АО "НТЦ ИТ РОСА"

**Операционная система**

Инв. № подл.

Подпись и Дата

Подпись и Дата

Инв. № дубл.

Взам.инв.№.

**РОСА "ХРОМ" 12**

**Руководство администратора**

РСЮК.10501-01 32 01

Листов 301

2025

АННОТАЦИЯ

Данный документ является руководством администратора операционной системы РОСА "ХРОМ" версии 12 РСЮК.10501-01 (далее – ОС, Система) в вариантах "рабочая станция" (далее – ОС рабочей станции) и "сервер" (далее – серверная ОС).

ОС зарегистрирована в реестре российского ПО от 05.09.2016 №1607.

Предполагается, что читатель является системным администратором ОС Linux с опытом работы в интерфейсе командной строки. Также рекомендуется ознакомиться с принципами работы Системы в графическом интерфейсе пользователя, которые описаны в документе "ОС РОСА "ХРОМ". Руководство пользователя. РСЮК.10501-01 34 01".

В настоящем руководстве приведен состав ОС, инструкции по нескольким типам установки ОС (локальная, установка с использованием средств маскирования информации, установка в виртуальную машину, автоматизированная установка), общие сведения о работе в ОС, ее настройке и организации работы пользователей в ней.

Перед установкой, настройкой и эксплуатацией ОС рекомендуется внимательно ознакомиться с настоящим руководством.

Для разработки документа использованы ссылки на следующие стандарты:

* ГОСТ Р 2.105-2019 "Единая система конструкторской документации (ЕСКД). Общие требования к текстовым документам";
* ГОСТ 2.601 "Единая система программной документации. Виды программных документов";
* ГОСТ 19.101-77 "Единая система программной документации.  
  Виды программ и программных документов";
* ГОСТ 19.105-78 "Единая система программной документации. Общие требования к программным документам";
* [ГОСТ 19.503-79](https://docs.cntd.ru/document/1200007674) "Единая система программной документации. Руководство системного программиста".

Настоящий документ подготовлен в соответствии с технологической инструкцией "РОСА. Регламент формирования документации к программным продуктам" (шифр РСЮК.11001-02 90 01).

Содержание

[1 Общие сведения 11](#_Toc208508947)

[1.1 Общее описание 11](#_Toc208508948)

[1.2 Основные характеристики 11](#_Toc208508949)

[1.2.1 Аппаратная поддержка 11](#_Toc208508950)

[1.2.2 Программная платформа 11](#_Toc208508951)

[1.2.3 Программное обеспечение 11](#_Toc208508952)

[1.2.4 Сети и развертывание 12](#_Toc208508953)

[1.2.5 Защита информации 13](#_Toc208508954)

[1.2.6 Состав дистрибутива 13](#_Toc208508955)

[1.2.7 Доступные в репозитории компоненты 14](#_Toc208508956)

[1.3 Приемка ОС 15](#_Toc208508957)

[1.4 Требования к техническим средствам 15](#_Toc208508958)

[1.5 Состав ОС 16](#_Toc208508959)

[2 Установка Системы 17](#_Toc208508960)

[2.1 Безопасная установка и эксплуатация ОС 17](#_Toc208508961)

[2.1.1 Условия безопасной эксплуатации 17](#_Toc208508962)

[2.1.2 Доверенная загрузка 17](#_Toc208508963)

[2.1.3 Защита информации 17](#_Toc208508964)

[2.1.4 Безопасная конфигурация ОС и ее целостность 18](#_Toc208508965)

[2.1.5 Пользователи и администраторы 18](#_Toc208508966)

[2.1.6 Настройка точного времени 18](#_Toc208508967)

[2.2 Локальная установка 19](#_Toc208508968)

[2.2.1 Подготовка к установке 19](#_Toc208508969)

[2.2.2 Установка ОС на рабочую станцию 19](#_Toc208508970)

[2.2.3 Установка серверной версии ОС 26](#_Toc208508971)

[2.2.4 Парольная защита загрузчика ОС 26](#_Toc208508972)

[2.3 Установка с использованием средств маскирования информации 27](#_Toc208508973)

[2.3.1 Общие вопросы 27](#_Toc208508974)

[2.3.2 Разметка носителя с маскированием информации 27](#_Toc208508975)

[2.3.3 Поддержка tpm2 для автоматического ввода паролей 29](#_Toc208508976)

[2.4 Установка на виртуальную машину 29](#_Toc208508977)

[2.5 Автоматизированное развертывание 29](#_Toc208508978)

[2.5.1 Введение в kickstart-сценарии 29](#_Toc208508979)

[2.5.2 Типовой сценарий автоматизированной установки 30](#_Toc208508980)

[2.5.3 Дополнительные действия в kickstart-сценариях 33](#_Toc208508981)

[2.6 Сетевая загрузка с возможностью установки 34](#_Toc208508982)

[2.6.1 Технология Intel PXE 34](#_Toc208508983)

[2.6.2 Начальная настройка сервера для работы PXE 35](#_Toc208508984)

[2.6.3 Настройка сервера DHCP (BOOTP) 35](#_Toc208508985)

[2.6.4 Настройка веб-сервера 36](#_Toc208508986)

[2.6.5 Настройка сервера TFTP 37](#_Toc208508987)

[2.6.6 Настройка сети в LiveCD 38](#_Toc208508988)

[2.7 Графическая установка с использованием VNC 39](#_Toc208508989)

[2.8 OEM-установка 40](#_Toc208508990)

[2.9 Установщик iso2img 42](#_Toc208508991)

[3 Общие сведения о работе в Системе 43](#_Toc208508992)

[3.1 Интерфейсы Системы 43](#_Toc208508993)

[3.2 Администраторы и пользователи 44](#_Toc208508994)

[3.2.1 Использование команды sudo 44](#_Toc208508995)

[3.2.2 Управление учетными записями пользователей 45](#_Toc208508996)

[3.2.3 Восстановление забытого пароля пользователя 47](#_Toc208508997)

[3.3 Команды 49](#_Toc208508998)

[3.4 Системный менеджер systemd 52](#_Toc208508999)

[3.4.1 Управление службами 53](#_Toc208509000)

[3.4.2 Unit-файлы 54](#_Toc208509001)

[3.4.3 Состояние Системы 61](#_Toc208509002)

[3.4.4 systemd-resolved 62](#_Toc208509003)

[3.5 Файловые системы 64](#_Toc208509004)

[3.5.1 Поддерживаемые файловые системы 64](#_Toc208509005)

[3.5.2 Организация каталогов в Системе 65](#_Toc208509006)

[3.5.3 Монтирование 67](#_Toc208509007)

[3.5.4 Размонтирование 70](#_Toc208509008)

[3.5.5 Файловая система GFS2 71](#_Toc208509009)

[3.6 Установка и удаление программ (репозитории) 73](#_Toc208509010)

[3.6.1 Использование rpm 74](#_Toc208509011)

[3.6.2 Файловый менеджер dnf 74](#_Toc208509012)

[3.6.3 Установка программ из сторонних источников 76](#_Toc208509013)

[3.6.4 Использование Notamock для сборки пакетов 76](#_Toc208509014)

[3.7 Управление жесткими дисками 81](#_Toc208509015)

[3.7.1 Fdisk 81](#_Toc208509016)

[3.7.2 Диспетчер разделов KDE 83](#_Toc208509017)

[3.7.3 GParted 85](#_Toc208509018)

[3.8 Режимы работы Системы 87](#_Toc208509019)

[3.8.1 Режим восстановления 88](#_Toc208509020)

[3.8.2 Аварийный режим 88](#_Toc208509021)

[3.8.3 Синхронизация с репозиториями 89](#_Toc208509022)

[3.8.4 Восстановление функционирования ОС 89](#_Toc208509023)

[3.8.5 Восстановление загрузчика GRUB2 (MBR и EFI) 90](#_Toc208509024)

[3.8.6 Восстановление Системы из среды chroot 92](#_Toc208509025)

[3.9 Перезагрузка и остановка Системы 94](#_Toc208509026)

[3.9.1 Перезагрузка 94](#_Toc208509027)

[3.9.2 Остановка 95](#_Toc208509028)

[3.9.3 Shutdown 95](#_Toc208509029)

[3.10 Обновление Системы 97](#_Toc208509030)

[3.10.1 Общие сведения 97](#_Toc208509031)

[3.10.2 Индикация обновлений 97](#_Toc208509032)

[3.10.3 Порядок получения обновлений 98](#_Toc208509033)

[3.11 Сохранение вывода консоли в файл 99](#_Toc208509034)

[3.12 Запись экрана из консоли 101](#_Toc208509035)

[3.12.1 Перекодирование видео 102](#_Toc208509036)

[4 Аутентификация пользователей и управление доступом 103](#_Toc208509037)

[4.1 Общие сведения о PAM 103](#_Toc208509038)

[4.2 Механизм работы PAM 103](#_Toc208509039)

[4.3 Иерархия и приоритетность конфигурации PAM 104](#_Toc208509040)

[4.4 Конфигурация распространённых модулей PAM 105](#_Toc208509041)

[4.4.1 pam\_unix.so 105](#_Toc208509042)

[4.4.2 pam\_env.so 106](#_Toc208509043)

[4.4.3 pam\_faillock.so 106](#_Toc208509044)

[4.4.4 pam\_limits.so 108](#_Toc208509045)

[4.4.5 pam\_succeed\_if.so 109](#_Toc208509046)

[4.4.6 pam\_p11.so 110](#_Toc208509047)

[4.5 Блокирование неактивных учётных записей 111](#_Toc208509048)

[4.5.1 Блокирование при входе через GDM 111](#_Toc208509049)

[4.5.2 Блокирование при входе через терминал (TTY, SSH) 112](#_Toc208509050)

[4.5.3 Просмотр справки по модулю pam\_lastlog 112](#_Toc208509051)

[4.6 Отладка и проверка работы PAM 112](#_Toc208509052)

[4.6.1 Просмотр системного журнала 113](#_Toc208509053)

[4.6.2 Проверка конфигурации и тестирование изменений 113](#_Toc208509054)

[4.6.3 Использование временного доступа через root-tty 113](#_Toc208509055)

[4.6.4 Примеры отладочной информации от модулей 114](#_Toc208509056)

[4.6.5 Проверка порядка выполнения модулей 114](#_Toc208509057)

[4.7 Настройка двухфакторной аутентификации 114](#_Toc208509058)

[4.7.1 Подготовка рабочих станций 115](#_Toc208509059)

[4.7.2 Подготовка токена Рутокен 115](#_Toc208509060)

[4.7.3 Извлечение и перенос публичного ключа 116](#_Toc208509061)

[4.7.4 Подготовка целевой станции 116](#_Toc208509062)

[4.7.5 Список ключей пользователя 117](#_Toc208509063)

[4.7.6 Создание общего файла настроек PAM 118](#_Toc208509064)

[4.7.7 Расширенная настройка 119](#_Toc208509065)

[5 Сетевые подключения 124](#_Toc208509066)

[5.1 Способы настройки сетевых соединений 124](#_Toc208509067)

[5.1.1 NetworkManager 124](#_Toc208509068)

[5.1.2 Использование nmtui 125](#_Toc208509069)

[5.1.3 Использование nmcli 126](#_Toc208509070)

[5.1.4 Использование ifconfig и ip 127](#_Toc208509071)

[5.1.5 Виртуальный сетевой интерфейс Loopback 129](#_Toc208509072)

[5.1.6 Сетевой интерфейс ens\* 130](#_Toc208509073)

[5.2 Настройка статических соединений 131](#_Toc208509074)

[5.2.1 Настройка статического IP-адреса 133](#_Toc208509075)

[5.3 Настройка динамических соединений 133](#_Toc208509076)

[5.3.1 Настройка динамических соединений (кроме DNS) 133](#_Toc208509077)

[5.4 Переадресация 134](#_Toc208509078)

[5.4.1 Включение переадресации IPv4 135](#_Toc208509079)

[5.4.2 Включение переадресации IPv6 135](#_Toc208509080)

[5.4.3 Настройка правил переадресации с использованием nftables 136](#_Toc208509081)

[5.5 Мониторинг и диагностика сети 137](#_Toc208509082)

[5.6 Межсетевой экран Nftables 138](#_Toc208509083)

[5.6.1 Общие сведения о Nftables 138](#_Toc208509084)

[5.6.2 Установка и запуск Nftables 140](#_Toc208509085)

[5.6.3 Работа с Nftables 140](#_Toc208509086)

[6 Контроль доступа к файлам 143](#_Toc208509087)

[6.1 Описание технологии ACL 143](#_Toc208509088)

[6.2 Примеры настройки ACL 144](#_Toc208509089)

[7 Резервное копирование данных 147](#_Toc208509090)

[7.1 Планирование резервного копирования и восстановления данных 148](#_Toc208509091)

[7.2 Создание резервных копий 149](#_Toc208509092)

[7.3 Восстановление данных из резервной копии 153](#_Toc208509093)

[8 Контроль целостности Системы 154](#_Toc208509094)

[8.1 Общие принципы работы AIDE 154](#_Toc208509095)

[8.2 Установка и создание базы данных 154](#_Toc208509096)

[8.2.1 Настройка конфигурационного файла 155](#_Toc208509097)

[8.3 Проверка целостности 157](#_Toc208509098)

[8.3.1 Автоматизация проверки 157](#_Toc208509099)

[9 Формирование замкнутой программной среды 159](#_Toc208509100)

[9.1 Общие принципы 159](#_Toc208509101)

[9.2 Создание ключей 159](#_Toc208509102)

[9.3 Создание политики IMA 161](#_Toc208509103)

[9.4 Подписывание файлов 162](#_Toc208509104)

[9.5 Подготовка файловой системы 163](#_Toc208509105)

[9.6 Добавление в образ начальной загрузки 164](#_Toc208509106)

[9.7 Пробный запуск подписанной Системы 164](#_Toc208509107)

[9.8 Рабочий запуск и проверка работы 164](#_Toc208509108)

[10 Управление идентификацией и аутентификацией с FreeIPA 166](#_Toc208509109)

[10.1 Настройка сервера домена IPA 166](#_Toc208509110)

[10.1.1 Создание пользователей 168](#_Toc208509111)

[10.1.2 Настройка сети 168](#_Toc208509112)

[10.2 Настройка клиента домена IPA 169](#_Toc208509113)

[10.3 Вход в домен 170](#_Toc208509114)

[11 Аудит 171](#_Toc208509115)

[11.1 Общие сведения о auditd 171](#_Toc208509116)

[11.2 Управление службой аудита 171](#_Toc208509117)

[11.3 Настройка конфигурации аудита 171](#_Toc208509118)

[11.4 Настройка правил аудита 172](#_Toc208509119)

[11.4.1 Использование ключа поиска событий 173](#_Toc208509120)

[11.5 Выборка из логов аудита 174](#_Toc208509121)

[11.6 Примеры работы со службой аудита 175](#_Toc208509122)

[11.6.1 Отслеживание изменений в директории или файле 175](#_Toc208509123)

[11.6.2 Отслеживание запуска определенного приложения 175](#_Toc208509124)

[11.6.3 Отслеживание системных вызовов 176](#_Toc208509125)

[11.6.4 Отслеживание доступа к конфиденциальным файлам 176](#_Toc208509126)

[11.6.5 Отслеживание сетевых подключений 176](#_Toc208509127)

[11.6.6 Запись в журнал аудита при подключении устройства USB 176](#_Toc208509128)

[11.6.7 Централизованный сбор событий аудита 177](#_Toc208509129)

[12 Настройка и управление сетевыми сервисами 179](#_Toc208509130)

[12.1 Сервер DHCP 179](#_Toc208509131)

[12.1.1 Установка 179](#_Toc208509132)

[12.1.2 Настройка 179](#_Toc208509133)

[12.1.3 Присвоение адреса 181](#_Toc208509134)

[12.1.4 Настройка клиента 182](#_Toc208509135)

[12.2 Сервер DNS 182](#_Toc208509136)

[12.2.1 Установка 182](#_Toc208509137)

[12.2.2 Настройка 183](#_Toc208509138)

[12.2.3 Переадресация 184](#_Toc208509139)

[12.3 Сервер Samba 185](#_Toc208509140)

[12.3.1 Настройка сети 185](#_Toc208509141)

[12.3.2 Запуск контроллера домена 187](#_Toc208509142)

[12.3.3 Перенаправление 189](#_Toc208509143)

[12.3.4 Использование внешнего DNS-сервера BIND 191](#_Toc208509144)

[12.3.5 Подключение клиента ROSA к домену 192](#_Toc208509145)

[12.4 Веб-серверы 197](#_Toc208509146)

[12.4.1 Apache 197](#_Toc208509147)

[12.4.2 Веб-сервер Nginx 202](#_Toc208509148)

[12.4.3 Веб-сервер Angie 219](#_Toc208509149)

[12.5 Веб-интерфейс управления Cockpit 233](#_Toc208509150)

[12.6 Сервер журналирования 235](#_Toc208509151)

[12.6.1 Общие сведения 235](#_Toc208509152)

[12.6.2 Установка сервера 235](#_Toc208509153)

[12.6.3 Настройка сервера 236](#_Toc208509154)

[12.6.4 Настройка клиента 238](#_Toc208509155)

[12.6.5 Запуск логирования journald 239](#_Toc208509156)

[12.6.6 Проверка работоспособности 239](#_Toc208509157)

[12.7 Сервер печати CUPS 240](#_Toc208509158)

[12.7.1 Добавление и настройка принтеров 241](#_Toc208509159)

[12.8 Почтовый сервер OpenSMTPD 244](#_Toc208509160)

[12.8.1 Настройка OpenSMTPD 245](#_Toc208509161)

[12.8.2 Проверка работы OpenSMTPD 246](#_Toc208509162)

[12.8.3 Интеграция OpenSMTPD с Dovecot 248](#_Toc208509163)

[12.8.4 Дополнительные рекомендации по безопасности и интеграции 249](#_Toc208509164)

[13 Удаленный доступ к серверу 253](#_Toc208509165)

[13.1 Доступ по OpenSSH 253](#_Toc208509166)

[13.1.1 Общие сведения об OpenSSH 253](#_Toc208509167)

[13.1.2 Настройка SSH 253](#_Toc208509168)

[13.1.3 Пары ключей 255](#_Toc208509169)

[13.2 Терминальный доступ с использованием XRDP 257](#_Toc208509170)

[13.2.1 Настройка локальных пользователей 258](#_Toc208509171)

[13.2.2 Настройка доменных пользователей 259](#_Toc208509172)

[13.2.3 Подключение с клиента 259](#_Toc208509173)

[14 Конфигурация системы для защиты информации 261](#_Toc208509174)

[14.1 Общее описание 261](#_Toc208509175)

[14.2 Настройки службы SSH 261](#_Toc208509176)

[14.3 Отключение файловых систем без атрибутов безопасности 261](#_Toc208509177)

[14.4 Отключение редких сетевых протоколов 262](#_Toc208509178)

[14.5 Отключение дампов оперативной памяти 262](#_Toc208509179)

[14.6 Настройка переменных ядра 262](#_Toc208509180)

[14.7 Настройки аудита 262](#_Toc208509181)

[14.8 Отключение настроек блокировки экрана 263](#_Toc208509182)

[14.9 Настройка автоблокировки консольного режима 263](#_Toc208509183)

[15 Системы управления базами данных 264](#_Toc208509184)

[15.1 Доступные СУБД 264](#_Toc208509185)

[15.2 PostgreSQL 264](#_Toc208509186)

[15.3 MySQL 265](#_Toc208509187)

[15.4 SQLITE 266](#_Toc208509188)

[16 Автоматизация инфраструктуры с Ansible 267](#_Toc208509189)

[16.1 Установка и подключение хостов 267](#_Toc208509190)

[16.2 Проверка подключения 268](#_Toc208509191)

[16.3 Описание сценариев Ansible 268](#_Toc208509192)

[16.4 Создание сценариев 270](#_Toc208509193)

[16.5 Выполнение одиночных команд 270](#_Toc208509194)

[17 Контейнеризация 272](#_Toc208509195)

[17.1 Docker 272](#_Toc208509196)

[17.2 Kubernetes 274](#_Toc208509197)

[17.2.1 Общие сведения 274](#_Toc208509198)

[17.2.2 Установка kubernates master-node 275](#_Toc208509199)

[17.2.3 Установка kubernates worker-nodes (pods-контейнеры) 276](#_Toc208509200)

[18 Cбор отчетов о производительности 279](#_Toc208509201)

[18.1 Общие сведения о Zabbix 279](#_Toc208509202)

[18.2 Установка и настройка 280](#_Toc208509203)

[18.2.1 Установка компонентов Zabbix 280](#_Toc208509204)

[18.2.2 Настройка веб-интерфейса 281](#_Toc208509205)

[18.2.3 Установка Zabbix Agent на целевых машинах 281](#_Toc208509206)

[18.2.4 Завершение настройки 282](#_Toc208509207)

[19 Кластеризация и высокая доступность 283](#_Toc208509208)

[19.1 Общие сведения 283](#_Toc208509209)

[19.1.1 Типы ресурсов 284](#_Toc208509210)

[19.1.2 Политики управления 284](#_Toc208509211)

[19.2 Развёртывание и настройка Pacemaker и Corosync 285](#_Toc208509212)

[19.3 Управление ресурсами и политиками в кластере 286](#_Toc208509213)

[19.3.1 Отключение STONITH 286](#_Toc208509214)

[19.3.2 Отключение контроля кворума 287](#_Toc208509215)

[19.3.3 Добавление ресурсов в кластер 287](#_Toc208509216)

[19.3.4 Настройка политики совместного размещения ресурсов 288](#_Toc208509217)

[19.4 Проверка работоспособности кластера 288](#_Toc208509218)

[19.5 Мониторинг и управление отказами 289](#_Toc208509219)

[19.6 Управление ресурсами и диагностика 289](#_Toc208509220)

[20 Локальная виртуализация с использованием qemoo 291](#_Toc208509221)

[20.1 Запуск виртуальных машин с помощью qemoo 291](#_Toc208509222)

[20.1.1 Загрузка в режиме EFI 292](#_Toc208509223)

[20.1.2 Добавление дополнительных устройств и параметров 292](#_Toc208509224)

[20.1.3 Загрузка в режиме, имитирующем запись ISO-образа на носитель 292](#_Toc208509225)

[20.1.4 Проброс USB-устройств 292](#_Toc208509226)

[20.2 Установка ОС на виртуальный диск 293](#_Toc208509227)

[20.3 Сетевые режимы и проброс устройств 293](#_Toc208509228)

[20.3.1 Автоматическое подключение к подсети virbr0 294](#_Toc208509229)

[20.3.2 Настройка собственного сетевого моста 294](#_Toc208509230)

[20.3.3 Проброс каталогов в гостевую систему 294](#_Toc208509231)

[20.3.4 Проброс USB-устройств 294](#_Toc208509232)

[20.4 Работа в демон-режиме (SPICE) 295](#_Toc208509233)

[20.4.1 Запуск в SPICE-режиме 295](#_Toc208509234)

[20.4.2 Управление виртуальной машиной через systemd 295](#_Toc208509235)

[20.5 Конфигурация и приоритет параметров 296](#_Toc208509236)

[20.5.1 Типовые параметры конфигурации 297](#_Toc208509237)

[21 Создание собственного репозитория пакетов 298](#_Toc208509238)

[21.1 Зеркалирование репозиториев 298](#_Toc208509239)

[21.2 Создание собственного репозитория 299](#_Toc208509240)

[21.3 Подключение локального репозитория 300](#_Toc208509241)

[21.4 Доступ к репозиторию по сети 300](#_Toc208509242)

[21.4.1 Доступ по HTTP 300](#_Toc208509243)

[21.4.2 Доступ по NFS 302](#_Toc208509244)

[Перечень терминов и сокращений 304](#_Toc208509245)

1. Общие сведения
   1. Общее описание

ОС РОСА "ХРОМ" предназначена для оснащения рабочих мест пользователей в корпоративной и государственной среде, предоставляя пользователям современный и интуитивно понятный интерфейс.

Для ОС доступны образы с окружением рабочего стола "KDE Plasma 5".

Версии ОС представлены в исполнениях "рабочая станция" и "сервер", которые позволяют использовать Систему в качестве базовой платформы по выполнению рутинных задач разной степени сложности.

* 1. Основные характеристики
     1. Аппаратная поддержка

Образ продукта собран на новом LTS-ядре Linux 6.12. Ядра веток 6.6, 6.1, 5.10 и 5.15 доступны в репозиториях.

Поддержка множества Wi-Fi-карт осуществляется с помощью модулей ядра, а также bluetooth-модулей – в большинстве случаев "из коробки".

Обеспечена поддержка новейших видеокарт NVIDIA: репозитории содержат драйверы Nvidia 340, 390, 470, 510, 515, 520, 525 в виде модулей ядра с автоустановкой посредством фирменной консольной утилиты kroko-cli.

Система содержит обновлённые библиотеками трёхмерной графики MESA.

* + 1. Программная платформа

Программная платформа 2021.1 построена на независимой от других дистрибутивов пакетной базе.

Платформа реализована на основе собственной среда сборки abf.io, в которой пользователи могут сами собирать нужные им пакеты и создавать собственные репозитории.

В платформе используются системные программные продукты: glibc 2.33, gcc 11.2, Java версий 1.8, 11, 17.

* + 1. Программное обеспечение

В качестве прикладного ПО в составе ОС используются:

* свежие пакеты программ, в большинстве своем соответствующие стабильной ветке Upstream;
* рабочая оболочка на основе KDE Plasma 5.27 с собственным вариантом оформления и поддержкой темной темы, в том числе добавлен новый алгоритм позиционирования иконок на рабочем столе из версии Plasma 6.3;
* фирменный индикатор обновлений Системы, автопереключение на резервные "зеркала" при недоступности основных;
* в Систему добавлены российские корневые сертификаты ТЦИ.

В состав ОС включены следующие категории дополнительных системных приложений:

* архиваторы и утилиты сжатия;
* средства управления пакетами и обновлениями (DNF/RPM);
* приложения мониторинга, журналирования и диагностики;
* приложения для работы с файлами и файловыми системами;
* средства настройки системы и учётных записей (локализация, дата/время, службы systemd);
* настройка параметров загрузки и восстановления (GRUB2, целевые состояния systemd);
* настройка оборудования (графика, звук, печать, Bluetooth и др.);
* утилиты настройки сети (NetworkManager);
* межсетевой экран и NAT на базе nftables;
* средства удалённого доступа и администрирования (OpenSSH, Xrdp);
* инструменты резервного копирования и переноса данных;
* средства работы с образами и установочными носителями (создание/запись ISO).
  + 1. Сети и развертывание

В Системе реализованы следующие механизмы сетевой конфигурации и автоматизированного развертывания:

* автоматизированная установка по сценариям kickstart: задание разметки дисков, набора пакетов, сетевых параметров, pre-/post-скриптов; поддерживается самостоятельное задание учётных данных при первом входе в Систему;
* установка по сети с использованием PXE/iPXE, загрузка по HTTP/HTTPS (включая UEFI), подключение установочных репозиториев по HTTPS с проверкой криптографических подписей пакетов;
* управление сетевыми подключениями на базе NetworkManager (утилиты nmcli/nmtui): проводные и беспроводные сети, статическая и динамическая (DHCPv4/DHCPv6) конфигурация, DNS и маршрутизация, поддержка IPv4/IPv6 (dual-stack), VLAN, мостирование (bridge), агрегирование (bonding), 802.1X, VPN (например, OpenVPN, IPsec, WireGuard), настройка прокси;
* синхронизация времени по NTP (chrony) с поддержкой настройки источников времени и корректной работы в доменной среде;
* межсетевой экран на базе nftables для фильтрации трафика и NAT на уровне узла; интеграция с systemd (управление сервисами правил, журналирование и контроль состояния);
* присоединение к доменным инфраструктурам: IPA и Samba/Active Directory (через realmd/SSSD, Kerberos/LDAP), централизованная аутентификация, применение политик доступа, автоматическое создание домашних каталогов пользователей;
* использование репозиториев программного обеспечения по HTTPS, возможность организации локальных зеркал и кэширующих прокси для массовых развертываний; управление установками и обновлениями через DNF при низкоуровневом учёте пакетов в RPM;
* средства постустановочной конфигурации для тиражирования настроек (скрипты оболочки, профили systemd, при необходимости — инструменты оркестрации, такие как Ansible).
  + 1. Защита информации

В части защиты информации в ОС поддерживается:

* запутывающее кодирование по ГОСТ в браузерах и OpenSSL;
* технология подписей исполняемых файлов IMA;
* установка и загрузка с шифрованных разделов без ввода пароля при загрузке (технология tpm2).
  + 1. Состав дистрибутива

В состав образа сервера входят следующие компоненты:

* графический, консольный и автоматизированный установщики;
* ядро Linux 6.12;
* системный менеджер systemd;
* сервер SSH openssh-server;
* служба синхронизация времени chrony;
* служба автоматической или ручной настройки сети NetworkManager;

В состав образа рабочей станции так же входит графическая оболочка на основе KDE Plasma5 и комплект прикладных программ.

* + 1. Доступные в репозитории компоненты

Другие компоненты можно установить из обширного официального [репозитория](http://mirror.rosalinux.ru/rosa/rosa2021.1/repository/x86_64/) РОСА, например:

* служба каталогов FreeIPA (пакет ipa-server);
* служба каталогов Samba AD;
* DNS-сервер bind;
* база данных MariaDB (MySQL) (пакет mariadb);
* база данных PostgreSQL (пакет postgresql, либо postgresql14, либо postgresql15);
* веб- и прокси-сервер apache (с [поддержкой](https://abf.io/import/apache/blob/rosa2021.1/README.GOST) пакета apache-base);
* веб- и прокси-сервер nginx и различные модули для него (пакеты nginx-module-\*);
* система мониторинга [Zabbix](http://wiki.rosalab.ru/ru/index.php/Zabbix) (пакет zabbix5.0), Nagios, Monitorix, Monit и т.д.;
* виртуализация (qemu+libvirt);
* российское ответвление nginx – [angie](https://www.opennet.ru/opennews/art.shtml?num=58036) и те же самые модули, что и для nginx:
* angie-module-perl;
* angie-module-rtmp;
* angie-module-upload;
* angie-module-http\_auth\_pam;
* angie-module-ndk;
* angie-module-array-var;
* angie-module-aws-auth;
* angie-module-brotli;
* angie-module-spnego-auth;
* и многие другие.

Репозиторий ОС "сервер" является общим с репозиторием ОС "рабочая станция".

Также можно установить различные графические оболочки для ОС, однако рекомендуется использовать Систему с имеющейся в образе оболочкой KDE Plasma5, в остальных случаях разработчик не может гарантировать корректность работы ОС. Возможные варианты с графическими средами приведены ниже (таблица 1).

Таблица 1 – Графические оболочки

|  |  |
| --- | --- |
| Графическая оболочка | Пакет для установки |
| XFCE | task-xfce для XFCE с настройками ОС или task-xfce-minimal для XFCE без доп. настроек |
| KDE Plasma5 | task-plasma5 или task-iso-plasma5 |
| GNOME | task-iso-gnome для установки GNOME с настройками, дополнениями и другим ПО, как в образе с GNOME; для установки "ванильного" GNOME: gdm gnome-session-x11-session gnome-session-wayland-session gnome-classic-session |
| LXQt | task-iso-lxqt |
| MATE | task-mate |
| Lumina | task-lumina |
| icewm | icewm или task-iso-icewm |

Дополнительно необходимо установить пакет task-x11.

Примечание – Проверить целостность образов можно командой:

checkisomd5 ИмяОбраза

Или выбрав при начальной загрузке Системы с инсталляционного носителя режим "Проверка целостности" в меню загрузчика "Диагностика".

