Операционная система «РЕД ОС»
Руководство администратора

RU.29926343.02.01-01 32 1-1

HTTPS://REDOS.RED-SOFT.RU/

Данный документ является руководством администратора операционной системы «РЕД ОС» версии 7.3 (далее РЕД ОС, ОС) 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 2020

Оглавление

Общие сведения о РЕД ОС

1 Общие сведения о РЕД ОС 10

1.1 Описание и область применения операционной системы 10

1.2 Основные функции РЕД ОС 10

1.3 Состав РЕД ОС 12

1.4 Документация в составе 13

1.5 Требования к персоналу (администратору) 13

1.5.1 Общие положения 13

1.5.2 Обязанности пользователей 14

1.6 Приёмка поставленного средства 15

1.7 Требования к среде исполнения 16

2 Разновидности файловых систем 17

2.1 Поддерживаемые файловые системы 17

2.1.1 Утилиты для работы с файловыми системами 17

3 Общие принципы работы РЕД ОС 20

3.1 Процессы и файлы 20

3.1.1 Процессы функционирования РЕД ОС 21

3.1.2 Файловая система РЕД ОС 21

3.1.3 Организация файловой структуры 21

3.1.4 Иерархическая организация файловой системы . . . 22
3.1.5 Имена дисков и разделов . . . 23
3.2 Работа с наиболее часто используемыми компонентами . . . 24
3.2.1 Командные оболочки (интерпретаторы) . . . 24
3.3 Использование многозадачности . . . 27
3.4 Режимы работы ОС . . . 28
3.4.1 Диагностические режимы работы . . . 28
3.4.2 Режимы отображения информации . . . 30

Установка РЕД ОС

4 Установка РЕД ОС . . . 33
4.1 Начало установки . . . 33
4.2 Альтернативные способы установки . . . 35
4.2.1 Источники установки . . . 36
4.2.2 Запуск сетевой установки . . . 36
4.3 Последовательность установки . . . 39
4.4 Язык . . . 40
4.5 Обзор установки . . . 40
4.6 Задание пароля администратора системы . . . 47
4.7 Установка системы . . . 49
4.8 Соглашение пользователя и Лицензионный договор . . . 50
4.9 Завершение установки . . . 51
4.10 Подготовительные процедуры . . . 51

Управление программным обеспечением

5 Управление ПО . . . 54
5.1 Введение: пакеты, зависимости и репозитории . . . 54
5.2 Основы работы с RPM . . . 55
5.3 Назначение DNF . . . 57
5.4 Источники программ (репозитории) . . . 58
5.5 Поиск пакетов . . . 59
5.6 Установка или обновление пакета . . . 59
5.7 Удаление установленного пакета . . . 60
5.8 Обновление всех установленных пакетов . . . 60
5.9 Puppet . . . 60

Система безопасности РЕД ОС

6 Система безопасности РЕД ОС 65

6.1 Программа sudo 65

6.2 IP-фильтр iptables: архитектура и синтаксис 65

6.2.1 Устройство фильтра iptables 66

6.2.2 Встроенные таблицы фильтра iptables 66

6.2.3 Команды утилиты iptables 67

6.2.4 Ключи утилиты iptables 70

6.2.5 Основные действия над пакетами в фильтре iptables 70

6.2.6 Основные критерии пакетов в фильтре iptables 72

6.2.7 Использование фильтра iptables 74

6.3 Аудит в РЕД ОС 74

6.3.1 Файлы и утилиты аудита РЕД ОС 74

6.4 Права доступа к файлам и каталогам 96

6.4.1 chmod 99

6.4.2 umask 102

6.4.3 chown 102

6.4.4 ACL 103

6.4.5 Chattr и lsattr 105

6.5 Systemd – управление компонентами ОС 107

6.6 SELinux 111

6.6.1 Введение 111

6.6.2 Установка 112

6.6.3 Утилиты 112

6.7 PAM 132

6.8 Rsyslog 140

6.9 Afick - верификация целостности 144

6.10 AMTU - утилита тестирования абстрактной машины 148

6.11 ntpdate 149

6.12 Отказоустойчивый кластер 152

6.13 Изменение приоритета процесса 156

6.14 Управление дисковыми квотами 158

6.15 Ограничение ресурсов пользователя 163

6.16 Контроль целостности запускаемых компонентов 164

6.17 Блокирование файлов 166

6.18 Очистка памяти 166

6.18.1 shred 167

6.18.2 wipe 167

6.18.3 Secure-Delete 169

6.19 Экспорт данных пользователя ... 171
6.20 Резервирование данных ... 173
6.21 Лимиты ресурсов ... 173
6.22 Монтирование файловых систем ... 174
6.23 Принудительное завершение сеанса пользователя ... 176

Управление пользователями

7 Управление пользователями ... 178
7.1 Общая информация ... 178
7.2 Утилита passwd ... 179
7.3 Добавления нового пользователя ... 179
7.4 Модификация пользовательских записей ... 183
7.5 Удаление пользователей ... 183
7.6 Пароли пользователей ... 184
7.7 Роли пользователей ... 188

Сеть и служебные программы

8 Настройка сети ... 197

9 Служебные программы ... 202
9.1 Служба xinetd ... 202
9.1.1 Параметры ... 203
9.1.2 Управление xinetd ... 204
9.1.3 Файлы ... 205
9.2 Crontab ... 205
9.3 Полноэкранный редактор vi ... 208
9.3.1 Режимы работы редактора ... 208
9.3.2 Ввод текста ... 208
9.3.3 Команды ... 208
9.3.4 Перемещение курсора ... 209
9.3.5 Редактирование ... 209
9.3.6 Командная строка ... 209
9.3.7 Блоки, буферы, окна редактирования. Повторители ... 210
9.3.8 Открыть/создать файл ... 211
9.3.9 Дополнительные опции ... 212
9.4 Редактор VIM ... 212
9.4.1 Режимы работы ... 212
9.4.2 Основные возможности ... 213
9.4.3 Конфигурация ... 215

9.5 Назначение пароля на загрузчик GRUB2 216

Средства виртуализации и контейнеризации

10 Средство виртуализации 219

10.1 Общие сведения 219

10.2 Средства управления средой виртуализации 219

10.2.1 Утилита qemu-img 221
10.2.2 Утилита virt-install 222
10.2.3 Утилита virsh 224
10.2.4 Менеджер виртуальных машин virt-manager 226

10.3 Установка среды виртуализации 227

10.3.1 Установка в процессе развертывания новой системы РЕД ОС 227
10.3.2 Установка в существующей системе РЕД ОС 229

10.4 Ролевой доступ 229

10.4.1 Создание новых ролей 230
10.4.2 Объекты и разрешения 231
10.4.3 Создание пользователей 234
10.4.4 Назначение и отзыв ролей 235

10.5 Подключение к гипервизору 235

10.5.1 Подключение к гипервизору через SSH 236
10.5.2 Подключение к гипервизору с помощью virsh 236
10.5.3 Подключение к гипервизору с помощью менеджера ВМ 238

10.6 Создание виртуальных машин 238

10.6.1 Создание ВМ с помощью утилиты virt-install 238
10.6.2 Создание ВМ с помощью менеджера virt-manager 240
10.6.3 Создание ВМ через файл конфигурации 244

10.7 Управление ВМ 247

10.7.1 Управление ВМ с помощью утилиты virsh 247
10.7.2 Управление ВМ с помощью менеджера virt-manager 255

10.8 Безопасность в среде виртуализации 269

10.8.1 Доверенная загрузка ВМ 269
10.8.2 Контроль целостности ВМ 270
10.8.3 Ограничение программной среды 270
10.8.4 Резервное копирование 270
10.8.5 Защита памяти 272
10.8.6 Управление потоками информации 273
10.8.7 Регистрация событий безопасности 278

11 Средство контейнеризации 280

11.1 Общие сведения 280

11.2 Установка среды контейнеризации 281
11.2.1 Установка в процессе развертывания новой системы РЕД ОС 281
11.2.2 Установка в существующей системе РЕД ОС 282
11.3 Доступ к среде контейнеризации 283
11.4 Компоненты средства контейнеризации 283
11.4.1 Dockerfile 283
11.4.2 Docker Registry 289
11.4.3 Docker Daemon 289
11.4.4 Docker Client 290
11.4.5 Объекты Docker 291
11.4.6 Docker volume 291
11.5 Управление контейнерами 295
11.5.1 Docker Container 295
11.5.2 Docker build 296
11.5.3 Docker image 296
11.5.4 Docker images 297
11.5.5 Docker create 298
11.5.6 Docker run 299
11.5.7 Docker start 301
11.5.8 Docker stop 301
11.5.9 Docker pause 301
11.5.10 Docker login 302
11.5.11 Docker attach 302
11.5.12 Docker exec 303
11.5.13 Docker logs 304
11.5.14 Docker stats 304
11.5.15 Docker tag 305
11.6 Безопасность в среде контейнеризации 306
11.6.1 Изоляция контейнеров 306
11.6.2 Проверка корректности конфигурации контейнеров 308
11.6.3 Контроль целостности контейнеров и их образов 310
11.6.4 Регистрация событий безопасности 312

Общие сведения о РЕД ОС

1. Общие сведения о РЕД ОС

1.1 Описание и область применения операционной системы

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

РЕД ОС на уровне драйверов поддерживает широкий перечень оборудования актуальных версий, доступного на рынке СВТ, а так же оборудования снятого с производства, но поддерживаемого производителями.

В РЕД ОС поддерживается инсталляция с оптических носителей информации, флеш-накопителей, разделов локального жёсткого диска, а так же установка по сети передачи данных.

1.2 Основные функции РЕД ОС

РЕД ОС 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. РЕД ОС предоставляет адекватные механизмы для разграничения пользователей и защиты их данных. Использование привилегированных команд ограничено и доступно только административным пользователям.

РЕД ОС 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, РЕД ОС поддерживает типы, которые могут быть связаны с объектами, и домены,

которые могут быть связаны с процессами. Роли определяются доменами, к которым они имеют доступ. Предопределённый файл политики, который является частью конфигурации РЕД ОС, определяет правила между доменами и типами. С вышеизложенным определением ролей и прав доступа, подразумеваемых индивидуальными ролями, РЕД ОС выполняет требования ролевого доступа.

РЕД ОС предназначена для работы в сетевом окружении с другими экземплярами РЕД ОС, 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.

РЕД ОС 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 РЕД ОС, может делегироваться пользователям РЕД ОС. Все данные находятся под управлением механизмов безопасности РЕД ОС. Данные сохраняются в поименованных объектах, и РЕД ОС 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. РЕД ОС подтверждает подлинность предъявленного идентификатора пользователя до того, как разрешать ему выполнять дальнейшие действия. РЕД ОС 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 РЕД ОС.

РЕД ОС 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.

РЕД ОС имеет существенные расширения элементов безопасности по сравнению со стандартными системами UNIX:

- списки управления доступом;
- реализацию доменов и типов;
- журналируемая файловая система (ext4);
- подключаемые модули аутентификации (PAM);
- специализированная система аудита, которая позволяет учитывать критичные события безопасности и предоставляет административному пользо-

вателю инструментальные средства конфигурирования системы аудита и оценки записей аудита;
- базовые функции проверки комплекта оборудования позволяют по требованию административного пользователя проверять, правильно ли обеспечиваются основные функции безопасности аппаратных средств, на которые полагается ОО.

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 РЕД ОС

РЕД ОС 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 РЕД ОС можно выделить следующие функциональные элементы:
- ядро ОС;
- системные библиотеки;
- встроенные средства защиты информации (КСЗ);
- системные приложения;
- программные серверы;
- прочие серверные программы;
- интерактивные рабочие среды и командные интерпретаторы;
- прочие системные приложения.

Комплекс встроенных средств защиты информации, является принадлежностью операционной среды РЕД ОС 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).

Интерактивные рабочие среды (ИРС) – программы (пакеты программ), предназначенные для работы пользователя в РЕД ОС и предоставляющие ему удобный интерфейс для общения с ней. Командные рабочие среды включают в свой состав командные интерпретаторы.

Командные интерпретаторы – специальные программы (терминалы), предназначенные для выполнения различных команд, подаваемых пользователем при работе с РЕД ОС.

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

В состав РЕД ОС 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 РЕД ОС представлена следующая документация:

- Электронная, контекстно-зависимая справочная система;
- Электронные справочники (man).

1.5 Требования к персоналу (администратору)

1.5.1 Общие положения

Администратор РЕД ОС должен иметь:

- базовые навыки администрирования ОС семейства Linux;
- навыки конфигурирования и настройки программных продуктов и ОС;
- опыт работы со стандартными элементами графического интерфейса приложений;
- навыки поддержания в работоспособном состоянии технических средств ПК.

1.5.2 Обязанности пользователей, определяемые предположениями безопасности

Предопределённое использование РЕД ОС

Доступ администраторов к РЕД ОС 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 РЕД ОС администраторы обязаны предотвращать и выявлять установку и запуск встроенных программ отладки.

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

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

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

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

Порядок обеспечения среды функционирования РЕД ОС

Администраторы должны использовать функции, предоставляемые РЕД ОС, 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 РЕД ОС.

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

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

Предполагается наличие (одного или более) компетентных лиц (администраторов), которые назначаются для управления безопасностью РЕД ОС и информации в нем. Они несут ответственность за следующие функции:

- создание и сопровождение ролей;
- установление и сопровождение отношений между ролями;
- назначение и аннулирование ролей, назначаемых пользователям.

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 РЕД ОС полномочиями и ролями, имеют права создавать новые объекты данных, владельцами которых они становятся.

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

Уполномоченные пользователи обладают необходимым разрешением на доступ в РЕД ОС, по крайней мере, к части информации, управляемой РЕД ОС, 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 РЕД ОС.

1.6 Приёмка поставленного средства

Для контроля качества и приёмки РЕД ОС силами потребителя производятся испытания полученного изделия. При этом производится проверка контрольных сумм дистрибутива РЕД ОС. 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 РЕД ОС, 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:

- РЕД ОС. Дистрибутив,
- РЕД ОС. Дополнительный диск,
- РЕД ОС. Формуляр,
- Копия сертификата ФСТЭК России,

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 РЕД ОС на физических носителях должны использоваться цифровые многоцелевые диски (DVD), не имеющие видимых механических повреждений, отрицательно влияющих на процессы воспроизведения информации. Носители РЕД ОС считаются прошедшими проверку, если установлено, что они соответствуют указанным требованиям.

В состав документации РЕД ОС должны входить документы:

- Операционная система «РЕД ОС» Формуляр. RU.29926343.02.01-01 30;
- Операционная система «РЕД ОС» Руководство администратора. RU.29926343.02.01-01 32 1-1;
- Операционная система «РЕД ОС» Руководство пользователя. RU.29926343.02.01-01 34 1-1.

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

Цифровые многоцелевые диски (DVD) с размещённой на них РЕД ОС должны быть промаркированы и упакованы. Маркировка цифровых многоцелевых дисков с размещённой на них РЕД ОС должна наноситься на внешнюю (нерабочую) поверхность диска и обязательно содержать наименование изделия и серийный номер, позволяющие однозначно идентифицировать РЕД ОС. Сертифицированные образцы должны быть маркированы специальным защитным знаком Системы сертификации СЗИ по требованиям безопасности информации № РОСС RU. 0001.01БИ00, наносимым в разделе 8 Формуляра RU.29926343.02.01-01 30. Для проверки маркировки и упаковки РЕД ОС необходимо удостовериться в их соответствии указанным требованиям.

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

РЕД ОС считается окончательно принятой, если она соответствует требованиям и успешно прошла испытания.

1.7 Требования к среде исполнения

В среде функционирования изделия должны использоваться средства доверенной загрузки, сертифицированные по 4 уровню доверия или выше в системе сертификации средств защиты информации по требованиям безопасности информации N РОСС RU.0001.01БИ00.

Установка и настройка данных средств доверенной загрузки должна выполняться в соответствии с формуляром и руководствами на эти средства.

2. Разновидности файловых систем

2.1 Поддерживаемые файловые системы

В дистрибутиве поддерживаются следующие файловые системы (ФС):

- журналируемая файловая система ext3;
- журналируемая файловая система ext4;
- файловая система ISO 9660 для накопителей для оптических магнитных дисков.

2.1.1 Утилиты для работы с файловыми системами

Общее назначение утилит приведено в таблице.

Утилита	Назначение
`mkfs`	создание новой файловой системы (make filesystem);
`fsck`	проверка файловой системы на ошибки (filesystem check);
`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;

Утилита	Назначение
`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.

Для различения файловых систем используется указание типа файловой системы после параметра -t или в качестве компонента имени утилиты, например:

```
mkfs -t ext2 /dev/sda1
fsck.ext2 /dev/sda2
```

Для преобразования файловой системы из ext2 в ext3 необходимо выполнить команду:

```
tune2fs -j /dev/sd<X>
```

Для обратного преобразования необходимо смонтировать этот раздел как ext2.

Для того, чтобы сохранить копию диска (например, CD-ROM), необходимо сделать следующее:

- убедиться в наличии в текущем каталоге достаточного количества свободного места;
- выполнить команду:

```
dd if=/dev/cdrom of=cdrom.iso bs=1M
```

- после этого можно просмотреть содержимое файла cdrom.iso, смонтировав его, например, так:

```
mount -o loop cdrom.iso /mnt/cdrom
```

В качестве исходного устройства для копирования также может выступать любое дисковое устройство, например, дискета или жёсткий диск. Кроме того, получившийся образ CD-ROM можно записать на матрицу CD-R/RW с использованием команды **cdrecord**, т.к. файл cdrom.iso является полным образом диска.

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

Системный вызов	Назначение
`mount`	монтирование файловых систем;

Системный вызов	Назначение
`umount`	размонтирование файловых систем;
`find`	поиск файлов в директориях;
`locate`	поиск файлов по определённому образцу имени;
`which`	поиск файла, который будет запущен при выполнении данной команды;
`cd`	смена текущего каталога/директории;
`pwd`	показать текущий каталог/директорию;
`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;
`more`	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 ext (append-only, immutable, safe deletion, no atime modified, no backup,...);
`lsattr`	просмотр атрибутов файлов для файловой системы ext.

3. Общие принципы работы РЕД ОС

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

Диалог с ОС осуществляется посредством командных интерпретаторов с системных библиотек. Каждая системная библиотека представляет собой набор программ, динамически вызываемых операционной системой.

Для удобства пользователей при работе с командными интерпретаторами используются интерактивные рабочие среды (ИРС), предоставляющие пользователю удобный интерфейс для работы с ОС.

В самом центре ОС РЕД ОС 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, а ИРС от командного интерпретатора.

3.1 Процессы и файлы

РЕД ОС 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 РЕД ОС

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 РЕД ОС, 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 РЕД ОС

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.

Например, файловая система CD-ROM в РЕД ОС монтируется по умолчанию в каталог /media/cdrom (путь в РЕД ОС обозначается с использованием «/», а не «\», как в DOS/Windows). Текущий каталог обозначается «./».

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.

Маршрут (путь) - это последовательность имён каталогов, представляющий собой путь в файловой системе к данному файлу, где каждое следующее имя отделяется от предыдущего наклонной чертой (слэшем). 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 каталога
`/boot`	место, где хранятся файлы, необходимые для загрузки ядра системы;
`/lib`	здесь располагаются файлы динамических библиотек, необходимых для работы большей части приложений, и подгружаемые модули ядра;
`/bin`	минимальный набор программ, необходимых для работы в системе;
`/sbin`	набор программ для административной работы с системой (программы, необходимые только суперпользователю);
`/home`	здесь располагаются домашние каталоги пользователей;
`/etc`	в данном каталоге обычно хранятся общесистемные конфигурационные файлы для большинства программ в системе;
`/etc/rc?.d`, `/etc/init.d`, `/etc/rc.boot`, `/etc/rc.d`	директории, где расположены командные файлы, выполняемые при запуске системы или при смене ее режима работы;
`/etc/passwd`	база данных пользователей, в которой содержится информация об имени пользователя, его настоящем имени, личном каталоге, зашифрованный пароль 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 udev;

Каталог	Описание каталога
/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. Отдельно устанавливаемые пакеты программ и другие файлы размещаются в каталоге /usr/local. Некоторые подкаталоги системы /usr рассмотрены ниже;
/usr/bin	практически все команды, хотя некоторые находятся в /bin или в /usr/local/bin;
/usr/sbin	команды, используемые при администрировании системы и не предназначенные для размещения в файловой системе root;
/usr/local	здесь рекомендуется размещать файлы, установленные без использования пакетных менеджеров, внутренняя организация каталогов практически такая же, как и у корневого каталога;
/usr/man	каталог, где хранятся файлы справочного руководства man;
/usr/share	каталог для размещения общедоступных файлов большей части приложений.

Содержимое каталога «/var» представлено в таблице.

Каталог	Описание каталога
/var/log	место, где хранятся файлы аудита 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.

3.1.5 Имена дисков и разделов

Все физические устройства компьютера отображаются в каталоге /dev файловой системы РЕД ОС. Диски (в том числе IDE/SATA/SCSI жесткие диски, USB-диски) имеют имена:

> /dev/sda - первый диск;
> /dev/sdb - второй диск и т.д.

Диски обозначаются /dev/sdX, где X - a,b,c,d,e,... в зависимости от порядкового номера диска на шине.

Раздел диска обозначается числом после его имени.

Например, /dev/sdb4 - четвёртый раздел второго диска.

3.2 Работа с наиболее часто используемыми компонентами

3.2.1 Командные оболочки (интерпретаторы)

Как было сказано выше, для управления ОС используются командные интерпретаторы (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 РЕД ОС используется командная оболочка Bash (Bourne Again Shell). Она ведёт историю команд и предоставляет возможность их редактирования.

Чтобы проверить, какую оболочку Вы используете, наберите команду:

```
echo $SHELL
```

У каждой командной оболочки свой синтаксис команд. В РЕД ОС рекомендуется использовать командную оболочку Bash. Дальнейшее описание и примеры будут приведены с использованием данной командной оболочки.

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

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:

```
bu buildhash builtin bunzip2
```

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:

```
./prog
```

Базовые команды оболочки Bash

Все команды, приведённые ниже, могут быть запущены в режиме консоли.

Для получения более подробной информации используйте команду «man». Например:

```
man ls
```

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 команды
`cd ..`	позволяет сделать текущим родительский каталог;
`cd -`	позволяет вернуться в предыдущий каталог;
`cd <без_параметров>`	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>
```

Крайне полезно просмотреть файл, перед тем как его редактировать. Для выхода 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. Колонка команд указывает имя процесса, колонка PID (идентификатор процесса) - номер процесса (этот номер используется, для операций с процессом, например чтобы «убить» его командой «kill»).

```
ps <аргументы>
```

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_номер>
```

Иногда необходимо будет использовать «kill -9 <PID_number>» (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>
```

3.3 Использование многозадачности

РЕД ОС - это многозадачная система. Продемонстрируем на двух примерах, как это можно использовать.

Первый пример - запуск программы в фоновом режиме. Для того чтобы это сделать, вам нужно набрать «&» после имени программы. После этого оболочка даёт возможность запускать другие приложения. Пользователь должен быть внимательным, так как некоторые программы интерактивны, и их запуск в фоновом режиме не имеет смысла (подобные программы просто остановятся, будучи запущенными в фоновом режиме). Для того, чтобы вернуть их в обычный режим, наберите:

```
fg <имя_программы>
```

Второй метод представляет собой запуск нескольких независимых сеансов. В консоли, нажмите «Ctrl + Alt» и одну из клавиш, находящихся в интервале от «F1» до «F6». На экране появится новое приглашение системы, и вы сможете открыть новый сеанс. Этот метод также позволяет вам работать на другой консоли, если консоль, которую вы использовали до этого, не отвечает, или вам необходимо остановить зависшую программу.

3.4 Режимы работы ОС

3.4.1 Диагностические режимы работы

С точки зрения функционирования ОС можно выделить 3 режима: нормальный(штатный), аварийный и режим восстановления.

Обычно ОС нормально функционирует и выполняет возложенные на неё функции в нормальном режиме. В этом режиме пользователь получает ожидаемый отклик на свои действия от ОС (в этом разделе нормальный режим рассмотрен не будет, ему посвящены остальные разделы руководства). Однако в ряде случаев, если в работе системы возникают проблемы, ОС может выполнить загрузку в режиме восстановления или аварийном режиме с целью диагностики и исправления проблем.

Режим восстановления

Режим восстановления позволяет загрузить минимальное окружение ОС с дистрибутивного носителя вместо загрузки с жёсткого диска. Этот режим предусмотрен для восстановления в случае сбоя. В штатном режиме ОС использует файлы на жёстком диске компьютера для запуска программ, хранения информации и прочих операций.

Однако не исключены ситуации, когда не получается полностью запустить ОС, чтобы иметь возможность обращения к файлам на жёстком диске. В режиме восстановления можно получить доступ к файлам, даже если не удалось запустить ОС с этого диска.

Если у вас нет возможности загрузиться с дистрибутивного носителя, то перейти в режим восстановления можно, выполнив следующие действия:

1. Начните загрузку ПК и дождитесь появления меню GRUB. Нажмите клавишу «E» для редактирования параметров загрузки.
2. Добавьте в конец строки, начинающейся с «linux /boot/vmlinuz ...», следующую запись:

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

Используйте сочетание клавиш «Ctrl+A» (для перехода в начало строки) и «Ctrl+E» (для перехода в конец строки).

3. Нажмите «Ctrl+X» для загрузки ОС с указанными параметрами.

Загрузив систему, необходимо будет ответить на несколько простых вопросов, в частности, выбрать используемый язык и расположение корректного образа восстановления.

Если вы выбрали образ восстановления, который не требует подключения к сети, будет предложено определить, хотите ли вы установить сетевое подключение. Подключение к сети рекомендуется, если, например, нужно скопировать файлы на другой компьютер или установить какие-то RPM-пакеты с общего сетевого ресурса.

В режиме восстановления будет выполнена попытка найти установку ОС и подключить ее в /mnt/sysimage. После этого вы сможете внести необходимые изменения. Нажмите «Продолжить». Также можно подключить файловые системы в режиме чтения вместо чтения-записи. Если это не удалось, нажмите

кнопку «Пропустить» для перехода в командную оболочку.

При выборе «Продолжить» система попытается подключить файловую систему в /mnt/sysimage. Если смонтировать раздел не удастся, появится сообщение. При выборе варианта «только для чтения» будет предпринята попытка подключения файловой системы в /mnt/sysimage/ в режиме чтения. Если вы выберете «Пропустить», файловая система не будет подключена. Выберите «Пропустить», если считаете, что файловая система повреждена.

Как только система загрузится в режиме восстановления, на виртуальных консолях появится приглашение (используйте «Ctrl+Alt+Fx» для перехода в нужную консоль).

Даже если файловая система подключена, в режиме восстановления корневым разделом по умолчанию становится временный раздел, а не тот, что используется при работе в обычном режиме. Если файловая система была смонтирована успешно, можно сменить корневой раздел окружения режима восстановления на корневой раздел вашей файловой системы, выполнив команду:

```
chroot /mnt/sysimage
```

Это может пригодиться для выполнения команд, требующих, чтобы корневой раздел системы был подключен как / (таких как rpm). Чтобы выйти из окружения chroot, выполните команду «exit».

При выборе «Пропустить» можно попытаться смонтировать раздел или логический том LVM2 вручную в режиме восстановления, создав каталог, к примеру, с именем /foo, и выполнив следующую команду:

```
mount -t ext4 /dev/mapper/VolGroup00-LogVol02 /foo
```

В приведённой выше команде /foo — созданный вами каталог, а /dev/mapper/VolGroup00-LogVol02 — логический том LVM2, который вы хотите смонтировать. Если раздел имеет тип ext2 или ext3, замените ext4 на ext2 или ext3.

Если вы не знаете названий всех физических разделов, для их просмотра используйте команду:

```
fdisk -l
```

Если вы не знаете названий всех ваших физических томов LVM2, логических томов и их групп, их можно узнать, выполнив следующие команды:

```
pvdisplay
vgdisplay
lvdisplay
```

В строке приглашения можно выполнить множество полезных команд, включая следующие:

```
ssh, scp и ping, если сеть запущена;
dump и restore, если вы используете ленточные накопители;
```

`parted` и `fdisk` для управления разделами;
`rpm` для установки и обновления программного обеспечения;
`vi` для редактирования текстовых файлов.

После завершения работы с повреждённой системой, можно перезагружать ОС в нормальном режиме.

Аварийный режим

В аварийном режиме система будет загружена с минимальным окружением. Корневая файловая система подключается в режиме чтения и почти ничего настраивать не надо. Основным преимуществом этого режима является то, что файлы init не загружаются. Если окружение init повреждено и не работает, вы все же можете смонтировать файловые системы, чтобы восстановить данные, которые были потеряны при переустановке.

Чтобы загрузиться в аварийном режиме, выполните следующие действия:

1. Начните загрузку ПК и дождитесь появления меню GRUB. Нажмите клавишу «E» для редактирования параметров загрузки.
2. Добавьте в конец строки, начинающейся с «linux /boot/vmlinuz ...», следующую запись:

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

Используйте сочетание клавиш «Ctrl+A» (для перехода в начало строки) и «Ctrl+E» (для перехода в конец строки).

3. Нажмите «Ctrl+X» для загрузки ОС с указанными параметрами.

3.4.2 Режимы отображения информации

С точки зрения представления информации для пользователя можно выделить 2 режима работы ОС: графический и консольный.

Консолью называется совокупность основных устройств ввода информации в компьютер (клавиатура и мышь) и вывода информации (монитор). ОС работает с несколькими так называемыми виртуальными консолями, из которых в каждый момент времени только одна может быть связана с реальной (физической) консолью (то есть, является активной).

Некоторые из консолей представляют информацию только в текстовом виде с использованием экранных шрифтов в форматах видеосистемы компьютера — консольный режим работы. Такие консоли называются иногда ещё текстовыми. Сама ОС и основные автоматически запускаемые приложения (такие как командный процессор) используют в таких консолях интерфейс командной строки. Другие приложения (например, менеджер файлов Midnight Commander) могут использовать оконный интерфейс, выделение объектов и выбор в меню и списках при помощи мыши или клавиатуры и т.п.

Другие консоли (графические) представляют информацию в графическом виде, используя Графический пользовательский интерфейс (GUI) — графический режим работы. Как правило, работа в таких консолях происходит при помощи развитых графических сред, таких как MATE.

Для нужд ОС консоли перенумерованы целыми положительными числа-

ми. Их общее количество может изменяться в зависимости от настроек ОС. Несколько первых консолей - текстовые, далее идут графические (в стандартной настройке - одна).

Если Вы работаете в графической консоли, для того чтобы сделать активной другую консоль с номером n (где n находится в интервале от 1 до 12), нажмите на клавиши «Ctrl+Alt+Fn», то есть, например клавиши «Ctrl+Alt+F2», если хотите перейти в консоль с номером 2. Для того чтобы сделать активной другую консоль вместо текущей текстовой консоли, нажмите на клавиши «Alt+Fn».

Установка РЕД ОС

4. Установка РЕД ОС

4.1 Начало установки

Для начала установки, необходимо настроить BIOS СВТ, на которое производится установка РЕД ОС, на загрузку с оптического носителя информации (DVD-диска), на котором записан дистрибутив. Для этого необходимо включить в BIOS СВТ опцию загрузки с CD/DVD-привода. Способ входа в меню BIOS и расположение конкретных настроек может сильно отличаться, в зависимости от используемой материнской платы оборудования СВТ. Чаще всего для входа в BIOS необходимо нажать функциональную клавишу «Delete» на стационарных СВТ или функциональную клавишу «F1» («F9») на мобильных СВТ (ноутбуках), в момент начала загрузки компьютера. Для получения более подробных сведений по настройке необходимо обратиться к документации на используемое оборудование.

Загрузка с установочного носителя начинается с меню (рисунок 4.1), в котором перечислено несколько вариантов загрузки, причём установка системы - это только одна из возможностей. Из данного меню можно:

- запустить тестирование носителя данных и только после этого перейти к процедурам установке ОС;
- используя меню «Решение проблем», запустить проверку памяти и диагностику оборудования;
- запустить уже установленную ОС на жёстком диске;
- загрузить ОС в аварийном режиме.

Манипулятор графической информации (мышь) на этом этапе установки не поддерживается, поэтому для выбора различных вариантов и опций установки необходимо воспользоваться функциональными клавишами клавиатуры - стрелками. Можно скорректировать параметры запуска любого пункта начального меню установки РЕД ОС, нажав функциональную клавишу клавиатуры «TAB».

Кроме установки с оптического носителя информации доступно несколько вариантов сетевой установки и установка с флеш-диска.

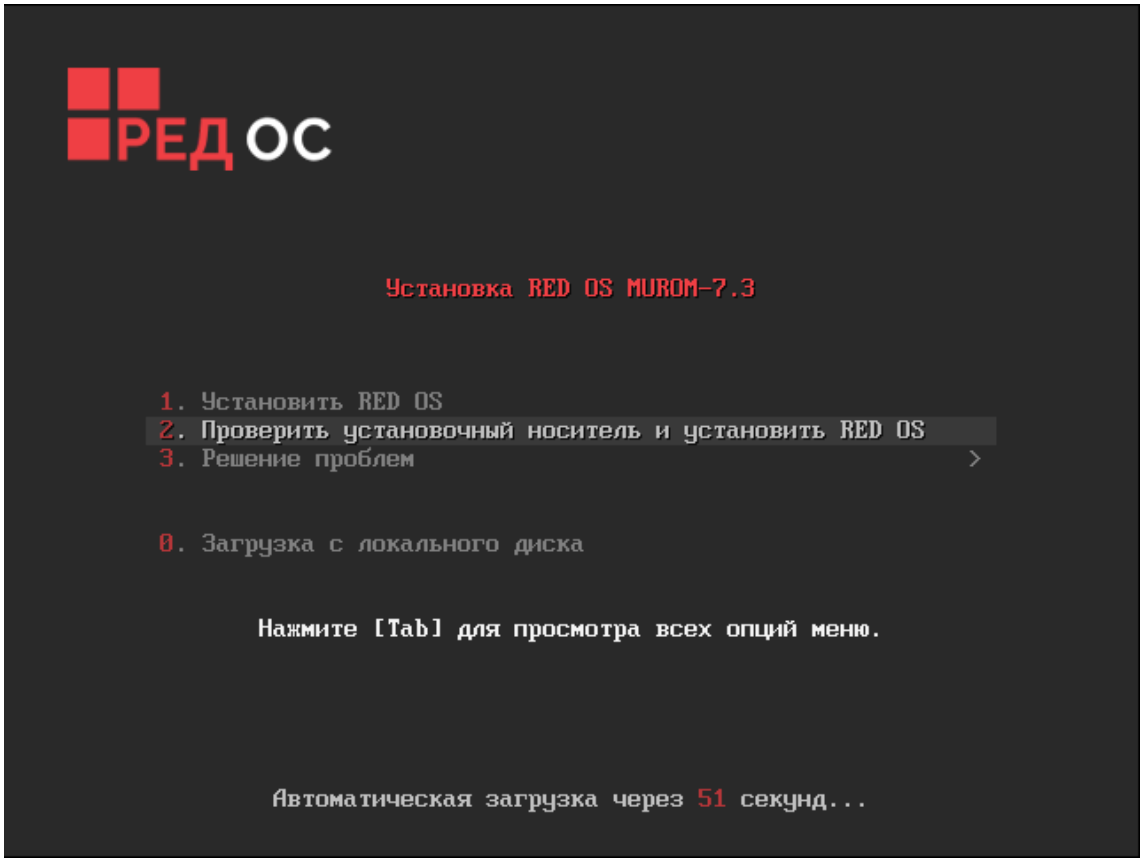

Рисунок 4.1 – Начальное меню установки РЕД ОС

Установка РЕД ОС автоматически осуществляется в графическом режиме с выводом текстовой информации на русском языке. Выбора языка установки РЕД ОС не осуществляется.

По умолчанию при установке используется графический видеорежим (разрешение экрана) 800 на 600 точек на дюйм. При возникновении проблем запуска установки в графическом режиме необходимо перезагрузить СВТ и повторить действия по установке, но с выбором в начальном меню установки с обычным видеодрайвером.

Чтобы начать процесс установки, нужно клавишами перемещения курсора «вверх», «вниз» выбрать пункт меню «Установить Red OS» и нажать «Enter». В начальном загрузчике установлено небольшое время ожидания продолжительностью 1 минута. Если в этот момент не предпринимать никаких действий, то будет загружена та система, которая уже установлена на жёстком диске. Если пропущен нужный момент, нужно перезагрузить компьютер и до истечения установленного времени ожидания выбрать нужный пункт в меню загрузчика.

Начальный этап установки не требует вмешательства пользователя: происходит автоматическое определение оборудования и запуск компонентов программы установки. Прервать начальный этап установки и вернуться в начальное меню установки РЕД ОС можно нажав клавишу «ESC».

Необходимость загрузиться в аварийном режиме может возникнуть в следующих случаях:

- невозможно загрузить ОС обычным образом;
- возникли программные или аппаратные проблемы и необходимо извлечь важные файлы с жёсткого диска.

Режим восстановления позволяет загрузить минимальное окружение ОС с дистрибутивного носителя вместо загрузки с жёсткого диска. В режиме восстановления можно получить доступ к файлам, даже если не удалось запустить ОС с этого диска.

В режиме восстановления будет выполнена попытка найти установку ОС и подключить ее в /mnt/sysimage. После этого можно будет внести необходимые изменения. Также можно подключить файловые системы в режиме чтения вместо чтения-записи.

В режиме восстановления также имеется возможность исправить работу загрузчика GRUB, который мог быть по ошибке удалён, повреждён или замещён загрузчиком другой операционной системы.

В ОС помимо режима восстановления есть аварийный режим. Режим восстановления эквивалентен однопользовательскому режиму и требует пароль root. Режим восстановления позволяет восстанавливать систему в ситуациях, когда она не может завершить обычный процесс загрузки. Режим восстановления попытается подключить все локальные файловые системы и запустить некоторые важные системные службы, но он не активирует сетевые интерфейсы и не разрешает вход в систему нескольким пользователям.

Аварийный режим обеспечивает минимальную возможную окружающую среду и позволяет восстанавливать систему даже в ситуациях, когда система не может войти в режим восстановления. В аварийном режиме система монтирует корневую файловую систему как доступную только для чтения, не пытается монтировать другие локальные файловые системы и не активирует сетевые интерфейсы.

4.2 Альтернативные способы установки

Для штатной установки дистрибутива РЕД ОС используется загрузочный оптический носитель информации - DVD-диск из комплекта поставки дистрибутива РЕД ОС. Если производится установка с такого диска, можно пропустить этот раздел и сразу перейти к следующему разделу.

Установка с оптического носителя информации (DVD-диска) - это лишь один из возможных способов установки РЕД ОС. Он подходит для большинства случаев, но не работает, например, в случае отсутствия на компьютере накопителя на оптических носителях информации - DVD-привода. Для таких случаев поддерживаются альтернативные методы установки. Важно понимать, что для начала установки необходимо две вещи: иметь возможность загрузить компьютер и иметь доступ к установочным файлам. В случае установочного дистрибутивного DVD-диска эти две возможности предоставляются самим диском: он является загрузочным и содержит все необходимые для установки файлы. Однако вполне допустим и такой вариант: первоначальная загрузка происходит

со специально подготовленного flash-диска, а установочные файлы берутся с FTP-сервера сети.

Таким образом, для установки дистрибутива необходимо:

- выбрать способ первоначальной загрузки компьютера;
- выбрать источник установки.

Для загрузки компьютера с целью установки операционной системы необходимо воспользоваться носителем, содержащим начальный загрузчик. Таким носителем может быть как сам загрузочный оптический диск, так и, например, flash-накопитель, который можно сделать загрузочным, воспользовавшись утилитой **iso2usb**.

4.2.1 Источники установки

После первоначальной загрузки с одного из поддерживаемых носителей, можно выбрать источник установки - место, откуда программа установки будет брать все необходимые при установке данные (прежде всего, устанавливаемое ПО). Так как установка системы возможна не только с лазерного диска, то можно выбрать один из поддерживаемых альтернативных источников установки.

Источники установки:

- Сетевые:
  - FTP-сервер;
  - NFS-сервер;
  - HTTP-сервер.
- Локальные:
  - Загрузочный флеш-диск.

Условием для всех способов установки является доступность дерева файлов, аналогичного содержимому установочного диска.

4.2.2 Запуск сетевой установки

Перед запуском сетевой установки производится настройка на сервере сети. Выполняется установка и настройка сервера DHCP.

```
dnf install dhcp
```

В файл конфигурации /etc/dhcp/dhcpd.conf вносятся следующие конфигурации:

```
subnet 10.YY.XX.0 netmask 255.255.255.0
{# диапазон ip-адресов организации
option subnet-mask 255.255.255.0;
option routers 10.YY.XX.1;
option broadcast-address 10.YY.XX.255;
option netbios-name-servers 10.YY.0.251;
option netbios-dd-server 10.YY.0.251;
option domain-name-servers 8.8.8.8;
option broadcast-address 10.YY.XX.255;
range 10.YY.XX.20 10.YY.XX.201;
```

```
}
default-lease-time 600;
max-lease-time 7200;
allow booting;
allow bootp;
option option-128 code 128 = string;
option option-129 code 129 = text;
next-server 10.YY.XX.1;
filename "/pxelinux.0";
```

Далее выполняется запуск сервиса:

```
systemctl start dhcpd
```

Выполняется установка и настройка tftp-сервера.

```
dnf install tftp-server
```

В файл конфигурации /etc/xinetd.d/tftp вносятся следующие конфигурации:

```
disable = yes на disable = no
```

Выполняем перезапуск xinetd:

```
systemctl restart xinetd
```

Выполняем подготовку необходимых файлов для начальной загрузки. Для этого устанавливаем утилиту syslinux.

```
dnf install syslinux
```

Копируем необходимые файлы в корневую директорию tftp-сервера:

```
cp /usr/share/syslinux/chain32.c32 /var/lib/tftpboot
cp /usr/share/syslinux/mboot.c32 /var/lib/tftpboot
cp /usr/share/syslinux/memdisk /var/lib/tftpboot
cp /usr/share/syslinux/menu.c32 /var/lib/tftpboot
cp /usr/share/syslinux/pxelinux.0 /var/lib/tftpboot
```

Создаём директории для конфигурации и дистрибутивов:

```
mkdir -p /var/lib/tftpboot/images/REDOS
mkdir -p /var/lib/tftpboot/pxelinux.cfg
```

Создаем файл меню /var/lib/tftpboot/pxelinux.cfg/default с содержимым:

```
default menu.c32
```

```
prompt 0
timeout 30
ONTIMEOUT local
MENU TITLE PXE Menu
LABEL REDOS
MENU LABEL REDOS
KERNEL images/REDOS/x86_64/7.3/images/pxeboot/vmlinuz
APPEND initrd=images/REDOS/x86_64/7.3/images/pxeboot/initrd.img
ramdisk_size=128000 ip=dhcp method=http://10.YY.X.1/images/REDOS
/x86_64/7.3/ devfs=nomount
LABEL REDOS 7 i686
MENU LABEL REDOS 7.3 i686
KERNEL images/REDOS/i686/7.3/images/pxeboot/vmlinuz
APPEND initrd=images/REDOS/images/pxeboot/initrd.img ramdisk_size
=128000 ip=dhcp method=http://10.YY.X.1/images/REDOS/ devfs=
nomount
```

Производим подготовку дистрибутивов РЕД ОС распаковкой содержимого установочных дисков РЕД ОС в соответствующие каталоги.

Для x86_64 архитектуры:

```
mount -t iso9660 -o loop REDOS-DVD1.iso /iso
cp -vR /iso/ /var/lib/tftpboot/images/REDOS/
umount /iso
```

Производим установку web-сервера для обеспечения раздачи установочных образов:

```
dnf install lighttpd
```

В файл конфигурации /etc/lighttpd/lighttpd.conf вносятся следующие изменения:

```
var.server_root = "/var/lib/tftpboot"
server.use-ipv6 = "disable"
server.bind = "10.YY.X.1"
server.document-root = server_root
```

Производим запуск web-сервера:

