационной строки.

Третье поле экрана, расположенное в нижней части экрана, содержит командную строку текущей оболочки. В этом же поле (самая нижняя строка экрана) содержится подсказка по использованию функциональных клавиш F1 - F10. Самая верхняя строка экрана содержит строку горизонтального меню. Эта строка может не отображаться на экране; в этом случае доступ к ней можно получить, нажав клавишу F9.

Панели Midnight Commander обеспечивают просмотр одновременно двух каталогов. Одна из панелей является активной в том смысле, что пользователь может выполнять некоторые операции с отображаемыми в этой панели файлами и каталогами. В активной панели подсвечено имя одного из каталогов или файлов, а также выделен цветом заголовок панели в верхней строке. Этот заголовок совпадает с именем отображаемого в данной панели каталога, который является текущим каталогом той оболочки, из которой запущена программа. Вторая панель — пассивна. Почти все операции выполняются в активной панели, то есть в соответствующем (текущем) каталоге. Некоторые операции (типа копирования или переноса файлов) по умолчанию используют каталог, отображаемый в пассивной панели, как место назначения операции.

Вы можете выполнить любую команду операционной системы или запустить на исполнение любую программу непосредственно из программы Midnight Commander, просто набрав имя этой команды (программы) в командной строке и нажав клавишу Enter.

9.2.1 Панели каталогов Midnight Commander

В настоящем разделе перечисляются команды, которые позволяют оперировать с содержимым панелей. Если вы хотите узнать, как изменить вид панели или способ представления информации на панели, смотрите раздел Меню левой и правой панелей.

Сокращения:

C-f означает — нажмите Control и, удерживая ее, нажмите (коротким щелчком) клавишу f.

Аналогично M-<символ> означает, что надо удерживать в нажатом состоянии клавишу Meta или Alt во время удара по клавише <символ>. Если на клавиатуре нет клавиш Meta и Alt, нажмите ESC, отпустите ее, а потом щелкните по клавише <символ>.

S-<символ> означает, что нужно держать в нажатом состоянии клавишу Shift во время нажатия клавиши <символ>.

Tab, C-i	Сменить текущую (активную) панель. Подсветка перемещается с панели, которая была активной ранее, в другую панель, которая становится активной.
Insert, C-t.	Чтобы отметить файл, на который указывает в данный момент подсветка, используйте клавишу Insert (the kich1 terminfo sequence) или комбинацию C-t (Control-t). Для снятия отметки с файла используются те же комбинации.
M-g, M-r, M-j	Используются для перемещения подсветки, соответственно, на самый верхний, средний или нижний файл из числа отображаемых в данный момент на панели.
C-s, M-s	Инициирует режим поиска имен файлов в текущем каталоге по первым символам имени. После нажатия одной из этих комбинаций, вводимые символы отображаются не в командной строке, а в строке поиска.
M-t	Циклически переключает режимы отображения списка файлов текущего каталога. С помощью этой комбинации клавиш можно быстро переключаться из режима стандартного вывода (long listing) к сокращенному или к режиму, определяемому пользователем.
C-	Показать Справочник каталогов и перейти к выбранному каталогу.
+	Эта клавиша используется для того, чтобы выбрать (отметить) группу файлов по регулярному выражению, задающему эту группу.
****	Клавиша "\" снимает отметку с группы файлов, то есть производит действие, обратное тому, которое вызывается по клавише "+".
M-o	Сделать текущий каталог активной панели также текущим каталогом неактивной панели. Если необходимо, перевести неактивную панель в режим отображения списка файлов.
M-S-h, M-H	Отображает историю перемещения по каталогам
M-y, M-u	Перемещение к предыдущему и следующему каталогу из истории перемещения по каталогам

9.2.2 Командная строка оболочки

M-Enter	Копирует подсвеченное имя файла или каталога в командную строку.
M-Tab	Пытается выполнить операцию Завершение ввода имени файла, названия команды, переменной, имени пользователя или имени машины (в зависимости от того, что вы начали набирать и какой элемент команды вводите).
C-x t, C-x C-t	Копирует в командную строку имена помеченных файлов (или подсвеченное имя, если нет помеченных) из активной панели (C-x t) или пассивной панели (C-x C-T).
C-x p, C-x C-p	Первая комбинация клавиш копирует в командную строку имя текущего каталога, а вторая — имя каталога, отображаемого в пассивной панели.
C-q	Эта команда (the quote command) используется для того, чтобы вставить символы, которые каким-то образом интерпретируются самим Midnight Commander-ом (например, символ '+').
M-p, M-n	Эти комбинации используются для перемещения по истории команд. M-p вызывает перемещение на команду назад по списку ранее запускавшихся команд, а M-n — перемещение на одну команду вперед.
M-h	Выводит историю текущей строки ввода (для командной строки — историю команд).