* 1. Приемка ОС

Для приемки изделия необходимо:

1. проверить комплектность изделия; в комплект должны входить:

* оптические диски с ОС и комплектом документации согласно выбранному варианту поставки;
* лицензионное соглашение.

1. проверить контрольные суммы установочных дистрибутивов с Системой в соответствии с КС, указанной в файле sha1, который поставляется вместе с образом ОС.
2. приемка изделия считается успешной, если все указанные требования выполнены.
   1. Требования к техническим средствам

Для корректной установки и функционирования ОС необходима следующая минимальная конфигурация оборудования (ПК):

* процессор с архитектурой x86\_64 или AArch64, Байкал;
* оперативная память – 2 Гб (4 Гб – при загрузке ОС по сети);
* дисковое пространство – 25 Гб;
* VGA-адаптер и монитор с поддержкой разрешения 1024×768 пикс. (24 бит);
* устройство чтения оптических дисков или возможность загрузки с USB-носителя;
* клавиатура;
* компьютерная "мышь".
  1. Состав ОС

ОС включает в себя следующие основные компоненты:

* ядро и системные библиотеки;
* системные сервисы и программы;
* пользовательские программы.

Ядро и системные библиотеки обеспечивают взаимодействие ОС с аппаратной частью и выделение ресурсов для всех других компонентов.

Системные сервисы и программы дают возможность настройки и реконфигурации ОС в зависимости от задач пользователя и обслуживание повседневной работы.

Пользовательские программы предназначены для решения задач конечного пользователя.

1. Установка Системы
   1. Безопасная установка и эксплуатация ОС
      1. Условия безопасной эксплуатации

Для безопасной эксплуатации ОС должна быть установлена на совместимую аппаратную часть, удовлетворяющую требованиям п.1.4.

Организационными или техническими средствами (с помощью пломб или замков) должна быть обеспечена защита физической целостности корпуса компьютера с установленной ОС от изменения аппаратной конфигурации.

Установку, настройку и обслуживание ОС должен проводить обладающий необходимыми компетенциями администратор ОС в соответствии с настоящим Руководством.

Эксплуатацию ОС должны выполнять обладающие необходимыми компетенциями пользователи в соответствии с документом "ОС РОСА "ХРОМ". Руководство пользователя. РСЮК.10501-01 34 01".

* + 1. Доверенная загрузка

Для безопасной работы в ОС должна быть реализована её доверенная загрузка, обеспечиваемая внешними средствами АПМДЗ/СДЗ и/или организационными мерами, а именно:

* защитой BIOS паролем от изменения;
* отключением в BIOS загрузки с внешних носителей (DVD, подключаемых накопителей) согласно документации к аппаратуре после установки;
* защитой паролем режима изменения параметров загрузчика GRUB (п.2.2.4).
  + 1. Защита информации

Администратор должен обеспечить механизмы резервного копирования и необходимый объем носителей для защищенного хранения резервных копий данных пользователей согласно инструкциям, описанным в п. 7.

При использовании парольной защиты для идентификации во всех случаях администратор должен использовать пароли достаточной степени сложности. При использовании небезопасных паролей ОС предупреждает об этом; при получении такого предупреждения нужно изменить пароль на более сложный.

Рекомендуется произвести установку ОС с использованием механизма маскирования информации на носителе согласно п.2.3.

Для предотвращения утечки информации с помощью записи её на съемные внешние накопители (USB-флеш, внешние HDD, диски DWD-RW и др.) должен быть организован процесс контроля их использования. Это возможно сделать как чисто организационными мерами, так и аппаратным отключением возможности подключения внешних накопителей на пользовательских компьютерах (с помощью настроек BIOS).

Для предотвращения несанкционированного доступа к информации в момент, когда авторизованный пользователь ОС делает перерыв в работе, он должен блокировать экран нажатием сочетания META+L на клавиатуре в графическом режиме или же командой vlock, если он работает в консольном текстовом режиме.

* + 1. Безопасная конфигурация ОС и ее целостность

ОС также может быть сконфигурирована для безопасной работы с помощью файлов конфигурации, описанных в п.14. Необходимо контролировать с помощью программы AIDE целостность Системы путем создания базы данных после инсталляции, как показано в п.8, и периодического осуществления контроля.

* + 1. Пользователи и администраторы

Первый созданный инсталлятором пользователь при установке по умолчанию обладает доступом к правам администратора через команду sudo, т. е. является администратором ОС. Эту учетную запись необходимо применять только для настройки ОС, пользователи должны работать под отдельной учетной записью с пользовательскими правами. Подробнее об этом см. п. 3.2.

* + 1. Настройка точного времени

Для правильного ведения журналов событий и аудита в компьютере и ОС должно быть установлено актуальное системное время. Администратору необходимо контролировать установку правильного часового пояса и сохранение этого времени при выключении компьютера (поддерживать актуальность автономного питания BIOS с помощью встроенной батареи).

ОС сконфигурирована так, что время автоматически синхронизируется по сети Интернет с публичными серверами. Доступ к этой настройке имеют администраторы ОС, пользователь менять время по умолчанию не может.

* 1. Локальная установка
     1. Подготовка к установке

Перед началом установки ОС рекомендуется выполнить следующие действия:

1. убедиться, что аппаратная конфигурация ПК удовлетворяет минимальным аппаратным требованиям, указанным в п.1.4;
2. убедиться в подлинности DVD-диска с дистрибутивом ОС (п.1.3);
3. скопировать с DVD-диска с дистрибутивом ОС образ на USB-носитель (если установка будет производиться с USB-носителя);
4. провести временную настройку BIOS/UEFI компьютера для обеспечения возможности загрузки с выбранного носителя, используя документацию на оборудование. В большинстве случаев выбор вариантов загрузки становится доступен при удержании клавиш Del, F12 или F2 при начале загрузки компьютера.

Для обеспечения доверенной загрузки ОС рекомендуется после успешной инсталляции режим загрузки с внешнего накопителя отключить.

* + 1. Установка ОС на рабочую станцию

Выбор позиций меню при установке осуществляется стрелками на клавиатуре и нажатием клавиши Enter или же выбором элемента меню курсором и нажатием левой кнопки "мыши".

Для того чтобы провести процедуру установки ОС, необходимо выполнить следующие действия:

1. вставить в устройство чтения DVD-дисков загрузочный диск изделия или USB-носитель с дистрибутивом ОС до того, как ПК начнет обращение к загрузочным дискам. Если обращение уже было произведено, и установка изделия не началась, необходимо перезагрузить ПК.
2. выбрать язык, на котором будет производиться установка.
3. выбрать вариант установки "Запуск и установка ROSA Linux (live)" (рекомендуется) или же режим "Установка ROSA Linux", при котором установка идет без предварительной загрузки ОС (рисунок 1). Такой режим применим для специальных случаев, например при недостатке оперативной памяти или при необходимости ускорения установки.

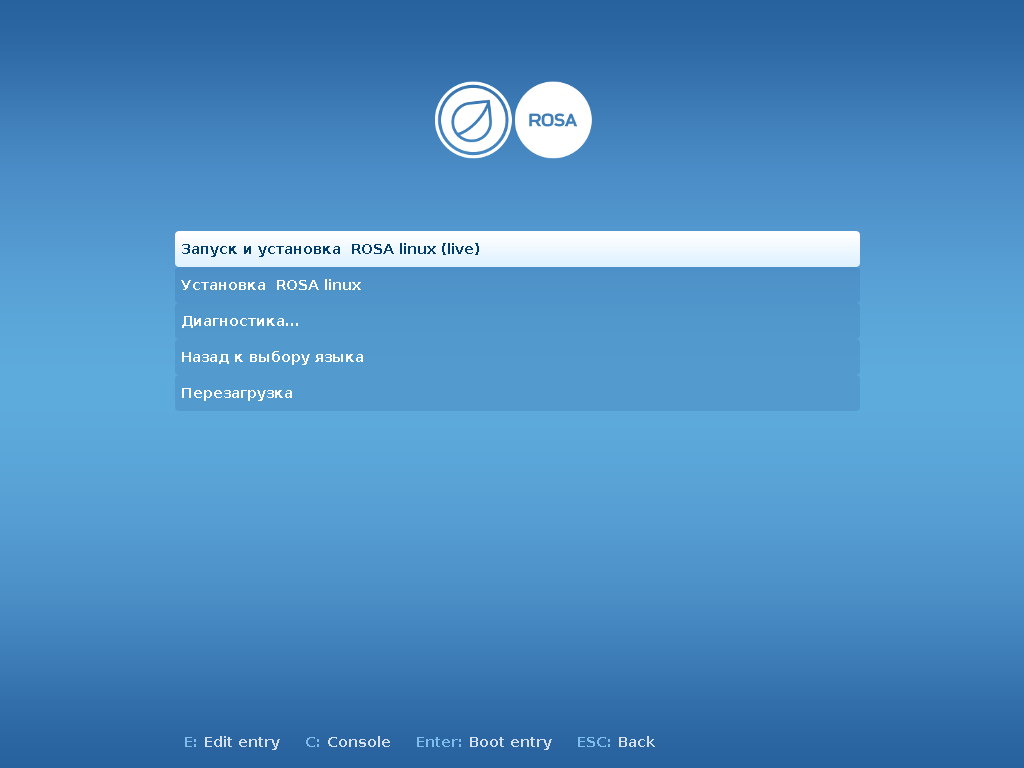


Рисунок 1 – Выбор варианта установки

1. При проблемах с установкой может быть полезным пункт "Диагностика", в подменю которого доступны варианты с проверкой контрольной суммы образа перед установкой, проверкой оперативной памяти ПК или установкой в режиме упрощенной графики либо в текстовом режиме. Перед обращением в техническую поддержку при каких-либо проблемах настоятельно рекомендуется как минимум провести диагностику оперативной памяти (полный цикл) и попробовать загрузить установочный образ с предварительным тестированием контрольной суммы, что позволит исключить ошибки аппаратного обеспечения. При наличии сбоев оперативной памяти необходимо заменить ее на исправную, при сбое контрольной суммы образа следует использовать другой носитель или другой USB-порт. Пример меню "Диагностика" приведен ниже (рисунок 2) (конкретный внутренний номер образа зависит от версии и может понадобиться при обращении в техническую поддержку).

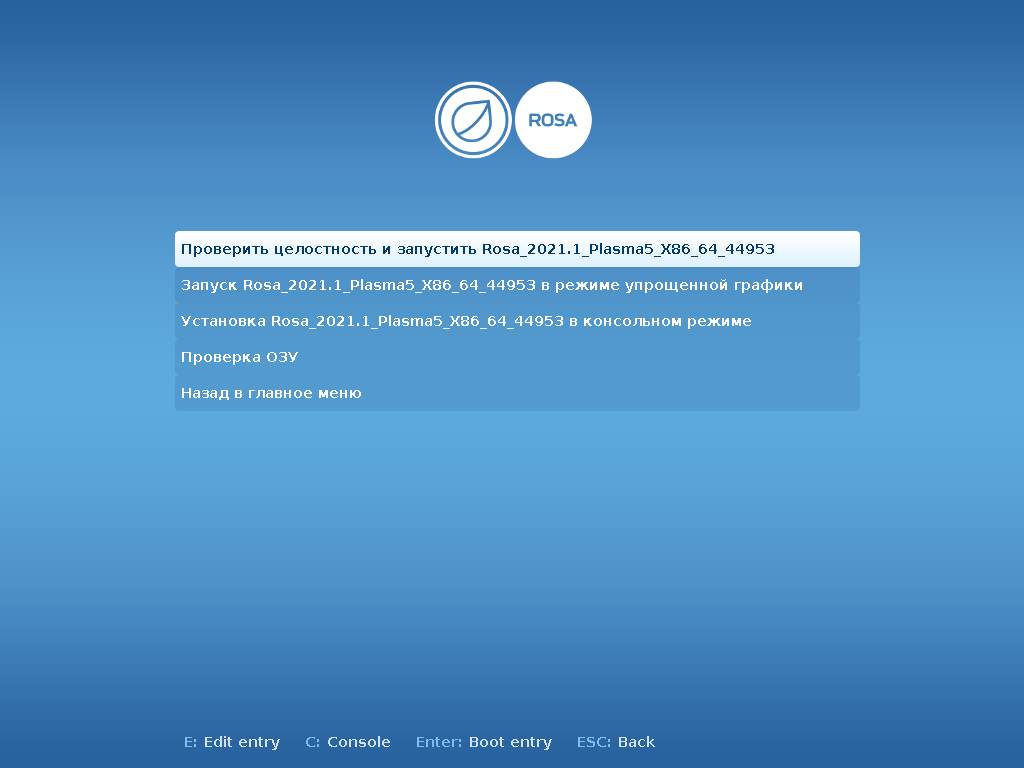


Рисунок 2 – Меню "Диагностика"

1. После загрузки в live-режиме появляется возможность полноценной проверки работы ОС на конкретном ПК с возможностью установки с помощью ярлыка инсталлятора на рабочем столе (рисунок 3). Перед установкой рекомендуется подключить ОС к сети (используя кнопку  на панели задач), тогда часть параметров будет заполнена автоматически.

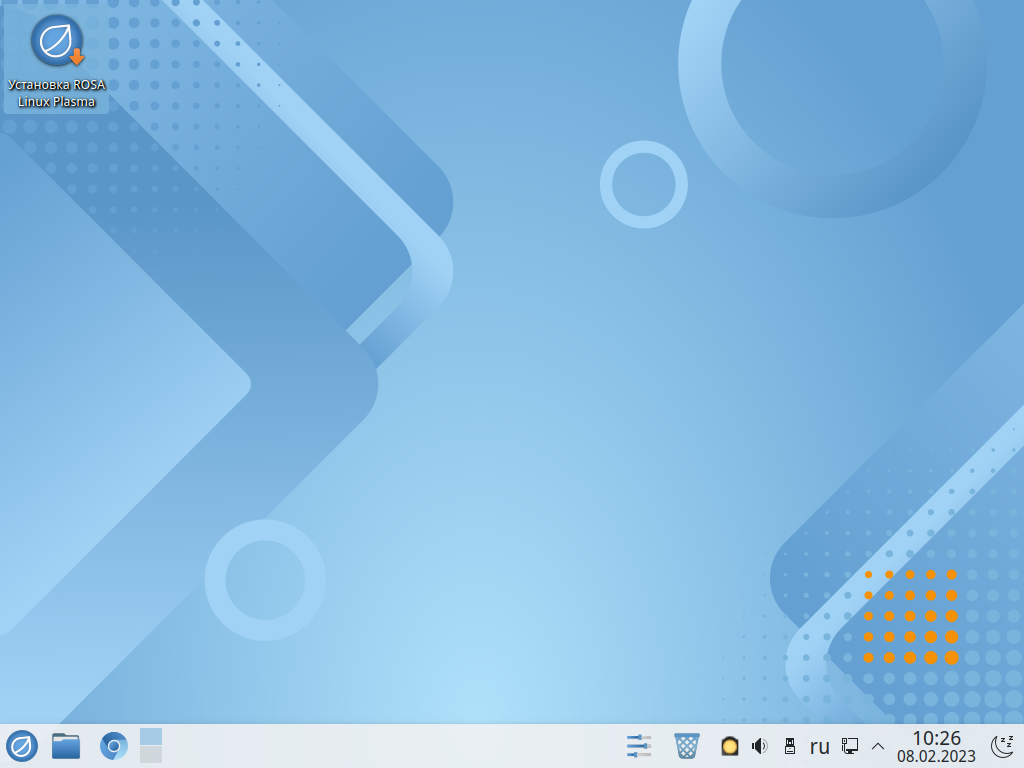


Рисунок 3 – Рабочий стол live-системы с ярлыком инсталлятора

Экран успешной проверки ОЗУ приведен ниже (рисунок 4Рисунок 4). Для выхода из режима проверки необходимо нажать клавишу Esc, при этом произойдет перезагрузка ПК.

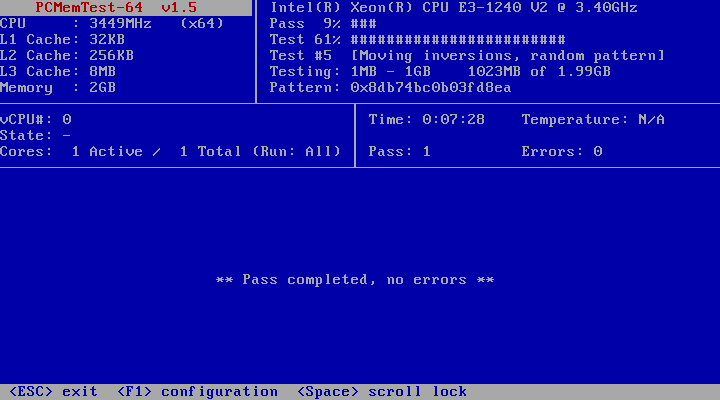


Рисунок 4 – Экран успешной проверки ОЗУ

1. Главный экран установщика ОС содержит семь разделов параметров, часть из которых заполнена автоматически, другая же часть (отмеченная красным цветом) требует вмешательства пользователя-установщика Системы (рисунок 5). Для неотмеченных красным разделов возможно тоже может понадобиться дополнительная настройка, например, при подключении ОС к локальной сети рекомендуется изменить значение "Имя узла" на на имя узла, соответствующее правилам вашей сети. В случае отсутствия подключения к сети или при ошибке автоопределения необходимо также правильно установить часовой пояс.

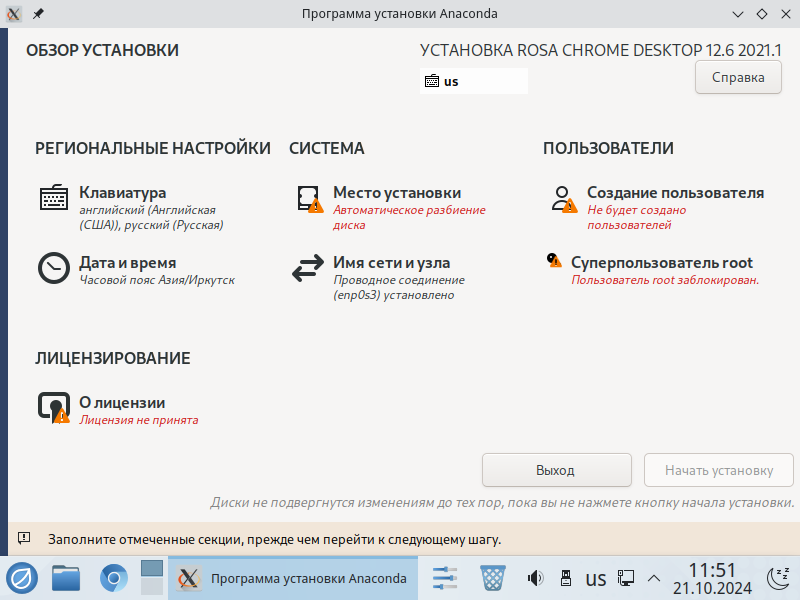


Рисунок 5 – Главный экран установщика Системы

1. Для разметки установочного накопителя используется раздел "Место установки" (рисунок 6).

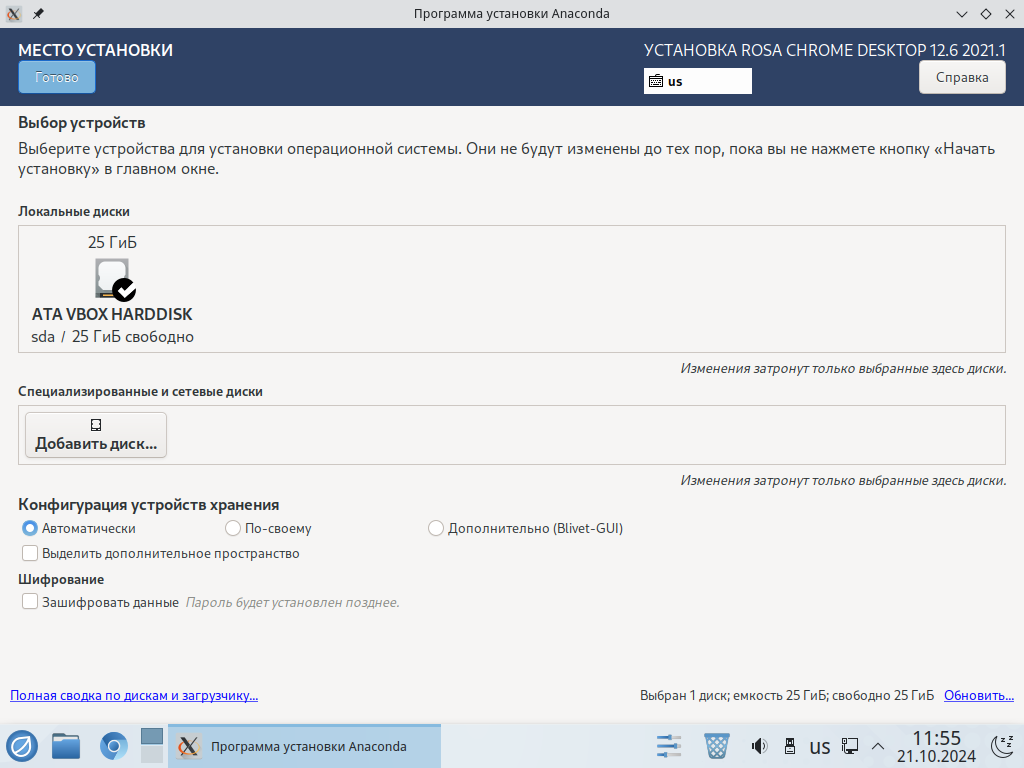


Рисунок 6 – Место установки системы

Если предполагается установка Системы на пустой носитель или же на выделенное на носителе пустое место, рекомендуется использовать автоматическую разметку. Для этого достаточно нажать на кнопку Готово*,* при этом, кроме технического раздела /boot/efi, будет создан отдельный корневой раздел под ОС и раздел под файлы пользователей /home, а также раздел swapдля расширения оперативной памяти. Стандартная разметка дает возможность быстрого восстановления работоспособности Системы при любых её отказах, для этого достаточно переустановить Систему в корневом разделе, не форматируя раздел с пользовательскими файлами.

По умолчанию ОС устанавливается на журналируемую ФС Ext4, но в инсталляторе также имеется возможность установки на Ext3, XFS и BTRFS. Кроме того, возможно форматировать и подключать диски с NTFS.

1. Для повышения защищенности информации есть возможность установить Систему на зашифрованный диск. Подробнее об установке системы с использованием средств маскирования рассмотрено в разделе 2.3.
2. Далее ввести учетные данные первого пользователя. По умолчанию он является администратором Системы, от его имени можно вводить консольные root-команды, используя sudo,и изменять критичные настройки Системы (рисунок 7). Подробнее о пользователях и администраторах см. раздел 3.2.

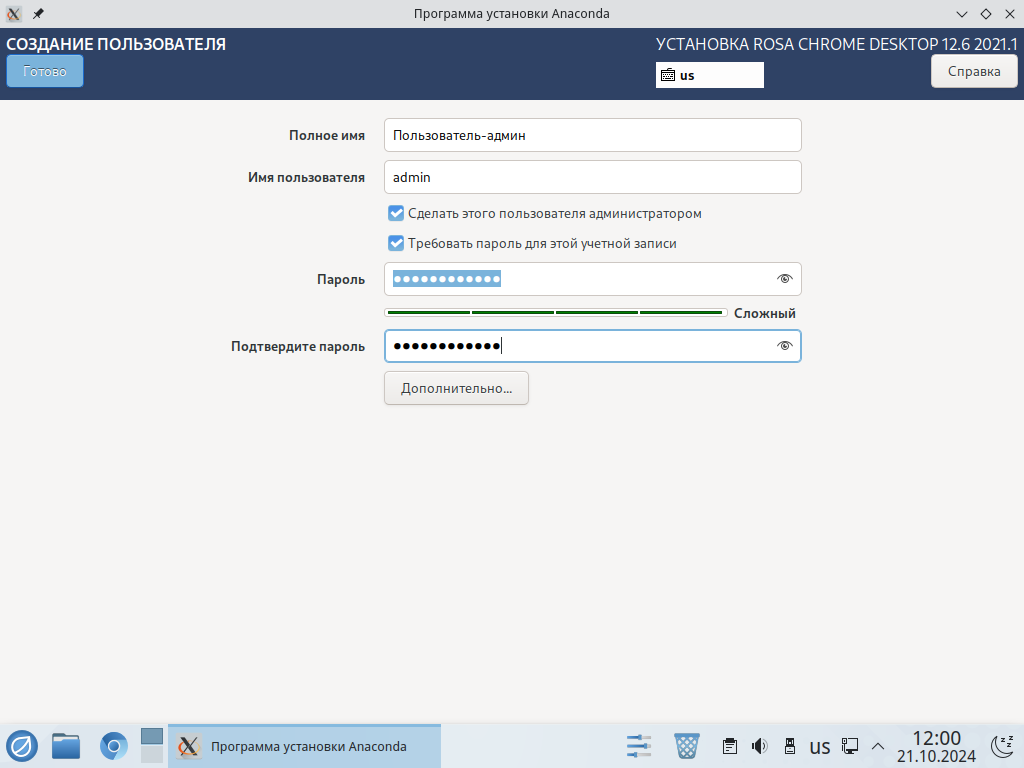


Рисунок 7 – Создание учетной записи пользователя

При вводе пароля контролируется его сложность: по длине, разнообразию знаков и по соответствию спискам часто применяемых паролей, но при этом сохраняется возможность использования небезопасных паролей при двойном нажатии на кнопку Готово*.*

1. Если при создании пользователя снять флажок "Сделать этого пользователя администратором"*,* то обязательным останется следующее поле "Суперпользователь root", в которое необходимо ввести пароль суперадминистратора. Такая возможность оставлена для совместимости, при этом вводимый по умолчанию пользователь-администратор с правом sudo считается более безопасным вариантом (подробнее об этом см. п. 3.2.1).
2. После того как все обязательные пункты для установки заполнены, можно начать собственно установку ОС на ПК нажатием на кнопку Начать установку на главном экране установщика (рисунок 5).
3. Установка ОС может занять некоторое время, по завершении которого отобразится соответствующее сообщение (рисунок 8). После чего необходимо выполнить перезагрузку ПК нажав кнопку Перезагрузка системы, в процессе которой необходимо извлечь установочный компакт-диск или флеш-накопитель.

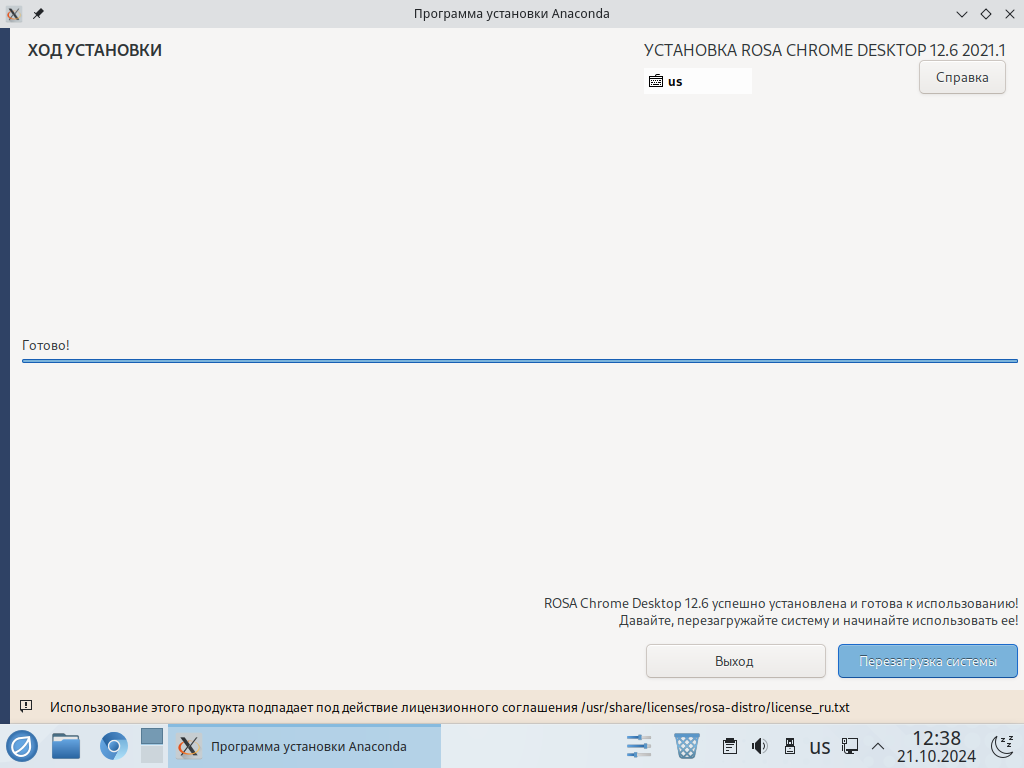


Рисунок 8 – Завершение процесса установки системы

1. После перезагрузки необходимо ввести (выбрать) логин и пароль учетной записи для выполнения входа в Систему. Если ОС была установлена на зашифрованные разделы, также понадобится ввести пароль для их расшифровки.
   * 1. Установка серверной версии ОС

Установка серверной версии ОС осуществляется полностью аналогично установке на рабочую станцию (п. 2.2.2), за исключением того, что у серверной версии отсутствует live-режим. В меню начальной загрузки следует выбрать пункт "Установка ROSA Linux"для установки без предварительной загрузки ОС.

* + 1. Парольная защита загрузчика ОС

Для повышения уровня безопасности возможно ограничить доступ к редактированию параметров загрузки, установив пароль на загрузчик GRUB2.

Этапы настройки пароля:

Генерация хеша пароля командой:

sudo grub2-mkpasswd-pbkdf2

Будет предложено ввести и подтвердить пароль. В результате отобразится хеш пароля, например:

Хэш PBKDF2 вашего пароля:

grub.pbkdf2.sha512.10000.7730DD2A5B6B5C31CD447BB9A8C313DC...

Создание конфигурационного файла с ограниченными правами доступа:

( umask 077 && touch /boot/grub2/user.cfg )

Запись сгенерированного хеша пароля в файл /boot/grub2/user.cfg:

echo "GRUB2\_PASSWORD=grub.pbkdf2.sha512.10000.7730DD2A5B6B5C31CD447BB9..." > /boot/grub2/user.cfg

Обновление конфигурации GRUB2 применением изменений:

sudo update-grub2

После выполнения данных действий при попытке редактирования записей загрузчика GRUB2 (например, при нажатии клавиши E в меню загрузки) потребуется ввод пароля. Имя пользователя — root, пароль соответствует ранее заданному при генерации хеша.

* 1. Установка с использованием средств маскирования информации
     1. Общие вопросы

В этом разделе будут рассмотрены только особенности установки ОС на компьютер пользователя с использованием средств маскирования информации на носителях. Так как термин "шифрование" используется в документации к защищенным ОС, только если это шифрование является сертифицированным для защиты информации, в данном документе для несертифицированных для применения в РФ алгоритмов и программ шифрования используется термин "маскирование информации". Такое маскирование не обеспечивает гарантированной защиты информации, но значительно затрудняет неавторизованный доступ к ней.

* + 1. Разметка носителя с маскированием информации

Для повышения защищенности информации есть возможность установить Систему с использованием маскирования информации, при этом разметку носителя можно осуществить автоматически, включив флажок "Зашифровать данные" или вручную.