```
systemctl start lighttpd
```

Производится подготовка к сетевой установке РЕД ОС на рабочей станции. При сетевой установке со стороны клиента (компьютера, на который производится установка) необходимо определить параметры соединения с сервером установки. В этом случае на экране будут появляться диалоги, например, с предложением выбрать сетевую карту (если их несколько) или указать тип IP-адреса: статический (потребуется вписать его самостоятельно) или динамический (DHCP).

Для установки РЕД ОС на рабочей станции в BIOS СВТ выбираем загрузка по сети: «Network Boot Agent» или «PXE boot». Конфигурирование выполняется в соответствии с технической документацией на конкретное оборудование СВТ рабочей станции. После конфигурирования производится перезагрузка СВТ и выполняется установка согласно штатной процедуре установки.

После успешного соединения с сервером в память компьютера будет загружен образ установочного диска, после чего начнётся установка системы так же, как и при установке с дистрибутивного DVD-диска.

4.3 Последовательность установки

До того, как будет произведена установка РЕД ОС на жёсткий диск СВТ, программа установки работает с образом системы, загруженным в оперативную память компьютера.

Если инициализация оборудования СВТ завершилась успешно, будет запущен псевдографический интерфейс программы-установщика (anaconda). Процесс установки реализован в виде «мастера» установки, который представляет собой интерактивный графический интерфейс, в котором пользователю предлагается отвечать на вопросы и указывать необходимые опции РЕД ОС. Мастер установки разделен на шаги, каждый шаг посвящён настройке или установке определённого сервиса системы. Шаги нужно проходить последовательно, переход к следующему шагу происходит по нажатию кнопки «Далее». При помощи кнопки «Назад» при необходимости можно вернуться к уже пройденному шагу и изменить настройки. На некоторых этапах установки возможность перехода к предыдущему шагу ограничена теми шагами, где нет зависимости от данных, введённых ранее.

Если по каким-то причинам возникла необходимость прекратить установку, нажмите «Reset» на системном блоке компьютера.

Важно! Совершенно безопасно прекращать установку только до шага «Подготовка диска», поскольку до этого момента не производится никаких изменений на жёстком диске. Если прервать установку между шагами «Подготовка диска» и «Установка загрузчика», вероятно, что после этого с жёсткого диска не сможет загрузиться ни одна из установленных систем. ■

Технические сведения о ходе установки можно посмотреть, нажав сочетание клавиш «Ctrl+Alt+F1», вернуться к программе установки - «Ctrl+Alt+F7». По нажатию «Ctrl+Alt+F2» откроется отладочная виртуальная консоль.

Во время установки РЕД ОС выполняются следующие шаги:

- выбор типа накопителя СВТ;
- присвоение имени компьютера в сети и настройка сетевых интерфейсов;
- выбор часового пояса;
- задание пароля администратора системы;
- подготовка разделов диска;
- выбор типа установки: сервер или рабочая станция;
- установка системы;

- установка загрузчика;
- перезагрузка системы;
- лицензионный договор;
- создание системного пользователя;
- установка даты и времени;
- сохранение настроек;
- завершение установки.

4.4 Язык

Язык интерфейса программы установки и графического интерфейса устанавливаемой РЕД ОС по умолчанию – русский, не конфигурируется и не изменяется в процессе установки. Дополнительным языком РЕД ОС является английский язык. Другие дополнительные языковые пакеты можно установить из репозитория после завершения установки РЕД ОС.

Переключение раскладки клавиатуры при установке РЕД ОС и в графическом интерфейсе РЕД ОС выполняется нажатием комбинации функциональных клавиш «Shift+Alt».

4.5 Обзор установки

После выбора языка необходимо произвести первоначальную конфигурацию установщика и параметров будущей системы (рисунок 4.2).

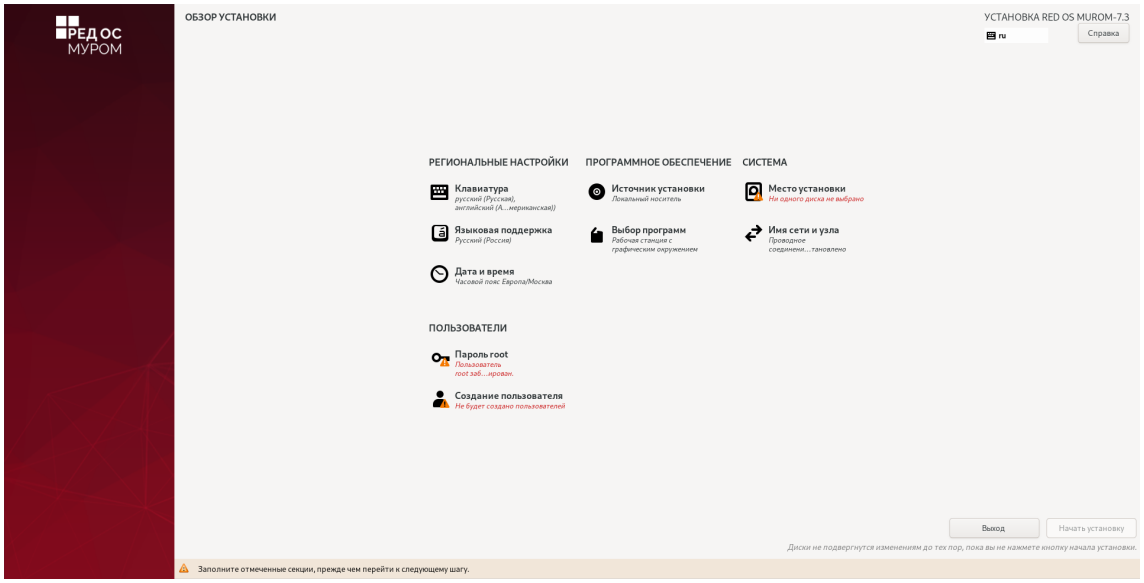

Рисунок 4.2 — Обзор установки

В этом окне необходимо задать региональные настройки, состав программного обеспечения и системные настройки. Здесь и в последующих окнах установщика красным цветом выводятся подсказки у тех вкладок, которые должны быть обязательно заполнены до перехода к следующему шагу установки.

Рассмотрим подробнее каждую вкладку окна.

Дата и время

В окне настройки даты и времени (рисунок 4.3) можно выбрать текущий регион и город и установить используемое локальное время, дату и формат времени.

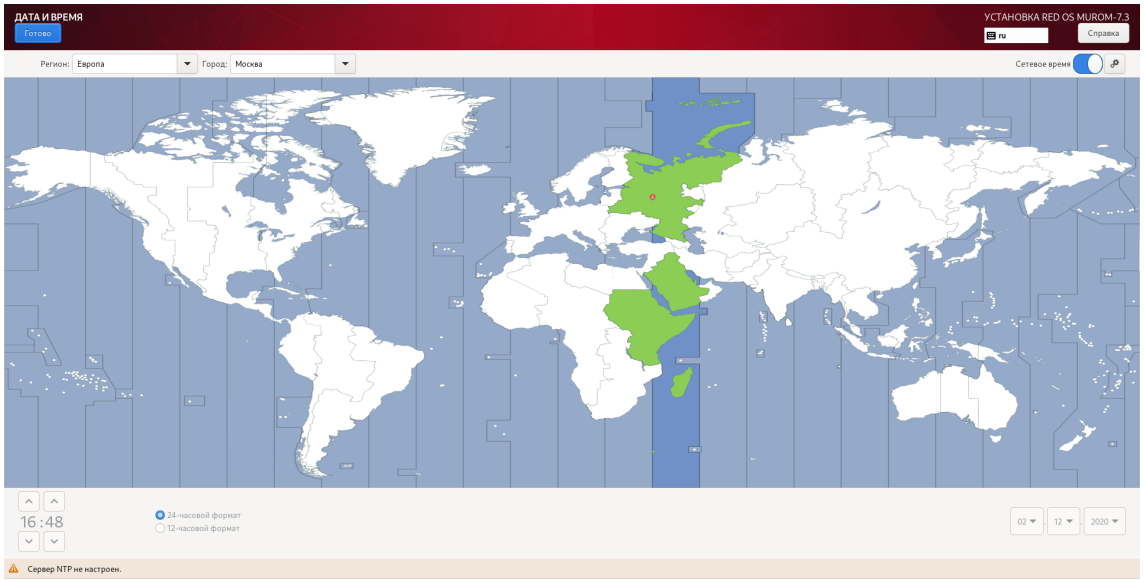

Рисунок 4.3 — Дата и время

Если установлена сеть, и есть доступ к глобальной сети, можно разрешить автоматическую настройку времени с помощью службы **ntp** — сетевое время.

Данную настройку можно изменить после завершения установки РЕД ОС. По умолчанию устанавливается часовой пояс UTC+03:00 (Европа/Москва).

Здесь и далее возврат в предыдущее меню осуществляется с помощью кнопки «Готово».

Клавиатура

В окне настройки клавиатуры (рисунок 4.4) можно выбрать используемые в ОС раскладки.

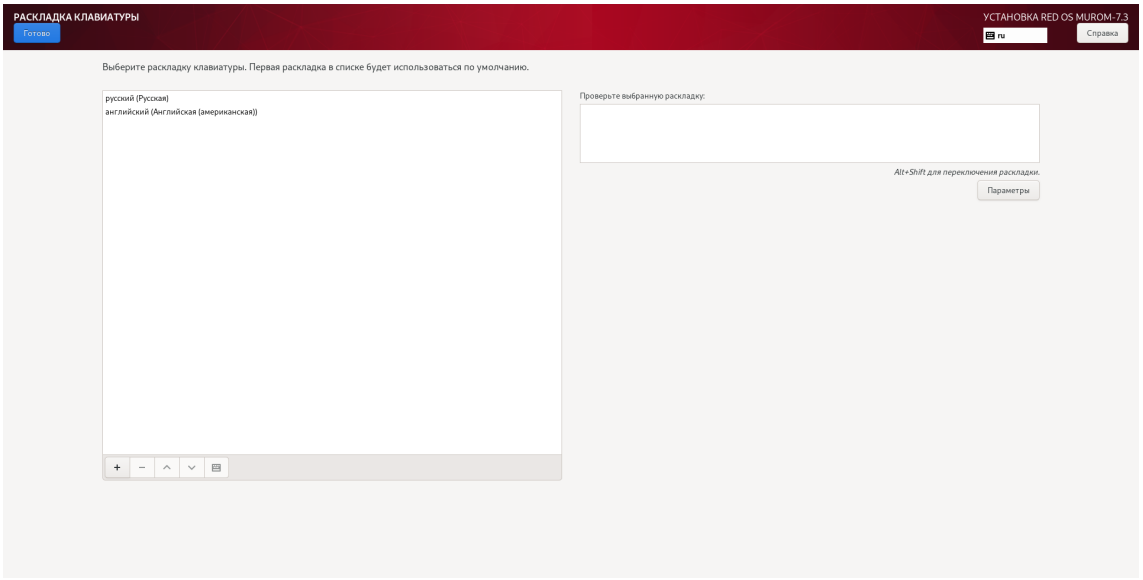

Рисунок 4.4 — Раскладка клавиатуры

В отдельном поле ввода можно проверить корректность отображения вводимых символов.

Первая раскладка в списке будет использоваться по умолчанию.

Языковая поддержка

В окне языковой поддержки (рисунок 4.5) можно задать языки, которые будут установлены в системе. Настройка и выбор соответствующего языка будут доступны в настройках системы после установки. Также после установки системы можно будет добавить дополнительные языки, которые не были установлены ранее. По умолчанию используется русский язык интерфейса.

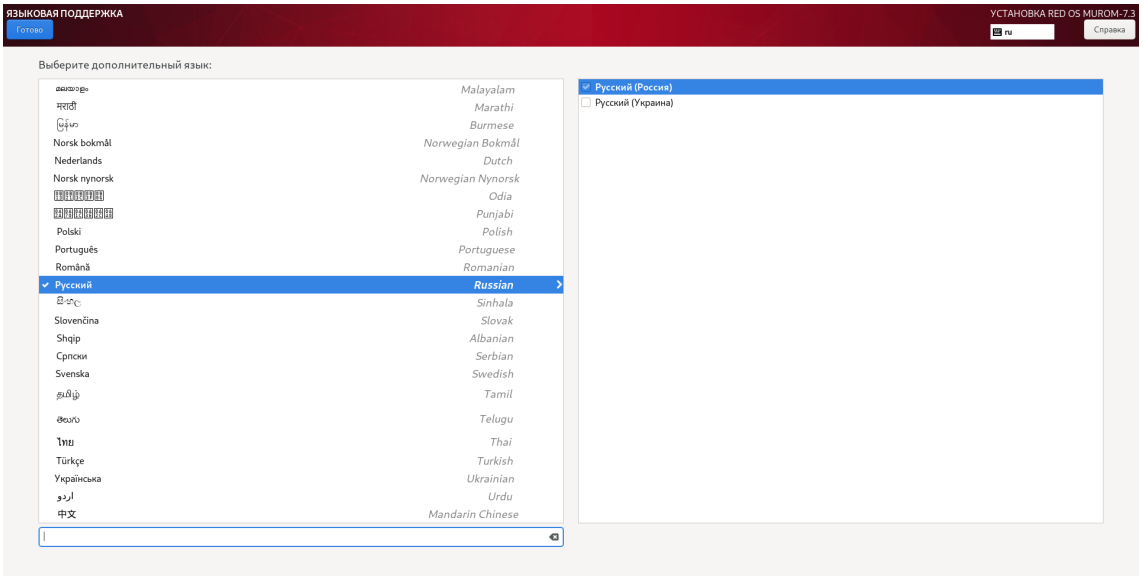

Рисунок 4.5 — Языковая поддержка

Источник установки

В меню выбора источников установки (рисунок 4.6) можно задать, откуда система будет получать пакеты в процессе установки.

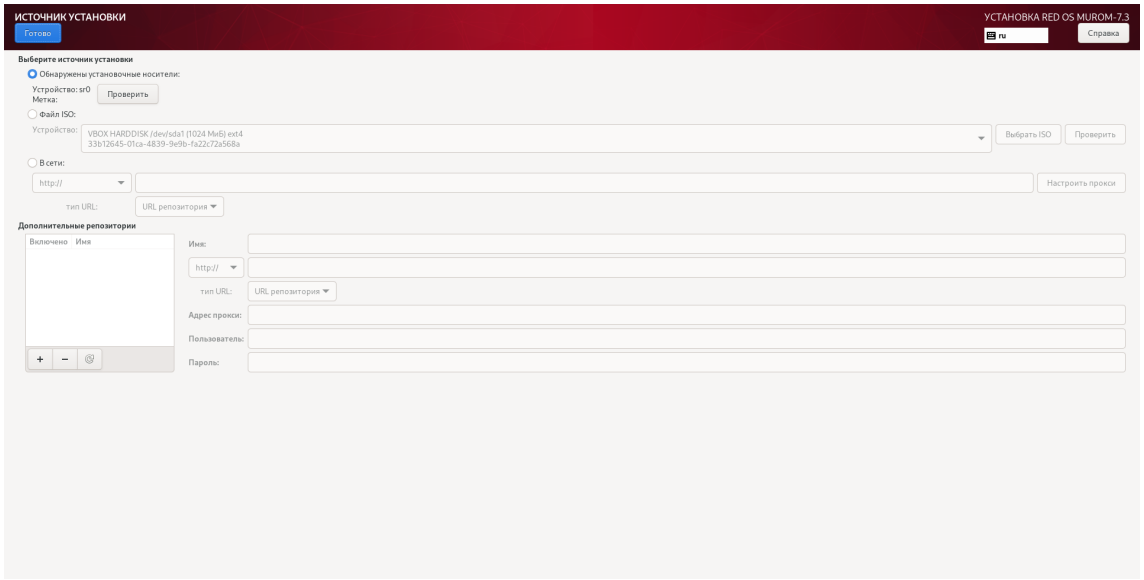

Рисунок 4.6 — Источник установки

Источниками установки могут быть локальные носители или сетевые источники.

Выбор программ

В меню выбора программ (рисунок 4.7) можно задать, какая конфигурация из числа базового окружения будет установлена:

- рабочая станция;
- сервер минимальный;
- сервер с графическим интерфейсом.

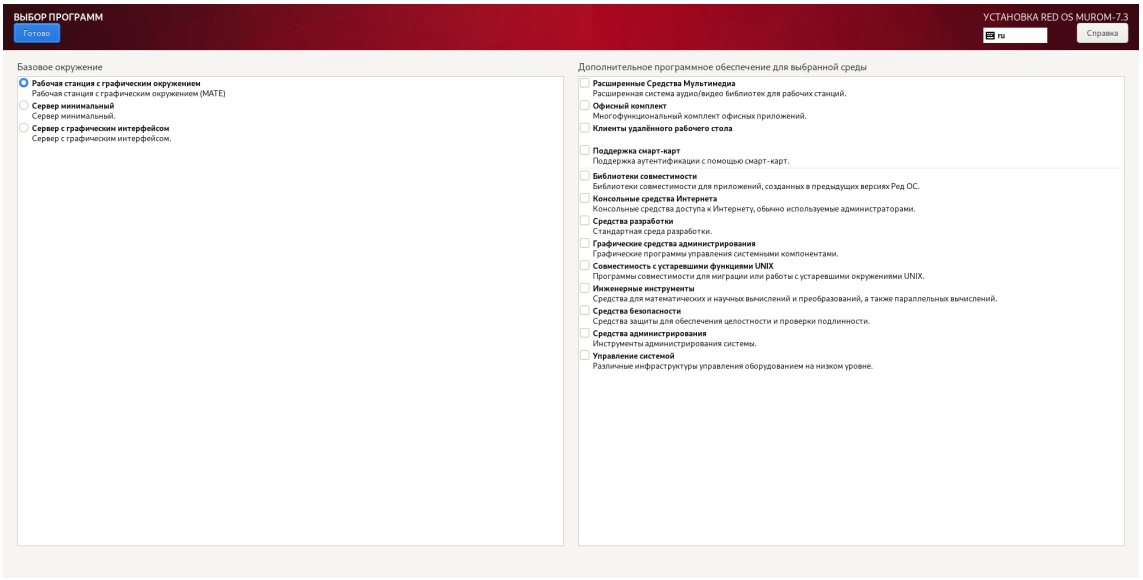

Рисунок 4.7 — Выбор программ

Дополнительно для каждой из выбранных конфигураций можно настроить группы пакетов, которые будут установлены.

В РЕД ОС все операции установки и удаления производятся над пакетами - отдельными компонентами системы. Пакет и программа соотносятся неоднозначно: иногда одна программа состоит из нескольких пакетов, иногда один пакет включает несколько программ.

В процессе установки РЕД ОС обычно не требуется детализированный выбор компонентов на уровне пакетов - это требует слишком много времени и знаний от проводящего установку. Тем более, что комплектация дистрибутива подбирается таким образом, чтобы из имеющихся программ можно было составить полноценную рабочую среду для соответствующей аудитории пользователей. Поэтому в процессе установки РЕД ОС пользователю предлагается выбрать из небольшого списка конфигураций, объединяющих пакеты, необходимые для решения наиболее распространённых задач.

Расположение установки

В меню выбора расположения установки (рисунок 4.8) можно выбрать устройство для установки операционной системы.

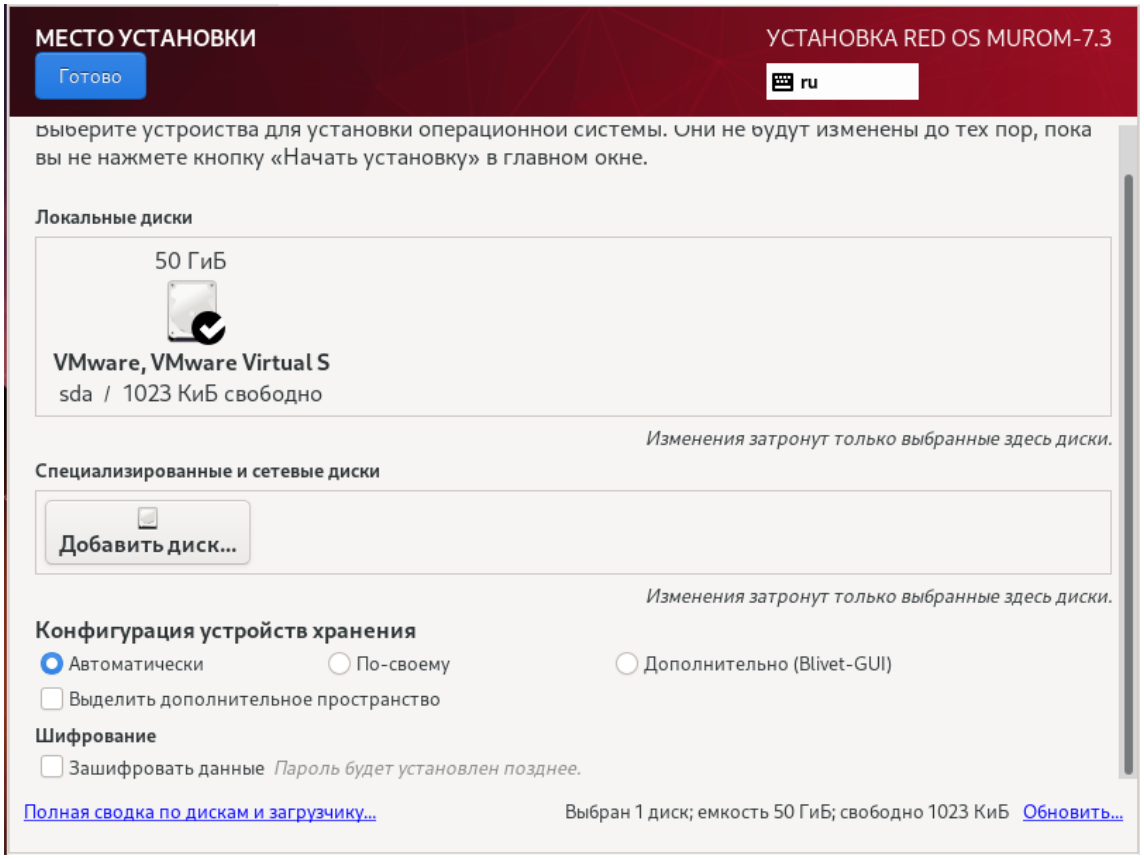

Рисунок 4.8 — Расположение установки

Переход к этому шагу может занять некоторое время. Время ожидания может быть разным и зависит от производительности компьютера, объёма и типов используемых накопителей, количества существующих разделов на них.

На этом этапе подготавливается площадка для установки РЕД ОС, в первую очередь - выделяется свободное место на диске.

Можно выбрать и использовать профили разбиения диска. Профиль - это шаблон распределения места на диске для установки РЕД ОС.

Можно выбрать:

- автоматически;
- по-своему;
- дополнительно (Blivet-GUI).

Первый профиль предполагает автоматическое разбиение диска. Будет выбрано оптимальное расположение ОС.

Необратимые изменения разделов на жёстком диске требуют подтверждения со стороны пользователя. После подтверждения внесённые изменения сохраняются на жестком диске/дисках СВТ.

Если для применения одного из профилей автоматической разметки доступного места окажется недостаточно, будет выведено сообщение о невозможности выполнения операции разбиения диска.

При необходимости можно освободить часть дискового пространства, для этого следует воспользоваться профилем разбиения вручную. Можно удалить

некоторые из существующих разделов или содержащиеся в них файловые системы. После этого можно создать необходимые разделы самостоятельно или вернуться к шагу выбора профиля. Выбор этой возможности требует знаний об устройстве диска и технологиях его разбиения.

По нажатию «Готово» будет произведена запись новой таблицы разделов на диск и форматирование разделов. Разделы, только что созданные на диске программой установки, пока не содержат данных и поэтому форматируются без предупреждения. Уже существовавшие, но изменённые разделы, которые будут отформатированы, помечаются специальным значком в колонке «Файловая система».

Не следует форматировать разделы с теми данными, которые необходимо сохранить, например, с пользовательскими данными (/home) или с другими операционными системами. С другой стороны, отформатировать можно любой раздел, который необходимо «очистить» (т.е. удалить все данные).

Blivet-gui является обособленной реализацией механизмов управления разделами и дисками, снабжённой привычным интерфейсом в стиле GParted. Программа поддерживает управление дисковыми разделами и хранилищами LVM2 (включая шифрованные разделы LUKS, логические тома и группы томов). Так же как и в GParted, изменения в blivet-gui применяются не сразу, а после подтверждения внесённого набора изменений.

Сеть и имя узла

В меню выбора настройки сети (рисунок 4.9) можно активировать соединение с сетью и задать имя узла.

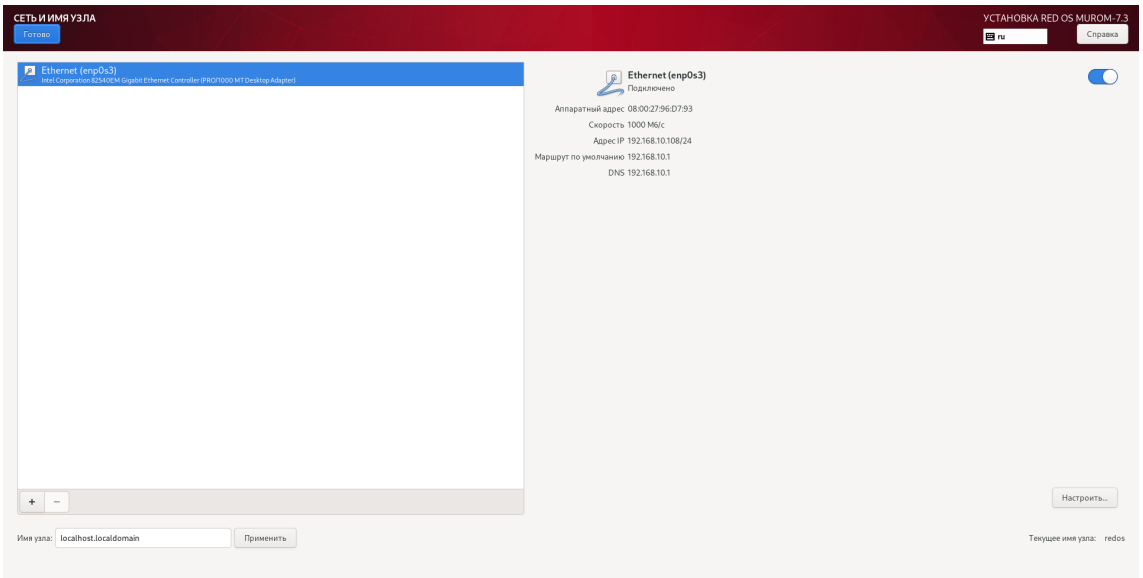

Рисунок 4.9 — Сеть и имя узла

На данном этапе необходимо задать имя компьютера в сети (хоста). При наличии сети имя компьютера используется для однозначного определения каждого компьютера. Имя компьютера состоит непосредственно из имени компьютера и

имени домена в сети, к которому принадлежит компьютер. Имя компьютера и имя домена разделяются знаком «.» .

При наличии домена сети имя должно даваться полностью. При отсутствии сети домену также может быть дано произвольное имя.

В качестве букв в имени компьютера разрешается использовать только буквы латиницы. В имени компьютера не допускается использование заглавных букв, пробелов и специальных символов.

Так же на данном этапе можно сконфигурировать параметры настройки сетевых интерфейсов: автоматическое включение интерфейсов, MAC-адреса сетевых интерфейсов, параметры сетевых протоколов.

4.6 Задание пароля администратора системы

После того как предварительная настройка системы завершена, начинается непосредственная установка системы. Параллельно с установкой необходимо настроить пароль администратора root и при необходимости создать учётные записи пользователей (рисунок 4.10)

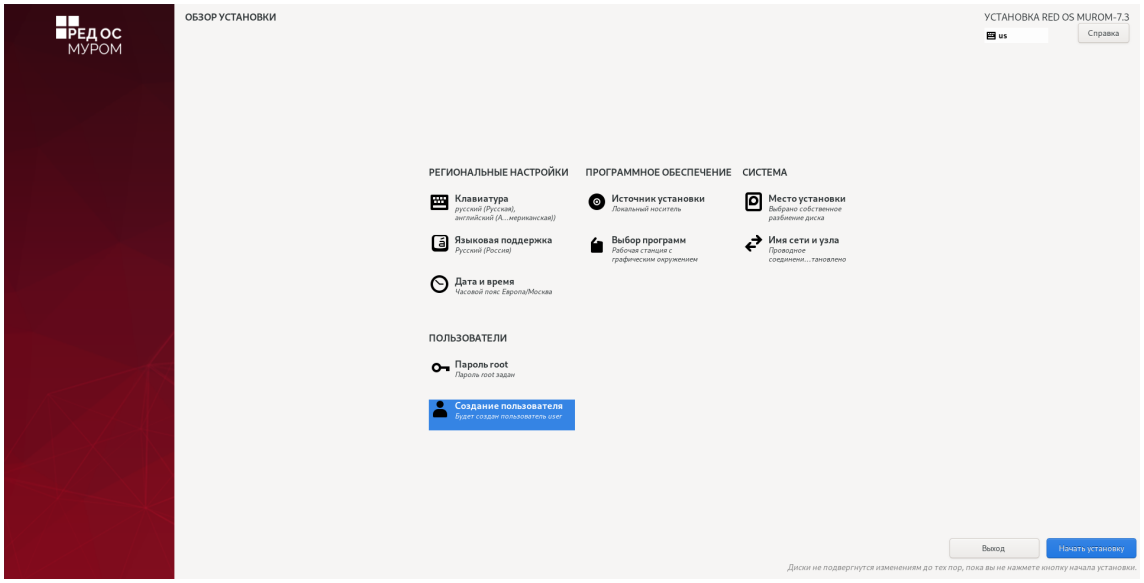

Рисунок 4.10 — Конфигурация ОС

РЕД ОС - это многопользовательская операционная система. На практике это означает, что для работы в системе нужно в ней зарегистрироваться, т.е. дать понять системе, кто именно находится за монитором и клавиатурой. Наиболее распространённый способ регистрации на сегодняшний день - использование системных имён (login name) и паролей. Это надёжное средство убедиться, что с системой работает тот, кто нужно, если пользователи хранят свои пароли в секрете и если пароль достаточно сложен и не слишком короток (иначе его легко угадать или подобрать).

В ОС всегда присутствует один специальный пользователь - администратор, он же суперпользователь или администратор РЕД ОС, для него зарезервировано стандартное системное имя - root.

Необходимо запомнить пароль root - его нужно будет вводить, чтобы получить право изменять настройки системы с помощью стандартных средств настройки РЕД ОС (рисунок 4.11).

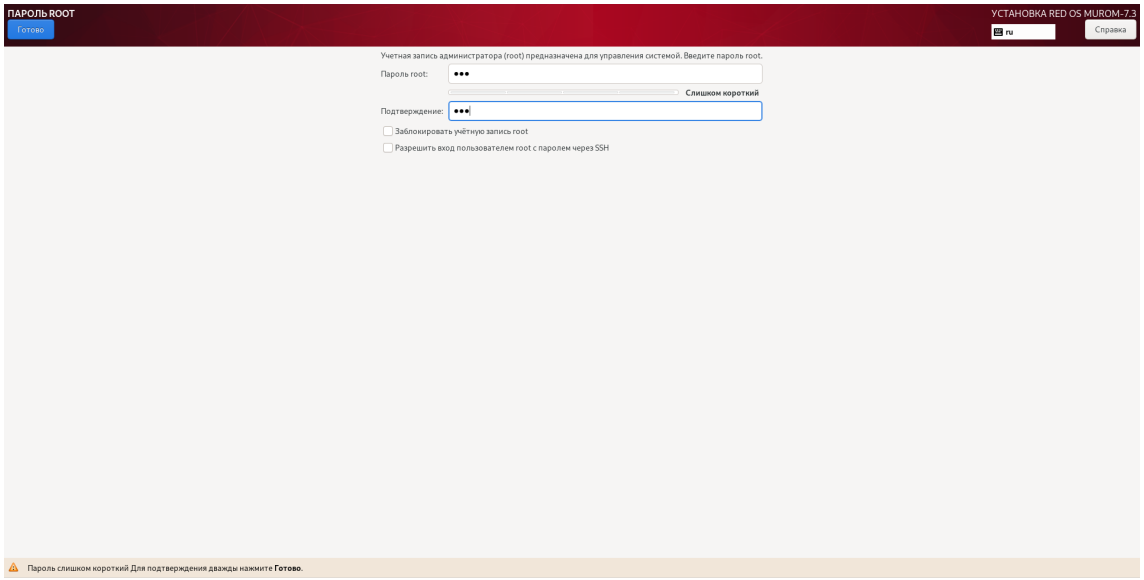

Рисунок 4.11 – Задание пароля администратора РЕД ОС

Ввод пароля защищён, при наборе пароля вместо символов на экране отображаются специальные символы. Чтобы избежать опечатки при вводе пароля, его предлагается ввести дважды. К введённому паролю в режиме реального времени применяется политика сложности пароля, т.е. производится его проверка, при слишком простом пароле или совпадении пароля с парольной последовательностью из словаря паролей системой будет предложено произвести смену пароля администратора РЕД ОС.

Администратор отличается от всех прочих пользователей тем, что ему позволено производить любые, в том числе самые разрушительные, изменения в системе. Поэтому выбор пароля администратора РЕД ОС - очень важный момент для безопасности: любой, кто сможет ввести его правильно (узнать или подобрать), получит неограниченный доступ к системе. Даже ваши собственные неосторожные действия от имени root могут иметь катастрофические последствия для всей системы.

Помимо администратора РЕД ОС (root) в систему необходимо добавить, по меньшей мере, одного обычного пользователя. Работа от имени администратора РЕД ОС считается опасной (можно по неосторожности повредить систему), поэтому повседневную работу в РЕД ОС следует выполнять от имени обычного пользователя, полномочия которого ограничены.

При добавлении пользователя предлагается ввести имя учётной записи (login name) пользователя. Имя учётной записи всегда представляет собой одно слово, состоящее только из строчных латинских букв (заглавные запрещены), цифр и символа подчёркивания «_» (причём цифра и символ «_» не могут стоять в начале слова). Чтобы исключить опечатки, пароль пользователя вводится дважды.

Так же, как при выборе пароля администратора РЕД ОС (root), действуют требования по сложности пароля.

В процессе установки предлагается создать только одну учётную запись обычного пользователя - чтобы от его имени системный администратор РЕД ОС мог выполнять задачи, которые не требуют привилегий суперпользователя.

Учётные записи для всех прочих пользователей системы можно будет создать в любой момент после её установки.

4.7 Установка системы

Этап установки происходит параллельно с заданием пароля администратора и представляет собой установку набора программ, необходимых для работы системы (рисунок 4.12).

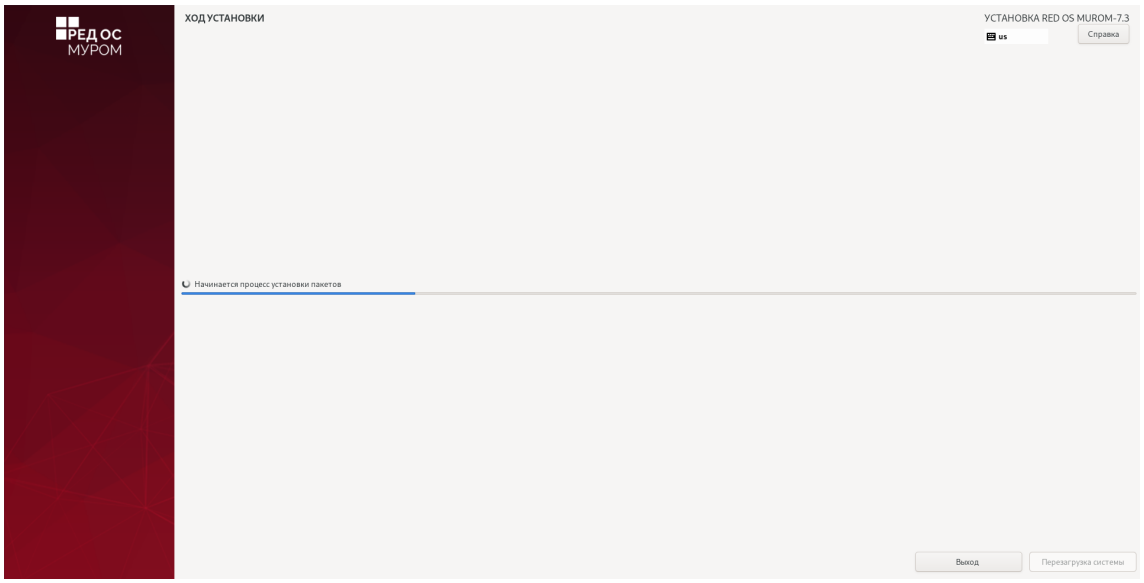

Рисунок 4.12 – Установка пакетов РЕД ОС

Получение пакетов осуществляется с источника, выбранного на этапе начальной загрузки. При сетевой установке (по протоколу FTP или HTTP) время выполнения этого шага будет зависеть от скорости соединения и может быть значительно большим, чем при установке с дистрибутивного DVD-диска.

Начиная с этого шага, программа установки работает с файлами только что установленной базовой системы. Все последующие изменения можно будет совершить после завершения установки посредством редактирования соответствующих конфигурационных файлов или при помощи модулей управления, включённых в дистрибутив.

Загрузчик — программа, которая позволяет загружать установленные операционные системы. При автоматическом разбиении разделов накопителя на жестких дисках средства вычислительной техники загрузчик автоматически устанавливается в начальный раздел диска.

Если же планируется использовать и другие операционные системы, уже установленные на этом компьютере, тогда имеет значение, на каком жёстком дис-

ке или разделе будет расположен загрузчик. В большинстве случаев программа установки правильно подберёт расположение загрузчика.

После выполнения копирования файлов РЕД ОС и установки загрузчика пользователю предлагается произвести перезагрузку РЕД ОС кнопкой «Перезагрузить».

4.8 Соглашение пользователя и Лицензионный договор

После успешной перезагрузки и перед продолжением установки следует внимательно прочитать условия пользовательского соглашения (рисунок 4.13).

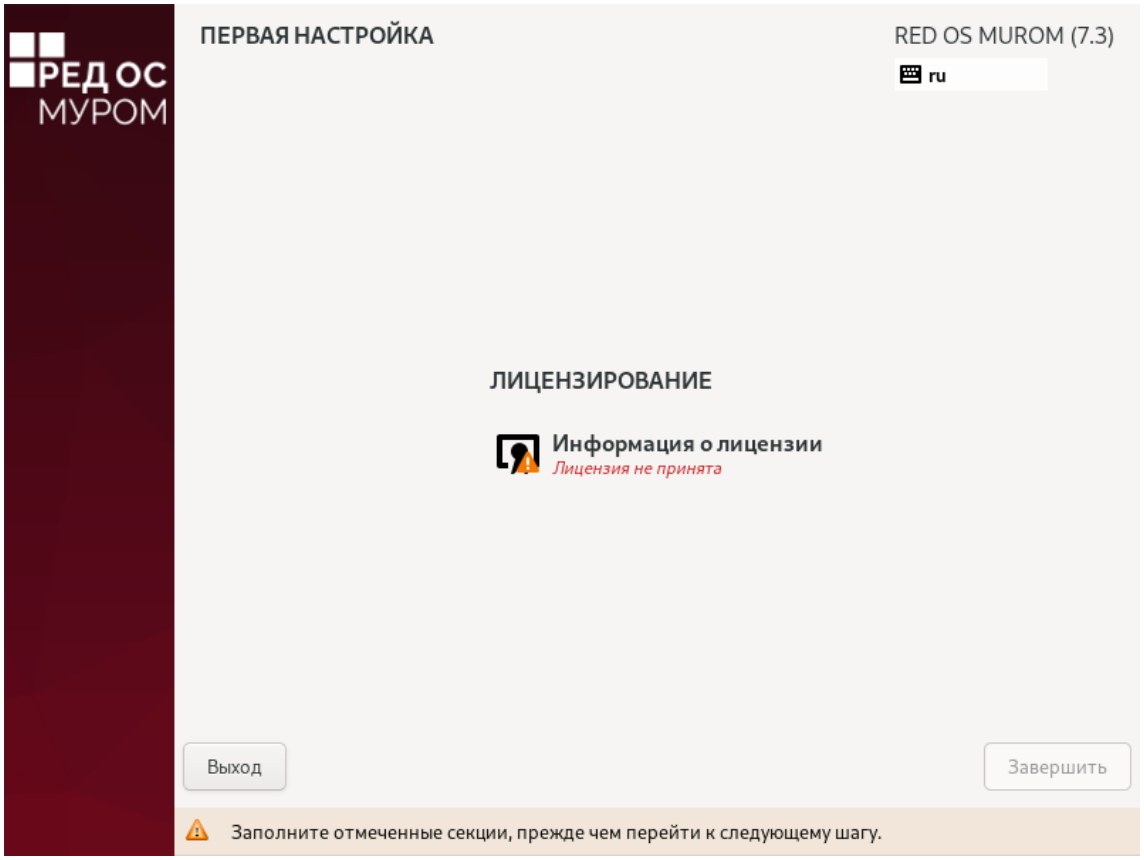

Рисунок 4.13 – Первая настройка

В лицензии (рисунок 4.14) говорится о правах распространения и гарантиях производителя.

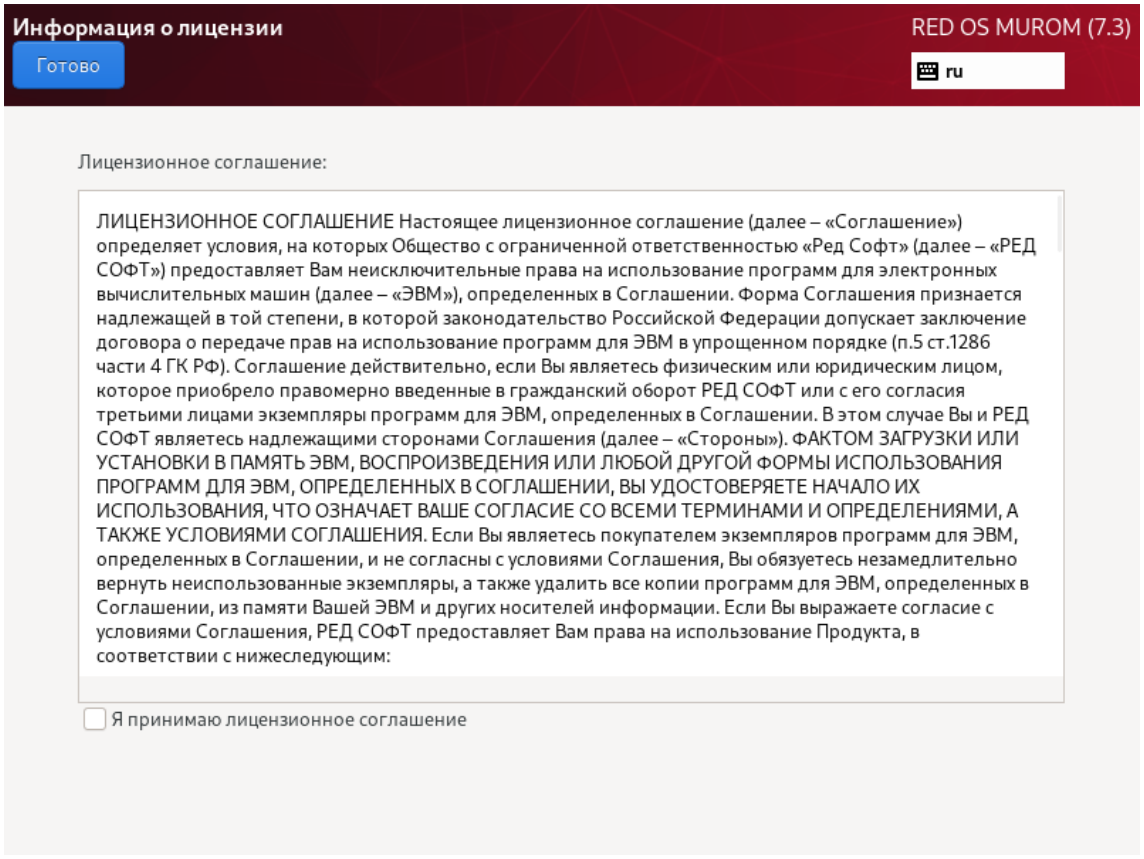

Рисунок 4.14 - Лицензия

При приобретении дистрибутива данное лицензионное соглашение прилагается в печатном виде к копии дистрибутива на вкладыше комплекта дистрибутива. Лицензия относится ко всему дистрибутиву РЕД ОС.

4.9 Завершение установки

После сохранения настроек и перезапуска системы пользователю предоставляется экран приветствия и приглашения к авторизации. На этом установка и настройка РЕД ОС завершена, и РЕД ОС полностью готова к использованию.

4.10 Подготовительные процедуры

После установки РЕД ОС не все функции безопасности включены по умолчанию. Администратор должен определить функциональную роль установленной копии ОС, перечень решаемых задач, и в соответствии с действующими политиками безопасности организации и роли пользователя, который планирует использовать ОС, произвести активацию необходимых служб, сервисов и приложений. А именно:

- сконфигурировать перечень отслеживаемых событий безопасности (принимая во внимание то, что журналы аудита при отслеживании всех системных событий могут иметь огромный размер и быстро исчерпать свободное место

на жёстком диске, по умолчанию в ОС включено отслеживание лишь определённого числа критически важных событий; как настроить необходимые в данный момент события см. в разделе «Аудит в РЕД ОС»);

- настроить действия в случае обнаружения критически важных событий безопасности (см. раздел «Аудит в РЕД ОС»);
- настроить действия для предотвращения потери данных аудита (см. описание конфигурационного файла auditd.conf);
- для защиты от несанкционированных изменений и прерывания нормальной загрузки ОС рекомендуется установить пароль на загрузчик GRUB (см. подраздел «Назначение пароля на загрузчик GRUB2»);
- создать необходимое количество пользователей, групп, ролей и связать пользователей с группами и ролями, исходя из необходимых им полномочий (см. подразделы «Управление пользователями», «SELinux»);
- установить права доступа на создаваемые вновь каталоги и файлы (см. подраздел «Права доступа к файлам и каталогам»);
- определить правила установки программ и правила запуска компонентов программного обеспечения (см. подраздел «Система управления пакетами DNF», «Systemd»);
- определить и настроить используемые механизмы идентификации и аутентификации (по умолчанию используется простая парольная аутентификация и идентификация по логину пользователя), сроки действия учётных записей и паролей, требования, предъявляемые к аутентификационным данным, задать для пользователей системы не более 4 неуспешных попыток аутентификации, после которых должна выполниться временная блокировка учётной записи пользователя (см. подраздел «PAM»);
- создать необходимые резервные копии данных, аудита и системных настроек и проводить периодическое их тестирование (см. подразделы «Утилиты для работы с файловыми системами», «Crontab», «AMTU», «Экспорт данных пользователя», «Резервирование данных»);
- настроить доверенные сервера точного времени (см. подраздел «ntpdate»);
- проверить целостность важных данных и системных файлов (см. подраздел «Afick»);
- организовать отказоустойчивый кластер при необходимости обеспечения пониженной отказоустойчивости (см. подраздел «Отказоустойчивый кластер»);
- определить квоты и приоритеты, выделяемые пользователям (см. подразделы «Изменение приоритета процесса», «Управление дисковыми квотами», «Ограничение ресурсов пользователя»);
- установить необходимые ограничения на сеансы пользователей (см. подраздел «PAM»);
- установить продолжительность бездействия пользовательского сеанса, после которого сеанс будет заблокирован или завершён (см. подраздел «Настройка хранителя экрана, блокировки и режима энергосбережения» документа «Операционная система «РЕД ОС». Руководство пользователя»).

Информация, приведённая в данном руководстве, должна использоваться совместно со встроенной справочной системой man.

Управление программным обеспечением

5. Управление ПО

5.1 Введение: пакеты, зависимости и репозитории

В РЕД ОС огромное число общих ресурсов, которыми пользуются сразу несколько программ: разделяемых библиотек, содержащих стандартные функции, исполняемых файлов, сценариев и стандартных утилит и т.д. Удаление или изменение версии одного из составляющих систему компонентов может повлечь неработоспособность других, связанных с ним компонентов, или даже вывести из строя всю систему. В контексте системного администрирования проблемы такого рода называют нарушением целостности системы. Задача администратора - обеспечить наличие в системе согласованных версий всех необходимых программных компонентов (обеспечение целостности системы).

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 и - если это предусмотрено - появляется в меню всех графических оболочек.

Важно! Благодаря менеджерам пакетов, пользователю обычно не требуется непосредственно обращаться к установочным процедурам отдельных

программ или непосредственно работать с каталогами, в которых установлены исполняемые файлы и компоненты программ (обычно это /usr/bin, /usr/share/<имя_пакета>) - всю работу делает менеджер пакетов. Поэтому установку, обновление и удаление программ обычно называют управлением пакетами. ■

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.

Однако менеджеры пакетов оказались неспособны предотвратить все возможные коллизии при установке или удалении программ, а тем более эффективно устранить нарушения целостности системы. Особенно сильно этот недостаток сказывается при обновлении систем из централизованного репозитория пакетов, в котором последние могут непрерывно обновляться, дробиться на более мелкие и т. п. Этот недостаток и стимулировал создание систем управления программными пакетами и поддержания целостности системы.

Для автоматизации этого процесса в РЕД ОС применяется система управления программными пакетами DNF.

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.

Команда	Описание команды
`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).

В аргументах обычно используется два варианта ссылок на пакеты.

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

```
~/RPMS/apache-1.3.3-1.i386.rpm
```

В принципе, rpm понимает имена файлов в виде ftp-URL, т.е.:

```
ftp://<сервер>/<директория>/<файл>.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:

```
acroread-3.01-4
acroread
```

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).

Поскольку некоторые пакеты содержат очень большое количество файлов, то стоит отправлять вывод от «rpm -ql» команде «less».

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. Особенно часто это требуется перед установкой пакета. Команда «rpm -y» пакет позволяет сравнить текущее состояние файлов пакета с информацией, записанной в базе данных. Это требуется, например, при проверке, не испорчены ли какие-нибудь важные для системы файлы (такое случается после внезапного отключения питания). При нахождении различий печатается ключевая строка, с обозначением отличий и имя файла, в котором они найдены.

5.3 Назначение DNF

Фактически, dnf представляет собой оболочку для rpm, обеспечивающую работу с репозиториями. Утилита dnf - это менеджер пакетов, который умеет запрашивать информацию о пакетах, получать пакеты из репозиториев, устанавливать и удалять их, используя автоматическое разрешение зависимостей, а также обновлять целиком систему до последних версий пакетов. DNF выполняет автоматическое разрешение зависимостей для пакетов, которые обновляются, устанавливаются или удаляются, и, таким образом, позволяет автоматически определять, получать и устанавливать все доступные по зависимостям пакеты. Для DNF можно настроить новые, дополнительные репозитории, или, по-другому, источники пакетов, кроме того, для него доступны многие дополнения, которые улучшают и расширяют его возможности. DNF позволяет выполнять многие из задач, которые выполняет RPM; кроме того, многие из опции командной строки у него также подобны опциям RPM. Утилита dnf обеспечивает простое и легкое управление пакетами на одной машине или же на группе машин.

DNF обеспечивает безопасное управление пакетами путем включения проверки сигнатур GPG для пакетов, подписанных с помощью GPG, для всех репозиториев пакетов или для отдельных репозиториев. В случае включения проверки сигнатур, DNF откажется устанавливать любые пакеты, не подписанные корректным ключом для данного репозитория. Это означает, что можно доверять пакетам RPM, которые скачиваются и устанавливаются на машине в том случае, если они получены из доверенных источников, и они не были изменены в процессе передачи.

DNF также позволяет легко создавать собственные репозитории RPM-пакетов для скачивания и установки их на других машинах.

5.4 Источники программ (репозитории)

Репозитории, с которыми работает DNF, отличаются от обычного набора пакетов наличием мета информации - индексов пакетов, содержащихся в репозитории, и сведений о них. Поэтому, чтобы получить всю информацию о репозитории, DNF достаточно получить его индексы.

DNF может работать с любым количеством репозиториев одновременно, формируя единую информационную базу обо всех содержащихся в них пакетах. При установке пакетов DNF обращает внимание только на название пакета, его версию и зависимости, а расположение в том или ином репозитории не имеет значения. Если потребуется, DNF в рамках одной операции установки группы пакетов может пользоваться несколькими репозиториями.

Важно! При использовании РЕД ОС (стандартной и сертифицированной редакции), в целях предотвращения нарушения целостности системы, ЗАПРЕЩАЕТСЯ настройка служб автоматического обновления на использование внешних сторонних репозиториев операционных систем.

Подключение репозитория сторонней ОС влечет за собой нарушение зависимостей в устанавливаемых пакетах. Принудительная установка пакетов из репозитория сторонней ОС может привести к неработоспособности как отдельных компонентов системы, так и ОС в целом.

Если для обеспечения рабочих процессов пользователя нужен пакет (программа, утилита), которого нет в РЕД ОС (и/или требуется обновление версий пакетов, которые уже есть в репозитории РЕД ОС), необходимо обратиться в техническую поддержку с запросом на добавление нового пакета или обновление версии пакетов РЕД ОС. ■

DNF позволяет взаимодействовать с репозиторием с помощью различных протоколов доступа. Наиболее популярные - HTTP и FTP, однако существуют и некоторые дополнительные методы.

Для того чтобы DNF мог использовать тот или иной репозиторий, информацию о нем необходимо поместить в папку /etc/yum.repos.d/

После того как отредактирован список репозиториев в sources.list, необходимо обновить локальную базу данных DNF о доступных пакетах. Это делается командой:

```
dnf update
```

При выборе пакетов для установки DNF руководствуется всеми доступными репозиториями вне зависимости от способа доступа к ним. Так, если в репозитории, доступном по сети Интернет, обнаружена более новая версия программы, чем на компакт-диске, то DNF начнёт загружать данный пакет из сети Интернет.

Для создания репозитория, следуйте следующим указаниям:

- Установить пакет createrepo:

```
dnf install createrepo
```

- Скопировать все пакеты в один каталог, например:

/mnt/local_repo.

- Запустить команду «createrepo --database», указав этот каталог:

```
createrepo --database /mnt/local_repo
```

5.5 Поиск пакетов

Если пользователь не знает точного названия пакета, для его поиска можно воспользоваться утилитой «dnf search», которая позволяет искать не только по имени пакета, но и по его описанию.

Команда

```
dnf search <подстрока>
```

позволяет найти все пакеты, в именах или описании которых присутствует указанная подстрока.

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

```
dnf info gdm-libs
```

5.6 Установка или обновление пакета

Установка пакета с помощью DNF выполняется командой:

```
dnf install <имя_пакета>
```

DNF 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.

Команда «dnf install <имя_пакета>» используется также для обновления уже установленного пакета или группы пакетов. В этом случае dnf дополнительно проверяет, не обновилась ли версия пакета в репозитории по сравнению с установленным в системе.

При помощи DNF можно установить и отдельный бинарный rpm-пакет, не входящий ни в один из репозиториев (например, полученный из Интернет). Для этого достаточно выполнить команду:

```
dnf install <путь_к_файлу.rpm>
```

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

Иногда, в результате операций с пакетами без использования DNF, целостность системы нарушается, и dnf отказывается выполнять операции установки, удаления или обновления. В этом случае необходимо повторить операцию, задав

опцию «-f», заставляющую dnf исправить нарушенные зависимости, удалить или заменить конфликтующие пакеты. В этом случае необходимо внимательно следить за сообщениями, выдаваемыми dnf. Любые действия в этом режиме обязательно требуют подтверждения со стороны пользователя.

5.7 Удаление установленного пакета

Для удаления пакета используется команда:

```
dnf 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:

```
dnf check-update
```

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

```
dnf upgrade
```

5.9 Puppet

Puppet - это кроссплатформенная структура, позволяющая системным администраторам выполнять общие задачи с использованием кода. Код позволяет выполнять различные задачи от установки новых программ до проверки прав доступа файлов или обновлений пользовательских учетных записей. В большинстве случаев puppet используется в конфигурации клиент/сервер.

Этот раздел показывает установку и настройку Puppet в конфигурации клиент/сервер. Этот простой пример демонстрирует как установить Apache с использованием Puppet.

Для установки Puppet введите в терминале:

```
dnf install puppet-server
```

На клиентской машине (или машинах) введите:

```
dnf install puppet
```

Прежде чем настраивать puppet, вам, возможно, понадобится добавить запись DNS CNAME для puppet.example.com, где example.com - это ваш домен.

По умолчанию клиенты Puppet проверяют DNS на наличие puppet.example.com в качестве имени puppet сервера (Puppet Master).

Если вы не предполагаете использовать DNS, вы можете добавить записи в файл /etc/hosts на сервере и клиенте. Например, в файл /etc/hosts Puppet-сервера добавьте:

```
127.0.0.1 localhost.localdomain localhost puppet
192.168.1.17 meercat02.example.com meercat02
```

На каждом Puppet клиенте добавьте запись для сервера:

```
192.168.1.16 meercat.example.com meercat puppet
```

Примечание. Не забудьте заменить IP-адреса и доменные имена из примера на ваши актуальные адреса и имена сервера и клиентов.

Теперь настроим некоторые ресурсы для Apache HTTP Server. Создайте файл /etc/puppet/ code/environments/production/modules/install_pkg/manifests/init.pp, содержащий следующее:

```
class install_pkg {
package {
'Apache HTTP Server':
ensure => installed
}
service {
'Apache HTTP Server':
ensure => true,
enable => true,
require => Package['Apache HTTP Server']
}
}
```

Обратите внимание, что был создан класс «install_pkg», совпадающий с именем модуля «install_pkg». Модули — это возможность использовать классы и функции в puppet. Модуль — это всего лишь директория с предопределённой структурой каталогов и файлов. Выглядит структура модуля примерно так: <имя_модуля>/

- files/ - содержит различные файлы;
- lib/ - содержит плагины, например, особые переменные (facts) или типы ресурсов (может вы создадите свой file_yours тип);
- manifests/ - хранит все манифесты;
- init.pp - должен быть обязательно;
- <класс1>.pp;
- <определяемый_тип>.pp;
- <класс1>/;
- <под_класс1>.pp;
- <под_класс2>.pp;

- templates/ - содержит используемые шаблоны «*.erb»;
- tests/;

Обращение в коде puppet к классам будет выглядеть как:

```
<имя_модуля> { ... }
<имя_модуля>::<класс1> { ... }
<имя_модуля>::<класс1>::<под_класс1> { ... }
```

Для того, чтобы модуль вообще заработал достаточно иметь папку с именем модуля, папку с манифестами и начальным манифестом «init.pp».

Требуется, чтобы в каждом из файлов <класс1> ... <под_класс1> был один и только один класс с соответствующим именем: <имя_модуля>::<класс1> ... <имя_модуля>::<класс1>::<под_класс1>. При этом файл init.pp должен содержать класс с именем <имя_модуля>.

Получить доступ к файлам модуля можно, например, по адресу puppet:///modules/<имя_модуля>/.

Далее создайте файл узла /etc/puppet/code/environments/production/manifests /site.pp со следующим содержимым:

```
node 'meercat02.example.com' {
include install_pkg
}
```

Замените «meercat02.example.com» на актуальное имя вашего Puppet клиента. Используйте для имени ноды ключевое слово default, если необходимо применить настройки для всех узлов.

Финальным шагом для этого простого Puppet сервера является перезапуск сервиса:

```
systemctl restart /etc/init.d/puppetmaster
```

Теперь на Puppet сервере все настроено и время настроить клиента.

Сначала настроим сервис Puppet агента для запуска.

Отредактируйте /etc/default/puppet, заменив значение «START» на «yes»:

```
START=yes
```

Далее запустите сервис:

```
systemctl start /etc/init.d/puppet
```

Возвращаемся на Puppet сервер для подписи клиентского сертификата с помощью команды:

```
puppetca --sign meercat02.example.com
```

Проверьте /var/log/syslog на любые ошибки конфигурации. Если все прошло хорошо, пакет Apache HTTP Server и его зависимости будут установлены на

Puppet клиенте.

Система безопасности РЕД ОС

6. Система безопасности РЕД ОС

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/sudo-<версия>/sample.sudoers.

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:

```
iptables <-t table> command <match> <target/jump>
```

Не обязательно 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, подпадает 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».

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.

Важно! В действительности поле MARK не изменяется, но в памяти ядра заводится структура, которая сопровождает данный пакет все время его прохождения через машину так, что другие правила и приложения на данной машине (и только на данной машине) могут использовать это поле в своих целях. Таблица имеет две цепочки PREROUTING и OUTPUT. ■

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 INPUT`	Добавляет новое правило в конец заданной цепочки.
`-D, -delete`	`iptables -D INPUT, -dport 80 -j DROP, iptables -D INPUT 1`	Удаление правила из цепочки. Команда имеет два формата записи, первый - когда задаётся критерий сравнения с опцией -D (см. первый пример), второй - порядковый номер правила. 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`	`iptables -F INPUT`	Удаление всех правил из заданной цепочки (таблицы). Если имя цепочки и таблицы не указывается, то удаляются все правила, во всех цепочках.

Команда	Формат вызова	Результат
`-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
```

или, что то же самое:

```
iptables -help
```

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

6.2.4 Ключи утилиты iptables

Ключ	Пример	Пояснения
-v, -verbose	-list, -append, -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	-list	Для всех чисел в выходных данных выводятся их точные значения без округления и без применения множителей K, M, G.
-n, -numeric	-list	Iptables выводит IP-адреса и номера портов в числовом виде, предотвращая попытки преобразовать их в символические имена.
-line-number	-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.
MASQUER ADE	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.

Действие	Пояснения
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 адрес места назначения. Точно так же может определять как 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.

Критерий	Пояснения
`-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 кот«zорого 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 так же 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.

Критерий	Пояснения
-state	Для использования данного критенря 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.

6.2.7 Использование фильтра iptables

РЕД ОС уже включает в себя предустановленный iptables. Для его настройки рекомендуется использовать возможности системы настройки сети /etc/netconfig и /etc/networks.

6.3 Аудит в РЕД ОС

6.3.1 Файлы и утилиты аудита РЕД ОС

Набор утилит управления РЕД ОС находится в главном меню - «Центр управления». Большинство утилит управления и конфигурирования РЕД ОС требуют привилегий администратора РЕД ОС.

Описание auditd.conf

В файле /etc/audit/auditd.conf определяются параметры службы аудита. На одной строке может быть не больше одной директивы. Директива состоит из ключевого слова (названия параметра), знака равенства и соответствующих ему данных (значения параметра). Допустимые ключевые слова приведены в таблице.

Ключ	Значение ключа
<log_file>	Полное имя файла, в который следует записывать протокол.
<log_format>	Оформление данных в протоколе. Допустимы два значения: raw и nolog. При указании RAW, данные будут записываться в том виде, в котором они получаются от ядра. Значение NOLOG отключает запись данных об аудите. Этот параметр не влияет на обработку данных диспетчером событий системы аудита.
<priority_boost>	Неотрицательное число, определяющее повышение приоритета выполнения службы аудита. Значение по умолчанию: 3. Для того чтобы не изменять приоритет, укажите 0.
<flush>	Стратегия работы с дисковым буфером. Допустимые значения: none, incremental, data и sync. Вариант none, отключает какие-либо дополнительные действия со стороны службы по синхронизации буфера с диском. При значении incremental запросы на перенос данных из буфера на диск выполняются с частотой, задаваемой параметром freq. При значении data данные файла синхронизируются немедленно. Значение sync указывает на необходимость немедленной синхронизации как данных, так и метаданных файла при записи на диск.
<freq>	Максимальное число записей протокола, которые могут храниться в буфере. При достижении этого числа производится запись буферированных данных на диск. Данный параметр допустим, только если flush имеет значение incremental.
<num_logs>	Максимальное число файлов с протоколами. Используется, если параметр <max_log_file_action> имеет значение rotate. Если указано число меньше 2, при достижении ограничения на размер файла он обнуляется. Значение параметра не должно превышать 99. Значение по умолчанию: 0. При указании большого числа может потребоваться увеличить ограничение на количество ожидающих запросов. Это можно сделать в файле / etc/audit/audit.rules.
<dispatcher>	Диспетчер - программа, которой (на стандартный ввод) будут передаваться копии сообщений о событиях аудита. Она запускается (с правами администратора) службой аудита при загрузке последней.

Ключ	Значение ключа
<disp_qos>	Разрешить ли блокирование при взаимодействии с диспетчером. Для передачи информации диспетчеру используется буфер размером 128 кб. Это значение является оптимальным для большинства случаев. Если блокирование запрещено (lossy), то все сообщения, поступающие при полном буфере, не будут доходить до диспетчера (записи о них по-прежнему будут вноситься в файл на диске, если только <log_format> не равно nolog). В случае, если блокирование разрешено (lossless), служба аудита будет ожидать появления свободного места в очереди, передавать сообщение диспетчеру и только потом записывать его на диск. Допустимые значения: lossy и lossless. Значение по умолчанию - lossy.
<max_log_file>	Ограничение на размер файла протокола в мегабайтах. Действие, выполняемое при достижении размера файла указанного значения, можно настроить с помощью следующего параметра.
<max_log_file_action>	Действие, предпринимаемое при достижении размером файла протокола максимального значения. Допустимые значения: ignore, syslog, suspend, rotate и keep_logs. Вариант ignore, отключает контроль над размером файла. При значении syslog в системный протокол будет внесено соответствующее сообщение. При значении suspend дальнейшее ведение протокола будет прекращено. Служба по-прежнему будет работать. При значении rotate текущий файл будет переименован и для протокола будет создан новый файл. Имя предыдущего протокола будет дополнено числом 1, а номера других протоколов (если они имеются) будут увеличены на единицу. Таким образом, чем больше номер у протокола, тем он старше. Максимальное число файлов определяется параметром <num_logs> (естественно, соответствие ему достигается за счёт удаления самых старых протоколов). Такое поведение аналогично поведению утилиты logrotate. Вариант keep_logs аналогичен предыдущему, но число файлов не ограничено.
<action_mail_acct>	Адрес электронной почты. Значение по умолчанию: root. Если адрес не локальный по отношению к данной системе, необходимо чтобы в ней был настроен механизм отправки почты. В частности, требуется наличие программы /usr/lib/sendmail.
<space_left>	Минимум свободного пространства в мегабайтах, при достижении которого должно выполняться действие, определяемое следующим параметром.

Ключ	Значение ключа
<space_left_action>	Действие, предпринимаемое при достижении объёмом свободного пространства на диске указанного минимума. Допустимые значения - ignore, syslog, email, exec, suspend, single и halt. При значении ignore, никаких действий не производится. При значении syslog в системный протокол добавляется соответствующая запись. При значении email по адресу, указанному в <action_mail_acct>, отправляется уведомление. При значении exec <путь_к_программе> запускается программа по указанному пути. Передача параметров не поддерживается. При значении suspend служба аудита прекратит вести протокол событий на диске, но будет продолжать работать. Указание single приведёт к переводу компьютера в однопользовательский режим. Указание halt приведёт к выключению компьютера.
<admin_space_left>	Критический минимум свободного пространства в мегабайтах, при достижении которого должно выполняться действие, определяемое следующим параметром. Данное действие следует рассматривать как последнюю меру, предпринимаемую перед тем, как закончится место на диске. Значение настоящего параметра должно быть меньше значения <space_left>.
<admin_space_left_action>	Действие, предпринимаемое при достижении объёмом свободного пространства на диске указанного критического минимума. Допустимые значения - ignore, syslog, email, exec, suspend, single и halt. При значении ignore, никаких действий не производится. При значении syslog в системный протокол добавляется соответствующая запись. При значении email по адресу, указанному в <action_mail_acct>, отправляется уведомление. При значении exec <путь_к_программе> запускается программа по указанному пути. Передача параметров не поддерживается. При значении suspend служба аудита прекратит вести протокол событий на диске, но будет продолжать работать. Указание single приведёт к переводу компьютера в однопользовательский режим. Указание halt приведёт к выключению компьютера.

Ключ	Значение ключа
<disk_full_action>	Действие, предпринимаемое при обнаружении отсутствия свободного пространства на диске. Допустимые значения - ignore, syslog, email, exec, suspend, single и halt. При значении ignore, никаких действий не производится. При значении syslog в системный протокол добавляется соответствующая запись. При значении email по адресу, указанному в <action_mail_acct>, отправляется уведомление. При значении exec </некоторый_путь> запускается сценарий по указанному пути. Передача параметров сценарию не поддерживается. При значении suspend служба аудита прекратит вести протокол событий на диске, но будет продолжать работать. Указание single приведёт к переводу компьютера в однопользовательский режим. Указание halt приведёт к выключению компьютера.
<disk_error_action>	Действие, предпринимаемое при возникновении ошибки в работе с диском. Допустимые значения - ignore, syslog, email, exec, suspend, single и halt. При значении ignore, никаких действий не производится. При значении syslog в системный протокол добавляется соответствующая запись. При значении email по адресу, указанному в <action_mail_acct>, отправляется уведомление. При значении exec </некоторый_путь> запускается сценарий по указанному пути. Передача параметров сценарию не поддерживается. При значении suspend служба аудита прекратит вести протокол событий на диске, но будет продолжать работать. Указание single приведёт к переводу компьютера в однопользовательский режим. Указание halt приведёт к выключению компьютера.

Для обеспечения полного использования раздела параметрам <max_log_file> и <num_logs> следует присвоить соответствующие значения. Учитывайте, что чем больше файлов создаётся на диске (и соответственно переименовывается), тем больше времени будет уходить на обработку событий при достижении размером очередного файла максимума. Параметру <max_log_file_action> рекомендуется присвоить значение keep_logs.

Значение <space_left> должно быть таким, которое позволит администратору вовремя среагировать на предупреждение. Обычно в число действий выполняемых администратором входит запуск aureport -t и архивирование самых старых протоколов. Значение <space_left> зависит от системы, в частности от частоты поступления сообщений о событиях. Значение <space_left_action> рекомендуется установить в email. Если требуется отправка сообщения snmp trap, укажите вариант exec.

Установите значение <admin_space_left> таким образом, чтобы хватило

свободного места для сохранения записей о последующих действиях администратора. Значение параметра <admin_space_left_action> следует установить в single, ограничив, таким образом, способы взаимодействия с системой консолью.

Действие, указанное в <disk_full_action>, выполняется, когда в разделе уже не осталось свободного места. Доступ к ресурсам машины должен быть полностью прекращён, т.к. нет возможности контролировать работу системы. Это можно сделать, указав значение single или halt.

Значение <disk_error_action> следует установить в syslog, single, либо halt в зависимости от соглашения относительно обращения со сбойным аппаратным обеспечением.

Служба аудита - auditd

Служба auditd – это прикладной компонент системы аудита. Он ведёт протокол аудита на диске. Для просмотра протоколов предназначены команды ausearch и aureport. Команда auditctl позволяет настраивать правила аудита. Кроме того, при загрузке загружаются правила из файла /etc/audit/auditd.rules. Некоторые параметры самой службы можно изменить в файле auditd.conf.

Синтаксис:

```
auditd <-f> <-l> <-n>
```

Основные опции службы auditd приведены в таблице.

Опция	Описание опции
`-f`	не переходить в фоновый режим (для отладки). Сообщения программы будут направляться в стандартный вывод для ошибок (stderr), а не в файл.
`-l`	включить следование по символическим ссылкам при поиске конфигурационных файлов.
`-n`	не создавать дочерний процесс. Для запуска из inittab.

Основные сигналы, используемые службой auditd, описаны в таблице.

Сигнал	Описание сигнала
`SIGHUP`	Перезагрузить конфигурацию - загрузить файл конфигурации с диска. Если в файле не окажется синтаксических ошибок, внесенные в него изменения вступят в силу. При этом в протокол будет добавлена запись о событии DAEMON_CONFIG. В противном случае действия службы будут зависеть от параметров <space_left_action>, <admin_space_left_action>, <disk_full_action>, <disk_error_action> файла auditd.conf.
`SIGTERM`	прекратить обработку событий аудита и завершить работу, о чём предварительно занести запись в протокол.

Сигнал	Описание сигнала
SIGUSR1	создать новый файл для протокола, перенумеровав старые файлы или удалив часть из них, в зависимости от параметра <max_log_size_action>.

Важно! Для того чтобы сделать возможным аудит всех процессов, запущенных до службы аудита, добавьте в строку параметров ядра (в конфигурации загрузчика) audit=1. В противном случае аудит некоторых процессов будет невозможен. ■

Утилита auditctl

Утилита auditctl используется для контроля поведения, получения состояния и добавления/удаления правил аудита, предоставляемого ядром.

Основные опции утилиты auditctl приведены в таблице.

Опция	Значение опции
-b backlog	Установить максимальное количество доступных для аудита буферов, ожидающих обработки (значение в ядре по умолчанию - 64). Если все буфера заняты, то флаг сбоя будет выставлен ядром для его дальнейшей обработки.
-e [0..2]	Установить флаг блокировки. «0» позволит на время отключить аудит, включить его обратно можно, передав «1» как параметр. Если установлено значение опции «2», то защитить конфигурацию аудита от изменений. Каждый, кто захочет воспользоваться этой возможностью, может поставить эту команду последней в audit.rules. После этой команды все попытки изменить конфигурацию будут отвергнуты с уведомлением в журналах аудита. В этом случае, чтобы задействовать новую конфигурацию аудита, необходимо перезагрузить систему аудита.
-f [0..2]	Установить способ обработки для флага сбоя. 0=silent, 1=printk, 2=panic. Эта опция позволяет определить, каким образом ядро будет обрабатывать критические ошибки. Например, флаг сбоя выставляется при следующих условиях: ошибки передачи в пространство службы аудита, превышение лимита буферов, ожидающих обработки, выход за пределы памяти ядра, превышение лимита скорости выдачи сообщений. Значение по умолчанию: «1». Для систем с повышенными требованиями к безопасности, значение «2» может быть более предпочтительно.
-h	Краткая помощь по аргументам командной строки.
-i	Игнорировать ошибки при чтении правил из файла.
-l	Вывести список всех правил по одному правилу в строке.

Опция	Значение опции
`-k <ключ>`	Установить на правило ключ фильтрации. Ключ фильтрации - это произвольная текстовая строка длиной не больше 31 символа. Ключ помогает уникально идентифицировать записи, генерируемые в ходе аудита за точкой наблюдения.
`-m <текст>`	Послать в систему аудита пользовательское сообщение. Это может быть сделано только из-под учётной записи root.
`-p [r|w|x|a]`	Установить фильтр прав доступа для точки наблюдения. r=чтение, w=запись, x=исполнение, a=изменение атрибута. Не путайте эти права доступа с обычными правами доступа к файлу - они определяют типы системных вызовов, которые выполняют данные действия. Заметьте, системные вызовы read и write не включены в этот набор, поскольку логи аудита были бы перегружены информацией о работе этих вызовов.
`-r <частота>`	Установить ограничение скорости выдачи сообщений в секунду (0 - нет ограничения). Если эта частота не нулевая и она превышается в ходе аудита, флаг сбоя выставляется ядром для выполнения соответствующего действия. Значение по умолчанию: 0.
`-R <файл>`	Читать правила из файла. Правила должны быть расположены по одному в строке и в том порядке, в каком они должны исполняться. Следующие ограничения накладываются на файл: владельцем должен быть root и доступ на чтение должен быть только у него. Файл может содержать комментарии, начинающиеся с символа «#». Правила, расположенные в файле, идентичны тем, что набираются в командной строке, без указания «auditctl».
`-s`	Получить статус аудита.
`-a <список>, <действие>`	Добавить правило с указанным действием к концу списка. Заметьте, что запятая разделяет эти два значения. Отсутствие запятой вызовет ошибку. Ниже описаны имена доступных списков.
`task`	Добавить правило к списку, отвечающему за процессы. Этот список правил используется только во время создания процесса - когда родительский процесс вызывает fork() или clone(). При использовании этого списка вы можете использовать только те поля, которые известны во время создания процесса: uid, gid и т.д.
`entry`	Добавить правило к списку, отвечающему за точки входа системных вызовов. Этот список применяется, когда необходимо создать событие для аудита, привязанное к точкам входа системных вызовов.

Опция	Значение опции
`exit`	Добавить правило к списку, отвечающему за точки выхода из системных вызовов. Этот список применяется, когда необходимо создать событие для аудита, привязанное к точкам выхода из системных вызовов.
`user`	Добавить правило, отвечающее за список фильтрации пользовательских сообщений. Этот список используется ядром, чтобы отфильтровать события, приходящие из пользовательского пространства, перед тем как они будут переданы службе аудита. Необходимо отметить, что только следующие поля могут быть использованы: uid, auid, gid и pid. Все остальные поля будут обработаны, как если бы они не совпали.
`exclude`	Добавить правило к списку, отвечающему за фильтрацию событий определённого типа. Этот список используется, чтобы отфильтровывать ненужные события. Например, если вы не хотите видеть avc сообщения, вы должны использовать этот список. Тип сообщения задаётся в поле msgtype.

В нижеприведенной таблице описаны доступные действия для правил:

Действие	Описание действия
`never`	Аудит не будет генерировать никаких записей. Это может быть использовано для подавления генерации событий. Обычно необходимо подавлять генерацию вверху списка, а не внизу, т.к. событие инициируется на первом совпавшем правиле.
`always`	Установить контекст аудита. Всегда заполнять его во время входа в системный вызов и всегда генерировать запись во время выхода из системного вызова.
`-A <список>, <действие>`	Добавить правило с указанным действием в начало списка.
`-d <список>, <действие>`	Удалить правило с указанным действием из списка. Правило удаляется только в том случае, если полностью совпали и имя системного вызова и поля сравнения.
`-D`	Удалить все правила и точки наблюдения.
`-S <имя или номер системного вызова>`	Любой номер или имя системного вызова может быть использован. Также возможно использование ключевого слова all. Если какой-либо процесс выполняет указанный системный вызов, то аудит генерирует соответствующую запись. Если значения полей сравнения заданы, а системный вызов не указан, правило будет применяться ко всем системным вызовам. В одном правиле может быть задано несколько системных вызовов - это положительно сказывается на производительности, поскольку заменяет обработку нескольких правил.

Действие	Описание действия
`-F [n=v | n!=v | n<v | n>v | n<=v | n>=v | n&v | n&=v]`	Задать поле сравнения для правила. Атрибуты поля следующие: объект, операция, значение. Вы можете задать до 64 полей сравнения в одной команде. Каждое новое поле должно начинаться с -F. Аудит будет генерировать запись, если произошло совпадение по всем полями сравнения. Допустимо использование одного из следующих 8 операторов: равно, не равно, меньше, больше, меньше либо равно, больше либо равно, битовая маска (n&v) и битовая проверка (n&=v). Битовая проверка выполняет операцию «and» над значениями и проверяет, равны ли они. Битовая маска просто выполняет операцию «and». Поля, оперирующие с идентификатором пользователя, могут также работать с именем пользователя - программа автоматически получит идентификатор пользователя из его имени. То же самое можно сказать и про имя группы.

Поля сравнения могут быть заданы для следующих объектов: четыре первых аргумента, переданных системному вызову. Строковые аргументы не поддерживаются. Это связано с тем, что ядро должно получать указатель на строку, а проверка поля по значению адреса указателя не желательна. Таким образом, необходимо использовать только цифровые значения.

Поле сравнения	Описание
`arch`	Архитектура процессора, на котором выполняется системный вызов. Используйте «uname -m», чтобы определить архитектуру. Если вы не знаете архитектуру вашей машины, но хотите использовать таблицу 32-х битных системных вызовов, и ваша машина поддерживает 32 бита, вы можете использовать x32. Подобно этому x64 может быть использовано для использования таблицы 64-х битных системных вызовов.
`auid`	Это аббревиатура: audit uid - идентификатор пользователя, использованный для входа в систему.
`devmajor`	Главный номер устройства (Device Major Number).
`devminor`	Вспомогательный номер устройства (Device Minor Number).
`egid`	Действительный идентификатор группы.
`euid`	Действительный идентификатор пользователя.
`exit`	Значение, возвращаемое системным вызовом при выходе.
`fsgid`	Идентификатор группы, применяемый к файловой системе.
`fsuid`	Идентификатор пользователя, применяемый к файловой системе.
`gid`	Идентификатор группы.

Поле сравнения	Описание
inode	Номер.
inode key	Альтернативный способ установить ключ фильтрации. Смотри выше описание опции «-k».
msgtype	Используется для проверки совпадения с числом, описывающим тип сообщения. Может быть использован только в списке exclude.
<obj_user>	Имя пользователя-владельца ресурса (в контексте SELinux).
<obj_role>	Роль ресурса (в контексте SELinux).
<obj_type>	Тип ресурса (в контексте SELinux).
<obj_lev_low>	Нижний уровень ресурса (в контексте SELinux).
<obj_lev_high>	Верхний уровень ресурса (в контексте SELinux).
path	Полный путь к файлу для точки наблюдения. Смотри ниже описание опции «-w». Может быть использован только в списке exit.
perm	Фильтр прав доступа для файловых операций. Смотри выше описание опции «-p». Может быть использован только в списке exit.
pers	Персональный номер операционной системы.
pid	Идентификатор процесса.
ppid	Идентификатор родительского процесса.
<subj_user>	Имя пользователя-владельца процесса (в контексте SELinux).
<subj_role>	Роль процесса (в контексте SELinux).
<subj_type>	Тип процесса (в контексте SELinux).
<subj_sen>	Чувствительность процесса (в контексте SELinux).
<subj_clr>	Допуск процесса (в контексте SELinux).
sgid	Установленный идентификатор группы.
success	Если значение, возвращаемое системным вызовом, больше либо равно 0, данный объект будет равен «true/yes», иначе «false/no». При создании правила используйте 1 вместо «true/yes» и 0 вместо «false/no».
suid	Установленный идентификатор пользователя.
uid	Идентификатор пользователя.

Поле сравнения	Описание
-w <путь>	Добавить точку наблюдения за файловым объектом, находящимся по указанному пути. Вы не можете добавлять точку наблюдения к каталогу верхнего уровня - это запрещено ядром. Групповые символы (wildcards) также не могут быть использованы, попытки их использования будут генерировать предупреждающее сообщение. Внутренне точки наблюдения реализованы как слежение за inode. Таким образом, если вы установите точку наблюдения за каталогом, вы увидите файловые события, которые в действительности будут означать обновления метаданных этой inode, и вы можете не увидеть событий, непосредственно связанных с файлами. Если вам необходимо следить за всеми файлами в каталоге, рекомендуется создавать индивидуальную точку наблюдения для каждого файла. В противоположность к правилам аудита системных вызовов, точки наблюдения не оказывают влияния на производительность, связанную с количеством правил посылаемых в ядро.
-W <путь>	Удалить точку наблюдения за файловым объектом, находящимся по указанному пути.

Примеры:

- Увидеть все системные вызовы, используемые определенным процессом:

```
auditctl -a entry,always -S all -F pid=1005
```

- Увидеть все файлы, открытые определенным пользователем:

```
auditctl -a exit,always -S open -F auid=510
```

- Увидеть неудачные попытки вызова системной функции «open»:

```
auditctl -a exit,always -S open -F success!=0
```

Утилита aureport

Утилита aureport - это инструмент, который генерирует итоговые отчёты на основе логов службы аудита. aureport может также принимать данные со стандартного ввода (stdin) до тех пор, пока на входе будут необработанные данные логов.

В шапке каждого отчёта для каждого столбца есть заголовок - это облегчает понимание данных. Все отчёты, кроме основного итогового отчёта, содержат номера событий аудита. Используя их, вы можете найти полные данные о событии с помощью «ausearch -a <номер_события>».

Если в отчёте слишком много данных, можно задать время начала и время окончания для уточнения временного промежутка. Отчёты, генерируемые

aureport, могут быть использованы как исходный материал для получения более развёрнутых отчётов.

Основные опции, используемые утилитой aureport, приведены в таблице.

Опция	Значение опции
`-au, --auth`	Отчёт о всех попытках аутентификации.
`-a, --avc`	Отчёт о всех avc сообщениях.
`-c, --config`	Отчёт о изменениях конфигурации.
`-cr, --crypto`	Отчёт о событиях, связанных с шифрованием.
`-e, --event`	Отчёт о событиях.
`-f, --file`	Отчёт о файлах.
`--failed`	Для обработки в отчётах выбирать только неудачные события. По умолчанию показываются и удачные и неудачные события.
`-h, --host`	Отчёт о хостах.
`-i, --interpret`	Транслировать числовые значения в текстовые. Например, идентификатор пользователя будет оттранслирован в имя пользователя. Трансляция выполняется с использованием данных с той машины, где запущен aureport. Т.е. если вы переименовали учётные записи пользователей или не имеете таких же учётных записей на вашей машине, то вы можете получить результаты, вводящие в заблуждение.
`-if, --input <файл>`	Использовать указанный файл вместо логов аудита. Это может быть полезно при анализе логов с другой машины или при анализе частично сохранённых логов.
`-l, --login`	Отчёт о попытках входа в систему.
`-m, --mods`	Отчёт об изменениях пользовательских учётных записей.
`-ma, --mac`	Отчёт о событиях в системе, обеспечивающей мандатное управление доступом - Mandatory Access Control (MAC).
`-p, --pid`	Отчёт о процессах.
`-r, --response`	Отчёт о реакциях на аномальные события.
`-s, --syscall`	Отчёты о системных вызовах.
`--success`	Для обработки в отчётах выбирать только удачные события. По умолчанию показываются и удачные и неудачные события.
`--summary`	Генерировать итоговый отчёт, который даёт информацию только о количестве элементов в том или ином отчёте. Такой режим есть не у всех отчётов.
`-t, --log`	Этот параметр генерирует отчёт о временных рамках каждого отчёта.