9.2.3 Редактирование строк ввода

C-a	Перемещает курсор в начало строки
C-e	Перемещает курсор в конец строки.
C-b	Перемещает курсор на одну позицию влево.
C-f	Перемещает курсор на одну позицию вправо.
M-f	Перемещает курсор на одно слово вперед.
M-b	Перемещает курсор на одно слово назад.
C-h, backspace	Удаляет символ, предшествующий курсору.
C-d, Delete	Удаляет символ в позиции курсора.
C-@	Устанавливает метку для того, чтобы вырезать (скопировать в буфер) часть текста.
C-w	Копирует текст, расположенный между курсором и меткой, в буфер, удаляя текст из строки ввода.
M-w	Копирует текст, расположенный между курсором и меткой, в буфер.

C-y	Вставляет содержимое буфера в строку ввода перед позицией курсора.
C-k	Удаляет текст от курсора до конца строки.
M-p, M-n	Эти комбинации используются для перемещения по истории команд. M-p перемещает к предыдущей команде, M-n — к следующей.
M-C-h, M-Backspace	Удалить предшествующее слово.
M-Tab	Пытается выполнить завершение ввода (completion) имени файла, команды, переменной, имени пользователя или имени машины.

Строка главного меню появляется в верхней части экрана после нажатия клавиши F9. Меню состоит из пяти пунктов: «Левая», «Файл», «Команды», «Настройки» и «Правая» (в английской версии соответственно «Left», «File», «Command», «Options» и «Right»). При выборе одного из этих пунктов появляется соответствующее выпадающее меню.

Пункты меню «Левая» и «Правая» позволяют изменить вид, соответственно, левой и правой панелей, и характер отображаемой в панели информации, в частности, выполнить соединения с удаленными компьютерами.

Меню «Файл» позволяет выполнить какие-то действия с выбранным файлом или группой помеченных файлов.

Меню «Команды» перечисляет действия, которые имеют более общий характер и не относятся только к выделенному в данный момент файлу или группе помеченных файлов.

Меню «Настройки» служит для задания ряда параметров, определяющих внешний вид и поведение программы Midnight Commander. Один из пунктов этого меню служит для сохранения настроек, заданных пользователем.

9.2.4 Главные функциональные клавиши MC

Программа Midnight Commander использует функциональные клавишиF1 - F10 как «горячие» клавиши для команд, включенных в меню «Файл». Escape-последовательности, генерируемые клавишами F1-F10, соответствуют функциям terminfo kf1 - kf10. На терминалах без поддержки функциональных клавиш можно достичь аналогичного эффекта, нажав клавишу ESC, а затем число в диапазоне от 1 до 9 или 0 (соответствует F1 - F9 и F10).

Меню «Файл» содержит следующие команды (соответствующие "горячие" клавиши указываются в скобках):

Помощь (F1)

Вызывает встроенную программу просмотра гипертекстовой подсказки. При нажатии на клавишу F1 вы получите полный список управляющих комбинаций клавиш.

Меню пользователя (F2)

Вызывает меню пользователя. Меню пользователя предоставляет простой способ расширения возможностей Midnight Commander за счет добавления в личное меню пользователя вызова часто используемых программ.

Просмотр файла (F3)

Просмотреть файл, на который указывает подсветка.

Просмотр вывода команды (Filtered View) (M-!)

По этой команде на экране появляется строка ввода, в которой вы можете ввести любую команду с параметрами (по умолчанию предлагается использовать в качестве параметра имя подсвеченного файла). Вывод этой команды будет отображаться на экране через встроенную программу просмотра.

Редактирование (F4)

Вызывается редактор, указанный в переменной окружения EDITOR

Копирование (F5)

Вызывается диалоговое окно, в котором предлагается скопировать подсвеченный файл из каталога, отображаемого в активной панели в каталог, отображаемый в пассивной панели. Имя каталога, в который будет производиться копирование, можно изменить.

В процессе выполнения копирования можно нажать C-c или ESC для того, чтобы прервать выполнение операции.

- Права доступа (C-x c). Позволяет изменить права доступа к выделенному или помеченным файлам.
- Жесткая ссылка (C-x l). Создает жесткую ссылку на текущий файл.
- Символич. ссылка (C-x s). Создает символическую ссылку на текущий файл. Символическая ссылка — это ссылка на имя исходного файла. Символическую ссылку легко отличить от первоначального имени файла и программа Midnight Commander указывает символические ссылки выводя знак "@" перед именем такой ссылки (кроме ссылок на каталоги, которые обозначаются знаком тильды "~"). Если на экран выводится строка мини-статуса (опция "Показывать мини-статус" ("Show mini-status") включена), то в ней отображается имя исходного файла. Используйте символические ссылки в тех случаях, когда хотите избежать путаницы, связанной с применением жестких ссылок.
- Владелец/группа (C-x o). Позволяет выполнить команду chown.