Для режима разметки вручную в окне "Место установки" нужно выбрать в разделе "Конфигурация устройств хранения" переключатель "По-своему" и нажать на кнопку Готово, а потом разметить носитель с отдельным незашифрованным /boot (и /boot/efi для efi-BIOS). Корневой раздел Системы, swap-раздел и пользовательский раздел /homeнужно при этом зашифровать (замаскировать), установив соответствующий флажок в поле"Зашифровать данные", затем кнопки Применить и Готово(рисунок 9)*.*

Следует обратить внимание, что замаскированные разделы обозначены как LUKS-разделы, рекомендуется использовать систему маскирования luks2.

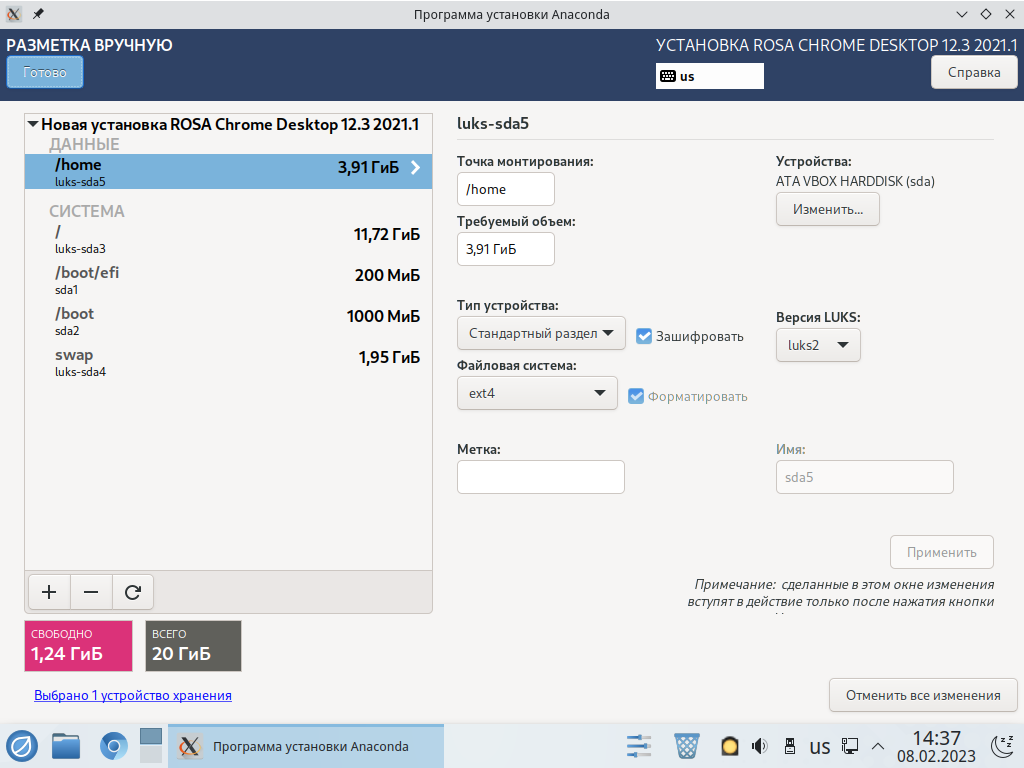


Рисунок 9 – Ручная разметка диска для использования маскирования

После этого можно выполнить установку в обычном режиме. При каждой загрузке Системы, использующей замаскированные разделы, у пользователя запрашивается пароль для расшифровки. Без ввода правильного пароля доступ к информации на носителях будет блокирован, причем эта блокировка сохраняется и при переносе носителя информации на другой ПК.

* + 1. Поддержка tpm2 для автоматического ввода паролей

Современные СВТ с архитектурой x86\_64 примерно с 2016-го года оснащаются встроенным модулем хранения конфиденциальной информации tpm2, который можно использовать для разблокировки замаскированного раздела LUKS. Пароль расшифровки маскированных разделов при этом будет храниться в tpm2 и вводиться автоматически. Пользовательская информация на изъятом из компьютера с tpm2 носителе будет недоступна при монтировании этого носителя на другом компьютере.

Для использования tpm2 в Системе, использующей разделы с маскированием информации, нужно ввести команду:

sudo luksunlock

и следовать дальнейшим инструкциям. Для возврата к ручному вводу паролей используется команда:

sudo lukslock

* 1. Установка на виртуальную машину

Образ ОС подготовлен для установки и работы на виртуальных машинах (далее – ВМ) в системах виртуализации ROSA Virtualization, VirtualBox, QEMU/KVM и ESXi VMWARE. В ОС предустановлены пакеты для поддержки совместимости (virtualbox-guest-addidtions, spice-vdagent и open-vm-tools), лишние из которых удаляются при установке в несовместимой среде.

Установка под управлением гипервизора должна осуществляться согласно документации к этому гипервизору с учетом минимальных требований ОС к аппаратному обеспечению (п.1.4).

* 1. Автоматизированное развертывание
     1. Введение в kickstart-сценарии

Автоматизированная установка позволяет выполнить полностью автоматическую установку в соответствии с преднастройками, может применяться для автоматического развертывания множества компьютеров одинаковой конфигурации.

Для автоматизированной установки создается файл с настройками – kickstart-сценарий – сценарий быстрого развертывания. Затем в параметры cmdline ядра записывается путь к этому файлу, по DHCP автоматически настраивается сеть, а модуль anaconda-dracut в initramfs (образ файловой системы, загружаемый в оперативную память вместе с ядром) загружает этот kickstart-сценарий и сохраняет в файле /run/install/ks.cfg.

При установке в любом из режимов, в том числе в интерактивном графическом, в установленной Системе по адресу /root/anaconda-ks.cfg сохраняется kickstart-сценарий, описывающий произведенную установку, который можно взять за основу для пользовательского сценария в дальнейшем.

* + 1. Типовой сценарий автоматизированной установки

Для реализации типового сценария автоматизированной установки нужно создать файл, например с именем a.ks, со следующим содержимым:

cmdline

# https://bugzilla.redhat.com/show\_bug.cgi?id=1874434

liveimg --url=file:///dev/mapper/live-base

# Автоматически принять лицензионное соглашение

eula --agreed

# System language

lang ru\_RU.UTF-8

# Keyboard layouts

keyboard --xlayouts=us,ru --switch=grp:alt\_shift\_toggle

# Network information

network --hostname=localhost.localdomain

# SELinux configuration

#selinux --disabled

# System services

services --enabled="chronyd"

autopart

# Partition clearing information

clearpart --all --initlabel

# System timezone

timezone Europe/Moscow --utc

# Passwords

rootpw 123456

user --groups=wheel,users --name=rosa --password=123456

%anaconda

pwpolicy root --minlen=6 --minquality=1 --notstrict --nochanges --notempty

pwpolicy user --minlen=6 --minquality=1 --notstrict --nochanges --emptyok

pwpolicy luks --minlen=6 --minquality=1 --notstrict --nochanges --notempty

%end

reboot --eject

Примечание – С документацией по kickstart-сценариям можно ознакомиться по адресу <https://pykickstart.readthedocs.io/en/latest/kickstart-docs.html>.

Код приведенного выше сценария состоит из следующих командных строк:

1. Выполняется полностью автоматизированная установка; запускается X-сервер, автоматически запускается установка, настройка не требуется (возможны варианты без X-сервера и установка по VNC):

cmdline

Копируется на накопитель Система из LiveCD (также возможна установка из сетевого репозитория):

liveimg --url=file:///dev/mapper/live-base

Устанавливается язык ОС ru\_RU.UTF-8 (в файле /etc/locale.conf):

lang ru\_RU.UTF-8

Добавляются в ОС две раскладки клавиатуры: us и ru, а переключение раскладок настраивается на комбинацию клавиш Alt+Shift:

keyboard --xlayouts=us,ru --switch=grp:alt\_shift\_toggle

Включается службa systemd chronyd (можно перечислить несколько служб через запятые без пробелов):

services --enabled="chronyd"

Пример отключения sshd:

services --disabled="sshd"

Выполняется полностью автоматизированная разметка диска с удалением всех данных (возможно прописать детализированные инструкции по разметке диска):

clearpart --all --initlabel

Устанавливается часовой пояс "Европа/Москва", а аппаратные часы (BIOS) будут выставлены по Гринвичу (рекомендуется):

timezone Europe/Moscow --utc

Устанавливается последовательность символов, например "123456", в качестве root-пароля:

rootpw 123456

Создается пользователь с именем, например "rosa", добавляется в группы wheel и users; устанавливается пароль пользователя, например "123456":

user --groups=wheel,users --name=rosa --password=123456

Примечание – Группа wheel – специальная группа пользователей, используемая в некоторых системах UNIX для контроля доступа к команде su, которая позволяет пользователю маскироваться под другого пользователя (обычно суперпользователя).

Пароли можно записать в зашифрованном виде, например, установить пароль пользователю root:

rootpw --iscrypted $6$BHWkSXIw6TgPrdvJ$V1ARg/yAP.kRe5mo5FawOOvf7SRm/GDM.Ly6RGGmSBR3Zc81/XzYlpoDnYKFMMPZodKBIVyBvjJ9evM1sey0E/

Хеш, приведенный выше, является результатом выполнения команды:

python3 -c "import crypt; print(crypt.crypt(input('clear-text pw: '), crypt.mksalt(crypt.METHOD\_SHA512)))"

Данная команда позволяет получить хеш введенного с клавиатуры пароля. Таким образом можно избежать хранения пароля в открытом виде. Например, хеш пароля "123456" выглядит следующим образом:

user --groups=wheel,users --name=rosa --password=$6$BHWkSXIw6TgPrdvJ$V1ARg/yAP.kRe5mo5FawOOvf7SRm/GDM.Ly6RGGmSBR3Zc81/XzYlpoDnYKFMMPZodKBIVyBvjJ9evM1sey0E/ --iscrypted

1. По завершении установки необходимо перезагрузить ПК (в процессе перезагрузки необходим установочный носитель):

reboot --eject

Примечание – При установке на ВМ рекомендуется использовать установку с ISO-образа, подключенного как DVD-диск, т.к. гипервизор умеет отключать виртуальный привод по команде извлечения диска из гостевой Системы: тогда после перезагрузки Система загрузится не с образа, а с диска с установленной ОС.

Параметры в cmline ядра и в kickstart-сценарии дополняют друг друга и могут использоваться в различных комбинациях.

* + 1. Дополнительные действия в kickstart-сценариях

В kickstart-сценарий могут быть добавлены произвольные действия, выполняемые в интерпретаторе shell после завершения основной установки. Это позволяет настраивать Систему сразу после инсталляции.

Для выполнения действий в установленной Системе (в chroot) используется секция %post.

Пример: запись даты и времени установки в файл /var/log/installtime.

%post

LC\_ALL=C date > /var/log/installtime

%end

Если необходимо выполнить действия вне установленной Системы (в окружении LiveCD), в секцию %post добавляется параметр --nochroot:

%post –nochroot

LC\_ALL=C date > "${INSTALL\_ROOT}/var/log/installtime"

%end

Дополнительно возможно загрузить внешние файлы с помощью утилит curl, wget и других средств сетевого доступа, при наличии доступа к локальному или внешнему веб-серверу.

Удаление и установка пакетов могут быть выполнены также внутри секции %post:

* Удаление ненужных пакетов:

%post

sudo dnf -q -y remove firefox-esr52

%end

* Установка дополнительных пакетов:

%post

dnf -q -y install sconfigs-detailed-audit

%end

Прямая установка пакетов через секцию %packages в данной конфигурации не поддерживается. Рекомендуется использовать утилиту dnf с ключом -q для подавления лишнего вывода, что повышает стабильность автоматизированной установки.

Для тестирования сценария автоматизированной установки можно воспользоваться встроенным HTTP-сервером на машине оператора:

1. Определить IP-адрес оператора:

ip a

1. Перейти в директорию с файлом сценария (например, a.ks) и запустить сервер:

python3 -m http.server

В качестве альтернативы возможно использование других веб-серверов, например nginx или Apache httpd.

1. На целевой машине при загрузке LiveCD в командную строку ядра необходимо добавить:

systemd.unit=anaconda.target inst.ks=http://192.168.1.173:8000/a.ks

где:

* 192.168.1.173 — IP-адрес машины оператора,
* 8000 — порт запущенного HTTP-сервера.

Также рекомендуется добавить параметр inst.text.

Данный параметр позволит запустить консольный режим установщика вместо графического. Установка будет произведена автоматически без участия оператора.

* 1. Сетевая загрузка с возможностью установки
     1. Технология Intel PXE

Возможность загрузки ОС по сети реализована с помощью стандартной технологии PXE, которая состоит из следующих этапов:

1. Встроенный в BIOS компьютера загрузчик PXE:

* получает IP-адрес по протоколу BOOTP (протокол, родственный DHCP);
* скачивает по протоколу TFTP с сервера TFTP файлы загрузчика, в т.ч. сам загрузчик и конфигурационный файл с описанием меню загрузки, и элементы vmlinuz и initrd в качестве минимальной версии ОС;
* запускает минимальную версию ОС с указанными в конфигурационном файле параметрами.

1. Минимальная версия ОС, запущенная на клиентском компьютере:

* скачивает указанные в параметрах запуска initrd установочный образ и дополнительные компоненты полной версии ОС по протоколу HTTP с установленного на сервере PXE веб-сервера, сохраняя их в оперативной памяти;
* запускает скачанную полную версию ОС на клиенте.

Таким образом, на клиентском компьютере запускается тот же установочный образ LiveCD, что и при загрузке с USB-накопителя или DVD-диска.

Следует учесть, что для сетевой загрузки ОС на клиентской системе требуется на 2 Гб оперативной памяти больше, чем для обычной установки.

* + 1. Начальная настройка сервера для работы PXE

Для сетевой загрузки и установки на рабочих станциях необходим сервер, с которого будут загружаться образы ОС, и локальная сеть, по которой будет происходить загрузка. В качестве сервера можно использовать как установленную локально или в виртуальной машине версию ОС для рабочей станции, так и серверную версию ОС.

Для того чтобы ОС могла работать в качестве сервера PXE, необходимо установить и настроить серверные пакеты. Для настройки понадобятся сервера DHCP и TFTP (простейший веб-сервер для загрузки по HTTP уже предустановлен), а также загрузчик syslinux:

dnf install dhcp-server tftp-server syslinux

* + 1. Настройка сервера DHCP (BOOTP)

Конкретная настройка DHCP-сервера специфична для каждой локальной сети. В качестве примера можно рассмотреть по шагам простейший случай:

1. установить на сервере PXE для текущего сетевого соединения статический IP-адрес 192.168.1.1 и маску подсети 255.255.255.0;
2. заменить содержимое файла настройки /etc/dhcpd.conf на следующее:

option domain-name "rosa.lan";

option domain-name-servers ns1.rosa.lan, ns2.rosa.lan;

default-lease-time 3600;

max-lease-time 7200;

authoritative;  
allow booting;

allow bootp;

filename "pxelinux.0";

subnet 192.168.1.0 netmask 255.255.255.0 {

option routers 192.168.1.1;

option subnet-mask 255.255.255.0;

option domain-search "rosa.lan";

option domain-name-servers 192.168.1.1;

range 192.168.1.10 192.168.1.100;

}

1. включить сервис DHCP командой:

sudo systemctl enable --now dhcpd

1. статус запущенного сервера (должен быть active) можно проверить командой:

systemctl status dhcpd

Правильность подключения можно проверить, войдя в сеть с клиентского компьютера: он должен получить от сервера DHCP IP-адрес 192.168.1.10.

* + 1. Настройка веб-сервера

В предустановленном на образах ОС языке Python есть простой веб-сервер, который можно использовать для тестовой PXE-установки (потребуется ISO-образ ОС), последовательно выполнив следующие команды:

1. создать папку для веб-сервера:

mkdir -p "$HOME/webserver"

1. перейти в папку $HOME/webserver:

cd "$HOME/webserver"

1. создать папку для ISO-образа:

mkdir -p ISO

1. монтировать папку для ISO-образа:

sudo mount -o loop "<путь к ISO-образу>" "$HOME/webserver/ISO"

1. запустить HTTP-сервер из папки:

python3 -m http.server

Далее необходимо скопировать в папку $HOME/webserver файл kickstart-сценария a.ks, создание которого описано в п.2.5.2.

Консоль, в которой будет работать веб-сервер, следует оставить открытой. Проверить правильность его настройки можно, открыв на компьютере браузер по адресу [http://192.168.1.1:8000](http://192.168.1.1:8000/)**.**

В браузере должны быть выведены файлы из директории веб-сервера, в том числе должны быть доступны папка ISO, ее содержимое, и файл kickstart-сценария a.ks, а в консоли веб-сервера должен быть выведен лог запросов к нему.

* + 1. Настройка сервера TFTP

Для настройки сервера TFTP нужно выполнить следующие действия:

1. скопировать в папку сервера TFTP компоненты первоначального загрузчика:

( cd /usr/share/syslinux && cp -v menu.c32 pxelinux.0 ldlinux.c32 libutil.c32 /var/lib/tftpboot/ )

1. скопировать минимальную версию ОС из ISO-образа в папку TFTP-сервера (от имени пользователя, но не root):

( cd "$HOME/webserver/ISO/isolinux" ; sudo cp -v initrd0.img vmlinuz0 /var/lib/tftpboot/ )

1. создать папку для дополнительных конфигурационных файлов TFTP-сервера:

mkdir -p /var/lib/tftpboot/pxelinux.cfg

1. открыть на редактирование файл /var/lib/tftpboot/pxelinux.cfg/default (будет создан, если не существует):

nano /var/lib/tftpboot/pxelinux.cfg/default

со следующим содержимым:

default menu.c32

prompt 0

timeout 300

ONTIME local

menu title #================ PXE BOOT ===============#

label 1

menu label ^1) Automatically install ROSA

kernel /vmlinuz0

append initrd=/initrd0.img systemd.unit=anaconda.target enforcing=0 root=live:http://192.168.1.1:8000/ISO/LiveOS/squashfs.img inst.text inst.ks=http://192.168.1.1:8000/a.ks

label 2

menu label ^2) Install ROSA

kernel /vmlinuz0

append initrd=/initrd0.img enforcing=0 root=live:http://192.168.1.1:8000/ISO/LiveOS/squashfs.img

Для сохранения данных и выхода из редактора nano нажимают клавиши Ctrl+O и Ctrl+X соответственно.

В строках, начинающихся с append, определены параметры cmdline ядра для вариантов загрузки в меню. В первом случае запускается автоматизированная консольная установка по kickstart-сценарию, а во втором случае запускается LiveCD. При отсутствии выбора со стороны пользователя какого-либо пункта меню в течение 300 секунд будет произведена загрузка первого пункта.

Следует обратить внимание, что приведенный выше kickstart-сценарий автоматически стирает весь диск, т.е. в приведенной конфигурации, если не выбрать второй пункт меню, будет произведена автоматическая установка с форматированием диска. При запуске ОС по PXE можно выбрать нужный пункт меню. При необходимости можно адаптировать меню под свои потребности (дополнительная информация доступна по ссылке <https://wiki.syslinux.org/wiki/index.php?title=PXELINUX>).

1. запустить и добавить в автозагрузку TFTP-сервер:

sudo systemctl enable --now tftp

1. выполнить настройки BIOS клиентской машины на загрузку по сети (PXE) и произвести загрузку.
   * 1. Настройка сети в LiveCD

По умолчанию при запуске LiveCD, включая режим работы исключительно установщика ОС, настройка сети осуществляется автоматически по протоколу DHCP.

В случае необходимости задания статических сетевых параметров соответствующая конфигурация может быть передана в ядро через параметр cmdline, используя синтаксис, поддерживаемый модулем dracut network-legacy.

Пример задания параметров сети вручную:

ip=192.168.122.50::192.168.122.1:255.255.255.0:::off

где:

* 192.168.122.50 — IP-адрес сетевого интерфейса;
* 192.168.122.1 — шлюз по умолчанию;
* 255.255.255.0 — маска подсети;
* off — отключение автоматической настройки через DHCP.

Дополнительные сведения о параметрах и форматах можно найти на справочной странице dracut.cmdline(7).

* 1. Графическая установка с использованием VNC

Графическая установка с применением технологии VNC (Virtual Network Computing) позволяет запускать интерфейс установщика ОС на удалённой машине оператора. Этот способ особенно полезен в случаях, когда установка производится на удалённый компьютер, не имеющий локальных устройств ввода и отображения (монитор, клавиатура, мышь), либо физический доступ к нему затруднён.

Примечание – При использовании установки через VNC запускается встроенный VNC-сервер, который может быть доступен без фильтрации трафика и, в зависимости от конфигурации, без пароля. Не рекомендуется запускать установку по VNC в незащищённых или общедоступных сетях без дополнительных мер безопасности.

Необходимые условия для выполнения установки через VNC:

* целевая машина должна иметь установленный VNC-клиент;
* целевая машина, на которую производится установка ОС, должна быть доступна по сети и поддерживать загрузку с LiveCD или по PXE;
* сеть должна быть настроена по DHCP или вручную, с обеспечением двусторонней сетевой доступности между клиентом и целевым компьютером.

Установка ОС через VNC выполняется в несколько этапов:

1. Установка клиента TigerVNC на целевом ПК командой:

sudo dnf install /usr/bin/vncviewer

1. Запуск клиента TigerVNC в режиме ожидания входящего подключения на целевом ПК:

vncviewer -listen 4444

где 4444 — произвольно выбранный свободный порт, который будет использоваться для подключения.

1. Получение IP-адреса целевого компьютера, на котором выполняется vncviewer, командой:

/sbin/ip a

1. При загрузке целевого компьютера с LiveCD или по сети (PXE-загрузка) необходимо изменить параметры командной строки ядра. В режиме ввода параметров загрузки (boot prompt) добавить следующее:

systemd.unit=anaconda.target inst.vnc\ inst.vncconnect=192.168.1.173:4444

где:

* 192.168.1.173 — IP-адрес целевого компьютера;
* 4444 — порт, указанный при запуске vncviewer.

1. Завершение установки и запуск графического интерфейса. После загрузки и автоматической настройки сети установщик подключится к оператору по указанному адресу. На целевом компьютере автоматически откроется окно графического установщика ОС РОСА "ХРОМ", идентичное локальному интерфейсу установки. Можно приступить к стандартному графическому процессу установки (об этом подробнее см. раздел 2.2 Локальная установка), управляя им удалённо.
   1. OEM-установка

OEM-установка предназначена для массового развёртывания ОС на однотипные компьютеры. В этом методе на жёсткий диск целевой машины побайтово копируется ISO-образ, а не клон уже установленной Системы. После копирования машины могут быть переданы конечным пользователям без дополнительной настройки.

При первом запуске администратор подтверждает установку ОС, после чего у него отсутствует возможность вмешаться в процесс. После завершения установки будет предложено создать пользователей и подтвердить лицензионное соглашение.

Преимущества OEM-установки:

* размер образа для OEM-установки равен размеру стандартного ISO, что значительно меньше по сравнению с клон-образом уже установленного диска;
* после подтверждения пользователем установки исходный ISO-образ остаётся на первом разделе, что позволяет при неповреждённой файловой системе вернуть Систему к состоянию "первого старта";
* создание ISO-образа для OEM-установки происходит быстрее и проще, что снижает вероятность ошибок.

OEM-образ формируется в два этапа:

1. Сборка образа на ABF.Необходимо, чтобы в сборке присутствовал пакет oem-install. С 2025 года данный пакет включается во все стандартные образы ОС РОСА "ХРОМ". Он не мешает установке через Anaconda и будет автоматически удалён на этапе initial-setup.
2. Пересборка с помощью скрипта oem-iso-rebuilder**.** Основная задача этого скрипта — заменить конфигурационный файл загрузчика GRUB2 в ISO-образе, чтобы при загрузке запускался oem-install вместо Live OS или Anaconda. Кроме того, скрипт меняет таблицу разделов на GPT и даёт возможность добавить файлы, влияющие на поведение OEM-установки.

Пример пересборки:

sudo oem-iso-rebuilder\ ./ROSA\_2021.1\_PLASMA5\_x86\_64\_53042.iso

После запуска утилита предложит приостановить выполнение, чтобы добавить собственные конфигурационные файлы. По завершении будет получен новый образ с расширением .oem.iso.

Установка OEM-образа производится побайтовым копированием ISO-образа на носитель, для этого можно использовать утилиту dd:

dd if=./ROSA\_2021.1\_PLASMA5\_x86\_64\_53042.oem.iso of=/dev/sda bs=4M

При пересборке ISO допускается добавить в его корень следующие файлы:

* oem-install.cfg — конфигурация разметки дисков, например:

PARTITONS=btrfs

ROOTSIZE=70000

SWAPSIZE=10000

EFISIZE=200

Подробнее обо всех параметрах можно узнать, выполнив команду:

/usr/libexec/oem-install --help

* anaconda-ks.cfg — kickstart-файл, который обычно создаётся Anaconda, будет скопирован в /root установленной Системы;
* oem-install.css — стили интерфейса GUI для oem-install, шаблон доступен в /usr/share/oem-install/oem-install.css.
  1. Установщик iso2img

Утилита iso2img представляет собой простой скриптовый инструмент, позволяющий в один шаг установить ОС РОСА "ХРОМ" из ISO-образа в образ виртуальной машины (qcow2, vmdk, vdi и др.). При этом будет создан пользователь rosa с паролем rosa и правами администратора.

Установка и использование утилиты iso2img:

sudo dnf install iso2img

sudo iso2img Rosa.iso

В результате будет получен, например, файл Rosa.qcow2, который можно запустить напрямую:

qemoo Rosa.qcow2

Для получения дополнительных параметров необходимо обратиться к справке через команду:

iso2img --help

1. Общие сведения о работе в Системе
   1. Интерфейсы Системы

Система поддерживает два основных режима работы пользовательского окружения: текстовый консольный интерфейс (виртуальные терминалы TTY) и графический интерфейс (графический сервер с рабочим столом и менеджером входа).

**Серверный вариант ОС** по умолчанию загружается в текстовый консольный интерфейс TTY.При необходимости допускается установка графической оболочки через менеджера пакетов dnf. Для установки рабочего стола KDE Plasma следует выполнить команду:

sudo dnf install task-iso-plasma5

После установки рекомендуется включить менеджер входа, выполнивкоманду:

sudo systemctl enable --now gdm

Для немедленного перехода в графический режим может быть использована команда

sudo systemctl isolate graphical.target

Для назначения графического режима в качестве режима загрузки по умолчанию следует выполнить:

sudo systemctl set-default graphical.target

Возврат к консольному режиму задаётся командой:

sudo systemctl set-default multi-user.target

**ОС рабочей станции** по умолчанию запускается с графическим столом KDE Plasma 5 и менеджером входа GDM. Для работы с командной строкой в графическом режиме используется эмулятор терминала. В ОС он запускается через "Меню приложений → Konsole".

Взаимодействие с Системой осуществляется через оболочку командной строки (shell) GNU Bash.

При входе в консоль отображается стандартное приглашение командной строки (prompt):

user@hostname ~ $

где:

* user — имя текущего пользователя;
* hostname — имя хоста, формируемое по умолчанию случайным образом;
* @ — разделитель;
* ~ — текущий каталог (в Bash символ тильды соответствует домашнему каталогу /home/user);
* $ — символ окончания приглашения для обычного пользователя;
* # — символ окончания приглашения, указывающий на выполнение команд от имени суперпользователя root.

В Системе доступны двенадцать виртуальных терминалов (TTY). Переключение между ними выполняется комбинациями клавиш Ctrl+Alt+Fn, где n — номер терминала от 1 до 12 (например, Ctrl+Alt+F1 — первая консоль, Ctrl+Alt+F12 — двенадцатая).

* 1. Администраторы и пользователи

По умолчанию при установке ОС первый, созданный в инсталляторе, пользователь является пользователем-администратором Системы

Как типичная Linux-система, ОС является многопользовательской. Администратор может управлять учетными записями других пользователей с помощью консольного интерфейса.

Также для организации работы пользователей в ОС используется объединение пользователей в группы для обеспечения доступа к ресурсам. Пользователи могут принадлежать одной или нескольким группам.

* + 1. Использование команды sudo

Вход в Систему от имени суперпользователя root по умолчанию заблокирован. Для повышения полномочий пользователя до административных рекомендуется использовать команду sudo с вводом пароля Администратора Системы. Первый созданный в Системе пользователь-администратор по умолчанию имеет право на использование этой команды и повышения своих прав.

Команда sudo (superuser do) позволяет временно получить права суперпользователя для выполнения одной команды. Этот подход обеспечивает дополнительный уровень безопасности, так как пользователи не работают постоянно под учетной записью root, что снижает риски случайного или вредоносного изменения Системы.

Пример базового использования команды sudo, в котором пользователь вводит необходимую команду, требующую прав суперпользователя, предварив её ключевым словом sudo:

sudo команда

Если пользователь входит в группу, имеющую соответствующие права, Система запросит ввод пароля, после чего команда будет выполнена с правами администратора.

Для настройки прав доступа к sudo используется конфигурационный файл /etc/sudoers, который определяет, какие пользователи или группы могут использовать sudo и какие команды они могут выполнять. Редактирование этого файла должно происходить с использованием команды visudo, которая предотвращает ошибки при сохранении и синтаксические ошибки.

Для добавления пользователей в группу sudoers:

usermod -aG wheel username

Пользователи группы wheel по умолчанию имеют право выполнять команды с использованием sudo.

В целях безопасности рекомендуется ограничить список пользователей, имеющих доступ к команде sudo. Кроме того, можно настроить специальные правила для выполнения только определенных команд.

Пример команды, которая разрешает пользователю user выполнять только команду обновления Системы:

user ALL=(ALL) NOPASSWD: /usr/bin/dnf update

Все команды, выполненные с использованием sudo, записываются в системный журнал. Логи можно просмотреть в файле /var/log/auth.log или /var/log/secure. Это позволяет отслеживать действия пользователей, имеющих права суперпользователя, и выявлять возможные попытки несанкционированного доступа.

Несколько практических рекомендаций по использованию sudo:

* sudo не используется для выполнения команд, не требующих привилегий суперпользователя;
* всегда проверяйте команду перед её выполнением с правами root;
* правила использования sudo должны быть настроены так, чтобы пользователи могли выполнять только те действия, которые необходимы для их работы.

Подробное описание всех команд приведено в man sudo.

* + 1. Управление учетными записями пользователей

Рассмотрим основные аспекты по управлению учетными записями пользователей в Системе.

1. Управление пользователями:

* Создание пользователя и его профиль в /home:

sudo useradd -m <имя\_пользователя>

* Установка пароля для нового пользователя:

sudo passwd <имя\_пользователя>

* Удаление пользователя и связанных с ним данных:

sudo userdel <имя\_пользователя>

Если необходимо удалить также домашний каталог пользователя, используется опция -r:

sudo userdel -r <имя\_пользователя>

1. Управление группами:

* Создание новой группы пользователей:

sudo groupadd <название\_группы>

Также новую группу можно создать путем редактирования файла /etc/group и ввода необходимой информации о группе. Каждой группе присваивается уникальный идентификационный номер и ОС при работе учитывает номер группы, а не имя. Поэтому, при присвоении двум группам одинакового номера, ОС будет воспринимать две группы как одну и ту же. Описание команды и ее параметров приведено в man addgroup.

Информация о группах содержится в файле /etc/group в следующем формате:

Admin :: 21: user1, user2, user3

где:

* Admin — имя группы,
* 21 — идентификатор,
* user1, user2, user3 — члены группы.

Пользователь может состоять в нескольких группах и переходить из одной в другую в процессе работы. Описание файла /etc/group приведено в man group.