Опция	Значение опции
`-te, --end <дата> <время>`	Искать события, которые произошли раньше (или во время) указанной временной точки. Формат даты и времени зависит от ваших региональных настроек. Если дата не указана, то подразумевается текущий день (today). Если не указано время, то подразумевается текущий момент (now). Используйте 24-часовую нотацию времени, а не AM/PM. Например, дата может быть задана как 10/24/2005, а время - как 18:00:00. Вы можете также использовать ключевые слова: now, recent, today, yesterday, this-week, this-month, this-year. today означает первую секунду после полуночи текущего дня. recent - 10 минут назад. yesterday - первую секунду после полуночи предыдущего дня. this-week означает первую секунду после полуночи первого дня текущей недели, первый день недели определяется из ваших региональных настроек. this-month означает первую секунду после полуночи первого числа текущего месяца. this-year означает первую секунду после полуночи первого числа первого месяца текущего года.
`-tm, --terminal`	Отчёт о терминалах.
`-ts, --start <дата> <время>`	Искать события, которые произошли после (или во время) указанной временной точки. Формат даты и времени зависит от ваших региональных настроек. Если дата не указана, то подразумевается текущий день (today). Если не указано время, то подразумевается полночь (midnight). Используйте 24-часовую нотацию времени, а не AM/PM. Например, дата может быть задана как 10/24/2005, а время - как 18:00:00. Вы можете также использовать ключевые слова: now, recent, today, yesterday, this-week, this-month, this-year. today означает первую секунду после полуночи текущего дня. recent - 10 минут назад. yesterday - первую секунду после полуночи предыдущего дня. this-week означает первую секунду после полуночи первого дня текущей недели, первый день недели определяется из ваших региональных настроек. this-month означает первую секунду после полуночи первого числа текущего месяца. this-year означает первую секунду после полуночи первого числа первого месяца текущего года.
`-u, --user`	Отчёт о пользователях.
`-v, --version`	Вывести версию программы и выйти.

Утилита ausearch

Программа ausearch является инструментом поиска по журналу аудита. Утилита ausearch может также принимать данные со стандартного ввода (stdin) до тех пор, пока на входе будут необработанные данные логов. Все условия, указанные в параметрах, объединяются логическим «И». К примеру, при указании

«-m» и «-ui» в качестве параметров будут показаны события, соответствующие заданному типу и идентификатору пользователя.

Стоит отметить, что каждый системный вызов ядра из пользовательского пространства и возвращение данных в пользовательское пространство имеет один уникальный (для каждого системного вызова) идентификатор события.

Различные части ядра могут добавлять дополнительные записи. Например, в событие аудита для системного вызова «open» добавляется запись PATH с именем файла. ausearch показывает все записи события вместе. Это означает, что при запросе определенных записей результат может содержать записи SYSCALL.

Также помните, что не все типы записей содержат указанную информацию. Например, запись PATH не содержит имя узла или loginuid.

Основные опции, используемые утилитой ausearch, приведены в таблице.

Опция	Значение опции
`-a, --event audit-event-id`	Искать события с заданным идентификатором события. Сообщения обычно начинаются примерно так: msg=audit(1116360555.329:2401771). Идентификатор события - это число после «:». Все события аудита, связанные с одним системным вызовом, имеют одинаковый идентификатор.
`-c, --comm comm-name`	Искать события с заданным comm name. comm name - имя исполняемого файла задачи.
`-f, --file file-name`	Искать события с заданным именем файла.
`-ga, --gid-all all-group-id`	Искать события с заданным эффективным или обычным идентификатором группы.
`-ge, --gid-effective effective-group-id`	Искать события с заданным эффективным идентификатором группы или именем группы.
`-gi, --gidgroup-id`	Искать события с заданным идентификатором группы или именем группы.
`-h, --help`	Справка.
`-hn, --host host-name`	Транслировать числовые значения в текстовые. Например, идентификатор пользователя будет оттранслирован в имя пользователя. Трансляция выполняется с использованием данных с той машины, где запущен ausearch. Т.е. если вы переименовали учётные записи пользователей или не имеете таких же учётных записей на вашей машине, то вы можете получить результаты, вводящие в заблуждение.
`-if, --input file-name`	Использовать указанный файл вместо логов аудита. Это может быть полезно при анализе логов с другой машины или при анализе частично сохранённых логов.

Опция	Значение опции
`-k, --key key-string`	Искать события с заданным ключевым словом.
`-m, --message message-type| comma-sep-message-type-list`	Искать события с заданным типом. Вы можете указать список значений, разделённых запятыми. Можно указать несуществующий в событиях тип ALL, который позволяет получить все сообщения системы аудита. Список допустимых типов большой и будет показан, если указать эту опцию без значения. Тип сообщения может быть строкой или числом. В списке значений этого параметра в качестве разделителя используются запятые и пробелы недопустимы.
`-o, --object SE-Linux-context-string`	Искать события с заданным контекстом (объектом).
`-p, --pid process-id`	Искать события с заданным идентификатором процесса.
`-pp, --ppid parent-process-id`	Искать события с заданным идентификатором родительского процесса.
`-r, --raw`	Необработанный вывод. Используется для извлечения записей для дальнейшего анализа.
`-sc, --success syscall-name-or-value`	Искать события с заданным системным вызовом. Вы можете указать его номер или имя. Если вы указали имя, оно будет проверено на машине, где запущен ausearch.
`-se, --context SE-Linux-context-string`	Искать события с заданным контекстом SELinux (stcontext/subject или tcontext/object).
`-su, --subject SE-Linux-context-string`	Искать события с заданным контекстом SELinux - scontext (subject).
`-sv, --success success-value`	Искать события с заданным флагом успешного выполнения. Допустимые значения: «yes» (успешно) и «no» (неудачно).

Опция	Значение опции
`-te, --end [end-date] [end-time]`	Искать события, которые произошли раньше (или во время) указанной временной точки. Формат даты и времени зависит от ваших региональных настроек. Если дата не указана, то подразумевается текущий день (today). Если не указано время, то подразумевается текущий момент (now). Используйте 24-часовую нотацию времени, а не AM/PM. Например, дата может быть задана как 10/24/2005, а время - как 18:00:00. Вы можете также использовать ключевые слова: now, recent, today, yesterday, this-week, this-month, this-year. today означает первую секунду после полуночи текущего дня. recent - 10 минут назад. yesterday - первую секунду после полуночи предыдущего дня. this-week означает первую секунду после полуночи первого дня текущей недели, первый день недели определяется из ваших региональных настроек (см. localtime). this-month означает первую секунду после полуночи первого числа текущего месяца. this-year означает первую секунду после полуночи первого числа первого месяца текущего года.
`-ts, --start [start-date] [start-time]`	Искать события, которые произошли после (или во время) указанной временной точки. Формат даты и времени зависит от ваших региональных настроек. Если дата не указана, то подразумевается текущий день (today). Если не указано время, то подразумевается полночь (midnight). Используйте 24-часовую нотацию времени, а не AM/PM. Например, дата может быть задана как 10/24/2005, а время - как 18:00:00. Вы можете также использовать ключевые слова: now, recent, today, yesterday, this-week, this-month, this-year. today означает первую секунду после полуночи текущего дня. recent - 10 минут назад. yesterday - первую секунду после полуночи предыдущего дня. this-week означает первую секунду после полуночи первого дня текущей недели, первый день недели определяется из ваших региональных настроек. this-month означает первую секунду после полуночи первого числа текущего месяца. this-year означает первую секунду после полуночи первого числа первого месяца текущего года.
`-tm, --terminal terminal`	Искать события с заданным терминалом. Некоторые службы (такие как cron и atd) используют имя службы как имя терминала.
`-ua, --uid-all all-user-id`	Искать события, у которых любой из идентификатора пользователя, эффективного идентификатора пользователя или loginuid (auid) совпадают с заданным идентификатором пользователя.

Опция	Значение опции
-ue, --uid-effective effective-user-id	Искать события с заданным эффективным идентификатором пользователя.
-ui, --uid user-id	Искать события с заданным идентификатором пользователя.
-ul, --loginuid login-id	Искать события с заданным идентификатором пользователя. Все программы, которые его используют, должны использовать pam_loginuid.
-v, --verbose	Показать версию и выйти.
-w, --word	Совпадение с полным словом. Поддерживается для имени файла, имени узла, терминала и контекста SELinux.
-x, --executable executable	Искать события с заданным именем исполняемой программы.

Утилита autrace

Утилита autrace - это программа, которая добавляет правила аудита для того, чтобы следить за использованием системных вызовов в указанном процессе подобно тому, как это делает strace.

После добавления правил она запускает процесс с указанными аргументами. Результаты аудита будут либо в логах аудита (если служба аудита запущена), либо в системных логах.

Внутри autrace устроена так, что удаляет все предыдущие правила аудита перед тем, как запустить указанный процесс и после его завершения. Поэтому, в качестве дополнительной меры предосторожности, программа не запустится, если перед ее использованием правила не будут удалены с помощью audtictl – предупреждающее сообщение известит об этом.

Опции утилиты autrace приведены в таблице.

Опция	Значение опции
-r	Ограничить сбор информации о системных вызовах только теми, которые необходимы для анализа использования ресурсов. Это может быть полезно при моделировании внештатных ситуаций, к тому же позволяет уменьшить нагрузку на логи.

Примеры:

- Обычное использование программы:

```
autrace /bin/ls /tmp
ausearch --start recent -p 2442 -i
```

- Режим ограниченного сбора информации:

```
autrace -r /bin/ls
ausearch --start recent -p 2450 --raw | aureport --file -summary
ausearch --start recent -p 2450 --raw | aureport --host -summary
```

Графическая утилита просмотра журналов аудита

После запуска программы просмотра системных журналов отображается следующее окно (рисунок 6.1).

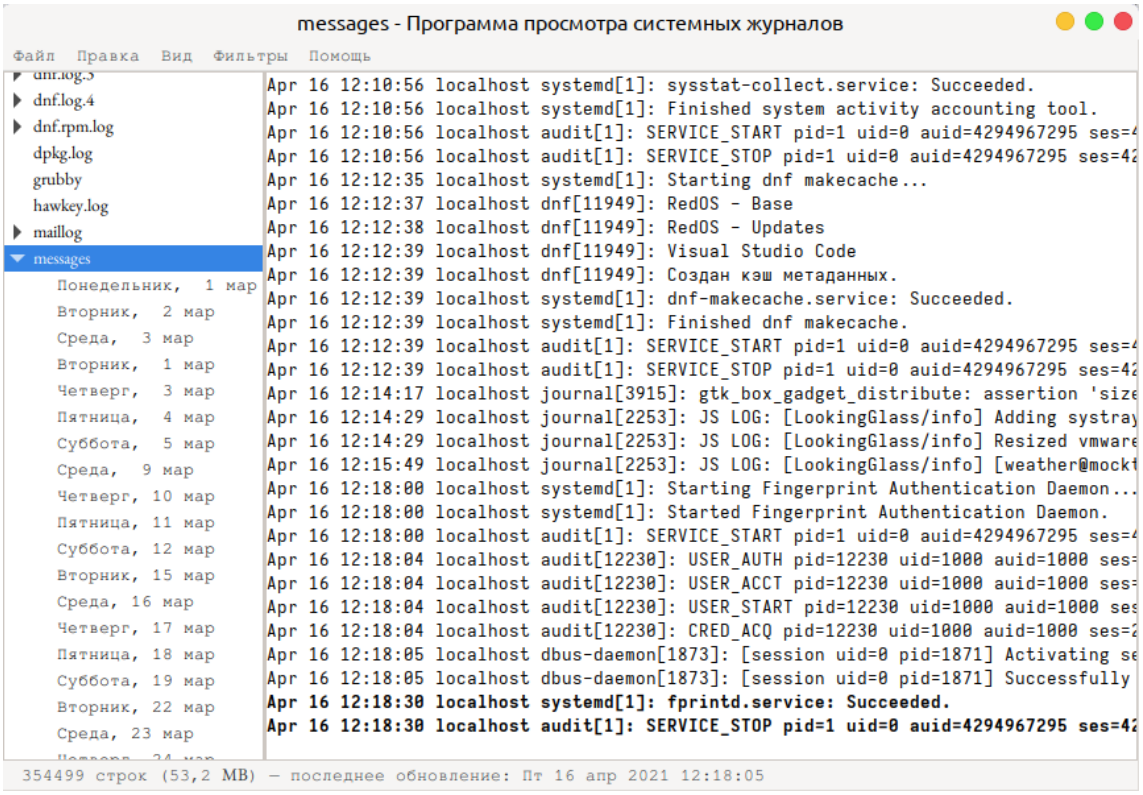

Рисунок 6.1 - Окно программы просмотра системных журналов

Для доступа к журналам аудита пользователь должен ввести пароль администратора. После первого запуска программы просмотра системных журналов приложение по умолчанию отображает несколько файлов журналов (такие, как /var/log/messages). Открытые логи (журналы) представлены в списке в левой части окна приложения. Выбранный из списка журнал отображается в главной части окна приложения.

По умолчанию программа просмотра системных журналов отображает изменения каждого открытого журнала, и любые изменения появляются автоматически в главном окне.

Заметьте, что программа просмотра системных журналов также позволяет вам открывать архивированные журналы (названия файлов которых оканчиваются на «.gz»).

Чтобы скопировать одну или более строк журнала в буфер обмена, просто выделите строки в главной части окна и выберите Правка ▶ Копировать.

Если вы хотите копировать весь журнал в буфер обмена, выберите его полностью, используя настройку Правка ▶ Выделить всё, затем выберите Правка ▶ Копировать.

Вы можете использовать программу просмотра системных журналов для мониторинга логов (журналов). По умолчанию, ко всем журналам, открытым в программе просмотра системных журналов, применяются текущие изменения. Если новые строки добавлены к журналу, просматриваемому с помощью программы просмотра системных журналов, наименование журнала выделится жирным начертанием шрифта в списке журналов. Если журнал отображается в настоящее время в главной части окна, новые линии появятся автоматически в конце журнала и, по истечении пяти секунд, наименование журнала снова вернётся к обычному начертанию в списке журналов.

Информация о журнале обычно отображена в строке состояния, включая:

- количество строк в журнале;
- размер журнала в байтах;
- дату последнего обновления (изменения) журнала.

Строка состояния может быть показана или скрыта через пункт меню Вид ▶ Строка состояния.

Экспорт данных аудита

Для экспорта данных аудита для другого доверенного ИТ-продукта предлагается использовать сжатые архивные файлы с парольной защитой. Архив позволит значительно уменьшить объем передаваемых данных, а пароль защитит от несанкционированного раскрытия при передаче. Для сжатия журналов аудита необходимо использовать утилиту zip.

Zip - это утилита для сжатия и упаковки файлов и каталогов. Чтобы сжать файл или каталог с помощью команды zip, наберите в командной строке:

```
zip -r <имя_файла>.zip <каталог>
```

В этом примере <имя_файла>.zip — создаваемый вами файл, а <каталог> — каталог, который будет помещён в новый zip-файл. Опция «-r» указывает, что все файлы из каталога <каталог> будут включены рекурсивно.

Установить пароль на архив можно с помощью ключа «-P», а ключ «-e» скроет пароль при вводе.

ОС, при соответствующей настройке, формирует ряд журналов аудита. Типы событий безопасности и места размещения соответствующих журналов приведены в таблице.

Событие	Журнал аудита
Запуск и завершение выполнения функций аудита	/var/log/audit/audit.log
Запись аудита для событий, связанных с истечением установленного администратором срока действия пароля	/var/log/secure

Событие	Журнал аудита
Запись аудита для событий, связанных с истечением установленного администратором срока действия идентификатора пользователя ОС (учетной записи)	/var/log/secure
Действия, предпринимаемые в ответ на возможные нарушения безопасности	/var/log/maillog
Все модификации конфигурации аудита, происходящие во время сбора данных аудита	/var/log/audit/audit.log
Блокирование учётной записи в результате превышения максимального числа неуспешных попыток входа систему	/var/log/secure
Все модификации политики аудита	/var/log/audit/audit.log
Все модификации значений атрибутов безопасности, используемых для смены начальной аутентификационной информации пользователя ОС после однократного использования	/var/log/secure
Все модификации аутентификационной информации	/var/log/secure
Полнотекстовая запись привилегированных команд (команд, управляющих системными функциями)	/var/log/audit/audit.log
Применение механизма восстановления информации	/var/log/messages
Изменение настроек механизмов уничтожения (стирания) данных	/var/log/audit/audit.log
Сбои в работе механизма изоляции процессов	/var/log/messages
Попытки установки внешних модулей уровня ядра, не проверенных разработчиком (производителем), или внешних модулей уровня ядра с нарушенной целостностью	/var/log/audit/audit.log
Попытки запуска компонентов программного обеспечения, целостность которых была нарушена, и попытки запуска компонентов программного обеспечения, произведённых в нарушение установленных правил запуска компонентов программного обеспечения	/var/log/audit/audit.log
Чтение информации из записей аудита, неуспешные попытки читать информацию из записей аудита	/var/log/audit/audit.log
Предпринимаемые действия после превышения порога заполнения журнала аудита, предпринимаемые действия при сбое хранения журнала аудита	/var/log/messages
Все запросы на выполнение операций на объекте, на который распространяется политика дискреционного или ролевого доступа	/var/log/audit/audit.log
Все попытки экспортировать информацию	/var/log/audit/audit.log

Событие	Журнал аудита
Все решения по запросам на сетевые потоки при использовании фильтрации сетевых потоков	/var/log/messages
Отклонение или принятие любого проверенного пароля при проверке его на сложность/заданные метрики	/var/log/secure
Все случаи использования механизма аутентификации, результат действия каждого активизированного механизма вместе с итоговым решением	/var/log/secure
Все случаи использования механизма идентификации пользователя, включая представленный идентификатор пользователя	/var/log/secure
Успешное или неуспешное связывание атрибутов безопасности пользователя с субъектом	/var/log/secure
Назначение срока действия для атрибута и действия, предпринятые по истечении назначенного срока.	/var/log/secure; /var/log/audit/audit.log
Модификация группы пользователей	/var/log/audit/audit.log
Использование функций управления	/var/log/messages
Изменения внутреннего представления времени	/var/log/messages
Выполнение и результаты самотестирования	/var/log/messages
Тип сбоя или прерывания обслуживания	/var/log/messages
Все операции ОС, прерванные из-за сбоя	/var/log/messages
Все попытки использования функции распределения ресурсов с учетом приоритетности обслуживания	/var/log/audit/audit.log
Все обращения к функциям распределения ресурсов	/var/log/audit/audit.log
Отклонение нового сеанса, основанное на ограничении числа параллельных сеансов	/var/log/secure
Все попытки разблокирования интерактивного сеанса	/var/log/secure
Завершение интерактивного сеанса механизмом блокирования сеанса	/var/log/messages

Записи журналов аудита имеют следующий вид:

Журнал аудита	Пример записи события безопасности
`/var/log/audit/audit.log`	type=DAEMON_START msg=audit(1505824570.753:4329): op=start ver=2.6.5 format=raw kernel=4.9.30-1.el7.x86_64 auid=4294967295 pid=614 subj=system_u:system_r:auditd_t:s0 res=success type=SERVICE_START msg=audit(1505824570.773:6): pid=1 uid=0 auid=4294967295 ses=4294967295 subj=system_u:system_r:init_t:s0 msg='unit=rngd comm="systemd" exe="/usr/lib/systemd/s
`/var/log/secure`	Oct 11 09:51:01 redos login[4682]: LOGIN ON tty2 BY ivanov Oct 11 09:56:40 redos vlock[5049]: pam_unix(vlock:auth): authentication failure; logname= uid=1001 euid=1001 tty=pts/0 ruser= rhost= user=ivanov
`/var/log/maillog`	Oct 17 10:47:11 redos postfix/cleanup[9004]: E66AE400BE: message-id=<20171017074711.E66AE400BE@redos.localdomain> Oct 17 10:47:11 redos postfix/qmgr[2780]: E66AE400BE: from=<root@redos.local domain>, size=496, nrcpt=1 (queue active)
`/var/log/messages`	Oct 11 09:49:38 redos systemd[1]: Starting Local File Systems (Pre). Oct 11 09:49:38 redos systemd[1]: Mounting /mnt/test_disk... Oct 11 09:49:38 redos kernel: EXT4-fs (sdb1): mounted filesystem with ordered data mode. Opts: (null)

6.4 Права доступа к файлам и каталогам

РЕД ОС — 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 РЕД ОС 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.

```
ls -l /bin/ls
```

```
-rwxr-xr-x 1 root root 49940 Sep 12 1999 /bin/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.

6.4.1 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).

Вот несколько примеров использования команды chmod:

- Предоставление всем пользователям системы прав на выполнение данного файла:

```
chmod a+x <имя_файла>
```

- Удаление права на чтение и запись для всех, кроме владельца файла:

```
chmod go-rw <имя_файла>
```

- Установка всем прав на чтение, запись и выполнение:

```
chmod ugo+rwx <имя_файла>
```

Если опустить указание на то, кому предоставляется данное право, то подразумевается, что речь идёт вообще обо всех пользователях, т. е. вместо:

```
chmod a+x <имя_файла>
```

можно записать просто:

```
chmod +x <имя_файла>
```

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 <имя_файла>
```

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 +s <имя_файла>
```

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:

```
chmod 4775 <имя_файла>
```

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. Вот небольшой пример того, как это будет выглядеть:

```
ls -l prim1
```

```
-rwSrwsrwT 1 kos root 12 Dec 18 23:17 prim1
```

6.4.2 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.

6.4.3 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.

6.4.4 ACL

ACL (Access Control List - Список Контроля Доступа) предоставляет расширенный и более гибкий механизм распределения прав файловых систем. Он предназначен для расширения прав доступа к файлам. ACL позволяет устанавливать разрешения любым пользователям или группам для различных файловых ресурсов.

Чтобы включить ACL, файловая система должна быть смонтирована с опцией «acl». Используйте «fstab» для постоянного монтирования с данной опцией.

Для изменения прав ACL используйте команду «setfacl».

Добавить разрешения для пользователя (<user> здесь имя пользователя или его ID):

```
setfacl -m "u:<user>:permissions" <file/dir>
```

Добавить разрешения для группы (<group> здесь имя группы или её ID):

```
setfacl -m "g:<group>:permissions" <file/dir>
```

Все файлы и каталоги при создании будут наследовать записи ACL родительского каталога:

```
setfacl -dm "entry" <dir>
```

Удалить определённую ACL запись:

```
setfacl -x "entry" <file/dir>
```

Удалить все ACL записи:

```
setfacl -b <file/dir>
```

Посмотреть права ACL:

```
getfacl <file/dir>
```

Удаление всех записей расширения ACL:

```
setfacl -b abc
```

Проверка разрешений:

```
getfacl abc
```

Знак «+»(плюс) в выводе команды «ls -l», который следует за правами Unix, указывает на использование ACL.

```
ls -l /dev/audio
```

```
crw-rw----+ 1 root audio 14, 4 nov. 9 12:49 /dev/audio
```

```
getfacl /dev/audio
```

```
getfacl: Removing leading '/' from absolute path names
# file: dev/audio
# owner: root
# group: audio
user::rw-
user:solstice:rw-
group::rw-
mask::rw-
other::---
```

Поиск файлов с правами ACL:

```
getfacl -R -s -p <dir> | sed -n 's/^# file: //p'
```

Данная команда ищет в указанном каталоге и его подкаталогах все файлы и директории, которые имеют права ACL.

6.4.5 Chattr и lsattr

chattr — изменяет атрибуты файлов на файловых системах ext3, ext4.
Синтаксис:

```
chattr [ -RV ] [ -v <версия> ] [ <атрибуты> ] <файлы>...
```

Основные опции утилиты chattr приведены в таблице.

Опция	Значение опции
-R	Рекурсивно изменять атрибуты каталогов и их содержимого. Все найденные символические ссылки будут игнорироваться
-V	Выводит более полную выводимую информацию и версию программы chattr
-f	Отключить вывод большинства ошибок
-p <проект>	Установить номер проекта
-v <версия>	Установить номер версии/генерации файла

Формат символьного режима:

```
+-=[ASacDdIijsTtu]
```

Оператор «+» обозначает добавление указанных атрибутов к существующим; «-» обозначает их снятие; «=» обозначает установку только этих атрибутов файлам.

Символы «ASacDdijsu» указывают на новые атрибуты файлов:

Символ	Значение символа
a (append only)	Файл может быть открыт только в режиме до записи.
A (no atime updates)	Не обновлять поле atime (время последнего доступа) файла. Уменьшает количество операций записи на устройство.
c (compressed)	Файл записан на диск с использованием сжатия.
C (no copy-on-write)	Отключение режима «Copy-on-write» для указанного файла.
d (no dump)	Отключает создание архивной копии файла программой dump.

Символ	Значение символа
D (synchronous directory updates)	Включает синхронную запись изменений в данном каталоге. Это эквивалентно опции dirsync при монтировании файловой системы.
e (extent format)	Включает использование extent при выделении места на устройстве. Атрибут не может быть отключён с помощью chattr.
E	Атрибут экспериментальных методов сжатия. Атрибут не может быть установлен или снят с помощью chattr.
i (immutable)	Указывает, что файл защищен от изменений: не может быть удалён или переименован, никакая ссылка (жёсткая) не может быть создана на этот файл, никакие данные не могут быть записаны в файл.
I	Указывает что указанный каталог проиндексирован с помощью хеш-дерева.
j (data journalling)	Все данные файла перед записью будут полностью записаны в журнал ext3/ext4, несмотря на опции монтирования «data=ordered» или «data=writeback». В режиме «data=journal» бессмыслен.
P (project hierarchy)	Указывает, что каталог с вложенными файлами является иерархической структурой проекта.
s (secure deletion)	Атрибут защищённого удаления файла, перед удалением все содержимое файла полностью затирается «00».
S (synchronous updates)	Атрибут синхронной записи для данного файла, аналогичен опции монтирования «sync» файловой системы.
t (no tail-merging)	Отключает метод tail-merging для файла.
T (top of directory hierarchy)	Указывает что каталог является головой иерархии каталогов.
u (undeletable)	Указывает системе, что при удалении файла его содержимое должно быть сохранено с возможностью дальнейшего восстановления.

lsattr - выводит атрибуты файла расширенной файловой системы.
Синтаксис:

```
lsattr [ -RVadv ] <файлы>
```

lsattr выводит атрибуты файла, которые могли быть установлены ранее командой chattr.

Основные опции утилиты lsattr приведены в таблице.

Опция	Значение опции
-R	Рекурсивно выводит атрибуты каталогов и их содержимого.
-V	Выводит версию программы.
-a	Выводит информацию по всем файлам в каталогах, включая скрытые файлы, чьи имена начинаются с «.».
-d	Отображает имена каталогов так же, как и остальные обычные файлы (взамен вывода списков их содержимого).
-v	Выводит версию или номер поколения файла.

6.5 Systemd – управление компонентами ОС

Systemd – менеджер системы и сервисов в операционной системе РЕД ОС.

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 серви-

сов, поддерживает этот тип активации и передаёт сокеты этим сервисам сразу 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;

Команда	Описание
`systemctl reload name.service`	перезагрузка конфигурации сервиса;
`systemctl status name.service`	проверка, запущен ли сервис с детальным выводом состояния сервиса;
`systemctl is-active name.service`	проверка, запущен ли сервис с простым ответом: active или inactive;
`systemctl list-units --type service --all`	отображение статуса всех сервисов;
`systemctl enable name.service`	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 РЕД ОС 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 set-default name.target
```

Для изменения текущей цели:

```
systemctl isolate name.target
```

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

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

`systemctl halt` – останавливает систему;
`systemctl poweroff` – выключает систему;
`systemctl reboot` – перезагружает систему.

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.

```
[Unit]
Description=Daemon to detect crashing apps
After=syslog.target
```

```
[Service]
ExecStart=/usr/sbin/abrtd
Type=forking
[Install]
WantedBy=multi-user.target
```

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.

6.6 SELinux

SELinux (Security-Enhanced Linux) обеспечивает усиление защиты путём внесения изменений как на уровне ядра, так и на уровне пространства пользователя. Далее описываются основные принципы, на которых построена работа используемых библиотек SELinux, а также их реализация в РЕД ОС.

6.6.1 Введение

В большинстве операционных систем имеются средства управления доступом, которые определяют, может ли определённый объект (пользователь или программа) получить доступ к определённому ресурсу. В РЕД ОС применяется разграничительный контроль доступа (discretionary access control, DAC). Этот метод позволяет ограничить доступ к объектам на основе групп, к которым они принадлежат. Например, для каждого файла определены владелец, группа, а также указаны права доступа к этому файлу. Правами доступа определяется, кто может получить доступ к файлу, кто может открыть его для чтения, кто может внести в него изменения, кто может запустить этот файл на выполнение. Права доступа определены для трёх категорий: пользователь (владелец файла), группа (все пользователи, которые являются членами группы) и другие (все пользователи, которые не являются ни владельцем файла, ни членами группы).

Другим методом управления доступом является управление доступом на основе ролей (role-based access control, RBAC). При использовании RBAC права доступа предоставляются на основе ролей, выдаваемых системой безопасности. Отличие концепции ролей от традиционных групп состоит в том, что группа представляет одного или нескольких пользователей, в то время как роль, хотя она также может быть применена к нескольким пользователям, представляет совокупность полномочий на выполнение определенных действий.

Используемые библиотеки SELinux добавляют в операционную систему поддержку RBAC.

6.6.2 Установка

Для установки SELinux требуется установить пакет selinux-policy. Это можно сделать, например, воспользовавшись командной строкой с привилегиями root пользователя:

```
dnf install selinux-policy
```

После перезагрузки SELinux проиндексирует содержимое жёсткого диска, это может занять некоторое время.

Для настройки SELinux можно использовать различные текстовые редакторы, чтобы настроить вручную его файл конфигурации /etc/selinux/config.

6.6.3 Утилиты

Утилита audit2allow

Утилита audit2allow создает разрешающие правила политики SELinux из файлов журналов, содержащих сообщения о запрете операций.

Эта утилита сканирует журналы в поиске сообщений, появляющихся, когда система не дает разрешения на операцию. Далее утилита генерирует ряд правил, которые, будучи загруженными в политику, могли бы разрешить эти операции. Однако, данная утилита генерирует только разрешающие правила Type Enforcement (TE). Некоторые отказы в использовании разрешений могут потребовать других изменений политики. Например, добавление атрибута в определение типа, для разрешения существующего ограничения (constraint), добавления разрешающего правила для роли или модификации ограничения (constraint). В случае сомнений для диагностики можно попробовать использовать утилиту audit2why.

Следует с осторожностью работать с выводом данной утилиты, убедившись, что разрешаемые операции не представляют угрозы безопасности. Обычно бывает лучше определить новый домен и/или тип или произвести другие структурные изменения. Лучше избирательно разрешить оптимальный набор операций вместо того, чтобы вслепую применить иногда слишком широкие разрешения, рекомендованные этой утилитой. Некоторые запреты на использование разрешений бывают не принципиальны для приложения. В таких случаях вместо использования разрешительного правила («allow» rule) лучше просто подавить журналирование этих запретов при помощи правила «dontaudit».

Синтаксис:

```
audit2allow [options]
```

Основные опции утилиты audit2allow представлены в таблице.

Опция	Значение опции
`-a | --all`	Прочесть входную информацию из журналов «message» и «audit». Не используется вместе с опцией «-i».
`-d | --dmesg`	Прочесть входную информацию из вывода команды /bin/dmesg. Обратите внимание, что когда работает auditd, не все сообщения аудита доступны через dmesg. Вместо этого используйте «ausearch -m avc | audit2allow» или «-a».
`-f | --fcfile <File Context File>`	Добавить файл контекстов в генерируемый пакет модуля. Требует опцию -M.
`-h | --help`	Вывести краткую справку по использованию.
`-i <inputfile> | --input <inputfile>`	Прочесть входную информацию из <inputfile>.
`-l | --lastreload`	Прочесть только часть входной информации, начиная с момента последней перезагрузки политики.
`-m <modulename> | --module <modulename>`	Генерировать модуль. Требуется вывод <modulename>.
`-M <modulename>`	Генерировать загружаемый пакет модуля. Опция конфликтует с «-o».
`-o <outputfile> | --output <outputfile>`	Дописать вывод в <outputfile>.
`-r | --requires`	Генерировать вывод в синтаксисе загружаемого модуля.
`-R | --reference`	Генерировать эталонную политику (reference policy), используя установленные макросы. Требуется пакет selinux-policy-devel.
`-t | --tefile`	Указывает, что выходной файл является te-файлом (type enforcement). Может использоваться для конвертации старого формата политик в новый формат.
`-v | --verbose`	Включить подробный вывод.

Пример:
Этот пример подходит для системы, использующей пакет audit. Если вы не

> используете пакет audit, сообщения AVC будут появляться в /var/log/messages. В этом случае замените /var/log/audit/audit.log, используемый в примере, на /var/log/messages.

Использование audit2allow для генерации монолитной (не модульной) политики:

```
   cd /etc/selinux/$ SELINUXTYPE/src/policy
   cat /var/log/audit/audit.log | audit2allow >> domains/misc/
local.te
   cat domains/misc/local.te
```

```
   allow cupsd_config_t unconfined_t:fifo_file { getattr ioctl }
```

Просмотрите domains/misc/local.te и измените его:

```
   make load
```

Использование audit2allow для генерации модульной политики:

```
   cat /var/log/audit/audit.log | audit2allow -m local > local.te
   cat local.te
```

```
   module local 1.0;
   require {
   role system_r;
   class fifo_file { getattr ioctl }; type cupsd_config_t; type
unconfined_t;};
   allow cupsd_config_t unconfined_t:fifo_file { getattr ioctl };
```

Просмотрите local.te и измените его.
Создание модуля политики вручную:

- Скомпилируйте модуль:

```
   checkmodule -M -m -o local.mod local.te
```

- Создайте пакет:

```
   semodule_package -o local.pp -m local.mod
```

- Загрузите модуль в ядро:

```
   semodule -i local.pp
```

- Использование audit2allow для генерации и создания модуля политики:

```
   cat /var/log/audit/audit.log | audit2allow -M local
```

Создаётся новый type enforcment файл: local.te.

- Компиляция политики:

```
checkmodule -M -m -o local.mod local.te
```

- Создание пакета:

```
semodule_package -o local.pp -m local.mod
```

Для того чтобы загрузить только что созданный пакет политики в ядро, вам необходимо выполнить команду:

```
semodule -i local.pp
```

Утилита secon

Утилита secon — позволяет просмотреть контекст SELinux для файла, программы или ввода пользователя.

Просматривает часть контекста. Контекст берется из файла, идентификатора процесса, ввода пользователя или контекста, в котором была запущена утилита secon.

Синтаксис:

```
secon [-hVurtscmPRfLp] [CONTEXT] [--file] <FILE> [--link] <FILE>
[--pid] <PID>
```

Основные опции утилиты secon приведены в таблице.

Опция	Значение опции
`-V, --version`	Посмотреть текущую версию secon.
`-h, --help`	Вывести информацию по использованию secon.
`-P, --prompt`	Вывести данные в формате, подходящем для подсказки.
`-u, --user`	Показать пользователя контекста безопасности.
`-r, --role`	Показать роль контекста безопасности.
`-t, --type`	Показать тип контекста безопасности.
`-s, --sensitivity`	Показать уровень чувствительности (sensitivity level) контекста безопасности.
`-c, --clearance`	Показать уровень допуска (clearance level) контекста безопасности.
`-m, --mls-range`	Показать для контекста безопасности в виде диапазона уровень чувствительности (sensitivity level) и уровень допуска (clearance).
`-R, --raw`	Вывести уровень чувствительности (sensitivity level) и уровень допуска (clearance) в формате без трансляции.
`-f, --file`	Получить контекст заданного файла <FILE>.
`-L, --link`	Получить контекст заданного файла <FILE> (не следовать по символическим ссылкам).

Опция	Значение опции
`-p, --pid`	Получить контекст заданного процесса по идентификатору <PID>.
`--pid-exec`	Получить exec контекст заданного процесса по идентификатору <PID>.
`--pid-fs`	Получить fscreate контекст заданного процесса по идентификатору <PID>.
`--current, --self`	Получить контекст из текущего процесса.
`--current-exec --self-exec`	Получить exec контекст из текущего процесса.
`--current-fs, --self-fs`	Получить fscreate контекст из текущего процесса.
`--parent`	Получить контекст из родительского процесса текущего процесса.
`--parent-exec`	Получить exec контекст из родительского процесса текущего процесса.
`--parent-fs`	Получить fscreate контекст из родительского процесса текущего процесса.

Если не было задано опций, то для того, чтобы secon получил контекст из другого источника, может быть задан дополнительный аргумент CONTEXT. Если этим аргументом является знак дефиса (-), то контекст будет прочитан из стандартного ввода. Если аргументов не было заданно, то secon будет пытаться прочитать контекст со стандартного ввода, но только в том случае, если стандартный ввод не терминал (tty). В этом случае secon будет вести себя как будто была передана опция «--self».

Если не задана ни одна опция из «--user», «--role», «--type», «--level» или «--mls-range», то все они будут использованы.

Утилита audit2why

Утилита audit2why — позволяет определить из сообщения аудита SELinux причину запрета доступа.

Эта утилита обрабатывает сообщения аудита SELinux, принятые со стандартного ввода, и сообщает, какой компонент политики вызвал каждый из запретов. Если задана опция «-p», то используется указанная этой опцией политика, в противном случае используется активная политика. Есть три возможные причины запрета доступа:

- отсутствует или отключено разрешительное TE правило (TE allow rule);
- нарушение ограничения (a constraint violation);
- отсутствует разрешительное правило для роли (role allow rule).

В первом случае разрешительное TE правило может присутствовать в политике, но было отключено через булевы атрибуты. Если же разрешительного правила не было, то оно может быть создано при помощи утилиты audit2allow.

Во втором случае могло произойти нарушение ограничения. Просмотрите policy/constraints или policy/mls для того, чтобы определить искомое ограничение. Обычно проблема может быть решена добавлением атрибута типа к домену.

В третьем случае была произведена попытка сменить роль, при том что не существует разрешительного правила для участвующей при этом пары ролей. Проблема может быть решена добавлением в политику разрешительного правила для этой пары ролей.

Синтаксис:

```
audit2why <options>
```

Основные опции утилиты audit2why приведены в таблице.

Опция	Значение опции
`--help`	Вывести короткую подсказку.
`-p <policyfile>`	Указать альтернативный файл политики.

Пример:

```
/usr/sbin/audit2why < /var/log/audit/audit.log
type=KERNEL msg=audit(1115316408.926:336418): avc: denied
{ getattr } for path=/home/ sds dev=hda5 ino=1175041
scontext=root:secadm_r: secadm_t:s0-s9:c0.c127
tcontext=user_u:object_r:user_home_dir_t:s0 tclass=dir
```

Причиной является: Отсутствие или отключение разрешительного TE правила. Разрешительное TE правило может присутствовать в политике, но было отключено через булевы переключатели; проверьте булевы переключатели. Вы можете посмотреть необходимые разрешительные правила, запустив audit2allow и подав это сообщение аудита на ввод.

```
type=KERNEL msg=audit(1115320071.648:606858): avc: denied
{ append } for name=.bash_history dev=hda5 ino=1175047
scontext=user_u:user_r:user_t:s1-s9:c0.c127 tcontext=user_u:
object_r:user_home_t:s0 tclass=file
```

Причиной является: Нарушение ограничения. Проверьте policy/constraints. Обычно, для разрешения ограничения необходимо добавить в домен атрибут типа.

Утилита chcat

Утилита chcat — позволяет изменить категорию безопасности SELinux для файла.

Изменить/Удалить категорию безопасности для каждого файла или пользователя. Используйте +/- для добавления/удаления категории у файла или пользователя.

Примечание. При удалении категории вы должны задать «--» в командной строке до использования синтаксиса «-Category». Это сообщит команде, что вы закончили вводить опции и теперь задаёте имя категории.

Синтаксис:

```
chcat category <file...>
chcat -l category <user...>
chcat [[+|-]category...] <file...>
chcat -l [[+|-]category...] <user...>
chcat [-d] <file...>
chcat -l [-d] <user...>
chcat -L [ -l ] [ <user ...> ]
```

Основные опции утилиты chcat приведены в таблице.

Опция	Значение опции
-d	Удалить категорию из каждого ФАЙЛА/ПОЛЬЗОВАТЕЛЯ.
-L	Вывести доступные категории.
-l	Сообщить команде chcat, что нужно работать с пользователями вместо файлов.

При работе с файлами данный скрипт вызывает команду chcon.

Утилита fixfiles

Утилита fixfiles — позволяет восстановить контекст безопасности SELinux для файла.

Синтаксис:

```
fixfiles [-F][ -R rpmpackagename[,rpmpackagename...] ]
[ -C PREVIOUS_FILECONTEXT ] [-l logfile ] [-o outputfile ] {check
| restore | [-F] relabel | verify }" fixfiles [-F]  [-l logfile ]
[-o outputfile ]{ check | restore|[-f] relabel | verify }
[[dir/file] ... ]
```

Утилита fixfiles используется, в первую очередь, когда необходимо скорректировать базу данных с контекстами безопасности (расширенные атрибуты) для файловой системы.

Его также можно запускать в любое время для добавления поддержки новой политики или чтобы убедиться в том, что файлам присвоены нужные контексты. По умолчанию скрипт обновляет метки на всех смонтированных файловых системах ext2, ext3, xfs и jfs, даже если они не имеют опции монтирования с контекстом безопасности. Вы можете использовать флаг «-R» для того, чтобы в качестве альтернативы использовать rpm-пакет.

Основные опции, используемые утилитой fixfiles, представлены в таблице.

Опция	Значение опции
`-l logfile`	Сохранить вывод в заданном файле logfile.
`-o outputfile`	Сохранить все имена файлов, имеющих <file_context>, отличающийся от контекста по умолчанию в файл outputfile.
`-F`	Принудительно установить контекст как в <file_context> для настраиваемых файлов (customizable files).
`-f`	Не предупреждать об удалении из директории /tmp.
`-R rpm_package_name [,rpm_package_name..`	При помощи базы данных rpm, найти все файлы из заданного пакета и восстановить для них контекст (-a выдаст все файлы из базы данных RPM).
`-C PREVIOUS_FILECONTEXT`	Запустить diff для сравнения файла PREVIOUS_FILECONTEXT с текущим и восстановить контекст на всех затронутых файлах.

Аргументы утилиты fixfiles описаны в таблице.

Аргумент	Значение аргумента
`check`	Вывести все некорректные метки контекста файлов. Показать и старый и новый контекст, но не изменять его.
`restore`	Изменить все некорректные метки контекста файлов.
`relabel`	Задать вопрос об удалении содержимого директории /tmp и, затем, изменения некорректных меток контекста файлов для приведения в соответствие с файлом <file_contexts>.
`verify`	Вывести все файлы с некорректным контекстом, не менять его.
`<dir/file>...`	Список файлов или директорий, для которых вы хотите проверить контекст.

Утилита genhomedircon

Утилита genhomedircon — позволяет создать спецификации с контекстом SELinux для файлов домашних директорий пользователей.

Синтаксис:

```
genhomedircon [ -d selinuxdir ] [-n | --nopasswd] [-t selinuxtype
] [-h]
```

Основные опции утилиты genhomedircon представлены в таблице.

Опция	Значение опции
`-h`	Вывести короткую подсказку.
`-d selinuxdir (--directory)`	Директория, где установлены файлы selinux. По умолчанию /etc/selinux.
`-n --nopasswd`	Указать утилите не брать информацию о домашних директориях из базы паролей.
`-t selinuxtype (--type)`	Указать тип установки selinux. По умолчанию - «целевая» («targeted»).

Данная утилита используется для генерации спецификации с контекстом SELinux для файлов домашних директорий пользователей, основываясь на их записи prefix в записи пользователя semanage. genhomedircon запускается во время создания политики. Утилита также может запускаться автоматически каждый раз, когда утилита semanage изменяет запись user или login. А именно, мы заменяем макроссы HOME_ROOT, HOME_DIR и ROLE в файле /etc/selinux/<<SELINUXTYPE>>/contexts/files/homedir_template на специфичные для пользователя сгенерированные значения.

HOME_ROOT и HOME_DIR заменяются на месторасположения пользовательских домашних директорий. По умолчанию - /home.

ROLE заменяется на значение, основанное на записи prefix в записи. user genhomedircon ищет все строчки для простых (не системных) пользователей в базе данных паролей. Такими пользователями будут те, чей UID больше или равен STARTING_UID (по умолчанию - 500), и для кого в поле login shell не стоит «/sbin/nologin» или «/bin/false».

Утилита load_policy

Утилита load_policy — позволяет загрузить новую политику SELinux в ядро.

Синтаксис:

```
load_policy [-bq]
```

load_policy - утилита, используемая для загрузки/замены политики в ядре. По умолчанию при загрузке политики load_policy сохраняет текущие значения переключателей.

Основные опции утилиты load_policy приведены в таблице.

Опция	Значение опции
`-b`	Сбросить переключатели в значения, определённые в политике.
`-q`	Скрыть предупреждения.

Утилита open_init_pty

Утилита open_init_pty – указывает запустить программу в псевдо-терминале.

Используется run_init для того, чтобы запустить программу после установки

правильного контекста. Эта программа берет новый псевдо-терминал, порождает (forks) процесс-наследник, который привязывается к новому псевдо-терминалу. Затем он подключает физический терминал, с которого он был вызван, к псевдо-терминалу, передавая ввод с клавиатуры процессу-наследнику и отправляя вывод процесса-наследника на физический терминал.

Он устанавливает атрибуты псевдо-терминала, основываясь на атрибутах физического терминала, а затем устанавливает пользовательский терминал в RAW-режим, не забыв сбросить его при выходе.

Синтаксис:

```
open_init_pty SCRIPT [[ARGS]...]
```

Утилита restorecon

Утилита restorecon - позволяет восстановить заданный по умолчанию контекст безопасности SELinux для файла (файлов).

Эта программа используется, в первую очередь, когда необходимо установить контекст безопасности (расширенные атрибуты) для одного или более файлов.

Ее можно запускать в любое время для коррекции ошибок, для добавления поддержки новой политики или с опцией «-n», чтобы убедиться в том, какой контекст присвоен файлу.

Синтаксис:

```
restorecon [-o outfilename ] [-R] [-n] [-v] [-e directory ]
pathname... restorecon -f infilename [-o outfilename ]
[-e directory ] [-R] [-n] [-v] [-F]
```

Основные опции утилиты restorecon приведены в таблице.

Опция	Значение опции
`-i`	Игнорировать несуществующие файлы.
`-f <infilename>`	<infilename> содержит список файлов, которые будут обработаны. Для того чтобы использовать стандартный ввод, используйте «-» (символ дефис).
`-e <directory>`	Задать директорию, которую нужно исключить из обработки (для нескольких директорий повторите опцию).
`-R -r`	Рекурсивно поменять метки для файлов и директорий.
`-n`	Не менять метки файлов.
`-o <outfilename>`	Сохранить список файлов с некорректным контекстом в <outfilename>.
`-v`	Показать изменения меток файлов.
`-vv`	Показать изменения меток файлов, если изменился тип, роль или пользователь.

Опция	Значение опции
`-F`	Если произошли изменения, то принудительно установить контекст как в <file_context> для настраиваемых файлов (customizable files) или секции пользователя.

Ниже описаны аргументы, используемые утилитой restorecon.

Аргумент	Значение аргумента
`pathname...`	Путь к файлу(файлам), требующим изменения меток.
`restorecon`	Не обрабатывает файлы и директории по символическим ссылкам на них.

Утилита restorecond

Утилита restorecond - служба, который отслеживает создание файлов, и выставляет для них заданный по умолчанию контекст SELinux.

Данное руководство описывает программу restorecond. Используя inotify, данный демон следит за файлами, перечисленными в /etc/selinux/restorecond.conf. Демон позволяет убедиться, что эти файлы при создании будут иметь правильный контекст, который определен в политике.

Синтаксис:

```
restorecond [-d]
```

Опции утилиты restorecond описаны в таблице.

Опция	Значение опции
`-d`	Включить режим отладки. Приложение все так же исполняется на переднем плане, но начинает выводиться большое число отладочных сообщений.

Утилита run_init

Утилита run_init - запускает скрипт init в правильном контексте, который определён в /etc/selinux/targeted/contexts/initrc_context.

Синтаксис:

```
run_init SCRIPT [[ARGS]...]
```

Основные файлы, используемые утилитой run_init, описаны в таблице.

Файл	Описание файла
`/etc/passwd`	Информация об учетных записях пользователей.
`/etc/shadow`	Зашифрованные пароли и информация об устаревании паролей.

Файл	Описание файла

`/etc/selinux/`
`targeted/`
`contexts/`
`initrc_context`
	Содержит контекст, в котором должны исполняться инициализационные скрипты.

Утилита semanage

Утилита semanage - утилита управления политикой SELinux.

Утилита semanage используется для настройки некоторых элементов политики SELinux без необходимости модификации или повторной компиляции исходного текста политики. В число таких настроек входит сопоставление имён пользователей ОС пользователям SELinux (которые контролируют исходный контекст безопасности, присваиваемый пользователям ОС во время их регистрации в системе, и ограничивают доступный набор ролей).

Также в число настраиваемых элементов входит сопоставление контекстов безопасности для различных видов объектов, таких как: сетевые порты, интерфейсы, сетевые узлы (хосты), а также контексты файлов.

Обратите внимание, что при вызове команды «semanage login» сопоставляются имена пользователей ОС (logins) пользователям SELinux. А при вызове команды «semanage user» сопоставляются пользователи SELinux доступному набору ролей. В большинстве случаев администратором выполняется настройка только первого типа сопоставлений. Второй тип сопоставлений, как правило, определяется базовой политикой и обычно не требует модификации.

Синтаксис:

```
semanage {login|user|port|interface|fcontext|translation} -l [-n]
semanage login -{a|d|m} [-sr] <login_name>
semanage user -{a|d|m} [-LrRP] <selinux_name>
semanage port -{a|d|m} [-tr] [-p protocol] <port> | <port_range>
semanage interface -{a|d|m} [-tr] <interface_spec>
semanage fcontext -{a|d|m} [-frst] <file_spec>
semanage translation -{a|d|m} [-T] <level>
```

Основные опции, используемые утилитой semanage, описаны в таблице.

Опция	Значение опции
`-a, --add`	Добавить запись ОБЪЕКТА с заданным ИМЕНЕМ.
`-d, --delete`	Удалить запись ОБЪЕКТА с заданным ИМЕНЕМ.
`-f, --ftype`	Тип файла. Эта опция используется совместно с «fcontext». Тип необходимо указывать в таком же виде, как и в выводе программы ls, иными словами, «--» указывает на обыкновенные файлы, а «-d» – только на директории.
`-h, --help`	Показать справку.
`-l, --list`	Вывести список ОБЪЕКТОВ.

Опция	Значение опции
-L, --level	Уровень чувствительности SELinux, s0 по умолчанию (только для систем с поддержкой MLS/MCS).
-m, --modify	Изменить ОБЪЕКТ с заданным ИМЕНЕМ записи.
-n, --noheading	Не выводить шапку таблицы при печати списка ОБЪЕКТОВ.
-p, --proto	Протокол, используемый для указанного порта (tcp|udp).
-r, --range	Диапазон безопасности MLS/MCS (только для систем с поддержкой MLS/MCS).
-R, --role	Роли SELinux. При указании нескольких ролей нужно отделить их пробелами и заключить в кавычки или указывать опцию «-R» необходимое число раз.
-P, --prefix	Префикс SELinux. При установке меток на домашние директории пользователей префикс добавляется к <home_dir_t> и <home_t>.
-s, --seuser	Имя пользователя SELinux.
-t, --type	Тип объекта SELinux.
-T, --trans	Трансляция SELinux (SELinux Translation).

Примеры:
Просмотреть сопоставления для пользователей SELinux:

```
semanage user -l
```

Разрешить joe регистрироваться как staff_u:

```
semanage login -a -s staff_u joe
```

Определить контекст файлов для всего, расположенного в /web (потом это определение используется утилитой restorecon):

```
semanage fcontext -a -t httpd_sys_content_t "/web(/.*)?"
```

Разрешить демону Apache прослушивать порт с номером 81:

```
semanage port -a -t http_port_t -p tcp 81
```

Утилита semodule

Утилита semodule - утилита, используемая для управления пакетами модулей политики SELinux. Управление включает в себя задачи установки, обновления, просмотра и удаления пакетов модулей. semodule можно также использовать

для принудительного пересоздания политики из хранилища модулей и/или для принудительной перезагрузки политики без выполнения каких-либо других операций. semodule работает с пакетами модулей, созданными утилитой semodule_package. По соглашению такие файлы имеют расширение .pp (policy package). Однако присваивать именно такое расширение не обязательно.

Синтаксис:

```
semodule [options]... MODE [MODES]...
```

Основные опции утилиты semodule описаны в таблице.

Опция	Значение опции
-R, --reload	Принудительно перезагрузить политику.
-B, --build	Принудительно пересоздать политику (если не используется опция -n, то происходит и ее перезагрузка).
-i,--install= MODULE_PKG	Установить/заменить модуль пакета.
-u,--upgrade= MODULE_PKG	Обновить существующий модуль пакета.
-b,--base= MODULE_PKG	Установить/заменить базовый модуль пакета.
-r,--remove= MODULE_NAME	Удалить существующий модуль.
-l,--list-modules	Показать список установленных модулей (кроме базовых).
-s,--store	Имя хранилища, с которым производятся операции.
-n,--noreload	Не перезагружать политику после выполнения операции.
-h,--help	Вывести подсказку.
-v,--verbose	Подробный вывод.

Примеры:
Установить или заменить базовый пакет политики:

```
semodule -b base.pp
```

Установить или заменить не базовый пакет политики:

```
semodule -i httpd.pp
```

Показать список установленных модулей (кроме базовых):

```
semodule -l
```

Установить или заменить все не базовые пакеты в текущей директории:

```
semodule -i *.pp
```

Установить или заменить все модули в текущей директории:

```
ls *.pp | grep -Ev "base.pp|enableaudit.pp" | xargs
/usr/sbin/semodule -b base.pp -
```

Утилита semodule_deps

Утилита semodule_deps - показать зависимости между пакетами модулей политики SELinux.

Для каждого модуля она выводит список необходимых для удовлетворения зависимостей модулей. Утилита имеет дело только с абсолютно необходимыми, а не опциональными зависимостями.

Для того чтобы получить от semodule_deps действительно полезную информацию, необходимо, чтобы передаваемые ей пакеты не имели не удовлетворённых зависимостей. На практике это означает, что, как правило, список модулей будет весьма длинным.

По умолчанию, базовые модули исключаются, поскольку почти каждый модуль зависит от базовых. Опция «-b» позволяет включить и эти зависимости.

В дополнение к выводу, форматированному для удобного восприятия человеком, semodule_deps с опцией «-g» умеет выгружать данные в формате Graphviz (http://www.graphviz.org/). Это удобно для создания графического представления зависимостей.

Синтаксис:

```
semodule_deps [-v -g -b] basemodpkg modpkgl [modpkg2 ... ]
```

Основные опции утилиты semodule_deps приведены в таблице.

Опция	Значение опции
-v	Подробный вывод.
-g	Вывод информации о зависимостях в формате Graphviz.
-b	Включить в вывод зависимостей базовые модули (по умолчанию не используется).

Утилита semodule_expand

Утилита semodule_expand - утилита добавления пакета модуля политики SELinux.

Данная утилита не входит в число утилит, необходимых при ежедневной работе с SELinux.

Обычно такие операции, как добавление (expanding) выполняются незаметно для пользователя при помощи libsemanage в ответ на команды semodule.

Базовые пакеты модулей политики могут создаваться утилитой semodule_package или semodule_link (когда в единый пакет связываются несколько пакетов).

Синтаксис:

```
semodule_expand [-V -c <version>] basemodpkg <outputfile>
```

Основные опции утилиты semodule_expand приведены в таблице.

Опция	Значение опции
-V	Отобразить версию.
c <version>	Версия создаваемой политики.

Утилита semodule_link

Утилита semodule_link - утилита объединения нескольких пакетов модулей политики SELinux.

Данная утилита не входит в число утилит, необходимых при каждодневной работе с SELinux. Обычно, такие операции как связывание выполняются незаметно для пользователя при помощи «libsemanage» в ответ на команды semodule. Пакеты модулей создаются утилитой semodule_package.

Синтаксис:

```
semodule_link [-Vv] [-o <outfile>] basemodpkg <modpkgl>
[<modpkg2>]...
```

Основные опции утилиты semodule_link приведены в таблице.

Опция	Значение опции
-V	Отобразить версию.
-v	Подробный вывод.
-o <output file>	Объединённый пакет модулей политики, созданный утилитой.

Утилита semodule_package

Утилита semodule_package - это утилита, используемая для создания пакета модуля политики SELinux из бинарного модуля политики и других источников данных, таких как файл контекстов. semodule_package создает пакеты из бинарных модулей политики, созданных, в свою очередь, при помощи «checkmodule». Пакеты политик, созданные semodule_package, могут быть установлены утилитой semodule.

Синтаксис:

```
semodule_package -o <output file> -m <module> [-f <file contexts>]
```

Примеры:
Создать пакет модуля политики для базового модуля.

```
semodule_package -o base.pp -m base.mod -f file_contexts
```

Создать пакет модуля политики для модуля httpd.

```
semodule_package -o httpd.pp -m httpd.mod -f httpd.fc
```

Создать пакет модуля политики для локальных TE-правил (TE rules) без использования файла контекстов.

```
semodule_package -o local.pp -m local.mod
```

Основные опции утилиты semodule_package приведены в таблице.

Опция	Значение опции
-o --outfile <output_file>	Пакет модуля политики, создаваемый данной утилитой.
-s --seuser <seuser_file>	Файл seuser будет включён в пакет.
-u --user_extra <user_extra_file>	Файл <user_extra> будет включён в пакет.
-m --module <module file>	Файл модуля политик будет включён в пакет.
-f --fc <file context file>	Файл с контекстами файлов для модуля (опционально).
n --nc <netfilter_context_file>	Включить в пакет файл контекста netfilter.

Утилита sestatus

Утилита sestatus - позволяет просмотреть состояние SELinux.

Эта утилита предназначена для просмотра статуса системы, использующей SELinux. Она показывает информацию о том, включён ли SELinux в целом, загруженную политику, а также в каком режиме система - режиме блокировок (enforcing mode) или режиме предупреждений (permissive mode). Утилита также может применяться для просмотра контекста безопасности файлов и процессов, описанных в файле /etc/sestatus.conf.

Пример запуска:

```
sestatus
```

```
SELinux status: enabled SELinuxfs mount: /selinux Current Mode:
permissive Policy version: 16
```

Синтаксис:

```
sestatus [-v] [-b]
```

Основные опции утилиты sestatus приведены в таблице.

Опция	Значение опции
-v	Просмотреть контекст безопасности файлов и процессов, описанных в файле /etc/sestatus.conf. Для символической ссылки также будет показан и контекст файла, на который она ссылается.
-b	Посмотреть текущее состояние переключателей.

Утилита setfiles

Утилита setfiles - позволяет установить контекст безопасности SELinux для файла.

Эта программа используется, в первую очередь, когда необходимо инициализировать базу данных с контекстами безопасности (расширенные атрибуты) для одной или более файловых систем.

Первоначально программа исполняется как часть процесса установки SELinux. Ее можно запускать в любое время для следующих задач: коррекция ошибок, добавление поддержки новой политики, просто чтобы убедиться в том, какой контекст присвоен файлу (с опцией «-n»).

Синтаксис:

```
setfiles [-c <policy> ] [-d] [-l] [-n] [-e <directory> ]
[-o <filename> ] [-q] [-s] [-v] [- vv] [-W] [-F] <spec_file>
pathname...
```

Основные опции утилиты setfiles приведены в таблице.

Опция	Значение опции
-c	проверить соответствие контекста тому, что определён в двоичном файле политики.
-d	показать, какая спецификация соответствует каждому файлу «-l» – отразить изменения меток в syslog.
-n	не менять метки файлов.
-q	подавить вывод, не относящийся к ошибкам.
-r <rootpath>	использовать альтернативный путь к корневой директории.

Опция	Значение опции
`-e <directory>`	задать директорию, которую нужно исключить из обработки (для нескольких директорий повторите опцию).
`-F`	Принудительно установить контекст как в <file_context> для настраиваемых файлов (customizable files).
`-o <filename>`	сохранить список файлов с некорректным контекстом в <filename>.
`-s`	получить список файлов со стандартного ввода, вместо использования пути, определяемого в командной строке.
`-v`	показать изменения меток файлов, если изменился тип или роль.
`-vv`	показать изменения меток файлов, если изменился тип, роль или пользователь.
`-W`	вывести предупреждения, если встретятся спецификации, которым не соответствует ни один файл.

Основные аргументы утилиты setfiles описаны в таблице.

Аргумент	Значение аргумента
`spec_file`	Файл со спецификациями, содержащий строки следующего вида: regexp [-type] («context» | «none»). В начале и в конце строки расположены регулярные выражения. Необязательное поле указывает тип файла в том же виде, как в выводе программы ls, иными словами, «--» указывает только на обыкновенные файлы, а «-d» – только на директории. В поле «context» может быть указан контекст безопасности или строка «none», которая указывает, что контекст файла не изменяется. Приоритет имеют спецификации, указанные последними. Если заданно несколько разных спецификаций с различающимся контекстом безопасности для нескольких жестких ссылок на файл, то, несмотря на выводимые предупреждения, файл получит контекст, указанный в последней из таких спецификаций. Если последняя из таких спецификаций содержит «none», то она не учитывается.
`pathname...`	Путь к корневой директории для каждой из файловых систем, для которых проставляются контексты. Не используется вместе с опцией «-s».

Утилита setsebool

Утилита setsebool устанавливает текущее значение для одного переключателя или целого списка переключателей SELinux. В качестве значения можно устанавливать «1», «true» или «on» для включения переключателя. Для выключения

можно использовать «0», «false» или «off».

Если не использовать опцию «-P», то изменения коснутся только текущих значений переключателей. Значения по умолчанию, которые устанавливаются во время загрузки, при этом не меняются.

Если задана опция «-P», то все текущие значения переключателей сохраняются в файле политики на диске. Таким образом, их значение сохранится и после перезагрузки.

Синтаксис:

```
setsebool [ -P ] <boolean> <value> | <bool1=val1> <bool2=val2>...
```

Утилита sealert

Утилита sealert позволяет в режиме реального времени отображать для текущего пользователя обнаруженные конфликты и возможные нарушения информационной безопасности.

В случае если система обнаружила сбой в области системного трея возникает всплывающее уведомление (рисунок 6.2).

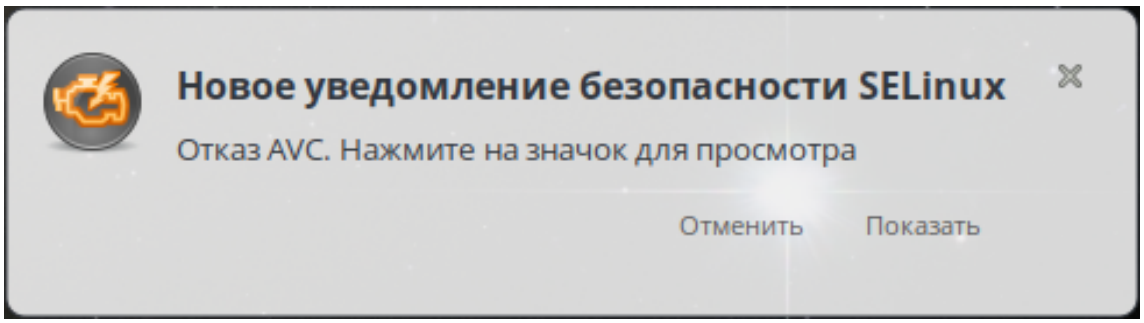

Рисунок 6.2 — Уведомление о нарушении безопасности

По нажатию кнопки «Показать» отображается окно с расширенной информацией о событии (рисунок 6.3).

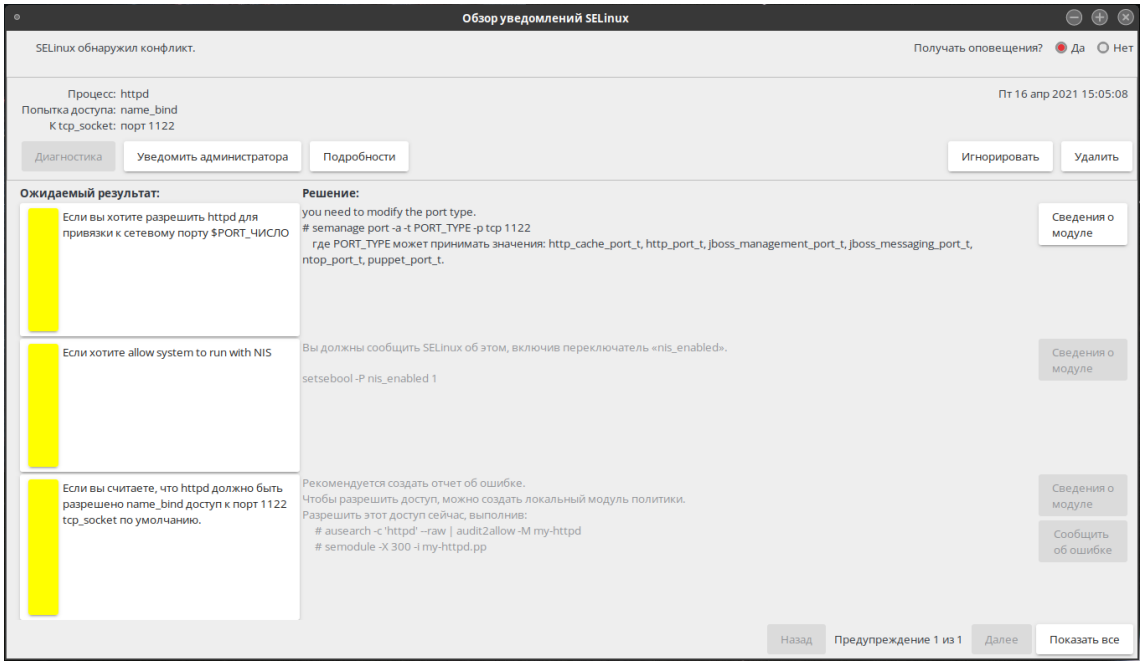

Рисунок 6.3 — Окно уведомлений

Также пользователь может вызвать это окно с консоли командой:

```
sealert
```

Из текущего окна пользователь может просмотреть подробности произошедшего события, уведомить администратора ОС, удалить информацию данного уведомления или игнорировать информацию.