Переименование (F6)

Вызывается диалоговое окно, в котором предлагается перенести подсвеченный файл из каталога, отображаемого в активной панели (или группу отмеченных файлов, если в активной панели отмечен хотя бы один файл) в каталог, отображаемый в пассивной панели. Имя каталога, в который будет производиться перенос, можно изменить, воспользовавшись соответствующей строкой ввода.

В остальном диалоговое окно аналогично окну, появляющемуся при вызове команды копировании файлов (смотрите выше).

Создание каталога (F7)

Появляется диалоговое окно и создается каталог с введенным именем.

Удаление (F8)

Удаляется файл, имя которого подсвечено (или группа файлов, имена которых помечены) в активной панели. Операцию можно прервать, нажав C-c или ESC во время ее исполнения.

- Быстрая смена каталога (Quick cd) (M-c).

Используйте быструю смену каталога если вы знаете полный путь к каталогу, в который хотите перейти (который хотите сделать текущим).

Выход (F10)

Выйти из программы Midnight Commander.

9.3 Полноэкранный редактор mcedit

В дистрибутиве SMART Linux используется редактор mcedit редактор можно запустить из командной строки. Для этого выполните одну из команд:

mcedit

mcedit <filename>

Чтобы вернуть действие (undo) нажмите комбинацию **CTRL+U**, чтобы повторить (redo) — **ALT+R**.

Справку можно получить, нажав клавишу **F1**. После окончания работы с программой нажмите **F2** — сохранить и **F10** — выйти.

Основные аргументы:

-V, --version	Вывод текущей версии
-f, --datadir	Вывод директории mcedit
-F, --datadir-info	Расширенная информация о директории, пути к конфигам, скриптам и всему что использует mcedit
-b	Переключиться на черно-белый вариант редактора
-V	Вывести версию программы

9.3.1 Задание макросов

Чтобы определить макрос, нажмите Ctrl-R, а затем введите что вы хотите выполнить. Когда закончите, снова нажмите Ctrl-R. Далее нужно будет выбрать клавишу, на которую будет назначен новый макрос. После чего макрос будет выполняться при нажатии назначенной клавиши.

Макросы могут быть найдены в файле

~/.local/share/mc/mc.macros

Пример содержания файла:

[editor]

\=InsertChar:113;InsertChar:113;InsertChar:113;

Символ «\» это клавиша заданная для выполнения макроса. После «=» Идет сам макрос, в данном случае вставить три символа.

Внешние макросы определены в файле ~/.local/share/mc/mcedit/macros.d/ и должны называться macro.XXXX.sh где XXXX это номер от 0 до 9999

Сокращения, которые можно использовать во внешних макросах:

%c	Номер позиции столбца курсора.
%i	Отступ пустого пространства, равный позиции курсора в столбце.
%y	Тип синтаксиса текущего файла.
%b	Имя блочного файла.
%f	Имя текущего файла.
%n	Только имя текущего файла без расширения.
%x	Расширение текущего файла.
%d	Имя текущего каталога.
%F	Текущий файл в неактивной панели.
%D	Имя каталога неактивной панели.
%t	Все файлы с тегами, но без учёта выбранных в данный момент.
%T	Все файлы с тегами.
%u	Не помеченные файлы без учёта выбранных.
%U	Не помеченные файлы.
%s	Выбранные файлы без учёта непомеченных.
%S	Выбранные файлы.

Пример внешнего макроса:

```
cat macro.XX.sh
e       edit file
mcedit `cat %b`
```

в файле ~/.local/share/mc/mc.macros добавляем клавишу для макроса

```
[editor]
ctrl-W=ExecuteScript:XX;
```

где XX это номер в названии файла внешнего макроса.

9.3.2 Расширенная настройка mcedit

Расширенную информацию по редактору можно найти в файлах:

- **/usr/share/mc/examples/macros.d** — примеры макросов;
- **/usr/share/mc/help/mc.hlp.ru** — расширенная документация mcedit;
- **/usr/share/mc/hints/mc.hint.ru** — Советы по использованию;
- **~/.config/mc/ini** — файл с конфигурацией mcedit для конкретного пользователя;
- **/usr/share/mc/mc.lib** — файл с глобальными настройками mcedit для всех пользователей;
- **/usr/share/mc/syntax/*** — файлы синтаксиса mcedit.

stech.ru

info@stech.ru

+7 495 777 54 43

г. Москва, Киевское ш., 22-й км, домовладение 6, стр. 1, БЦ «Комсити»