* Добавление пользователя в группу:

sudo usermod -aG <название\_группы> <имя\_пользователя>

* Удаление пользователя из группы:

sudo gpasswd -d <имя\_пользователя> <название\_группы>

1. Управление паролями:

* Смена пароля. Пользователь может изменить свой пароль с помощью команды:

passwd

Администратор может изменить пароль любого пользователя с помощью команды:

sudo passwd <имя\_пользователя>

* Истечение срока действия пароля. Для настройки срока действия пароля используется команда:

sudo chage -M <количество\_дней> <имя\_пользователя>

1. Управление пользовательскими сессиями:

* Просмотр списка активных пользователей в Системе:

who

* Завершение сессии пользователя

sudo pkill -KILL -u <имя\_пользователя>

1. Проверканеудачных попыток входа в Систему:

Просмотр информации о неудачных попытках входа с помощью команды:

faillog

Информация о таких попытках входа хранится в файле /var/log/faillog. Данная команда также позволяет управлять счетчиком неудачных попыток входа и устанавливать ограничения на их количество. При запуске команды faillog без дополнительных опций, она выводит записи только тех пользователей, у которых были неудачные попытки входа.

Для каждой учетной записи предусмотрен лимит в 10 попыток входа. Чтобы обнулить счетчик неудачных попыток, следует использовать опцию -r.

Более подробное описание команды и структуры файла /var/log/faillog можно найти в справочных страницах по команде

man faillog

* + 1. Восстановление забытого пароля пользователя

В случае утраты пароля к локальной учётной записи, включая пользователя root, восстановление доступа возможно с использованием Live-образа дистрибутива. Порядок действий по восстановлению пароля пользователя в ОС:

1. Подготовка загрузочного носителя:
   1. загрузка ISO-образа дистрибутива ОС (или совместимого Linux-дистрибутива);
   2. запись образ на USB-накопитель или DVD-диск;
   3. загрузка Системы в режиме Live (без установки на диск).
2. Определение раздела с установленной Системой:
   1. необходимо открыть программу GParted (в Live-образах ОС РОСА "ХРОМ" она включена по умолчанию, подробнее о работе с программой см. раздел 3.7.3). При использовании другого дистрибутива установить GParted по необходимости;
   2. определить раздел с установленной ОС: это раздел с точкой монтирования, например, /dev/sda2 (рисунок 10).

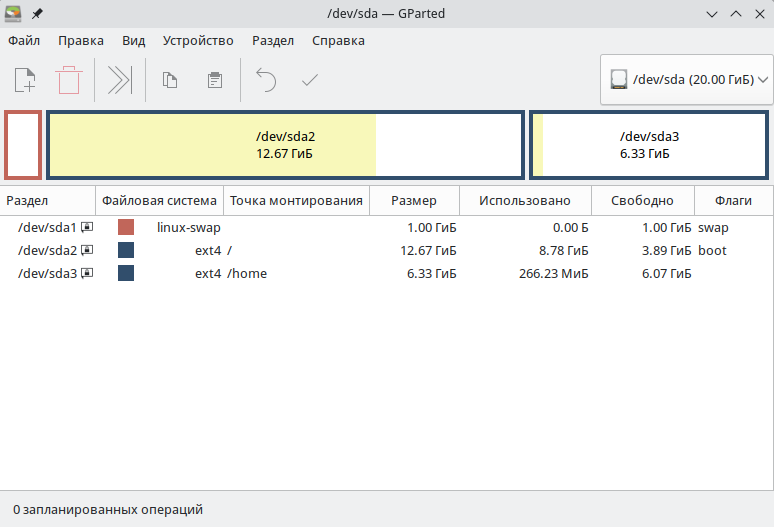


Рисунок 10 – Интерфейс программы GParted

1. Монтирование установленной Системы и вход в chroot:
   1. открыть терминал;
   2. создать директорию для монтирования:

sudo mkdir /chroot

* 1. смонтировать корневой раздел установленной Системы:

sudo mount /dev/sdXY /chroot

где необходимо заменить sdXY на имя раздела с точкой монтирования, определённое ранее (например, sda2);

* 1. перейти в окружение установленной Системы с помощью chroot:

sudo chroot /chroot

1. Сброс пароля:

* для смены пароля пользователя root используется:

passwd

* В запросах терминала необходимо ввести и подтвердить новый пароль.
* для смены пароля другого пользователя:

passwd <имя\_пользователя>

Если имя пользователя неизвестно, его можно определить одной из команд:

* просмотр содержимого директории /home;
* вывод всех пользователей:

cut -d: -f1 /etc/passwd

1. Завершение работы:
   1. завершить сеанс chroot, выйти из Live-режима и перезагрузить компьютер;
   2. при следующей загрузке войти в Систему с новым паролем.
   3. Команды

В консольном интерфейсе для получения подробной справки о любой команде нужно выполнить man <команда>**,** например:

man mc

Для текстовых команд в консоли доступно автодополнение по клавише Tab, например, если ввести "справка" и нажать Tab, то будут показаны все аргументы команды справка. Если же после команды справкаввести через пробел букву"ф"и нажатьTab**,** то автоматически сгенерируется параметр **"**файл"(рисунок 11)**.**

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 11 – Получение справки об операциях с файлами через консоль

Далее рассмотрим ключевые команды, необходимые для администрирования и работы с Системой.

* Информация о Системе:
* uname -a — выводит информацию о ядре и Системе;
* hostnamectl — показ информации о хосте и Системе;
* lscpu — отображение информации о процессоре;
* lsblk — показ подключенных дисков и их разделов;
* Управление пакетами:
* sudo dnf install <пакет> — установка пакетов с помощью пакетного менеджера dnf;
* sudo dnf update — обновление Системы;
* sudo dnf remove <пакет> — удаление пакета;
* rpm -q <пакет> — проверка установленного пакета;
* Управление сервисами (**systemd**):
* sudo systemctl start <сервис> — запуск сервиса;
* sudo systemctl stop <сервис> — остановка сервиса;
* sudo systemctl restart <сервис> — перезапуск сервиса;
* sudo systemctl status <сервис> — проверка статуса сервиса;
* sudo systemctl enable <сервис> — включение автозапуска сервиса при старте Системы;
* sudo systemctl disable <сервис> — отключение автозапуска сервиса;
* Работа с файлами и каталогами:
* ls -la — просмотр содержимого каталога с подробной информацией;
* cp <файл> <путь> — копирование файла;
* mv <файл> <путь> — перемещение или переименование файла;
* rm <файл> — удаление файла;
* mkdir <каталог> — создание нового каталога;
* rmdir <каталог> — удаление пустого каталога;
* Мониторинг Системы:
* top или htop — отображение активных процессов и использование ресурсов;
* df -h — информация о свободном и занятом месте на дисках;
* du -sh <каталог> — подсчет занимаемого места каталогом;
* free -h — показ использования оперативной памяти;
* Сетевые команды:
* ip addr — информация о сетевых интерфейсах;
* ping <хост> — проверка доступности удаленного узла;
* traceroute <хост> — маршрут до удаленного узла;
* netstat -tuln или ss -tuln — показ открытых портов и прослушиваемых сервисов;
* Работа с файлами и архивами:
* tar -cvf <архив.tar> <папка> — архивация папки;
* tar -xvf <архив.tar> — извлечение содержимого архива;
* gzip <файл> — сжатие файла;
* gunzip <файл.gz> — распаковка файла;
* Управление пользователями и правами:
* sudo useradd <имя> — создание нового пользователя;
* sudo passwd <имя> — изменение пароля пользователя;
* sudo usermod -ag <группа> <имя> — добавление пользователя в группу;
* sudo chown <пользователь>:<группа> <файл> — изменение владельца и группы файла;
* sudo chmod 755 <файл> — изменение прав доступа к файлу;
* Журналы:
* journalctl -xe — просмотр системных журналов;
* journalctl — просмотр всех записей журнала с момента последней загрузки Системы;
* journalctl -f — просмотр последних записей в реальном времени;
* dmesg — журнал ядра (информация о загрузке и драйверах).
  1. Системный менеджер systemd

Для управления сервисами в Системе используется системный и сервисный менеджер systemd. systemd отвечает за управление процессами, службами и различными системными компонентами.

К основным задачам systemd относятся:

* инициализация системы при запуске. systemd запускает все необходимые службы и процессы во время старта Системы;
* управление всеми системными службами;
* мониторинг состояния системы: сбор и предоставление информации о текущем состоянии всех служб и ресурсов системы;
* управление зависимостями между службами, что гарантирует корректный порядок их запуска.

Systemd — это системный и сервисный менеджер, обеспечивающий инициализацию Системы, запуск и контроль служб, управление зависимостями и сбор журналов. Systemd применяется на всех этапах жизненного цикла службы: от автоматического запуска при старте до мониторинга состояния и перезапуска при сбоях.

Под "сервисами" в контексте Системы и systemd понимаются службы (daemon), то есть компоненты, выполняющиеся в фоновом режиме и предоставляющие функциональность Системе или приложениям.

Сервисы описываются юнитом типа .service и управляются по жизненному циклу: запуск, остановка, перезапуск, перезагрузка конфигурации, проверка состояния.

Основные задачи systemd:

* Инициализация Системы при загрузке и параллельный запуск необходимых компонентов;
* Управление службами: запуск, остановка, перезапуск, загрузка конфигурации, контроль зависимостей и порядка инициализации;
* Мониторинг состояния: определение статусов active, inactive, failed и др., автоматический перезапуск по политике;
* Журналирование и диагностика: сбор логов через компонент journald и предоставление доступа к журналам;
* Управление зависимостями и целями: формирование целевых состояний (targets), гарантия корректной последовательности запуска.

Все команды управления службами systemd выполняются через утилиту systemctl.

Для диагностики применяется просмотр журналов через journald; рекомендуется использовать подписанные и проверенные конфигурации юнитов и избегать ручного изменения файлов в /usr/lib/systemd/system, выполняя переопределения в /etc.

* + 1. Управление службами

Команда systemctl является основным инструментом для взаимодействия с systemd. Она используется для запуска, остановки, перезагрузки и мониторинга служб.

Основные команды по управлению сервисами в ОС:

* Запуск сервиса:

sudo systemctl start <имя\_сервиса>

* Остановка сервиса:

sudo systemctl stop <имя\_сервиса>

* Перезапуск сервиса:

sudo systemctl restart <имя\_сервиса>

* Проверка статуса сервиса:

systemctl status <имя\_сервиса>

* Включение автозапуска сервиса при старте Системы:

sudo systemctl enable <имя\_сервиса>

* Отключение автозапуска сервиса при старте Системы:

sudo systemctl disable <имя\_сервиса>

* Перезагрузка с целью применения изменений конфигурации без перезапуска служб:

sudo systemctl reload <service\_name>

* Управление группами служб, например, можно запускать или останавливать группу служб:

sudo systemctl start <group\_name>

sudo systemctl stop <group\_name>

Фильтрация вывода systemctl возможна по различным критериям, таким как состояние службы с использованием опции--state или тип с использованием опции–type.

* + 1. Unit-файлы

Unit-файлы являются основными конфигурационными файлами, которые используются в systemd для управления службами, монтированием файловых систем, сокетами и другими системными объектами. Именно они определяют, как и когда должны запускаться процессы, как обрабатывать их зависимости, и другие аспекты работы службы или ресурса.

Существуют различные типы unit-файлов:

* service (расширение .service) – определяет параметры для управления службами;
* target (расширение .target) – группирует несколько unit-файлов и управляет их совместным запуском;
* socket (расширение .socket) – управляет сокетами, которые используются для активации служб;
* mount (расширение .mount) – управляет точками монтирования файловых систем;
* timer (расширение .timer) – предназначены для создания и управления таймерами, выполняют роль планировщика задач.

Unit-файлы имеют следующую базовую структуру:

* [Unit] – описание метаданных и зависимостей;
* [Service]/ [Timer] (или другой секции в зависимости от типа unit-файла) – определяет параметры запуска и управления процессом;
* [Install] – настройки, связанные с автозапуском и включением unit-файла в зависимости от системных целей (targets).

Настройки параметров unit-файлов помогают детально управлять поведением и запуском различных системных служб и компонентов. Алгоритм написания и настройки unit-файлов в systemd:

1. **Секция [Unit]** задает общие сведения о юните и его зависимости:

* Description= – краткое описание юнита;
* Wants= – указывает на необязательную зависимость: основной юнит пытается запустить другой юнит, указанный в параметре. Ошибки при запуске зависимого юнита не мешают запуску основного. Если параметры After= и Before= не указаны, юниты запускаются параллельно;
* Requires= – обозначает обязательную зависимость: основной юнит требует запуск другого юнита. Если запуск зависимого юнита неудачен, основной юнит также не запустится. Без параметров After= и Before= оба юнита будут запущены одновременно;
* After= – указывает, что основной юнит должен запускаться только после другого, указанного в параметре. Этот параметр часто используется вместе с Wants= и Requires=;
* Before= – противоположен After=, определяет запуск основного юнита перед другим юнитом.

1. **Секция [Install]** используется для настройки автозапуска юнита:

* Alias= – позволяет задать дополнительные имена для юнита, при этом создаются символические ссылки на основной юнит. Поддерживается не всеми типами юнитов;
* WantedBy= – указывает на цель, с которой связан данный юнит; при активации юнита команда systemctl enable создаст символическую ссылку в соответствующем каталоге;
* Also= – перечисляет другие юниты, которые также будут добавлены в автозапуск или удалены из него при изменении состояния текущего юнита.

1. **Секция [Service]** задает параметры управления службами:

* Type= – определяет, как служба будет запускаться:
* simple – запускает процесс немедленно, без создания дочерних процессов;
* forking – запускает службу, которая создает дочерний процесс и завершает родительский. Часто используется для демонов;
* oneshot – для одноразовых задач, которые завершаются после выполнения;
* notify – аналогично simple, но с дополнительным уведомлением о готовности;
* dbus – служба считается готовой, когда определенное имя появляется на шине D-Bus;
* idle – откладывает запуск службы до завершения всех других задач;
* PIDFile= – путь к PID-файлу;
* WorkingDirectory= – рабочий каталог, в котором будет работать процесс;
* User= – пользователь, от имени которого запускается служба;
* Group= – группа, от имени которой запускается служба;
* OOMScoreAdjust= – регулирует приоритет процесса при нехватке памяти;
* ExecStop= – команда для остановки службы;
* ExecStart= – команда, которая выполняется при запуске службы;
* RemainAfterExit= – указывает, что служба считается активной даже после завершения.

1. **Секция [Socket]** управляет параметрами сокетов:

* ExecStart= – команда для запуска;
* ExecReload= – команда для перезапуска;
* KillMode= – режим завершения;
* Restart= – правила перезапуска в случае ошибки.

Пример unit-файла службы:

[Unit]

Description=Пример службы

After=network.target

[Service]

ExecStart=/usr/bin/example-service

Restart=on-failure

[Install]

WantedBy=multi-user.target

где параметры:

* Description= предоставляет краткое описание службы;
* After= указывает, что служба должна быть запущена после достижения определенной цели (например, network.target);
* ExecStart= определяет команду, которую необходимо выполнить для запуска службы;
* Restart= управляет политикой перезапуска, в данном случае on-failure означает, что служба перезапустится при её сбое;
* WantedBy= определяет, в каких режимах работы (targets) Система должна запускать эту службу.

Различные unit-файлы располагаются в следующих каталогах Системы:

* /etc/systemd/system/ или /lib/systemd/system/ – для системных файлов. Файлы в этом каталоге имеют приоритет перед другими;
* /usr/lib/systemd/system/ – для пользовательских unit-файлов;
* /run/systemd/system/ – для временных unit-файлов, которые генерируются во время работы Системы.
  + - 1. Планирование задач с файлами .timer

Файлы с расширением .timer предназначены для создания и управления таймерами. В ОС они выполняют роль планировщика задач, являясь частью systemd и заменяя классический cron.

Файлы с расширением .timer работают в паре с .service-файлами, которые описывают действия, которые должны быть выполнены в запланированное время. Таймеры активируют определённые сервисы в моменты, заданные в конфигурации .timer-файла.

Файлы с расширением .timer имеют следующую основную структуру:

[Unit]

Description=Описание таймера

Requires=my.service #Запуск таймера, только при наличии соответствующего сервис файла “my.service”

[Timer]

OnCalendar=\*-\*-\* 02:00:00 # Указание расписания (пример: каждый день в 2:00)

Persistent=true # Выполнять задачу после пропуска, если Система была выключена

Unit=my.service #Таймер, при срабатывании, запускает соответствующий сервис “my.service”

[Install]

WantedBy=timers.target # Активирует файл .target если есть и запущен timers.target

**Основные секции файла .timer**

1. [Unit] — стандартная секция для всех файлов systemd, содержит описание таймера:

* Description — краткое описание цели таймера;
* Requires=– запуск таймера произойдет только если существует соответствующий указанный сервис файл;

1. [Timer] — основная секция, где задаются параметры работы таймера:

* OnCalendar — задаёт расписание запуска (например, ежедневно, ежемесячно, по конкретным датам);
* OnBootSec= — время после загрузки Системы (например, запуск через 5 минут после загрузки);
* OnUnitActiveSec= — время после последнего запуска службы (например, через 10 минут после последнего выполнения).
* Persistent= — если true, то задача будет выполнена сразу после включения Системы, если во время её выключения был пропущен запуск.
* Unit=— данный параметр означает, что таймер, при срабатывании, запускает указанный сервис файл.

1. [Install] — содержит информацию о том, как таймер должен быть активирован.

* WantedBy= — если есть и запущен файл timers.target (который активирует все таймеры systemd), то активируется целевой .target файл и начинается отсчет времени таймера.

**Элементы расписания OnCalendar**

Значения параметра OnCalendar указывают на время или периодичность выполнения задания файла .service.

Формат параметра OnCalendar:

\* \*-\*-\* \*:\*:\*

DayOfWeek Year-Month-Day Hour:Minute:Second

Этот формат можно комбинировать или опускать некоторые элементы. Синтаксис параметра поддерживает различные вариации, чтобы можно было точно настроить расписание.

Элементы расписания OnCalendar:

* DayOfWeek — день недели (Sun, Mon, Tue, Wed, Thu, Fri, Sat).

Mon \*-\*-\* 02:00:00 — каждый понедельник в 2:00.

* Year-Month-Day — год, месяц и день.

2024-10-01 — 1 октября 2024 года.

* Использование символов подстановки: \* — любая дата (например, \*-\*-01 означает первый день каждого месяца).

Запись \*-\*-\* означает каждый день в любом году и месяце.

* Hour:Minute — время в формате часов, минут и секунд.

14:30:00 — 14 часов 30 минут ровно.

Если нужно указать только часы и минуты, можно опустить секунды: 14:30.

Параметры опции OnCalendar:

* Ежедневный запуск в полночь:

OnCalendar=daily

* Еженедельный запуск (по умолчанию воскресенье в 0:00). :

OnCalendar=weekly

* Ежемесячный запуск (в полночь первого числа каждого месяца):

OnCalendar=monthly

* Ежегодный запуск (первого января каждого года).

OnCalendar=yearly

или

OnCalendar=annually

* Запуск каждый день в 2:00 ночи:

OnCalendar=\*-\*-\* 02:00:00

* Запуск по будням в 8:00 утра:

OnCalendar=Mon..Fri \*-\*-\* 08:00:00

* Запуск в первый день каждого месяца ровно в полночь:

OnCalendar=\*-\*-01 00:00:00

* Запуск каждый год в конкретный день (например, 25 декабря каждого года):

OnCalendar=\*-12-25 00:00:00

* Запуск раз в час:

OnCalendar=hourly

* Запуск каждые 15 минут:

OnCalendar=\*:00/15

Если таймер содержит опцию Persistent=true, это означает, что, если Система была выключена в момент времени, когда задача должна была запуститься, она будет выполнена сразу после следующего включения. Это полезно для гарантий выполнения важных задач.

systemd позволяет комбинировать параметры настроек. Например:

OnCalendar=Mon \*-\*-\* 03,15:00:00

Здесь задается запуск задачи каждый понедельник в 3:00 и 15:00.

Чтобы проверить правильность работы таймера, можно использовать команду:

systemctl list-timers

Она покажет все таймеры, когда они были последовательно выполнены и когда будут выполнены в следующий раз.

* + - 1. Работа таймеров с файлами .service

Для работы таймеров необходимо создать соответствующий файл WantedBy=multi-user.target.service, который описывает, что должно выполняться, когда таймер срабатывает.

Файл .timer определяет расписание запуска.

Файл .service определяет, что нужно выполнить по этому расписанию.

Таймер запускает указанный сервис в момент времени, заданный в .timer файле.

Для примера рассмотрим файл .service для выполнения задачи резервного копирования:

[Unit]

Description=Задача резервного копирования

Wants=my.timer

[Service]

Type=oneshot

ExecStart=/usr/bin/rsync -av /home/ /mnt/backup

[Install]

WantedBy=multi-user.target

где:

* Параметр Wants=my.timer указывает, что необходимый сервис запустится и будет в режиме ожидания если запущен и работает таймер "my.timer".
* Параметр WantedBy=multi-user.target указывает, на каком уровне загрузки Системы будет работать данный сервис.

Основные команда для управления таймерами:

* Запуск таймера вручную:

sudo systemctl start backup.timer

* Активация таймера при запуске Системы:

sudo systemctl enable backup.timer

* Проверка статуса таймера:

systemctl status backup.timer

* Просмотр всех активных таймеров:

systemctl list-timers

* + 1. Состояние Системы

Состояние Системы и набор загружаемых сервисов определяется целями (targets) systemd. Цели (targets) systemd представляют собой группы служб и юнитов, которые должны быть активированы в определенных состояниях Системы.

Основные цели systemd:

* poweroff.target – используется для полного отключения Системы.
* rescue.target – вводит Систему в однопользовательский режим для диагностики и восстановления. Этот режим предназначен для выполнения административных задач, доступных только для root.
* multi-user.target – полный многопользовательский режим с поддержкой сети. Эта цель запускает текстовый интерфейс, и большинство серверов работают именно в этом режиме.
* graphical.target – запускает Систему в многопользовательском режиме с графическим интерфейсом (GUI). Этот режим используется на настольных системах.
* reboot.target – перезагружает Систему.

Помимо стандартных целей, systemd включает и другие специализированные цели:

* emergency.target – самый минималистичный режим с доступом только к корневой консоли. Используется для восстановления в критических ситуациях.
* halt.target – останавливает Систему без полного выключения (зависает на экране "halt").
* suspend.target, hibernate.target, hybrid-sleep.target – Эти цели управляют различными состояниями энергосбережения, такими как приостановка (suspend), гибернация (hibernate) и гибридный сон.

Цели (targets) в systemd определены в виде файлов с расширением .target. Системные файлы .target расположены в каталоге /lib/systemd/system/ или /etc/systemd/system/ и отвечают за запуск Системы в указанном состоянии. Пользовательские файлы .target хранятся в /usr/lib/systemd/user/.

Переключаться между целями можно с помощью команды:

sudo systemctl isolate <цель>

Например, чтобы перевести Систему в графический режим:

sudo ystemctl isolate graphical.target

Чтобы просмотреть активную цель:

sudo systemctl get-default

* + 1. systemd-resolved

systemd-resolved — системная служба, предназначенная для разрешения сетевых имён, входящая в состав пакета systemd. Служба обеспечивает поддержку различных протоколов разрешения имён, включая:

* традиционную систему доменных имён (DNS), в том числе DNSSEC и DNS over TLS;
* Multicast DNS (mDNS);
* Link-Local Multicast Name Resolution (LLMNR).
  + - 1. Поддерживаемые режимы работы

Служба systemd-resolved может функционировать в одном из четырёх режимов. Наиболее часто применяются следующие два:

* Работа через DNS-заглушку systemd (рекомендуемый режим): в этом режиме используется локальный адрес 127.0.0.53 в качестве единственного DNS-сервера. Необходимо создать символическую ссылку:

ln -sf /run/systemd/resolve/stub-resolv.conf\ /etc/resolv.conf

Это обеспечивает корректную работу всех процессов с использованием systemd-resolved.

* Работа с прямым использованием файла /etc/resolv.conf: в этом режиме служба использует указанный файл напрямую, как обычный DNS-клиент. Подходит для ситуаций, когда требуется сохранить совместимость с устаревшими приложениями.

Режим работы определяется автоматически, в зависимости от содержимого или назначения символической ссылки /etc/resolv.conf.

Для получения сведений о текущих настройках systemd-resolved, включая активные DNS-серверы, используется команда:

resolvectl status

Служба работает автоматически с сетевыми менеджерами, использующими файл /etc/resolv.conf, в том числе с systemd-networkd и NetworkManager. При этом DNS-серверы передаются службе systemd-resolved автоматически.

При необходимости можно вручную задать параметры DNS-серверов и fallback-серверов в файле /etc/systemd/resolved.conf. Пример настройки:

[Resolve]

DNS=192.168.35.1 fd7b:d0bd:7a6e::1

FallbackDNS=208.67.222.222 208.67.220.220

В случае отсутствия информации от сетевого менеджера и при неуказанных вручную серверах служба использует зарезервированные DNS-адреса (например, Яндекс 77.88.8.8, Google 8.8.8.8).

Управление службой systemd-resolved:

* перезапуск службы:

sudo systemctl restart systemd-resolved.service

* остановка службы:

sudo systemctl stop systemd-resolved.service

По умолчанию служба systemd-resolved включена и активна.

* + - 1. Конфликт порта 53 при использовании systemd-resolved

Служба systemd-resolved по умолчанию использует порт 53 (протокол DNS) на локальном интерфейсе 127.0.0.53 для организации локальной DNS-заглушки. В случае установки и запуска на том же узле стороннего DNS-сервера (например, named в составе BIND), также предназначенного для работы на порту 53, возникает конфликт ресурсов: обе службы не могут одновременно прослушивать один и тот же порт.

Подобная ситуация характерна, в частности, при настройке контроллера домена Samba с использованием внешнего DNS-сервера. Для предотвращения конфликта требуется отключить использование порта 53 службой systemd-resolved.

Для отключения порта 53 необходимо выполнить следующие действия:

1. Создать каталог для дополнительной конфигурации службы:

mkdir -p /etc/systemd/resolved.conf.d

1. Создать в нём файл с настройкой, отключающей локальную DNS-заглушку:

echo -e '[Resolve]\nDNSStubListener=no' >\ /etc/systemd/resolved.conf.d/no53port.conf

1. Перезапустить службу systemd-resolved, чтобы применить изменения:

sudo systemctl try-restart systemd-resolved

После выполнения указанных действий служба systemd-resolved прекращает использовать порт 53, что позволяет другим компонентам (например, named) корректно его задействовать.

* 1. Файловые системы

Файловая система (ФС) определяет способ организации и хранения данных на носителях информации, таких как жесткие диски, SSD и USB-накопители. Она управляет тем, как данные записываются, хранятся и читаются с устройств хранения, предоставляя структуру каталогов и файлов.

ОС поддерживает различные файловые системы (ФС) для обеспечения гибкости и оптимальной производительности при работе с дисками и разделами. По умолчанию используется журналируемая файловая система Ext4. Однако во время установки Системы администратору предоставляется возможность выбора других файловых систем (см. раздел 2.2.2), таких как Ext3, XFS, и BTRFS. Также поддерживается работа с дисками, отформатированными в NTFS (файловая система Windows).

* + 1. Поддерживаемые файловые системы

Рассмотрим общие аспекты каждой из ФС:

* **Ext4** (Fourth Extended Filesystem) является основной и рекомендованной ФС для ОС. Она представляет собой усовершенствованную версию Ext3, предлагая улучшенную производительность и дополнительные функции, такие как журналирование, поддержка больших файлов и разделов, быстрое выделение места для файлов и защита от фрагментации. Даная ФС предпочтительна при работе в ОС поскольку является стабильным и широко поддерживаемым выбором для большинства пользователей.

Журналирование обеспечивает сохранность данных в случае сбоев. Все изменения в каталогах сначала записываются в журнал, и только после этого применяются к основным данным. Это гарантирует целостность файловой системы и минимизирует риск повреждения данных

Ext4 включает механизм автоматического выравнивания блоков данных и индексных записей, что помогает поддерживать высокую производительность даже в случае фрагментированных данных.

Как и в других UNIX-подобных системах, в Ext4 доступ к каталогам управляется через систему прав доступа (rwx).

* **Ext3** (Third Extended Filesystem) является предшественником Ext4 и также поддерживается ОС РОСА ХРОМ. Основное преимущество — журналирование, но она менее производительна и обладает меньшими возможностями по сравнению с Ext4. Ext3 может использоваться для совместимости с более старыми системами или программами.
* **XFS** — это высокопроизводительная файловая система, разработанная компанией Silicon Graphics. Она отличается хорошей масштабируемостью и эффективной работой с большими файлами и разделами. XFS также обладает функциями журналирования, быстрым выделением пространства для файлов и достаточно высокой производительностью при записи данных, но не поддерживает нативное создание снимков (snapshot), что ограничивает его функциональность в современных сценариях резервного копирования и восстановления данных.
* **BTRFS** (B-Tree File System) — современная копируемая при записи (COW) файловая система, разработанная для повышения надежности данных и управления их объемом. Несмотря на то, что она поддерживает все аналогичные функции, как и Ext4, некоторых сценариях (особенно на серверах) производительность BTRFS может уступать Ext4, особенно при высокой нагрузке на запись или в случае большого количества мелких файлов, а также при создании множества снимков. Документация и поддержка BTRFS могут быть менее обширными и детализированными, чем у Ext4, что может вызывать затруднения при администрировании ОС.
* **NTFS** (New Technology File System) — файловая система, разработанная Microsoft для ОС Windows. В ОС РОСА «ХРОМ» поддержка NTFS реализована через встроенные драйверы ядра, которые позволяют монтировать и работать с NTFS-разделами напрямую. NTFS часто используется для обмена данными между Linux-подобными ОС и ОС семейства Windows.
  + 1. Организация каталогов в Системе

Каталоги в файловой системе Ext4 являются специальным типом файлов, которые содержат список имен файлов и их соответствующих inode-номеров. В inode хранится информация о файле или каталоге, включая права доступа, владельца, размер, а также указатели на блоки данных.

В Ext4 каталоги организованы как древовидная структура, где каждый каталог может содержать файлы и подкаталоги. Корневой каталог / является начальной точкой этой структуры. Внутри него находятся системные каталоги, такие как /bin, /etc, /home, /var и другие.

В Ext4 реализована схема HTree (сбалансированное дерево), которая позволяет увеличить скорость поиска и доступа к файлам в больших каталогах.

Пределы и ограничения на хранение данных в каталогах ФС:

* Ext4 поддерживает каталоги размером до 2 ТБ, что существенно увеличивает их емкость по сравнению с Ext3.
* В одном каталоге Ext4 может содержать до 1018 файлов, что практически не ограничивает количество файлов в каталоге.

Стандартная структура каталогов в ОС позволяет администраторам и пользователям эффективно управлять Системой и поддерживать её в рабочем состоянии.