В расширенной информации приводятся сведения об ожидаемом результате и возможные пути решения проблемы.

6.7 PAM

PAM или Pluggable Authentication Modules (подключаемые модули аутентификации) — 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 аутентификации 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».

console.perms – несколько выбивается своим названием, но не функциями. Здесь определяются права, получаемые привилегированными пользователями к консоли во время входа в систему и возвращаемые при выходе. Модуль «pam_console».

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

pam_nologin: Тип «auth». Стандартная реакция на наличие файла /etc/nologin. 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

необходимо 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. Суперпользователь 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 пользователей группы «wheel» от необходимости вводить пароль при смене uid на 0.

Для отслеживания попыток неуспешной аутентификации необходимо от имени администратора root отредактировать файлы «system-auth» и «password-auth»:

```
vi /etc/pam.d/system-auth
vi /etc/pam.d/password-auth
```

Добавить в секцию «auth» две строки, устанавливающие блокировку учётной записи на время не менее 15 минут в случае ввода не более 4 неправильных паролей. И 1 строку, подключающую нужный модуль.

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.

Для многофакторной аутентификации используется модуль «pam_usb». Чтобы использовать FLASH-накопитель в качестве токена аутентификации, необходимо подключить к ЭВМ FLASH-накопитель и убедиться, что ОС успешно его подмонтировала.

Добавить FLASH-накопитель, который будет использоваться как токен для аутентификации:

```
   pamusb-conf --add-device <Auth-Stick>
```

где <Auth-Stick> произвольное имя, присваиваемое токену.

Далее необходимо выбрать устройство из числа обнаруженных:

```
   Please select the device you wish to add.
   * Using "JetFlash TS128MJF2A (JetFlash_TS128MJF2A-0:0)"
(only option)
   Which volume would you like to use for storing data ?
```

И подтвердить его использование:

```
   * Using "/dev/sdc1 (UUID: 1CD6-B3F1)" (only option)
   Name    : Auth-Stick
   Vendor    : JetFlash
   Model    : TS128MJF2A
   Serial    : JetFlash_TS128MJF2A-0:0
   UUID    : 1CD6-B3F1
```

Подтвердить сохранение параметров:

```
   Save to /etc/pamusb.conf ?
```

```
   [Y/n] y
```

```
   Done.
```

Далее необходимо добавить пользователей, которые могут использовать этот токен:

```
   pamusb-conf --add-user='petrov'
```

```
Which device would you like to use for authentication ?
* Using "Auth-Stick" (only option)
User    : petrov
Device    : Auth-Stick
Save to /etc/pamusb.conf ?
```

```
[Y/n] y
```

```
Done.
```

В файлах /etc/pam.d/system-auth и /etc/pam.d/password-auth нужно в начало файла добавить строку, отвечающую за загрузку модуля, и установить уровень управляющей директивы, как было описано ранее:

```
auth sufficient pam_usb.so
```

6.8 Rsyslog

Rsyslog — это очень быстрый, расширяемый сервис для управления логами с огромным количеством возможностей. Среди его возможностей можно отметить поддержку фильтрации контента, а также передачу логов по сетям. Основные возможности:

- Многопоточность;
- TCP, RELP;
- Поддержка MySQL, PostgreSQL, Oracle;
- Фильтрация журналов;
- Полностью настраиваемый формат вывода.

Все программы ведут лог путём отправки сообщений об ошибках или своём состоянии с помощью сокета «syslog» или просто записывая все сообщения в файл, который будет находиться в каталоге /var/log/.

Но важное значение имеет уровень подробности логирования. Вы можете настраивать подробность в каждой отдельной программе, или же с помощью «syslog». Это поможет уменьшить использование дискового пространства на хранение логов. Но тут нужно найти компромисс между количеством информации и использованием диска.

Например, ядро ОС определяет такие уровни логов, или как мы будем называть их ниже — приоритеты:

KERN_EMERG — система неработоспособна.
KERN_ALERT — нужно немедленно принять меры.
KERN_CRIT — критическая ошибка.
KERN_ERR — обычная ошибка.
KERN_WARNING — предупреждение.
KERN_NOTICE — замечание.
KERN_INFO — информационное сообщение.
KERN_DEBUG — сообщения отладки.

Подобные уровни лога поддерживаются в большинстве программ, которые ведут логи.

Все настройки Rsyslog находятся в файле /etc/rsyslog.conf и других конфигурационных файлах из /etc/rsyslog.d. Вы можете посмотреть существуют ли у вас эти файлы выполнив:

```
ls /etc/rsys*
rsyslog.conf rsyslog.d/
```

Основной конфигурационный файл — /etc/rsyslog.conf, в нем подключены все файлы из папки rsyslog.d с помощью директивы «IncludeConfig» в самом начале файла:

```
IncludeConfig /etc/rsyslog.d/*.conf
```

В этих файлах могут содержаться дополнительные настройки, например, аутентификация на Rsyslog сервере. В главном конфигурационном файле содержится очень много полезных настроек. Обычно он обеспечивает управление локальными логами по умолчанию, но для работы через сеть нужно добавить настройки.

Синтаксис конфигурационного файла очень прост:

```
<переменная> <значение>
```

Все директивы начинаются со знака доллара, содержат имя переменной, а дальше связанное с ней значение. Так выглядит каждая строка конфигурационного файла. В его первой части размещены общие настройки программы и загрузка модулей. Во второй — ваши правила сортировки и фильтрации лог файлов.

```
ModLoad imuxsock # provides support for local system logging
$ModLoad imklog # provides kernel logging support
#$ModLoad immark # provides --MARK-- message capability
# provides UDP syslog reception
#$ModLoad imudp
#$UDPServerRun 514
# provides TCP syslog reception
#$ModLoad imtcp
#$InputTCPServerRun 514
```

В этом участке загружаются все необходимые модули программы. Существуют четыре типа модулей.

- Модули ввода — можно рассматривать, как способ сбора информации из различных источников, начинаются с «im»;
- Модули вывода — позволяют отправлять сообщения в файлы или по сети, или в базу данных, имя начинается на «om»;
- Модули фильтрации — позволяют фильтровать сообщения по разным параметрам, начинаются с «fm»;

- Модули парсинга — предоставляют расширенные возможности для синтаксического анализа сообщения, начинаются с «pm».

Сначала загружается модуль «imuxsock», который позволяет сервису получать сообщения из сокета, а второй «imklog» получает сообщения ядра. Следующим загружается модуль «mark», который позволяет маркировать соединения, или же выводить сообщения о том, что «syslog» все ещё работает. Например, вы можете попросить rsyslog выводить сообщения каждые 20 минут:

```
MarkMessagePeriod 1200
```

Дальше идут глобальные директивы:

```
ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat
```

В этой строке мы указываем, что нужно использовать стандартный формат хранения времени, в секундах с 1970 года.

Дальше следует набор прав разрешений для файлов журналов, которые будут созданы в вашей системе:

```
FileOwner root
FileGroup adm
FileCreateMode 0640
DirCreateMode 0755
Umask 0022
```

После создания этих файлов, их права можно менять на те, которые вам нужны.

Выше была более общая настройка «syslog», ну а теперь правила сортировки логов. Вот набор правил по умолчанию:

```
auth,authpriv.* - /var/log/auth.log
*.*;auth,authpriv.none - /var/log/syslog
cron.* - /var/log/cron.log
daemon.* - /var/log/daemon.log
kern.* - /var/log/kern.log
lpr.* - /var/log/lpr.log
mail.* - /var/log/mail.log
user.* - /var/log/user.log
```

Каждое правило имеет свой синтаксис, сначала идёт источник и приоритет, затем действие. Если источник и приоритет совпадают, сообщение отправляется в указанный файл. Например, мы можем отправить больше сообщений в лог системы /var/messages:

```
*.info;mail.none;authpriv.none;cron.none /var/log/messages
```

В этой строке мы выбираем все сообщения уровня info, кроме mail, authpriv и cron. Шаблон mail.none будет означать, что не нужно логировать ни один

уровень сообщений. Соответственно конструкция *.info — значит логировать сообщения от всех источников, но только с уровнем info, *.* — это вообще все сообщения, со всеми уровнями.

Источник и приоритет нечувствительны к регистру. Также заметьте, что приоритеты уровней error, warn и panic больше не используются, так как считаются устаревшими. В целом, в качестве источников вы можете использовать:

- auth;
- authpriv;
- cron;
- daemon;
- kern;
- lpr;
- mail;
- mark;
- news;
- security (эквивалентно auth);
- syslog;
- user;
- uucp;
- local0 ... local7.

А в качестве приоритетов вы можете применить:

- emerg (раньше panic);
- alert;
- crit;
- error (раньше err);
- warn (раньше warning);
- notice;
- info;
- debug.

Звёздочки на любом месте означают все варианты. Для фильтрации логов могут использоваться не только источник и приоритет, но и более сложные выражения на основе условий и сравнений.

Вы можете выполнять фильтрацию сообщений с помощью такого синтаксиса:

```
:<поле>, <сравнение>, <значение> <путь_к_файлу>
```

В качестве операции сравнения вы можете использовать такие варианты:

- contains — поле содержит указанное значение;
- isequal — поле должно быть идентичным значению;
- startswith — поле должно начинаться со значения;
- regex — сравнивает поле с регулярным выражением.

Например, отфильтруем сообщения только от определённой программы:

```
:syslogtag, isequal, "giomanager:" /var/log/giomanager.log
& stop
```

Кроме того, можно использовать более простой синтаксис, в виде выражения if. Вот основной синтаксис:

```
if $<поле> <сравнение> <значение> then <файл>
```

Здесь используются те же самые компоненты, только выглядит немного проще. Например:

```
if $syslogtag == 'giomanager' then /var/log/giomanager.log
```

Отправить лог на удалённый сервер достаточно просто, для этого достаточно указать @ и ip-адрес удалённой машины, на которой запущен rsyslog:

```
*.info;mail.none;authpriv.none;cron.none @xx.xx.xx.xx:514
```

Здесь 514 — это порт, на котором слушает rsyslog. Настройка rsyslog на приём логов заключается в запуске сервиса с модулями imtcp и imudp. Далее, все что нужно для того, чтобы получить логи с определённой машины, отфильтровать их из общего потока с помощью фильтров:

```
if $fromhost-ip contains '192.168.1.10' then
/var/log/proxyserver.log
```

Без фильтров сообщения с разных машин будут писаться в один общий лог системы в зависимости от того, как вы их распределите.

6.9 Afick - верификация целостности

Afick - это быстрая и доступная утилита, помогающая при обнаружении вторжений, а также позволяющая контролировать общую целостность системы.

Afick контролирует изменения в файловой системе и сразу сообщает вам о них, таким образом, предоставляя вам выбор решить, действительно ли ожидались эти изменения. Эта информация может помочь вам в расследовании инцидента, когда необходимо определить, какие были произведены изменения в системе в результате взлома.

В процессе установки Afick формирует базу данных файлов, каталогов и их соответствующих им MD5 контрольных сумм. Файлы и каталоги, включенные в эту базу данных, выбираются соответственно входным данным из файла конфигурации Afick, называемого afick.conf, после того, как Afick установит этот файл в /etc каталог. Файл конфигурации afick.conf имеет простую синтаксическую структуру. По вашему усмотрению Вы можете очень быстро добавить или удалить типы файлов, каталоги и т.д. Ниже приведено содержимое файла afick.conf. Обратите внимание, что элементы в файле конфигурации чувствительны к регистру.

```
afick config sample file
```

```
# directives
############
database:=/var/lib/afick/afick
```

Определяет, какую базу данных будет использовать Afick.

```
# report_url := stdout
```

Определяет, куда Afick будет выводить результаты своей работы.

```
# verbose := no
# warn_dead_symlinks := no
# report_full_newdel := no
# warn_missing_file := no
# running_files := no
# timing := no
# text files
exclude_suffix := log LOG html htm HTM txt TXT xml
```

Определяет, что Afick должен игнорировать текстовые файлы с такими расширениями.

```
# help files
exclude_suffix := hlp pod chm
```

Определяет, что Afick должен игнорировать файлы справки с такими расширениями.

```
# old files
exclude_suffix := tmp old bak
```

Определяет, что Afick должен игнорировать временные файлы с такими расширениями.

```
# fonts
exclude_suffix := fon ttf TTF
```

Определяет, что Afick должен игнорировать файлы шрифтов с такими расширениями.

```
# images
exclude_suffix := bmp BMP jpg JPG gif png ico
```

Определяет, что Afick должен игнорировать файлы изображений с такими расширениями.

```
# audio
exclude_suffix := wav WAV mp3 avi
```

Определяет, что Afick должен игнорировать медиа файлы с такими расширениями.

```
# macros
########
# used by cron
@@define MAILTO root
```

Определяет пользователя, которому будут отсылаться отчёты по работе Afick.

```
@@define LINES 1000
```

Определяет максимальное количество строк в отчёте.

```
# list the file or directories to scan
# syntaxe :
# file action
# to have action on file (see below
# ! file
# to remove file from scan
# file with blank character have to be quoted
```

```
# action : a list of item to check
```
Ниже описаны опции, определяющие, какие атрибуты файлов нужно контролировать.
```
# md5 : md5 checksum
# d : device
# i : inode
# p : permissions
# n : number of links
# u : user
# g : group
# s : size
# b : number of blocks
# m : mtime
# c : ctime
# a : atime
#R: p+d+i+n+u+g+s+m+c+md5
#L: p+d+i+n+u+g
# action alias may be configured with
# your_alias = another_alias|item[+item][-item]
# all is a pre-defined alias for all items except "a"
# alias :
#########
DIR = p+i+n+u+g
ETC = p+d+i+u+g+s+md5
Logs = p+n+u+
MyRule = p+d+i+n+u+g+s+b+md5+m .
# files to scan
###############
=/ DIR
```
Проверка с использованием описанных выше правил для каталогов.
```
#
/bin MyRule
/boot MyRule
!/boot/map
```
Игнорируется указанный каталог.
```
!/boot/System.map
```
Игнорируется указанный файл.
```
/etc ETC
/etc/mtab ETC - i
/etc/adjtime ETC - md5
/etc/aliases.db ETC - md5
/etc/mail/statistics ETC - md5
!/etc/map
!/etc/webmin/sysstats/modules/
!/etc/cups/certs/0
/lib MyRule
/lib/modules MyRule -m
/root MyRule
!/root/.viminfo
```

```
!/root/.bash_history
!/root/.mc
/sbin MyRule
/usr/bin MyRule
/usr/sbin MyRule
/usr/lib MyRule
/usr/local/bin MyRule
/usr/local/sbin MyRule
/usr/local/lib MyRule
/var/ftp MyRule
/var/log Logs
/var/www MyRule
```

Как видно, файл конфигурации Afick довольно прост для понимания. Ниже приведён пример, в котором к проверке целостности Afick добавляется ваш основной каталог. К примеру, вам необходимо, чтобы файлы в этом каталоге проверялись на изменения при монопольном доступе, изменение прав доступа, изменения размера файлов и времени последнего обращения к файлу. Для начала нужно создать новый элемент, под разделом #alias в файле конфигурации afick.conf, как показано ниже:

```
HOME = u+g+p+m+s
```

Затем в разделе #files to scan необходимо добавить следующую строку:

```
/home/yourusername HOME
```

Теперь, при следующем запуске, Afick добавит ваш каталог в свою базу данных и будет контролировать находящиеся в нем файлы, согласно указанным вами критериям. Если вы хотите, чтобы ваши изменения применились немедленно, то можно запустить Afick вручную, используя следующую команду:

```
Afick -- update
```

Иначе, вам придётся ждать запуска крон задачи Afick. Эта задача добавляется автоматически во время инсталляции программы и запускается один раз в день. Результаты работы данного задания вы получите по электронной почте на адрес, указанный в разделе <MAILTO> файла конфигурации «afick.conf». Используя почтового клиента, вы сможете увидеть ежедневный отчёт примерно в таком виде:

```
This is an automated report generated by Another File Integrity
Checker on
+localhost.localdomain at 07:46:07 AM on 02/25/2004.
Output of the daily afick run:
new file : /var/log/afick/afick.log.2
new file : /var/log/afick/error.log.2
```

```
deleted file : /etc/sysconfig/iptables
changed file : /etc/adjtime
changed file : /etc/aliases.db
changed file : /etc/mail/statistics
changed file : /etc/prelink.cache
changed file : /etc/printcap
detailed changes
changed file : /etc/adjtime
MD5 : 7+bTDZQbxsTXEJXhyI2GCw ao6a/yDwoBR8GSL1AKlWXQ
changed file : /etc/aliases.db
MD5 :GT/eP5D+B8apNoa7L5CLRw soh7MnLDuQw4gI9KHlhpTA
changed file : /etc/mail/statistics
MD5 :oshq17jZ2a0o5pYhVBRgwQ vb69gMWXvpIEEZ4fmOl9/Q
changed file : /etc/prelink.cache
MD5 : SKh/403FRMUqBNdCIInQ9A zeC+5EPFfWBR4OeT7xZdbw
changed file : /etc/printcap
MD5 : b5e3g2//bGaxeCxVyRJqaw QFY1NJGy/kdt32B1YV0TXQ
filesize : 194 581
```

В примере выше Afick сообщает нам, что некоторые файлы были изменены, созданы или удалены. Также нам показываются начальные и текущие контрольные суммы файлов, и сообщается, что в одном из файлов был изменён его размер. Afick узнает о происходящих изменениях в файле, сравнивая его атрибуты с атрибутами, которые были сохранены при последнем запуске Afick. Примером этого могут служить файлы в папке /usr/bin или в /sbin. Обычно эти файлы редко изменяются, если только вы не изменили их, обновляя программу (в противном случае они не должны изменяться).

Обратите внимание на то, где вы сохраняете вашу базу данных Afick (по умолчанию - /var/lib/afick/), потому что может возникнуть ситуация, когда ваша система была взломана, а вы об этом не узнаете, т.к. была нарушена целостность базы данных. Возможным решением данного вопроса может быть сохранение базы на защищённых от записи носителях (например, CD-ROM), после чего следует изменить файл конфигурации «afick.conf», чтобы указать на выбранное вами место сохранения базы.

6.10 AMTU - утилита тестирования абстрактной машины

amtu — утилита тестирования абстрактной машины (Abstract Machine Test Utility — AMTU).

Синтаксис:

```
amtu [ -dmsinph ]
```

AMTU - это административная утилита, которая проверяет, соблюдаются ли основные механизмы защиты аппаратного обеспечения.

AMTU выполняет следующие тесты:

- **память:** произвольно записывает данные в области памяти, затем считывает память, чтобы гарантировать, что записанные значения остаются

неизменными;

- **разделение памяти:** обеспечивает, чтобы программы пользовательского пространства не могли читать и записывать в области памяти, используемые такими элементами, как видеопамять и код ядра;
- **контроллер ввода / вывода — сеть:** проверяет, что переданные случайные данные также являются данными, полученными для каждого настроенного сетевого устройства. Проверяет только настроенные устройства Ethernet и Token Ring. Не проверяет асинхронные устройства;
- **контроллер ввода / вывода — диск:** проверяет, что случайные данные, записанные на диски, остаются неизменными. Проверяются только контроллеры IDE и SCI, связанные с установленными файловыми системами. Контроллеры дисков с файловыми системами, доступными только для чтения, не проверяются;
- **инструкции режима супервизора:** обеспечивает, что принудительное использование свойств привилегированных инструкций в режиме супервизора по-прежнему действует. Набор привилегированных инструкций, проверенных для подтверждения этого, зависит от архитектуры.

Для команды amtu доступны следующие параметры:

Параметр	Значение параметра
`-d`	выводить отладочные сообщения;
`-m`	выполнить проверку памяти;
`-s`	выполнить проверку разделения памяти;
`-i`	выполнить тест контроллера ввода-вывода;
`-n`	выполнить сетевой тест;
`-p`	выполнить проверку инструкций режима супервизора;
`-h`	показать справку.

Команда amtu выдаёт следующие коды возврата при выполнении:

- «-1» — ошибка выполнения;
- «0» — успех.

6.11 ntpdate

В РЕД ОС синхронизировать время можно следующими основными способами:

- вручную утилиты ntpdate;
- автоматически при помощи сервиса ntp.

Программа ntpdate – позволяет разово синхронизировать локальное время с эталонным сервером времени в интернете. Подобных эталонов существует достаточно много. Для примера можно воспользоваться одним из них — pool.ntp.org

Запускаем синхронизацию времени:

```
ntpdate pool.ntp.org
```

Утилита провела синхронизацию, в результате которой к системному времени было добавлено число секунд, необходимое для приближения к эталонному. Если в результате работы синхронизации возникает ошибка: «no server suitable for synchronization found», то попробуйте в работе утилиты использовать непривилегированный порт. По-умолчанию ntpdate работает по 123 порту. Если он закрыт на фаерволе, то помочь в синхронизации поможет следующий параметр:

```
ntpdate -u pool.ntp.org
```

Если у вас запуск ntpdate завершается ошибкой — the NTP socket is in use, exiting, значит у вас уже установлена и запущена служба ntpd, которая заняла свободный udp порт, необходимый для работы ntpdate.

Сервер времени ntp использует в своей работе одноимённый протокол — Network Time Protocol, которому для работы необходим UDP порт 123. Так что перед установкой и настройкой службы времени убедитесь, что на фаерволе открыт этот порт.

Устанавливаем сервер ntp:

```
dnf -y install ntp
```

Теперь отредактируем файл конфигурации /etc/ntp.conf , удалив все лишнее:

```
cat /etc/ntp.conf
```

```
driftfile /var/lib/ntp/drift
restrict default nomodify notrap nopeer noquery
restrict 127.0.0.1
restrict ::1
server 0.centos.pool.ntp.org iburst
server 1.centos.pool.ntp.org iburst
server 2.centos.pool.ntp.org iburst
server 3.centos.pool.ntp.org iburst
disable monitor
logfile /var/log/ntp.log
```

Параметры:

server - список серверов для синхронизации времени;

driftfile – задаёт адрес файла, в котором хранится история изменений времени во время синхронизации. Если по каким-то причинам синхронизация времени с внешними источниками станет невозможна, служба времени изменит системные часы в соответствии с записями в этом файле;

restrict 127.0.0.1 – указывает, что пользоваться нашим сервером времени можно только непосредственно с локального интерфейса. Если вам необходимо разрешить другим компьютерам в вашей локальной сети синхронизировать

время с текущей машины, то укажите в данном параметре адрес вашей сети, например:

```
restrict 192.168.10.0 mask 255.255.255.0
```

restrict default nomodify notrap nopeer noquery – параметры указывают на то, что клиентам данного сервиса времени запрещено изменять его настройки, получать его статус. Они могут только забрать с него значения точного времени;

disable monitor – данный параметр повышает безопасность, предотвращая использования одной из уязвимостей сервиса ntpd, которую можно использовать для проведения DDoS-атак;

logfile – указывает путь к файлу с логами сервиса.

После завершения редактирования файла настроек запускаем службу синхронизации времени:

```
systemctl start ntpd
```

Проверяем запустился ли сервер:

```
netstat -tulnp | grep 123
```

Все в порядке, служба слушает положенный порт 123. Проверим ещё на всякий случай системные логи ОС:

```
cat /var/log/messages | grep ntpd
cat /var/log/ntp.log
```

Теперь настроим автозапуск ntp вместе с загрузкой ОС:

```
systemctl enable ntpd
```

Наблюдать за работой службы ntp можно с помощью команды:

```
ntpq -p
```

Данные вывода:

remote – адрес удалённого эталона времени, с которого была синхронизация;

refid – указывает, откуда каждый эталон получает точное время. Это могут быть другие сервера времени, система GPS и другое;

st (stratum (уровень)) – это число от 1 до 16, которое указывает на точность эталона. «1» - максимальная точность, «16» — сервер недоступен. Уровень вашего сервера будет равен уровню наименее точного удалённого эталона плюс 1;

poll – интервал в секундах между опросами;

reach – восьмеричное представление массива из 8 бит, отражающего результаты последних восьми попыток соединения с эталоном. Бит выставлен, если удалённый сервер ответил;

delay – время задержки ответа на запрос о точном времени;

offset – разница между вашим и удаленным сервером;

jitter – дисперсия – это мера статистических отклонений от значения смещения (поле offset) по нескольким успешным парам запрос-ответ. Чем меньше значение дисперсии, тем лучше, поскольку позволяет точнее синхронизировать время.

6.12 Отказоустойчивый кластер

Corosync - это программа с открытым исходным кодом, которая предоставляет возможности кластерного членства и обмена сообщениями, часто называемые уровнем обмена сообщениями, на клиентские серверы. Pacemaker - это менеджер ресурсов кластера с открытым исходным кодом (CRM), который координирует ресурсы и службы, которые управляются и становятся доступными кластеру. По сути, Corosync позволяет серверам связываться как кластер, в то время как Pacemaker предоставляет возможность управлять тем, как ведёт себя кластер.

Архитектура pacemaker состоит из трёх уровней:

- Кластеронезависимый уровень – на этом уровне располагаются сами ресурсы и их скрипты, которыми они управляются и локальный демон, который скрывает от других уровней различия в стандартах, использованных в скриптах.
- Менеджер ресурсов (Pacemaker), который представляет из себя мозг. Он реагирует на события, происходящие в кластере: отказ или присоединение узлов, ресурсов, переход узлов в сервисный режим и другие административные действия. Pacemaker, исходя из сложившейся ситуации, делает расчёт наиболее оптимального состояния кластера и даёт команды на выполнение действий для достижения этого состояния (остановка/перенос ресурсов или узлов).
- Информационный уровень – на этом уровне осуществляется сетевое взаимодействие узлов, т.е. передача сервисных команд (запуск/остановка ресурсов, узлов и т.д.), обмен информацией о полноте состава кластера (quorum) и т.д. На этом уровне работает Corosync.

Для наглядности рассмотрим создание простого кластера с плавающим ip-адресом. В случае если работающий сервер отказывает, второй автоматически запускается и начинает работать вместо первого.

Если вы хотите иметь доступ к настройкам кластера через доменное имя, необходимо настроить DNS-адресацию серверов. Если у вас нет имени домена для использования, можно использовать плавающий IP-адрес для доступа к настройкам.

Всякий раз, когда есть несколько серверов, обменивающихся друг с другом информацией, особенно с помощью программного обеспечения для кластеризации, важно обеспечить синхронизацию их часов. Рекомендуется использовать NTP (Network Time Protocol) для синхронизации серверов.

Corosync использует UDP-транспорт между портами 5404 и 5406 . Если вы используете брандмауэр, убедитесь, что между этими серверами разрешена связь.

Установка Corosync и Pacemaker:

```
dnf install corosync pcs pacemaker
```

Для управления кластером рекомендуется пользоваться утилитой pcs. При установке pacemaker автоматически будет создан пользователь hacluster. Для использования pcs, а также для доступа в веб-интерфейс нужно задать пароль пользователю hacluster:

```
passwd hacluster
Changing password for user hacluster.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

Запуск сервиса:

```
systemctl start pcsd
```

Настраиваем аутентификацию (на одном узле):

```
pcs cluster auth <Сервер1> <Сервер2>
Username: hacluster
Password:
<Сервер1>: Authorized
<Сервер2>: Authorized
```

Вместо имён <Сервер1> и <Сервер2> используйте ваши доменные имена серверов.

После этого кластером можно управлять с одного узла.

Создаём кластер:

```
pcs cluster setup --name mycluster <Сервер1> <Сервер2>
```

```
Shutting down pacemaker/corosync services...
Redirecting to /bin/systemctl stop pacemaker.service
Redirecting to /bin/systemctl stop corosync.service
Killing any remaining services...
Removing all cluster configuration files...
<Сервер1>: Succeeded
<Сервер2>: Succeeded
Synchronizing pcsd certificates on nodes <Сервер1>, <Сервер2>...
<Сервер1>: Success
<Сервер2>: Success
Restaring pcsd on the nodes in order to reload the certificates...
<Сервер1>: Success
<Сервер2>: Success
```

Запускаем кластер:

```
pcs cluster start --all
```

```
<Сервер2>: Starting Cluster...
<Сервер1>: Starting Cluster...
```

Настройка автоматического включения кластера при загрузке:

```
pcs cluster enable --all
```

```
<Сервер1>: Cluster Enabled
<Сервер2>: Cluster Enabled
```

Смотрим результат:

```
pcs status cluster
```

```
Cluster Status:
Stack: unknown
Current DC: NONE
2 nodes and 0 resources configured
Node <Сервер1>: UNCLEAN (offline)
Node <Сервер2>: UNCLEAN (offline)
PCSD Status:
<Сервер1>: Online
<Сервер2>: Online
```

Проверка синхронизации узлов кластера:

```
corosync-cmapctl | grep members
```

```
runtime.totem.pg.mrp.srp.members.1.config_version (u64) = 0
runtime.totem.pg.mrp.srp.members.1.ip (str) = r(0)
ip(192.168.100.201)
runtime.totem.pg.mrp.srp.members.1.join_count (u32) = 1
runtime.totem.pg.mrp.srp.members.1.status (str) = joined
runtime.totem.pg.mrp.srp.members.2.config_version (u64) = 0
runtime.totem.pg.mrp.srp.members.2.ip (str) = r(0)
ip(192.168.100.202)
runtime.totem.pg.mrp.srp.members.2.join_count (u32) = 1
runtime.totem.pg.mrp.srp.members.2.status (str) = joined
pcs status corosync
Membership information
----------------------
Nodeid Votes Name
1 1 <Сервер1> (local)
```

```
   2 1 <Сервер2>
```

Настройка основных параметров кластера.

Механизм STONITH (Shoot-The-Other-Node-In-The-Head) — защита от Split-Brain : позволяет выключать/включать/перезагружать узлы кластера. Обычно этот механизм реализуется с помощью специальных сетевых устройств с удаленным управлением или через специальные платы управления сервером. В pacemaker устройства STONITH реализованы в виде кластерных ресурсов.

Кворум определяет минимальное число работающих узлов в кластере, при котором кластер считается работоспособным. По умолчанию, кворум считается неработоспособным, если число работающих узлов меньше половины от общего числа узлов. Так как узла у нас всего два, то кворума у нас не будет, поэтому необходимо отключить эту политику:

```
   pcs property set no-quorum-policy=ignore
```

Посмотреть какие получились настройки можно командой:

```
   pcs property
```

```
   Cluster Properties:
   cluster-infrastructure: corosync
   cluster-name: mycluster
   dc-version:
   have-watchdog: false
   last-lrm-refresh: 1480973783
   no-quorum-policy: ignore
   stonith-enabled: false
```

Настройки кластера можно проверить с помощью утилиты «crm_verify». Опция «-L» осуществляет диагностику всех работающих узлов кластера.

Чтобы сервисы кластера запускались автоматически при перезагрузке сервера, нужно выполнить на каждом узле кластера:

```
   systemctl enable pcsd
   systemctl enable corosync
   systemctl enable pacemaker
```

Создание виртуального плавающего ip-адреса (floatingip) VIP:

```
   pcs resource create virtual_ip ocf:heartbeat:IPaddr2
ip=192.168.100.203 cidr_netmask=32 op monitor interval=30s
   pcs status resources
   virtualip (ocf::heartbeat:IPaddr2): Started <Сервер1>
```

Если узлы кластера не одинаковы по производительности или доступности, то можно увеличить приоритет одного из узлов, чтобы он был активным всегда когда работает:

```
pcs constraint location virtualip prefers <Сервер1>=100
```

Настройка сервисов, ресурсов и прочего производится согласно документации, поставляемой разработчиком приложения, и может быть найдена в открытых источниках в глобальной сети Интернет и на сайте разработчика.

6.13 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 -n 10 -p 5645
```

```
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
```

6.14 Управление дисковыми квотами

Система diskquota обеспечивает механизм для управления используемым дисковым пространством. Ограничения могут быть установлены для каждого пользователя в отдельности, для любой или для всех файловых систем. Система ограничений (quota) будет предупреждать пользователей, когда они превысят свой дозволенный лимит, но будет позволять использовать некоторое дополнительное пространство для текущей работы. Система ограничений (quota system) является частью ядра ОС.

Команда:

```
quota
```

позволяет просмотреть любые ограничения дискового пространства для каждого пользователя.

Доступны два типа ограничений, которые могут быть наложены на пользователя, обычно если используется одно из ограничений, то и второе тоже будет использоваться. Ограничение может быть установлено как на все дисковое пространство пользователя , которое используется этим пользователем, так и на число файлов (inodes), которыми он может владеть. Quota обеспечивает информацию на ограничения, которые были установлены системным администратором, на каждую из областей, которые используются в данный момент. Ограничения по inodes и block накладываются как на uid (идентификатор пользователя), так и на gid (идентификатор группы). Так если вы входите в группу, которая превысила наложенное на неё ограничение, то вы не сможете использовать дисковое пространство, даже если вы все ещё можете использовать его как пользователь.

Существуют четыре числа для каждого ограничения:

- используемое в данное время ограничение;
- «мягкое» ограничение (softlimit);
- «жёсткое» ограничение (hardlimit);
- промежуток времени , после истечения которого «мягкое» ограничение интерпретируется как «жёсткое».

«Мягкое» ограничение определяет число блоков размером 1 Кбайт, которое пользователь может немного превысить. «Жёсткое» ограничение не может быть превышено ни каким образом. Если пользователь пытается превысить данное число, то он получает сообщение о невозможности сделать это. При этом ядро возвращает код ошибки EDQUOT. После того как пользователь превысит доступное для него «мягкое» ограничение (softlimit) устанавливается время, после истечения которого «мягкое» ограничение становится «жёстким» (hardlimit). Обычно срок этого периода истекает после 7 дней (1 неделя). В этот период времени пользователь может удалить ненужные ему файлы, после чего он вновь может использовать «мягкое» ограничение до момента истечения указанного

промежутка времени. После истечения указанного промежутка времени «мягкое» ограничение становится «жёстким» ограничением и у пользователя больше нет ресурсов для создания новых файлов.

Для того чтобы установить систему ограничений дискового пространства (quota) в ОС, системному администратору необходимо сделать несколько шагов:

- выбрать файловую систему, на которую будут накладываться ограничения;
- разрешить (включить) систему ограничений;
- произвести проверку файловой системы на ограничения дискового пространства;
- произвести проверку ограничений дискового пространства как для пользователей, так и для групп;
- запретить ограничения для пользователей и групп.

В первую очередь необходимо решить, на какую файловую систему необходимо наложить ограничения (quotas). Чаще всего ограничения накладываются на файловую систему, в которой располагаются домашние каталоги пользователей или на файловую систему, которая смонтирована в каталог /usr, и пользователи имеют право записывать на неё информацию. Для того чтобы разрешить ограничения на дисковое пространство, на необходимой файловой системе вы должны отредактировать файл /etc/fstab, добавив к указанной системе опции для ограничения дискового пространства (как для пользователей, так и для групп).

Отредактированный файл /etc/fstab может иметь вид:

```
#
# /etc/fstab
#
/dev/hda1 / ext2 defaults
/dev/hda2 none swap sw
/dev/hda3 /usr ext2 defaults
/dev/hdb1 /usr/users ext2 defaults,usrquota,grpquota
/dev/hdb2 /usr/src ext2 defaults,usrquota
none /proc proc defaults
```

Зарезервированное слово «usrquota», в поле опций включает ограничение дискового пространства (quotas) для пользователей (userquota) на данном устройстве. Зарезервированное слово «grpquota» включает ограничение дискового пространства для групп (groupquota) на данном устройстве. Во то время когда вы используете опции «usrquota» и «grpquota» без «=», ваши файлы ограничений (quotafile) будут находится в корневом каталоге каждой файловой системы, в которой используются ограничения на дисковое пространство. Файл называемый «aquota.user» будет использоваться для ограничений пользователей, а файл «aquota.group» будет использоваться для ограничений групп. Однако, вы сами можете определить ваши файлы ограничений.

> Например, строка «usrquota=/usr/adm/quotasrc.user» установит файл ограничений для пользователей в каталоге /usr/adm, который будет назы-

ваться «quotarc.user». Однако, будьте внимательны, и отслеживайте длину строки в файле /etc/fstab (смотрите ее определение в файле mntent.h).

Периодически (в основном после некорректной перезагрузки системы и во время первого разрешения ограничений на дисковое пространство) записи, содержащиеся в файле ограничений, должны быть проверены на целостность действительного числа блоков и файлов, выделенных для пользователя. Для выполнения этой операции может быть использована команда «quotacheck».

Данную команду необязательно выполнять для неподмонтированных файловых систем или в файловых системах, на которых отключено ограничение на дисковое пространство (quotas). Для проверки файловой системы на число блоков, используемых пользователем, а так же для установки и изменения всех файлов ограничений (quotafiles), выполните команду:

```
quotacheck -avug
```

Вы можете вставить данную команду в один из rc-скриптов и запускать ее на файловой системе так же как и fsck, только тогда, когда не установлен флаг «fastreboot». Данная программа не поддерживает одновременную параллельную проверку нескольких файловых систем. Для того чтобы включить систему ограничений на дисковое пространство на вашем компьютере, добавьте строку

```
/usr/bin/quotaon -avug
```

в один из ваших /etc/rc файлов. Данная команда будет включать поддержку ограничений дискового пространства во время загрузки вашей системы.

Только суперпользователь может использовать команду «quota» для проверки используемых ограничений любого пользователя и команду «repquota» для проверки используемого пространства и ограничений для всех пользователей на данной файловой системе.

Для того чтобы получить информацию об ограничениях, наложенных на всех пользователей, необходимо выполнить команду:

```
repquota -ua
```

После чего на экране можно будет видеть таблицу, в которой указаны ограничения, наложенные на каждого пользователя. Содержимое экрана после выполнения вышеприведённой команды может иметь вид:

```
Block limits File limits
User used soft hard grace used soft hard grace
root -- 487082 0 0 12147 0 0
daemon -- 399 0 0 2 0 0
news -- 626 0 0 46 0 0
```

Для получения информации о группах необходимо выполнить команду:

```
repquota -ga
```

Для редактирования ограничения дискового пространства необходимо использовать программу «edquota». Для редактирования ограничений дискового пространства для конкретного пользователя необходимо использовать программу «edquota» с опцией «-u», а для группы необходимо использовать ту же программу, но с опцией «-g». Редактировать необходимо только число, которое следует за словом «hard» или «soft». Например, после выполнения команды:

```
edquota -u cola
```

содержимое экрана может иметь вид:

```
Quotas for user cola:
/dev/hdb3: blocks in use: 345, limits (soft = 1, hard = 0)
inodes in use: 94, limits (soft = 0, hard = 0)
```

После чего вы можете изменить значения, указанные после слов «hard» и «soft». При редактировании используется редактор, который указывается в переменной окружения EDITOR.

Для каждой файловой системы, имеющей ограничение на дисковое пространство, вы видите две строки. Слово «soft» означает, что на данную файловую систему наложено «мягкое» ограничение (softlimit), при этом пользователь или группа имеют некоторый интервал времени (grace period), в течении которого они могут превысить указанное значение. Данный интервал времени можно изменить с помощью команды:

```
edquota -t
```

Например, после выполнения данной команды, содержимое экрана может иметь вид:

```
Time units may be: days, hours, minutes, or seconds
Grace period before enforcing soft limits for users:
/dev/hdb3: block grace period: 10 minutes, file grace period:
10 minutes
```

После чего вы можете изменить значения, указанные по умолчанию. По умолчанию используется значение, определённое в файле LINUX quota.h. Если пользователь или группа не удалят лишних файлов в течение указанного периода времени, то данные файлы переходят в разряд «жестких» ограничений (hardlimit). Слово «hard» означает, что на данную файловую систему наложено «жёсткое» ограничение. Жёсткое ограничение является максимальным значением, которое может иметь пользователь или группа на данной файловой системе. Пользователь или группа не могут иметь больше файлов или inodes, чем указано в «hard».

Строка

```
/dev/hdb3: blocks in use: 345, limits (soft = 1, hard = 0)
```

сообщает, какое число блоков может быть выделено для данного пользователя или группы.

Строка

```
inodes in use: 94, limits (soft = 0, hard = 0)
```

сообщает, какое число inode (файлов, устройств, поименованных каналов(pipes) и т.д.) может быть выделено для данного пользователя или группы.

В большинстве случаев вам приходится иметь группу пользователей, которые должны иметь одинаковые ограничения на дисковое пространство на указанной файловой системе. Самый быстрый путь отредактировать ограничения на дисковое пространство для этих пользователей заключается в создании прототипа одного из пользователей или группы для всех других. Вам нужно с помощью команды:

```
edquota -u <имя_пользователя/группы_который_станет_прототипом>
```

отредактировать ограничения на дисковое пространство, после чего с помощью команды:

```
edquota -p <имя_прототипа>*
```

отредактировать ограничения для всех оставшихся пользователей.

Например, для того чтобы в качестве ограничений дискового пространства (quota) для пользователя ola использовались ограничения, наложенные на пользователя cola, вам необходимо выполнить команду:

```
edquota -p cola ola
```

Для проверки ограничений дискового пространства для пользователя и группы используется программа quota.

Синтакс этой программы:

```
quota [-guqv]
quota [-qv] -u <username> ...
quota [-qv] -g <groupname> ...
```

Используйте опцию «-v» для вывода информации о:

- файловых системах, которые не имеют включёнными ограничения на дисковое пространство (quota),
- файловых системах, на которых вы уже установили систему ограничения дискового пространства, но на которых ещё не занят ни один блок.

Используйте опцию «-q» для просмотра файловых систем, на которых превышено значение «мягкого» (softlimit) и «жесткого» (hardlimit) ограничения.

Опция «-g» даёт вам возможность просмотреть все ограничения, которые наложены на группы, членом которых вы являетесь.

Например, с помощью команды:

```
quota -u cola
```

вы можете получить информацию об ограничениях дискового пространства, которые наложены на пользователя cola. Содержимое экрана после выполнения данной команды может иметь вид:

```
Disk quotas for user cola (uid 1002):
Filesystem blocks quota limit grace files quota limit grace
/dev/hdb3 345* 1 0 none 94 0 0
```

Если вы захотели запретить ограничения, наложенные на какого-то пользователя или на группу, то вам необходимо воспользоваться программой «edquota» для редактирования значений «hard» и «soft». Установите эти значения в 0. После чего для данного пользователя или группы не будет существовать ограничений на дисковое пространство, и он/она сможет создавать неограниченное количество файлов (ограниченное только дисковым пространством).

6.15 Ограничение ресурсов пользователя

Для ограничения ресурсов, доступных пользователю, используется конфигурационный файл /etc/security/limits.conf. Для включения лимитов необходимо добавить библиотеку PAM pam_limits.so в соответствующем модуле. Например:

```
session required pam_limits.so
```

в файле /etc/pam.d/login.

Для изменения лимитов редактируем файл /etc/security/limits.conf. Формат файла:

```
<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/пользователя можно использовать групповой символ * (для всех) и групповой символ % для wildcast'a групп.

6.16 Контроль целостности запускаемых компонентов программного обеспечения

Для контроля целостности компонентов ОС используется IMA. Компонент архитектуры измерения целостности (IMA) выполняет проверки целостности во время выполнения файлов с использованием хэшей, сравнивая их со списком допустимых хэшей.
Для осуществления контроля целостности компонентов ОС необходимо:
1. Установить утилиты для работы с IMA:

```
dnf install ima-manage openssl-gost-engine keyutils
```

2. Обновить пакет dracut:

```
dnf update dracut
```

3. Включить ГОСТ в openssl:

```
openssl-switch-config gost
```

4. Сгенерировать пару ключей, выполнив следующие действия:
- создать файл test-ca.conf со следующим содержимым:

```
— req . distinguished_name = req_distinguished_name
prompt = no
string_mask = utf8only
x509_extensions = v3_ca

[ req_distinguished_name ]
O = IMA-CA
CN = IMA/EVM certificate signing key
emailAddress = ca@ima-ca

[ v3_ca ]
basicConstraints=CA:TRUE
```

```
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid:always,issuer
```

Значения полей O, CN, emailAddress являются произвольными, можно заменить их на свои. Место расположения файла не играет роли, он нужен только для создания пары ключей один раз.

- сгенерировать приватный ключ:

```
   openssl genpkey -algorithm gost2012_512 -pkeyopt paramset:A
-out privkey_evm.pem
```

- создать запрос:

```
   openssl req -new -config test-ca.conf -md_gost12_512 -key
privkey_evm.pem -out cert.req
```

- отправить файл запроса в Техническую поддержку РЕД СОФТ. Получить публичный ключ (например, x509_evm.der). В случае получения сертификата в CRT-формате, его можно перевести в формат DER следующим образом:

```
   openssl x509 -outform der -in pub.crt -out x509_evm.der
```

5. Создать папку /etc/keys/ima/ и скопировать в нее открытый ключ x509_evm.der.

```
mkdir /etc/keys/ima/
cp x509_evm.der /etc/keys/ima/x509_evm.der
```

Секретный ключ «privkey_evm.pem» сохранить в каталоге, недоступном для всех пользователей, кроме администратора root, например:

```
mv privkey_evm.pem /root/privkey_evm.pem
```

6. Запустить утилиту ima-manage с ключом init:

```
ima-manage init
```

7. В файле /etc/ima-manage.conf изменить значение HASHALGO="sha256" на HASHALGO="streebog512".

Важно! Значение битности параметра HASHALGO должно совпадать с битностью сгенерированных ключей. Например, если приватный ключ был сгенерирован с параметром gost2012_256, то значение параметра HASHALGO в таком случае должен быть равен streebog256. ■

8. Запустить утилиту «ima-manage» с ключом «signfs» и указанием пути размещения секретного ключа:

```
ima-manage signfs /root/privkey_evm.pem
```

9. Запустить утилиту ima-manage с ключом «enforce»:

```
ima-manage enforce
```

Дождаться завершения работы утилиты и предложения перезагрузки. Согласиться на перезагрузку ЭВМ.

Для запуска сторонних файлов, не имеющих подписи, необходимо от имени администратора подписать файл:

```
evmctl ima_sign --hashalgo md_gost12_512 --key <путь_к_приватному_ключу_privkey_evm.pem> <путь_к_файлу>
```

Для включения режима журналирования (система не будет запрещать запуск неподписанных файлов, но будет записывать попытки их запуска) необходимо в файле /etc/default/grub изменить параметр ima_appraise=enforce на ima_appraise=log.

После этого следует обновить конфигурацию загрузчика ОС:

- для систем, использующих BIOS:

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

- для систем, использующих UEFI:

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

6.17 Блокирование файлов

В ряде случаев бывает необходимо защитить файлы, открытые кем-либо и используемые в текущий момент, от удаления любым другим пользователем, даже если он имеет на это права. Для поддержки этого функционала в РЕД ОС необходимо установить пакет fileprotect:

```
dnf install fileprotect
```

После установки пакета и перезагрузки ОС, блокируются попытки удаления файлов, если в момент обращения к файлу субъекта доступа (процесса) он используется другим субъектом доступа (процессом). Никаких дополнительных настроек не требуется.

6.18 Очистка памяти

В состав ОС входят утилиты «shred», «wipe», «nwipe» и набор консольных утилит Secure-Delete.

6.18.1 shred

Переписывает несколько раз указанные файлы для того, чтобы сделать более сложным восстановление даже с использованием очень дорогого оборудования.

Использование:

```
shred <КЛЮЧ>... <ФАЙЛ>...
```

Если <ФАЙЛ> задан как «–», измельчать стандартный вывод.

Ключи утилиты shred описаны в таблице.

Ключ	Описание ключа
`-f, --force`	изменять права, разрешая запись, если необходимо.
`-n, --iterations=N`	переписать N раз вместо (3) по умолчанию.
`--random-source=ФАЙЛ`	получать случайные числа из <ФАЙЛА> (по умолчанию /dev/urandom).
`-s, --size=N`	очистить N байт (возможны суффиксы вида K, M, G).
`-u`	обрезать и удалять файл после перезаписи.
`--remove [=КАК]`	подобно «-u», но задаётся КАК удалять.
`-v, --verbose`	показывать ход выполнения.
`-x, --exact`	не округлять размеры файлов до следующего целого блока (по умолчанию для не простых файлов).
`-z, --zero`	перезаписать в конце нулями, чтобы скрыть измельчение.
`--help`	показать эту справку и выйти.
`--version`	показать информацию о версии и выйти.

Удаляет <ФАЙЛ(ы)>, если указан --remove (-u). По умолчанию файлы не удаляются, так как часто обрабатываются файлы-устройства вроде /dev/hda, а такие файлы не надо удалять.

Необязательным параметром КАК задаётся способ удаления каталога:

unlink – использовать стандартный вызов unlink.

wipe – также, сначала испортить байты имени.

wipesync – также, синхронизировать каждый испорченный байт на диске.

Режим по умолчанию — «wipesync».

6.18.2 wipe

wipe — это небольшая программа для безопасного стирания файлов с магнитных носителей.

Внимание! Wipe надёжно работает только в магнитной памяти, следовательно, для твердотельных дисков (памяти) необходимо использовать другие

методы. ■

Использование:

```
wipe <опции> <файлы>...
```

Основные опции утилиты wipe приведены в таблице.

Опция	Значение опции
`-a`	прервать при ошибке;
`-b <buffer-size-lg2>`	установить размер индивидуального буфера ввода/ вывода, указав его логарифм по основанию 2. Могут быть выделены до 30 этих буферов;
`-c`	делать chmod() на защищённых от записи файлах;
`-D`	следовать символическим ссылкам (конфликтует с «-r»);
`-e`	использовать точный размер файла: не округлять размер файла для стирания возможного мусора, остающегося на последнем блоке;
`-f`	форсировать, т. е. не спрашивать подтверждения;
`-F`	не пытаться стирать имена файлов;
`-h`	показать справку;
`-i`	информативный (вербальный) режим;
`-k`	сохранить файлы, т. е. не удалять их после перезаписи;
`-l <длина>`	установить длину стирания на <длину> байтов, где <длина> это целое число, за которым следует K (Kilo:1024), M (Mega:K^2) или G (Giga:K^3);
`-M (l|r)`	установить алгоритм PRNG для заполнения блоков (и порядка проходов): • l – использовать вызов библиотеки random(); • a – использовать алгоритм шифрования arcfour.
`-o <сдвиг>`	установить сдвиг очистки на <сдвиг>, где <сдвиг> имеет тот же формат, что и <длина>;
`-P <проходы>`	установить количество проходов для очистки имени файла. По умолчанию это 1;
`-Q <количество>`	установить количество проходов для быстрой очистки;
`-q`	быстрая очистка, менее безопасная, по умолчанию четыре случайных прохода;
`-r`	рекурсия по каталогам, по символическим ссылкам не будет переходов;

Опция	Значение опции
-R	установить устройство рандомизации (или команду сидов рандомизации -S c);
-S (r|c|p)	метод рандомизации сидов: • r – считывать с устройства рандомизации (надёжно); • c – считывать из вывода команды рандомизации сидов; • p – использовать pid(), clock() и т.д. (самый слабый вариант).
-s	тихий режим — подавлять весь вывод;
-T <попытки>	установить максимальное число попыток для свободного поиска имени файла (по умолчанию это 10);
-v	показать информацию о версии;
-Z	не пытаться стирать имя файла;
-X <число>	пропустить это число проходов (полезно для продолжения операции очистки);
-x <pass1,pass2, ...>	задать очередь проходов.

6.18.3 Secure-Delete

Secure-Delete – инструмент для стирания файлов, освобождения дискового пространства, swap'а и памяти. Этот набор консольных утилит предназначен для безвозвратного удаления данных и удаления остаточной информации. В своей работе использует Метод Гутмана.

В своём составе имеет набор из четырёх консольных утилит: srm, sfill, sswap, sdmem.

Srm - выполняет безопасную перезапись/переименование/удаление целевого файла(ов). При использовании ключа «-z» на последнем цикле удаления пишутся нули вместо случайных данных. По умолчанию включён безопасный режим (38 записей). Процесс безопасного удаления данных srm выглядит следующим образом:

- 1 проход с 0xff;
- 5 случайных проходов (если доступно, используется /dev/urandom для безопасного secure RNG);
- 27 проходов со специальными значениями, определёнными разработчиком;
- 5 случайных проходов (если доступно, используется /dev/urandom для безопасного RNG);
- переименование файлов на случайные значения;
- обрезка файлов.

В качестве дополнительной меры безопасности, файл открывается в режиме O_SYNC и после каждого прохода делается вызов fsync(). srm записывает 32k блоков в целях быстродействия, заполняя буферы дисковых кэшей, для прину-

дительного их сброса и перезаписи старых данных, принадлежащих стираемому файлу, новыми данными.

Основные опции утилиты Srm приведены в таблице.

Опция	Значение опции
`-d`	игнорирует специальные файлы «.» и «..»;
`-f`	использует не безопасный, быстрый режим: без режимов /dev/urandom и synchronize;
`-l`	снижение безопасности (двойное использование для полной не безопасности);
`-r`	использует рекурсивный режим для удаление всех подкаталогов;
`-v`	подробный вывода экран;
`-z`	затирание нулями на последнем шаге (вместо случайных чисел).

Sfill – аналогична предыдущей, только обрабатывает свободное место на диске, зачищая следы данных. sfill выполняет безопасную перезапись свободного места, занимаемого директорией, и всех свободных инодов заданной директории. По умолчанию безопасный режим (38 записей).

Процесс безопасного удаления данных sfill выглядит следующим образом:

- 1 проход с 0xff;
- 5 случайных проходов (если доступно, используется /dev/urandom для безопасного secure RNG);
- 27 проходов со специальными значениями, определёнными разработчиком;
- 5 случайных проходов (если доступно, используется /dev/urandom для безопасного RNG).

После этого создаётся так много временных файлов, насколько это возможно, чтобы очистить свободное пространство индексных дескрипторов (inode). Когда больше невозможно создать временных файлов, они удаляются и sfill завершает свою работу.

Основные опции утилиты Sfill приведены в таблице.

Опция	Значение опции
`-f`	использует не безопасный, быстрый режим: без режимов /dev/urandom и synchronize;
`-l`	снижение безопасности (двойное использование для полной не безопасности);
`-r`	использует рекурсивный режим для удаление всех подкаталогов;
`-v`	подробный вывода экран;
`-z`	затирание нулями на последнем шаге (вместо случайных чисел).

Sswap – зачищает раздел подкачки /swap, если он присутствует в системе. Перед использованием необходимо отключить swap. По умолчанию безопасный режим (38 записей).

Процесс безопасного удаления данных Sswap выглядит следующим образом:

- 1 проход с 0xff;
- 5 случайных проходов (если доступно, используется /dev/urandom для безопасного secure RNG);
- 27 проходов со специальными значениями, определёнными разработчиком;
- 5 случайных проходов (если доступно, используется /dev/urandom для безопасного RNG).

Основные опции утилиты Sswap приведены в таблице.

Опция	Значение опции
`-f`	использует не безопасный, быстрый режим: без режимов /dev/urandom и synchronize;
`-l`	снижение безопасности (двойное использование для полной не безопасности);
`-r`	использует рекурсивный режим для удаление всех подкаталогов;
`-v`	подробный вывода экран;
`-z`	затирание нулями на последнем шаге (вместо случайных чисел).

Sdmem – уничтожает следы данных в оперативной памяти. sdmem выполняет безопасную перезапись оперативной памяти (RAM), поскольку содержимое памяти может быть восстановлено даже после отключения. По умолчанию безопасный режим (38 записей).

Основные опции утилиты Sdmem приведены в таблице.

Опция	Значение опции
`-f`	использует не безопастный, быстрый режим: без режима /dev/urandom;
`-l`	снижение безопасности (двойное использование для полной не безопасности);
`-v`	подробный вывода экран.

6.19 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;

Опция	Значение опции
-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.

6.20 Резервирование данных

В состав дистрибутива ОС входит утилита резервного копирования rsync. rsync (англ. Remote Synchronization) — программа, которая выполняет синхронизацию файлов и каталогов в двух местах с минимизированием трафика, используя сжатие данных при необходимости. Важным отличием rsync от многих других программ/протоколов является то, что зеркалирование осуществляется одним потоком в каждом направлении (а не по одному или несколько потоков на каждый файл). rsync может копировать или отображать содержимое каталога и копировать файлы, опционально используя сжатие и рекурсию.

rsyncd — демон, реализующий протокол rsync. По умолчанию использует TCP-порт 873.

Базовый синтаксис:

```
rsync <опции> <источник> <место_назначения>
```

Наиболее полезные опции rsync приведены в таблице:

Опция	Значение опции
-v	подробный режим;
-r	копировать данные рекурсивно (но без сохранения информации о времени изменения файлов и правах доступа);
-a	режим архивирования, позволяет копировать данные рекурсивно с сохранением симлинков, правах доступа на файлы/-каталоги и другую информацию);
-z	сжатие данные;
-h	вывод данных в human-readable формате.

6.21 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'a групп.

Параметры вступают в силу сразу после перелогина пользователя. Что бы посмотреть какие действуют ограничения, используйте команду ulimit:

```
   ulimit -aH
```

6.22 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.

6.23 Принудительное завершение сеанса пользователя

Для отслеживания времени бездействия пользователя и завершения сеанса используется скрипт /usr/bin/gui-kick. Сценарий использует системную переменную «xprintidle», которая показывает время бездействия пользователя в графическом сеансе в миллисекундах. Есть две границы «предупреждение» и «завершение сеанса». После превышения границы «предупреждение», пользователю через службу notify-send формируется предупреждение о скором завершении сеанса и выводится в графическом сеансе соответствующее сообщение. После превышения второго заданного в сценарии максимального значения времени бездействия, сценарий принудительно завершает работу графического сеанса.

Для отслеживания бездействия пользователя необходимо добавить новое задание в демон cron:

```
   echo "* * * * * root /usr/bin/gui-kick" >> /etc/crontab
```

Установить интервал бездействия пользователя (в секундах) можно, присваивая новое значение переменной TIMEOUT в скрипте /usr/bin/gui-kick.

За 60 секунд до завершения сессии выводится системное уведомление с предупреждением о том, что по истечении 1 минуты завершится сессия, и чтобы этого избежать, необходимо предпринять какие-либо действия. Если в течение следующих 60 секунд пользователь по-прежнему не предпринял никаких действий, сессия пользователя завершается принудительно.

Управление пользователями

7. Управление пользователями

7.1 Общая информация

РЕД ОС – 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.

Внимание! В связи с тем, что в дистрибутивах РЕД ОС большинство привилегированных системных утилит имеют не SUID-, а SGID-бит, будьте предельно внимательны и осторожны в переназначении групповых прав на системные каталоги. ■

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», затем введён надёжный пароль:

```
useradd test1
passwd test1
```

Смена пароля для пользователя test1:

```
Новый пароль:
НЕУДАЧНЫЙ ПАРОЛЬ: СЛИШКОМ короткий
НЕУДАЧНЫЙ ПАРОЛЬ: СЛИШКОМ простой
Повторите ввод нового пароля:
passwd: все токены проверки подлинности успешно обновлены.
```

В результате описанных действий в системе появился пользователь test1 с некоторым паролем. Если пароль оказался слишком слабым с точки зрения системы, она об этом предупредит (как в примере выше). Пользователь в дальнейшем может поменять свой пароль при помощи команды «passwd –», но если он попытается поставить слабый пароль, система откажет ему (в отличие от root) в изменении.

В дистрибутивах ОС для проверки паролей на слабость используется модуль PAM «passwdqc».

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 <идентификатор>`	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.

Опция	Значение опции
`-g <группа>`	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 <группа> [[, <группа>] ...]`	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 <NGROUPS_MAX> - 1, поскольку общее количество дополнительных групп для пользователя плюс основная группа не должно превосходить <NGROUPS_MAX>. Эта опция игнорируется, если новое регистрационное имя будет администрироваться сетевой информационной службой (NIS).
`-d <каталог>`	Начальный каталог (home directory) нового пользователя. Длина этого поля не должна превосходить 256 символов. По умолчанию используется HOMEDIR/<рег_имя>, где HOMEDIR - базовый каталог для начальных каталогов новых пользователей, а <рег_имя> - регистрационное имя нового пользователя.
`-s <shell>`	Полный путь к программе, используемой в качестве начального командного интерпретатора для пользователя сразу после регистрации. Длина этого поля не должна превосходить 256 символов. 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 <комментарий>`	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. Длина этого поля не должна превосходить 128 символов.
`-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.

Опция	Значение опции
`-k <skel_dir>`	Копирует содержимое каталога <skel_di> в начальный каталог нового пользователя, вместо содержимого стандартного «скелетного» каталога /etc/skel. Каталог <skel_dir> должен существовать. Стандартный «скелетный» каталог содержит стандартные файлы, определяющие среду работы пользователя. Заданный администратором каталог <skel_dir> может содержать аналогичные файлы и каталоги, созданные для определённой цели.
`-f <inactive>`	Максимально допустимое количество дней между использованиями регистрационного имени, когда это имя ещё не объявляется недействительным. Обычно в качестве значений используются положительные целые числа.
`-e <expire>`	Дата, начиная с которой регистрационное имя больше нельзя будет использовать; после этой даты никакой пользователь не сможет получить доступ под этим регистрационным именем. (Эта опция удобна при создании временных регистрационных имён.) 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 <passgen>`	Указывает, что поле FLAG в файле /etc/shadow должно быть установлено в указанное значение. К этому полю обращается команда «passwd», чтобы определить, действует ли для данного пользователя генератор паролей. Если опция «-p» явно не задана, проверяется запись <FORCED_PASS> в файле /etc/default/useradd, чтобы определить значение для соответствующего поля в /etc/shadow. Если записи <FORCED_PASS> нет в /etc/default/useradd, в соответствующем поле записи в /etc/shadow значения не будет. Если значение <FORCED_PASS> равно 1, запись в /etc/shadow получает значение 1. Если значение passgen не пустое и не является печатным символом ASCII, выдаётся диагностическое сообщение.
`-a <событие>`	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).

Опция	Значение опции
<рег_имя>	Строка печатных символов, задающая регистрационное имя для нового пользователя. В имени пользователя допускается латиница нижнего 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 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). Оно не имеет чётко определённого синтаксиса.

Тем не менее, демон «fingerd» предполагает, что в нём содержатся следующие элементы, разделённые запятыми:

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

С помощью утилиты «chfn» можно изменять эту информацию, а с помощью «finger» — узнать, например, полное имя любого пользователя в системе (или даже на другом компьютере сети).

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/<имя_пользователя>.

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, обычно vi), 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> <user>
```

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

Предел числа используемых ранее паролей.

Изменить в файле /etc/pam.d/system-auth следующий параметр:

```
   password sufficient pam_unix.so sha512 shadow nullok
try_first_pass use_authtok remember=5
```

Установить минимальную длину пароля.

Изменить в конфигурационном файле /etc/security/pwquality.conf параметр:

```
   minlen= -8
```

Установить минимальное число строчных букв в новом пароле.

Изменить в конфигурационном файле /etc/security/pwquality.conf параметр:

```
   lcredit = -1
```

Установить минимальное число заглавных букв в новом пароле.

Изменить в конфигурационном файле /etc/security/pwquality.conf параметр:

```
   ucredit = -1
```

Установить минимальное число цифр в новом пароле.

изменить в конфигурационном файле /etc/security/pwquality.conf параметр:

```
   dcredit = -1
```

Установить минимальное число спецсимволов в новом пароле.

Изменить в конфигурационном файле /etc/security/pwquality.conf параметр:

```
   ocredit = -1
```

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 <mindays> 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 <maxdays> на максимальное число дней, в течение которых пароль ещё действителен. Когда сумма <maxdays> и <lastday> 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 <lastday> 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 <expiredate> есть число дней с 1 января 1970. Дата также может быть указана в формате <ГГГГ-ММ-ДД> (или формат, используемый в вашем регионе). 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 <inactive> 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 <warndays> 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 [ ].

7.7 Роли пользователей

В ОС пользователи по умолчанию сопоставляются SELinux-пользователю <unconfined_u>. SELinux-пользователь <unconfined_u>, в свою очередь, сопоставлен ролям <unconfined_r> и <system_r>.

Обе роли <unconfined_r> и <system_r> сопоставляются доменам безопасности SELinux. Домены безопасности SELinux - это определённые окружения безопасности для процессов.

Неограниченный домен безопасности – <unconfined_t> – зарезервированное окружение для процессов, которые в значительной степени освобождены от ограничений SELinux. Роль <system_r> сопоставляется доменам безопасности для системных процессов.

SELinux-пользователь <unconfined_u> имеет доступ к роли <system_r>, чтобы иметь возможность запускать системные процессы в своих доменах безопасности. SELinux-пользователь <unconfined_u> работает в домене безопасности <unconfined_t> через роль <unconfined_r>, которой он сопоставлен.