Основные каталоги ОС, которые используются Системой по умолчанию:

* / (Корневой каталог) - начальная точка всей ФС, все другие каталоги и файлы находятся внутри этого каталога;
* /bin - содержит основные исполняемые файлы (команды), необходимые для работы Системы, которые могут использовать все пользователи (Например: bash, ls, cp);
* /boot - содержит файлы, необходимые для загрузки системы, такие как ядро, загрузчик и начальный RAM диск;
* /dev - содержит файловые интерфейсы к устройствам (драйверам). Здесь находятся файлы устройств, которые представляют периферийные устройства, такие как жесткие диски, терминалы и другие;
* /etc - содержит конфигурационные файлы Системы и служб. Это место, где хранятся конфигурации практически всех программ и служб на Системе;
* /home - содержит домашние каталоги пользователей. Каждый пользователь имеет свой собственный подкаталог, например /home/user;
* /lib - содержит общие библиотеки, используемые системными программами, и модули ядра;
* /media - место для автоматического монтирования съемных носителей, таких как CD-ROM, USB-диски и другие;
* /mnt - каталог, используемый для временного монтирования файловых систем;
* /opt - каталог для установки дополнительного ПО, которое не входит в стандартный набор программ ОС;
* /proc - виртуальная файловая система, предоставляющая информацию о текущем состоянии Системы, включая информацию о процессах;
* /root - домашний каталог пользователя root (суперпользователя);
* /run - каталог для временных данных, которые нужны при запуске Системы, таких как PID-файлы;
* /sbin: содержит системные программы, которые обычно используются только администратором Системы (root);
* /srv - каталог, предназначенный для хранения данных, которые обслуживаются Системой, например веб-сервером или FTP-сервером;
* /tmp - каталог для временных файлов. Эти файлы обычно удаляются при перезагрузке Системы;
* /usr - содержит вторичные программы, библиотеки и документацию. Внутри него также находятся такие подкаталоги, как:
* /usr/bin - дополнительные исполняемые файлы;
* /usr/lib - дополнительные библиотеки;
* /usr/share - общие файлы данных, такие как значки и документация;
* /usr/local - место для установки программного обеспечения вручную, которое не будет затронуто при обновлении Системы;
* /var - каталог, содержащий данные, которые изменяются во время работы Системы, такие как логи, файлы временного хранения, очереди задач и т.д.

В пользовательских каталогах наряду с персональными файлами также имеются и конфигурационные файлы, которые в целях безопасности срыты. Файлы и каталоги считаются скрытым, если их имена начинаются с точки (.). (например .config является скрытым).

Для того чтобы увидеть скрытые файлы в командной строке, можно использовать команду ls с параметром -a:

ls -a

Чтобы сделать файл или каталог скрытым, достаточно переименовать его, добавив в начало имени точку. Например, переименуем filename в .filename:

mv filename .filename

* + 1. Монтирование

Прежде чем приступить к работе с ФС, её нужно смонтировать, что означает подключение к файловой структуре Системы. Этот процесс позволяет ОС правильно взаимодействовать с ФС, делая её доступной как обычный каталог, известный как точка монтирования.

Для успешного монтирования ФС, важно убедиться, что каталог, выбранный в качестве точки монтирования, существует в Системе. Если точка монтирования является не пустым каталогом, его содержимое будет недоступно до тех пор, пока файловая система не будет размонтирована. Поэтому рекомендуется использовать специально созданные каталоги для этой цели, которые обычно находятся в /mnt или /media.

Допустим, необходимо смонтировать ISO-образ в каталог /mnt. Этот каталог должен существовать заранее, иначе операция завершится неудачно. После успешного монтирования в каталоге появится содержимое ФС, в противном случае каталог останется пустым.

Чтобы определить, к какой ФС относится текущий каталог и объём свободного места на ней, используют команду:

df -h

В ОС для монтирования ФС используется команда mount. Общий синтаксис команды:

sudo mount устройство точка\_монтирования

где:

* устройство — это физический носитель данных, который необходимо подключить;
* точка\_монтирования — каталог, к которому будет привязана ФС.

Из соображений безопасности команда mount доступна для выполнения только с правами суперпользователя. Команда mount может включать дополнительные параметры, такие как:

* -f - имитирует монтирование, выполняя все действия, кроме фактического монтирования файловой системы;
* -v - предоставляет подробный отчёт о выполняемых действиях;
* -w - монтирует файловую систему с доступом для чтения и записи;
* -r - монтирует файловую систему только для чтения;
* -n - монтирует без записи в файл /etc/mtab;
* -t type - указывает тип монтируемой файловой системы;
* -a - монтирует все файловые системы, указанные в /etc/fstab;
* -o list\_of\_options - применяет список опций к файловой системе, где опции разделены запятыми, подробнее об этом см. в man mount.

Подробное описание опций команды mount доступно в руководстве man mount.

Если у команды не указаны опции, mount попытается определить их по файлу /etc/fstab.

Примеры команд монтирования:

* Монтирование раздела жесткого диска:

sudo mount /dev/hdb3 /mnt

* Монтирование всех ФС NFS, указанных в /etc/fstab:

sudo mount -vat nfs

Если возникли ошибки при монтировании, можно использовать команду:

sudo mount -vf устройство точка\_монтирования

Эта команда выполнит все действия, кроме монтирования, и выдаст подробный отчет о каждом шаге.

Файл /etc/fstab — это конфигурационный файл, который определяет порядок монтирования файловых систем в ОС. Этот файл играет ключевую роль в автоматическом подключении устройств и разделов при загрузке Системы, а также упрощает управление монтированием для системных администраторов.

Файл /etc/fstab состоит из строк, каждая из которых описывает отдельную файловую систему или устройство. Рассмотрим типовую структуру файла fstab:

# # <файловая система> <точка монтирования> <тип> <параметры> <dump> <pass>

UUID=abcd1234-ef56-7890-abcd-1234567890ab / ext4 defaults 1 1

UUID=abcd1234-ef56-7890-abcd-1234567890ac /home ext4 defaults 1 2

UUID=abcd1234-ef56-7890-abcd-1234567890ad /var xfs defaults 1 2

UUID=abcd1234-ef56-7890-abcd-1234567890ae swap swap defaults 0 0

/dev/cdrom /media/cdrom auto noauto,ro 0 0

Каждая строка включает следующие поля:

* <файловая система> — идентификатор файловой системы, которая должна быть смонтирована. Обычно это UUID (уникальный идентификатор) раздела, но может быть также использован путь к блочному устройству (например, /dev/sda1).

В рассмотренном выше примере строка: UUID=abcd1234-ef56-7890-abcd-1234567890ab, /dev/cdrom.

* <точка монирования> — каталог, в который будет подключена файловая система.

В рассмотренном выше примере: /, /home, /var, /media/cdrom.

* <тип> — тип файловой системы, который будет использоваться при монтировании. Это может быть ext4, xfs, btrfs, swap для разделов подкачки, или auto для автоматического определения типа.

В рассмотренном выше примере: ext4, xfs, swap, auto.

* <параметры> — монтировочные опции, которые определяют поведение файловой системы. Наиболее часто используются опции:
* defaults — включает стандартный набор опций: rw, suid, dev, exec, auto, nouser, async.
* noauto — файловая система не монтируется автоматически при загрузке.
* ro — монтирование в режиме только для чтения.

В рассмотренном выше примере: defaults, noauto,ro.

* <dump> — опция для утилиты dump, которая выполняет резервное копирование файловой системы. Если установлено в 1, резервное копирование включено, 0 — отключено.

В рассмотренном выше примере: 1, 0.

* <pass> — определяет порядок проверки файловых систем при загрузке:
* 0 — файловая система не будет проверяться.
* 1 — корневая файловая система проверяется первой.
* 2 — другие файловые системы проверяются после корневой.

В рассмотренном выше примере: 1, 2, 0.

Редактирование файла /etc/fstab требует прав суперпользователя. Изменения в этом файле немедленно не применяются, поэтому после редактирования необходимо либо вручную смонтировать файловые системы командой

sudo mount -a

либо перезагрузить Систему для применения настроек.

Чтобы избежать ошибок при редактировании /etc/fstab, рекомендуется тщательно проверять правильность введенных данных, так как ошибки могут привести к тому, что Система не сможет загрузиться корректно.

Для получения полной информации о возможных опциях для файла /etc/fstab смотрите справочную страницу по команде man fstab.

* + 1. Размонтирование

Для отключения файловой системы от ОС используется команда umount. Операция размонтирования может потребоваться для выполнения проверок и восстановления ФС с использованием команды fsck.

В случае проблем с сетью, удаленные ФС также могут быть отключены.

Команда umount имеет следующий формат:

* umount device : mountpoint
* umount -a
* umount -t fstype

Здесь device обозначает физическое устройство, которое нужно отключить, а mountpoint — каталог, к которому оно подключено (можно указать только device или mountpoint). Параметр -a отключает все подключенные ФС, а параметр -t fstype отключает ФС указанного типа.

Команда umount не сможет отключить ФС, если она в данный момент используется. Например, если ФС подключена к /mnt, и вы попытаетесь выполнить следующие команды:

cd /mnt

sudo umount /mnt

Вы получите сообщение об ошибке, так как Система будет считать, что ФС занята. Чтобы успешно отключить /mnt, необходимо сначала выйти из этого каталога, перейдя в другой каталог.

Для принудительного отключения устройства, даже если оно используется, можно применить опцию -f:

sudo umount -f /cdrom

Для безопасного извлечения сменных носителей можно использовать команду eject.

Более подробное описание команды доступно в справочном руководстве man umount.

* + 1. Файловая система GFS2

Файловая система GFS2 (Global File System 2) является кластерной распределённой файловой системой, которая обеспечивает одновременный доступ к одному и тому же хранилищу с нескольких узлов под управлением ОС на базе Linux. Это решение позволяет организовать совместную работу серверов в кластере с общей файловой структурой при сохранении целостности данных и согласованности доступа.

ОС включают поддержку GFS2 в составе ядра и репозитория пакетов, что позволяет использовать данную файловую систему без необходимости сборки модулей вручную.

GFS2 использует модель общего доступа к блочному устройству (Shared Block Storage) с распределённым управлением блокировками через DLM (Distributed Lock Manager). Все узлы в кластере обнаруживают один и тот же том, а механизм DLM обеспечивает согласование одновременного чтения и записи.

Основные особенности GFS2:

* согласованное использование метаданных между узлами;
* управление журналами транзакций;
* возможность расширения размера файловой системы;
* работа поверх общих хранилищ, подключаемых через SAN, iSCSI или Fibre Channel.

Для координации работы нескольких серверов используется кластерный менеджер (Pacemaker/Corosync), который отвечает за согласование конфигурации кластера и мониторинг доступности узлов.

Для работы с GFS2 требуется установить необходимые утилиты пользовательского пространства с помощью команды:

sudo dnf install gfs2-utils

Создание файловой системы GFS2 на общем устройстве выполняется командой:

sudo mkfs.gfs2 -p lock\_dlm -t cluster\_fs /dev/sdb

где:

* lock\_dlm — механизм блокировок DLM;
* cluster\_fs — имя кластера, которое должно соответствовать конфигурации Pacemaker/Corosync;
* /dev/sdb — общий раздел или блочное устройство.

Для монтирования файловой системы GFS2 на сервере выполняются следующие действия:

sudo mkdir -p /mnt/gfs2

sudo mount -t gfs2 /dev/sdb /mnt/gfs2

На клиентских узлах монтирование производится аналогично, с указанием сетевого имени сервера:

sudo mount -t gfs2 server1:/mnt/gfs2 /mnt/gfs2

где server1 — DNS-имя или IP-адрес сервера.

Для обеспечения корректной синхронизации узлов в кластере используется связка Pacemaker и Corosync. Необходимо:

* настроить Corosync для обмена сообщениями между узлами;
* сконфигурировать Pacemaker для управления ресурсом файловой системы (в том числе fencing, если требуется);
* зарегистрировать ресурс в Pacemaker, указав имя кластера, совпадающее с параметром -t при создании ФС.

Для мониторинга состояния GFS2 и диагностики возможных ошибок следует использовать следующие инструменты:

* journalctl — просмотр системных сообщений и логов ядра;
* gfs2\_tool —проверка и получение состояния ФС;
* gfs2\_quota — управление квотами в GFS2;
* fsck.gfs2 — проверка целостности файловой системы (применяется только при отключённом кластере).

Также рекомендуется контролировать загрузку сети и задержки, так как GFS2 критично зависит от стабильности межузловых соединений.

* 1. Установка и удаление программ (репозитории)

Для установки, удаления и обновления программ в ОС используется пакетная модель. Программное обеспечение поставляется в виде пакетов стандартного формата (для данной ОС – это rpm), содержащих исполняемые файлы и вспомогательные данные. Менеджер пакетов проверяет наличие необходимых зависимостей и регистрирует приложение в среде (доступ из командной строки и в меню графической оболочки).

Так как ОС построена по модели совместного использования кода, программы в ней гораздо меньше дублируются и более взаимозависимы. Это с одной стороны приводит к уменьшению занимаемого ОС пространства на диске, а с другой это может вызвать проблемы совместимости при установке программ, собранных для других дистрибутивов Linux.

В качестве основной системы управления пакетами используется dnf. Dnf работает с репозиториями, автоматически разрешает зависимости, проверяет подписи и целостность пакетов и тем самым снижает риск нарушений согласованности. Низкоуровневые операции выполняются через rpm и отражаются в базе установленных пакетов.

В ОС программы устанавливаются из собственных централизованных сетевых хранилищ, называемых репозиториями. Каждый репозиторий содержит инсталляционные пакеты, связанные друг с другом зависимостями.

Для обеспечения безошибочной и безопасной работы Системы рекомендуется ставить пакеты только из официальных репозиториев РОСА, предназначенных именно для этой версии.

Репозитории в ОС требуют ключа для их использования, подробнее о размещении ключа в Системе см. п. 3.10.1.

Также для управления подключением онлайн-репозиториев служит файл /etc/dnf/vars/enable\_online\_repos установленное значение "1" в этом файле включает сетевые репозитории, "0" — выключает.

Сетевые репозитории для данной версии ОС по умолчанию подключены.

Для этого нужно выполнить команды:

sudo mkdir /mnt/disk1

sudo mount /dev/cdrom /mnt/disk1

После этого работа с дисковым репозиторием будет аналогична работе с сетевым. Файл его подключения можно найти по адресу /etc/yum.repos.d/disk1.repo**,** для его включения нужно установить в этом файлеenabled=1**.**

* + 1. Использование rpm

Rpm — это формат пакетов программного обеспечения и утилита низкого уровня для управления установленными пакетами в ОС. Инструмент обеспечивает установку и удаление отдельных пакетов, ведение базы установленных компонентов, запрос сведений о пакетах и проверку целостности файлов.

Rpm не управляет репозиториями и не подбирает зависимости пакетов автоматически, используется как вспомогательный инструмент для диагностики и контроля установленного ПО.

Штатная установка, удаление и обновление программ выполняется высокоуровневым менеджером пакетов dnf. Все операции dnf (установка, обновление, удаление) выполняются через библиотеки rpm и фиксируются в базе rpm.

Типовые запросы rpm:

* Определение, каким пакетом установлен файл:

rpm -qf /путь/к/файлу

* Перечень файлов, входящих в пакет:

rpm -ql имя\_пакета

* Проверка целостности файлов пакета:

rpm -V имя\_пакета

* + 1. Файловый менеджер dnf

Dnf — менеджер пакетов высокого уровня, который используется в ОС. Dnf обеспечивает установку, обновление и удаление программ из репозиториев. При работе автоматически подбираются и устанавливаются необходимые зависимости, что упрощает администрирование и снижает риск ошибок. Инструмент рекомендуется как основной для повседневного управления программным обеспечением и имеет графическую оболочку в виде программы "Пакеты (dnfdragora)", доступ к которой осуществляется через главное меню приложений Системы.

Перед установкой dnf выполняет проверку подлинности пакетов (криптографические подписи), что помогает защититься от подмены. Изменения применяются по принципу "целиком или никак": если на любом этапе возникает ошибка (например, не найдена зависимость), установка не выполняется, и Система остаётся в прежнем рабочем состоянии. Выполненные операции можно просматривать в истории, а ход работы фиксируется в журналах менеджера (например, /var/log/dnf.log).

Синтаксис команд dnf:

dnf <действие> <пакет> <ключ>

Основные команды dnf:

* Установка пакетов:

sudo dnf install <имяпакета>

После выполнения команды Система автоматически загрузит и установит выбранный пакет, а также все его зависимости.

* Удаление пакетов:

sudo dnf erase <имяпакета>

Эта команда удаляет не только выбранный пакет, но и все его зависимости.

* Поискпакетов в Системе:

dnf search <ключевое\_слово>

Эта команда позволяет найти программу или узнать, какие пакеты доступны в репозиториях по ключевому слову, например browser.

* Просмотр информации о пакете:

dnf info <название\_пакета>

* Очистка кеша и ненужных пакетов:

sudo dnf autoremove

sudo dnf clean all

Полный список команд dnf доступен в консоли по команде:

man dnf

* + 1. Установка программ из сторонних источников

Следует обратить внимание, что установка программ из сторонних репозиториев **не рекомендуется и может нарушить целостность и безопасность Системы**, однако при необходимости она возможна.

Лучшим способом установки программ из сторонних репозиториев будет самостоятельная сборка пакетов из исходных текстов, пользуясь онлайн-средой сборки abf.io (приглашение для регистрации в среде можно получить по запросу к сотрудникам компании-разработчика ОС РОСА). При этом администратор становится владельцем своего отдельного репозитория и сможет с помощью инфраструктуры РОСА собирать требуемые пакеты. Данный способ является самым желательным и распространенным при необходимости установки программ, отсутствующих в репозиториях РОСА.

Установка сторонних программ также возможна и с помощью использования пакетов, созданных для схожих с ОС РОСА "ХРОМ" дистрибутивов. Наиболее совместимыми с пакетами РОСА являются пакеты для OpenMandriva, Mageia и Fedora: их можно попробовать установить командой:

sudo dnf install <ФайлПакета>

Возможно, зависимости таких пакетов не будут совпадать с зависимостями РОСА и блокировать установку – в таком случае можно попробовать установить скачанный пакет командой:

sudo rpm -i --nodeps <ФайлПакета>

Такой способ часто используется для установки сторонних драйверов.

Также возможно установить автономные пакеты flatpack или appimage, которые полностью поддерживаются в ОС в качестве систем установки.

* + 1. Использование Notamock для сборки пакетов

Notamock представляет собой набор утилит для создания изолированных окружений при сборке RPM-пакетов в дистрибутивах ОС из семейства РОСА. Название отражает его сходство с утилитой mock, но при этом реализованы принципиальные отличия — использование многослойных контейнеров с общей базовой корневой файловой системой (rootfs) аналогично технологии Docker. Это позволяет ускорить сборку за счет того, что минимальное окружение всегда готово, а устанавливаются только необходимые зависимости.

Основные компоненты Notamock:

* notamock — утилита для запуска локальной сборки одного проекта;
* notamocks — утилита для массовой (параллельной) локальной сборки нескольких проектов;
* notamockc — утилита для запуска контейнера с любой командой (по умолчанию – bash);
* notamockd — утилита для создания сборочных контейнеров для системы ABF;
* notamocka — скрипт запуска ABF Builder внутри контейнера (устанавливается из пакета notamock-abf).

Все утилиты используют общий конфигурационный файл /etc/notamock.cfg, а также дополнительные настройки из каталога /etc/notamock.d/. При необходимости может быть указан иной конфигурационный файл с параметром --config. Все утилиты требуют прав суперпользователя (root) из-за операций с монтированием.

Базовая корневая файловая система (rootfs) создается для каждой поддерживаемой платформы только один раз и затем многократно переиспользуется. Количество одновременно работающих контейнеров, процессов notamock и builders не влияет на количество rootfs. Для пересоздания базовой rootfs необходимо запустить любую утилиту с параметром -u в момент, когда rootfs не используется другими процессами.

При этом для актуализации rootfs достаточно обновления самой Cистемы, дополнительных действий не требуется.

Использование утилит Notamock:

* **notamock**

Утилита notamock применяется для локальной сборки одного проекта. В типичном сценарии взаимодействует с ABF (Automated Build Farm), но также может пересобрать пакет по src.rpm. Пример пересборки проекта lsof:

sudo dnf install abb notamock

abb clone lsof

cd lsof

notamock

При первом запуске будет создана базовая rootfs, что занимает больше времени. Последующие сборки выполняются значительно быстрее.

Полезные параметры notamock:

* -v — подробный вывод сообщений;
* -r — указание целевой платформы (например -r 13);
* -e — остановка в контейнере перед запуском сборки (удобно для отладки);
* RPMSAVE=yes — сохранение всех скачанных сборочных зависимостей во внутреннем репозитории notamock;
* REUSABLE=yes — повторное использование контейнера для ускорения последовательных сборок одного проекта;
* PM\_config — возможность подмены конфигурации пакетного менеджера, например PM\_config=/home/user/dnf.conf;
* TMPD — указание собственного каталога вместо /tmp/notamock.

Все параметры могут быть переопределены напрямую через командную строку или настроены в /etc/notamock.cfg. Более подробную справку можно получить по команде notamock --help.

* **notamocks**

Утилита notamocks предназначена для массовых сборок сразу нескольких проектов в параллельном режиме. Предполагается, что проекты располагаются в подкаталогах внутри одного каталога.

Пример запуска утилиты по команде:

cd /путь/к/проектам

notamocks

Полезные параметры notamocks:

* -p — количество параллельных сборок;
* -s — пропускать уже собранные проекты;
* -n (или --chain) — цепочная сборка с добавлением результатов предыдущих итераций в репозиторий;
* -l — каталог для хранения логов.
* **notamockd**

Утилита notamockd предназначена для интеграции со сборочной системой ABF. Может запускаться в двух режимах:

* создание и запуск контейнера с builders напрямую;
* создание Docker-образа и запуск контейнера через Docker.

Для запуска в основном режиме необходимо указать токен сборщика:

notamockd -t <TOKEN>

Полезные опции утилиты notamockd:

* -p — количество builders;
* -b — имя builder;
* -B — внешний каталог для сборки;
* --watch — мониторинг логов;
* --tmux — запуск в tmux внутри контейнера для отладки.

Если не указан внешний каталог через опцию -B, сборка будет происходить в /tmp контейнера. В случае необходимости можно изменить поведение на использование chroot опцией CHROOT=INTERNAL в конфигурации или указать TMPD с собственным путем.

Если требуется создание Docker-образа с builder используются следующие команды:

sudo dnf install docker

sudo systemctl start docker

notamockd --docker -b builder\_name

docker image ls

docker run -it --rm --privileged -e BUILD\_TOKEN="ABF\_TOKEN"\ -e PARALLEL=3 -v /tmp:/tmp <docker image ID>

Примечание –Для корректной работы требуется смонтировать том через "-v", иначе overlayfs внутри контейнера не будет работать.

* **notamockc**

Утилита notamockc используется для запуска контейнера с любыми командами в среде chroot, основанной на слоеных rootfs. Это позволяет получить доступ к окружению сборки без необходимости отдельного скачивания образа с ABF или создания rootfs вручную.

Для запуска утилиты достаточно выполнить команду:

notamockc

При необходимости можно указать целевую платформу с помощью параметра -r, например:

notamockc -r 13

Также возможно сразу передать команду, которая будет выполнена в chroot:

notamockc -r 13 -- dnf provides thunderbird

Если требуется сохранить изменения между запусками, следует создать именованный контейнер:

notamockc -n my\_container

Чтобы изменения контейнера сохранялись и между перезагрузками хоста, необходимо указать путь для хранения временных данных через TMPD как в командной строке, так и в конфигурационном файле /etc/notamock.cfg.

Для проброса необходимых файлов в контейнер удобно использовать параметр BIND. Например:

notamockc BIND='/home/user/project::/root/project'

Также можно указать дополнительные опции монтирования, например, только для чтения и без возможности исполнения:

notamockc BIND='/home/user/project::/root/project::ro::noexec'

Помимо каталогов, возможно монтирование squashfs-архивов, образов или разделов диска. В данном случае происходит полноценное монтирование, а не bind. Опция bind доступна как в командной строке, так и через конфигурационные файлы, и может применяться во всех утилитах Notamock.

Notamock может применяться для сборки пакетов с последующим статическим анализом кода с помощью комплекса SVACE. Для этого нужно выполнить следующие шаги:

1. На хостовой Cистеме распаковать архив SVACE по команде, например:

/home/user/svace/svace-3.4.240312-x64-linux

Создать файл конфигурации пакетного менеджера, например /etc/notamock-dnf.conf:

[main]

gpgcheck=0

best=True

skip\_if\_unavailable=False

keepcache=True

reposdir=''

[rosa-contrib]

name=rosa-contrib

baseurl=http://abf-downloads.rosalinux.ru/rosa$releasever/repository/$arch/contrib/release/

gpgcheck=0

[rosa-main]

name=rosa-main

baseurl=http://abf-downloads.rosalinux.ru/rosa$releasever/repository/$arch/main/re\lease/

gpgcheck=0

Cоздать отдельный профиль Notamock, например /etc/notamock.d/svace, командой:

PM\_config=/etc/notamock-dnf.conf

BUILD\_cmd='/opt/svace/bin/svace init && /opt/svace/bin/svace build rpmbuild -ba \*.spec --define="\_sourcedir $INTERNAL\_DIR" --define="\_topdir $INTERNAL\_DIR"'

BIND\_SOURCE=yes

BIND='/home/user/svace/svace-3.4.240312-x64-linux:/opt/svace:ro'

Для запуска сборки выполнить команду:

sudo notamock -r 2021.15 name-X-Y.src.rpm

Результаты анализа будут сохранены в каталоге name/.svace-dir относительно текущей директории запуска. При необходимости последующего анализа эти данные можно перенести на другую машину и использовать утилиту svace analyze.

* 1. Управление жесткими дисками

Правильное управление и обслуживание жестких дисков играют ключевую роль в обеспечении надежности и производительности Системы, а также в защите данных от потерь. Эффективное управление дисковыми устройствами напрямую влияет на производительность Системы. Помимо физических дисков, Система может работать с виртуальными дисковыми устройствами, которые используются в контейнеризации, виртуализации и других технологиях.

Одна из необходимостей разделения жесткого диска — это хранение разных ОС на одном физическом жестком диске.

Основные инструменты, которые используются в ОС для работы с жесткими дисками:

* текстовые утилиты – fdisk;
* графические интерфейсы – диспетчер разделов kde; gparted;
* разбиение диска на разделы при установке ОС.

Разбиение на разделы в ходе инсталляции Системы осуществляется на шаге выбора места установки и подробнее описано в разделе 2.2.2.

* + 1. Fdisk

fdisk — это мощная текстовая утилита для работы с таблицами разделов на жестких дисках. С ее помощью можно создавать, изменять, удалять и просматривать разделы дисков.

Для просмотра списка доступных дисков и разделов используются команды

lsblk

или

disk -l

Примечание – Для начала работы с fdisk необходимо открыть терминал и выполнить команду, указав путь к устройству, с которым требуется работать:

Основные команды для работы с fdisk:

sudo fdisk /dev/sda

где /dev/sda — это имя устройства, соответствующего вашему жесткому диску. После перехода к жесткому диску вводят следующие параметры команды:

* m —выводит список всех доступных команд в fdisk; используется для полного ознакомления с утилитой;
* p — выводит на экран текущую таблицу разделов, чтобы увидеть, какие разделы уже созданы на диске;
* n — создать новый раздел; Система предложит выбрать тип раздела (основной или расширенный) и указать начальный и конечный сектора;
* d —удаляет выбранный раздел; рекомендуется использовать с осторожностью, так как удаление раздела приведет к потере всех данных на нем;
* w —сохраняет все изменения, сделанные в таблице разделов, и завершает работу fdisk. После записи изменений они вступают в силу, и их нельзя отменить;
* q — выйти без сохранения изменений, сделанных в текущем сеансе.

Примеры использования fdisk:

* Создание нового раздела:
  1. запустить fdisk для нужного диска:

sudo fdisk /dev/sda

* 1. чтобы создать новый раздел, необходимо ввести n;
  2. выполнить инструкции утилиты, указав:
* какой тип раздела создать: основной (primary) или расширенный (extended). Обычно используется основной;
* номер раздела (например, 1 для первого раздела);
* начальный сектор. Если оставить поле пустым, будет использован первый доступный сектор;
* конечный сектор или размер раздела (например, +10G для 10 Гб);
  1. если всё прошло успешно, ввести p, чтобы просмотреть таблицу разделов и убедиться, что новый раздел создан;
  2. для того чтобы записать изменения и выйти, ввести w;
* Удаление раздела:
  1. запустить fdisk для нужного диска:

sudo fdisk /dev/sda

* 1. для того чтобы удалить раздел, ввести d;
  2. утилита предложит выбрать номер раздела, который необходимо удалить;
  3. ввести p, чтобы убедиться, что раздел удален;
  4. ввести w для того, чтобы записать изменения и выйти.

Примечание – Перед внесением любых изменений в разделы диска рекомендуется сделать резервную копию всех важных данных, так как изменения в таблице разделов могут привести к потере данных.

* + 1. Диспетчер разделов KDE

Диспетчер разделов KDE — стандартный графический инструмент для управления разделами в среде KDE. Он предоставляет удобный интерфейс для выполнения операций над разделами, включая создание, удаление, изменение размеров и перемещение. Диспетчер разделов поддерживает все основные типы файловых систем, используемых в Unix-подобных системах.

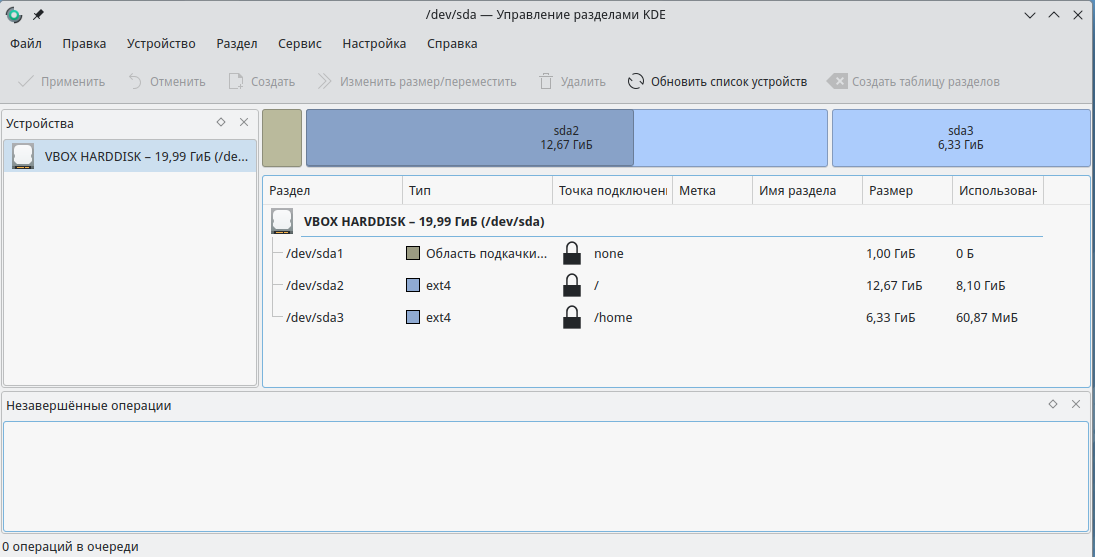


Рисунок 12 – Диспетчер разделов KDE

Для перехода к приложению нужно войти в "Главное меню Системы → Утилиты → Диспетчер разделов KDE"(рисунок 12).

Диспетчер разделов KDE предоставляет следующие основные функции:

* просмотр информации о дисках и разделах: о размере, файловой системе и свободном месте;
* создание новых разделов с выбором файловой системы;
* удаление разделов;
* изменение размеров разделов без потери данных, если файловая система поддерживает такие операции;
* форматирование разделов в различные файловые системы, такие как ext4, xfs, btrfs и другие;
* присвоение меток разделам для удобства их идентификации.

Практические инструкции по работе сДиспетчером разделов KDE:

1. Создание нового раздела:
   1. выбрать диск, на котором будет создан новый раздел;
   2. в верхнем навигационном меню программы перейти в "Раздел ⟶ Создать";
   3. задать тип раздела (например, основной или логический) и размер раздела;
   4. выбрать файловую систему для нового раздела, например ext4;
   5. при необходимости задать метку раздела;
   6. нажать на кнопку Применить, чтобы подтвердить создание нового раздела. Изменения будут применены только после подтверждения.
2. Изменение размера раздела:
   1. выбрать раздел, который необходимо изменить;
   2. необходимо по нажатию правой кнопкой мыши на разделе выбрать кнопкуИзменить размер;
   3. в открывшимся окне ввести новый желаемый размер раздела (рисунок 13);
   4. нажатием на кнопку ОК сохранить все внесенные изменения;

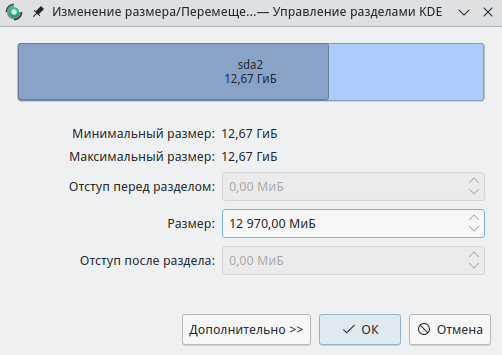


Рисунок 13 – Изменение размера раздела в Диспетчере разделов KDE

1. Удаление раздела:
   1. выбрать раздел, который необходимо удалить;
   2. необходимо по нажатию правой кнопкой мыши на разделе в контекстном меню выбрать кнопкуУдалить;
   3. для подтверждения действия нажать на кнопку Применить.

Примечание – Перед выполнением операций с разделами рекомендуется создать резервные копии важных данных, так как изменение разделов может привести к потере данных.

Примечание – Следует быть осторожными при работе с разделами, на которых установлена ОС или важные данные, чтобы избежать потери доступа к Системе.

* + 1. GParted

GParted — это мощный и гибкий графический инструмент для управления разделами жесткого диска. Он поддерживает широкий спектр файловых систем и позволяет выполнять разнообразные операции с дисками и разделами, такие как создание, удаление, изменение размера и перемещение разделов. GParted особенно полезен для сложных задач по переразметке дисков и работает в среде GNOME.

GParted позволяет просматривать и отменять действия до их применения. Также можно видеть все запланированные изменения в журнале перед их окончательной реализацией.

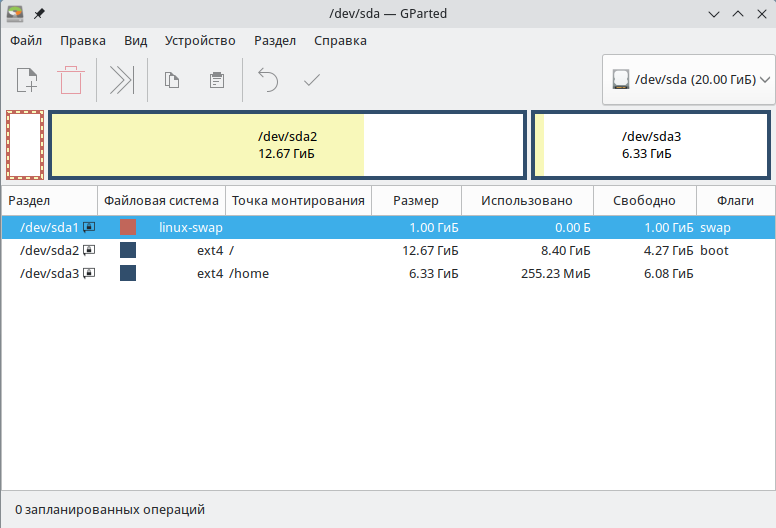


Рисунок 14 – Приложение GParted

Для перехода к приложению нужно войти в "Главное меню Системы → Все приложения → GParted". После запуска GParted открывается основной интерфейс программы (рисунок 14):

* Меню выбора диска: в верхнем правом углу в выпадающем списке можно выбрать нужный диск, если у вас несколько физических дисков.
* Основное окно, в котором отображаются все разделы выбранного диска в виде графической схемы.
* Информация о разделе: под схемой отображается информация о выделенном разделе: его файловая система, размер, статус (занятое и свободное место), а также метка.

Основные операции с разделами:

1. Создание нового раздела:
   1. выбрать диск с неразмеченным пространством;
   2. по нажатию правой кнопкой мыши по неразмеченному пространству на диске в открывшимся контекстном меню выбрать кнопку Создать;
   3. в открывшемся окне выбрать размер раздела, файловую систему (например, ext4, NTFS и т.д.) и метку;
   4. нажать на кнопку Добавить, чтобы сохранить изменения;
   5. для применения изменений нажать Применить в верхней панели инструментов.
2. Изменениеразмера раздела:
   1. выбрать раздел, который необходимо изменить;
   2. нажать на кнопку  Изменить размер или перенести выбранный раздел;
   3. ввести новый размер раздела;
   4. нажать на кнопку  Применить, чтобы сохранить изменения.
3. Удаление раздела:
   1. выбрать раздел, который необходимо удалить;
   2. нажать на кнопку  Удалить выбранный раздел;
   3. подтвердить действие нажатием на кнопкуПрименить**.**
4. Форматирование раздела:
   1. выбрать раздел, который нужно отформатировать;
   2. необходимо нажатием правой кнопкой мыши по разделу выбрать опцию "Форматировать в…" или перейти в пункт навигационного меню "Раздел → Форматировать в…", затем выбрать файловую систему, в которую необходимо отформатировать раздел;
   3. подтвердить форматирование нажатием на кнопку Применить;
5. Создание таблицы разделов:
   1. выбрать раздел, с которым будет продолжена работа;
   2. перейти в навигационном меню в раздел "Устройство → Создать таблицу разделов";
   3. в открывшемся окне при необходимости указать дополнительные параметры.

Примечание – Перед выполнением операций с разделами рекомендуется создать резервные копии важных данных, так как изменение разделов может привести к потере данных.

* 1. Режимы работы Системы

Функционирование ОС возможно в трех режимах – нормальном, в режиме восстановления и в аварийном режиме. По умолчанию при включении ПК ОС загружается в нормальном рабочем режиме, для включения других режимов необходимо добавление параметров загрузки в меню начального загрузчика.

Для работы в режиме восстановления и в аварийном режиме в общем случае нужны полномочия администратора Системы.

Для изменения параметров загрузчика нужно в начальном меню GRUB нажать клавишу E и в открывшемся редакторе вписать в параметр загрузки в строке, начинающейся с "linux /boot/vmlinuz...", например рядом с параметром quet. После этого нужно нажать клавишу F10 для загрузки ОС с указанными параметрами.

И режим восстановления, и аварийный режим являются однопользовательскими; в них невозможна одновременная авторизация нескольких пользователей, также в них не активируются сетевые интерфейсы.

* + 1. Режим восстановления

В ОС есть возможность загрузки в режиме восстановления (rescue mode) без графической оболочки и необязательных сервисов. Этот режим дает шанс восстановить Систему, которая перестала загружаться в графическом режиме, например, при сбое обновления или при указании нерабочих настроек в файлах конфигурации.

Для загрузки ОС в режиме восстановления необходимо при включении Системы, как описано в п.3.8, загрузиться с параметром systemd.unit=rescue.target.

После такой загрузки возможен вход в Систему с помощью консольного интерфейса. Система попытается смонтировать локальные разделы обычным образом, и появится возможность изменять файлы конфигурации редактором nano или встроенным редактором текстовой оболочки mc.

* + 1. Аварийный режим

Аварийный режим (emergency mode) используется для восстановления ОС или спасения данных в случаях серьезных ошибок функционирования, например, при повреждениях ФС, когда не удается войти даже в режиме восстановления.

Для загрузки Системы в аварийном режиме необходимо при включении Системы, как описано в п.3.8, загрузиться с параметром systemd.unit=emergency.target.

После такой загрузки возможен вход и управление ОС с помощью консольного текстового интерфейса. В аварийном режиме корневой раздел монтируется "только для чтения", другие разделы не монтируются, и сетевые интерфейсы не активируются.

* + 1. Синхронизация с репозиториями

Если сбой ОС произошел во время обновления вследствие проблем с сетью или ошибок питания, есть возможность в консольном интерфейсе запустить сеть командой:

sudo systemctl start NetworkManager

При необходимости установить сетевое соединение командой nmtui и синхронизировать текущую Систему с репозиториями, приведя версии ее пакетов в соответствие с актуальными. Это можно сделать командой:

sudo dnf distrosync

Следует обратить внимание, что при такой операции могут быть удалены сторонние пакеты, установленные не из подключенных репозиториев.

* + 1. Восстановление функционирования ОС
       1. Переустановка ОС с сохранением данных

Самым быстрым и простым способом восстановления Системы является ее переустановка с инсталляционного образа с сохранением пользовательских данных. В процессе установки важно не форматировать раздел /home, где хранятся пользовательские данные. Для этого в инсталляторе ОС Хром необходимо вручную выбрать раздел /home, убедившись, что не установлен флажок "Форматировать". Таким образом, все пользовательские файлы и настройки, находящиеся в этом разделе, сохранятся, а Система будет переустановлена в оптимальном состоянии.

Это подходит в случаях, когда системные файлы повреждены, но данные пользователей остаются целыми.

* + - 1. Использование утилиты TestDisk

В состав ОС включена утилита TestDisk, которая предназначена для восстановления потерянных данных и восстановления разделов, если произошли ошибки файловой системы. TestDisk может быть использован в ситуациях, когда произошел сбой в работе дисков или были случайно удалены разделы. Утилита поддерживает восстановление различных файловых систем.

Для восстановления данных с помощью TestDisk:

1. Установка TestDisk (если утилита не была предустановлена) командой:

sudo dnf install testdisk

ЗапускTestDisk с правами суперпользователя:

sudo testdisk

Выбор диска, с которым возникли проблемы. Далее следовать пошаговым инструкциям на экране. Утилита проанализирует диск и предложит возможные способы восстановления разделов или данных.

* + - 1. Замена носителя и восстановление данных из резервной копии

В случае, когда носитель полностью вышел из строя и восстановление данных с помощью TestDisk невозможно, рекомендуется заменить неисправный диск на новый. После этого необходимо восстановить данные из созданных ранее резервных копий. Для этого подойдет любая созданная ранее резервная копия (например, с использованием инструментов резервного копирования, таких как rsync, Bacula или Tar). Подробнее о резервном копировании данных см. раздел 7.

Ключевые шаги при замене носителя и восстановлении данных:

* замените физический носитель (жесткий диск или SSD);
* загрузитесь с инсталляционного образа и установите ОС;
* после установки восстановите данные из резервной копии, скопировав их на новый носитель.

Восстановление из резервной копии позволяет минимизировать потери данных и ускоряет процесс восстановления Системы в случае серьезных сбоев.

* + 1. Восстановление загрузчика GRUB2 (MBR и EFI)

В случае повреждения загрузочной записи GRUB2 может потребоваться её восстановление вручную. Для выполнения восстановления рекомендуется использовать Live-образ ОС РОСА или совместимого дистрибутива Linux с доступом к сети и средствам управления разделами (например, GParted).

Порядок восстановления загрузчика:

1. загрузка с Live-носителя (USB или DVD) с возможностью запуска в Live-режиме (без установки ОС);
2. определение корневого раздела и загрузочного диска ОС. Для этого рекомендуется воспользоваться утилитой fdisk или графическим интерфейсом GParted, доступным в Live-среде. Утилита fdisk выводит список разделов командой:

fdisk -l

В результате будет выведен список всех доступных дисков и разделов. Требуется определить:

* корневой раздел, на котором установлена ОС РОСА (обычно это раздел типа Linux, например /dev/sda2);
* загрузочный диск, на который необходимо установить загрузчик GRUB2 (обычно это физический диск, например /dev/sda).

Полученные значения будут использоваться в дальнейших командах по монтированию и установке загрузчика:

1. Монтирование Системы:

sudo -i

mkdir /chroot

sudo mount /dev/sdXX /chroot

где sdXX необходимо заменить на имя корневого раздела.

При использовании файловой системы BTRFS:

sudo mount -o subvol=root /dev/sdXX /chroot

где sdXX необходимо заменить на имя корневого раздела.

1. Монтирование необходимых виртуальных файловых систем и конфигураций:

for i in dev proc sys etc/resolv.conf; do mount --bind /$i\ /chroot/$i; done

Если используется EFI-загрузка, необходимо также смонтировать переменные прошивки:

sudo mount --bind /sys/firmware/efi/efivars\ /chroot/sys/firmware/efi/efivars

1. Переход в chroot-окружение командой:

chroot /chroot

1. Монтирование всех разделов из fstab:

sudo mount -a

Это обеспечит доступ к дополнительным разделам, включая /boot и /boot/efi, если таковые имеются.

1. Установка загрузчика GRUB2 в главный загрузочный сектор:

sudo grub2-install /dev/sdX

где sdX необходимо заменить на имя загрузочного диска.

1. Обновление конфигурации загрузчика. После установки необходимо сгенерировать конфигурационный файл GRUB2:

sudo update-grub2

Это обеспечит добавление всех обнаруженных систем (например, Windows) в загрузочное меню.

1. Выход из окружения chroot и перезагрузка ОС:

exit

sync

reboot

Также есть альтернативный способ восстановления загрузчика GRUB2**:** можно воспользоваться утилитой grub2recovery, доступной в репозитории contrib. Для её установки нужно выполнить:

sudo dnf install grub2recovery

Подробности о применении утилиты доступны на официальном форуме: <https://forum.rosa.ru/viewtopic.php?f=58&t=10649>.

* + 1. Восстановление Системы из среды chroot

В случае повреждения Системы или необходимости выполнения административных операций без загрузки основной ОС может использоваться метод восстановления через среду Live (LiveCD/LiveUSB) с переходом в установленную Систему в режиме chroot. Для этого выполняют следующие шаги:

1. Загрузка в Live-среду с установочного носителя (например, USB-накопителя или DVD) в Live-режиме, выбрав пункт "Запустить без установки" или аналогичный.
2. Определение корневого раздела и загрузочного диска ОС. Для этого рекомендуется воспользоваться утилитой fdisk или графическим интерфейсом GParted, доступным в Live-среде. В терминале выполнить команду:

fdisk -l

В результате будет выведен список всех доступных дисков и разделов. Требуется определить:

* корневой раздел, на котором установлена ОС РОСА (обычно это раздел типа Linux, например /dev/sda2);
* загрузочный диск, на который необходимо установить загрузчик GRUB2 (обычно это физический диск, например /dev/sda).

Полученные значения будут использоваться в дальнейших командах по монтированию и установке загрузчика.

монтирование корневой файловой системы в каталог, подготовленный для chroot (во всех командах параметр sdXX необходимо заменить на имя коревого раздела):

mkdir /chroot

sudo mount /dev/sdXX /chroot

Если используется файловая система BTRFS, указать нужный подтом:

sudo mount -o subvol=@ /dev/sdXX /chroot

Подключить необходимые виртуальные файловые системы:

for i in dev sys proc; do mount --bind -v /$i /chroot/$i;\ done

Для обеспечения работы сетевых функций (например, получения пакетов или использования утилит) скопировать файл конфигурации DNS:

cp /etc/resolv.conf /chroot/etc/resolv.conf

1. Вход в установленную Систему с помощью команды:

chroot /chroot

После выполнения данной команды будет организован доступ к установленной Системе в режиме полного административного управления.

1. Выполнение всех необходимых операций в среде chroot, включая:

* изменение паролей;
* переустановка загрузчика;
* исправление конфигурационных файлов;
* установка или удаление пакетов.

1. Завершение работы и перезагрузка Системы.После завершения всех операций необходимо выйти из chroot по команде:

exit

Отключить файловые системы (при необходимости) и перезагрузить Систему командой:

reboot

После перезагрузки будет выполнена загрузка основной установленной Системы.

* 1. Перезагрузка и остановка Системы

Использование различных способов перезагрузки и остановки Системы позволяет гибко управлять её состоянием в зависимости от текущих задач. Выбор конкретного метода зависит от ситуации и уровня доступа пользователя.

Перезагрузка Системы может потребоваться для:

* применения обновлений Системы;
* изменения в конфигурационных файлах, которые применяются только после перезагрузки;
* устранения сбоев в Системе.
  + 1. Перезагрузка

Для выполнения перезагрузки через терминал воспользуйтесь одним из следующих способов:

* Команда, которая немедленно инициирует перезагрузку Системы:

sudo reboot

В случае зависания Системы можно использовать принудительную перезагрузку:

sudo reboot -f

Опция -f (--force) перезагружает Систему без учёта активных процессов и может использоваться в крайних случаях.

* Перезагрузка с использованием команды init, которая позволяет указать конкретный уровень выполнения (для перезагрузки используется уровень 6:

sudo init 6

* С использованием systemd, предпочтительно использовать следующую команду, которая обеспечивает корректную остановку сервисов перед перезагрузкой:

sudo systemctl reboot

* С использованием команды shutdown (подробное использование рассмотрено в разделе 3.9.3):

sudo shutdown

* + 1. Остановка

Остановка Системы может выполняться для её полного выключения или планового технического обслуживания.

Для остановки Системы через командную строку используйте одну из следующих команд:

* Завершение работы всех сервисов и выключение ПК:

sudo poweroff

* Остановка с использованием команды systemctl:

sudo systemctl poweroff

* Принудительное выключение Системы можно выполнить с помощью команды:

sudo poweroff -f

Используйте эту команду только при необходимости, так как она может привести к потере данных.

* Остановка Системы с помощью команды shutdown. Система завершит работу через 1 минуту:

sudo shutdown

При подключении по SSH используются те же команды, что и в локальном терминале. Например:

ssh user@hostname sudo reboot

В случае, если Система полностью зависла и не реагирует на команды, можно использовать аппаратные кнопки на корпусе:

* Кнопка перезагрузки (Reset): перезагружает Систему принудительно.
* Кнопка включения/выключения (Power): удерживайте кнопку несколько секунд для полного выключения Системы.
  + 1. Shutdown

Команда shutdown — один из самых удобных и гибких инструментов для управления выключением, перезагрузкой и остановкой Системы. В этом разделе мы рассмотрим основные варианты использования команды shutdown с примерами.

Общий синтаксис команды

sudo shutdown [OPTIONS] [TIME] [MESSAGE]

OPTIONS: Дополнительные параметры, указывающие тип действия (выключение, перезагрузка и т. д.).

TIME: Время, через которое произойдет действие. Это может быть указано как абсолютное время (в формате часы:минуты, например 14:30) или как относительное время (например, +5 для действия через 5 минут) или с аргументом now для немедленного выполнения действия.

MESSAGE: Сообщение, которое будет отправлено всем пользователям перед завершением работы.

Основные параметры команды shutdown и их использование:

* Немедленное выключение Системы (по умолчанию):

sudo shutdown now

* Для перезагрузки Системы используется опция -r:

sudo shutdown -r now

Также возможно указать время перезагрузки, например, перезагрузка через 5 минут:

sudo shutdown -r +5

Или в определённое время, например, в 22:00:

sudo shutdown -r 22:00

* Отмена запланированного завершения или перезагрузки Системы с помощью опции -c:

sudo shutdown -c

Также возможно отправить сообщение всем пользователям о том, что выключение отменено:

sudo shutdown -c "Перезагрузка отменена."

* Выключение Системы с остановкой всех процессов и питания с помощью опции -h:

sudo shutdown -h now

Или аналогичным способом указать время выключения питания. Например, через 10 минут:

sudo shutdown -h +10

* Перевод Системы в спящий режим (halt) без полного выключения питания (в зависимости от настроек Системы и оборудования):

sudo shutdown -H now

* 1. Обновление Системы
     1. Общие сведения

Пакеты обновлённой версии Системы размещаются в официальном репозитории разработчика. Список доступных репозиториев можно найти в установленной ОС в каталоге /etc/yum.repos.d/.

Загрузка пакетов обновлений осуществляется только из официальных репозиториев разработчика при наличии подключения к сети Интернет.

При загрузке пакетов обновлений из официального репозитория требуется использование персонализированного ключа, ограничивающего возможность получения обновлений третьими лицами. Персонализированные ключи предоставляются потребителю при получении лицензии на использование Системы.

При первом запуске ОС ключ будет запрошен в графическом интерфейсе программы обновления Системы. Также ключ может быть указан позже:

* через терминал: путем размещения его в файле /etc/dnf/vars/token. Для этого можно воспользоваться, например текстовым редактором nano, и открыть файл командой:

sudo nano /etc/dnf/vars/token

* в графическом режиме: по нажатию правой кнопкой мыши на значок светофора в системном лотке, в контекстном меню выбрать "Установить ключ репозитория".
  + 1. Индикация обновлений

Система имеет наглядный графический инструмент для отображения информации об обновлениях – значок-индикатор светофор  в системном лотке:

* зелёный цвет — обновления отсутствуют;
* жёлтый цвет — найдены пакеты, требующие обновления;
* красный цвет — невозможно проверить наличие обновлений.

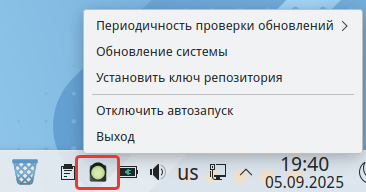


Рисунок 15 – Индикатор обновлений (светофор) и его контекстное меню

Для принудительной проверки обновлений Системы необходимо нажать правой кнопкой мыши на иконку светофора и в открытом контекстном меню (рисунок 15) выбрать "Обновление системы", после чего будет инициализирован процесс проверки обновлений пакетов. По завершению проверки выводиться информационное о окно о наличии или отсутствии пакетов с обновлениями.

Когда индикатор светофора горит желтым () по наведению на него курсором мыши можно увидеть список пакетов, требующих обновления.

Для настройки периодичности проверки обновлений в ОС необходимо правой кнопкой мыши нажать на значок светофора, далее в контекстном меню выбрать "Периодичность проверки обновлений" и выбрать необходимый параметр:

* 15 минут;
* 1 час;
* 6 часов;
* 12 часов;
* при старте Системы.

Система обновлений учитывает схему питания и заряд батареи ноутбука.

* + 1. Порядок получения обновлений

Произвести обновление ОС возможно двумя путями:

* через терминал: необходимо запустить менеджер пакетов dnf и выполнить команду:

sudo dnf update

Данная команда проверяет наличие обновлений в репозиториях и устанавливает доступные пакеты.

* в графическом режиме: через контекстное меню индикатора обновлений (значок светофора ) ⟶ "Обновление системы", после чего Система перейдет к проверке доступных пакетов и их установке. Прогресс будет отображаться в информационном окне (рисунок 16).

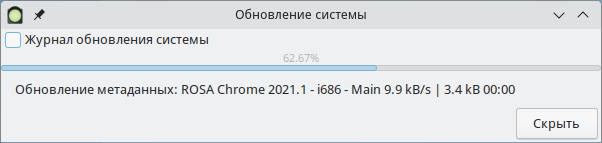


Рисунок 16 – Прогресс обновления Системы

* 1. Сохранение вывода консоли в файл

В процессе администрирования нередко возникает необходимость сохранить результаты выполнения команд терминала в файл – это может потребоваться для диагностики, документирования, пересмотра выполненных операций или подготовки отчётов.

**Перенаправление вывода в файл**

Для записи вывода команды в файл используется оператор перенаправления:

необходимая\_команда parameters > logfile.txt

Если файл logfile.txt существует, его содержимое будет перезаписано. Для добавления новых данных к существующему содержимому следует использовать двойной оператор >>:

необходимая\_команда parameters >> logfile.txt

**Одновременный вывод на экран и в файл**

Для одновременного отображения результата команды в терминале и записи в файл применяется утилита tee:

необходимая\_команда parameters | tee -a logfile.txt

Опция -a указывает на добавление данных в конец файла без перезаписи.

**Снимок экрана консоли с помощью setterm**

Утилита setterm позволяет сохранить содержимое одной страницы виртуальной консоли:

setterm -dump -file /путь/к/файлу

Пример:

setterm -dump -file /root/screenlog

Содержимое текущей виртуальной консоли будет сохранено в указанный файл. Опции утилиты:

* -dump — создаёт дамп консоли;
* -file имя\_файла — задаёт имя файла для сохранения;
* -append — добавляет данные к уже существующему файлу.

Для указания номера консоли используется форма:

setterm -dump [N] -file /root/screenlog

где N — номер консоли (например, 1 для /dev/tty1).

**Использование утилиты script**

Утилита script предназначена для записи всего, что происходит в терминале, включая команды и их вывод:

script --timing=terminal-timing.log terminal.log

После запуска запись начинается немедленно. Все действия будут сохранены в файл terminal.log, а временные характеристики — в terminal-timing.log.

Для завершения записи используется команда:

exit

Для воспроизведения записанного сеанса применяются команды:

scriptreplay terminal-timing.log terminal.log

или

scriptreplay terminal-timing.log terminal.log 2

где число в конце обозначает скорость воспроизведения (2 — в два раза быстрее и т.д.).

**Копирование текста с помощью утилиты screen**

Менеджер окон screen позволяет копировать данные из буфера прокрутки:

1. запустить screen:

screen

1. перейти к выполнению нужных команд;
2. войти в режим копирования нажатием комбинации клавиш: Ctrl+A, затем Ctrl+[;

установить курсор на начало, нажать Пробел, затем переместить в конец и снова нажать Пробел;

1. переключиться на новое окно: Ctrl+A, затем: c;
2. перейти в новое окно: Ctrl+A, затем: 1;
3. вставить текст: открыть редактор (например, mc), нажать Ctrl+A, затем Ctrl+];
4. сохранить текст в редакторе;
5. вернуться в предыдущее окно: Ctrl+A, затем: 0.

**Горячие клавиши в Konsole и Yakuake**

В эмуляторах терминала Konsole и Yakuake предусмотрена встроенная функция сохранения вывода. По умолчанию используется сочетание клавиш Ctrl+Shift+S

Также можно изменить глубину буфера прокрутки в настройках. Стандартное значение — 1000 строк.

* 1. Запись экрана из консоли

При необходимости записи видеоролика с рабочего стола без использования графических программ может быть использована утилита ffmpeg, работающая в командной строке и поддерживающая большое количество аудио- и видеоформатов.

FFmpeg устанавливается из стандартного репозитория командой:

sudo dnf install ffmpeg

Для записи видео с экрана в формате QuickTime (MOV) без звука используется следующая команда:

ffmpeg -r 50 -g 500 -s 1366x768 -f x11grab -i :0.0 -vcodec\ qtrle screencast.mov

где параметры:

* -f x11grab — источник захвата, вывод X-сервера;
* -s 1366x768 — размер захватываемой области;
* -r 50 — частота кадров в секунду;
* -g 500 — интервал между ключевыми кадрами;
* -i :0.0 — экран X-сервера (дополнительно можно указать смещение: -i :0.0+10,20);
* -vcodec qtrle — видеокодек QuickTime RLE;
* screencast.mov — имя выходного файла.

Примечание – Параметры -r и -g подбираются эмпирически, в зависимости от производительности Системы и цели записи.

Для записи видео со звуком необходимо определить устройство захвата аудио, выполнив команду:

arecord -l

Пример использования FFmpeg с записью звука:

ffmpeg -f alsa -ac 1 -i hw:2,0 -acodec pcm\_s16le -f x11grab\ -s 1366x768 -r 50 -g 500 -i :0.0 -vcodec qtrle screencast.mov

где параметры:

* -f alsa — захват звука через ALSA;
* -ac 1 — количество аудиоканалов (1 — моно, 2 — стерео);
* -i hw:2,0 — аудиоустройство (card 2, device 0);
* -acodec pcm\_s16le — кодек без сжатия;
* остальные параметры аналогичны описанным в предыдущем примере команды.
  + 1. Перекодирование видео

В случае необходимости сжатия или изменения формата видеозаписи возможно использование следующих команд:

* Простое перекодирование (по умолчанию):

ffmpeg -i screencast.mov screencast2.avi

* Кодирование с использованием libx264 в формат MKV:

ffmpeg -i screencast.mov -vcodec libx264 -crf 22 -threads 0\ -acodec libmp3lame -ar 44100 -ab 128k screencast2.mkv

* Кодирование с использованием libxvid в формат AVI:

ffmpeg -i screencast.mov -vcodec libxvid -qscale 8 -acodec\ libmp3lame -ab 128k -ac 2 screencast2.avi

где параметры:

* -crf — качество видео (чем меньше значение, тем выше качество);
* -qscale — аналогично crf, значение от 1 до 31 (меньше — лучше качество);
* -ac 2 — запись стерео;
* -threads 0 — использование всех доступных потоков CPU.

1. Аутентификация пользователей и управление доступом
   1. Общие сведения о PAM

PAM (Pluggable Authentication Modules) — это модульная архитектура, предназначенная для реализации механизмов аутентификации, авторизации, управления сессиями и паролями в ОС РОСА "ХРОМ". Система PAM обеспечивает унифицированный программный интерфейс между прикладными программами (например, sshd, login, su, sudo, gdm и др.) и механизмами проверки подлинности пользователей.

Ключевое преимущество PAM заключается в возможности централизованного и гибкого управления политиками безопасности без необходимости изменения исходного кода прикладного программного обеспечения. Все изменения поведения системы аутентификации выполняются путём редактирования конфигурационных файлов PAM, что повышает управляемость и расширяемость Системы.

PAM реализуется как набор библиотек, подключаемых динамически и обрабатывающих запросы на вход пользователя. Сценарий выполнения зависит от результатов работы каждого модуля, конфигурации порядка их подключения и используемой логики (control flags).

* 1. Механизм работы PAM

Работа системы PAM строится вокруг понятия стека аутентификации, в рамках которого выполняется последовательный вызов определённых модулей в установленном порядке. Для каждого прикладного сервиса (например, sshd, login, sudo, gdm-password) создаётся отдельный конфигурационный файл в каталоге /etc/pam.d/.

Каждая строка в конфигурационном файле имеет следующую структуру:

<тип> <контроль> <модуль> [аргументы]

где:

* **тип (type)** — указывает назначение модуля:
* auth — проверка подлинности пользователя (например, ввод пароля или PIN-кода);
* account — проверка политики доступа и допустимости входа (например, срок действия учётной записи, членство в группе);
* password — управление паролем (например, смена, проверка сложности, политика смены);
* session — инициализация и завершение пользовательской сессии (например, монтирование ресурсов, логирование активности).
* **контроль (control flag)** — определяет, как результат выполнения модуля влияет на общее решение (разрешить или запретить доступ):
* required — модуль должен выполниться успешно; при ошибке вход будет запрещён, даже если другие модули завершатся успешно;
* requisite — аналогично required, но при ошибке выполнение следующих модулей будет прекращено;
* sufficient — при успешном выполнении модуля дальнейшая проверка может быть пропущена, если это допускается конфигурацией;
* optional — результат выполнения модуля игнорируется, если он не является единственным в своей категории;
* include — подключение правил из другого PAM-файла;
* substack — подключение стека из другого файла с сохранением логики типа.
* **модуль** — путь к библиотеке или её имя (если библиотека находится в стандартном каталоге /lib/security/ или /lib64/security/);
* **аргументы** — параметры, передаваемые модулю (например, try\_first\_pass, nullok, debug, inactive=N и т.д.).

Пример записи:

auth required pam\_unix.so try\_first\_pass nullok

Данная строка означает, что при аутентификации (auth) обязательно требуется успешное выполнение модуля pam\_unix, использующего локальные пароли пользователей. Модуль будет использовать ранее введённый пароль (если он уже есть в стеке PAM) и разрешит вход без пароля, если поле пароля пустое (nullok).

* 1. Иерархия и приоритетность конфигурации PAM

В ОС РОСА "ХРОМ" используются два уровня конфигурации PAM:

* Общие шаблоны, которые располагаются в файлах:
* /etc/pam.d/system-auth;
* /etc/pam.d/password-auth.