Команда «semanage»позволяет добавлять, изменять и удалять сопоставления между ОС- и SELinux-пользователями, так же как и другие настройки, касающиеся управления SELinux.

Чтобы с помощью команды «semanage» посмотреть какому SELinux- пользователю по умолчанию сопоставлены пользователи ОС наберите:

```
semanage login -l | grep default
```

В примере выше пользователи по умолчанию сопоставлены SELinux- пользователю <unconfined_u>.

Чтобы изменить это и сопоставить пользователей по умолчанию ограниченному SELinux-пользователю с именем <user_u> просто наберите:

```
semanage login -m -s user_u "__default__"
```

В результате все новые пользователи будут по умолчанию сопоставляться ограниченному SELinux-пользователю <user_u>.

Вы можете переопределить данное сопоставление при добавлении пользователей командой «useradd» используя опцию «-Z». Эта опция определяет какому SELinux-пользователю должен быть сопоставлен пользователь ОС.

Например наберите:

```
useradd -Z guest_u joe
```

В результате будет создан новый пользователь с именем joe, который будет сопоставлен SELinux-пользователю <guest_u>, вместо определённого по умолчанию SELinux-пользователя.

Также для изменения сопоставления между пользователем и SELinux- пользователем может быть использована команда «usermod» с опцией «-Z».

Существует несколько предопределённых профилей SELinux-пользователей. Список данных профилей можно вывести командой «semanage», наберите:

```
semanage user -l
```

SELinux-пользователь <unconfined_u> это среда, в которой по умолчанию работают все пользователи. Этот пользователь в значительной степени освобождён от ограничений, накладываемых SELinux. Если Вы хотите повысить безопасность вашей системы, тогда не следует сопоставлять реальных пользователей, не пользователя root, SELinux-пользователю <unconfined_u>.

Вход пользователя root должен быть запрещён. Root должен иметь возможность войти с терминала только в случае крайней необходимости. Чтобы повысить безопасность использования root можно сопоставить пользователя root SELinux-пользователю <sysadm_u>.

Ролевой доступ SELinux используется для обеспечения возможности назначения нескольких ограниченных окружений SELinux одному SELinux-пользователю.

В SELinux роли пользователя (User Roles) - это домены пользователя или пользовательские домены (User Domains). Когда упоминаются роли, часто имеется в виду дополнительный (secondary) пользовательский домен пользователя SELinux. Часто роли, предназначенные для использования в качестве дополнительных доменов пользователя, отличаются от основных (primary) доменов. Это объясняется тем, что пользователи ОС фактически не используют для входа в систему роли, созданные как дополнительные домены пользователя. Вместо этого пользователи Linux, используя команду «sudo» или комбинацию команд «su» и «newrole», осуществляют переключение домена или доменный переход (Domain Transition) к их дополнительной роли.

Роль администратора - Sysadm.

Пример роли, предназначенной для использования в качестве основного домена пользователя. SELinux пользователь <sysadm_u> сопоставлен роли <sysadm_r>. Это сопоставление можно увидеть, выполнив команду:

```
semanage user -l | grep sysadm_u
```

По умолчанию пользователи ОС, сопоставленные пользователю SELinux <sysadm_u>, работают с ролью <sysadm_r>. В этом случае <sysadm_r> — это основной домен пользователя.

SELinux пользователь <staff_u> также сопоставлен роли <sysadm_r>. Основным пользовательским доменом пользователя SELinux <staff_u> является (роль) <staff_r>. Контексты по умолчанию, определённые в /etc/selinux/targeted/contexts/users/staff_u, определяют с какими пользовательскими доменами пользователи будут подключаться по умолчанию и с какими пользовательскими доменами пользователи могут войти, указав домен пользователя при входе.

По умолчанию SELinux пользователь <staff_u> входит в систему в домен пользователя <staff_t>. Роль <sysadm_r>, предназначенная для использования в качестве основного домена пользователя, может быть использована при входе, например, следующим образом:

```
ssh joe/sysadm_r@localhost.localdomain.tld
```

Также роль <sysadm_r> может быть использована при переходе домена выполнением команд «sudo» и «su» вместе с «newrole», как это предусмотрено для дополнительных доменов пользователей.

Пользовательский домен <sysadm_t> является окружением по умолчанию для SELinux-пользователя <sysadm_u> и предназначен для использования в качестве дополнительного окружения для SELinux пользователя <staff_u>, хотя даже SElinux пользователь <staff_u> может настроить подключаемый модуль аутентификации (Pluggable Authentication Module) <pam_selinux> для

использования <sysadm_t> в качестве его основного домена пользователя.

Роль <user_r> — пользователь системы.

Пользовательский домен, используемый только в качестве основного. Описание пользователя SELinux <user_u> приведено ранее.

Какие-то домены пользователей могут использоваться пользователями для входа в систему, потому что эти домены, например, имеют полномочия по доступу к домашней директории пользователя. Такие домены в предыдущей части назывались основными (primary) пользовательскими доменами. Другие домены созданы в качестве дополнительных. Пользователи, используя команды «sudo» или «su» вместе с «newrole», могут осуществить доменный переход (Domain Transition) к этим дополнительным доменам.

Дополнительные домены не могут использоваться для входа в систему. Далее будет продемонстрировано как создаётся такой дополнительный пользовательский домен. Целью является реализовать привилегированную роль SELinux, которой предоставлены полномочия по управлению DNS службой Bind.

Начнём с создания нового дополнительного домена пользователя с названием «bindadm», основанного на предустановленном домене SELinux <webadm_t>. Новый домен пользователя основан на двух файлах с исходным текстом политики SELinux (SELinux Source Policy):

- webadm.te;
- webadm.if.

Файлы исходных текстов политики SELinux с расширением ".te" называются Type Enforcement файлы. Файлы с расширением ".if" называются интерфейсными (Interface files). Файлы Type Enforcement содержат описания (Declarations) и политику (Policy), персональные или локальные для данного домена (Domain). Интерфейсные файлы содержат описания и политики, общие для этого домена и других доменов, желающими взаимодействовать с данным доменом.

```
   mkdir ~/bindadm; cd ~/bindadm
   echo "policy_module(bindadm, 0.0.1)" > bindadm.te
   echo "role bindadm_r;" >> bindadm.te
   echo "userdom_base_user_template(bindadm)" >> bindadm.te
   echo "allow bindadm_t self:capability { dac_override
dac_read_search kill sys_ptrace sys_nice " >> bindadm.te
   echo "files_dontaudit_search_all_dirs(bindadm_t)" >> bindadm.te
   echo "files_manage_generic_locks(bindadm_t) >> bindadm.te
   echo "files_list_var(bindadm_t)" >> bindadm.te
   echo "selinux_get_enforce_mode(bindadm_t)" >> bindadm.te
   echo "seutil_domtrans_setfiles(bindadm_t)" >> bindadm.te
   echo "logging_send_syslog_msg(bindadm_t)" >> bindadm.te
   echo "userdom_dontaudit_search_user_home_dirs(bindadm_t)" >>
bindadm.te
```

Это основное содержимое для привилегированного дополнительного домена пользователя. Исключена политика, специфичная для управления службой Apache.

Теперь добавим политику, специфичную для управления службой Bind.

Можно позаимствовать эту политику из исходных текстов политики для Bind. Как уже отмечалось, общая политика располагается в интерфейсных файлах. Это значит, что если мы хотим включить общую политику, относящуюся к Bind, можно посмотреть в соответствующем файле политики bind.if. Для включения этой политики в наш Type Enforcement файл требуется всего лишь добавить вызов этого интерфейса:

```
   echo "bind_admin(bindadm_t, bindadm_r)" >> bindadm.te
```

На этом Type Enforcement файл bindadm заканчивается. Далее необходимо обеспечить другим доменам возможность взаимодействия с нашим доменом <bindadm_t>. Реализуем эту часть политики, взяв за основу файл webadm.if.

```
   echo "## Bind administrator role" > bindadm.if
   echo "########################################" >> bindadm.if
   echo "## " >> bindadm.if
   echo "## Change to the bind administrator role." >> bindadm.if
   echo "## " >> bindadm.if
   echo '## ' >> bindadm.if
   echo "## " >> bindadm.if
   echo "## Role allowed access." >> bindadm.if
   echo "## " >> bindadm.if
   echo "## " >> bindadm.if
   echo "## " >> bindadm.if
   echo "#" >> bindadm.if
   echo "interface(\`bindadm_role_change',\`" >> bindadm.if
   echo " gen_require(\`" >> bindadm.if
   echo " role bindadm_r;" >> bindadm.if
   echo " ')" >> bindadm.if
   echo " allow \$1 bindadm_r;" >> bindadm.if
   echo "')" >> bindadm.if
```

Позже общая политика администратора Bind (Bind Administrator Shared policy) будет использоваться в нашем основном специально созданном пользовательском домене SELinux. Следующим шагом будет создание специализированного пользовательского домена SELinux для обслуживания Bind, разрешающего данному домену осуществлять переход к роли <bindadm_r> вызовом интерфейса <bindadm_role_change>. Создаваемый домен SELinux будет основан на пользовательском домене <staff_t>, описание политики смотрим в файле staff.te

```
   echo "policy_module(bindguy, 0.0.1)" > bindguy.te
   echo "role bindguy_r;" >> bindguy.te
   echo "userdom_unpriv_user_template(bindguy)" >> bindguy.te
   echo "sudo_role_template(bindguy, bindguy_r, bindguy_t)" >>
bindguy.te
```

```
   echo "ssh_role_template(bindguy, bindguy_r, bindguy_t)" >>
bindguy.te;
   echo "kernel_read_ring_buffer(bindguy_t)" >> bindguy.te
   echo "kernel_getattr_core_if(bindguy_t)" >> bindguy.te
   echo "kernel_getattr_message_if(bindguy_t)" >> bindguy.te
   echo "kernel_read_software_raid_state(bindguy_t)" >> bindguy.te
   echo "auth_domtrans_pam_console(bindguy_t)" >> bindguy.te
   echo "libs_manage_shared_libs(bindguy_t)" >> bindguy.te
   echo "seutil_run_newrole(bindguy_t, bindguy_r)" >> bindguy.te
   echo "netutils_run_ping(bindguy_t, bindguy_r)" >> bindguy.te
   echo "domain_read_all_domains_state(bindguy_t)" >> bindguy.te
   echo "domain_getattr_all_domains(bindguy_t)" >> bindguy.te
   echo "domain_obj_id_change_exemption(bindguy_t)" >> bindguy.te
   echo "files_read_kernel_modules(bindguy_t)" >> bindguy.te
   echo "kernel_read_fs_sysctls(bindguy_t)" >> bindguy.te
   echo "modutils_read_module_config(bindguy_t)" >> bindguy.te
   echo "modutils_read_module_deps(bindguy_t)" >> bindguy.te
   echo "miscfiles_read_hwdata(bindguy_t)" >> bindguy.te
   echo "term_use_unallocated_ttys(bindguy_t)" >> bindguy.te
```

Чтобы разрешить домену <bindguy_t> осуществлять переход в ограниченное окружение SELinux <bindadm_t>, добавим вызов интерфейса <bindadm_role_change>, определённого в нашем интерфейсном файле bindadm.if:

```
   echo "bindadm_role_change(bindguy_r)" >> bindguy.te
```

Можно также автоматически создать сопоставление пользователя SELinux с именем <bindguy_u> ролям <bindguy_r>, <bindadm_r> и <system_r>. Роль <system_r> включается, чтобы <bindadm_t> мог остановить, запустить и перезапустить системную службу bind.

```
   echo "gen_user(bindguy_u, user, bindguy_r system_r bindadm_r, s0,
s0 - mls_systemhigh, mcs_allcats)" >> bindguy.te
```

Чтобы программы входа знали, что bindguy допустимый пользователь, добавим контексты по умолчанию для этого пользователя, взяв за основу контексты по умолчанию пользователя <staff_u>:

```
   cp /etc/selinux/targeted/contexts/users/staff_u
   /etc/selinux/targeted/contexts/users/bindguy_u
   sed -i 's/staff/bindguy/g'
   /etc/selinux/targeted/contexts/users/bindguy_u
```

Теперь можно собрать и установить обе политики bindguy и bindadm:

```
   make -f /usr/share/selinux/devel/Makefile
   semodule -i bindguy.pp bindadm.pp
```

Далее командой «useradd», опцией «-Z» и параметром <bindguy_u> можно создать нового пользователя Linux:

```
useradd -Z bindguy_u bindguy
```

Для того, чтобы разрешить пользователю bindguy использовать команду «sudo» для автоматического перехода к пользователю root и SELinux-домену <bindadm_t>, добавим в /etc/sudoers:

```
echo "bindguy ALL=(ALL) ROLE=bindadm_r TYPE=bindadm_t ALL" >>
/etc/sudoers
```

При входе пользователя bindguy в систему он оказывается в пользовательском домене <bindguy_t>, основанном на домене <staff_t>. Домен <bindguy_t> может осуществлять переход в домен <bindadm_t>, используемый с правами root и позволяющий управлять службой Bind.

Например, выполнив следующую команду, пользователь bindguy сможет перезапустить службу bind:

```
systemctl restart named
```

Также пользователю bindguy разрешается редактировать различные конфигурационные файлы и файлы информационного наполнения Bind. При этом, например, пользователь bindguy не имеет прав изменить пароль пользователя root.

Подведём итог. Процессы и файлы маркируются метками – контекстом SELinux, который содержит информацию: пользователь SELinux, роль, тип и уровень (опционально). Когда SELinux включён, вся эта информация используется для принятия решения о предоставлении доступа.

В контексте SELinux используется следующий синтаксис SELinux

```
<user>:<role>:<type>:<level>:<пользователь_SELinux >
```

<Пользователь_SELinux> – это сущность, определённая в политике, которая отвечает за определённый набор ролей и за определённый набор MLS-уровней. Каждый пользователь ОС сопоставлен пользователю SELinux посредством политики SELinux. Это позволяет пользователям наследовать ограничения, установленные на пользователей SELinux. Сопоставленные сущности пользователей SELinux используются в контексте SELinux для процессов в сессии, в порядке определения для каких ролей и уровней они применимы.

Роль – это часть модели безопасности Ролевого управления доступом Role-Based Access Control (RBAC). Роль – это атрибут RBAC. Пользователи SELinux авторитетны для ролей, а роль авторитетна для доменов. Определённая роль определяет, какие домены могут быть доступны для пользователей с этой ролью. Роль служит промежуточным звеном между доменами и пользователями SELinux. Роль, которой обладает пользователь, определяет, в какие домены может попасть пользователь - фактически, этот механизм управляет доступностью объектов. Таким образом, уменьшается риск, связанный с уязвимостью

повышения привилегий в системе. По умолчанию пользователям назначена неопределённая роль, не накладывающая ограничений на пользователя. Но есть и предопределённые роли пользователя и администратора системы, которые могут быть назначены администратором для повышения безопасности системы.

Тип – это атрибут Type Enforcement. Тип определяет домен для процессов и тип для файлов. Правила политики SELinux определяют, как типы взаимодействуют друг с другом, является ли тип доменом, получающим доступ к типу, или доменом, получающим доступ к другому домену. Доступ разрешается, только если существует определённое правило политики SELinux, позволяющее данное действие.

Контекст безопасности записывается в атрибуты файла (в файловой системе) и создаётся при установке SELinux (операция labeling). Уже присвоенный контекст безопасности может быть впоследствии изменён - операцией «transition».

Сеть и служебные программы

8. Настройка сети

Настройка сети в РЕД ОС производится с помощью программы Network Manager. Для ручной настройки сетевого адаптера необходимо в графическом интерфейсе рабочего стола РЕД ОС на системной панели в разделе апплетов нажать правой кнопкой мыши по значку апплета «Network Manager» (рисунок 8.1).

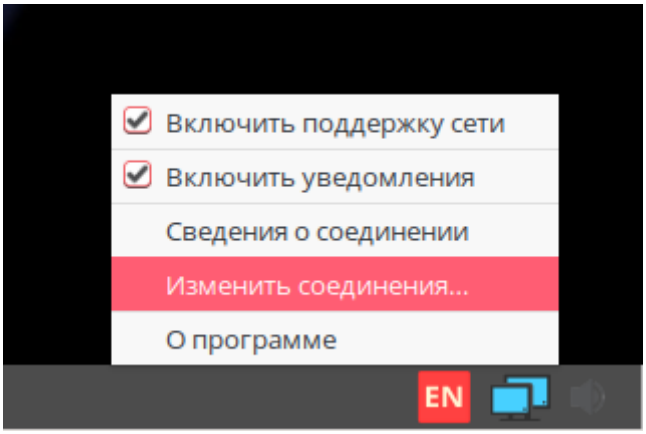

Рисунок 8.1 – Запуск утилиты настройки сетевых соединений

В открывшемся окне утилиты «Сетевые соединения» необходимо выделить нужный сетевой адаптер и нажать на кнопку с изображением шестеренки «Изменить выбранное соединение» внизу окна (рисунок 8.2).

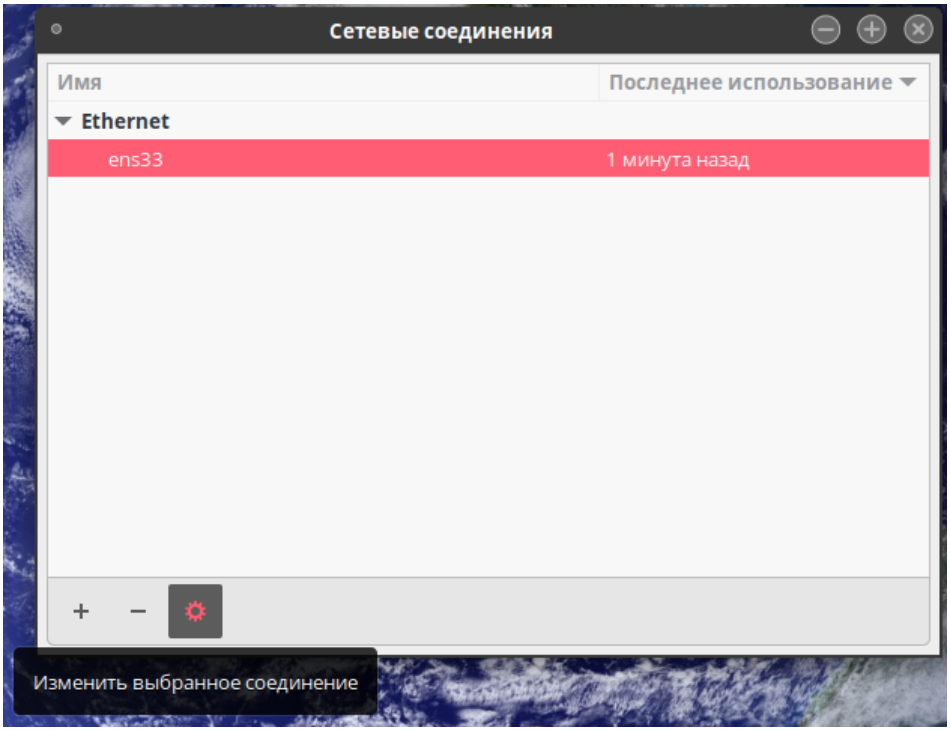

Рисунок 8.2 – Изменение сетевого адаптера

В открывшемся окне свойств сетевого адаптера (рисунок 8.3) можно установить свойства автоматического подключения - «Подключаться автоматически» и «Доступно всем пользователям», во вкладке «Параметры IPv4» задать вручную задания метрики сетевого подключения, нажав кнопку «Добавить».

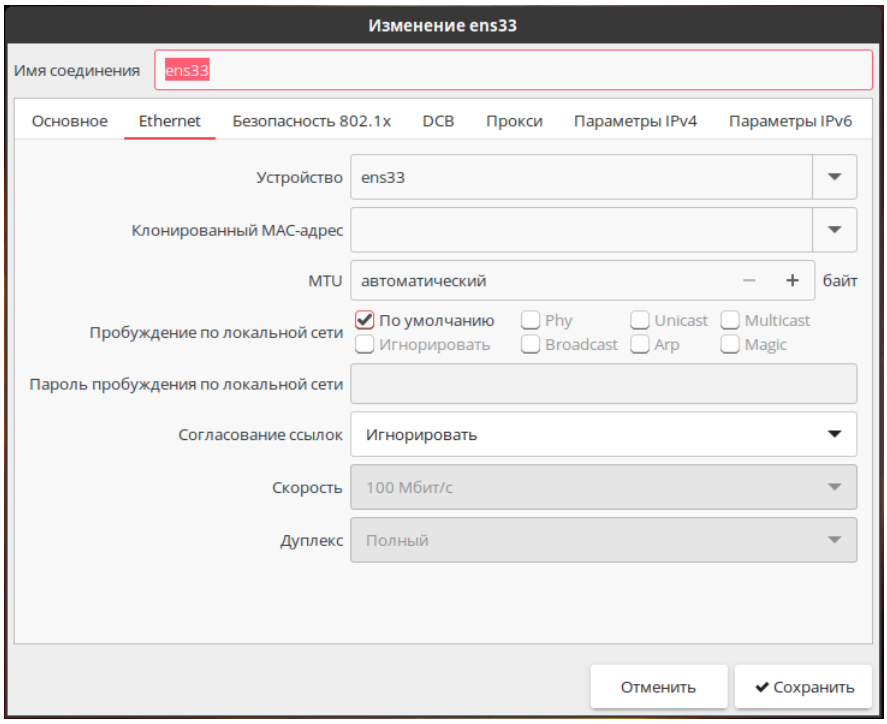

Рисунок 8.3 – Изменение свойств сетевого адаптера

Для подтверждения изменений необходимо нажать кнопку «Сохранить».

Для того чтобы изменения вступили в силу, необходимо переподключить сетевое соединение или перезапустить службу NetworkManager.

Для настройки сети в консоли нужно установить пакет net-tools.

Для просмотра сетевых настроек нужно ввести команду:

```
ifconfig
```

Или воспользоваться командой ip с параметром addr:

```
ip addr
```

Тут можно увидеть параметры и название сетевой карты.

Допустим, необходимо сменить или установить ip-адрес. Для этого переходим в директорию /etc/sysconfig/network-scripts и открываем на редактирование файл устройства. Этот файл имеет примерно следующее содержание:

```
cat ifcfg-eno16777736
TYPE=Ethernet
BOOTPROTO=dhcp
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=no
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_PEERDNS=yes
IPV6_PEERROUTES=yes
IPV6_FAILURE_FATAL=no
NAME=eno16777736
UUID=dc1636be-5281-4a07-8681-fcdc8b161c8c
DEVICE=eno16777736
ONBOOT=no
PEERDNS=yes
PEERROUTES=yes
```

Для установки статического IP-адреса нам необходимо на строчке BOOTPROTO установить

```
BOOTPROTO=none
```

и дописать:

- Указать ДНС:

```
DNS1=8.8.8.8
```

- Указать IP:

```
IPADDR0=172.16.0.30
```

- Указть нужную маску:

```
PREFIX0=24
```

- И шлюз по умолчанию:

```
GATEWAY0=172.16.0.1
```

- И чтобы сетевая карта активировалась при запуске ОС, необходимо в этом файле найти параметр ONBOOT и установить ему «yes».

Для немедленного применения изменений перезапустим сеть:

```
/etc/init.d/network restart
```

Предположим, что сетевая карта настроена на статический IP, а вы хотите получать настройки по DHCP. Переходим в папку /etc/sysconfig/network-scripts и открываем файл на редактирование с названием вашей сетевой карты. Находим там и удаляем параметры «DNS», «IPADDR», «PREFIX», «GATEWAY», а в параметре BOOTPROTO указываем значение «dhcp».

В файл настройки сетевой карты можно добавить столько DNS серверов, сколько требуется. Например:

```
DNS1=172.16.0.1
DNS2=8.8.8.8
DNS3=8.8.4.4
```

Проверить шлюз, по умолчанию установленный в системе:

```
netstat -nr
```

```
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
0.0.0.0 172.16.0.1 0.0.0.0 UG 0 0 0 eno16777736
172.16.0.0 0.0.0.0 255.255.255.0 U 0 0 0 eno16777736
```

Строка с «Destination 0.0.0.0» определяет адрес шлюза. Если у вас ее нет, либо в поле «Gateway» установлен неверный шлюз, то можно это изменить. Устанавливаем шлюз по-умолчанию:

```
route add default gw 172.16.0.1
```

Если вы не сменили при установке имя сервера или вы хотите его изменить, то сделать это можно следующим образом. Для начала проверим, какой hostname у нас установлен:

```
hostname
```

Допустим, мы хотим изменить имя: server.work, для этого необходимо отредактировать файл /etc/hostname или использовать команду:

```
hostnamectl set-hostname name
```

Для смены hostname перезагрузка не требуется.

После того, как TCP-пакеты были переданы ОС, среди других действий, их обрабатывают два файла - /etc/hosts.allow и /etc/hosts.deny. Эти два файла работают так же, как и стандартные правила брандмауэра. Первоначально демон tcpd обрабатывает пакеты пропуская их через содержимое файла hosts.allow, а затем через файл hosts.deny.

Синтаксис этих файлов:

```
<служба>: <IP-адрес> или <имя_хоста>
```

Так, например, если необходимо блокировать все smtp-пакеты, идущие от test.ru, необходимо ввести в файл hosts.deny следующую строчку:

```
smtp: test.ru
```

Мы можем также указать вместо имени хоста его IP-адрес (лучше всего физический):

```
smtp: 192.168.1.10
```

Можно использовать знак «.» в строке адреса, для обозначения сети. К примеру, если мы хотим разрешить доступ по http из всей сети 192.168.10.0/24, то мы должны написать в hosts.allow следующее:

```
http: 192.168.10
```

Также мы можем использовать этот знак, если хотим разрешить доступ к какой-либо службе всем компьютерам нашего домена, указав в hosts.allow следующее:

```
http: .test.ru
```

9. Служебные программы

9.1 Служба xinetd

Служба xinetd запускает процессы, которые предоставляют различные сервисы сети Интернет. В отличие от сервисов, которые стартуют во время инициализации системы и пребывают в бездействии в ожидании запросов, xinetd представляет собой только один процесс, слушающий на всех портах сервисов, перечисленных в файле конфигурации xinetd.conf. Когда приходит запрос, xinetd запускает соответствующий сервер. По причине такой работы xinetd называют ещё супер-сервером.

Сервисы, перечисленные в конфигурационном файле xinetd можно разделить на две группы.

Сервисы из первой группы называются multi-threaded, и на каждый новый запрос запускается новый серверный процесс. Для таких сервисов xinetd продолжает слушать сеть на соответствующем порту, ожидая новых запросов и готовый породить новый процесс.

В другую группу включаются сервисы, службы которых в состоянии обрабатывать новые соединения. Такие сервисы называются single-threaded, и xinetd прекращает обработку новых запросов до тех пор, пока серверный процесс не завершит свою работу. Сервисы в этой группе обычно datagram-based.

Итак, причиной существования супер-сервера является факт сохранения системных ресурсов за счёт не запуска множества серверных процессов, которые, возможно, будут бездействовать большую часть своей жизни. Полностью соответствуя назначению запускать требуемые сервисы, xinetd осуществляет так же функции контроля доступа и регистрации событий. Кроме того, xinetd не ограничен сервисами, перечисленными в файле /etc/services. Можно использовать xinetd для запуска сервисов специального назначения.

9.1.1 Параметры

Параметр	Значение параметра
-d	Активирует режим отладки. Этот параметр выводит много отладочной информации и позволяет отладить работу xinetd.
-syslog <syslog _facility>	Этот параметр разрешает регистрацию событий, используя syslog. Сообщения, производимые службой xinetd, регистрируются, используя одну из syslog_facility. Поддерживаются следующие facility: daemon, auth, user, local [0-7]. Эта опция не действует, если активирован режим отладки, так как при отладке все сообщения посылаются на консоль.
-filelog <logfile>	Производимые xinetd сообщения помещаются в указанный файл. Новые сообщения всегда добавляются к существующим. Если файл не существует, то он создаётся. Опция не действует в режиме отладки.
-f <config_file>	Определяет положение файла конфигурации xinetd. По умолчанию - это /etc/xinetd.conf.
-pid	Идентификатор процесса (PID) выводится на устройство стандартной ошибки. Не действует в режиме отладки.
-loop <rate>	Этот параметр устанавливает ограничение скорости запускания серверов, указывается число запускаемых серверов в секунду. При превышении указанного значения запуск новых процессов блокируется, и сервис становится временно недоступным. Значение этого параметра определяется производительностью компьютера. По умолчанию - 10.
-reuse	Если указан этот параметр, то xinetd устанавливает <SO_REUSEADDR> для сокета, используемого сервисом. Это позволяет использовать адрес, даже если активны программы, которые уже его используют. Это может случиться, когда предыдущая копия xinetd пытается запустить сервер, который уже выполняется. Параметр не эффективен с RPC-сервисами.
-limit <proc_limit>	Этот параметр устанавливает предел на число одновременно выполняющихся процессов, запущенных службой xinetd. Использование параметра позволяет предотвратить переполнение таблицы процессов.
-logprocs <limit>	Параметр устанавливает максимальное значение одновременно выполняемых процессов, запущенных по запросу удалённого userid.
-shutdownprocs <limit>	Параметр устанавливает предельное число одновременно выполняющихся серверов сервиса shutdown (concurrently running servers for service shutdown).

Параметр	Значение параметра
-cc <interval>	Параметр задаёт промежуток времени в секундах, через который xinetd производит периодическую проверку внутренней целостности.

Параметры <syslog> и <filelog> взаимно исключающие. Если ничего не указано, то по умолчанию используется syslog daemon facility. Не путайте сообщения xinetd с сообщениями, относящимися к функциям регистрации событий. Последние журналируются, только если это указано в файле конфигурации.

9.1.2 Управление xinetd

Служба xinetd выполняет определённые действия при получении определённых сигналов. Действия, ассоциированные с соответствующими сигналами, могут быть переопределены путём редактирования config.h и последующей компиляции.

```
SIGUSR1
```

Вызывает «мягкую» переконфигурацию. Это означает, что xinetd перечитывает файл конфигурации и подстраивается под изменения.

```
SIGUSR2
```

Вызывает «жёсткую» переконфигурацию. Это значит то же самое, что и «мягкая» переконфигурация, за исключением того, что все серверы для сервисов, которые стали недоступны, принудительно завершаются. Контроль доступа заново выполняется для выполняющихся серверных процессов путём проверки удалённого хоста, времени доступа и количества выполняющихся серверов. Если число выполняющихся процессов превышает заданное, произвольным образом выбранные процессы «убиваются», чтобы число оставшихся удовлетворяло поставленному условию. Также, если флаг «INTERCEPT» был сброшен или установлен, все серверы, обеспечивающие этот сервис, завершаются. Все это выполняется для того, чтобы после «жёсткой» реконфигурации не осталось ни одного работающего процесса, обрабатывающего запросы с адресов, не соответствующих критериям контроля доступа.

Сигнал	Описание сигнала
SIGQUIT	приводит к завершению программы.
SIGTERM	завершает все выполняющиеся серверы перед завершением xinetd.
SIGHUP	приводит к генерации дампа внутреннего состояния (по умолчанию файл дампа /tmp/xinetd.dump).
SIGIOT	вызывает проверку внутренней целостности, чтобы проверить, что структуры данных, используемые программой, не повреждены. После завершения проверки xinetd генерирует сообщение о результате проверки.

При реконфигурации лог-файлы закрываются и вновь открываются. Это позволяет удалять старые логи.

9.1.3 Файлы

/etc/xinetd.conf – стандартный конфигурационный файл.
/var/run/xinetd.dump – стандартный файл дампа.

9.2 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. У служб «Uucp» и «News» обычно есть свои собственные «crontab», устраняющие необходимость в явном запуске «su» в рамках команды «cron».

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). Значения переменных HOME и SHELL можно переопределить директивами «crontab».

В дополнение к LOGNAME, HOME и SHELL «cron» может использовать переменную <MAILTO> в случаях, если в данном «crontab» была указана отправка почты. Если <MAILTO> определена (и не пуста), электронная почта отправляется указанному в переменной пользователю. Если <MAILTO> определена, но пустая, (MAILTO = ' '), электронная почта отправляться не будет.

В противном случае, почта посылается владельцу «crontab». Эта переменная полезна при запуске команд от псевдопользователей, для которых не определены почтовые адреса в системе.

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 РЕД ОС. 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 1-23
<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(8).
@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 * * * *».

9.3 Полноэкранный редактор vi

Редактор vi - универсальный полноэкранный текстовый редактор.

Имеющиеся на многих типах терминалов функциональные клавиши практически не используются. Если клавиатура терминала имеет стрелочные клавиши, то они используются, но в ограниченном контексте.

9.3.1 Режимы работы редактора

Ввод текста

В этом режиме все, что набирается на клавиатуре, отображается на экране терминала и запоминается в буфере редактора.

В данном режиме нет возможности осуществлять операции редактирования текста, за исключением стирания последнего набранного символа (с помощью комбинации клавиш «Ctrl+H»).

Командный режим

В этом режиме символы клавиатуры выполняют специальные функции (перемещение курсора, стирание частей текста, и т. д.), то есть функции редактирования.

В данном режиме набираемые команды не отображаются на экране.

Режим командной строки

Режим командной строки позволяет производить более глобальные операции с текстом: записывать отредактированный текст в файл, считывать новый файл, выходить из vi, производить настройку редактора, поиск по шаблону, а также осуществлять некоторые функции редактирования.

Команды отображаются в нижней части экрана (в командной строке редактора).

9.3.2 Ввод текста

Действие	Описание действия
«Return»	создаёт пустую строку и переводит курсор в ее начало;
«Ctrl+H»	уничтожает последний введённый символ (это действие не отображается на экране до выхода в командный режим);
«Ctrl+[» или «Esc»	переводят редактор в командный режим.

В режиме ввода текста стрелочная клавиатура не работает.

9.3.3 Команды

Команда	Описание команды
i	переход в режим набора методом вставки перед текущим символом;
a	переход в режим набора методом вставки за текущим символом;
R	переход в режим набора текста методом набивки.

9.3.4 Перемещение курсора

Действие	Описание действия
h, j, k, l	на один символ (одну строку), как показано стрелками;
^ или 0	в начало текущей строки;
$	в конец текущей строки;
w	на слово вправо;
b	на слово влево;
}	на параграф вперёд (параграф – это блок текста, отделённый пустой строкой);
{	на параграф назад;
[[	в начало текста;
]]	в конец текста.

Кнопки стрелочной клавиатуры также позволяют перемещаться по тексту.

9.3.5 Редактирование

Действие	Описание действия
dd	стирание текущей строки;
d <движение_курсора>	стирание от текущего положения курсора до нового, задаваемого символом перемещения курсора. Нажатие кнопок стрелочной клавиатуры не является движением курсора и не может использоваться в комбинированных командах;
J	слияние текущей строки со следующей;
u	отмена последней команды;
.	повтор последней команды;
:	переход в режим командной строки.

9.3.6 Командная строка

Опция	Значение опции
`:q или :q!`	выход из редактора без сохранения изменений;
`:x`	выход из редактора с записью, если файл был модифицирован;
`:w или w <filename> или w! <filename>`	запись файла и возвращение в командный режим;
`:e <filename> или :e! <filename>`	загрузка файла <filename>;
`:r <filename>`	добавить содержимое указанного файла к редактируемому сразу за текущей строкой;
`:set nu`	включить нумерацию строк;
`:set nonu`	отключить нумерацию строк;
`:!command`	выполнить команду UNIX не покидая редактора;
`:/<word>`	выполнить поиск слова <word> в тексте;
`:/`	повторить поиск слова <word> далее по тексту.

9.3.7 Блоки, буферы, окна редактирования. Повторители

Повторители

Командам и движениям курсора можно давать повторители (числа), например:

Команда	Описание команды
`2w`	передвинуть курсор на два слова вперёд;
`10l`	передвинуть курсор на десять символов вправо;
`d10l`	стереть десять символов справа от курсора;
`2d10l`	стереть двадцать символов справа от курсора;
`5J`	слить пять последующих строк в одну;
`4.`	повторить последнюю введённую команду четыре раза;

Буферы vi

Редактор имеет три типа буферов: буфер стирания (0-9), неименованный буфер и именованные буферы (a-z).

В буферы стирания автоматически заносятся стираемые элементы. В буфере «0» хранится последний стёртый элемент, в буфере «1» - предпоследний и т. д.

Занести в буфер:

Опция	Значение опции
yy	занести текущую строку в неименованный буфер;
y <движение_курсора>	занести указанный движением курсора блок текста в неименованный буфер;
"ayy	занести текущую строку в именованный буфер a;
"Ayy	добавить текущую строку к содержимому именованного буфера a;
"by10j	занести последующие 10 строк в именованный буфер b.

Вставить из буфера:

Команда	Описание команды
p	вставить в текущую позицию содержимое неименованного буфера;
"ap	вставить в текущую позицию содержимое именованного буфера a;
"1p	вставить в текущую позицию содержимое буфера стирания 1.

Многооконное редактирование

Редактировать сразу несколько файлов можно либо пользуясь командной редактора :e <filename>, либо указав все необходимые файлы в командной строке при вызове редактора (например: vi <file1> <file2> <file3>). В последнем случае вы двигаетесь по списку файлов с помощью команд:

Команда	Описание команды
:n	переход к следующему файлу в списке;
:rew	возврат к редактированию первого файла в списке.

Именованные буферы сохраняют своё содержимое при переходе к редактированию другого файла.

9.3.8 Открыть/создать файл

- vi mamapapa.txt – открыть один файл.
- vi mama.txt papa.txt – открыть файл mama.txt, после выхода из него открыть файл papa.txt.

Файл открывается в командном режиме с помощью команды vi. Здесь мы можем просмотреть файл, переместиться по его содержимому, стереть текст, но внести изменения или ввести текст в этом режиме нельзя.

Создание файла происходит при помощи той же команды. Создание файла происходит в момент сохранения.

Для открытия или создания нового файла в командном режиме необходимо выполнить команду:

```
:e <имя_файла>
```

Перед этим нужно сохранить предыдущий файл:

Команда	Описание команды
:w	сохраняет файл с существующим именем;
:sav <имя_файла>	«Сохранить как».

9.3.9 Дополнительные опции

Дополнительные возможности редактора:

Команда	Описание команды
G	перейти в конец файла
<number>G	перейти на конкретную строку <number>
:<number>	перейти на <number> строк вперёд
:set nu[mber]	отобразить слева нумерацию строк (^ nonu[mber] - спрятать нумерацию)
:set wrap	переносить длинные строки (:set nowrap - не переносить)
:syntax on/off	включить/выключить подсветку синтаксиса
:colorscheme <name>	задать цветовую тему (где <name> - имя темы, «TAB» работает как авто-дополнение)
/мама	поиск текста «мама» в файле
n	повторить поиск
:h или :help	список возможной помощи (:viusage, :exusage)
:set fileformat=dos :set fileformat=unix	привести концы строк в файле к виду dos или unix, соответственно
:set ts=4	задать размер табуляции в 4 пробела

9.4 Редактор VIM

9.4.1 Режимы работы

В ViM существуют 3 режима работы:

Основной – предназначен для просмотра файла, ввода команд и перехода из него в другие режимы. Из любого режима в командный можно попасть при нажатии «ESC». При нажатии клавиши «:» становится доступна командная строка ViM, в которой можно вводить команды. Основные команды – команда выхода «quit» (ViM понимает сокращения, поэтому можно давать команду одной

буквой «q»), команда сохранения «write» (или «w»), параметром которой может быть имя файла, и вызов справки по, очевидно, «help» (или «h»). На остальные клавиши (и их последовательности) можно присвоить любое действие, либо использовать значения по умолчанию.

Визуальный – предназначен, в первую очередь, для выделения блоков текста. Для запоминания предлагаются 3 варианта перехода в этот режим – клавишей «v» для посимвольного выбора, «Shift+V» для построчного и «Ctrl+V» для блочного. В нормальном режиме (при переходе по «v») можно оперировать следующими сущностями: слово («w»), предложение («s»), параграф («p») и блок («b»). Выделение при этом начинать с позиции курсора («a»), или же с начала блока («i»). Например, выделение текущего блока (участка, ограниченного парными элементами) можно произвести следующим образом «Esc+VIB». Копирование в буфер выделенного текста осуществляется по «y», вырезание по «d», а вставка соответственно «p».

Режим редактирования – переход в режим редактирования осуществляется нажатием клавиши «Ins».

9.4.2 Основные возможности

Все возможности и команды редактора ViM перечислить весьма затруднительно. Перечисленные ниже команды вводятся в основном режиме (если нет специального уточнения). Все они имеют командностроочные аналоги и могут быть легко переопределены пользователями.

Переходы

Для перехода на строку с номером n воспользуйтесь командой «G». Так, для перехода к началу текста наберите 0G, для сотой строки 100G, а для конца документа - $G. Для перехода на n символов в нужную вам сторону можете использовать клавиши со стрелками. То есть для перехода на 1000 символов вниз наберите «1000» и нажмите стрелку «вниз».

Для перемещения по тексту используйте следующие команды:

Команда	Описание команды
`(, )`	для перемещения по предложениям;
`{, }`	для параграфов;
`[[, ]]`	для функций;
`%`	переход к парной скобке;
`", "`	к предыдущему положению;
`CTRL+O, CTRL+I`	соответственно назад и вперёд по истории переходов.

Метки

Используются для отметки позиции ^<метка>, где <меткой> является любая буква, и быстрого к ней перехода (метка). Метки нижнего регистра действительны в пределах данного файла, метки же верхнего регистра действуют

во всех открытых файлах. Список всех меток можно получить командой «marks».

Регистры

В редакторе доступно множество именованных регистров (хранилищ данных, буферов). Регистр отмечается «"<буква>». К нему применимы все стандартные действия - копирование в него ("<метка> y), вырезание ("<метка>^) и вставка из него ("<метка> p, можете вместо p использовать [p,]p для вставки соответственно перед или после курсора). В режиме редактирования вставка из регистра осуществляется по ««Ctrl+R» <метка>». Для добавления данных в регистр используйте заглавную метку.

Также вы можете писать в регистр, воспользовавшись командой «^метка» и завершая запись по «q». Таким образом, вы сохраняете макрос, выполнить который можно по «@метка».

Регистры с метками «*» и «+» совпадают с X-Window clipboards, «%» - соответствует редактируемому файлу. Для просмотра содержимого всех регистров воспользуйтесь командой:

```
:registers
```

либо

```
:reg <метка1><метка2>...
```

для просмотра некоторых.

Фолды

Предназначены для сокрытия ненужных в данный момент данных, дабы те не отвлекали внимание. Например, кода подпрограммы, с которой вы в данный момент не работаете. По умолчанию фолды активированы в режиме их ручной расстановки. Если вы хотите их автоактивации по отношению к табуляции, то добавьте в конфиг строку:

```
set foldmethod=indent
```

Все команды для работы с фолдами начинаются с «z». Открытие фолда производится, например, по «zo» (или стрелке вправо) на нем, закрытие кода в фолд - по «zc».

Сессии

При ведении группы проектов нередко желательно сохранить текущее состояние и настройки редактора, чтобы в дальнейшем продолжить работу с того же места. Для этого и предназначены сессии, что создаются командой:

```
:mksession /path/to/Session.vim
```

а читаются простой командой:

```
:so /path/to/Session.vim
```

Гораздо чаще, впрочем, возникает нужда в сохранении не всей сессии, но только текущего контекста (во что входит, например, положение курсора в коде, текущая расстановка фолдов и много другое, о чем читайте в документации). Это действие выполняет команда:

```
:mkview
```

чтение:

```
:loadview
```

Очень удобно сделать сохранение и чтение контекста автоматическим при начале и окончании редактирования файла. Это может быть реализовано следующим кодом (применяется для всех файлов, имеющих точку в имени):

```
au BufWinLeave *.* mkview au BufWinEnter *.* silent loadview
```

Поиск и замена

Поиск осуществляется командами «/» для поиска (по регулярному выражению) вперёд, а «?» в обратном направлении. Для продолжения поиска используйте «n», а для прошлого варианта «N». Для поиска слова под курсором используются соответственно «#» и «*».

Для поиска с заменой используйте ^s^TO/m что/gic, где «%» означает работу со всем текстом (а не с текущей строкой), «g» – глобальная замена (а не первое совпадение), «i» – игнорирование регистра, а «c» – подтверждение каждого действия.

Автодополнение

Производится по содержимому данного файла, а также указанных в переменной <dictionary> по нажатию клавиш ''.

Отмена

«u» – для отмены и '' – смены регистра.

«~» – для выделенного участка (или буквы под курсором).

«U» – принудительно установить верхний регистр, а «u», соответственно, нижний.

Повторить

«.».

9.4.3 Конфигурация

Основным конфигурационным файлом является ~/.vimrc. Активация русского шрифта в GUI-режиме, плюс выбор темы для обоих режимов осуществляется, например, следующим кодом:

```
if has(<gui_running>)
colorscheme candy
set guifont=-cronyx-courier-medium-r-normal-*-*-120-*-*-m-*-koi8-
r endif
```

```
if !has(<gui_running>) colorscheme elflord endif
```

Перечень наиболее используемых «горячих» клавиш:

- Выход по F10:

```
nmap <F10> :q<CR>
imap <F10> <ESC>:q<CR>
```

- Сохранение по F2:

```
nmap <F2> :w>CR?
imap <F2> <ESC>:w<CR>i<Right>
```

- Компиляция по F9:

```
nmap <F9> :make<CR>
imap <F9> <ESC>:make<CR>
```

9.5 Назначение пароля на загрузчик GRUB2

Для защиты от несанкционированных изменений и прерывания нормальной загрузки ОС рекомендуется установить пароль на загрузчик GRUB.

Для этого перейдите в сеанс пользователя root:

```
su -
```

Затем выполните команду, в ответе которой назначьте новый пароль и, для исключения ошибок ввода, повторите его:

```
grub2-setpassword
Enter password: <новый_пароль>
Confirm password: <повторите_пароль>
```

Для применения настроек перезагрузите систему:

```
reboot
```

Далее при запуске системы нажмите клавишу «**e**».

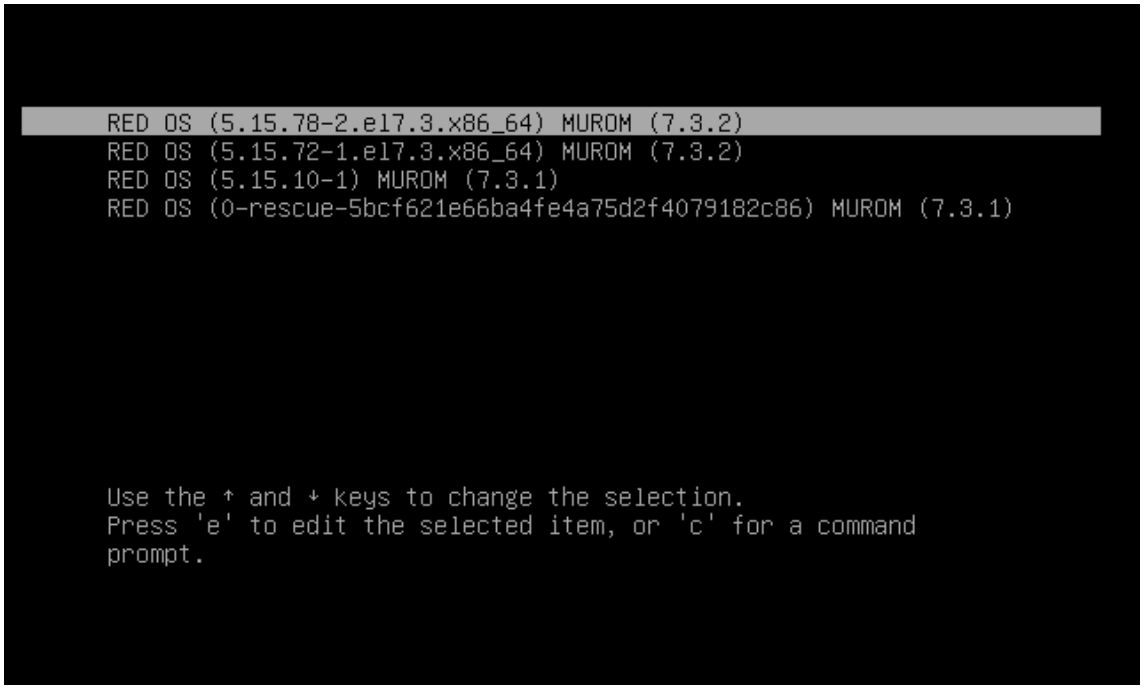

Рисунок 9.1 – Запуск системы

Будет открыта форма для авторизации:

```
Enter username:
root
Enter password:
<введите_пароль_для_grub2>
```

Если необходимо с определенного пункта меню GRUB убрать запрос пароля, в конфигурационном файле **/boot/grub2/grub.cfg** найдите загрузочную строку и добавьте параметр **--unrestricted**.

Рисунок 9.2 – Добавление дополнительных параметров

Для систем с UEFI файл конфигурации располагается в по адресу **/boot/efi/EFI/redos/grub.cfg**.

Средства виртуализации и контейнеризации

10. Средство виртуализации

10.1 Общие сведения

Виртуализация — предоставление набора вычислительных ресурсов или их логического объединения, независимое от аппаратной реализации и обеспечивающее при этом логическую изоляцию друг от друга вычислительных процессов, выполняемых на одном физическом ресурсе.

Поддержка виртуализации в РЕД ОС обеспечивается технологией KVM (Kernel-based Virtual Machine). Программное обеспечение KVM состоит из загружаемого модуля ядра kvm.ko, предоставляющего базовый сервис виртуализации, процессорно-специфического загружаемого модуля kvm-amd.ko или kvm-intel.ko и компонентов пользовательского режима QEMU.

QEMU — средство эмуляции аппаратного обеспечения различных платформ, поддерживающее также аппаратную виртуализацию, основанную на технологии Intel-VT или AMD-V, на процессорах Intel и AMD.

Управление средой виртуализации в РЕД ОС осуществляется посредством набора инструментов libvirt, предоставляющего единый API к различным технологиям виртуализации.

10.2 Средства управления средой виртуализации

Управление средой виртуализации осуществляется с помощью сервера виртуализации libvirt. Главным компонентом сервера виртуализации является служба libvirtd.

Для выполнения необходимых задач управления гостевыми системами служба libvirtd должна быть запущена на хостовых ОС (в данном случае РЕД ОС Сервер). Под задачами управления понимаются такие действия, как создание, запуск, остановка, мониторинг и контроль ВМ, а также настройка сети, управление

хранилищем и образами дисков.

Основным файлом конфигурации сервера виртуализации является файл /etc/libvirt/libvirtd.conf. В нем задаются настройки и параметры, необходимые для корректной работы службы libvirtd, такие как настройка безопасных сетевых соединений, политика разграничения доступа, способы аутентификации, уровень журналирования и пр.

Описание ключевых параметров файла конфигурации /etc/libvirt/libvirtd.conf:

- listen_tls = 0 — флаг прослушивания безопасных TLS-соединений с использованием сертификатов. По умолчанию параметр включен, чтобы отключить — раскоммертируйте строку;
- listen_tcp = 1 — флаг прослушивания незашифрованных TCP-соединений. По умолчанию параметр отключен, чтобы включить — раскомментируйте строку;
- tls_port = "16514", tcp_port = "16509" — номера портов для обеспечения сетевого соединения;
- listen_addr = "192.168.0.1" — IP-адрес или имя хоста сетевого интерфейса для приема соединений;
- auth_tcp = "sasl", auth_tls = "none" — способ аутентификации для сокетов TCP и TLS. Могут принимать значения:
  - none — не выполнять проверку;
  - sasl — использовать структуру sasl;
  - polkit — использовать политику polkit.
- log_level = 3 — уровень журналирования. Может принимать значения:
  - 4 — ошибка;
  - 3 — предупреждение;
  - 2 — информационное сообщение;
  - 1 — отладочное сообщение.

Помимо основного файла конфигурации сервер виртуализации взаимодействует со следующими каталогами файловой системы хостовой ОС:

- /var/lib/libvirt — рабочий каталог сервера виртуализации;
- /var/log/libvirt — каталог журналов сервера виртуализации.

Для управления средой виртуализации могут использоваться как консольные, так и графические утилиты.

Набор инструментов libvirt включает в себя следующие основные компоненты управления:

- virt-install — утилита командной строки, предназначенная для создания ВМ;
- virt-manager — графическая утилита, предназначенная для управления ВМ;
- virsh — утилита командной строки, предназначенная для управления ВМ, сетями и дисками.

Для управления образами ВМ используется утилита qemu-img, входящая в состав средства эмуляции аппаратного обеспечения QEMU.

10.2.1 Утилита qemu-img

Утилита qemu-img входит в состав пакета qemu-kvm и предназначена для манипуляций с образами дисков ВМ, таких как форматирование, создание и преобразование.

Qemu-img поддерживает работу со следующими форматами образов дисков:

- raw — используется по умолчанию, достоинствами являются простота и возможность экспортирования в другие эмуляторы;
- qcow2 — формат QEMU, наиболее гибкий формат, рекомендуется использовать для небольших образов;
- qcow — старый формат QEMU, используется только в целях обеспечения совместимости со старыми версиями;
- cow — формат COW (Copy On Write), используется только в целях обеспечения совместимости со старыми версиями;
- vmdk — формат образов, совместимый с VMware 3 и 4;
- cloop — формат CLOOP (Compressed Loop);
- vpc (vhd) — формат, поддерживающий полную структуру и содержание, сходные с жёстким диском.

Для манипуляций с образами дисков используются следующие команды:

- create — создание нового образа диска;
- check — проверка образа диска на ошибки;
- convert — конвертация существующего образа диска в другой формат;
- info — получение информации о существующем образе диска;
- snapshot — управление снимками состояний (snapshot) существующих образов дисков;
- commit — запись произведенных изменений на существующий образ диска;
- rebase — создание нового базового образа на основании существующего.

Синтаксис утилиты имеет следующий вид:

```
qemu-img <команда> [<опции>]
```

С подробной информацией об использовании утилиты можно ознакомиться, выполнив команду:

```
man qemu-img
```

Примеры использования утилиты:

1. Получение информации об образе:

```
qemu-img info /var/lib/libvirt/images/ro732.qcow2
```

```
image: /var/lib/libvirt/images/ro732.qcow2
file format: qcow2
virtual size: 5 GiB (5368709120 bytes)
disk size: 5 GiB
```

```
   cluster_size: 65536
   Format specific information:
       compat: 1.1
       compression type: zlib
       lazy refcounts: true
       refcount bits: 16
       corrupt: false
       extended l2: false
```

2. Создание нового образа:

```
   qemu-img create -f qcow2 /var/lib/libvirt/images/redos_vm.qcow2
20G
```

3. Преобразование формата существующего образа:

```
   qemu-img convert -f vpc vpc_image.vhd -O raw raw_image.dsk
```

10.2.2 Утилита virt-install

Утилита virt-install является инструментом командной строки и позволяет создавать ВМ с использованием библиотеки управления гипервизором libvirt. Конфигурационные файлы ВМ сохраняются в виде XML-файла. После создания ВМ сохраненный конфигурационный XML-файл можно отредактировать.

Утилита virt-install поддерживает графическую установку с использованием VNC или SPICE, а также установку в текстовом режиме из консоли. Гостевая ОС может быть настроена на использование одного или нескольких виртуальных дисков, сетевых интерфейсов, аудиоустройств, а также физических USB- или PCI-устройств.

Установочный носитель может храниться как локально (ISO-образ или CD-ROM), так и на удаленном сервере (HTTP, HTTPS или FTP). В случае расположения установочного носителя на FTP-сервере утилита получит минимальный набор файлов для запуска процесса установки, позволяя получать пакеты по мере необходимости. Кроме того поддерживается загрузка компонентов гостевых ОС по сети (PXE) и импорт образа диска с возможностью создания ВМ без запуска процесса установки.

С помощью соответствующих опций утилиты virt-install можно автоматизировать процесс создания виртуальных машин.

Большинство опций не являются обязательными для заполнения. Если при создании ВМ указано значение для параметра --os-variant, остальные параметры будут заполнены автоматически. Если параметр --os-variant не указан, минимально необходимыми опциями для создания ВМ являются:

- --name — имя новой ВМ;
- --ram — размер ОЗУ;
- --disk или --nodisks — настройка хранилища.

Установка утилиты осуществляется командой:

```
dnf install virt-install
```

Синтаксис утилиты имеет следующий вид:

```
virt-install [<опции>]
```

Далее будут рассмотрены ключевые опции утилиты virt-install.
Подключение к серверу виртуализации:

- --connect <URI> — подключение к гипервизору не по умолчанию. Допустимые варианты:
  - qemu:///system — для создания ВМ KVM и QEMU, которые будут запускаться системным экземпляром libvirtd. Данный режим является режимом по умолчанию;
  - qemu:///session — для создания ВМ KVM и QEMU для libvirtd, запущенной от имени обычного пользователя.

Общие параметры:

- -n <имя_ВМ>, --name=<имя_ВМ> — имя новой ВМ, должно быть уникальным в рамках одного гипервизора;
- --memory <опции> — память, выделяемая для ВМ (в МБ);
- --vcpus=<число> — количество виртуальных процессоров для ВМ. Могут быть указаны некоторые дополнительные параметры:
  - maxvcpus=MAX — максимальное кол-во виртуальных процессоров, выделяемых для ВМ;
  - sockets=# — количество сокетов;
  - cores=# — количество ядер;
  - threads=# — количество потоков выполнения.
- --cpu <параметры> — модель ЦП и его параметры, например: --cpu host или --cpu core2duo,+x2apic,disable=vmx.

Параметры установки:

- -c, --cdrom <путь> — путь к ISO-файлу или устройство CD-ROM для использования в качестве установочного носителя ВМ;
- -l, --location <путь> — источник установки дистрибутива, например, https://host/path;
- --pxe — установка из сети с использованием PXE;
- --import — пропустить процесс установки ОС и создать гостевую машину на основе существующего образа диска;
- --disk <параметры> — настройка параметров хранилища ВМ, таких как тип носителя, размер (в ГБ), формат диска и пр.;
- --filesystem <каталог> — экспорт каталога хостовой ОС в ВМ;
- -w <параметры>, --network <параметры> — настройка сети для ВМ;
- --vnc — позволяет открыть графическое окно установки виртуальной машины.

С подробной информацией об использовании утилиты можно ознакомиться, выполнив команду:

```
man virt-install
```

10.2.3 Утилита virsh

Утилита virsh — инструмент командной строки, предназначенный для управления ВМ и гипервизором KVM.

Утилита virsh использует API libvirt и является альтернативой графического менеджера ВМ virt-manager.

С помощью virsh можно управлять жизненным циклом ВМ, хранилищами данных, виртуальными сетями, сохранять состояние ВМ и переносить ВМ между гипервизорами.

Синтаксис утилиты имеет следующий вид:

```
virsh [<параметры>] ... [<команда_в_интерактивном_режиме>]
virsh [<параметры>] ... <команда> [<агрументы>]
```

Используемые параметры:

- -c, --connect=<URI> — URI подключения к гипервизору;
- -d, --debug=<ЧИСЛО> — уровень отладки [0-4];
- -e, --escape <символ> — управляющая последовательность для консольной команды;
- -h, --help — справка;
- -k, --keepalive-interval=<СЕК> — интервал отправки сообщений keepalive в секундах, 0 — отключает механизм keepalive;
- -K, --keepalive-count=<ЧИСЛО> — максимально допустимое число пропущенных сообщений keepalive;
- -l, --log=<ФАЙЛ> — вывод сообщений в файл;
- -q, --quiet — режим без уведомлений;
- -r, --readonly — подключиться в режиме чтения;
- -t, --timing — показывать время команд;
- -v — краткая версия;
- -V — подробная информация о версии;
- --version[=<ТИП>] — <ТИП> может принимать значения short (краткий номер версии) и long (подробная информация). По умолчанию используется значение short.

Команды управления ВМ:

- list — просмотр всех виртуальных машин;
- guestinfo — запросить информацию о гостевой системе (с помощью агента);
- create — создать виртуальную машину из файла конфигурации XML и запустить ее;
- start — запустить неактивную виртуальную машину;
- shutdown — корректно завершить работу виртуальной машины;
- destroy — принудительно остановить работу виртуальной машины;
- reboot — перезагрузить виртуальную машину;
- suspend — приостановить работу виртуальной машины;
- resume — возобновить работу приостановленной виртуальной машины;

- define — определить файл конфигурации XML для заданной виртуальной машины;
- dumpxml — вывести файл конфигурации XML для заданной виртуальной машины;
- domid — просмотр идентификатора виртуальной машины;
- domuuid — просмотр UUID виртуальной машины;
- dominfo — просмотр сведений о виртуальной машине;
- domname — просмотр имени виртуальной машины;
- domstate — просмотр состояния виртуальной машины;
- restore — восстановить сохраненную в файле виртуальную машину;
- save — сохранить состояние виртуальной машины в файл;
- undefine — удалить все файлы виртуальной машины.

Команды управления ресурсами ВМ:

- attach-device — подключить определенное в XML-файле устройство к ВМ;
- attach-disk — подключить новое дисковое устройство к ВМ;
- attach-interface — подключить новый сетевой интерфейс к ВМ;
- detach-device — отключить устройство от ВМ (принимает те же определения XML, что и attach-device);
- detach-disk — отключить дисковое устройство от ВМ;
- detach-interface — отключить сетевой интерфейс от ВМ;
- setmem — определить размер выделенной ВМ памяти;
- setmaxmem — ограничить максимально доступный гипервизору объем памяти;
- setvcpus — изменить число предоставленных ВМ виртуальных процессоров;
- vcpuinfo — просмотр информации о виртуальных процессорах;
- vcpupin — настройка соответствий виртуальных процессоров;
- domblkstat — получить статистику блочного устройства домена;
- domifstat — получить статистику сетевого интерфейса домена;
- domblklist — список блочных устройств домена;
- dominfo — информация о домене;
- snapshot-create — создать снимок на основе XML;
- snapshot-create-as — создать снимок на основе набора аргументов;
- snapshot-delete — удалить снимок домена;
- snapshot-list — показать снимки домена;
- snapshot-revert — восстановить домен из снимка;
- pool-create-as — создать пул на основе набора аргументов;
- pool-create — создать пул из файла XML;
- pool-list — список пулов;
- vol-create-as — создать том на основе набора аргументов;
- vol-create — создать том из файла XML;
- vol-delete — удалить том;
- vol-list — список томов;
- vol-resize — изменить размер тома;
- vol-download — загрузить содержимое тома в файл;
- vol-upload — отправить содержимое файла в том;

- vol-wipe — очистить том.

Ознакомиться со списком всех доступных команд или параметров можно, используя команду:

```
virsh --help
```

Примеры использования утилиты:
1. Подключение к локальному гипервизору:

```
virsh --connect qemu:///system
```

2. Сохранение XML-файла конфигурации виртуальной машины:

```
virsh dumpxml redos-7.3
```

Команда выводит содержимое XML-файла в стандартный вывод (stdout), для сохранения данных можно перенаправить вывод в файл, например:

```
virsh dumpxml redos-7.3 > ro73.xml
```

3. Создание виртуальной машины на основе файла конфигурации:

```
virsh create ro73.xml
```

4. Отправка сигнала на завершение работы виртуальной машины:

```
virsh shutdown redos-7.3
```

10.2.4 Менеджер виртуальных машин virt-manager

Менеджер виртуальных машин virt-manager предоставляет графический интерфейс для доступа к гипервизорам и виртуальным машинам как в локальной, так и в удаленных системах.

Графический менеджер virt-manager позволяет выполнять следующие задачи управления ВМ:

- создание, редактирование, запуск и остановка виртуальных машин;
- просмотр и управление каждой виртуальной машиной посредством консоли;
- просмотр производительности и статистики по каждой виртуальной машине;
- просмотр всех запущенных виртуальных машин на хосте и их производительность;
- использование виртуальных машин, запущенных локально или удаленно.

Установка менеджера ВМ производится командой:

```
dnf install virt-manager
```

После установки инструмент будет доступен из «Главного меню» — «Систем-

ные» — «Менеджер виртуальных машин».

Главное окно менеджера выглядит следующим образом:

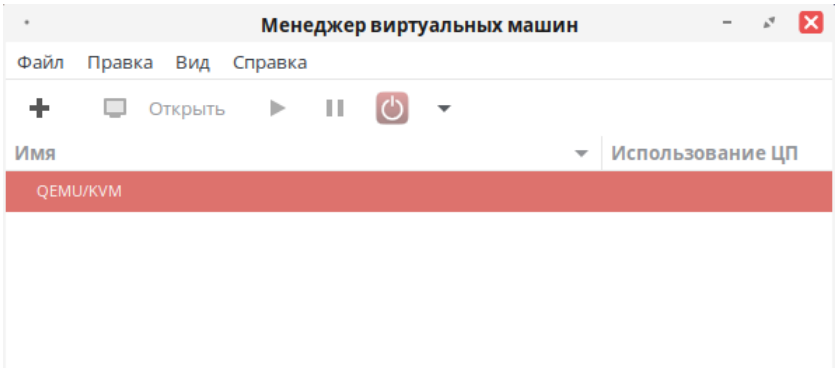

Рисунок 10.1 – Окно Менеджера виртуальных машин

Здесь отображаются доступные подключения и установленные ВМ, а также используемые ими ресурсы. Для открытия графической консоли и параметров ВМ необходимо дважды кликнуть по ее имени.

В основном меню менеджера можно настроить подключение к гипервизору, создать новые ВМ, просмотреть параметры существующих ВМ и ознакомиться с информацией о программе.

На панели инструментов доступно создание ВМ и управление их состоянием (запуск, пауза, выключение).

10.3 Установка среды виртуализации

Существует два варианта установки среды виртуализации:

- установка в процессе развертывания новой системы РЕД ОС;
- установка в существующей системе РЕД ОС.

10.3.1 Установка в процессе развертывания новой системы РЕД ОС

После запуска интерактивной установки РЕД ОС с установочного носителя (USB, CD-ROM, PXE) параметры будущей системы настраиваются в штатном режиме (выбор языка, часового пояса, раскладки клавиатуры, источника установки, настройка сети), кроме параметра «Выбор программ».

Для развертывания среды виртуализации необходимо выполнить установку РЕД ОС конфигурации Сервер (с графическим интерфейсом или минимальный).

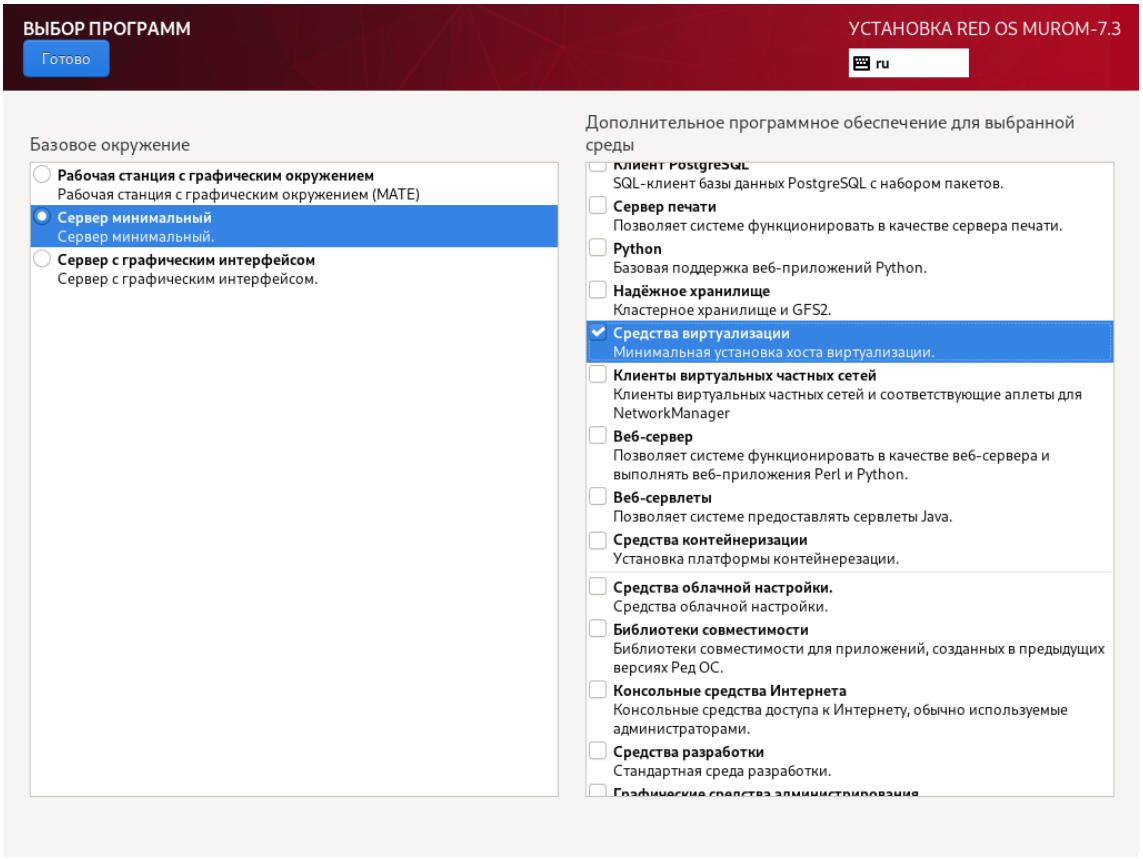

Рисунок 10.2 – Установка средства виртуализации

При выборе конфигурации Сервер (с графическим интерфейсом или минимальный) следует определить дополнительное программное обеспечение — *Средства виртуализации.*

Далее необходимо задать локального пользователя системы и его пароль, а также настроить пароль для суперпользователя root. Для возможности удаленного подключения к серверу виртуализации необходимо установить флаг в поле «Разрешить вход пользователем root с паролем через SSH».

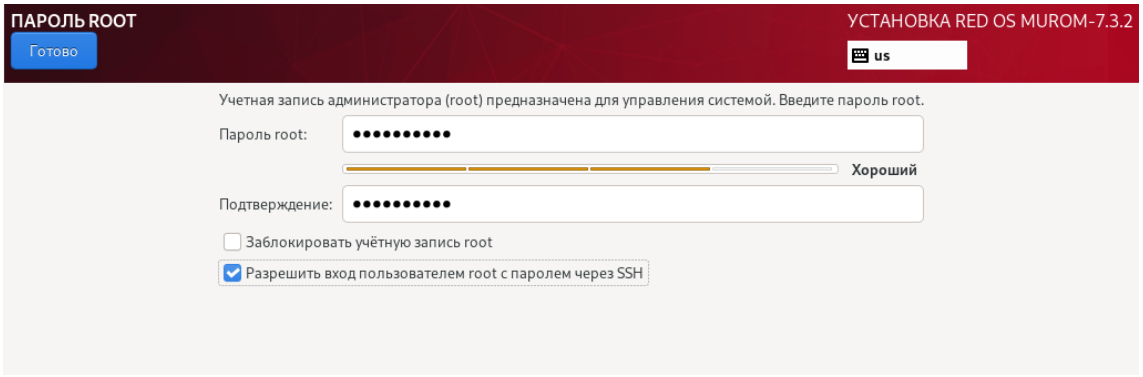

Рисунок 10.3 – Создание пароля суперпользователя root

После определения всех настроек нажмите кнопку «Начать установку».

Для завершения процесса установки потребуется перезагрузить систему и принять Лицензионное соглашение.

После авторизации в системе необходимо проверить статус сервиса libvirtd командой:

```
systemctl status libvirtd
```

В статусе должно быть указано **active (running)**. Если по каким-либо причинам статус не активен, выполните команду запуска сервиса с правами пользователя root:

```
systemctl start libvirtd
```

После этого ОС будет готова к использованию.

10.3.2 Установка в существующей системе РЕД ОС

Сначала необходимо проверить, поддерживает ли ваш процессор аппаратную виртуализацию. Проверка осуществляется командой:

```
egrep '(vmx|svm)'/proc/cpuinfo
```

Если вывод команды не пуст, значит процессор поддерживает аппаратную виртуализацию.

После проверки можно приступать к развертыванию среды виртуализации.

Для установки пакетов сервера виртуализации libvirt выполните команду:

```
dnf install libvirt qemu-kvm virt-install virt-manager libvirt-
cert-profile
```

Затем запустите и добавьте в автозагрузку службу сервера libvirtd:

```
systemctl enable libvirtd --now
```

Проверьте статус службы сервера libvirtd:

```
systemctl status libvirtd
```

В статусе должно быть указано active (running).

10.4 Ролевой доступ

В средстве виртуализации реализован ролевой метод управления доступом. Используется 4 основные роли пользователей:

- разработчик виртуальной машины — **vm-developer**;
- администратор безопасности средства виртуализации — **security-admin**;
- администратор средства виртуализации — **superuser**;
- администратор виртуальных машин — **vm-admin**.

Каждая роль подразумевает ряд предопределенных прав доступа и управления средой виртуализации.

Роль разработчика виртуальной машины (*vm-developer*) позволяет:

- создавать виртуальные машины;
- изменять конфигурации виртуальных машин.

Роль администратора безопасности средства виртуализации (*security-admin*) позволяет:

- иметь доступ на чтение к журналу событий безопасности средства виртуализации;
- формировать отчеты с учетом заданных критериев отбора, выгрузку (экспорт) данных из журнала событий безопасности средства виртуализации.

Роль администратора средства виртуализации (*superuser*) позволяет:

- создавать учетные записи пользователей средства виртуализации;
- управлять учетными записями пользователей средства виртуализации;
- назначать права доступа пользователям средства виртуализации к виртуальным машинам;
- создавать и удалять виртуальное оборудование средства виртуализации;
- изменять конфигурации виртуального оборудования средства виртуализации;
- управлять доступом виртуальных машин к физическому и виртуальному оборудованию;
- управлять квотами доступа виртуальных машин к физическому и виртуальному оборудованию;
- управлять перемещением виртуальных машин;
- удалять виртуальные машины;
- запускать и останавливать виртуальные машины;
- создавать снимки состояния виртуальных машин, включающих файл конфигурации виртуальной машины, образа виртуальной машины и образа памяти виртуальной машины.

Роль администратора виртуальной машины (*vm-admin*) позволяет:

- осуществлять доступ пользователя средства виртуализации к виртуальной машине посредством интерфейса средства виртуализации.

10.4.1 Создание новых ролей

Средство виртуализации также обеспечивает возможность определения полномочий для пользователей средства виртуализации в пределах назначенных им ролей.

Файл, определяющий полномочия пользователей, должен быть создан в каталоге /etc/libvirt/privileges.d/ и иметь название **roles.user**. Доступ к файлу может иметь только суперпользователь root.

Создание новых ролей производится вручную посредством внесения изменений в файл /etc/libvirt/privileges.d/roles.user. Для создания новой роли

необходимо добавить наименование роли и правила разрешений в формате **org.libvirt.api.<объект>.<разрешение>**.

Создание новой роли осуществляется по следующему шаблону:

```
nano /etc/libvirt/privileges.d/roles.user
```

```
"<имя_роли>": [
    "org.libvirt.api.<объект>.<разрешение>",
    "org.libvirt.api.<объект>.<разрешение>",
    "org.libvirt.api.<объект>.<разрешение>",
    ...    ...    ...
    "org.libvirt.api.<объект>.<разрешение>"
  ]
```

10.4.2 Объекты и разрешения

Средство виртуализации применяет контроль доступа ко всем основным типам объектов в своем API. Каждый тип объекта, в свою очередь, имеет определенный набор разрешений. Далее будут рассмотрены основные объекты и разрешения средства виртуализации.

Объект connect

Объект connect определяет настройки соединения. Для него существуют следующие разрешения:

- detect-storage-pools — определение пулов хранения;
- getattr — доступ к соединению;
- interface-transaction — транзакции интерфейса;
- pm-control — управление питанием хоста;
- read — чтение конфигурации соединения;
- search-domains — список доменов;
- search-interfaces — список интерфейсов;
- search-networks — список сетей;
- search-node-devices — список устройств хоста;
- search-nwfilter-bindings — список привязок сетевых фильтров;
- search-nwfilters — список сетевых фильтров;
- search-secrets — список секретов;
- search-storage-pools — список пулов хранения;
- write — запись конфигурации соединения на хост.

Объект domain

Объект domain определяет настройки доступа к домену (ВМ). Для него существуют следующие разрешения:

- block-read — чтение из блочных устройств домена;
- block-write — запись в блочные устройства домена;
- checkpoint — создание контрольной точки;
- core-dump — создание coredump-файла при аварийном завершении домена;
- delete — удаление домена;

- fs-freeze — заморозка файловой системы домена;
- fs-trim — очистка удаленных блоков на блочном устройстве домена
- getattr — доступ к домену;
- hibernate — ввод домена в спящий режим;
- init-control — разрешение, необходимое для управления выключением/перезагрузкой гостевой ОС;
- inject-nmi — отправка nonmasked-interrupt домену;
- mem-read — чтение памяти домена;
- migrate — миграция домена;
- open-device — разрешение на открытие последовательного канала/графической консоли домена;
- open-graphics — разрешение открытия графической консоли;
- pm-control — управление питанием;
- read — чтение конфигурации домена;
- read-secure — разрешение на чтение конфигурации домена, с включением секретной информации (например, пароли для доступа к блочным устройствам);
- reset — перезагрузка домена;
- save — сохранение состояния домена;
- screenshot — создание скриншота домена;
- send-input — разрешение на отправку кодов клавиш;
- send-signal — разрешение на отправку сигналов определённым процессам внутри домена;
- set-password — установка паролей пользователей внутри домена;
- set-time — установка времени в домене;
- snapshot — создание снапшотов домена;
- start — запуск домена;
- stop — остановка домена;
- suspend — приостановка домена;
- write — запись конфигурации домена.

Объект interface

Объект interface определяет настройки доступа к интерфейсам. Для него существуют следующие разрешения:

- delete — удаление интерфейса;
- getattr — доступ к интерфейсу;
- read — чтение конфигурации интерфейса;
- save — сохранение интерфейса;
- start — запуск интерфейса;
- stop — остановка интерфейса;
- write — запись конфигурации интерфейса.

Объект network

Объект network определяет настройки доступа к сети. Для него существуют следующие разрешения:

- delete — удаление настроек сети;
- getattr — доступ к настройкам сети;

- read — чтение настроек сети;
- save — сохранение настроек сети;
- search-ports — список доступных сетевых портов;
- start — запуск сети;
- stop — остановка сети;
- write — запись сетевых настроек.

Объект network-port

Объект network-port определяет настройки доступа к сетевым портам. Для него существуют следующие разрешения:

- create — назначение сетевых портов;
- delete — удаление сетевых портов;
- getattr — доступ к управлению сетевыми портами;
- read — чтение конфигурации сетевых портов;
- write — запись конфигурации сетевых портов.

Объект node-device

Объект node-device определяет настройки доступа к устройствам хоста. Для него существуют следующие разрешения:

- delete — удаление устройства хоста;
- detach — отсоединение устройства хоста;
- getattr — доступ к устройствам хоста;
- read — чтение конфигурации устройств хоста;
- start — запуск устройств хоста;
- stop — остановка устройства хоста;
- write — запись конфигурации устройства хоста.

Объект nwfilter

Объект nwfilter определяет настройки доступа к фильтрам сети. Для него существуют следующие разрешения:

- delete — удалить фильтр сети;
- getattr — доступ к настройке фильтра сети;
- read — чтение настроек фильтра сети;
- save — сохранение настроек фильтра сети;
- write — запись настроек фильтра сети.

Объект storage-pool

Объект storage-pool определяет настройки доступа к пулу хранения. Для него существуют следующие разрешения:

- delete — удаление пула хранения;
- format — формат пула хранения;
- getattr — доступ к пулу хранения;
- read — чтение конфигурации пула хранения;
- refresh — обновление пула хранения;
- save — сохранение пула хранения;
- search-storage-vols — список томов в пуле хранения;
- start — запуск пула хранения;

- stop — остановка пула хранения;
- write — запись конфигурации пула хранения.

Объект storage-vol

Объект storage-vol определяет настройки доступа к томам хранения. Для него существуют следующие разрешения:

- create — создание пула хранения;
- data-read — чтение данных из тома хранения;
- data-write — запись данных в тома хранения;
- delete — удаление тома хранения;
- format — формат тома хранения;
- getattr — доступ к томам хранения;
- read — чтение конфигурации тома хранения;
- write — запись конфигурации тома;
- resize — изменение размера тома хранения.

10.4.3 Создание пользователей

В среде виртуализации изначально должен быть создан хотя бы один пользователь — администратор средства виртуализации (superuser). Для его создания необходимо с правами пользователя root выполнить команду:

```
useradd <имя_пользователя>
```

и задать созданному пользователю пароль:

```
passwd <имя_пользователя>
```

Далее необходимо назначить данному пользователю роль администратора средства виртуализации (superuser) командой:

```
libvirt-rbac-tool assign-role <имя_пользователя> superuser
```

Все остальные пользователи создаются администратором средства виртуализации (superuser) с помощью утилиты libvirt-dac-tool.

Синтаксис утилиты:

```
libvirt-dac-tool <команда> <имя_пользователя>
```

Доступные команды утилиты:

- useradd — создание нового пользователя;
- passwd — создание пароля пользователю;
- usermod — добавление пользователя в группу;
- userdel — удаление пользователя.

Приведенные выше команды по структуре аналогичны одноименным системным командам. Для просмотра подробных сведений о командах и их доступных параметрах выполните:

```
man <команда>
или
<команда> --help
```

Например:

```
man useradd
```

10.4.4 Назначение и отзыв ролей

Для настройки прав доступа пользователей к средству виртуализации предназначена утилита **libvirt-rbac-tool**. Права доступа пользователям может назначать только администратор средства виртуализации (*superuser*).

Для назначения какой-либо роли пользователю средства виртуализации выполните команду:

```
libvirt-rbac-tool assign-role <имя_пользователя> <имя_роли>
```

Для отзыва какой-либо роли у пользователя выполните команду:

```
libvirt-rbac-tool revoke-role <имя_пользователя> <имя_роли>
```

Для просмотра назначенных пользователю ролей выполните команду:

```
libvirt-rbac-tool list-user-roles <имя_пользователя>
```

Для того чтобы назначить пользователю права доступа к какой-либо ВМ, выполните команду:

```
libvirt-rbac-tool assign-role-to-vm <имя_пользователя> <имя_роли>
<имя_ВМ>
```

Для отзыва у пользователя прав доступа к какой-либо ВМ выполните команду:

```
libvirt-rbac-tool revoke-role-from-vm <имя_пользователя> <имя_
роли> <имя_ВМ>
```

Для просмотра назначенных прав доступа к ВМ выполните команду:

```
libvirt-rbac-tool list-vm-roles <имя_ВМ>
```

10.5 Подключение к гипервизору

Подключение к гипервизору виртуализации может осуществляться тремя способами:

1. С использованием безопасного протокола удаленного доступа SSH.
2. С помощью утилиты командной строки virsh.
3. Через графический менеджер ВМ virt-manager.

10.5.1 Подключение к гипервизору через SSH

Для подключения к гипервизору виртуальных машин с использованием протокола безопасных соединений SSH на клиентском ПК, с которого планируется подключение, необходимо сгенерировать SSH-ключ. Публичную часть ключа следует скопировать на сервер виртуализации.

Для этого на клиентском ПК необходимо выполнить следующие команды с правами того пользователя, от имени которого будет осуществляться подключение:

```
ssh-keygen -t rsa
ssh-copy-id <имя_пользователя>@<IP-адрес_сервера_виртуализации>
```

Для доступа к среде виртуализации подключающемуся пользователю потребуется назначить соответствующие права. Подробную информацию о настройке прав доступа см. в разделе 10.4 «Ролевой доступ» настоящего руководства.

Также для доступа к среде виртуализации можно использовать учетную запись пользователя root. Для этого необходимо скопировать публичный ключ для root и выполнить подключение к серверу:

```
ssh-copy-id root@<IP-адрес_сервера_виртуализации>
ssh root@<IP-адрес_сервера_виртуализации>
```

10.5.2 Подключение к гипервизору с помощью virsh

Подключение к гипервизору можно осуществить с помощью утилиты командной строки virsh.

Команда подключения выглядит следующим образом:

```
virsh --connect <URI>
```

где параметр <**URI**> может принимать следующие значения:

- **qemu:///system** – для локального подключения к службе, управляющей доменами KVM/QEMU. Данный вариант используется по умолчанию;
- **qemu:///session** – для локального подключения к службе, управляющей доменами KVM/QEMU и запущенной от имени непривилегированного пользователя;
- **qemu+ssh://<имя_пользователя>@<IP-адрес_сервера>/system** — для подключения к удаленной службе, управляющей доменами KVM/QEMU.

После успешного подключения будет открыт интерактивный терминал гипервизора, в котором непосредственно выполняются команды управления ВМ и их ресурсами.

Для удобства обращения можно настроить переменные окружения, благодаря которым не будет необходимости каждый раз заново подключаться к гипервизору и открывать его интерактивный терминал, вся работа будет выполняться сразу из терминала пользователя средства виртуализации.

Для настройки переменных окружения с правами пользователя root создайте

файл /**etc**/**environment**:

```
nano /etc/environment
```

и добавьте в него следующую строку:

```
LIBVIRT_DEFAULT_URI=<URI>
```

где <**URI**> может принимать значения, описанные выше.

Настройку можно произвести для подключения как к локальному гипервизору, так и к удаленному.

Также есть возможность установить соединение только для чтения, для этого необходимо в команде указать параметр **--readonly**.

Примеры использования:

1. Создание локального подключения к среде виртуализации:

```
virsh --connect qemu:///system list --all
```

```
 ID   Имя             Состояние
--------------------------------
 2    live732         работает
 4    ro              работает
```

2. Подключение к локальному гипервизору с предварительной настройкой переменной окружения:

```
virsh list --all
```

```
 ID   Имя             Состояние
--------------------------------
 2    live732         работает
 4    ro              работает
```

3. Создание удаленного подключения к гипервизору:

```
virsh --connect qemu+ssh://virt-su@10.81.186.94/system
```

```
Добро пожаловать в virsh - интерактивный терминал виртуализации.
Введите «help» для получения справки по командам
«quit», чтобы завершить работу и выйти.
virsh # list --all
 ID   Имя              Состояние
--------------------------------
 -    redos-732        выключен
```

где:

- virt-su — имя пользователя, имеющего доступ к гипервизору;
- 10.81.186.94 — IP-адрес или имя сервера виртуализации.

10.5.3 Подключение к гипервизору с помощью менеджера ВМ

Для создания нового подключения в графическом менеджере ВМ необходимо в основном меню выбрать «Файл» - «Добавить соединение...».

Будет открыта форма добавления нового подключения, где необходимо из выпадающего списка выбрать гипервизор, к которому требуется осуществить подключение. Далее нужно установить флаг в поле «Connect to remote host over SSH» и указать в активировавшихся полях имя пользователя и IP-адрес сервера виртуализации.

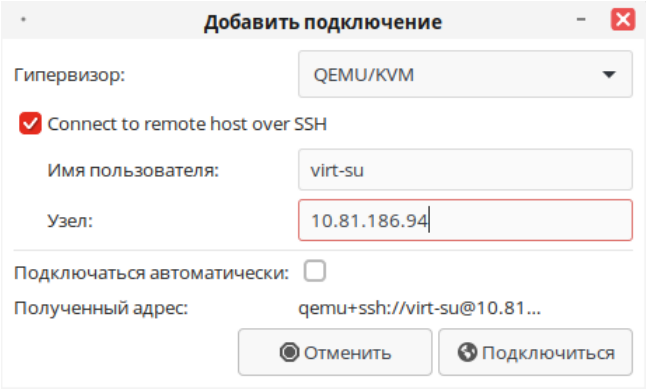

Рисунок 10.4 – Добавление нового подключения

При необходимости можно установить флаг для автоматического подключения к выбранному гипервизору.

Также открыть менеджер ВМ и подключиться к гипервизору можно из консоли, выполнив команду:

```
virt-manager -c qemu+ssh://virt-su@10.81.186.94/system
```

где:

- virt-su — имя пользователя, имеющего доступ к гипервизору;
- 10.81.186.94 — IP-адрес или имя сервера.

10.6 Создание виртуальных машин

Создание виртуальных машин является ключевым моментом при работе в среде виртуализации. Создание и установка ВМ может быть осуществлена как через терминал с помощью утилиты virt-install, так и через графический менеджер virt-manager.

10.6.1 Создание ВМ с помощью утилиты virt-install

Загрузите ISO-образ системы, которую необходимо установить. В примере будет использоваться образ РЕД ОС 7.3.2 с официального сайта.

После загрузки образа необходимо переместить его в доступный для libvirt каталог образов:

```
   cp /home/user/Загрузки/redos-MUROM-7.3.2-20221027.0-Everything-
x86_64-DVD1.iso /var/lib/libvirt/images/
```

После этого можно приступать к созданию ВМ. Для этого с правами администратора средства виртуализации (*superuser*) необходимо выполнить команду:

```
   virt-install --connect qemu:///system --name ro732 --ram=1024
--disk pool=default,size=20,format=qcow2 --cdrom /var/lib/libvirt/
images/redos-MUROM-7.3.2-20221027.0-Everything-x86_64-DVD1.iso --vnc
```

где:

- ro732 — имя ВМ;
- 1024 — объем ОЗУ;
- pool=default — настройка хранилища;
- size=20 — объем выделенного пространства хранения (в ГБ);
- format=qcow2 — формат хранилища;
- /var/lib/libvirt/images/redos-MUROM-7.3.2-20221027.0-Everything-x86_64-DVD1.iso — путь к ISO-образу ОС;
- --vnc — запуск графического менеджера (на хостовой ОС должен быть установлен пакет virt-viewer).

Если все параметры заданы верно, будет открыта консоль с предложением установки гостевой ОС.

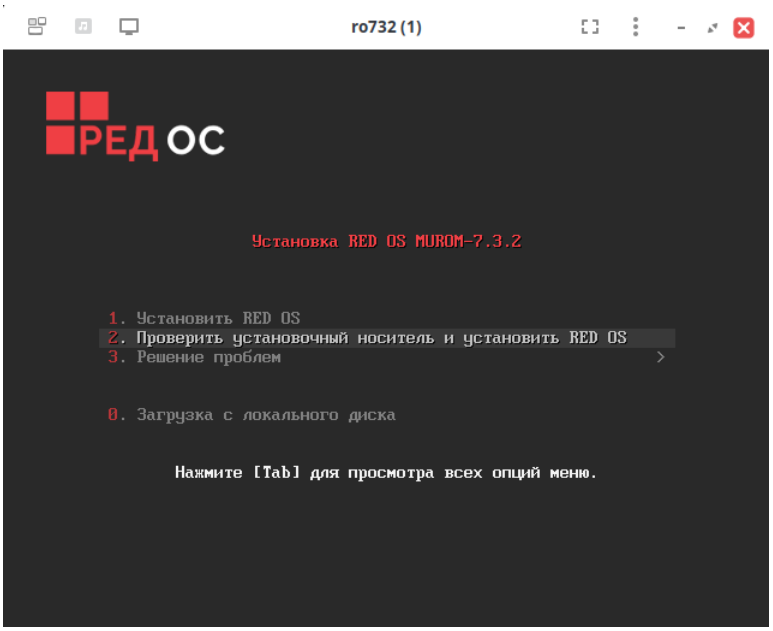

Рисунок 10.5 – Установка гостевой ОС

10.6.2 Создание ВМ с помощью менеджера virt-manager

Примечание. Необходимо обязательно подключиться к гипервизору, иначе в создании ВМ будет отказано и в окне «Новая виртуальная машина» будет отображено следующее сообщение об ошибке:

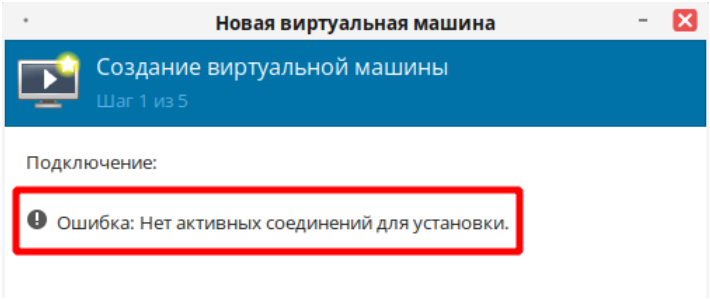

Рисунок 10.6 – Ошибка создания ВМ

Запустите менеджер ВМ через «Главное меню» — «Системные» — «Менеджер виртуальных машин» или через терминал командой:

```
virt-manager
```

Выполните подключение к гипервизору. Для этого в основном меню программы выберите «Файл» — «Добавить соединение. . . ».

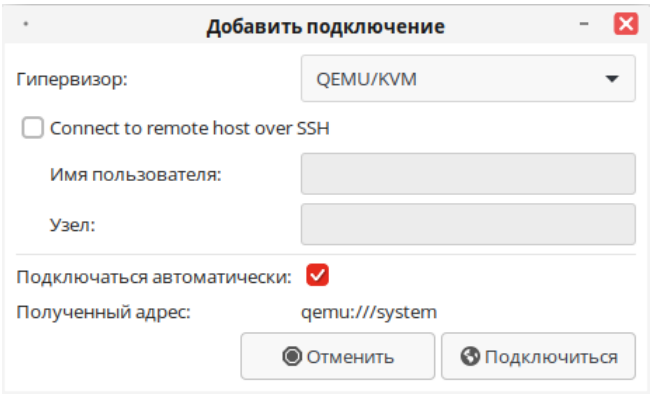

Рисунок 10.7 – Подключение к гипервизору

Выберите необходимый гипервизор и нажмите «Подключиться». После успешного подключения к гипервизору можно создавать ВМ. Для этого в панели инструментов нажмите на кнопку + или выберите «Файл» — «Создать виртуальную машину».

В открывшемся окне «Новая виртуальная машина» необходимо выбрать метод установки ОС — локальный или по сети.

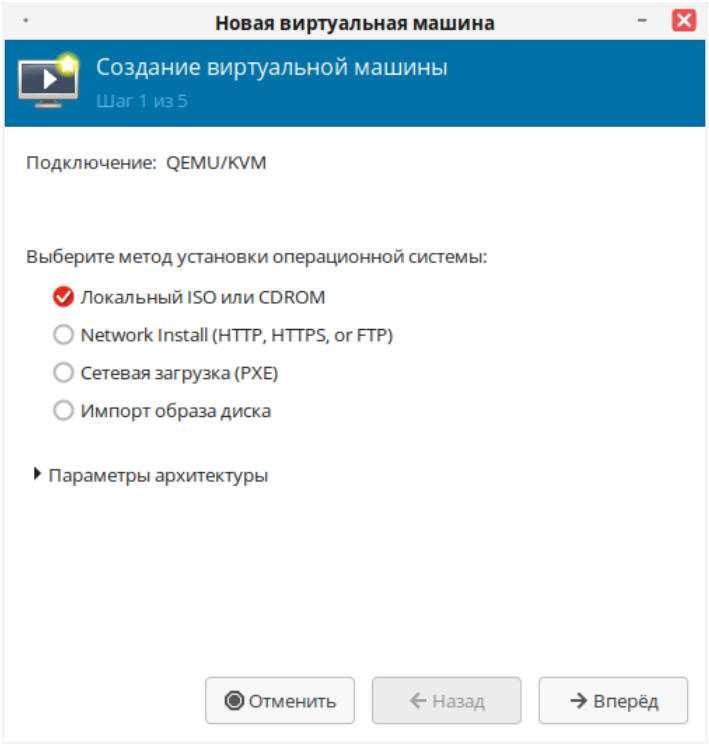

Рисунок 10.8 – Выбор метода установки новой ВМ

Далее необходимо указать путь к ISO-образу устанавливаемой системы, нажав кнопку «Обзор...».

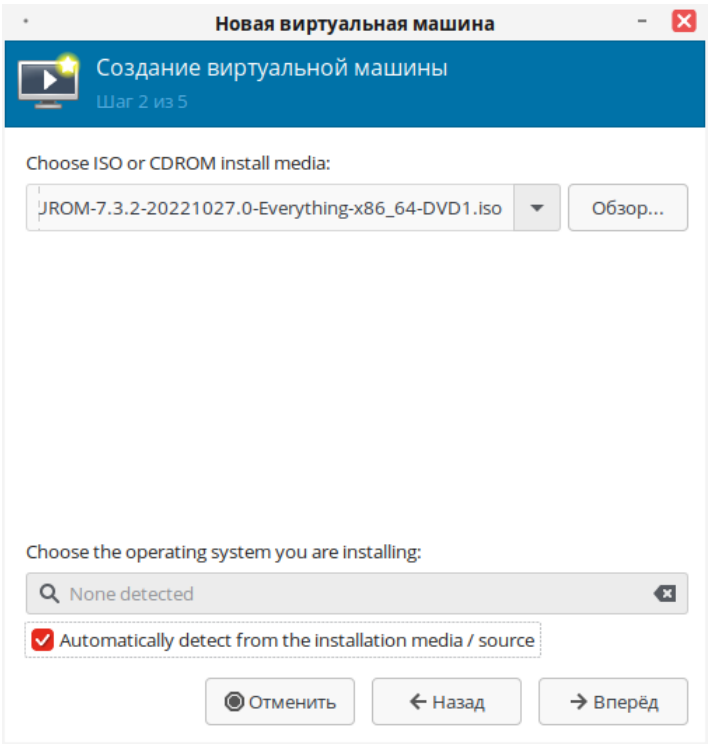

Рисунок 10.9 – Выбор типа ОС и ISO-образа

Снимите флаг в поле «Automatically detect from the installation madia/source» и в строке выбора типа ОС начните вводить ее название (в примере redos), будет выведен список доступных ОС, из которых потребуется выбрать нужную систему. Если нужной ОС нет в списке, выберите стандартную конфигурацию — Generic.

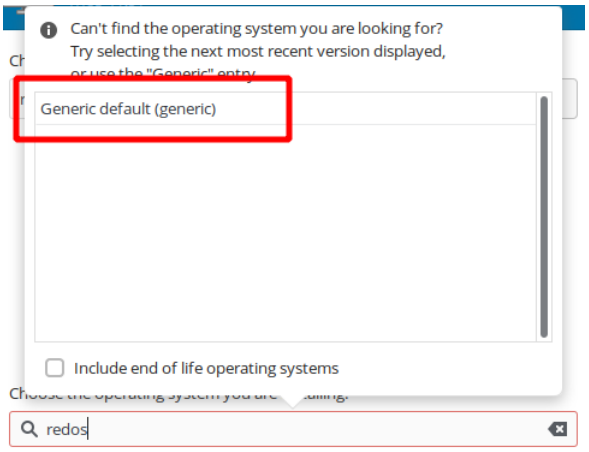

Рисунок 10.10 – Выбор стандартной конфигурации

Определите оптимальное количество ОЗУ и процессоров, исходя из доступных данных. Виртуальной машине понадобится достаточный для ее работы объем оперативной памяти (как минимум 512 Мбайт). Стоит помнить, что ВМ используют физическую память. Выполнение слишком большого числа гостевых ОС или предоставление ОС недостаточного объема памяти может привести к повышенному использованию виртуальной памяти и области подкачки.

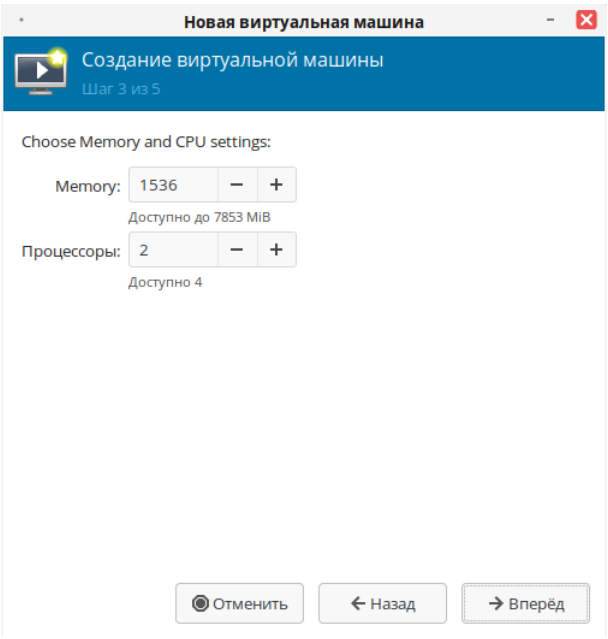

Рисунок 10.11 – Определение ОЗУ и процессоров

Далее необходимо настроить хранилище данных. Объем хранилища должен превышать минимально необходимый объем памяти, требуемый для установки системы.

Также, при желании, можно выделить дополнительное пространство для гостевой ОС.

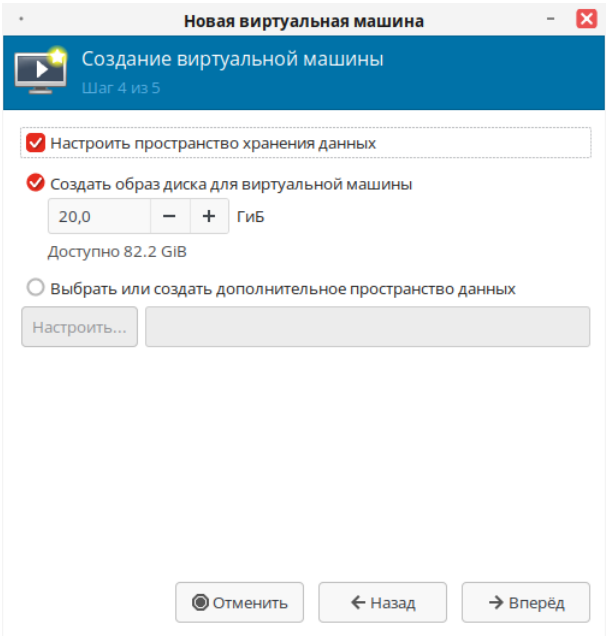

Рисунок 10.12 – Настройка пространства хранения данных

Проверьте правильность всех настроенных параметров. Если необходимо внести изменения, нажмите кнопку «Назад». Если все настроено верно, нажмите кнопку «Готово».

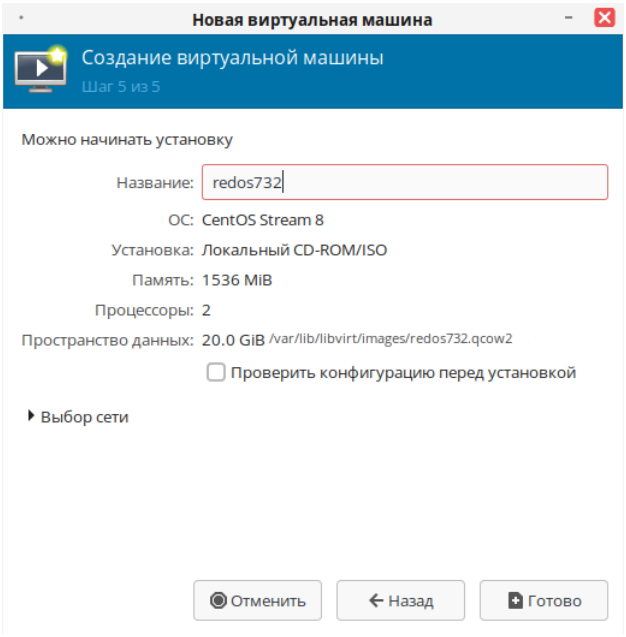

Рисунок 10.13 – Проверка параметров

Будет запущен процесс создания ВМ.

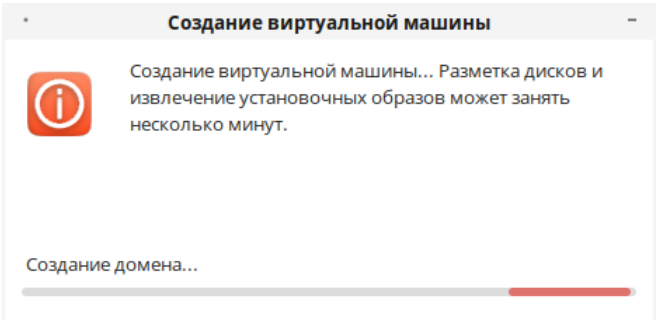

Рисунок 10.14 – Процесс создания ВМ

После успешного создания ВМ будет открыта консоль, в которой можно приступить к стандартной установке ОС.

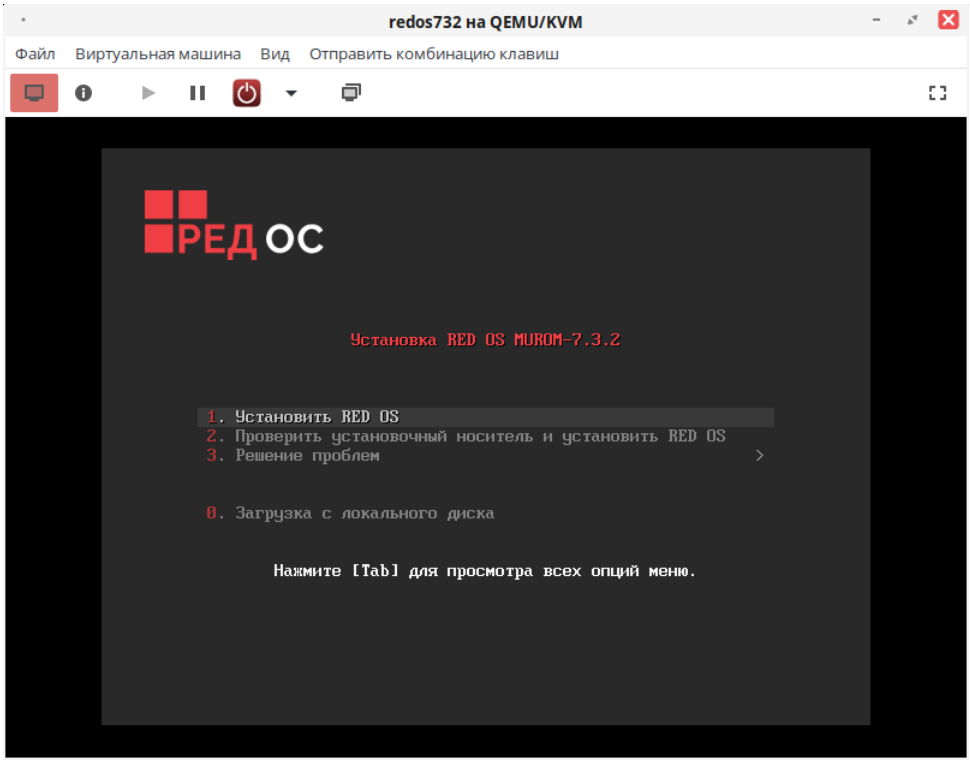

Рисунок 10.15 – Консоль установленной ВМ

10.6.3 Создание ВМ через файл конфигурации

Создание виртуальных машин возможно также с использованием файлов конфигурации формата XML.

За основу можно взять XML-файл уже существующей ВМ или создать собственный файл со своими параметрами.

Готовый XML-файл существующей ВМ можно просмотреть, выполнив команду вида:

```
virsh dumpxml <имя_ВМ>
```

Например:

```
virsh dumpxml ro732
```

```
<domain type='kvm'>
  <name>ro732</name>
  <uuid>2941e345-7cad-482b-9653-6849fd247638</uuid>
  <memory unit='KiB'>1048576</memory>
  <currentMemory unit='KiB'>1048576</currentMemory>
  <vcpu placement='static'>1</vcpu>
  <os>
    <type arch='x86_64'machine='pc-i440fx-6.1'>hvm</type>
    <boot dev='hd'/>
  </os>
  <features>
    <acpi/>
    <apic/>
  </features>
  <cpu mode='host-model'check='partial'/>
  <clock offset='utc'>
    <timer name='rtc'tickpolicy='catchup'/>
    <timer name='pit'tickpolicy='delay'/>
    <timer name='hpet'present='no'/>
  </clock>
  <on_poweroff>destroy</on_poweroff>
  <on_reboot>restart</on_reboot>
  <on_crash>destroy</on_crash>
  <pm>
    <suspend-to-mem enabled='no'/>
    <suspend-to-disk enabled='no'/>
  </pm>
  <devices>
    <emulator>/usr/bin/qemu-system-x86_64</emulator>
    <disk type='file'device='disk'>
      <driver name='qemu'type='qcow2'/>
      <source file='/var/lib/libvirt/images/ro732.qcow2'/>
      <target dev='hda'bus='ide'/>
      <address type='drive'controller='0'bus='0'target='0'
unit='0'/>
    </disk>
    <disk type='file'device='cdrom'>
      <driver name='qemu'type='raw'/>
      <target dev='hdb'bus='ide'/>
```

```
        <readonly/>
        <address type='drive'controller='0'bus='0'target='0'
unit='1'/>
      </disk>
      <controller type='usb'index='0'model='ich9-ehci1'>
        <address type='pci'domain='0x0000'bus='0x00'slot='0x04'
function='0x7'/>
      </controller>
      <controller type='usb'index='0'model='ich9-uhci1'>
        <master startport='0'/>
        <address type='pci'domain='0x0000'bus='0x00'slot='0x04'
function='0x0'multifunction='on'/>
      </controller>
      <controller type='usb'index='0'model='ich9-uhci2'>
        <master startport='2'/>
        <address type='pci'domain='0x0000'bus='0x00'slot='0x04'
function='0x1'/>
      </controller>
      <controller type='usb'index='0'model='ich9-uhci3'>
        <master startport='4'/>
        <address type='pci'domain='0x0000'bus='0x00'slot='0x04'
function='0x2'/>
      </controller>
      <controller type='pci'index='0'model='pci-root'/>
      <controller type='ide'index='0'>
        <address type='pci'domain='0x0000'bus='0x00'slot='0x01'
function='0x1'/>
      </controller>
      <interface type='network'>
        <mac address='52:54:00:23:ee:2b'/>
        <source network='default'/>
        <model type='e1000'/>
        <address type='pci'domain='0x0000'bus='0x00'slot='0x03'
function='0x0'/>
      </interface>
      <serial type='pty'>
        <target type='isa-serial'port='0'>
          <model name='isa-serial'/>
        </target>
      </serial>
      <console type='pty'>
        <target type='serial'port='0'/>
      </console>
      <input type='tablet'bus='usb'>
        <address type='usb'bus='0'port='1'/>
```

```
      </input>
      <input type='mouse'bus='ps2'/>
      <input type='keyboard'bus='ps2'/>
      <graphics type='vnc'port='-1'autoport='yes'>
        <listen type='address'/>
      </graphics>
      <audio id='1'type='none'/>
      <video>
        <model type='qxl'ram='65536'vram='65536'vgamem='16384'
heads='1'primary='yes'/>
        <address type='pci'domain='0x0000'bus='0x00'slot='0x02'
function='0x0'/>
      </video>
      <memballoon model='virtio'>
        <address type='pci'domain='0x0000'bus='0x00'slot='0x05'
function='0x0'/>
      </memballoon>
    </devices>
  </domain>
```

Как уже рассматривалось ранее (см. п. 10.2.3 «Утилита virsh»), данная команда выводит XML-файл на стандартный вывод (stdout). Для сохранения вывода в отдельный файл, выполните команду:

```
virsh dumpxml ro732 > ro732_config.xml
```

Для внесения каких-либо изменений в файл конфигурации выполните команду:

```
virsh edit ro732_config.xml
```

Для создания ВМ из XML-файла выполните команду:

```
virsh create ro732_config.xml
```

10.7 Управление ВМ

Управление ВМ подразумевает такие действия, как:

- мониторинг состояния и ресурсов ВМ;
- управление питанием ВМ;
- создание снимков гостевой системы.

10.7.1 Управление ВМ с помощью утилиты virsh

Запуск ВМ

Для запуска ВМ выполните команду вида:

```
virsh start <ID_домена, имя_домена, UUID_домена>
```

Приостановка ВМ

Для приостановки ВМ выполните команду вида:

```
virsh suspend <ID_домена, имя_домена, UUID_домена>
```

В этом состоянии виртуальная машина все еще продолжает использовать системную память, но освобождает ресурсы процессора. Операции ввода и вывода приостанавливаются. Работа виртуальной машины может быть возобновлена с помощью команды resume.

Возобновление работы ВМ

Для возобновления работы ВМ выполните команду вида:

```
virsh resume <ID_домена, имя_домена, UUID_домена>
```

Работа машины будет возобновлена немедленно.

Сохранение состояния ВМ

Для сохранения текущего состояния ВМ выполните команду:

```
virsh save <ID_домена, имя_домена, UUID_домена> <файл>
```

В результате выполнения данной команды виртуальная машина будет остановлена, а ее данные будут сохранены в заданный файл (это может занять некоторое время в зависимости от доступного гостевой ОС объема памяти). Состояние может быть позднее восстановлено с помощью команды *restore*.

Восстановление виртуальной машины

Чтобы восстановить виртуальную машину, ранее сохраненную с помощью команды *virsh save*, выполните команду:

```
virsh restore <файл>
```

Сохраненная машина будет восстановлена из файла и перезапущена, что может занять некоторое время. Имя и идентификатор UUID виртуальной машины останутся неизменными, но будет предоставлен новый идентификатор домена.

Создание снимка ВМ

Для создания снимка (снапшота) ВМ выполните команду:

```
virsh snapshot-create <ID_домена, имя_домена, UUID_домена>
```

Снимок диска или памяти будет успешно создан. Позднее состояние ВМ можно будет востановить из созданного снапшота командой *snapshot-revert*.

Восстановление ВМ из снимка (снапшота)

Для восстановления состояния ВМ из снапшота выполните команду:

```
virsh snapshot-revert <ID_домена, имя_домена, UUID_домена>
[--snapshotname <имя_снимка>]
```

Снимок будет восстановлен и состояние ВМ будет возвращено к указанному или текущему (последнему) снапшоту.

Завершение работы виртуальной машины

Для завершения работы ВМ выполните:

```
virsh shutdown <ID_домена, имя_домена, UUID_домена>
```

Поведение выключаемой гостевой ОС можно контролировать с помощью параметра *on_ shutdown* в ее файле конфигурации.

Перезагрузка виртуальной машины

Для перезагрузки ВМ выполните:

```
virsh reboot <ID_домена, имя_домена, UUID_домена>
```

Поведение перезагружаемой гостевой ОС можно контролировать с помощью параметра *on_ reboot* в ее файле конфигурации.

Принудительная остановка виртуальной машины

Для принудительной остановки ВМ выполните:

```
virsh destroy <ID_домена, имя_домена, UUID_домена>
```

Эта команда выполнит немедленное отключение и остановит все сессии гостевых доменов.

Важно! Такое внезапное завершение может привести к повреждению файловых систем виртуальной машины. Команду *destroy* рекомендуется использовать только в случае, если виртуальная машина не отвечает на запросы. ■

Удаление ВМ

Перед удалением ВМ должна быть выключена командой *shutdown* либо *destroy*.

Также не может быть удалена ВМ, которая имеет снапшоты. Проверить наличие снапшотов у удаляемой ВМ можно командой:

```
virsh snapshot-list --domain <имя_домена>
```

Если снапшоты присутствуют, сначала потребуется удалить их. Для этого выполните команду:

```
virsh snapshot-delete <имя_домена> --snapshotname <имя_снапшота>
--children
```

Снапшот будет удален вместе с его дочерними элементами (параметр --children).

После этого удалите ВМ (для удаления всех принадлежащих домену томов хранения данных используйте параметр --remove-all-storage):

```
virsh undefine <имя_домена> --remove-all-storage
```

ВМ и используемые тома будут удалены. Для безопасного удаления данных с томов хранения ВМ без возможности их дальнейшего восстановления можно использовать дополнительный параметр --wipe-storage. Команда очистки данных хранилища и полного удаления ВМ выглядит следующим образом:

```
undefine <имя_домена> --wipe-storage --remove-all-storage
```

ВМ и все данные будут полностью удалены.

Создание томов хранения для ВМ

Для создания нового тома хранения выполните команду:

```
virsh vol-create-as <имя_пула> <имя_тома> <размер_тома_в_байтах>
[--format <строка>]
```

где:

- <имя_пула> — пул, в котором необходимо создать новый том;
- <имя_тома> — имя нового тома хранения;
- <размер_тома_в_байтах> — размер нового тома в байтах;
- [--format <строка>] — формат нового тома хранения: raw, bochs, qcow, qcow2, qed, vmdk.

Новый том хранения будет успешно создан. Размер нового тома хранения не должен превышать объем доступного свободного пространства на хостовой машине.

Список доступных томов хранения

Для вывода списка доступных томов хранения используется команда:

```
virsh vol-list <имя_пула>
```

Например:

```
virsh vol-list default
```

```
 Имя                  Путь
-------------------------------------------------------------------
 redos-...-7.3.2.iso  /var/lib/libvirt/images/redos-...-7.3.2.iso
 ro-test.qcow2        /var/lib/libvirt/images/ro-test.qcow2
```

Изменение размера тома хранения

Для изменения размера тома хранения выполните команду:

```
virsh vol-resize <имя_тома> <размер_тома>
```

где

- <имя_тома> — том, для которого необходимо изменить размер;
- <размер_тома> — новый размер тома, по умолчанию в байтах.

Безопасно использовать только для томов, которые не используются активной гостевой системой. При добавлении параметра **--delta** установленный размер тома будет увеличен (или уменьшен) на указанный размер. Например:

```
virsh vol-resize /var/lib/libvirt/images/ro.qcow2 1G --delta
```

```
Размер тома «ro.qcow2» успешно изменён на 1G
```

В примере размер указанного тома будет увеличен на 1ГБ.

Сохранение состояния жесткого диска ВМ

Для сохранения содержимого тома хранения ВМ в какой-либо локальный файл выполните команду:

```
virsh vol-download <имя_тома> <имя_файла>
```

Данные тома хранения будут записаны в указанный файл, который впоследствии можно скопировать и восстановить на другой ВМ или на этой же ВМ.

Восстановление состояния жесткого диска ВМ

Для восстановления содержимого тома хранения ВМ из файла выполните команду:

```
virsh vol-upload <имя_тома> <имя_файла>
```

Данные тома хранения будут восстановлены.

Определение идентификатора домена

Для определения идентификатора домена виртуальной машины выполните:

```
virsh domid <имя_домена, UUID_домена>
```

Определение имени домена

Для определения имени домена виртуальной машины выполните:

```
virsh domname <ID_домена, UUID_домена>
```

Определение UUID

Для определения универсального идентификатора UUID виртуальной машины выполните:

```
virsh domuuid <ID_домена, имя_домена>
```

Пример вывода команды:

```
virsh domuuid ro732
```

```
2941e345-7cad-482b-9653-6849fd247638
```

Получение информации о виртуальной машине

Для получения информации о ВМ выполните:

```
virsh dominfo <ID_домена, имя_домена, UUID_домена>
```

Например:

```
virsh dominfo ro732
```

```
ID:          1
Имя:         ro732
UUID:        2941e345-7cad-482b-9653-6849fd247638
Тип ОС:      hvm
Состояние:   работает
CPU:         1
Время CPU:   8,7s
Макс.память: 1048576 KiB
Занято памяти: 1048576 KiB
Постоянство: yes
Автозапуск: выкл.
Управляемое сохранение: no
Модель безопасности: selinux
DOI безопасности: 0
Метка безопасности: system_u:system_r:svirt_t:s0:c476,c967
(permissive)
```

Получение информации об узле

Для получения информации об узле выполните:

```
virsh nodeinfo
```

Например:

```
virsh nodeinfo
```

```
Модель процессора:  x86_64
CPU:                4
Частота процессора: 1600 MHz
Сокеты:             1
Ядер на сокет:      4
Потоков на ядро:    1
Ячейки NUMA:        1
Объём памяти:       8041816 KiB
```

Просмотр списка виртуальных машин

Для просмотра списка виртуальных машин и их состояния выполните:

```
virsh list
```

Можно добавить аргументы:

- --inactive — покажет список неактивных доменов (неактивным считается тот домен, который был определен, но в настоящий момент не является активным).
- --all — покажет все виртуальные машины независимо от их состояния.

Например:

```
virsh list --all
```

```
 ID    Имя        Состояние
----------------------------
 1     ro732      работает
 -     redos732   выключен
```

Столбец «Состояние» может содержать следующие значения:

- работает — работающие виртуальные машины, т. е. те машины, которые используют ресурсы процессора в момент выполнения команды;
- заблокирован — заблокированные, неработающие машины. Такой статус может быть вызван ожиданием ввода/вывода или пребыванием машины в спящем режиме;
- приостановлен — приостановленные домены. Данное состояние наступает, когда администратор нажал кнопку паузы в окне менеджера виртуальных машин или выполнил команду virsh suspend. В приостановленном состоянии гостевая ОС продолжает потреблять ресурсы, но не может занимать больше процессорных ресурсов;
- выключен — виртуальные машины, завершающие свою работу. При получении виртуальной машиной сигнала завершения работы, она начнет завершать все процессы. Стоит отметить, что некоторые операционные системы не отвечают на такие сигналы;
- dying — сбойные домены и домены, которые не смогли корректно завершить свою работу;
- crashed — сбойные домены, работа которых была прервана. В этом состоянии домены находятся, если не была настроена их перезагрузка в случае сбоя.

Получение информации о виртуальных процессорах

Для получения информации о виртуальных процессорах выполните:

```
virsh vcpuinfo <ID_домена, имя_домена, UUID_домена>
```

Пример вывода:

```
virsh vcpuinfo ro732
```

```
Виртуальный процессор:: 0
CPU:        3
Состояние:  работает
Время CPU:  39,1s
Соответствие ЦП: yyyy
```

Изменение числа виртуальных процессоров

Для изменения числа процессоров для домена выполните:

```
virsh setvcpus <ID_домена, имя_домена, UUID_домена> <число>
```

Обратите внимание, что заданное число не может превышать значение, определенное при создании гостевой ОС.

Изменение выделенного объема памяти

Для изменения выделенного виртуальной машине объема памяти выполните:

```
virsh setmem <ID_домена, имя_домена> <число>
```

Объем памяти, определяемый заданным числом, должен быть указан в килобайтах. Обратите внимание, что объем не может превышать значение, определенное при создании виртуальной машины, но в то же время не должен быть меньше 64 МБ. Изменение максимального объема памяти может оказать влияние на функциональность ВМ только в том случае, если указанный размер меньше исходного. В таком случае использование памяти будет ограничено.

Получение информации о блочных устройствах

Для получения информации о блочных устройствах работающей виртуальной машины выполните:

```
virsh domblkstat <виртуальная_машина> <блочное_устройство>
```

Получение информации о сетевых устройствах

Для получения информации о сетевых интерфейсах работающей виртуальной машины выполните:

```
virsh domifstat <виртуальная_машина> <интерфейс>
```

Управление виртуальными сетями

Для просмотра списка виртуальных сетей выполните:

```
virsh net-list
```

Пример вывода этой команды:

```
virsh net-list
```

```
 Имя       Состояние    Автозапуск    Постоянный
-------------------------------------------------
 default   активен      yes           yes
```

Просмотреть информацию для заданной виртуальной сети можно командой:

```
virsh net-dumpxml <имя_сети>
```

Пример вывода этой команды (в формате XML):

```
virsh net-dumpxml default
```

```
<network connections='9'>
  <name>default</name>
  <uuid>0f99e8c3-cfb0-4970-abcc-4bfbd9823c67</uuid>
  <forward mode='nat'>
    <nat>
      <port start='1024' end='65535'/>
    </nat>
  </forward>
  <bridge name='virbr0' stp='on'delay='0'/>
  <mac address='52:54:00:d8:21:b1'/>
  <ip address='192.168.122.1' netmask='255.255.255.0'>
    <dhcp>
      <range start='192.168.122.2' end='192.168.122.254'/>
    </dhcp>
  </ip>
</network>
```

Другие команды управления виртуальными сетями:

- virsh net-autostart <имя_сети> — автоматический запуск заданной сети;
- virsh net-create <файл_XML> — создание и запуск новой сети на основании существующего XML-файла;
- virsh net-define <файл_XML> — создание нового сетевого устройства на основании существующего XML-файла. Устройство не будет запущено;
- virsh net-destroy <имя_сети> — удаление заданной сети;
- virsh net-name <UUID_сети> — преобразование заданного идентификатора в имя сети;
- virsh net-uuid <имя_сети> — преобразование заданного имени в идентификатор UUID;
- virsh net-start <имя_неактивной_сети> — запуск неактивной сети;
- virsh net-undefine <имя_неактивной_сети> — удаление определения неактивной сети.

10.7.2 Управление ВМ с помощью менеджера virt-manager

Менеджер virt-manager предоставляет графический интерфейс для доступа к гипервизорам и виртуальным машинам в локальной и удаленных системах.

Кроме того, virt-manager выполняет управляющие функции:

- выделение памяти;
- выделение виртуальных процессоров;
- мониторинг производительности;
- сохранение и восстановление, приостановка и возобновление работы, запуск и завершение работы виртуальных машин;
- доступ к текстовой и графической консоли.

Производительность ВМ

Для просмотра графиков производительности какой-либо ВМ дважды нажмите ЛКМ по имени нужной ВМ, будут открыты подробные сведения.

На вкладке «Производительность» отображаются диаграммы и статистика использования ресурсов в реальном времени.

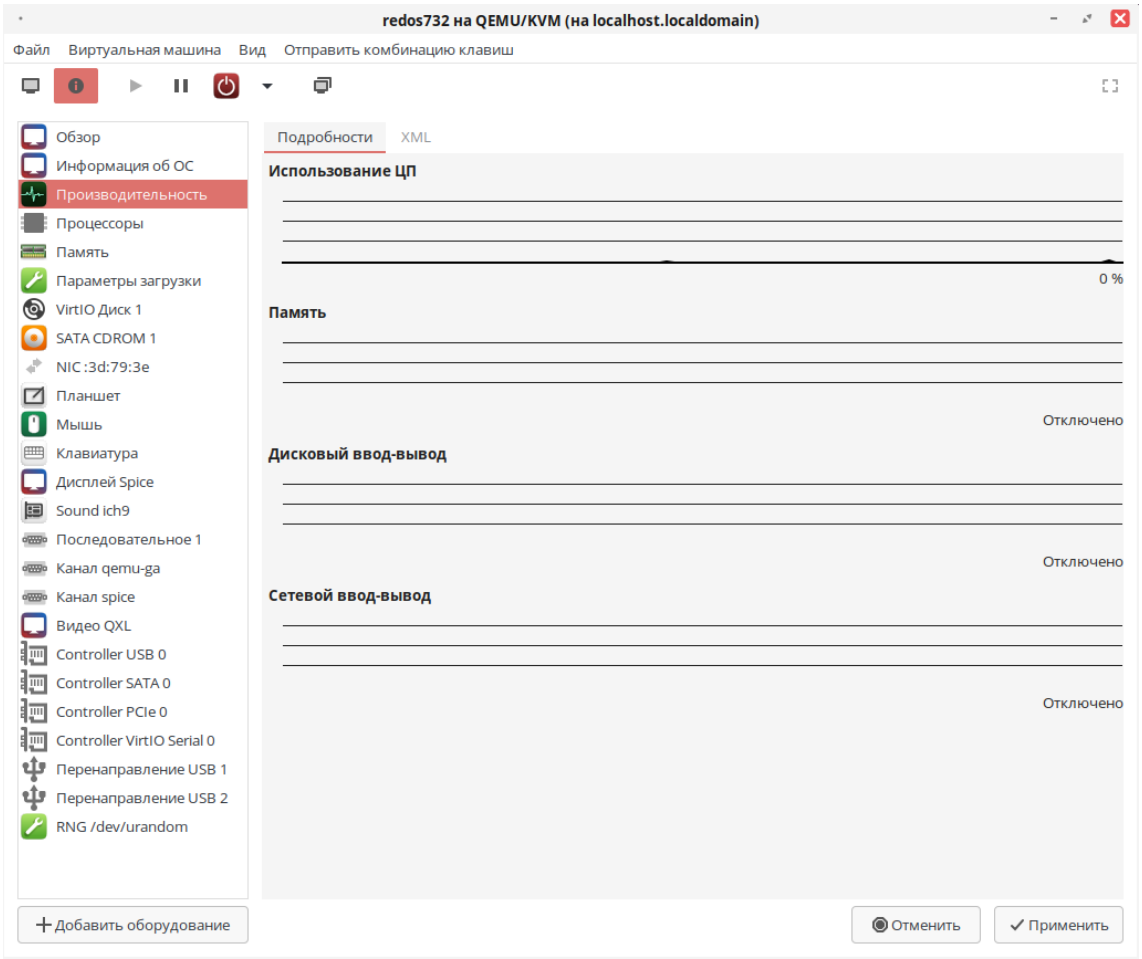

Рисунок 10.16 – Просмотр графиков производительности ВМ

Графическая консоль виртуальной машины

Окно содержит графическую консоль виртуальной машины. По умолчанию используется протокол SPICE для обеспечения доступа к консоли. Доступ к графической консоли ВМ имеет только пользователь с правами администратора ВМ.