Эти файлы содержат общие правила аутентификации, применяемые по умолчанию во всех сервисах через директивы include system-auth или include password-auth. Изменения, внесённые в эти файлы, распространяются на большинство сервисов.

* Специфические настройки для сервисов. Каждый сервис может переопределить или дополнить правила PAM в своём отдельном файле, например:
* /etc/pam.d/login — консольный вход (TTY);
* /etc/pam.d/sshd — вход по SSH;
* /etc/pam.d/sudo — выполнение команд от имени другого пользователя;
* /etc/pam.d/gdm-password — вход через графический интерфейс GDM.

При наличии противоречий между общими и локальными файлами приоритет имеет локальный файл конкретного сервиса.

Пример включения общих правил:

auth include system-auth

Эта директива указывает PAM использовать те же правила аутентификации, что и в файле /etc/pam.d/system-auth.

* 1. Конфигурация распространённых модулей PAM
     1. pam\_unix.so

Модуль pam\_unix.so является базовым модулем для аутентификации и управления паролями с использованием локальной базы данных пользователей /etc/passwd и /etc/shadow.

Модуль pam\_unix.so обеспечивает стандартную проверку пароля путём сравнения введённого значения с хешем, хранящимся в /etc/shadow. При смене пароля также используется модуль pam\_unix.so, если в стеке PAM он подключён в секции password. Модуль поддерживает использование различных параметров, позволяющих более гибко управлять политиками безопасности, например: ограничением на использование пустых паролей, принудительным использованием одного и того же пароля в цепочке модулей и др.

Основные параметры модуля:

* nullok — разрешение входа при пустом пароле;
* try\_first\_pass — попытка использовать уже введённый пароль из предыдущего модуля;
* use\_authtok — обязательное использование пароля, полученного от предыдущего модуля (обычно используется в стеке password);
* shadow — чтение паролей из /etc/shadow (по умолчанию включено).

Пример использования:

auth sufficient pam\_unix.so try\_first\_pass nullok

account required pam\_unix.so

password sufficient pam\_unix.so use\_authtok

* + 1. pam\_env.so

Модуль pam\_env.so используется для задания и инициализации переменных окружения в сеансах пользователей при аутентификации через PAM. Он позволяет определить, какие переменные среды должны быть установлены, а также откуда они должны быть загружены.

Конфигурационные файлы модуля pam\_env.so:

* /etc/security/pam\_env.conf — основной файл, содержащий переменные окружения в формате "ключ=значение". Может содержать как статические переменные, так и значения, полученные из текущего окружения.
* /etc/environment — альтернативный источник переменных окружения; используется как вспомогательный механизм. Поддерживает только простые объявления переменных без логики и условий.

Модуль pam\_env.so выполняется до выполнения остальных PAM-модулей и позволяет задать окружение, которое будет использоваться в рамках сеанса пользователя. Это может быть полезно, например, для установки языковых локалей (LANG, LC\_ALL), прокси-переменных, путей к пользовательским скриптам и т. д.

Конфигурационный файл /etc/security/pam\_env.conf поддерживает:

* комментарии (начинаются с "#");
* подстановки переменных (например, ${home});
* управление значениями на основе текущего окружения.
  + 1. pam\_faillock.so

Модуль pam\_faillock.so отвечает за реализацию механизма блокировки учётной записи после нескольких неудачных попыток входа. Рекомендуется для защиты от атак перебора паролей.

Блокировка применяется как для локального входа (через TTY, GDM), так и для удалённого (через SSH), если PAM настроен соответствующим образом. Модуль сохраняет информацию о неудачных попытках входа и отслеживает их накопление для каждой учётной записи.

Блокировка через pam\_faillock.so действует только на вход через PAM; вызовы, обходящие PAM, не учитываются.

По умолчанию модуль использует файл /var/log/faillock для хранения состояния.

Модуль работает в двух режимах:

* preauth — анализирует количество предыдущих неудачных попыток до выполнения основной аутентификации;
* authfail — регистрирует неудачную попытку после того, как пользователь ввёл неверный пароль.

При достижении заданного количества попыток доступ блокируется на определённый промежуток времени (unlock\_time) или до административной разблокировки.

Основные параметры:

* preauth — активация проверки до ввода пароля;
* authfail — регистрация неудачной попытки после неверного пароля;
* deny=N — максимальное количество попыток перед блокировкой (по умолчанию: 3);
* unlock\_time=T — продолжительность блокировки в секундах;
* fail\_interval=T — время, за которое считаются попытки (по умолчанию: 900);
* even\_deny\_root — распространение блокировки на пользователя root;
* audit — запись в системный журнал о блокировках и разблокировках;
* silent — скрытие вывода о количестве оставшихся попыток;
* local\_users\_only — применение только к локальным пользователям;
* no\_log\_info — отключение дополнительной информации в выводе.

Администратор может вручную удалить информацию о блокировках, тем самым разблокировав пользователя. Пример команды:

faillock --user <имя\_пользователя> --reset

Просмотр статистики неудачных попыток:

faillock --user <имя\_пользователя>

Пример конфигурации /etc/pam.d/system-auth для блокировки учетной записи пользователя после N неудачных попыток входа:

auth required pam\_faillock.so preauth silent deny=5 unlock\_time=900

auth required pam\_unix.so

auth [default=die] pam\_faillock.so authfail deny=5 unlock\_time=900

account required pam\_faillock.so

В данном примере при пяти неудачных попытках вход блокируется на 15 минут.

* + 1. pam\_limits.so

Модуль pam\_limits.so отвечает за применение ограничений на ресурсы пользователей (число процессов, размер памяти, количество открытых файлов и др.).

Ограничения применяются в соответствии с конфигурационными файлами limits.conf и дополнительными файлами из каталога /etc/security/limits.d/.

Ограничения вступают в силу при открытии новой сессии, поэтому необходимо убедиться, что pam\_limits.so подключён в секции session соответствующего PAM-файла (например, login, sshd, su, gdm-password и др.).

Ограничения могут устанавливаться по следующим ресурсам:

* core – максимальный размер файла core dump (в блоках);
* data – максимальный размер сегмента данных процесса;
* fsize – максимальный размер создаваемого файла (в блоках);
* memlock – максимальный объём заблокированной памяти;
* nofile – максимальное количество открытых файлов;
* nproc – максимальное количество процессов пользователя;
* rss – максимальный объём оперативной памяти;
* stack – максимальный размер стека;
* cpu – лимит времени процессора (в минутах);
* maxlogins – максимальное количество одновременных логинов;
* priority – приоритет (nice) запускаемых процессов;
* locks – максимальное количество блокировок файлов;
* sigpending – максимальное количество ожидающих сигналов;
* msgqueue – максимальный объём POSIX-очередей сообщений;
* nice – максимально допустимое значение nice;
* rtprio – максимальный приоритет реального времени.

Каждая строка в файлах имеет следующий формат:

<домен> <тип> <ресурс> <значение>

* <домен> — имя пользователя, группа (@groupname) или спец. символ \* (все пользователи);
* <тип> — soft (мягкое ограничение), hard (жёсткое ограничение) или "-" (оба сразу);
* <ресурс> — название ограничиваемого ресурса;
* <значение> — числовое значение ограничения или unlimited.

Настройки ограничений применяются со следующим приоритетом и порядком:

* настройки из /etc/security/limits.d/\*.conf дополняют или переопределяют правила из limits.conf при совпадении домена и ресурса;
* приоритет имеют правила, указанные позже (например, в алфавитно последнем файле из limits.d);
* если для одного пользователя задано несколько значений одного ресурса, используется более строгое (жёсткое) ограничение.

Некоторые ограничения (например, nofile) могут быть переопределены средствами оболочки или системным демоном (например, systemd user services). В таких случаях требуется дополнительная настройка через systemd (директивы LimitNOFILE= и т.п.).

При отладке рекомендуется использовать команду ulimit -a для просмотра текущих применённых ограничений к сессии.

Примеры конфигурации модуля:

* Ограничение количества процессов:

@developers soft nproc 2048

@developers hard nproc 4096

* Ограничение количества одновременно открытых файлов:

user1 - nofile 1024

* Принудительное окончание сессии и ограничение пользователя одной активной сессией (например, для терминального сервера). Изменения вносятся в файл /etc/security/limits.d/10-maxlogins.conf:

\* hard maxlogins 1

* + 1. pam\_succeed\_if.so

Модуль pam\_succeed\_if.so используется для условного выполнения последующих или текущих PAM-модулей в зависимости от заданных условий. Применяется для реализации гибкой логики доступа, например, разрешения или запрета входа определённым группам пользователей, пользователям с конкретными идентификаторами (UID), оболочками (shell), типами сервисов и т.д. Это особенно полезно при создании дифференцированных правил аутентификации в различных сценариях: GDM, TTY, SSH и других.

Модуль pam\_succeed\_if.so не выполняет саму аутентификацию, а лишь влияет на ход выполнения PAM-стека, пропуская или блокируя модули по заданным условиям.

Основные параметры модуля:

* user ingroup <имя\_группы> — проверка, состоит ли пользователь в указанной группе;
* uid < N / uid >= N — проверка числового идентификатора пользователя (например, для различия между системными и обычными пользователями);
* shell = /bin/bash — проверка используемой оболочки пользователя;
* service = gdm — проверка PAM-сервиса (имя PAM-конфигурационного файла), в рамках которого вызывается модуль.

Модуль поддерживает как положительные, так и отрицательные условия, а также логические операторы. Для управления логикой перехода в стеке используются параметры управления, например:

* [default=1 success=ignore] — при выполнении условия переход к следующему модулю, в противном случае — пропуск следующего модуля;
* [success=ok default=bad] — возврат успешного статуса при выполнении условия, иначе — неудача.

Пример использования для ограничения входа только пользователям, состоящим в группе tokenlogin, используется следующая строка в PAM-конфигурации:

auth [default=1 success=ignore] pam\_succeed\_if.so user ingroup\ tokenlogin

auth requisite pam\_deny.so

В данном примере, если пользователь не входит в группу tokenlogin, переход к следующей строке будет невозможен, выполнение pam\_deny.so завершится отказом в доступе.

Данное правило может быть добавлено, например, в начало файла /etc/pam.d/system-auth или /etc/pam.d/login.

* + 1. pam\_p11.so

Модуль pam\_p11.so предназначен для аутентификации пользователей с использованием аппаратных токенов (по стандарту PKCS#11). Используется в рамках реализации двухфакторной аутентификации в ОС РОСА "ХРОМ". pam\_p11.so позволяет выполнять проверку подлинности за счёт криптографической подписи, формируемой закрытым ключом, хранящимся на токене.

Особенности работы модуля:

* Модуль взаимодействует с аппаратным токеном напрямую через PAM и не зависит от наличия службы sshd или ключей openssh.
* Применяется как для локального входа, так и в составе универсальных PAM-стеков, например, для GDM, SSH, TTY и других служб.
* В сценарии двухфакторной аутентификации модуль инициирует запрос PIN-кода от токена и проверяет криптографическую подпись случайной последовательности, тем самым реализуя второй фактор.

Подробные инструкции по подготовке токена, настройке целевой и рабочей станции, а также интеграции pam\_p11.so с различными сервисами приведены в разделе 4.7 Настройка двухфакторной аутентификации.

* 1. Блокирование неактивных учётных записей

Для обеспечения безопасности и минимизации риска использования забытых или устаревших учётных записей модуль pam\_lastlog позволяет реализовать блокирование входа пользователей, не осуществлявших вход в Систему в течение определённого количества дней.

* + 1. Блокирование при входе через GDM

Для включения блокировки неактивных учётных записей при входе через графический менеджер входа (по умолчанию используется GDM) требуется внести изменения в файл /etc/pam.d/password-auth. Для этого необходимо:

1. открыть файл для редактирования с правами суперпользователя:

sudo nano /etc/pam.d/password-auth

добавить в него строку:

auth required pam\_lastlog.so inactive=90

между строками:

auth required pam\_env.so

и:

auth sufficient pam\_unix.so try\_first\_pass nullok

также добавить строку:

account required pam\_lastlog.so inactive=90

перед строкой:

account required pam\_unix.so

Параметр inactive=90 указывает, что вход будет заблокирован, если последний вход пользователя был выполнен более 90 суток назад. При необходимости значение может быть изменено.

* + 1. Блокирование при входе через терминал (TTY, SSH)

Для включения аналогичного механизма при входе через терминал или SSH необходимо внести аналогичные изменения в файл /etc/pam.d/system-auth. Для этого необходимо:

1. открыть файл для редактирования:

sudo nano /etc/pam.d/system-auth

добавить в файл строку:

auth required pam\_lastlog.so inactive=90

между строками:

auth required pam\_env.so

и

auth sufficient pam\_unix.so try\_first\_pass likeauth nullok

также добавить строку:

account required pam\_lastlog.so inactive=90

перед строкой:

account required pam\_unix.so

Данная конфигурация блокирует доступ к Системе пользователям, чья учётная запись не использовалась в течение указанного срока.

* + 1. Просмотр справки по модулю pam\_lastlog

Для получения дополнительной информации о возможных параметрах модуля pam\_lastlog рекомендуется использовать встроенную справку:

man pam\_lastlog

* 1. Отладка и проверка работы PAM

Корректная работа PAM имеет критическое значение для обеспечения доступа к Системе. Ошибки в конфигурации PAM-файлов могут привести к блокировке входа для всех пользователей, включая администратора. Поэтому любые изменения в конфигурации должны сопровождаться обязательной проверкой и, по возможности, тестироваться в отдельной сессии.

* + 1. Просмотр системного журнала

Большинство PAM-модулей, включая pam\_unix, pam\_faillock, pam\_lastlog, pam\_p11, записывают диагностические сообщения в системный журнал. Для просмотра сообщений, связанных с PAM, рекомендуется использовать команду:

sudo journalctl -b | grep pam

Для анализа конкретного модуля, например pam\_p11, используется команда:

sudo journalctl -b | grep pam\_p11

Параметр -b ограничивает вывод сообщениями текущей загрузки Системы.

* + 1. Проверка конфигурации и тестирование изменений

Перед внесением изменений в основные PAM-файлы (например, /etc/pam.d/system-auth, /etc/pam.d/password-auth, /etc/pam.d/sshd) рекомендуется, предварительно обязательно создав резервную копию изменяемого файла:

sudo cp /etc/pam.d/system-auth /etc/pam.d/system-auth.bak

1. создать отдельную тестовую сессию (например, новую вкладку в терминале TTY или подключение по SSH от другого пользователя), не завершая активную административную сессию;
2. выполнить проверку работы входа или сервиса с новой конфигурацией.

Если вход становится невозможен, следует восстановить резервную копию из открытой сессии:

sudo cp /etc/pam.d/system-auth.bak /etc/pam.d/system-auth

* + 1. Использование временного доступа через root-tty

Для минимизации рисков полной потери доступа к Системе рекомендуется:

* оставлять root-доступ через терминал TTY активным (файл /etc/pam.d/login);
* не изменять правила PAM для sudo, su и login одновременно;
* иметь доступ к Системе через физическую консоль или загрузку в однопользовательском режиме (recovery mode), если доступ по SSH будет заблокирован.
  + 1. Примеры отладочной информации от модулей

Для повышения информативности можно включить режим отладки в отдельных модулях:

* В модуле pam\_unix:

auth required pam\_unix.so debug

* В модуле pam\_faillock:

auth required pam\_faillock.so preauth silent deny=5\ unlock\_time=900 debug

Сообщения с пометкой pam\_unix(debug) или pam\_faillock(debug) будут отображаться в journalctl.

* + 1. Проверка порядка выполнения модулей

Для анализа последовательности выполнения PAM-модулей можно использовать повышенное логирование, активируемое на уровне службы (например, sshd). Для этого в файле /etc/ssh/sshd\_config необходимо добавить или изменить параметр:

LogLevel DEBUG

после чего перезапустить службу:

sudo systemctl restart sshd

В journalctl будут отображаться дополнительные сведения о ходе аутентификации.

* 1. Настройка двухфакторной аутентификации

Настройка двухфакторной (многофакторной) аутентификации в ОС выполняется с использованием программного модуля pam\_p11 и аппаратного криптографического устройства (токена) Рутокен ЭЦП PKI.

Рассматривается разделение на рабочую станцию администратора — устройство, с которого выполняется настройка токена, и целевую станцию — устройство, на котором осуществляется вход в ОС с использованием второго фактора. В некоторых случаях обе роли могут быть совмещены на одном устройстве.

Первым фактором аутентификации выступает наличие устройства с уникальным закрытым ключом, вторым фактором — знание PIN-кода, ассоциированного с данным устройством.

* + 1. Подготовка рабочих станций

Подготовка рабочих станций осуществляется в несколько этапов:

1. Для Систем на базе платформы ROSA2021.1 и новее требуется установка следующих пакетов:

sudo dnf install task-smartcards pam\_p11

При установке пакета task-smartcards автоматически создаётся группа tokenlogin. Пользователи, входящие в данную группу, проходят аутентификацию с использованием второго фактора. Остальные пользователи проходят аутентификацию по паролю;

1. После установки пакета task-smartcards необходимо выполнить активацию службы pcscd.socket:

sudo systemctl enable --now pcscd.socket

В версиях ОС, начиная с 12.3, данная служба активирована по умолчанию. Повторная активация не приведёт к негативным последствиям.

1. Далее необходимо провести проверку и при необходимости настройку политики polkit. Пакет pcsc-lite содержит политику org.debian.pcsc-lite.policy, разрешающую доступ к токенам всем пользователям. При необходимости допускается создание пользовательской политики в каталоге /etc/polkit-1/. В большинстве случаев дополнительная настройка не требуется.
   * 1. Подготовка токена Рутокен

Подготовка токена осуществляется в несколько этапов:

1. Подключение устройства. Токен Рутокен ЭЦП PKI должен быть подключён к рабочей станции администратора. Устройство должно определяться в Системе, например:

Bus 001 Device 005: ID 0a89:0030 Aktiv Rutoken ECP

1. Очистка данных с токена производится с помощью команды:

pkcs15-init --erase-card -p rutoken\_ecp

1. Создание нового контейнера PKCS#15 осуществляется командой:

pkcs15-init --create-pkcs15 --so-pin "87654321" --so-puk ""

где 87654321 — PIN администратора.

1. Присвоение контейнеру пользовательского PIN (12345678) и PIN администратора (87654321):

pkcs15-init --store-pin --label "User PIN" --auth-id 02 –pin\ "12345678" --puk "" --so-pin "87654321" --finalize

1. Генерация ключевой пары (RSA длиной 2048 бит) с помощью команды:

pkcs15-init -G rsa/2048 --auth-id 02 --id 42 --label "SSH\ User's Key" --public-key-label "SSH User's Public Key"

1. При выполнении команды будет запрошен PIN пользователя:

User PIN [User PIN] required. Please enter User PIN [User\ PIN]:

Необходимо ввести заданный ранее пароль 12345678.

* + 1. Извлечение и перенос публичного ключа

При необходимости можно использовать токен в качестве второго фактора аутентификации при входе по SSH. Описанные далее действия можно выполнить на любой из станций.

Для использования токена при входе по SSH необходимо извлечь публичный ключ:

ssh-keygen -D /usr/lib64/opensc-pkcs11.so -I 0:42 >> key.pub

Содержимое файла key.pub требуется добавить в файл ~/.ssh/authorized\_keys соответствующего пользователя на целевой станции.

Файл ~/.ssh/authorized\_keys используется PAM-модулем pam\_p11 независимо от наличия и настроек сервера OpenSSH.

* + 1. Подготовка целевой станции

Подготовка целевой станции осуществляется в несколько этапов.

На целевой станции требуется установка модуля pam\_p11:

sudo dnf install pam\_p11

Добавление пользователей в группу tokenlogin (группа автоматически создается при установке пакета task-smartcards) выполняется следующими действиями:

1. Добавление локальных пользователей:

sudo usermod -a -G tokenlogin user

где вместо user используется имя целевого пользователя.

1. Добавление доменных пользователей с использованием механизма сопоставления групп, который реализуется с помощью пакета libnss-role.

Данный механизм позволяет считать члена одной группы (например, из домена) одновременно членом другой — локальной.

Предполагается, что на контроллере домена (Microsoft Active Directory / Samba AD либо FreeIPA) имеется группа с именем "Token Login". В случае корректного включения станции с установленной ОС РОСА "ХРОМ" в домен, Система будет видеть принадлежность доменного пользователя к данной группе.

Для активации механизма сопоставления необходимо выполнить установку пакета:

sudo dnf install libnss-role

В процессе установки модуль role будет автоматически добавлен в строку "group" конфигурационного файла /etc/nsswitch.conf.

Далее следует определить соответствие между доменной группой и локальной:

sudo roleadd 'Token Login' tokenlogin

В результате в файл /etc/role будет добавлена соответствующая запись. После этого при выполнении команды id доменного пользователя в выводе отразится его принадлежность как к группе "Token Login", так и к локальной группе tokenlogin.

* + 1. Список ключей пользователя

Для обеспечения возможности входа в Систему с использованием второго фактора (токена) необходимо добавить открытый ключ, извлечённый из токена, в файл ~/.ssh/authorized\_keys соответствующего пользователя.

При подключённом токене требуется выполнить следующую команду:

ssh-keygen -D /usr/lib64/opensc-pkcs11.so -I 0:42

Результат выполнения команды представляет собой строку, содержащую открытый ключ. Данный ключ необходимо внести в файл ~/.ssh/authorized\_keys пользователя, которому предполагается разрешение входа с использованием токена.

Процедура выполняется от имени данного пользователя и включает следующие шаги:

1. создание каталога .ssh (при отсутствии):

mkdir --mode=700 -p ~/.ssh

1. открытие файла ~/.ssh/authorized\_keys на редактирование:

nano ~/.ssh/authorized\_keys

вставка строки с открытым ключом, полученным ранее, и сохранение файла.

Файл ~/.ssh/authorized\_keys используется PAM-модулем pam\_p11 как при удалённом входе по SSH, так и при локальном входе через терминал. Наличие соответствующего открытого ключа в этом файле является обязательным условием для успешной аутентификации с использованием токена.

* + 1. Создание общего файла настроек PAM

Для организации централизованной конфигурации аутентификации с использованием токена рекомендуется создать отдельный файл /etc/pam.d/p11, содержащий настройки, специфичные для работы PAM-модуля pam\_p11.

Создание и редактирование файла выполняется следующим образом:

sudo nano /etc/pam.d/p11

В файл необходимо внести следующую конфигурацию:

auth required pam\_env.so

#auth required pam\_faillock.so preauth silent deny=5\ unlock\_time=900

auth required pam\_p11.so /usr/lib64/opensc-pkcs11.so

#auth required pam\_faillock.so authfail deny=5\ unlock\_time=900

password optional pam\_p11.so /usr/lib64/opensc-pkcs11.so

После внесения изменений файл следует сохранить (в редакторе nano для этого используется комбинация клавиш Ctrl+O), затем нажать Enter и выйти по Ctrl+X.

Если планируется использование модуля pam\_faillock для защиты от подбора PIN-кода, необходимо удалить символ "#" в начале соответствующих строк, тем самым активировав их (раскомментировать).

В дальнейшем при необходимости следует указать подключение данного файла (стека p11) в других PAM-конфигурациях, например, в файлах login, sshd или gdm-password, посредством директивы substack p11.

* + - 1. Особенности работы pam\_p11

Если ни один из ранее вызванных модулей PAM не запросил пароль, pam\_p11 инициирует его запрос и использует введённый PIN-код для доступа к закрытому ключу на токене. Если же пароль уже получен предыдущим модулем, pam\_p11 использует его как PIN-код. В конфигурациях, приведённых в данном разделе руководства, модуль pam\_p11 всегда вызывается первым и самостоятельно инициирует ввод PIN-кода.

* + 1. Расширенная настройка

Далее в разделах 4.7.7.1–4.7.7.6 рассматриваются типовые примеры настройки PAM для использования второго фактора аутентификации. При необходимости допускается реализация более сложных и специфичных конфигураций PAM-модулей.

Для ознакомления с официальной справочной информацией по формату конфигурационных файлов PAM рекомендуется использовать встроенную документацию:

man pam.conf

* + - 1. Просмотр логов

После выполнения попытки входа в Систему возможно получение диагностической информации о работе PAM-модулей, в том числе pam\_p11, посредством системного журнала:

sudo journalctl -b | grep pam\_p11

Вывод ограничивается сообщениями текущего сеанса загрузки Системы. При необходимости фильтрация может быть адаптирована под другие модули или условия.

* + - 1. Настройка входа в TTY

Настройка, при которой токен используется исключительно при входе в Систему через терминал TTY (переключение с помощью сочетания клавиш Alt+Ctrl+F[N]), выполняется в конфигурационном файле /etc/pam.d/login.

Для внесения изменений необходимо открыть файл с правами суперпользователя:

sudo nano /etc/pam.d/login

Затем внести следующие изменения в файл:

1. Замена строки "auth include system-auth" на блок:

auth [default=1 success=ignore] pam\_succeed\_if.so user\ ingroup\ tokenlogin

auth substack p11

auth [default=ignore success=1] pam\_succeed\_if.so user\ ingroup\ tokenlogin

auth substack system-auth

Замена строки "password include system-auth" на блок:

password [default=1 success=ignore] pam\_succeed\_if.so user\ ingroup tokenlogin

password substack p11

password [default=ignore success=1] pam\_succeed\_if.so user\ ingroup tokenlogin

password substack system-auth

После сохранения изменений рекомендуется выполнить вход через терминал TTY под пользователем, для которого ранее был настроен ключ SSH. При возникновении проблем следует обратиться к журналу Системы, как описано в п. 4.7.7.1.

* + - 1. Настройка входа через GDM

Для настройки входа через графический дисплей-менеджер GDM требуется изменение файла /etc/pam.d/gdm-password.

Необходимо открыть файл на редактирование с правами суперпользователя (например, с помощью редактора nano):

sudo nano /etc/pam.d/gdm-password

Далее внести в файл /etc/pam.d/gdm-password следующие изменения:

1. закомментировать строки (добавить символ "#" в начало строки):

auth substack password-auth

auth optional pam\_gnome\_keyring.so

добавить после закомментированных строк следующий блок:

auth [default=1 success=ignore] pam\_succeed\_if.so user\ ingroup tokenlogin

auth substack p11

auth [default=ignore success=1] pam\_succeed\_if.so user\ ingroup tokenlogin

auth substack password-auth

аналогично закомментировать следующие строки:

password substack password-auth

-password optional pam\_gnome\_keyring.so use\_authtok

**Следует обратить внимание**, что отключение модуля pam\_gnome\_keyring необходимо в данном случае, поскольку его назначение — автоматическая разблокировка пользовательского хранилища ключей при совпадении пароля от хранилища с системным паролем пользователя — теряет смысл при использовании PIN-кода и токена.

добавить после них блок:

password [default=1 success=ignore] pam\_succeed\_if.so user\ ingroup tokenlogin

password substack p11

password [default=ignore success=1] pam\_succeed\_if.so user\ ingroup tokenlogin

password substack system-auth

сохранить изменения в файле и выполнить вход в Систему через GDM. При корректной настройке вместо стандартного пароля будет запрошен PIN-код от токена.

* + - 1. Настройка входа по SSH

При входе в Систему по SSH с использованием токена в качестве второго фактора аутентификации токен должен быть физически подключён к рабочей станции, с которой производится подключение, а на целевой станции должен быть зарегистрирован только открытый ключ.

Содержимое открытого ключа, извлечённого с токена, должно быть добавлено в файл ~/.ssh/authorized\_keys соответствующего пользователя на целевой станции. Порядок извлечения ключа приведён в п. 4.7.3 Извлечение и перенос публичного ключа.

* + - 1. Вход по SSH с вводом PIN-кода и пароля

Настройка целевой станции

Рассматривается конфигурация, при которой при входе по SSH пользователь сначала вводит PIN-код токена, а затем — свой пароль. В данном варианте PAM-модуль pam\_p11 **не требуется.**

Для активации данного режима необходимо запретить вход по паролю, разрешив только комбинацию открытого ключа и интерактивной аутентификации, командой:

echo "AuthenticationMethods publickey,keyboard-interactive"\ | sudo tee -a /etc/ssh/sshd\_config

Также необходимо убедиться, что параметр UsePAM в том же файле установлен в значение yes (данное значение задано по умолчанию в ОС):

UsePAM yes

При необходимости скорректировать значение и перезапустить службу SSH:

sudo systemctl restart sshd

Вход с рабочей станции

На рабочей станции, с которой выполняется подключение, должен быть установлен и подключён токен. Подключение осуществляется по следующей команде:

ssh -I /usr/lib64/opensc-pkcs11.so -p <порт>\ <пользователь>@<ip\

-адрес>

где параметры:

* <порт> — номер порта, на котором запущен sshd на целевой станции (по умолчанию – 22);
* <пользователь> — имя пользователя на целевой станции;
* <ip-адрес> — IP-адрес целевой станции.

После выполнения команды будет последовательно запрошено:

* PIN-код от токена;
* пароль пользователя.
  + - 1. Вход по SSH только с вводом PIN-кода

Настройка целевой станции

В данной конфигурации используется модуль pam\_p11, а пользователь выполняет вход исключительно по PIN-коду без ввода системного пароля:

1. открыть на редактирование PAM-конфигурацию для sshd:

sudo nano /etc/pam.d/sshd

1. закомментировать строку (добавить символа "#" в начало строки):

auth include system-auth

1. вставить вместо неё блок:

auth [default=1 success=ignore] pam\_succeed\_if.so user ingroup\ tokenlogin

auth substack p11

auth [default=ignore success=1] pam\_succeed\_if.so user ingroup\ tokenlogin

auth substack system-auth

аналогично закомментировать строку:

password include system-auth

вставить вместо неё блок:

password [default=1 success=ignore] pam\_succeed\_if.so user\ ingroup tokenlogin

password substack p11

password [default=ignore success=1] pam\_succeed\_if.so user\ ingroup tokenlogin

password substack system-auth

убедиться, что параметр UsePAM в /etc/ssh/sshd\_config установлен в значение "yes".

Вход с рабочей станции

Токен должен быть подключён к рабочей станции, с которой производится вход. Подключение выполняется командой:

ssh -I /usr/lib64/opensc-pkcs11.so -p <порт>\ <пользователь>@<ip\  
-адрес>

После этого будет запрошен только PIN-код. При корректной настройке вход будет выполнен без запроса системного пароля пользователя.

1. Сетевые подключения

После установки ОС, при наличии активного проводного соединения, Система автоматически подключается к сети TCP/IP с помощью NetworkManager — программного обеспечения для управления сетевыми соединениями. Это происходит автоматически, если в сети имеется сервер DHCP, который назначает IP-адреса.

Для управления сетевыми подключениями в ОС используются несколько сервисов и утилит, которые позволяют произвести операции как через графический, так и консольный интерфейсы. Для работы с проводными соединениями широко используется NetworkManager. Если соединение с сетью обеспечивается с помощью беспроводного соединения, его можно легко настроить с помощью псевдографического интерфейса nmtui, также доступен консольный интерфейс nmcli.

* 1. Способы настройки сетевых соединений
     1. NetworkManager

NetworkManager — это основной инструмент управления сетевыми подключениями в Системе, который представляет собой систему, управляющую сетевыми интерфейсами и соединениями, обеспечивающая их автоматическую настройку. NetworkManager поддерживает настройку как проводных, так и беспроводных соединений:

* при наличии подключенного сетевого кабеля NetworkManager автоматически подключает Систему к сети, используя настройки, предоставленные DHCP-сервером. Если требуется статическая настройка IP-адреса, это можно сделать через графический интерфейс или командную строку.
* при работе с Wi-Fi-сетями NetworkManager предоставляет возможность удобного выбора и подключения к доступным беспроводным сетям.