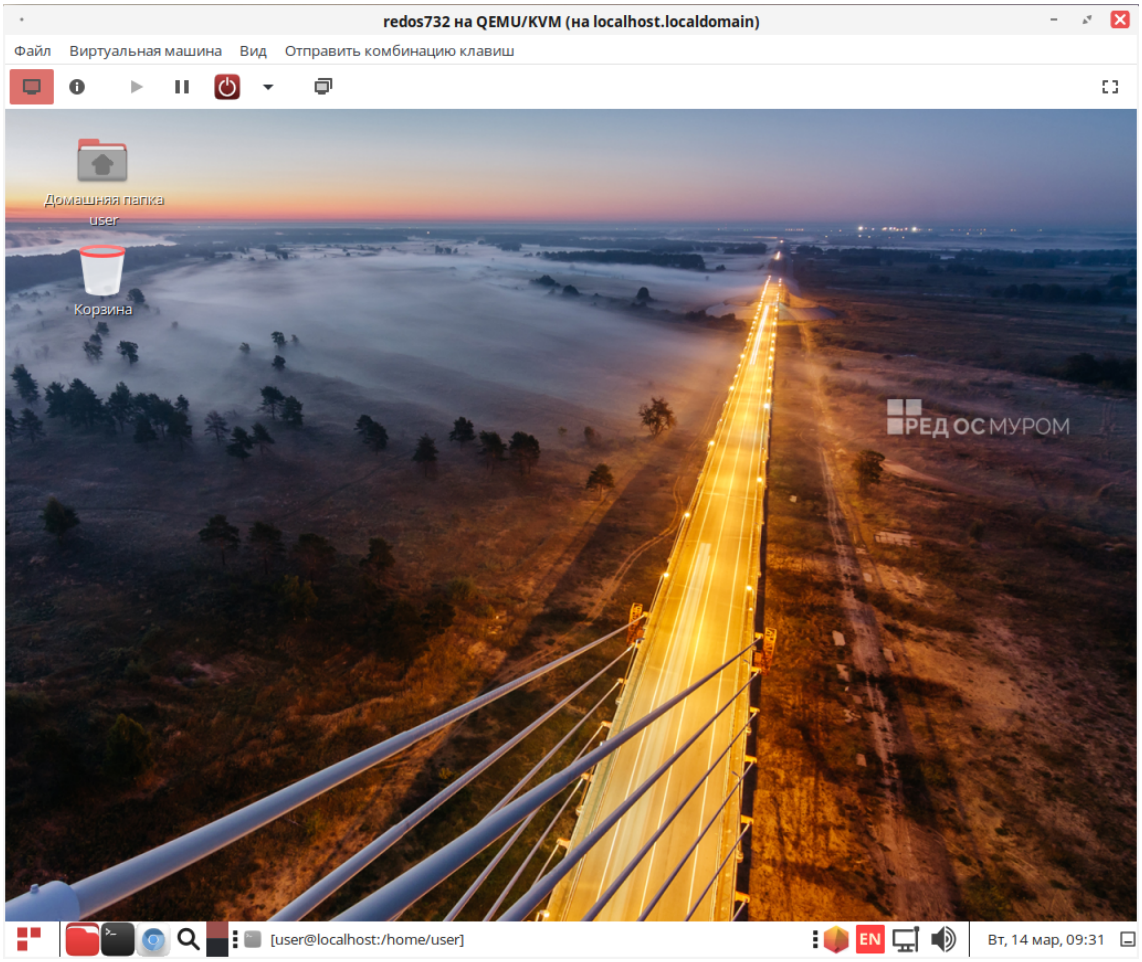

Рисунок 10.17 – Графическая консоль ВМ

Окружение локального рабочего стола способно перехватывать комбинации клавиш (например, Ctrl+Alt+F11) для предотвращения их отправки гостевой машине. Чтобы отправлять такие последовательности, используйте свойство "западания" клавиш virt-manager. Нажмите функциональную клавишу (Ctrl или Alt) 3 раза для ее перехода в нажатое состояние. Клавиша будет считаться нажатой до тех пор, пока не будет нажата любая другая клавиша, отличная от модификатора.

Таким образом, чтобы передать гостевой системе комбинацию Ctrl+Alt+F11, необходимо последовательно нажать Ctrl-Ctrl-Ctrl+Alt+F11.

Просмотр информации о гостевой системе

С помощью менеджера виртуальных машин можно получить доступ к подробной информации обо всех виртуальных машинах. Для этого выполните следующий алгоритм действий:

1. В главном окне менеджера выберите виртуальную машину.

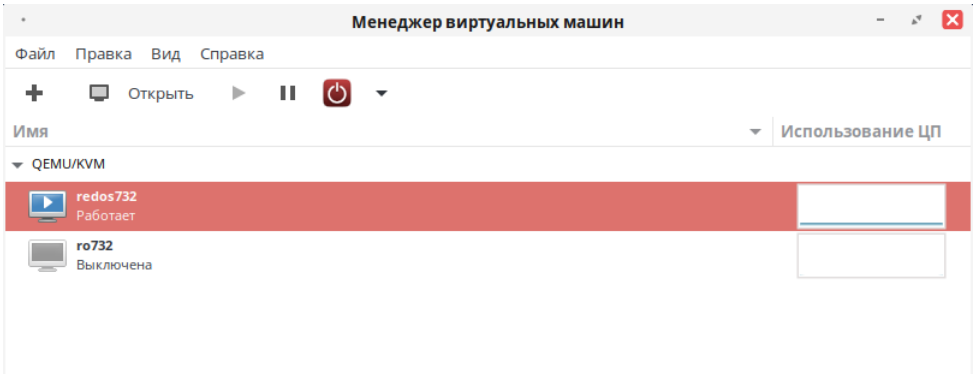

Рисунок 10.18 – Выбор ВМ

2. В пункте меню «Правка» выберите «Свойства виртуальной машины» или дважды нажмите ЛКМ по имени ВМ.

Будет выведено окно с подробными сведениями о виртуальной машине, на вкладке «Производительность» будет доступна информация об использовании ресурсов процессора и памяти.

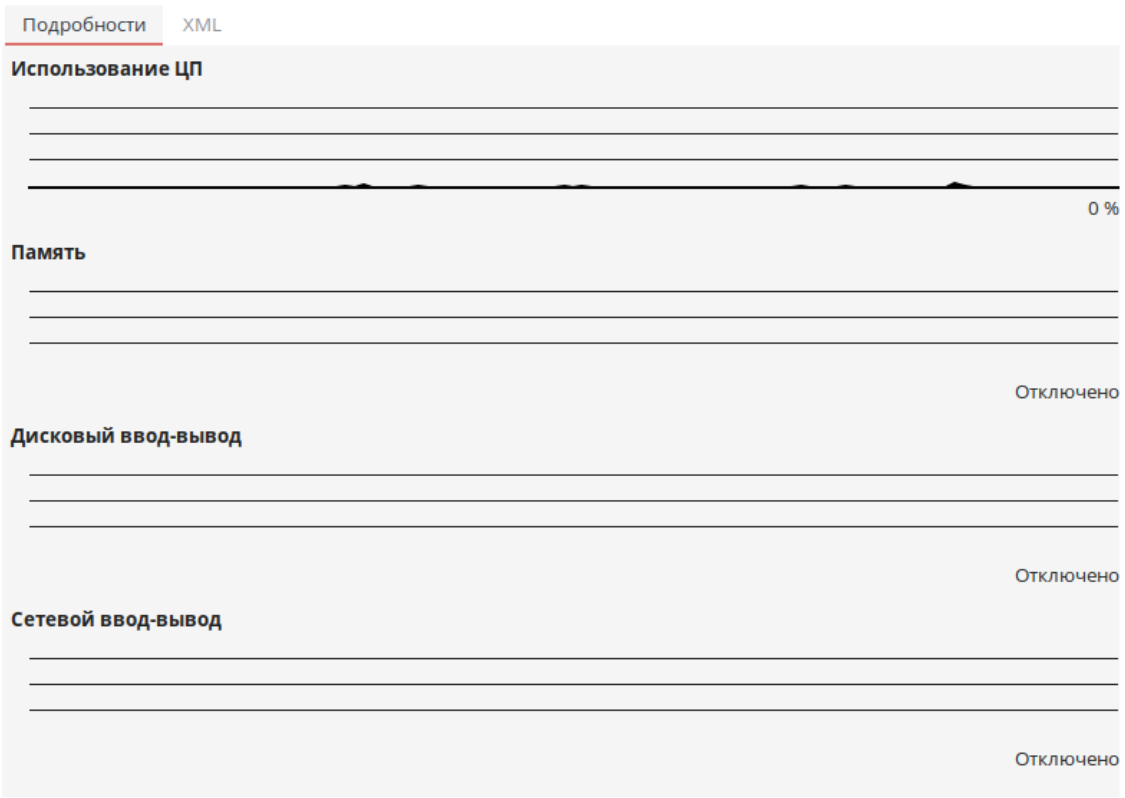

Рисунок 10.19 – Графики использования ресурсов процессора и памяти

3. Для просмотра или изменения числа виртуальных процессоров выберите «Процессоры».

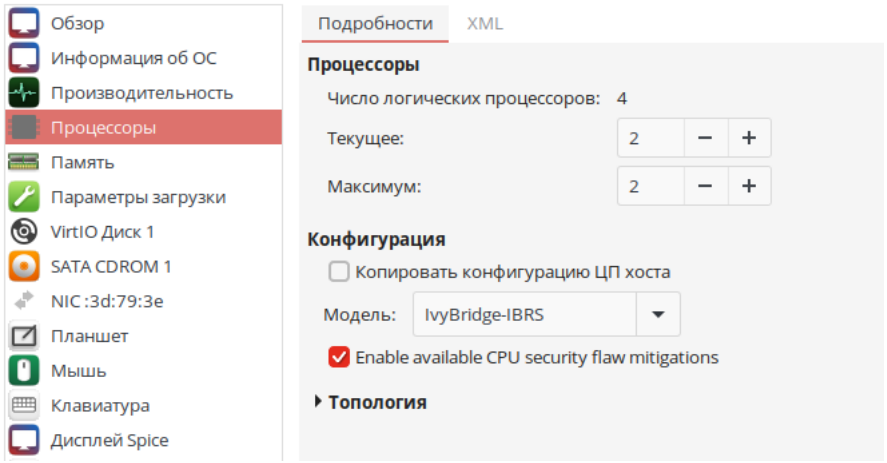

Рисунок 10.20 – Просмотр числа процессоров ВМ

4. Для просмотра или изменения распределения ресурсов памяти выберите «Память».

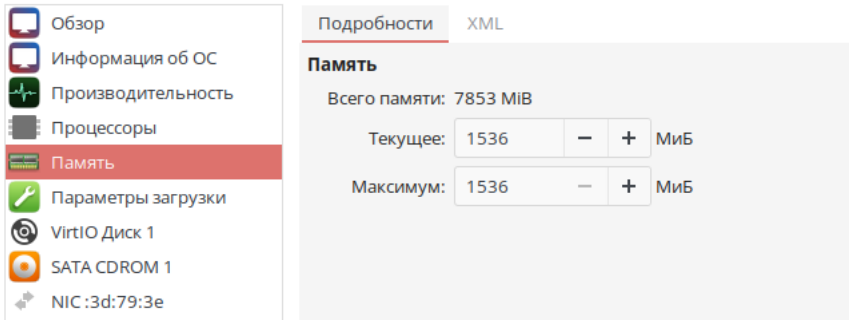

Рисунок 10.21 – Просмотр ресурсов ОЗУ

5. Для просмотра или изменения дисковой конфигурации выберите «Диск».

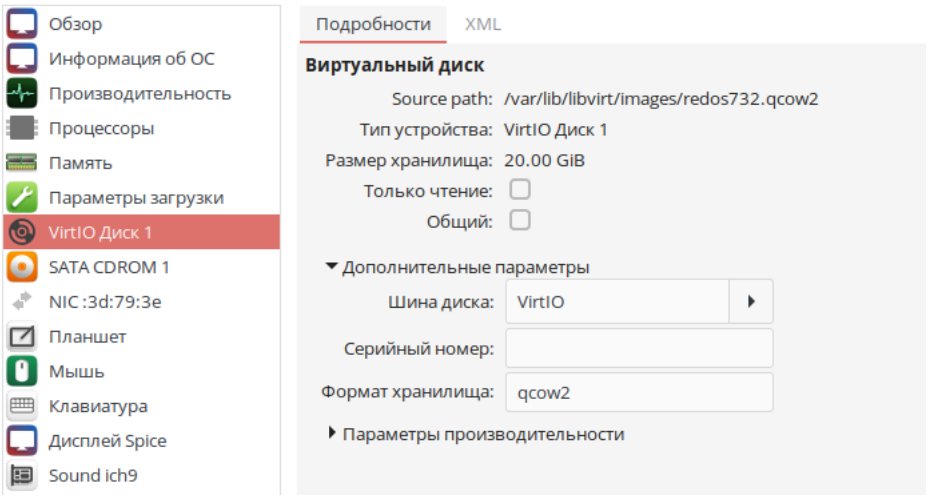

Рисунок 10.22 – Просмотр конфигурации диска

6. Для просмотра или изменения сетевой конфигурации выберите «NIC».

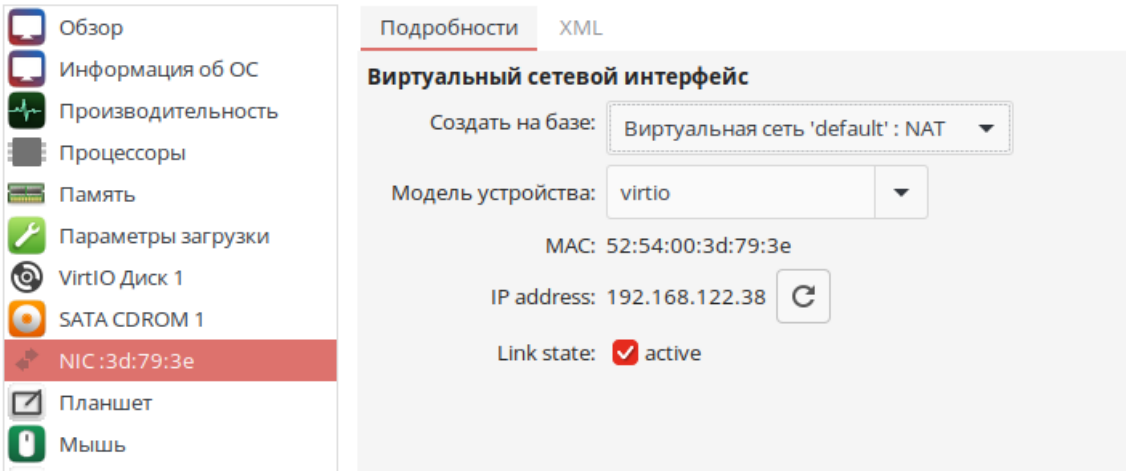

Рисунок 10.23 – Просмотр настроек сетевого интерфейса

Мониторинг состояния

С помощью менеджера ВМ можно редактировать настройки контроля статуса.

Для настройки мониторинга состояния и активации консолей в окне Менеджера ВМ в пункте меню «Правка» выберите «Параметры».

На вкладке «Статистика» укажите необходимый интервал обновления состояния ВМ.

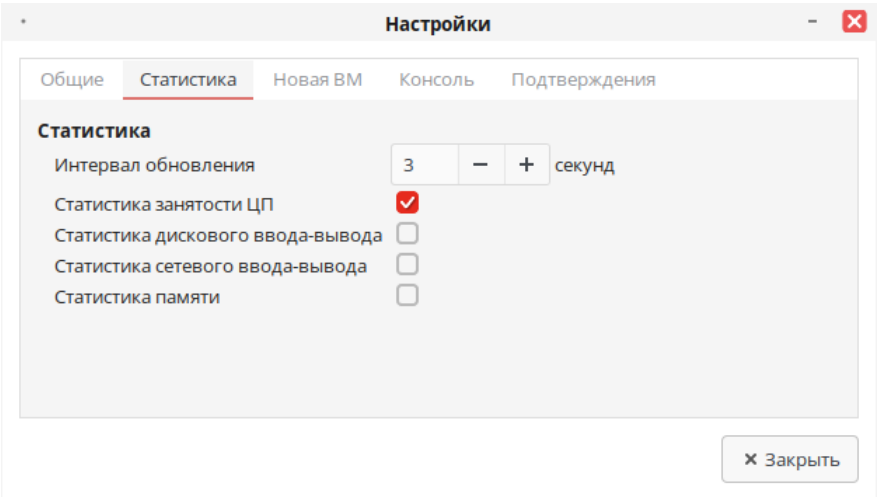

Рисунок 10.24 – Вкладка Статистика

На вкладке «Новая ВМ» выполните настройку графики, перенаправление USB, формат хранилища и ЦП.

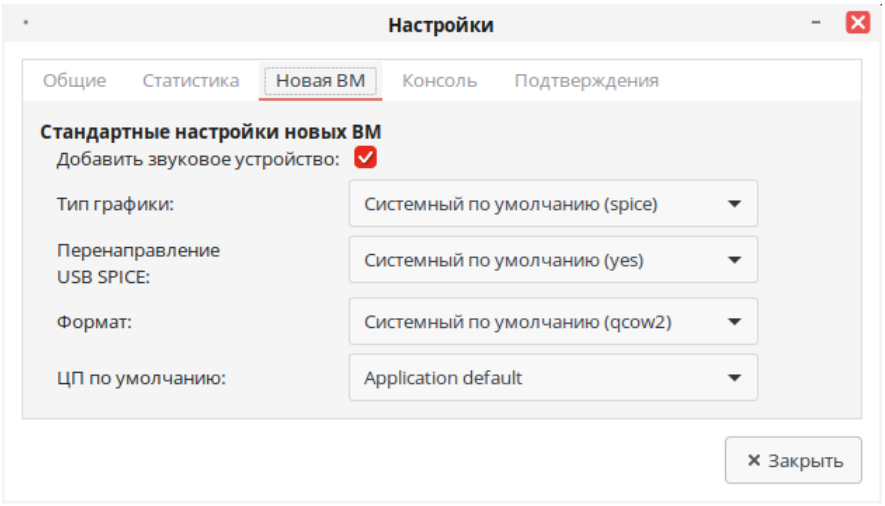

Рисунок 10.25 – Вкладка Новая ВМ

На вкладке «Консоль» можно настроить параметры графической консоли ВМ, такие как масштаб, разрешение окна, клавиши для освобождения курсора и пр.

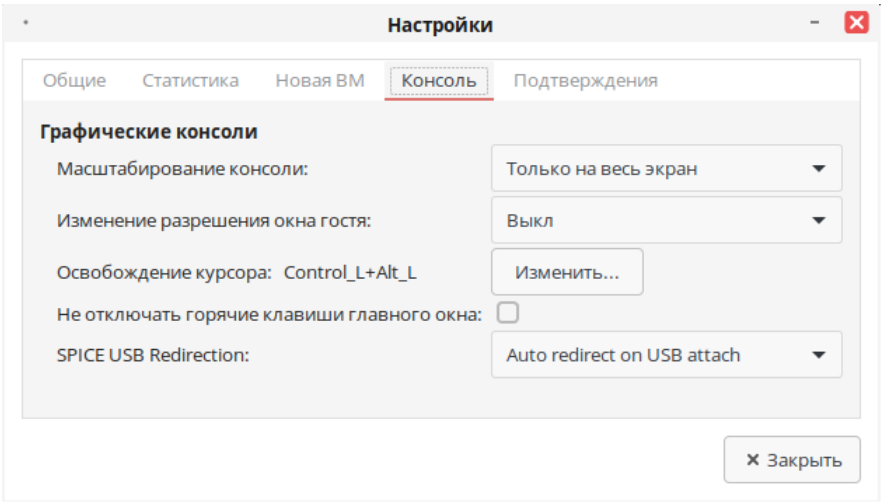

Рисунок 10.26 – Вкладка Консоль

На вкладке «Подтверждения» можно установить, на какие действия пользователя потребуется дополнительное подтверждение. Например, при выборе принудительного завершения сеанса будет выполнен дополнительный запрос, в котором пользователь может подтвердить данное действие или отказаться от него.

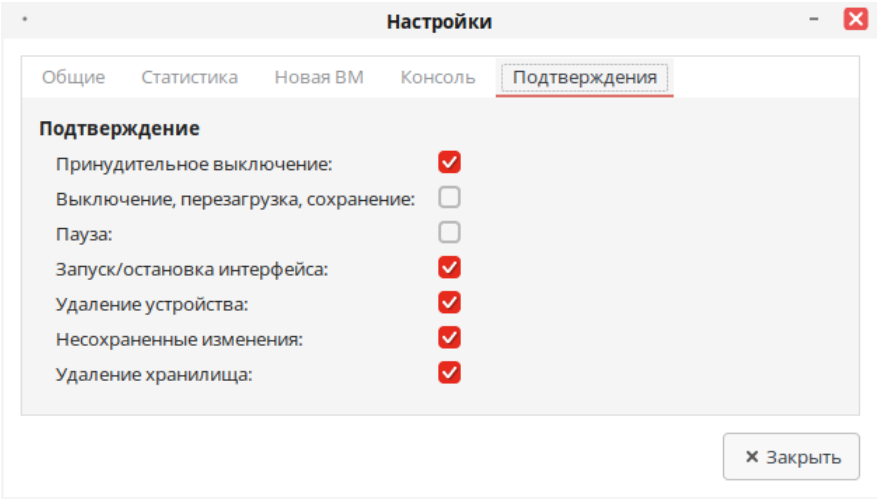

Рисунок 10.27 – Вкладка Подтверждения

Просмотр состояния виртуальной машины

Состояния ВМ отображаются в главном окне менеджера в списке созданных ВМ под их названием.

Состояния могут принимать следующие значения:

- работает;
- выключена;
- пауза;
- сохранено.

Рисунок 10.28 – Просмотр состояний ВМ

Управление виртуальной сетью

Для настройки виртуальных сетей необходимо:

1. В окне менеджера ВМ в пункте меню «Правка» выбрать «Свойства подключения».

2. Перейти на вкладку «Виртуальные сети».

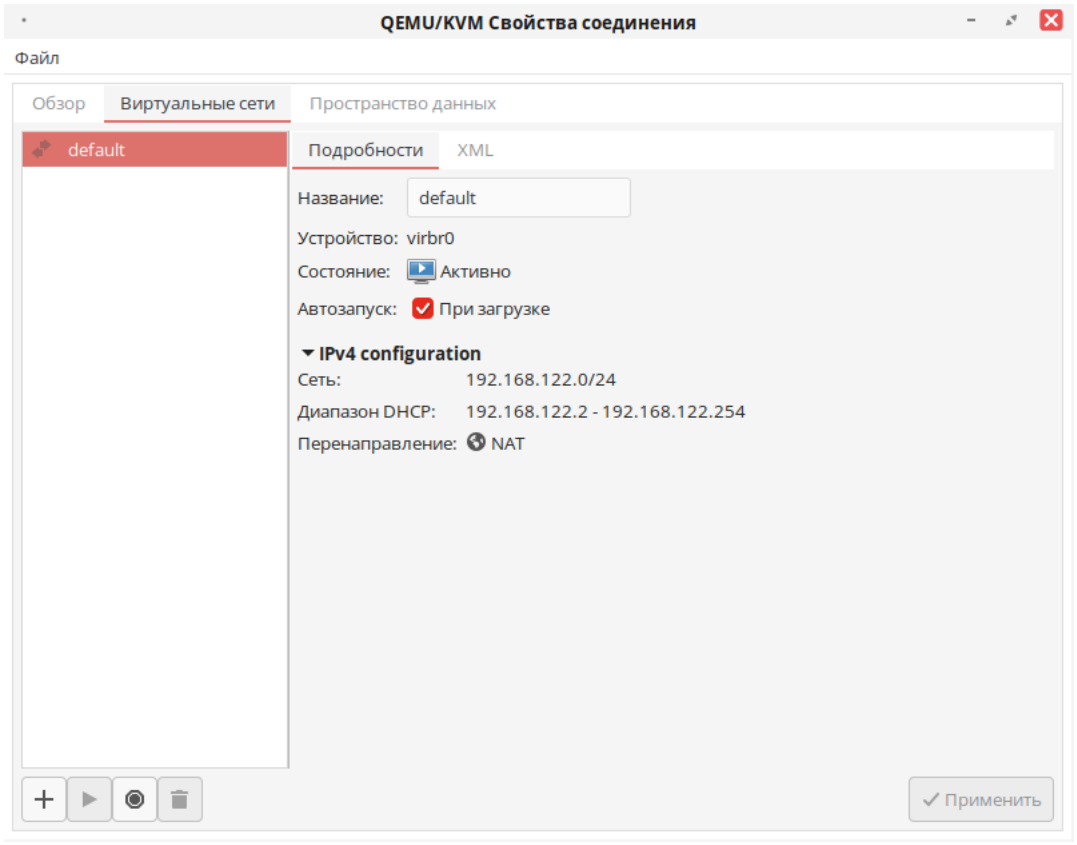

Рисунок 10.29 – Управление виртуальными сетями

3. Доступные виртуальные сети будут отображаться в левой части окна. Выберите сеть для доступа к ее настройкам.

4. Для сохранения внесенных изменений нажмите «Применить».

Создание виртуальной сети

Для создания новой виртуальной сети нажмите на кнопку + в левой части окна.

Будет открыто окно «Создание новой виртуальной сети», в котором необходимо указать название сети, выбрать из списка режим подключения и способ перенаправления сети.

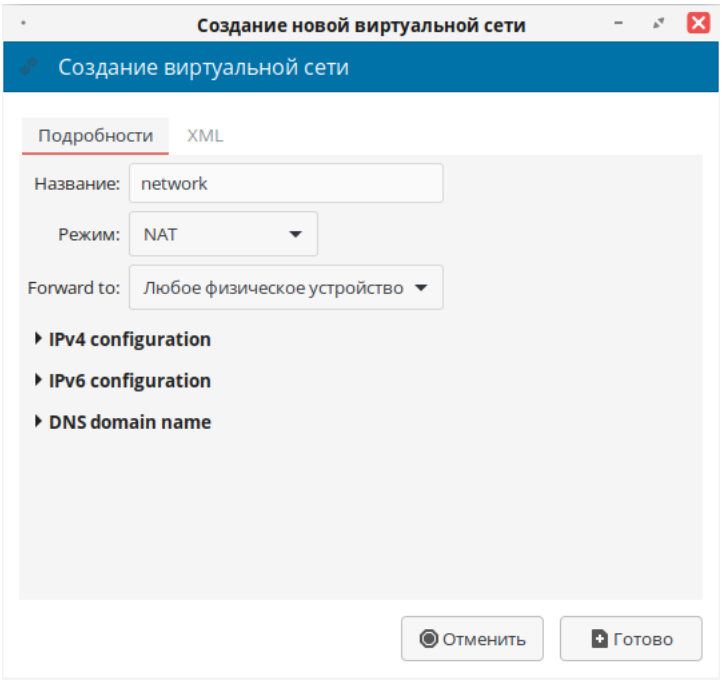

Рисунок 10.30 – Создание новой виртуальной сети

Далее потребуется ввести пространство адресов IPv4 для виртуальной сети и указать диапазон DHCP для вашей виртуальной сети, задав начальный и конечный адрес.

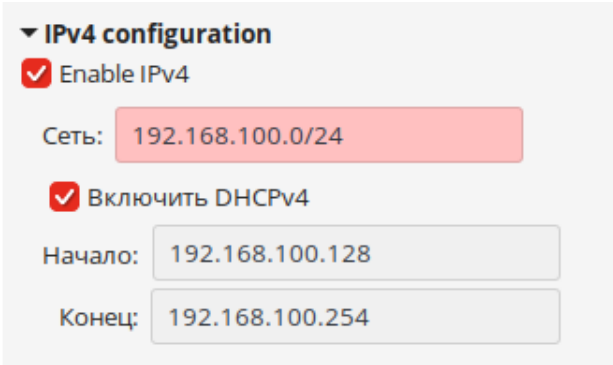

Рисунок 10.31 – Настройка пространства адресов IPv4

Проверьте правильность указанных данных и нажмите «Готово» для создания новой виртуальной сети.

Сведения о новой виртуальной сети можно получить на вкладке «Виртуальные сети» в окне «Свойства подключения».

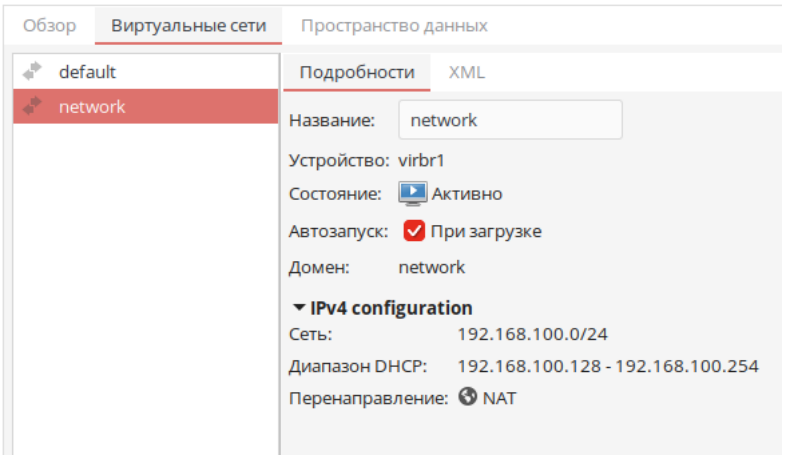

Рисунок 10.32 – Просмотр параметров новой виртуальной сети

Сохранение состояния ВМ

Сохранить текущее состояние ВМ можно двумя способами:

- из главного окна менеджера ВМ;
- из консоли ВМ.

Для сохранения состояния ВМ из главного окна менеджера выберите необходимую ВМ, нажмите ПКМ по ее имени. В контекстном меню выберите «Выключить» - «Сохранить».

Для сохранения состояния ВМ из консоли ВМ в пункте меню «Виртуальная машина» выберите «Выключить» - «Сохранить». После этого работа ВМ будет приостановлена и запустится процесс сохранения состояния.

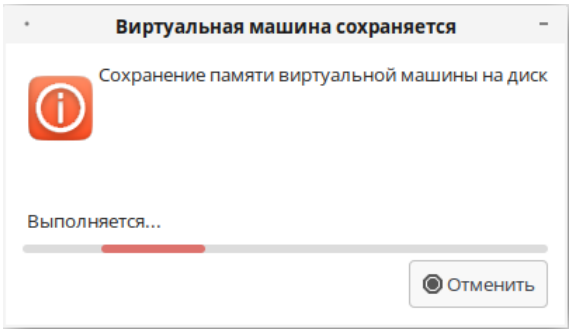

Рисунок 10.33 – Сохранение состояния ВМ

Сохраненные состояния ВМ доступны в каталоге /var/lib/libvirt/qemu/save.

Восстановление состояния ВМ

Восстановить ранее сохраненное состояние ВМ можно двумя способами:

- из главного окна менеджера ВМ;
- из консоли ВМ.

Для восстановления состояния ВМ из главного окна менеджера выберите необходимую ВМ, нажмите ПКМ по ее имени. В контекстном меню выберите

«Восстановить».

Для восстановления состояния ВМ из консоли ВМ в пункте меню «Виртуальная машина» выберите «Восстановить».

После этого запустится процесс восстановления состояния и ВМ будет запущена.

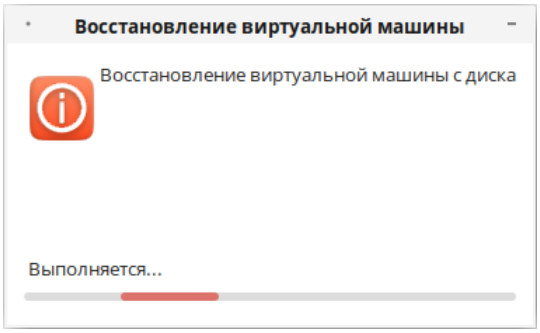

Рисунок 10.34 – Восстановление состояния ВМ

Работа со снимками ВМ

Для создания снимка (снапшота) ВМ необходимо открыть консоль ВМ и перейти на вкладку «Снимки», нажав кнопку «Управление снимками» ( ).

Для создания нового снимка ВМ нажмите кнопку «Создать новый снимок».

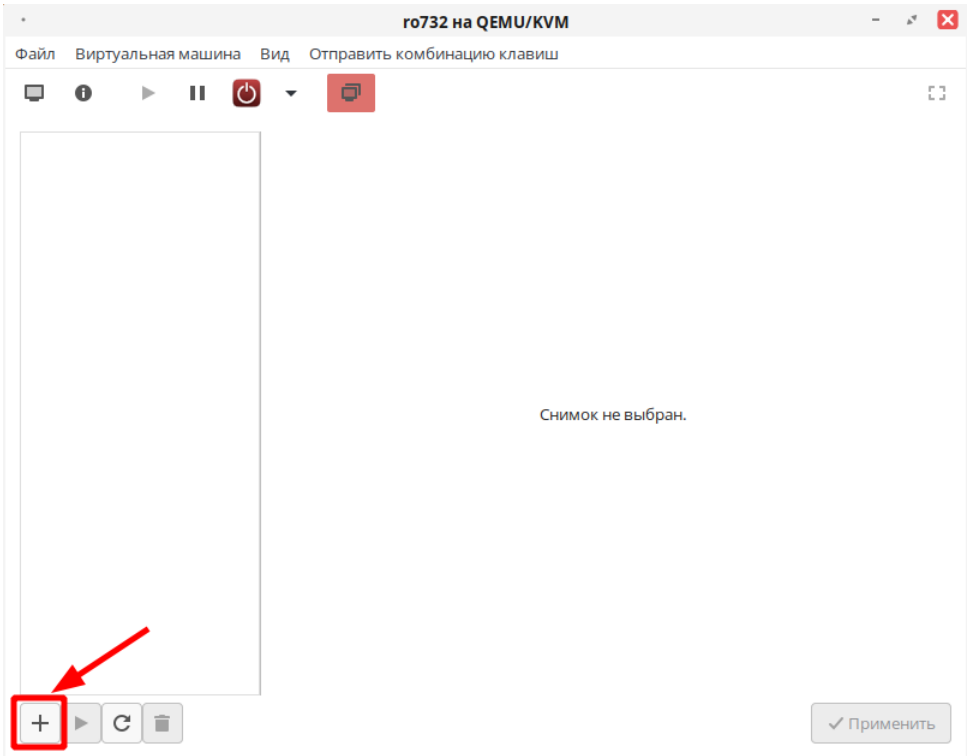

Рисунок 10.35 – Создание нового снимка ВМ

Будет открыто окно «Создание снимка», в котором необходимо указать название (обязательно) и описание (по желанию) снимка.

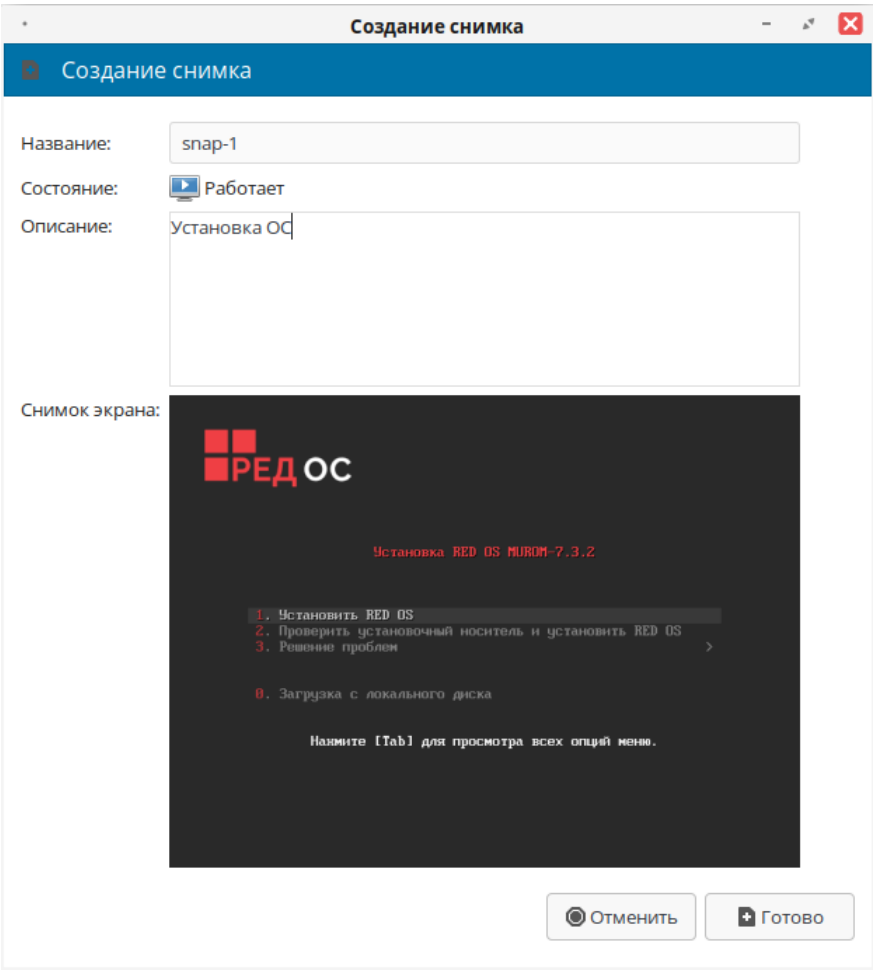

Рисунок 10.36 – Настройка параметров нового снимка ВМ

Нажмите кнопку «Готово» для запуска процесса создания снимка ВМ.

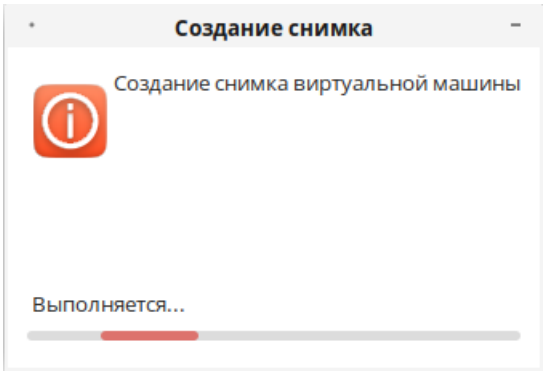

Рисунок 10.37 – Процесс создания нового снимка ВМ

После этого снимок будет доступен в списке для дальнейшей работы. При выборе конкретного снимка будет отображена подробная информация о нем.

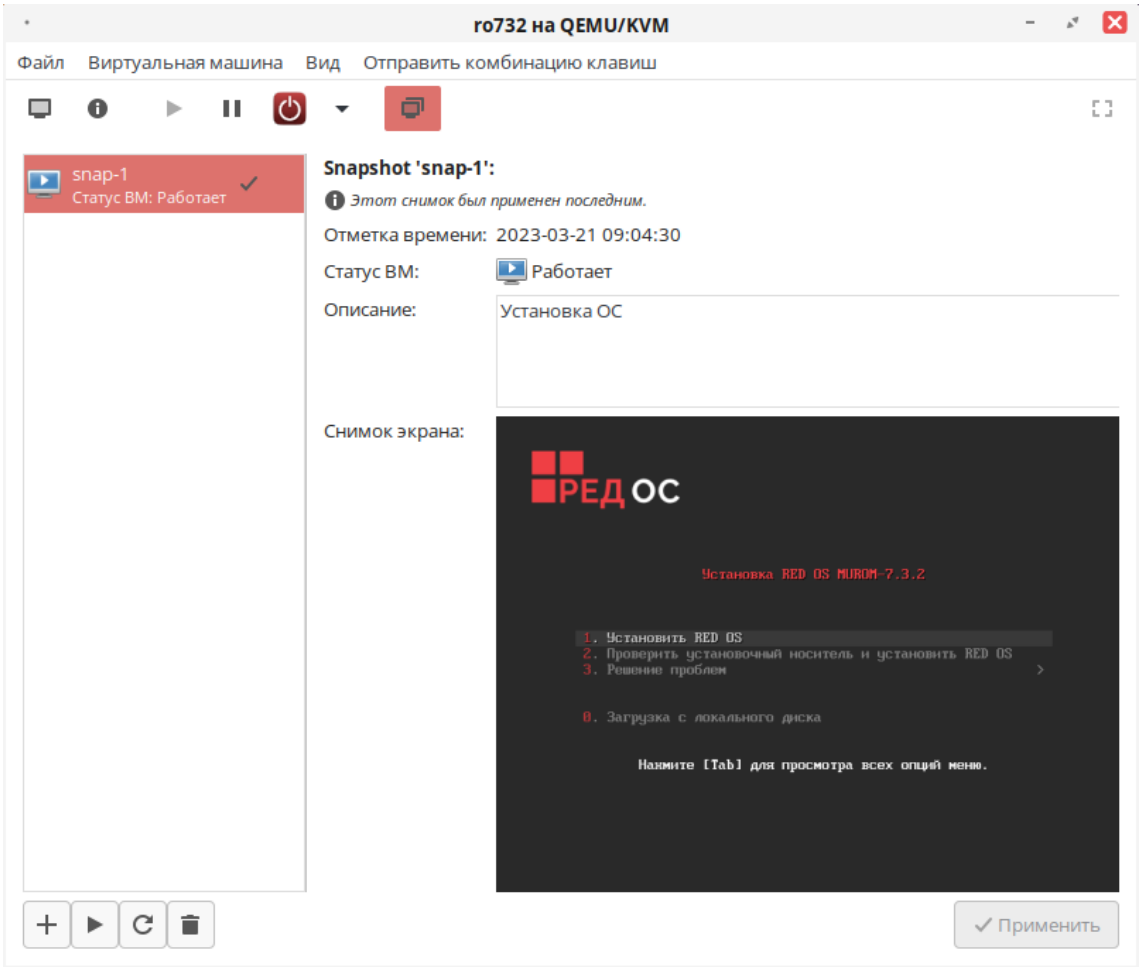

Рисунок 10.38 – Просмотр информации о снимке ВМ

Снимки можно восстанавливать и удалять, нажав ПКМ по нужному снимку либо выбрав снимок и нажав одну из кнопок — «Запуск выбранного снимка» ( ) или «Удалить выбранный снимок» ( ).

Обновить список доступных снимков можно, нажав кнопку «Refresh snapshot list» ( ).

Удаление ВМ

Для удаления ВМ в менеджере необходимо выбрать нужную ВМ и нажать ПКМ по ее имени. В открывшемся контекстном меню выберите «Удалить».

Далее потребуется указать необходимость удаления пространства хранения данных. После определения всех настроек нажмите «Удалить».

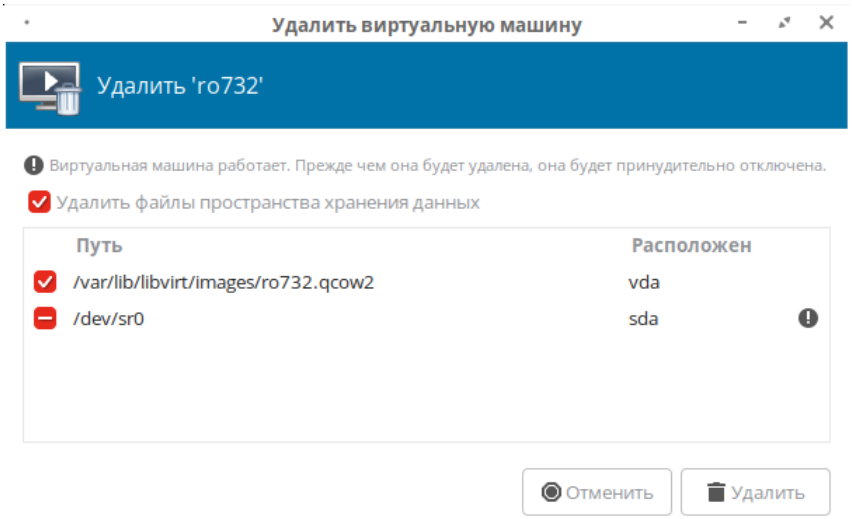

Рисунок 10.39 – Удаление ВМ

Подтвердите удаление ВМ и всех ее данных.

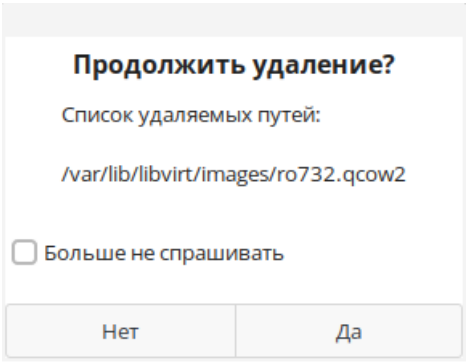

Рисунок 10.40 – Подтверждение удаления ВМ

После этого выбранная ВМ будет полностью удалена.

10.8 Безопасность в среде виртуализации

10.8.1 Доверенная загрузка ВМ

Средство виртуализации обеспечивает возможность доверенной загрузки ВМ. При выявлении нарушения целостности конфигурации виртуального оборудования ВМ средство виртуализации блокирует запуск данной ВМ.

Также средство виртуализации обеспечивает блокировку запуска ВМ при выявлении нарушения целостности файлов виртуальной базовой системы ввода-вывода (первичного загрузчика виртуальной машины) и исполняемых файлов гостевой операционной системы.

При попытке запуска ВМ с нарушением целостности конфигурации будет выведено сообщение об ошибке следующего вида:

```
ошибка: Failed to start domain 'ro732'
ошибка: доступ запрещён: доступ запрещён «QEMU»
```

10.8.2 Контроль целостности ВМ

Контроль целостности ВМ средства виртуализации обеспечивается посредством компонентов IMA и afick хостовой ОС.

Целостность ВМ обеспечивается как в процессе загрузки, так и динамически в процессе функционирования средства виртуализации. При нарушении целостности происходит немедленное информирование администратора безопасности средства виртуализации.

При попытке запуска ВМ, целостность которой была нарушена, будет выведено сообщение вида:

```
ошибка: ошибка при получении данных домена «ro732»
```

При этом в системном журнале будет зафиксировано уведомление для администратора безопасности примерно следующего вида:

```
Mar 24 17:28:12 localhost libvirtd[41711]: id=8f0d6547-cd2d-4148-
a285-e871024f3274, namespace=checksum, vm=ro732, result=deny
```

Также средством виртуализации фиксируются попытки нарушения целостности журналов **/var/log/libvirt/libvirt.log** и **/var/log/audit/audit.log**.

Записи в журналах имеют выглядят примерно следующим образом:

```
2023-03-27 10:27:04,425 CRITICAL id=187188a8-3b7e-4508-9d66-
81d59ede036b, namespace=journal-checksum, result=WRONG
```

10.8.3 Ограничение программной среды

Для осуществления контроля за запуском компонентов программного обеспечения используется инструмент архитектуры измерения целостности хостовой ОС — IMA. Данный инструмент позволяет выявлять и блокировать запуск компонентов программного обеспечения, не включенных в перечень (список) компонентов, разрешенных для запуска.

Также инструмент IMA обеспечивает блокировку запуска компонентов программного обеспечения, целостность которого нарушена, а также компонентов, не прошедших аутентификацию с использованием свидетельств подлинности модулей.

Подробную информацию о работе и настройке инструмента IMA см. в разделе 6.16 «Контроль целостности запускаемых компонентов» настоящего руководства.

10.8.4 Резервное копирование

Средство виртуализации обеспечивает резервное копирование образов ВМ и конфигураций виртуального оборудования ВМ. Также поддерживается резервное копирование параметров настройки средства виртуализации и сведений о событиях безопасности.

Для резервного копирования в средстве виртуализации используются следующие утилиты:

- libvirt-backup-settings — для создания резервных копий файлов конфигурации средства виртуализации;
- libvirt-backup-vm — для создания резервных копий ВМ и их файлов.

Синтаксис утилиты **libvirt-backup-settings** имеет следующий вид:

```
libvirt-backup-settings <каталог_для_РК>
```

В качестве обязательного аргумента утилите передается путь к каталогу, в котором будут храниться резервные копии параметров настроек средства виртуализации. В процессе создания резервной копии будет сформирован архив вида **libvirt-settings-<timestamp>.tar.gz**, в нем будут храниться все файлы настроек средства виртуализации.

Синтаксис утилиты **libvirt-backup-vm** имеет следующий вид:

```
libvirt-backup-vm <имя_ВМ> <каталог_для_РК>
```

Для создания резервной копии утилите необходимо передать два обязательных аргумента — имя ВМ и путь к каталогу хранения резервных копий. В процессе создания резервной копии так же будет сформирован архив вида **vm-<имя_ВМ>-<timestamp>.tar.gz**.

В среде виртуализации также поддерживается резервное копирование журналов безопасности. Для формирования архива с отметкой времени создания резервной копии используется команда вида:

```
tar -czf <каталог_для_РК>/libvirt-sec-journals-$(date +%Y-%m-%d_
%H-%M-%S).tar.gz /var/log/libvirt/libvirt.log* /var/log/pam.ext.log*
/var/log/audit/audit.log*
```

Создание расписания резервного копирования

Для создания расписания резервного копирования необходимо воспользоваться настройкой службы cron хостовой ОС. Для редактирования файла необходимо с правами пользователя root выполнить команду:

```
crontab -e
```

Каждая строка в системной конфигурации **cron**, расположенной в каталоге /**etc**, состоит из шести полей и команды, необходимой для выполнения:

```
<минута> <час> <число> <месяц> <день_недели> <пользователь> <ко-
манда>
```

Для резервного копирования настроек средства виртуализации в файл cron необходимо добавить скрипт:

```
/usr/sbin/libvirt-backup-settings <каталог_для_РК>
```

Для резервного копирования ВМ в файл cron необходимо добавить скрипт:

```
/usr/sbin/libvirt-backup-vm <имя_ВМ> <каталог_для_РК>
```

Для резервного копирования журналов можно использовать следующую команду:

```
tar -czf <каталог_для_РК>/libvirt-sec-journals-$(date +%Y-%m-%d_
%H-%M-%S).tar.gz /var/log/libvirt/libvirt.log* /var/log/pam.ext.log*
/var/log/audit/audit.log*
```

После этого в указанном каталоге <каталог_для_РК> будут созданы архивы с резервными копиями формата:

- для настроек средства виртуализации — libvirt-settings-<timestamp>.tar.gz;
- для виртуальных машин — vm-<имя_ВМ>-<timestamp>.tar.gz;
- для журналов безопасности — libvirt-sec-journals-<timestamp>.tar.gz.

10.8.5 Защита памяти

В средстве виртуализации обеспечивается очистка остаточной информации из памяти хоста при ее освобождении (распределении), а также удаление объектов файловой системы, используемых средством виртуализации.

Очистка памяти производится с помощью утилиты **virsh**. Уничтожение объектов файловой системы производится путем многократного перезаписывания стираемых объектов.

Для очищения только содержимого томов хранения ВМ используется команда **virsh vol-wipe**. Она позволяет безопасно уничтожить данные, ранее хранившиеся на томе без возможности чтения их в будущем. Можно использовать различные алгоритмы удаления, применив аргумент **--algorithm**. Аргумент может принимать следующие значения:

- **zero** — 1 проход записи нулями;
- **nnsa** — 3 прохода алгоритмом NNSA NAP-14.1-C;
- **dod** — 3 прохода алгоритмом DoD 5220.22-M;
- **bsi** — 9 проходов алгоритмом BSI;
- **random** — 1 проход случайной последовательностью;
- **random2** — 2 прохода случайной последовательностью;
- **schneier** — 7 проходов алгоритмом Брюса Шнайера;
- **pfitzner7** — 7 случайных проходов алгоритмом Роя Пфитцнера;
- **pfitzner33** — 33 случайных прохода алгоритмом Роя Пфитцнера;
- **gutmann** — 35 проходов алгоритмом Гутмана.

Алгоритмы становятся доступны к использованию после установки утилиты scrub на хостовой ОС. Утилита перезаписывает указанный файл (хранилище) случайными данными, чтобы скрыть его содержимое, и, в случае необходимости, удаляет его.

Для установки утилиты на хостовой ОС выполните команду:

```
dnf install scrub
```

После успешной установки можно перейти к очистке томов хранения ВМ. Для этого выполните команду вида:

```
virsh vol-wipe <том> [<пул_хранения>] [--algorithm <метод>]
```

Если метод очистки тома не задан, по умолчанию будет использоваться метод **nnsa**.

Пример использования утилиты:

```
vol-wipe /var/lib/libvirt/images/redos.qcow2 --algorithm bsi
```

```
Том /var/lib/libvirt/images/redos.qcow2 очищен
```

Для полного удаления как содержимого томов хранения, так и самих ВМ используется команда **virsh undefine** с дополнительными параметрами **--wipe-storage** и **--remove-all-storage**. Команда позволяет полностью удалить всю информацию о ВМ — тома хранения, конфигурационные файлы — без возможности восстановления.

Для полного удаления ВМ выполните команду вида:

```
virsh undefine <домен> --wipe-storage --remove-all-storage
```

Пример использования команды:

```
virsh undefine redos --wipe-storage --remove-all-storage
```

```
Domain 'redos' has been undefined
Освобождение тома «vda» (/var/lib/libvirt/images/redos.qcow2)...
Завершено.
Том «vda» (/var/lib/libvirt/images/redos.qcow2) был удалён.
Освобождение тома «vdb» (/var/lib/libvirt/images/redos-1.qcow2)...
Завершено.
Том «vdb» (/var/lib/libvirt/images/redos-1.qcow2) был удалён.
```

Все привязанные к домену тома будут очищены и удалены.

10.8.6 Управление потоками информации

Сетевые фильтры позволяют администраторам виртуализированных систем настраивать и применять правила фильтрации сетевого трафика на ВМ, а также управлять его параметрами.

Правила сетевых фильтров применяются на хостовой ОС при запуске ВМ и могут быть изменены в процессе ее работы. Несколько ВМ могут использовать один общий сетевой фильтр. При редактировании конфигурации такого фильтра обновляются и правила фильтрации сетевого трафика всех запущенных ВМ, использующих данный фильтр.

Подсистема фильтрации сетевого трафика позволяет настроить правила фильтрации на отдельных сетевых интерфейсах, которые настроены на определенные типы сетевых конфигураций. Поддерживаются следующие типы сетей:

- network;
- ethernet — должен использоваться в режиме bridge;
- bridge.

Сетевые фильтры записываются в формате XML и могут содержать ссылки на другие фильтры, правила фильтрации трафика либо их комбинацию.

Правила фильтрации могут быть организованы в связанные цепочки фильтров. Их можно представить в виде древовидной структуры, в которой отдельные цепочки фильтров будут являться «ветвями» этого дерева.

Пакеты проверяются фильтром в корневой цепочке, а затем могут быть перемещены и обработаны в других (отдельных) цепочках, после чего пакеты либо возвращаются обратно в корневую цепочку, либо принимаются (или отклоняются) правилами фильтрации в отдельной цепочке.

Система сетевой фильтрации средства виртуализации автоматически создает отдельные корневые цепочки для каждого сетевого интерфейса виртуальной машины, на котором пользователь активирует фильтрацию трафика. Пользователь может прописать правила фильтрации, которые непосредственно включаются в корневую цепочку, либо создавать цепочки фильтрации, специфичные для конкретного протокола.

Существуют следующие цепочки фильтрации трафика:

- root;
- mac;
- stp (spanning tree protocol);
- vlan (802.1Q);
- arp, rarp;
- ipv4;
- ipv6.

У каждой цепочки фильтров есть свой приоритет выполнения. В таблице ниже приведены приоритеты, установленные по умолчанию.

Цепочка (префикс)	**Приоритет**
stp	-810
mac	-800
vlan	-750
ipv4	-700
ipv6	-600
arp	-500
rarp	-400

Доступ к префиксам с более низким значением приоритета осуществляется

раньше, чем к префиксам с более высоким значением.

Переменные фильтров

Для использования подсистемой фильтрации сетевого трафика зарезервированы два имени переменных: **MAC** и **IP**.

MAC — это MAC-адрес сетевого интерфейса. Правило фильтрации, которое ссылается на эту переменную, будет автоматически создано с MAC-адресом интерфейса.

IP представляет собой IP-адрес, который гостевая ОС должна использовать на данном интерфейсе.

Элементы фильтров

Главный элемент, необходимый для всех сетевых фильтров, называется **filter** и имеет два возможных атрибута.

Атрибут **name** предоставляет уникальное имя данного фильтра.

Атрибут **chain** является необязательным, но позволяет лучше организовать определенные фильтры для более эффективной обработки подсистемой брандмауэра основного хоста.

В настоящее время система поддерживает только цепочки root, ipv4, ipv6, arp и rarp.

Ссылки на другие фильтры

Любой фильтр может содержать ссылки на другие фильтры. На отдельные фильтры можно ссылаться несколько раз в дереве фильтров, но ссылки между фильтрами не должны создавать циклов.

Например:

```
<filter name='clean-traffic'>
  <uuid>6ef53069-ba34-94a0-d33d-17751b9b8cb1</uuid>
  <filterref filter='no-mac-spoofing'/>
  <filterref filter='no-ip-spoofing'/>
  <filterref filter='allow-incoming-ipv4'/>
  <filterref filter='no-arp-spoofing'/>
  <filterref filter='no-other-l2-traffic'/>
  <filterref filter='qemu-announce-self'/>
</filter>
```

В данной конфигурации сетевой фильтр **clean-traffic** ссылается на несколько других фильтров.

Чтобы сослаться на другой фильтр, параметр **filterref** должен быть указан в значении **filter**. Данное значение должно содержать имя фильтра, на который нужно сослаться.

Новые сетевые фильтры могут быть определены в любое время и могут содержать ссылки на новые сетевые фильтры. Однако после запуска виртуальной машины все сетевые фильтры в дереве фильтров должны быть доступны. В противном случае виртуальная машина не запустится.

Правила фильтрации

Правило фильтрации всегда начинается с параметра rule, который может содержать до трех атрибутов:

- action — обязательный атрибут, может принимать следующие значения:
  - drop — отклоняет пакет без дальнейшего анализа;
  - reject — генерирует сообщение об отказе без дальнейшего анализа;
  - accept — принимает пакет без дальнейшего анализа;
  - return — проходит фильтр, при этом возвращая управление вызывающему фильтру для дальнейшего анализа;
  - continue — переходит к следующему правилу для дальнейшего анализа.
- direction — обязательный параметр, может принимать следующие значения:
  - in — если правило предназначено для входящего трафика;
  - out — если правило предназначено для исходящего трафика;
  - inout — если правило предназначено для входящего и исходящего трафика.
- priority — необязательный параметр, задает приоритет правила и контролирует порядок, в котором правило будет выполняться относительно других;
- statematch — необязательный параметр, возможные значения - «0» или «false» — для отключения проверки состояния базового соединения; по умолчанию «true».

Пример использования правил фильтрации:

```
<filter name='no-ip-spoofing'chain='ipv4'>
  <uuid>fce8ae33-e69e-83bf-262e-30786c1f8072</uuid>
  <rule action='drop'direction='out'priority='500'>
    <ip match='no'srcipaddr='$IP'/>
  </rule>
</filter>
```

Пример использования сетевого фильтра

Для настройки сетевых фильтров на хостовой ОС необходимо определить MAC-адрес шлюза командой:

```
ifconfig
```

```
...
virbr0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
  inet 192.168.122.1 netmask 255.255.255.0 broadcast 192.168.122.
255
  ether 52:54:00:d8:21:b1 txqueuelen 1000 (Ethernet)
  RX packets 21402 bytes 1187793 (1.1 MiB)
  RX errors 0 dropped 0 overruns 0 frame 0
  TX packets 35100 bytes 77208945 (73.6 MiB)
  TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Затем необходимо разрешить редактирование XML-конфигураций ВМ. Для этого в менеджере ВМ требуется выбрать пункт меню «Правка» - «Параметры» и на вкладке «Общие» установить флаг для параметра «Enable XML editing».

После этого в консоли ВМ необходимо открыть подробные сведения об оборудовании. Далее следует открыть вкладку настроек сети **NIC**, перейти в настройки **XML** и назначить фильтр на сетевой интерфейс.

Для этого в конфигурации вручную должны быть прописаны следующие строки:

```
<filterref filter="clean-traffic-gateway">
   <parameter name="GATEWAY_MAC" value="52:54:00:d8:21:b1"/>
</filterref>
```

где **GATEWAY_MAC** — это MAC-адрес шлюза.

Примечание. Внесенные в XML-конфигурацию изменения вступят в силу **после выключения** ВМ.

Итоговая настройка будет выглядеть примерно следующим образом:

Рисунок 10.41 – Настройка сетевого фильтра

Описание правила clean-traffic можно просмотреть командой:

```
virsh nwfilter-dumpxml clean-traffic
```

```
<filter name='clean-traffic' chain='root'>
  <uuid>3a4de05c-e2e4-4e7c-952f-741160051ad0</uuid>
  <filterref filter='no-mac-spoofing'/>
  <filterref filter='no-ip-spoofing'/>
  <rule action='accept' direction='out'priority='-650'>
    <mac protocolid='ipv4'/>
  </rule>
  <filterref filter='allow-incoming-ipv4'/>
  <filterref filter='no-arp-spoofing'/>
  <rule action='accept' direction='inout'priority='-500'>
    <mac protocolid='arp'/>
  </rule>
```

```
    <filterref filter='no-other-l2-traffic'/>
    <filterref filter='qemu-announce-self'/>
  </filter>
```

10.8.7 Регистрация событий безопасности

Средство виртуализации обеспечивает регистрацию событий безопасности и осуществляет сбор и хранение записей в журналах событий безопасности.

К событиям безопасности относятся следующие события:

- успешные и не успешные попытки аутентификации пользователей средства виртуализации;
- доступ пользователей средства виртуализации к виртуальным машинам;
- создание и удаление виртуальных машин;
- запуск и остановка средства виртуализации с указанием причины остановки;
- запуск и остановка виртуальных машин с указанием причины остановки;
- изменение ролевой модели;
- изменение конфигурации средства виртуализации;
- изменение конфигураций виртуальных машин;
- факты нарушения целостности объектов контроля.

К журналам событий безопасности относятся следующие файлы:

- /var/log/libvirt/libvirt.log – регистрация действий в средстве виртуализации;
- /var/log/pam.ext.log – регистрация успешных и не успешных попыток аутентификации пользователей средства виртуализации;
- /var/log/audit/audit.log – регистрация событий аудита, таких как запуск / остановка сервисов, предоставление прав, доступ к объектам контроля и пр.

Журналы безопасности доступны только для чтения и только пользователю с правами администратора безопасности средства виртуализации (*security-admin*).

Для чтения журналов безопасности используется утилита **libvirt-journal-read**.

Синтаксис утилиты имеет следующий вид:

```
libvirt-journal-read <main | pam | audit>
```

В качестве аргументов утилита принимает один из следующих параметров:

- main — выводит содержимое журнала /var/log/libvirt/libvirt.log;
- pam — выводит содержимое журнала /var/log/pam.ext.log:
- audit — выводит содержимое журнала /var/log/audit/audit.log.

Записи в журналах событий безопасности имеют примерно следующий вид:

- записи журнала /var/log/libvirt/libvirt.log:

```
 2023-04-06 10:01:23,669 INFO id=d9e9fbda-f326-4cec-8ed9-a6088ea-
22806, namespace=RBAC, permission=org.libvirt.rbac, (user=virt-sup,
```

```
session_id=16), result=allow
   2023-04-06 10:01:38,699 INFO id=86762d9a-5c30-45d4-8ad2-f9e41f8f-
9c5c, namespace=RBAC, permission=org.libvirt.rbac, (user=virt-sup,
session_id=16), result=allow
   2023-04-06 10:01:38,834 WARNING id=d07d04ec-5f67-4def-a3b5-2a31c-
f615e4b, namespace=RoleModel, user=virt-sup, role=security-admin,
action=revoke-role
   2023-04-06 10:01:44,215 INFO id=9eca307e-a44b-48e8-89ea-b471ceaa-
c173, namespace=RBAC, permission=org.libvirt.journal, (user=virt-sup,
session_id=16), result=deny
```

- записи журнала /var/log/pam.ext.log:

```
   [Thu Apr 6 09:20:03 2023] [e4901b4a-fe6c-4ab5-ba13-2e43858b6cb0]
INFO Session opened for user gdm.
   [Thu Apr 6 09:20:19 2023] [f1ac23ca-a98a-40fb-bbf1-2ac25e22528d]
INFO User sec-adm logging in.
   [Thu Apr 6 09:20:19 2023] [2cccb4ed-3d28-4ad7-8925-dfd1d92dfcda]
INFO Session opened for user sec-adm.
   [Thu Apr 6 09:20:19 2023] [aaa1ded6-cac7-40b0-a1da-93007185eea9]
INFO Session opened for user sec-adm.
```

- записи журнала/var/log/audit/audit.log:

```
   type=USER_END msg=audit(1680764764.307:3402937): pid=2930119
uid=0 auid=4294967295 ses=4294967295 subj=system_u:system_r:policy-
kit_t:s0 msg='op=PAM:session_close grantors=pam_keyinit,pam_limits,
pam_systemd,pam_unix acct="root" exe="/usr/bin/sudo" hostname=?
addr=? terminal=? res=success'UID="root" AUID="unset"
   type=CRED_DISP msg=audit(1680764764.307:3402938): pid=2930119
uid=0 auid=4294967295 ses=4294967295 subj=system_u:system_r:policy-
kit_t:s0 msg='op=PAM:setcred grantors=pam_env,pam_faillock,pam_unix
acct="root" exe="/usr/bin/sudo" hostname=? addr=? terminal=? res=suc-
cess'UID="root" AUID="unset"
   type=USER_START msg=audit(1680764764.328:3402939): pid=2930092
uid=1004 auid=1004 ses=16 subj=unconfined_u:unconfined_r:unconfined-
_t:s0-s0:c0.c1023 msg='op=PAM:session_open grantors=pam_keyinit,
pam_limits,pam_systemd,pam_unix acct="root" exe="/usr/bin/pkexec"
hostname=localhost.localdomain addr=? terminal=pts/5 res=success'
UID="sec-adm" AUID="sec-adm"
```

11. Средство контейнеризации

11.1 Общие сведения

Контейнеризация — метод виртуализации, при котором ядро операционной системы поддерживает несколько изолированных экземпляров пространства пользователя вместо одного. Эти экземпляры (обычно называемые контейнерами) с точки зрения выполняемых в них процессов идентичны отдельному экземпляру операционной системы. Ядро обеспечивает полную изолированность контейнеров, поэтому программы из разных контейнеров не могут воздействовать друг на друга.

В отличие от аппаратной виртуализации, при которой эмулируется аппаратное окружение и может быть запущен широкий спектр гостевых операционных систем, в контейнере может быть запущен экземпляр операционной системы только с тем же ядром, что и у хостовой операционной системы (все контейнеры узла используют общее ядро). При этом при контейнеризации отсутствуют дополнительные ресурсные накладные расходы на эмуляцию виртуального оборудования и запуск полноценного экземпляра операционной системы, характерные при аппаратной виртуализации.

В РЕД ОС в качестве средства контейнеризации используется контейнерная платформа Docker.

Docker — программное обеспечение для автоматизации развертывания и управления приложениями в средах с поддержкой контейнеризации. Позволяет "упаковать" приложение со всем его окружением и зависимостями в контейнер, который может быть развернут на любой Linux-подобной системе с поддержкой контрольных групп в ядре, а также предоставляет набор команд для управления этими контейнерами.

Для экономии пространства хранения проект использует файловую систему aufs с поддержкой технологии каскадно-объединенного монтирования: контейне-

ры используют образ базовой операционной системы, а изменения записываются в отдельную область. Также поддерживается размещение контейнеров в файловой системе btrfs с включенным режимом копирования при записи.

В состав программных средств входит демон — сервер контейнеров, клиентские средства, позволяющие из интерфейса командной строки управлять образами и контейнерами, а также REST API, позволяющий управлять контейнерами программно. Демон обеспечивает полную изоляцию запускаемых на узле контейнеров на уровне файловой системы (у каждого контейнера собственная корневая файловая система), на уровне процессов (процессы имеют доступ только к собственной файловой системе контейнера, а ресурсы разделены средствами containerd), на уровне сети (каждый контейнер имеет доступ только к привязанному к нему сетевому пространству имен и соответствующим виртуальным сетевым интерфейсам).

Набор клиентских средств позволяет запускать процессы в новых контейнерах, останавливать и запускать контейнеры, приостанавливать и возобновлять процессы в контейнерах. Набор команд позволяет осуществлять мониторинг запущенных процессов. Новые образы возможно создавать из специального файла, также возможно записать все изменения, произведенные в контейнере, в новый образ. Все команды могут работать как с docker-сервисом локальной системы, так и с любым сервером Docker, доступным по сети.

Кроме того, в интерфейсе командной строки встроены возможности по взаимодействию с различными централизованными хранилищами образов, в которых размещены предварительно собранные образы контейнеров.

11.2 Установка среды контейнеризации

Существует два варианта установки среды контейнеризации:

- установка в процессе развертывания новой системы РЕД ОС;
- установка в существующей системе РЕД ОС.

11.2.1 Установка в процессе развертывания новой системы РЕД ОС

После запуска интерактивной установки РЕД ОС с установочного носителя (USB, CD-ROM, PXE) параметры будущей системы настраиваются в штатном режиме (выбор языка, часового пояса, раскладки клавиатуры, источника установки, настройка сети), кроме параметра «Выбор программ».

Для развертывания среды контейнеризации необходимо выполнить установку РЕД ОС конфигурации Сервер (с графическим интерфейсом или минимальный).

При выборе конфигурации Сервер (с графическим интерфейсом или минимальный) следует определить дополнительное программное обеспечение — *Средства контейнеризации.*

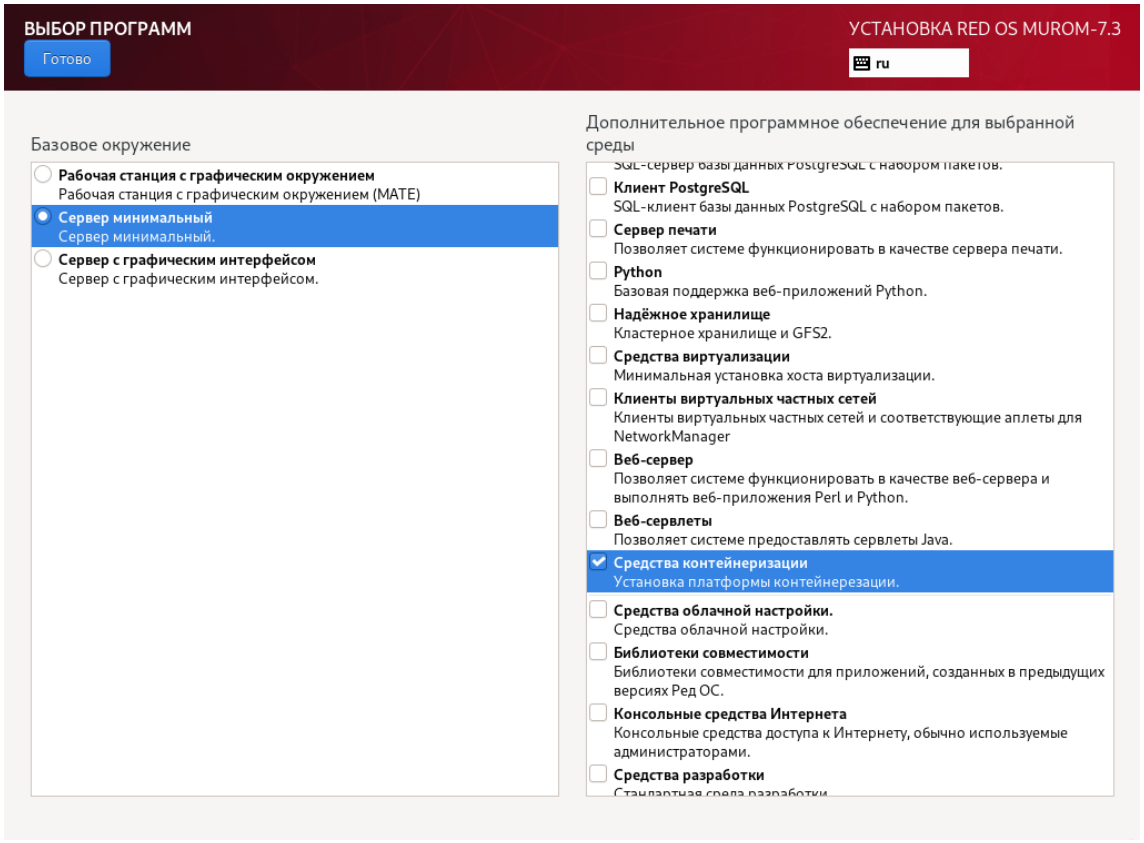

Рисунок 11.1 – Установка средства контейнеризации

После определения всех настроек нажмите кнопку «Начать установку».

Для завершения процесса установки потребуется перезагрузить систему и принять Лицензионное соглашение.

После авторизации в системе необходимо проверить статус сервиса docker командой:

```
systemctl status docker
```

В статусе должно быть указано **active (running)**. Если по каким-либо причинам статус не активен, выполните команду запуска сервиса с правами пользователя root:

```
systemctl start docker
```

После этого ОС будет готова к использованию.

11.2.2 Установка в существующей системе РЕД ОС

Для развертывания среды контейнеризации необходимо выполнить команду:

```
dnf install docker-ce docker-ce-cli
```

После успешной установки необходимо запустить сервис контейнеризации docker и добавить его в автозагрузку:

```
systemctl enable docker --now
```

Убедитесь, что сервис запущен, проверив статус запущенной службы:

```
systemctl status docker
```

В статусе должно быть отображено **active (running)**.
Для получения информации об установленном docker выполните команду:

```
docker info
```

При корректной настройке будет получен соответствующий ответ от сервиса Docker.

11.3 Доступ к среде контейнеризации

По умолчанию доступ к среде контейнеризации и запуску сервисов имеет только суперпользователь. Демон docker подключается к сокету Unix, к которому также имеет доступ только суперпользователь.

Для использования и управления средой контейнеризации обычным пользователем необходимо добавить его в отдельную группу, пользователям которой будет разрешено выполнять необходимые манипуляции. Такая группа создается в процессе установки среды контейнеризации и имеет название **docker**.

Добавьте необходимого пользователя в группу **docker** командой:

```
usermod -aG docker <имя_пользователя>
```

Для применения изменений выйдите из сеанса пользователя и авторизуйтесь снова либо перезагрузите систему. Проверить назначенные права можно, выполнив загрузку тестового образа с правами обычного пользователя:

```
docker info
```

При корректной настройке будет получен соответствующий ответ от сервиса Docker.

11.4 Компоненты средства контейнеризации

Docker использует клиент-серверную архитектуру. Клиент взаимодействует с демоном Docker, который выполняет задачи по созданию, запуску и распространению контейнеров Docker. Клиент и демон могут работать в одной системе, также можно подключить клиент к удаленному демону Docker. Клиент и демон взаимодействуют с помощью REST API, через сокеты UNIX или сетевой интерфейс.

11.4.1 Dockerfile

Dockerfile — это простой текстовый файл, включающий в себя инструкции по созданию контейнера Docker. Dockerfile описывает будущую операционную

систему, которая будет лежать в основе контейнера, а также языки, переменные среды, расположение файлов, сетевые порты и другие необходимые компоненты и действия контейнера после его запуска.

Важно! Не используйте корень файловой системы хостовой ОС (/) в качестве пути для создания образа Docker — это приведет к передаче демону всего содержимого жесткого диска вашей машины. ■

Инструкции выполняются последовательно по принципу «одна строка — одна команда». Результаты наиболее важных инструкций фиксируются в виде отдельных слоев образа.

Примечание. Каждая новая инструкция выполняется независимо от других — это означает, что выполнение RUN cd /tmp не будет иметь никакого эффекта для последующих инструкций.

Слои в образе создают только инструкции FROM, RUN, COPY, и ADD.
Формат файла имеет следующий вид:

```
<ИНСТРУКЦИЯ> <аргументы>
```

Инструкции не чувствительны к регистру, однако используется единый стиль их оформления с использованием заглавных букв.

Список инструкций, используемых в Dockerfile:

- FROM — определяет базовый (родительский) образ;
- LABEL — описывает метаданные, например, сведения о том, кто создал и поддерживает образ;
- ENV — устанавливает постоянные переменные среды;
- RUN — выполняет команду и создаёт слой образа, используется для установки в контейнер пакетов;
- COPY — копирует в контейнер файлы и папки;
- ADD — копирует файлы и папки в контейнер, может распаковывать локальные .tar-файлы;
- CMD — описывает команду с аргументами, которую нужно выполнить, когда контейнер будет запущен. Аргументы могут быть переопределены при запуске контейнера. В файле может присутствовать только одна инструкция CMD;
- WORKDIR — задаёт рабочую директорию для следующей инструкции;
- ARG — задаёт переменные для передачи Docker во время сборки образа;
- ENTRYPOINT — предоставляет команду с аргументами для вызова во время выполнения контейнера. Аргументы не переопределяются;
- EXPOSE — указывает на необходимость открыть порт;
- VOLUME — создаёт точку монтирования для работы с постоянным хранилищем.

С подробной информацией об использовании и применении Dockerfile можно ознакомиться в справочной странице:

```
man dockerfile
```

Инструкция FROM

Dockerfile должен начинаться с инструкции FROM.

Инструкция FROM задает базовый (родительский) образ, на основе которого будет создаваться новый образ. Перед инструкцией FROM могут быть указаны только директивы синтаксического анализатора (parser directives), комментарии или глобальные аргументы (ARG).

Строки, начинающиеся с символа # воспринимаются как комментарии и не обрабатываются.

Если вы хотите создать образ, основанный на Ubuntu, вы можете использовать следующую инструкцию FROM:

```
FROM ubuntu:latest
```

Эта инструкция указывает Docker, что будет использована последняя версия образа Ubuntu в качестве базового для создания нового образа.

Инструкция LABEL

Инструкция LABEL (метка) позволяет добавлять в образ метаданные. Она может включать в себя контактные сведения создателя образа, версию приложения, описание образа и т.д. Объявление меток не замедляет процесс сборки образа и не увеличивает его размер. Они содержат в себе полезную информацию об образе Docker, поэтому их рекомендуется включать в файл.

Пример использования инструкции LABEL:

```
LABEL maintainer="Ivan Ivanov <iva_iva@example.com>"
LABEL version="1.0"
LABEL description="This is a sample Docker image"
```

Метаданные могут быть полезны при управлении образами и контейнерами в больших проектах.

Инструкция ENV

Инструкция ENV позволяет задавать постоянные переменные среды, которые будут доступны в контейнере во время его выполнения.

Инструкция ENV подходит для задания констант. Если вы используете некоторое значение в Dockerfile несколько раз, например, при описании команд, выполняющихся в контейнере, и в будущем оно может измениться, имеет смысл записать данное значение в подобную константу.

Пример использования инструкции ENV:

```
ENV ADMIN="ivan"
```

Инструкция RUN

Инструкция RUN позволяет выполнять команды внутри контейнера во время создания образа. Пример использования инструкции RUN:

```
RUN dnf update && dnf install -y
    package1
    package2
    package3
```

Данная инструкция устанавливает пакеты package1, package2 и package3 внутри контейнера.

Инструкция RUN может использоваться для установки зависимостей, настройки окружения, запуска скриптов и т.д. Она также может использоваться для создания новых слоев в образе Docker.

Важно помнить, что каждая инструкция RUN создает новый слой в образе Docker, поэтому необходимо объединять несколько команд в одну строку с помощью && или использовать инструкцию SHELL для выполнения нескольких команд в одном слое.

Инструкция COPY

Инструкция COPY сообщает Docker о необходимости переноса файлов и папок из локального контекста сборки в текущую рабочую директорию образа. Если целевая директория не существует, инструкция её создаст.

Пример использования инструкции COPY:

```
COPY . ./app
```

Инструкция ADD

Инструкция ADD копирует файлы или директории из исходной директории на хостовой машине в контейнер. Пример использования инструкции ADD:

```
ADD app.py /app/
```

Эта инструкция копирует файл app.py из текущей директории на хостовой машине в директорию /app/ внутри контейнера. Инструкция ADD может использоваться для добавления файлов, архивов, конфигурационных файлов и в контейнер. Она также может использоваться для скачивания файлов из сети Интернет.

Важно помнить, что инструкция ADD создает новый слой в образе Docker, поэтому необходимо использовать ее с осторожностью и объединять несколько команд в один слой, если это возможно. Кроме того, не рекомендуется использовать инструкцию ADD для копирования больших файлов, так как это может привести к увеличению размера образа Docker.

Инструкция CMD

Инструкция CMD передает Docker команду, которую необходимо выполнить при запуске контейнера.

Пример использования инструкции CMD:

```
CMD ["python "./my_script.py"]
```

Инструкция CMD может использоваться для определения стандартного

поведения контейнера при запуске. Это может быть полезно для установки параметров по умолчанию, например, для запуска веб-сервера или приложения.

Стоит отметить, что инструкция CMD может быть переопределена при запуске контейнера с помощью флага docker run. Например, если необходимо выполнить другой скрипт при запуске контейнера, можно использовать следующую команду:

```
docker run my-image python another_app.py
```

Данная команда переопределит команду, определенную в инструкции CMD, и запустит скрипт another_app.py вместо app.py. Ограничения использования инструкции CMD:

- в одном файле Dockerfile может присутствовать только одна инструкция CMD. Если в файле есть несколько таких инструкций, система проигнорирует все кроме последней;
- инструкция CMD может иметь exec-форму. Если в инструкцию не входит упоминание исполняемого файла, тогда в файле должна присутствовать инструкция ENTRYPOINT. В таком случае обе инструкции должны быть представлены в формате JSON.
- аргументы командной строки, передаваемые docker run, переопределяют аргументы, предоставленные инструкции CMD в Dockerfile.

Инструкция WORKDIR

Инструкция WORKDIR позволяет изменить рабочую директорию контейнера. С этой директорией работают инструкции COPY, ADD, RUN, CMD и ENTRYPOINT, указанные за WORKDIR.

Пример использования инструкции WORKDIR:

```
WORKDIR /usr/src/my_app_directory
```

Ограничения использования инструкции WORKDIR:

- с помощью WORKDIR рекомендуется устанавливать абсолютные пути к папкам, не перемещаясь по файловой системе с помощью команды cd в Dockerfile;
- инструкция WORKDIR автоматически создаёт директорию в том случае, если она не существует;
- можно использовать несколько инструкций WORKDIR. Если таким инструкциям предоставляются относительные пути, то каждая из них меняет текущую рабочую директорию.

Инструкция ARG

Инструкция ARG определяет переменную, которую можно использовать во время сборки образа. Пример использования инструкции ARG:

```
ARG VERSION=latest
```

Данная инструкция определяет переменную VERSION со значением "latest". Указанную переменную можно использовать в других командах Dockerfile, на-

пример:

```
FROM ubuntu:$VERSION
```

Данная команда использует значение переменной VERSION для выбора версии Ubuntu для указания в качестве базового образа.

Инструкция ARG может использоваться для передачи аргументов во время сборки образа, например, версии приложения или настроек конфигурации. Это может быть полезно при автоматизации процесса сборки образов Docker.

Важно помнить, что переменные, определенные с помощью инструкции ARG, доступны только во время сборки образа и не сохраняются в контейнере. Кроме того, значения переменных могут быть переопределены при запуске сборки образа с помощью флага --build-arg.

Инструкция ENTRYPOINT

Инструкция ENTRYPOINT определяет исполняемый файл или команду, которые будут выполняться при запуске контейнера. Инструкция также может использоваться для определения стандартного поведения контейнера при запуске. Пример использования инструкции ENTRYPOINT:

```
ENTRYPOINT ["python "app.py"]
```

Данная инструкция определяет исполняемый файл, который будет выполнен при запуске контейнера — запуск скрипта app.py на языке Python.

В отличие от инструкции CMD, инструкция ENTRYPOINT не может быть переопределена при запуске контейнера с помощью команды docker run. Однако, если необходимо добавить аргументы или параметры при запуске контейнера, можно использовать инструкцию CMD совместно с инструкцией ENTRYPOINT.

Пример совместного использования инструкций ENTRYPOINT и CMD:

```
ENTRYPOINT ["python "app.py"]
CMD [-port "8080"]
```

Эти инструкции определяют исполняемый файл и параметры, которые будут переданы ему при запуске контейнера. В данном случае, при запуске контейнера будет запущен скрипт app.py на языке Python с параметром --port, равным 8080. Если необходимо изменить значение параметра --port, можно переопределить его при запуске контейнера с помощью флага docker run:

```
docker run my-image --port 9000
```

Данная команда переопределит значение параметра --port и запустит контейнер с параметром --port, равным 9000.

Инструкция EXPOSE

Инструкция EXPOSE указывает, какие порты планируется открыть для того, чтобы через них можно было бы связаться с работающим контейнером. Данная инструкция не открывает порты. Она является средством общения между тем,

кто собирает образ, и тем, кто запускает контейнер.

Для того чтобы открыть порт (или порты) и настроить перенаправление портов, необходимо выполнить команду docker run с ключом -p. Если использовать ключ -P, открыты будут все порты, указанные в инструкции EXPOSE.

Пример использования инструкции EXPOSE:

```
EXPOSE 8000
```

Инструкция VOLUME

Инструкция VOLUME позволяет указать каталог, который контейнер будет использовать для постоянного хранения файлов и для работы с такими файлами.

Пример использования инструкции VOLUME:

```
VOLUME /my_volume
```

11.4.2 Docker Registry

В централизованном хранилище образов хранятся доступные образы Docker.

Общедоступным хранилищем образов является Docker Hub, по умолчанию Docker настроен на поиск образов в нем. Также есть возможность настройки своего собственного хранилища образов.

При использовании команд docker pull или docker run требуемые образы будут извлекаться из настроенного хранилища образов. Когда используется команда docker push, образ помещается в настроенное хранилище образов.

11.4.3 Docker Daemon

Демон Docker (dockerd) прослушивает запросы Docker API и управляет объектами Docker, такими как образы, контейнеры, сети и тома. Демон также может взаимодействовать с другими демонами для управления службами Docker.

Существует два способа настройки демона Docker:

- использование файла конфигурации JSON, является предпочтительным вариантом, так как он хранит все конфигурации в одном месте;
- использование флагов при запуске dockerd.

Чтобы настроить демон Docker с помощью файла JSON, необходимо создать файл /etc/docker/daemon.json.

Пример файла конфигурации JSON:

```
{
  "experimental": true,
  "log-level": "warn",
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "10m",
    "max-file": "5"
  }
}
```

В данной конфигурации описывается настройка журнала. Значение «warn» параметра log-level сообщает, что в журнале будут отображаться записи со статусом *Warning*. Журнал будет вестись в формате json-файла с максимальным размером файла *10Мб*, количество журналов не должно превышать *5* (при достижении этого значения журналы будут перезаписываться).

Синтаксис ручного запуска демона:

```
dockerd <опции>
```

Пример запуска демона вручную:

```
dockerd --debug
--tls=true
--tlscert=/var/docker/server.pem
--tlskey=/var/docker/serverkey.pem
--host tcp://192.168.59.3:2376
```

В данной конфигурации демон Docker работает в режиме отладки, использует TLS и прослушивает трафик, направляемый на 192.168.59.3 через порт 2376.

Демон Docker сохраняет все данные в одном каталоге, что помогает отслеживать все, что связано с Docker, включая контейнеры, образы, тома. Каталогом по умолчанию является /var/lib/docker.

Часто используемые параметры:

- --config-file="/etc/docker/daemon.json"— путь к файлу конфигурации JSON;
- -D, --debug=true | false — режим отладки, по умолчанию значение false;
- -H, --host=tcp://[host:port] | unix://[/path/to/socket] — привязка одного или нескольких сокетов;
- --iptables=true | false — добавление правил iptables, значение по умолчанию true;
- --isolation="default"— тип изоляции контейнеров, значение по умолчанию default.

С подробной информацией о доступных параметрах можно ознакомиться, выполнив:

```
dockerd --help
```

11.4.4 Docker Client

Клиент Docker (docker) — это основной способ взаимодействия многих пользователей с Docker. Когда выполняются такие задачи, как запуск контейнера, клиент отправляет команды демону dockerd, который их выполняет. Команда docker при работе обращается к Docker API. Клиент Docker может взаимодействовать с несколькими демонами.

Синтаксис использования клиента:

```
docker [<опции>] <команда> [<аргументы...>]
```

Основные команды управления описаны в разделе «Управление контейнера-

ми». Подробнее о работе с клиентом Docker см. в справке:

```
docker --help
```

11.4.5 Объекты Docker

Docker Image

Docker Image — неизменяемые файлы (образы), из которых можно развернуть контейнер. Образы могут быть основаны на каких-либо других образах (шаблонах) с добавлением некоторых дополнительных настроек.

Перед созданием контейнера Docker имеет возможность верифицировать образ. Если образ не изменен — контейнер запускается (создается). По умолчанию данная функция отключена, для включения верификации образов необходимо в файл **daemon.json** добавить следующую строку:

```
"disable-validation": false
```

Container

Container — исполняемый экземпляр образа.

Можно создавать, запускать, останавливать, перемещать или удалять контейнеры с помощью Docker API или командной строки. Есть возможность подключить контейнер к одной или нескольким сетям, подключить к нему хранилище или создать новый образ на основе его текущего состояния.

По умолчанию контейнер хорошо изолирован от других контейнеров и своей хост-машины. Можно контролировать, насколько изолированы сеть контейнера, хранилище или другие базовые подсистемы от других контейнеров или от хост-системы.

Контейнер определяется своим образом, а также любыми параметрами конфигурации, которые предоставляются ему при создании или запуске. При удалении контейнера любые изменения его состояния, не сохраненные в постоянном хранилище, уничтожаются.

11.4.6 Docker volume

Docker volume — механизм, позволяющий сохранять данные вне контейнера Docker. Он представляет собой отдельный файл или директорию на хост-системе, которые монтируются внутрь контейнера. Это позволяет сохранять данные между запусками контейнеров и обмениваться данными между контейнерами. Docker volumes также обеспечивают удобный способ резервного копирования данных и обновления приложений без потери данных.

Синтаксис команлы имеет следующий вид:

```
docker volume <команда>
```

Docker volume позволяет управлять томами с помощью следующих команд:

- create — создание тома;
- inspect — отображение информации о томе;
- ls — отображение списка томов;
- prune — удаление неиспользуемых томов;

- rm — удаление одного или нескольких томов;
- update — обновление тома (только для кластерных томов).

Docker volume create

Команда создает новый том, который контейнеры могут использовать и хранить в нем данные. Если имя не указано, Docker генерирует случайное имя.

Синтаксис имеет следующий вид:

```
docker volume create [<опции>] [<том>]
```

Опции команды:

- --driver, -d — имя драйвера тома, значение по умолчанию local;
- --opt, -o — специальные параметры драйвера;
- --label <метка> — указать метку для тома.

Подробную информацию об использовании команды см. в справке:

```
docker volume create --help
```

Примеры использования:

1. Создание нового тома:

```
docker volume create hello
```

2. Создание тома tmpfs с именем foo размером 100 мегабайт и uid 1000:

```
docker volume create --driver local
    --opt type=tmpfs
    --opt device=tmpfs
    --opt o=size=100m,uid=1000
    foo
```

3. Создание тома bfrfs с именем foo и монтированием раздела /dev/sda2:

```
docker volume create --driver local
    --opt type=btrfs
    --opt device=/dev/sda2
    foo
```

Docker volume inspect

Команда отображает информацию о томе. По умолчанию эта команда отображает все результаты в формате JSON.

Синтаксис команды:

```
docker volume inspect [<опции>] <том> [<том>...]
```

Доступная опция:

- --format, -f — преобразование формата вывода с использованием пользовательского шаблона json.

Подробную информацию об использовании команды см. в справке:

```
docker volume inspect --help
```

Примеры использования:

```
docker volume inspect new_volume
```

```
[
  {
    "CreatedAt": "2023-03-15T13:45:16Z
    "Driver": "local
    "Labels": {},
    "Mountpoint": "/var/lib/docker/volumes/8140a838303144125b4f-
54653b47ede0486282c623c3551fbc7f390cdc3e9cf5/_data
    "Name": "new_volume
    "Options": {},
    "Scope": "local"
  }
]
```

2. Форматирование выходных данных с использованием шаблона для печати свойства Mountpoint:

```
docker volume inspect --format ' .Mountpoint 'new_volume
```

```
/var/lib/docker/volumes/new_volume/_data
```

Docker volume ls

Команда отображает все тома, известные Docker. Результат может быть отфильтрован с помощью опции -f (--filter).

Синтаксис команды:

```
docker volume ls [<опции>]
```

Доступные опции:

- --filter, -f <фильтр> — укажите значение фильтра (например, driver=local);
- --format <строка> — преобразование формата вывода с использованием пользовательского шаблона table, вывод в формате таблицы с заголовками столбцов (по умолчанию);
- --quiet, -q — отображение только имен томов.

Подробную информацию об использовании команды см. в справке:

```
docker volume ls --help
```

Примеры использования:

1. Отображение всех доступных томов:

```
docker volume ls
```

```
DRIVER      VOLUME NAME
local       test_1
local       test_2
```

2. Использование фильтра для отображения доступных томов:

```
docker volume ls -f driver=local
```

```
DRIVER      VOLUME NAME
local       test_1
local       test_2
```

Docker volume prune

Команда удаляет все неиспользуемые локальные тома. Неиспользуемыми томами считаются тома, на которые не ссылаются никакие контейнеры.

Синтаксис команды:

```
docker volume prune [<опции>]
```

Доступные опции:

- --filter <фильтр> — укажите значение фильтра, например, label=<label>.
- --force, -f — не запрашивать подтверждение.

Пример использования:

```
docker volume prune
```

```
WARNING! This will remove all local volumes not used by at least
one container.
Are you sure you want to continue? [y/N] y
Deleted Volumes:
07c7bdf3e34ab76d921894c2b834f073721fccfbbcba792aa7648e3a7a664c2e
my-named-vol

Total reclaimed space: 36 B
```

Docker volume rm

Команда позволяет удалить один или несколько томов. Однако не может быть удален том , используемый контейнером.

Синтаксис команды:

```
docker volume rm [<опции>] <том> [<том_2>...]
```

Доступная опция:
- --force, -f — принудительное удаление одного или нескольких томов.

Пример использования:

```
docker volume rm hello
```

11.5 Управление контейнерами

11.5.1 Docker Container

Docker Container используется для управления контейнерами.
Синтаксис команды:

```
docker container <команда>
```

Используемые команды управления контейнерами:
- docker container attach — прикрепить локальные стандартные потоки ввода, вывода и вывода ошибок к запущенному контейнеру;
- docker container commit — создание нового образа из изменений контейнера;
- docker container cp — копирование файлов/папок между контейнером и хостовой файловой системой;
- docker container create — создание нового контейнера;
- docker container diff — проверка изменений файлов или каталогов в файловой системе контейнера;
- docker container exec — выполнение команды в запущенном контейнере;
- docker container export — экспорт файловой системы контейнера в виде tar-архива;
- docker container inspect — вывод подробной информации об одном или нескольких контейнерах;
- docker container kill — принудительное завершение одного или нескольких запущенных контейнеров;
- docker container logs — просмотр журналов регистрации контейнера;
- docker container ls — вывод списка контейнеров;
- docker container pause — приостановка всех процессов в одном или нескольких контейнерах;
- docker container port — список портов контейнера;
- docker container prune — удаление всех остановленных контейнеров;
- docker container rename — изменение имени контейнера;
- docker container restart — перезапуск одного или нескольких контейнеров;
- docker container rm — удаление одного или нескольких контейнеров;
- docker container run — создание и запуск нового контейнера из образа;
- docker container start — запуск одного или нескольких остановленных контейнеров;
- docker container stat — отображение статистики использования ресурсов контейнера(ов) в реальном времени;

- docker container stop — остановка одного или нескольких запущенных контейнеров;
- docker container top — отображение запущенных процессов контейнера;
- docker container unpause — отключение всех процессов в одном или нескольких контейнерах;
- docker container update — обновление конфигурации одного или нескольких контейнеров;
- docker container wait — блокировка остановки одного или нескольких контейнеров, вывод их кодов выхода.

При использовании команд допускается сокращение записи до вида:

```
docker <команда>
```

Далее будут рассмотрены основные команды управления контейнерами.

С подробной информацией обо всех командах и их работе можно ознакомиться, выполнив:

```
docker container <команда> --help
```

11.5.2 Docker build

Команда docker build используется для создания образа на основе файла Dockerfile и контекста.

Контекст — это набор файлов, находящихся по пути, определенному с помощью переменной PATH или URL. PATH — это директория в локальной системе, а URL — это удаленный репозиторий. Контекст сборки обрабатывается рекурсивно, поэтому PATH включает как директорию, так и все ее поддиректории, а URL — как репозиторий, так и все его субмодули.

Синтаксис команды:

```
docker build [<опции>] PATH | URL | -
```

Опции команды:

- -f, --file <string> — имя Dockerfile;
- --isolation <string> — технология изоляции контейнеров, по умолчанию default;
- --label <list> — установка метаданных для образа;
- -m, --memory <bytes> — ограничение памяти;
- --network <string> Установить режим работы с сетью для инструкций RUN во время сборки (по умолчанию "default")
- --rm — удалять промежуточные контейнеры после успешной сборки (по умолчанию true).

11.5.3 Docker image

Команда docker image предназначена для управления образами.

Синтаксис команды:

```
docker image <команда>
```

Список доступных команд:
- docker image build — создание образа из Docker-файла;
- docker image history — вывод истории образа;
- docker image import — импорт содержимого из tarball для создания образа файловой системы
- docker image inspect — вывод подробной информации об одном или нескольких образах;
- docker image load — загрузка образа из tar-архива или STDIN;
- docker image ls — вывод списка образов;
- docker image prune — удаление неиспользуемых образов;
- docker image pull — загрузка образа из registry;
- docker image push — загрузка образа в registry;
- docker image rm — удалить один или несколько образов;
- docker image save — сохранить один или несколько образов в tar-архив (по умолчанию передается в STDOUT);
- docker image tag — создать тег TARGET_IMAGE, который ссылается на SOURCE_IMAGE.

11.5.4 Docker images

Команда docker images используется для вывода списка доступных образов. Синтаксис команды:

```
docker images [<опции>] [<репозиторий>[:<тег>]]
```

Используемые опции команды:
- --all, -a — показать все образы (по умолчанию промежуточные образы скрыты);
- --digests — показать дайджест;
- --filter, -f — фильтрация выходных данных на основе предоставленных условий;
- --format — форматирование вывода с использованием пользовательского шаблона: 'table': вывод в формате таблицы с заголовками столбцов (по умолчанию) 'table TEMPLATE';
- --no-trunc — не обрезать вывод;
- --quiet, -q — показывать только идентификаторы образов.

Примеры использования команды:
1. Вывод всех образов:

```
docker images
```

```
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
busybox       latest    7cfbbec8963d   7 days ago      4.86MB
```

```
ubuntu        latest    08d22c0ceb15   2 weeks ago     7.8MB
hello-world   latest    feb5d9fea6a5   18 months ago   13.3kB
```

2. Вывод образов из репозитория ubuntu:

```
docker images ubuntu
```

```
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
ubuntu       latest    08d22c0ceb15   2 weeks ago   77.8MB
```

3. Вывод образов, соответствующих определенному тегу:

```
docker images hello-world:latest
```

```
REPOSITORY    TAG       IMAGE ID       CREATED          SIZE
hello-world   latest    feb5d9fea6a5   18 months ago   13.3kB
```

11.5.5 Docker create

Команда docker container create (или сокращенно: docker create) создает новый контейнер из указанного образа, не запуская его.

При создании контейнера демон docker создает записываемый слой контейнера поверх указанного образа и подготавливает его к выполнению указанной команды. Идентификатор контейнера затем выводится в STDOUT. Действия похожи на выполнение команды docker run -d, за исключением того, что контейнер никогда не запускается. Затем можно использовать команду docker container start (или сокращенно: docker start) для запуска контейнера в любой момент.

Синтаксис команды:

```
docker create [<опции>] <образ> [<команда>] [<аргументы>...]
```

Основные опции команды:

- --attach (-a) — позволяет манипулировать вводом и выводом по мере необходимости, для вывода информации используется стандартный ввод/вывод контейнера STDIN, STDOUT и STDERR.
- --cpus — определяет количество процессоров;
- --device — добавить хост-устройство в контейнер;
- --hostname, -h — имя хоста контейнера;
- --interactive, -i — оставить STDIN открытым без присоединения к терминалу;
- --memory, -m — лимит памяти. Данный параметр устанавливает верхний предел памяти, доступной для контейнера. Предел задается в контрольной группе, и приложения в контейнере могут запрашивать ее по адресу /sys/fs/cgroup/memory/memory.limit_in_bytes;
- --rm — автоматическое удаление контейнер после его выхода;
- --tty, -t — выделить псевдотерминал;
- --user, -u — имя пользователя или UID (формат: <name|uid>[:<group|gid>]);
- --volume, -v — примонтировать том;

- --volumes-from — смонтировать тома из указанного(ых) контейнера(ов);
- --workdir, -w — задать рабочий каталог внутри контейнера.

Примеры использования команды:

1. Создание интерактивного контейнера с прикрепленным псевдотерминалом, запуск контейнера и последующее подключение к нему:

```
docker container create -i -t --name new_container ubuntu
```

```
6d8af538ec541dd581ebc2a24153a28329acb5268abe5ef868c1f1a261221752
```

```
docker container start --attach -i mycontainer
/ # echo hello world
```

```
hello world
```

Вышеуказанные команды эквивалентны реализации через docker run:

```
docker run -it --name mycontainer2 ubuntu
/ # echo hello world
```

```
hello world
```

2. Тома контейнеров инициализируются на этапе создания docker (т.е. при запуске docker). Создадим том data в контейнере, а затем попробуем использовать его из другого контейнера:

```
docker create -v /data --name data ubuntu
```

```
240633dfbb98128fa77473d3d9018f6123b99c454b3251427ae190a7d951ad57
```

```
docker run --rm --volumes-from data ubuntu ls -la /data
```

```
total 8
drwxr-xr-x 2 root root 4096 Mar 5 04:10 .
drwxr-xr-x 48 root root 4096 Mar 5 04:11 ..
```

11.5.6 Docker run

Команда docker run используется для создания и запуска контейнера из образа. Может принимать несколько аргументов, таких как имя образа, параметры запуска контейнера и команду, которую необходимо выполнить внутри контейнера.

Синтаксис команды:

```
docker run [<опции>] <образ> [<команды>] [<аргументы>...]
```

Основные опции команды:

- --attach (-a) — позволяет манипулировать вводом и выводом по мере необходимости, для вывода информации используется стандартный ввод/вывод контейнера STDIN, STDOUT и STDERR.
- --cpus — определяет количество процессоров;
- --detach, -d — запустить контейнер в фоновом режиме и напечатать ID контейнера.
- --device — добавить хост-устройство в контейнер.
- --hostname, -h — имя хоста контейнера;
- --interactive, -i — оставить STDIN открытым без присоединения к терминалу;
- --memory, -m — лимит памяти. Данный параметр устанавливает верхний предел памяти, доступной для контейнера. Предел задается в контрольной группе, и приложения в контейнере могут запрашивать ее по адресу /sys/fs/cgroup/memory/memory.limit_in_bytes;
- --privileged — предоставление расширенных прав контейнеру;
- --rm — автоматическое удаление контейнер после его выхода;
- --tty, -t — выделить псевдотерминал;
- --user, -u — имя пользователя или UID (формат: <name|uid>[:<group|gid>]);
- --volume, -v — примонтировать том;
- --workdir, -w — задать рабочий каталог внутри контейнера.

Примеры использования параметров:

1. Передача данных на стандартный ввод контейнера:

```
echo "test"| docker run -i -a stdin ubuntu cat -
```

2. Запуск контейнера в фоновом режиме и вывод ID контейнера. Например:

```
docker run -d ubuntu
```

```
b85f949c5d1e593fcf732db410502e999339549d1175947a0467652e7053c133
```

3. Добавление в непривилегированный контейнер (без использования флага --privileged) блочного устройства хранения данных, loop-устройства и аудиоустройства и предоставление приложению прямого доступа к ним.

```
docker run -it --rm
--device=/dev/sdc:/dev/xvdc
--device=/dev/sdd
--device=/dev/zero:/dev/foobar
ubuntu ls -l /dev/xvdc,sdd,foobar
```

```
brw-rw---- 1 root disk 8, 2 Mar 9 16:05 /dev/xvdc
brw-rw---- 1 root disk 8, 3 Mar 9 16:05 /dev/sdd
crw-rw-rw- 1 root root 1, 5 Mar 9 16:05 /dev/foobar
```

4. Выполнение команды внутри указанного каталога (если путь не существует, он будет создан внутри контейнера):

```
docker run -w /path/to/dir/ -i -t ubuntu pwd
```

11.5.7 Docker start

Команда docker start предназначена для запуска контейнера.

Синтаксис команды:

```
docker start [<опции>] <контейнер> [<контейнер_2>...]
```

Используемые опции команды:

- --attach, -a — прикрепить STDOUT/STDERR и переадресовать сигналы;
- --detach-keys — переопределение последовательности клавиш для отсоединения контейнера;
- --interactive, -i — прикрепить STDIN контейнера.

Пример использования команды:

```
docker start new_container
```

11.5.8 Docker stop

Команда docker stop предназначена для остановки работы контейнера.

Основной процесс внутри контейнера получит SIGTERM, а по истечении времени ожидания — SIGKILL.

Синтаксис команды:

```
docker stop [<опции>] <контейнер> [<контейнер_2>...]
```

Используемые опции команды:

- --time , -t — время ожидания (в секундах) перед принудительной остановкой контейнера.

Пример использования команды:

```
docker stop new_container
```

11.5.9 Docker pause

Команда docker pause приостанавливает все процессы в указанных контейнерах.

Синтаксис команды:

```
docker pause <контейнер> [<контейнер_2>...]
```

Пример использования команды:

```
docker pause new_container
```

11.5.10 Docker login

Для доступа в некоторые централизованные хранилища образов необходимо иметь учетную запись. После регистрации войти в систему можно, используя команду docker login.

Синтаксис команды:

```
docker login [<опции>] [<сервер>]
```

Для входа в систему необходимо указать имя пользователя и пароль своей учетной записи. Для доступа к собственным хранилищам образов необходимо указать имя сервера, если сервер не указан, он определяется демоном docker по умолчанию. Основные параметры команды:

- --password, -p <пароль> — пароль;
- --password-stdin — получить пароль со стандартного ввода;
- --username, -u <имя> — имя пользователя.

Примеры использования команды:

1. Вход в собственное хранилище образов:

```
docker login localhost:8080
```

2. Вход в хранилище образов с использованием стандартного ввода (stdin):

```
cat ./pass.txt | docker login -username user -password-stdin
```

где:

- ./pass.txt — путь к файлу, в котором хранится пароль учетной записи;
- user — имя пользователя для входа в хранилище образов.

11.5.11 Docker attach

Команда docker attach позволяет прикрепить стандартный ввод (STDIN), вывод (STDOUT) и вывод ошибок (STDERR) локального терминала к контейнеру, используя идентификатор или имя контейнера. Это позволит просматривать его текущий вывод или управлять им в интерактивном режиме, как если бы команды выполнялись непосредственно в локальном терминале.

Для остановки контейнера используется сочетание клавиш «CTRL+C», которое передаст контейнеру сигнал SIGKILL. Для остановки контейнера может быть использована команда exit, которая корректно завершает его работу – это наиболее предпочтительный вариант остановки контейнера.

Если контейнер был запущен с параметрами -i и -t, можно отсоединиться от контейнера и оставить его запущенным, используя последовательность клавиш «CTRL+P» и «CTRL+Q».

Синтаксис команды:

```
docker attach [<опции>] <контейнер>
```

Основные параметры команды:
- --detach-keys — переопределить последовательность клавиш для отсоединения контейнера;
- --no-stdin — не присоединять STDIN;
- --sig-proxy — передать все полученные сигналы процессу. Значение по умолчанию — true.

11.5.12 Docker exec

Команда docker exec запускает новую команду в работающем контейнере.

Команда, запущенная с помощью docker exec, выполняется только во время работы основного процесса контейнера (PID 1), и она не перезапускается, если перезапускается контейнер.

Команда запускается в директории контейнера по умолчанию. Если базовый образ имеет пользовательский каталог, указанный директивой WORKDIR в Dockerfile, тогда используется данный каталог.

Команда должна быть исполняемым файлом. Команда, заключенная в кавычки, не сработает.

Например, docker exec -it my_container sh -c "echo a && echo b" будет выполнена, а docker exec -it my_container "echo a && echo b" — нет.

Синтаксис команды:

```
docker exec [<опции>] <контейнер> <команда> [<аргументы>...]
```

Опции команды:
- -d, --detach — режим отсоединения: выполнение команды в фоновом режиме;
- --detach-keys <string> — переопределение последовательности клавиш для отсоединения контейнера;
- -e, --env <list> — установка переменных окружения;
- --env-file <list> — чтение файла с переменными окружения;
- -i, --interactive — оставить STDIN открытым, даже если он не подключен;
- --privileged — предоставить расширенные привилегии команде;
- -t, --tty — выделить псевдотерминал;
- -u, --user <string> — имя пользователя или UID (формат: <name|uid> [:<group|gid>]);
- -w, --workdir <string> — рабочая директория внутри контейнера.

Примеры использования команды:

1. Создание нового файла /tmp/execWorks внутри работающего контейнера mycontainer в фоновом режиме:

```
docker exec -d mycontainer touch /tmp/execWorks
```

2. Запуск нового сеанса оболочки в контейнере mycontainer:

```
docker exec -it mycontainer sh
```

11.5.13 Docker logs

Команда docker logs предназначена для вывода журналов контейнера.
Синтаксис команды:

```
docker logs [<опции>] <контейнер>
```

Используемые параметры команды:

- --details — показать дополнительные детали, предоставляемые в журнале;
- -f, --follow — следить за выводом журналов;
- --since <string> — показывать записи с указанной временной метки (например, 2022-03-01T13:23:37Z) или с определенного промежутка времени (например, 42m — отобразит записи журнала за последние 42 минуты);
- -n, --tail <string> — количество строк для отображения записей с конца журнала (по умолчанию "все");
- -t, --timestamps — показать временные метки;
- --until <string> — показать записи до указанной временной метки (например, 2022-03-01T13:23:37Z) или до определенного промежутка времени (например, 42m — отобразит записи, сделанные 42 минуты назад и ранее).

Примеры использования команды:

```
docker run --name test -d busybox sh -c "while true; do $(echo
date); sleep 1; done"
```

```
Fri Mar 24 08:16:28 UTC 2023
```

```
docker logs -f --until=2s test
```

```
Fri Mar 24 08:16:28 UTC 2023
Fri Mar 24 08:16:29 UTC 2023
Fri Mar 24 08:16:30 UTC 2023
Fri Mar 24 08:16:31 UTC 2023
```

11.5.14 Docker stats

Команда docker stats предназначена для отображения статистики использования ресурсов контейнеров в реальном времени. Для того чтобы ограничить данные одним или несколькими конкретными контейнерами, необходимо указать список имен или идентификаторов контейнеров, разделенных пробелом. Также можно указать остановленный контейнер, но остановленные контейнеры не возвращают никаких данных.

Синтаксис команды:

```
docker stats [<опции>] [<контейнер>...]
```

Используемые опции команды:

- --all, -a — показать все контейнеры (по умолчанию отображаются только запущенные);
- --format — форматирование вывода с использованием пользовательского шаблона: 'table': вывод в формате таблицы с заголовками столбцов (по умолчанию) 'table TEMPLATE';
- --no-stream — отключить потоковую статистику и получать только первый результат;
- --no-trunc — не обрезать вывод.

Примеры использования команды:
1. Вывод общей статистики использования ресурсов.

```
docker stats
```

```
CONTAINER ID   NAME         CPU %     MEM USAGE / LIMIT   MEM %
NET I/O       BLOCK I/O   PIDS
a94cd881360f   test_con3   0.17%     416KiB / 15.63GiB   0.00%
2.1kB / 0B    0B / 0B     1
4bc7571b9508   test_con2   0.15%     424KiB / 15.63GiB   0.00%
2.1kB / 0B    0B / 0B     1
9f426fb78c27   test_con1   0.15%     420KiB / 15.63GiB   0.00%
2.71kB / 0B   0B / 0B     1
```

2. Вывод статистики контейнера с именем test_con2 и получение вывода в json-формате.

```
docker stats test_con2 --no-stream --format "{{ json . }}"
```

```
"BlockIO":"0B / 0B", "CPUPerc":"0.20%","Container":"test_con2",
"ID":"4bc7571b9508", "MemPerc":"0.00%","MemUsage":"424KiB /
15.63GiB", "Name":"test_con2", "NetIO":"2.31kB / 0B", "PIDs":"1"
```

3. Вывод статистики для указанных имени и ID контейнеров.

```
docker stats 9f426fb78c27 test_con3
```

```
CONTAINER ID   NAME         CPU %     MEM USAGE / LIMIT   MEM %
NET I/O       BLOCK I/O   PIDS
9f426fb78c27   test_con1   0.00%     420KiB / 15.63GiB   0.00%
2.71kB / 0B   0B / 0B     1
a94cd881360f   test_con3   0.16%     416KiB / 15.63GiB   0.00%
2.31kB / 0B   0B / 0B     1
```

11.5.15 Docker tag

Команда docker tag предназначена для присвоения определенного имени и, по желанию, тега образу в локальном хранилище.

Синтаксис команды:

```
docker tag SOURCE_IMAGE[:<тег>] TARGET_IMAGE[:<тег>]
```

где SOURCE_IMAGE — это имя образа или его ID, которому требуется назначить новое имя и/или тег, а TARGET_IMAGE — новое имя образа, которое необходимо ему присвоить. Дополнительный параметр <тег> может использоваться для указания конкретной версии образа.

Если задано имя собственного хранилища образов, за ним может дополнительно следовать номер порта в формате :8080. Имя хранилища должно соответствовать стандартным правилам DNS, но не может содержать символы подчеркивания.

Компоненты имени могут содержать строчные буквы, цифры и разделители. Разделитель определяется как точка, один или два знака подчеркивания или один или несколько дефисов. Компонент имени не может начинаться или заканчиваться разделителем. Имя тега может содержать строчные и прописные буквы, цифры, символы подчеркивания, точки и дефисы. Имя тега не может начинаться с точки или дефиса и может содержать не более 128 символов.

Примеры использования команды:

1. Присвоение образу с именем myapp тега *v1.0* и последующее переименование в *myregistry/myapp:v1.0*:

```
docker tag myapp myregistry/myapp:v1.0
```

2. Присвоение тега *version1.0* локальному образу с идентификатором 0e557428-3393 и перемещение его в репозиторий fedora:

```
docker tag 0e5574283393 fedora/httpd:version1.0
```

11.6 Безопасность в среде контейнеризации

11.6.1 Изоляция контейнеров

Пространства имен (namespaces) в среде контейнеризации обеспечивают изоляцию запущенных процессов, ограничивая их доступ к системным ресурсам. Главная задача каждого пространства имен — отделить определенный глобальный ресурс хостовой системы в некоторую среду, которая заставит считать работающие процессы, что они выполняются в своем собственном экземпляре глобального ресурса.

В среде контейнеризации обеспечиваются следующие механизмы изоляции контейнеров:

- изоляция пространств идентификаторов процессов контейнеров (PID);
- изоляция пространств имен для межпроцессного взаимодействия контейнеров (IPC);
- изоляция пространств имен для пользователей и групп контейнеров (USER);
- изоляция пространств имен хостов и доменов контейнеров (UTS);
- изоляция сетевых пространств имен контейнеров (NETWORK);

- изоляция пространств имен для иерархии каталогов контейнеров (MOUNT).

Пространства имен идентификаторов процессов контейнеров (**PID namespaces**) изолируют пространство идентификаторов процессов, следовательно, процессы в разных пространствах могут иметь одинаковые PID. Одно из основных преимуществ PID namespaces заключается в том, что контейнеры можно перемещать между хостами, сохраняя при этом одинаковые идентификаторы процессов внутри контейнера.

PID namespaces также позволяют каждому контейнеру иметь свой собственный инициализирующий (родительский) процесс с PID 1, который управляет различными задачами инициализации системы. При завершении инициализирующего процесса ядро завершает все процессы, принадлежащие одному контейнеру, через сигнал SIGKILL.

Пространства имен для межпроцессного взаимодействия контейнеров (**IPC namespaces**) изолируют объекты IPC System V и очереди сообщений POSIX.

System V IPC включает в себя семафоры, разделяемую память и очереди сообщений. Каждый ресурс межпроцессного взаимодействия должен иметь свой уникальный идентификатор (ID). Это позволяет процессам, взаимодействующим между собой, обращаться к общему ресурсу. Объекты, созданные в этом пространстве имён, видны всем процессам, состоящим в этом пространстве, и не видны всем остальным процессам в других пространствах имен.

Пространства имён пользователей (**USER namespaces**) изолируют ID пользователей и групп, корневой каталог, ключи и права. Идентификаторы пользователя и группы процесса могут отличаться внутри и вне пространства имён пользователей. В частности, процесс может иметь непривилегированный ID пользователя вне пространства и, в то же время, иметь ID суперпользователя root внутри пространства.

Сетевые пространства имен (**NETWORK namespaces**) обеспечивают изоляцию системных сетевых ресурсов. Таким образом, каждое сетевое пространство имен имеет свои собственные сетевые устройства, IP-адреса, таблицы IP-маршрутизации, межсетевые экраны, номера портов и т. д.

Физические сетевые устройства могут принадлежать только одному пространству. При этом каждое пространство имён может иметь одно или несколько виртуальных устройств, для обеспечения доступа во внешнюю сеть между физическим и виртуальным устройством из разных пространств создается мост — туннель между разными пространствами имён сети. Когда последний процесс в пространстве завершается, физическое сетевое устройство возвращается не в родительское пространство, а в пространство хостовой машины.

Пространства имен хостов и доменов (**UTS namespaces**) изолируют два системных идентификатора — имя узла и имя домена. В контексте контейнеров функция пространств имен UTS позволяет каждому контейнеру иметь собственное имя хоста и доменное имя NIS.

Пространства имен для иерархии каталогов (**MOUNT namespaces**) изолируют набор точек монтирования файловой системы, видимых группой процессов. Таким образом, процессы в разных пространствах имен монтирования могут иметь разные представления иерархии файловой системы.

11.6.2 Проверка корректности конфигурации контейнеров

По умолчанию в контейнере недоступны устройства хостовой системы. Для подключения необходимых устройств на этапе запуска (создания или обновления конфигурации) контейнера требуется указать значение аргумента --device.

Для получения списка устройств контейнера необходимо запустить контейнер с подключенным устройством:

- для подключения жесткого диска:

```
   docker run -it --rm --name test --device=/dev/sda
registry.red-soft.ru/ubi7
```

- для подключения периферийного устройства, например CD-ROM:

```
   docker run -it --rm --name test --device=/dev/cdrom
registry.red-soft.ru/ubi7
```

Далее необходимо проверить список доступных контейнеру устройств:

```
   ls /dev
```

```
   console  fd    mqueue  ptmx  random  shm     stdin   tty     zero
   core     full  null    pts   sda     stderr  stdout  urandom
```

Для отображения информации о подключенных дисках в контейнере используются команды fdisk (для жестких дисков) и blkid (для блочных устройств). Например:

```
   fdisk -l
```

```
   Disk /dev/sda: 15.15 GiB, 16265093120 bytes, 31767760 sectors
   Disk model: VBOX HARDDISK
   Units: sectors of 1 * 512 = 512 bytes
   Sector size (logical/physical): 512 bytes / 512 bytes
   I/O size (minimum/optimal): 512 bytes / 512 bytes
   Disklabel type: dos
   Disk identifier: 0x2e69b634

   Device     Boot   Start       End   Sectors   Size Id Type
   /dev/sda1  *       2048   2099199   2097152     1G 83 Linux
   /dev/sda2       2099200  31766527  29667328  14.1G 8e Linux LVM
```

```
   blkid
```

```
   /dev/cdrom: BLOCK_SIZE="2048" UUID="2022-10-27-17-32-59-00"
LABEL="redos-MUROM-7.3.2 x86_64" TYPE="iso9660" PTUUID="0b10fcfb"
```

```
PTTYPE="dos"
```

Для получения списка доступных устройств на хостовой ОС выполните:

```
ls /dev
```

Для ограничения в контейнере количества операций ввода/вывода в секунду необходимо при запуске (создании или обновлении конфигурации) контейнера указать параметр --device-write-iops (или --device-read-iops). Например, установим ограничение 10 операций ввода/вывода в секунду:

```
docker run -it --rm --name test --device-write-iops /dev/sda:10
registry.red-soft.ru/ubi7
```

Для отслеживания статистики в контейнере выполните:

```
time dd if=/dev/zero of=test.out bs=1M count=1024 oflag=direct
```

```
1024+0 records in
1024+0 records out
1073741824 bytes (1.1 GB, 1.0 GiB) copied, 204.826 s, 5.2 MB/s

real    3m24.836s
user    0m0.025s
sys  0m0.400s
```

При запуске контейнера без ограничения количества операций ввода/вывода статистика выглядит следующим образом:

```
time dd if=/dev/zero of=test.out bs=1M count=1024 oflag=direct
```

```
1024+0 records in
1024+0 records out
1073741824 bytes (1.1 GB, 1.0 GiB) copied, 3.61733 s, 297 MB/s

real    0m3.623s
user    0m0.000s
sys  0m0.293s
```

Для монтирования файловой системы хостовой ОС необходимо использовать параметр назначения прав только для чтения — ro. Например, создайте контейнер и подключите к нему корневую директорию хостовой ОС командой:

```
docker run -it --rm --name test -v /:/mnt/:ro registry.red-soft.
ru/ubi7
```

Для вывода содержимого каталога контейнера /mnt выполните:

```
ls -l /mnt
```

```
total 68
lrwxrwxrwx. 1 root root 7 Mar 31 2021 bin -> usr/bin
dr-xr-xr-x. 6 root root 4096 Mar 13 08:10 boot
drwxr-xr-x. 20 root root 14180 Mar 16 12:25 dev
drwxr-xr-x. 100 root root 12288 Mar 16 12:25 etc
drwxr-xr-x. 6 root root 4096 Mar 16 07:13 home
```

При попытке создать какой-либо файл в данном каталоге будет выведено сообщение об ошибке вида:

```
touch /mnt/test_file
```

```
touch: cannot touch '/mnt/test_file': Read-only file system
```

11.6.3 Контроль целостности контейнеров и их образов

Контроль целостности образов контейнеров и их исполняемых файлов производится инструментом хостовой ОС — IMA, контроль целостности сведений о событиях безопасности осуществляется средством верификации целостности хостовой ОС — afick. Подробную информацию об инструментах см. в п. 6.9 «Afick — верификация целостности» и п. 6.16 «Контроль целостности запускаемых компонентов программного обеспечения» настоящего руководства.

По умолчанию из-за высокой нагрузки на ЭВМ функция контроля целостности выключена. Включение производится в файле настроек daemon.json установкой значения параметра **disable-validation** в **false**. Проверка целостности по расписанию (1 раз в неделю) по умолчанию также отключена по тем же причинам. Для включения проверки целостности образов контейнеров и файла конфигурации необходимо выполнить:

```
systemctl enable docker-validate.timer --now
```

Для изменения настроек расписания проверки целостности образов контейнеров и файла конфигурации средства контейнеризации необходимо изменить настройки таймера в файле /usr/lib/systemd/system/docker-validate.timer в секции [Timer].

По умолчанию файл имеет следующее содержимое:

```
[Unit]
Description=Checking the integrity of container images and configuration file once a week
ConditionVirtualization=!container
ConditionPathExists=/var/run/docker.sock
ConditionFileIsExecutable=/usr/bin/curl
```

```
[Timer]
OnCalendar=weekly
Persistent=true

[Install]
WantedBy=timers.target
```

Примечание. Не рекомендуется осуществлять проверку контроля целостности образов контейнеров и файла конфигурации средства контейнеризации чаще 1 раза в час. Индивидуальные настройки зависят от количества установленных образов и вычислительной мощности отдельно взятой ЭВМ.

После внесения каких-либо изменений в файл настройки таймера необходимо его перезапустить, чтобы внесенные изменения вступили в силу:

```
systemctl restart docker-validate.timer
```

Отключение проверки целостности образов по расписанию реализуется выполнением команды:

```
systemctl disable docker-validate.timer --now
```

При попытке создать (запустить) контейнер из образа, целостность которого нарушена, будет выведено сообщение об ошибке примерно следующего вида, а запуск (создание) будет отменено:

```
docker: Error response from daemon: The image is corupted!
See 'docker run --help'
```

В журнале хостовой ОС будет зафиксирована попытка запуска контейнера из поврежденного образа. Запись в журнале будет иметь примерно следующий вид:

```
journalctl -xe -u docker
```

```
...
мар 17 13:08:52 localhost.localdomain dockerd[10789]: time="2023
-03-17T13:08:52.600838411+03:00" level=error msg="The image is corup-
ted!" id=1c8b8da7-4f6c-4b1b-b56c-3b6a5e699135 image=70ea72b3cd35fd95
de1b279dfa3f5b50a49e191246ce880c935bc74af6535e8c type="Vadlidate
image" user=1001
...
```

Также средством контейнеризации осуществляется контроль целостности параметров настройки с помощью инструмента хостовой ОС — IMA.

В журнале безопасности хостовой ОС будут зарегистрированы соответствующие записи о нарушении целостности файла настроек средства контейнеризации:

```
journalctl -xe -u docker
```

```
...
map 17 14:54:51 localhost.localdomain dockerd[857833]: time="2023
-03-15T12:54:51.638012556+03:00" level=info msg="The settings file
has passed integrity control" id=642490c2-f3ab-4dcd-a933-3490ffba86-
dc result=OK type="Vadlidate config file" user=
...
```

11.6.4 Регистрация событий безопасности

Регистрация событий безопасности производится средствами хостовой ОС.

В журнале безопасности регистрируются такие виды событий, как:

- создание, модификация и удаление образов контейнеров;
- запуск и остановка контейнеров с указанием причины остановки;
- модификация запускаемых контейнеров;
- факты нарушения целостности объектов контроля.

Просмотр зарегистрированных действий осуществляется с помощью команды (с правами суперпользователя):

```
journalctl -xe -u docker
```

События по созданию, модификации и удалению образов отображаются в виде следующих записей:

```
map 20 10:33:18 localhost.localdomain dockerd[7669]: time="2023-
03-20T10:33:18.023189296+03:00" level=info msg="The image was crea-
ted." id=6da4d000-7261-441b-99eb-59098d519138 image=ad4f35aad12f1e3-
caefdd36b29e03b62e049053ae7b91f7994143dc57259c48e result=OK type=
"Build image" user=1001
...
map 20 10:35:02 localhost.localdomain dockerd[7669]: time="2023-
03-20T10:35:02.917209586+03:00" level=info msg="The image was commi-
ted." id=153d2f86-461d-49a2-bc9a-19c8ed94059c image=3d9cd2346b53888-
f0bcb77776b72f331f82e9fd0035963439e7249a834fd8502 parent=ad4f35aad1-
2f1e3caefdd36b29e03b62e049053ae7b91f7994143dc57259c48e result=OK
type="Commit image"user=1001
...
map 20 10:37:09 localhost.localdomain dockerd[7669]: time="2023-
03-20T10:37:09.488058315+03:00" level=warning msg="The image was de-
leted." id=25ffc1e4-d765-44a7-bd27-1ea1b758dc45 image=ad4f35aad12f1-
e3caefdd36b29e03b62e049053ae7b91f7994143dc57259c48e result=OK type=
"Delete image" user=1001
map 20 10:37:09 localhost.localdomain dockerd[7669]: time="2023-
03-20T10:37:09.682221342+03:00" level=warning msg="The image was de-
leted." id=3ed4b675-d8ad-4ac6-b246-991ec1eb3ad6 image=3d9cd2346b53-
888f0bcb77776b72f331f82e9fd0035963439e7249a834fd8502 result=OK type=
```

```
"Delete image" user=1001
   ...
```

Записи о запуске и остановке контейнеров выглядят следующим образом:

```
   ...
   мар 20 10:50:08 localhost.localdomain dockerd[7669]: time="2023-
03-20T10:50:08.980129228+03:00" level=info msg="The container has
been created." container=b630ae774b7495b52b2ad517dcbc0263ccc1f8bb68-
173fc77d5b85e7128e62cb id=c4f79a5e-8ac2-45ab-a459-22e65513731f
result=OK type="Create container" user=1001
   мар 20 10:50:12 localhost.localdomain dockerd[7669]: time="2023-
03-20T10:50:12.046570726+03:00"level=info msg="The container has
been started." container=b630ae774b7495b52b2ad517dcbc0263ccc1f8bb68-
173fc77d5b85e7128e62cb id=87536452-badf-43a0-8e84-afe9f10bbe63
result=OK type="Start container" user=1001
   ...
   мар 20 10:52:07 localhost.localdomain dockerd[7669]: time="2023-
03-20T10:52:07.419240037+03:00"level=info msg="The container has
been stopped." container=b630ae774b7495b52b2ad517dcbc0263ccc1f8bb68-
173fc77d5b85e7128e62cb id=8c13117b-7f36-4be9-815e-fed344adc92e rea-
son="Testing rea-
son" result=OK type="Stop container" user=1001
   ...
```

Записи о модификации запущенных контейнеров имеют следующий вид:

```
   ...
   мар 20 12:55:57 localhost.localdomain dockerd[7669]: time="2023-
03-20T12:55:57.938404067+03:00" level=info msg="The container has
been created." container=9cd7084fa3261c068353f26d265b4870912e45bb1-
ed5891fcf6c506543b96954 id=8b5369f6-be6d-4c51-b358-fbe528b15258
result=OK type="Create container" user=1001
   мар 20 12:56:01 localhost.localdomain dockerd[7669]: time="2023-
03-20T12:56:01.119789828+03:00" level=info msg="The container has
been started." container=9cd7084fa3261c068353f26d265b4870912e45bb1-
ed5891fcf6c506543b96954 id=98057125-22bc-4def-86ac-e860cdabd867
result=OK type="Start container" user=1001
   мар 20 12:57:17 localhost.localdomain dockerd[7669]: time="2023-
03-20T12:57:17.112772813+03:00" level=warning msg="The container has
been updated!" container=9cd7084fa3261c068353f26d265b4870912e45bb1-
ed5891fcf6c506543b96954 id=a34ea823-ac59-47b1-b031-aad18f95492d
result=OK type="Update container" user=1001
   ...
```

Журнал событий безопасности средства контейнеризации доступен только для чтения. Проверку установленных прав на журнал можно произвести командой:

```
ls -l /var/log/docker/docker.log
```

```
-rw-r--r--. 1 root root 3804 мар 20 16:49 /var/log/docker/docker.
log
```

Стандартная настройка ротации и архивации журнала производится при достижении размера 10Мб. Настройка параметров осуществляется только суперпользователем root в файле /etc/logrotate.d/docker.

Содержимое файла имеет следующий вид:

```
/var/log/docker/docker.log {
  missingok
  maxsize 10M
  compress
  rotate 10
  copytruncate
  delaycompress
  create 0600 root root
}
```

При достижении файлом журнала максимально установленного размера будет выполнено его архивирование. Копия журнала будет сохранена в каталоге /var/log/docker с указанием даты ее создания.

Пример файла:

```
ls -l /var/log/docker
```

```
-rw-r--r--. 1 root root     0 мар 20 17:38 docker.log
-rw-r--r--. 1 root root 13497 апр 3  14:15 docker.log-20230403
```