Настройка параметров NetworkManager осуществляется через графический интерфейс утилиты, который можно открыть несколькими способами:

* меню "Параметры Системы → Сеть и связь → Соединения";
* на нижней панели задач правой кнопкой мыши нажмите на иконку сети → (рисунок 17) → "Настроить сетевые соединения → Сеть и связь"

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Рисунок 17 – Сетевые подключения

* + 1. Использование nmtui

nmtui — это псевдографический интерфейс для управления сетевыми соединениями в текстовом режиме. Он предоставляет удобный способ настройки сетевых интерфейсов без необходимости работы с графическим интерфейсом.

Для запуска nmtui откройте терминал и выполните команду

nmtui

Откроется текстовый интерфейс (рисунок 18 ), в котором можно выбрать нужное действие, например подключение к сети или настройку нового соединения.

Изображение выглядит как текст, снимок экрана, дисплей, Шрифт

Автоматически созданное описание

Рисунок – Стартовый интерфейс утилиты nmtui

* Подключение к уже настроенным в Системе сетям осуществляется в разделе "Активировать соединение", где из списка можно выбрать доступное сетевое соединение и подключиться к нему.
* Создание нового соединения осуществляется в разделе "Изменить подключение", где можно настроить новое соединение, сначала выбрав необходимый тип соединения, затем указав параметры для проводного или беспроводного интерфейса (рисунок 19).

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 19 – Создание нового подключения в nmtui

* Для того, чтобы задать персонализированное имя хосту выберите соответствующий параметр "Настроить системное имя хоста" и далее в открывшимся окне задайте новое имя (рисунок 20).

Изображение выглядит как текст, снимок экрана, Цвет электрик, дисплей

Автоматически созданное описание

Рисунок 20 – Изменение имени хоста

* + 1. Использование nmcli

nmcli — это командная утилита для управления NetworkManager из командной строки. Она предоставляет полный контроль над сетевыми интерфейсами и соединениями.

Для просмотра доступных сетевых интерфейсов и соединений выполняют команду:

nmcli device status

Для подключения к проводной сети используется команда:

nmcli con up id <имя\_соединения>

Для подключения к беспроводной сети используется команда:

nmcli device wifi connect <SSID> password <пароль>

Для создания нового соединения можно использовать команду, указав все необходимые параметры:

nmcli con add

Для получения полной справки по использованию утилиты и ее параметрах используйте команду:

nmcli --help

* + 1. Использование ifconfig и ip

Для отображения информации о сетевых интерфейсах и их текущем состоянии используют команду:

ifconfig

Данная команда может использоваться и без прав root, но, если необходимо получить привилегии суперпользователя, выполните:

sudo ifconfig

Результатом выполнения команды станет получение списка имеющихся сетевых подключений и их параметров (рисунок 21).

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 21 – Вывод команды ifconfig

Наряду с ifconfigрекомендуется использовать команду ip, которая предоставляет более гибкий и расширенный функционал по сравнению с ifconfig, что делает её предпочтительным инструментом для управления сетевыми настройками в современных системах. Описание основных параметров и опций команды ip приведены ниже (таблица 222).

Таблица 2 - Описание параметров и опций команды ip

|  |  |  |
| --- | --- | --- |
| Наименование | Команда | Описание |
| Отображение информации о сетевых интерфейсах | ip link show | Отображает список всех сетевых интерфейсов на компьютере, включая их состояние и атрибуты |
| Отображение информации о сетевых адресах | ip addr show | Показывает IP-адреса и другие параметры сетевых интерфейсов, такие как маска подсети и статус |
| Назначение IP-адреса интерфейсу | sudo ip addr add <IP-адрес>/<маска\_подсети> dev <имя\_интерфейса> | Назначает указанный IP-адрес или маску подсети указанному сетевому интерфейсу |
| Удаление IP-адреса с интерфейса | sudo ip addr del <IP-адрес>/<маска\_подсети> dev <имя\_интерфейса> | Удаляет указанный IP-адрес или маску подсети с указанного сетевого интерфейса |
| Управление маршрутами | ip route show | Показывает текущие маршруты сети. Можно добавлять, изменять и удалять маршруты с помощью команд:  ip route add  ip route change  ip route del |
| Управление VLAN | sudo ip link add link <родительский\_интерфейс> name <vlan\_интерфейс> type vlan id <VLAN\_ID> | Создает виртуальный сетевой интерфейс VLAN на основе родительского интерфейса с указанным идентификатором VLAN |
| Просмотр IP-адреса сетевой карты | ip a hostname -I | Дает возможность просмотреть IP-адреса сетевой карты |

Примечание – В таблице приведены лишь некоторые из основных возможностей команды ip. Команда ipтакже поддерживает множество других опций и параметров, которые могут быть использованы для более детальной настройки и управления сетевыми подключениями в ОС.

Чтобы получить полную информацию по команде ip, можно открыть справку с помощью команды:

man ip

* + 1. Виртуальный сетевой интерфейс Loopback

Сетевой интерфейс Loopback (lo) представляет собой виртуальный сетевой интерфейс, который используется для тестирования сетевого стека Системы и для обеспечения локальной сетевой связности без необходимости подключения к физической сети.

В основном Loopback используется для тестирования сетевого стека ОС и отладки приложений, серверов, а также для работы локальных сервисов, таких как веб-серверы или базы данных. Поскольку пакеты, отправленные на адрес Loopback, не покидают компьютер, они полностью обрабатываются внутри сетевого стека. Это позволяет проверить, насколько правильно работает сетевая подсистема.

Использование интерфейса Loopback позволяет установить локальную сетевую связность без необходимости физического подключения к сети. Это полезно, например, для запуска локальных сетевых служб, которые должны быть доступны только локально.

Интерфейс Loopback также может использоваться для обеспечения безопасности и управления доступом. Некоторые службы и приложения могут быть настроены на прослушивание только на Loopback, чтобы они были доступны только локально и недоступны извне.

Основные характеристики Loopback интерфейса:

* интерфейс lo всегда присутствует в Системе и автоматически создается при старте ОС;
* стандартный IP-адрес, назначаемый этому интерфейсу — 127.0.0.1, который также известен как localhost. Этот адрес зарезервирован и используется исключительно для внутреннего общения Системы;
* обычно назначаемая маска подсети — 255.0.0.0, что указывает на широкую подсеть класса A.

Основные команды для работы с Loopback интерфейсом представлены ниже (таблица 3).

Таблица 3 - Описание параметров работы с интерфейсом Loopback

|  |  |  |
| --- | --- | --- |
| Наименование | Команда | Описание |
| Просмотр состояния интерфейса | ip addr show lo | Команда отображает текущее состояние интерфейса lo, включая его IP-адрес и маску подсети |
| Включение интерфейса | sudo ip link set lo up | Активирует интерфейс loopback, если он по каким-то причинам был отключен. |
| Отключение интерфейса | sudo ip link set lo down | Отключает интерфейс lo, что крайне редко требуется, так как его отключение может нарушить работу локальных сервисов. |
| Проверка соединения с localhost | ping 127.0.0.1 | Команда ping позволяет убедиться, что loopback интерфейс функционирует нормально. В случае успеха вы увидите сообщения с откликами от 127.0.0.1 |

* + 1. Сетевой интерфейс ens\*

Сетевые интерфейсы, обозначаемые как ens\*, являются стандартными именами для Ethernet-интерфейсов в современных Linux-системах. Эти имена заменили устаревшие обозначения типа eth0, eth1 и т.д. Структура имени ens\* основывается на физическом расположении устройства в Системе, что делает его уникальным и независимым от порядка загрузки устройств.

Формат имени ens\* обозначает Ethernet-интерфейс, где

* en указывает на тип подключения (Ethernet);
* s\* — на номер слота или порта, к которому подключен интерфейс.

Интерфейсы ens\* используются для проводных сетевых подключений. Они работают с различными типами сетей TCP/IP, обеспечивая взаимодействие компьютера с локальными сетями и Интернетом.

Для просмотра конфигурации всех ens\*интерфейсов используется команда:

ip addr show ens\*

В выводе команды будет представлена такая информация как IP-адрес подключений, состояние (включен или выключен), маску подсети и прочие параметры.

Для активации интерфейса ens\* и приведения его в состояние, готовое для использования команда:

ip link set ens\* up

Отключениеинтерфейса:

ip link set ens\* down

Назначение статического IP-адреса интерфейсу ens\*. Маска /24 обозначает стандартную маску подсети 255.255.255.0. IP-адреса:

sudo ip addr add 192.168.1.10/24 dev ens\*

Проверкасостояния соединения осуществляется командой:

ethtool ens\*

В выводе команды будет выведена низкоуровневая информация об интерфейсе, включая состояние физического соединения, скорость линка и поддержку различных режимов работы.

В Системе интерфейсы ens\* автоматически получают IP-адрес от DHCP-сервера при подключении, если не настроены на использование статического IP.

Для проверки текущего статуса DHCP или обновления IP-адреса можно использовать команду:

dhclient ens\*

Эта команда запрашивает у DHCP-сервера новый IP-адрес для интерфейса ens\*.

* 1. Настройка статических соединений

Для статической настройки всех соединений в сетевом интерфейсе можно воспользоваться одним из следующих способов:

* Через графический интерфейс в меню "Параметры Системы → Сеть и связь → Соединения" обычно заполняются такие параметры, как: "выбор DNS-сервера", "Домен поиска", "Метод настройки сети" (в данном случае – вручную), "IP-адрес", "Маска сети" и "Шлюз" (рисунок 22).

Изображение выглядит как текст, программное обеспечение, снимок экрана, Значок на компьютере

Автоматически созданное описание

Рисунок 22 – Настройка сетевых соединений в графическом интерфейсе ОС

* для того чтобы прописать такие же параметры в консоли, необходимо выполнить команду со следующими параметрами:

sudo nmcli connection modify enp0s3 connection.autoconnect yes ipv4.method manual ipv4.dns 192.168.1.100 ipv4.dns-search rosa.lan ipv4.addresses 192.168.1.200/24 ipv4.gateway 192.168.1.1

где:

* connection modify – изменение соединения на интерфейсе enp0s3;
* connection.autoconnect yes – поднятие соединения при загрузке Системы;
* ipv4.method manual – задание соединения как статического;
* ipv4.dns – задание IP-адреса DNS-сервера;
* ipv4.dns-search – задание домена поиска;
* ipv4.addresses – задание IP-адреса и маски "/24" интерфейса;
* ipv4.gateway – задание IP-адреса шлюза;

Затем перезапустить интерфейс командами:

sudo nmcli connection down enp0s3

sudo nmcli connection up enp0s3

Можно использовать альтернативные команды для перезапуска:

nmcli device disconnect enp0s3

nmcli device connect enp0s3

* + 1. Настройка статического IP-адреса

Для некоторых сетей может потребоваться использование статического IP-адреса. Это можно сделать как через nmtui, так и через nmcli.

* Использование nmtui: в открытом интерфейсе nmtui выберите "Изменить соединение", далее выберите нужное соединение и установите статический IP-адрес, маску подсети и шлюз.
* Использование nmcli: для того, чтобы задать статический IP-адрес и шлюз для выбранного соединения используйте команду:

nmcli con mod <имя\_соединения> ipv4.addresses <IP/маска> ipv4.gateway <шлюз>

* 1. Настройка динамических соединений

Чтобы настроить все соединения динамически в сетевом интерфейсе, необходимо выполнить следующие действия:

1. в графическом интерфейсе ОС перейти в меню "Параметры системы → Сеть и связь → Соединения"; обычно заполняются такие параметры, как "Метод настройки сети" (в данном случае – автоматически), "другие DNS-серверы", "Домены поиска", "Идентификатор клиента DHCP".
2. Для того чтобы прописать все параметры в консоли, нужно выполнить команду:

sudo nmcli connection modify enp0s3 ipv4.method auto ipv4.ignore-auto-dns no

где:

* connection modify – изменение соединения на интерфейсе enp0s3;
* ipv4.ignore-auto-dns – задание параметра no, таким образом не игнорируются полученные по dhcp серверы DNS;
* ipv4.method auto – задание соединения динамическим.

Затем перезапустить интерфейс командами:

sudo nmcli connection down enp0s3

sudo nmcli connection up enp0s3

* + 1. Настройка динамических соединений (кроме DNS)

При необходимости настроить все соединения динамически в сетевом интерфейсе, за исключением DNS, следует придерживаться следующих действий:

1. в NetworkManager обычно заполняются такие параметры как: "Метод настройки сети" (в данном случае – автоматически (только адреса)), "другие DNS-серверы", "Домены поиска", "Идентификатор клиента DHCP", "DNS";
2. для того чтобы прописать все параметры в консоли, выполнить следующую команду:

sudo nmcli connection modify enp0s3 ipv4.ignore-auto-dns yes ipv4.dns 192.168.1.100

где:

* connection modify – изменение соединения на интерфейсе enp0s3;
* ipv4.ignore-auto-dns – задание параметра yes, таким образом игнорируются полученные по dhcp серверы DNS;
* ipv4.dns – прописываем IP-адрес нашего DNS-сервера;

1. перезапустить интерфейс командами:

sudo nmcli connection down enp0s3

sudo nmcli connection up enp0s3

* 1. Переадресация

Переадресация сетевых пакетов (форвардинг) — это процесс передачи сетевых пакетов от одного сетевого интерфейса к другому, что позволяет Системе функционировать как маршрутизатор. Включение переадресации в Системе может быть полезно для настройки сетевых шлюзов, маршрутизаторов или при организации общего доступа к интернет-соединению.

Переадресация IP отключена в Системе по умолчанию, и для ее включения требуется наличие дополнительного файла. Если такого файла нет, нужно его создать. Имя файла состоит из имени сетевого интерфейса, за которым следует суффикс ".network"`, например "tun0.network" для сетевого интерфейса "/dev/tun0". При этом другие расширения игнорируются.

Далее в созданном файле необходимо внести изменения. Для примера рассмотрим настройку сетевого интерфейса "/dev/tun0" в файле "tun0.network". Он состоит из двух разделов: Match и Network. В разделе Match указывают имя сетевого интерфейса, а в разделе Network включают "IP Forwarding":

# cat /etc/systemd/network/tun0.network

[ Match ]

Name = tun0

[ Network ]

IPForward = ipv4

Чтобы проверить текущее значение функции net.ipv4.ip\_forward нужно выполнить команду:

cat /proc/sys/net/ipv4/ip\_forward

Если команда возвращает значение 0, значит функция IPForward выключена, если 1 – включена.

Также можно проверить значение через команду sysctl:

sysctl net.ipv4.ip\_forward

net.ipv4.ip\_forward = 0

* + 1. Включение переадресации IPv4

Для включения переадресации IPv4 необходимо изменить соответствующие настройки в системных параметрах. Основной параметр, отвечающий за переадресацию, находится в /proc/sys/net/ipv4/ip\_forward. Чтобы включить переадресацию, выполните следующую команду:

echo 1 > /proc/sys/net/ipv4/ip\_forward

Эта команда активирует переадресацию пакетов в Системе до момента ее перезагрузки, позволяя Системе пересылать их между сетевыми интерфейсами.

Чтобы сделать это изменение постоянным (переживающим перезагрузку Системы), необходимо добавить или изменить строку в файле /etc/sysctl.conf:

grep forward /etc/sysctl.conf

net.ipv4.ip\_forward = 1

Затем примените изменения с помощью команды:

sysctl -p

* + 1. Включение переадресации IPv6

Аналогично можно включить переадресацию для IPv6. Для этого следует использовать команду:

echo 1 > /proc/sys/net/ipv6/conf/all/forwarding

Чтобы сделать изменение постоянным, добавьте строку в файл /etc/sysctl.conf:

net.ipv6.conf.all.forwarding = 1

* + 1. Настройка правил переадресации с использованием nftables

С помощью nftables можно создать правила, управляющие перенаправлением пакетов между сетевыми интерфейсами.

Пример настройки NAT (Network Address Translation) для переадресации пакетов с внутреннего интерфейса ens33 на внешний ens34:

nft add table ip nat

nft add chain ip nat postrouting { type nat hook postrouting priority 100 \; }

nft add rule ip nat postrouting oifname "ens34" masquerade

nft add chain ip nat prerouting { type nat hook prerouting priority 0 \; }

nft add chain ip nat input { type nat hook input priority 0 \; }

nft add chain ip nat output { type nat hook output priority 100 \; }

nft add chain ip nat forward { type nat hook forward priority 0 \; }

Эти команды создают таблицу и правила, которые позволяют устройствам из внутренней сети (подключенным через ens33) выходить в интернет через интерфейс ens34.

После настройки переадресации можно использовать команду для проверки активных правил и диагностирования возможных проблем:

nft list ruleset

Для мониторинга сетевого трафика и подтверждения работы переадресации можно использовать утилиты, такие как tcpdump.

* + - 1. Настройка динамических соединений (кроме DNS)

При необходимости настроить все соединения динамически в сетевом интерфейсе, за исключением DNS, следует придерживаться следующих действий:

1. В графической оболочке NetworkManager обычно заполняются такие параметры как: "Метод настройки сети" (в данном случае – автоматически (только адреса)), "другие DNS-серверы", "Домены поиска", "Идентификатор клиента DHCP, DNS";
2. Для того чтобы прописать все параметры в консоли, выполнить следующую команду:

sudo nmcli connection modify enp0s3 ipv4.ignore-auto-dns yes ipv4.dns 192.168.1.100

где:

* connection modify – изменение соединения на интерфейсе enp0s3;
* ipv4.ignore-auto-dns – задание параметра yes, таким образом игнорируются полученные по dhcp серверы DNS;
* ipv4.dns – прописываем IP-адрес нашего DNS-сервера;

1. перезапустить интерфейс командами:

sudo nmcli connection down enp0s3

sudo nmcli connection up enp0s3

* 1. Мониторинг и диагностика сети

NetworkManager и утилиты командной строки также предоставляют инструменты для мониторинга состояния сети и диагностики проблем.

Для отображения текущего состояния NetworkManager:

nmcli general status

Для просмотра детальной информации о сетевых интерфейсах используется команда:

nmcli device show

Для диагностики проблем с сетевыми соединениями используются стандартные утилиты ping и traceroute.

Утилита ping предназначена для проверки доступности узлов в сети и измерения времени отклика между ними. Она отправляет ICMP-пакеты на указанный адрес и ожидает ответа. Использование команды:

ping <адрес\_хоста>

Команда отображает результаты в виде времени отклика и процента потерь пакетов, что помогает определить, доступен ли узел и имеются ли сетевые проблемы, такие как высокая задержка или потеря пакетов.

Утилита traceroute используется для диагностики маршрута, по которому проходят пакеты до указанного узла. Она выводит список промежуточных узлов (роутеров), через которые проходят пакеты. Это помогает выявить, на каком этапе возникают сетевые проблемы.

Команда для выполнения трассировки:

traceroute <адрес\_хоста>

Вывод покажет последовательность узлов и время, затраченное на прохождение пакетов через каждый из них, что позволяет идентифицировать узкие места или проблемы в сети.

* 1. Межсетевой экран Nftables
     1. Общие сведения о Nftables

Стандартным сетевым экран в ОС принят стандарт использования Nftables, который заменил собой использование iptables.

Nftables – это более простая и эффективная альтернатива iptables/ip6tables/arptables. Она предлагает унифицированную обработку IPv4 и IPv6, а также улучшенную производительность и поддержку современных сетевых протоколов. Для nftables была разработана новая система фильтрации пакетов, добавлена пользовательская утилита ntf, а также создан слой совместимости с iptables/ip6tables. Файервол nftables использует набор хуков, систему отслеживания соединений, систему очередей и подсистему логирования netfilter.

Как и iptables, nftables построен на правилах, определяющих действия. Эти правила содержаться в таблицах (table), к которым прикреплены цепочки (chain). Цепочка может содержать набор правил и исполняется в хуках (hook) netfilter. Таблица специфична для каждого из протоколов (ARP/IPv4/IPv6). Одно из основных отличий от iptables заключается в том, что больше нет предопределенных таблиц и цепочек.

Описание правил nftables:

* Таблицы (**table**):
* table – это контейнер для цепочек (chain);
* Семейства таблиц (**family**):
* ip – цепочки, связанные с IPv4;
* ip6 – цепочки, связанные с IPv6;
* arp – цепочки, связанные с ARP;
* bridge – соединение связанных цепочек;
* inet – смешанные цепочки ipv4 / ipv6;
* netdev – цепочки, которые фильтруют в начале стека;
* Цепочки (**chain**):
* chain – группировка правил;
* base chain – цепочка с зарегистрированным хуком netfilter, типом и приоритетом;
* other chain – цепочка без привязки к хуку и по умолчанию без отслеживания трафика;

Существует три вида цепочек:

* filter – стандартный тип; фильтр пакетов;
* nat – протоколы IP/IPv6/INET; хуки PREROUTING, INPUT, OUTPUT, POSTROUTING; преобразование NAT;
* route – протоколы IP/IPv6/; хуки OUTPUT; маршрутизация;
* **Хук (hook)** – это представление (правило) от том, на каком этапе происходит обработка:
* prerouting – до принятия решения о маршрутизации; для пакетов, входящих в машину;
* input – для всех пакетов, входящих в локальную Систему;
* forward – для пересылаемых пакетов вне локальной Системы;
* output – для пакетов, исходящих из локальной Системы;
* postrouting – после принятия решения о маршрутизации; для пакетов, покидающих машину;
* priority – число, используемое для упорядочивания цепочек или для установки приоритетов между некоторыми операциями;
* policy – для пакетов, которые явно не приняты или отклонены в содержащихся правилах;
* Правила (**rules**):
* rule – определяют, какое действие необходимо выполнить для каких пакетов; правила прикреплены к цепочкам; каждое правило может иметь выражение для сопоставления пакетов и одно или несколько действий, выполняемых при сопоставлении;
* statements – представляют собой действие, которое должно быть выполнено при совпадении правила:
* accept – прием пакета и прекращение выполнения текущего правила;
* drop – отказ от приема пакета и прекращение выполнения текущего правила;
* reject – отклонение пакета с сообщением ICMP;
* queue – постановка пакета в очередь в пользовательское пространство и прекращение выполнения текущего правила;
* continue – продолжение выполнения текущего и переход к следующему правилу;
* return – возврат из текущей цепочки и переход к следующему правилу последней цепочки; в базовой цепочке это эквивалентно accept;
* jump – продолжение с первого правила; будет продолжено в следующем правиле после выдачи оператора возврата;
* goto – аналогично jump, но после новой цепочки оценка продолжится с последней цепочки вместо той, которая содержит оператор goto;
* handle – внутренний номер, идентифицирующий определенное правило;
* position – внутренний номер, используемый для вставки правил до определенного handle.
  + 1. Установка и запуск Nftables

В ОС по умолчанию уже установлен фреймворк nftables.

Если по каким-то причинам пакет подсистемы удален, nftables можно установить следующей командой:

sudo dnf install nftables

Запуск сервиса nftables осуществляется командой:

sudo systemctl start ntfables.service

Чтобы включить запуск при старте Системы, нужно воспользоваться командой:

sudo systemctl enable nftables.service

* + 1. Работа с Nftables

В качестве базового примера работы с nftables нужно выполнить следующие шаги для закрытия/открытия порта:

1. Создать таблицу с помощью команды:

sudo /usr/sbin/nft add table inet my\_table

Формат команды:

nft add table <family> <name>

где:

* <family> – один из протоколов ip/ip6/inet/arp/bridge/netdev;
* <name> – имя новой таблицы.

1. Проверить наличие новой таблицы:

sudo nft -a list ruleset

1. Добавить цепочку:

sudo /usr/sbin/nft add chain inet my\_table input { type filter hook input priority 0 \; policy accept \; }

Формат команды:

nft add chain <family> <name table> <name chain> {hook}

где:

* <family> – один из протоколов ip/ip6/inet/arp/bridge/netdev;
* <name table> – имя таблицы;
* <name chain> – имя новой цепочки;

1. Проверить наличие новой цепочки:

sudo /usr/sbin/nft -a list ruleset

1. Добавить правило, с помощью которого ограничивается доступ к порту 80:

sudo /usr/sbin nft add rule inet my\_table input tcp dport 80 drop

Формат правила:

nft add rule <family> <name table> <protocol> <dport> <port> <event>

где:

* <family> – один из протоколов nftable ip/ip6/inet/arp/bridge/netdev;
* <name table> – имя таблицы;
* <name chain> – имя новой цепочки;
* <protocol> – протокол TCP/UDP/ICMP и т.д.;
* <dport> – входящий порт (destination port);
* <port> – номер входящего порта;
* <event> – событие drop/accept/jump и т.д.

1. Соответственно для открытия порта 80 нужно выполнить:

sudo /usr/sbin nft add rule inet my\_table input tcp dport 80 accept

Но получить доступ к порту 80 после этого не удастся, так как в списке правил сначала все пакеты на порт 80 блокируются, а потом "пропускаются". В рассматриваемом случае пропускать уже нечего, так как изначально на этом порту всё блокируется.

В этом случае нужно удалить запрещающее правило с drop, для чего с помощью отметки "handle 2", которая присутствует в выводе на экран, выполнить команду:

sudo /usr/bin/nft delete rule inet my\_table input handle 2

Далее следует проверить результат:

sudo /usr/sbin/nft -a list ruleset

Теперь доступ к порту 80 есть в полном объеме.

Если необходимо полностью очистить таблицу, то следует выполнить:

sudo /usr/sbin/nft delete table inet my\_table

Если требуется полностью очистить файервол (все цепочки и таблицы), то следует выполнить:

sudo /usr/sbin/nft flush ruleset

1. Контроль доступа к файлам
   1. Описание технологии ACL

ACL (Access Control List, список контроля доступа) – это механизм в ОС, который позволяет более гибко управлять доступом к файлам и папкам, чем это позволяют права доступа по умолчанию. С ACL можно установить права доступа для конкретных пользователей и групп пользователей на конкретные файлы и папки.

ACL обычно состоит из двух типов прав: пользовательские права и групповые права. Пользовательские права позволяют устанавливать права доступа для конкретных пользователей, а групповые права – для групп пользователей.

Преимущество использования ACL заключается в том, что он позволяет установить более точные права доступа, чем это возможно с помощью стандартных прав доступа на основе пользователей и групп. Например, если есть файл, к которому нужен доступ только для нескольких пользователей, можно установить ACL для этого файла, чтобы ограничить доступ только для этих пользователей, не давая права доступа всем остальным пользователям.

Команды для работы с ACL в Linux включают getfacl и setfacl. Команда getfacl используется для просмотра текущих прав доступа ACL на файлы и папки, а команда setfacl – для изменения или добавления новых прав доступа ACL на файлы и папки.

Знание ACL может быть полезно для системных администраторов, которые хотят более тонко настраивать доступ к файлам и папкам на своих серверах.

В ОС списки ACL хранятся в расширенных атрибутах (extended attributes, xattr), которые являются дополнительными метаданными, хранящимися в файловой системе. Большинство файловых систем, в т.ч. ext4, BTRFS, XFS, поддерживают хранение ACL в xattr. Однако, не все файловые системы поддерживают ACL и хранение их в xattr. Например, файловая система FAT32 полностью не поддерживает ACL.

Права доступа к файлам и каталогам в ОС разделяются на три категории:

* Владелец (user): права доступа для пользователя, которому принадлежит файл.
* Группа (group): права доступа для группы пользователей, связанной с файлом.
* Остальные (others): права для всех остальных пользователей Системы.

Права доступа включают в себя:

* r (read) — чтение: пользователь может просматривать содержимое файла или список файлов в каталоге.
* w (write) — запись: пользователь может изменять содержимое файла или создавать, удалять и переименовывать файлы в каталоге.
* x (execute) — выполнение: пользователь может запускать файл на выполнение (если это исполняемый файл или скрипт) или переходить в каталог.

Права доступа также можно задать с помощью чисел. Каждому праву доступа соответствует определенное число:

* r = 4
* w = 2
* x = 1

Суммируя эти значения, можно задать конкретные права. Например:

* 7 (rwx) = 4 (чтение) + 2 (запись) + 1 (выполнение)
* 5 (r-x) = 4 (чтение) + 1 (выполнение)

Пакет с утилитами для работы с ACL уже предустановлен в ОС. При необходимости его можно установить из репозитория командой:

sudo dnf install acl

* 1. Примеры настройки ACL

Типовые примеры настройки списков доступа:

1. Предоставлениеправ конкретному пользователю:

Чтобы дать пользователю user1 права на чтение и выполнение файла выполните:

setfacl -m u: user1:rx файл

1. Настройкаправ для группы editors на запись в каталог:

setfacl -m g:editors:rwx каталог

1. Удаление записей ACL для файла:

setfacl -b файл

1. Установка прав по умолчанию для всех новых файлов в каталоге:

setfacl -m d:u:john:rwx каталог

1. Настройка ACL для существующих файлов. В качестве примера используются следующие исходных данные:

* каталог /media/buh (на поддерживающей ACL файловой системе: ext4, BTRFS, XFS и др.);
* группа пользователей buh (бухгалтерия);
* пользователи ivanova, petrova, sidorova, входящие в группу buh.

Необходимо всех пользователей из группы buh – ivanova, petrova, sidorova – наделись следующими правами:

* читать файлы в /media/buh;
* создавать новые файлы и каталоги;
* изменять существующие файлы.

Для настройки ACL для существующих файлов нужно выполнить следующие действия:

1. Выставить права на каталог /media/buh для группы buh, чтобы ее пользователи могли записывать и читать файлы в этом каталоге:

sudo setfacl -R -m g:buh:rwx /media/buh

Эта команда добавляет права для группы buh на папку /media/buh и все ее подкаталоги и файлы. Опция -R означает, что изменения должны применяться рекурсивно для всех подкаталогов и файлов. Опция -m означает, что изменяются существующие права. Параметр g:buh:rwx означает, что выставляются права для группы buh на чтение, запись и выполнение.

1. Проверить, что права настроены правильно, командой:

sudo getfacl /media/buh

В выводе на экран должно быть примерно следующее:

# file: /media/buh/

# owner: <owner>

# group: buh

user::rwx

group::r-x

other::r-x

default:user::rwx

default:group::r-x

default:group:buh:rwx

default:other::r-x

В этом выводе для группы buh выставлены права на чтение, запись и выполнение, и группа buh имеет права на чтение и запись файлов в папке /media/buh, все ее подпапки и файлы в них.

1. Автоустановка ACL для создаваемых файлов и папок. Ключ -d в команде setfacl позволяет установить ACL по умолчанию на вновь создаваемые файлы и папки в указанной директории.

Для того, чтобы установить права по умолчанию для группы buh на каталог /media/buh, нужно выполнить следующую команду:

sudo setfacl -R -d -m g:buh:rwx /media/buh

Здесь ключ -dуказывает, что устанавливается ACL по умолчанию, а ключ -m указывает, что изменяются существующие права. Аргумент g:buh:rwx задает права для группы buh на чтение, запись и выполнение. Теперь при создании новых файлов и подкаталогов в каталоге /media/buh они будут автоматически получать ACL с правами, выставленными по умолчанию.

Чтобы проверить, что ACL по умолчанию настроены правильно, нужно выполнить команду:

sudo getfacl /media/buh

В выводе должно быть примерно следующее:

# file: /media/buh/

# owner: <owner>

# group: buh

user::rwx

group::r-x

other::r-x

default:user::rwx

default:group::r-x

default:group:buh:rwx

default:other::r-x

В этом выводе строка "default:group:buh:rwx" указывает, что группа buh имеет права на чтение, запись и выполнение для всех вновь создаваемых файлов и подкаталогов в каталоге /media/buh.

1. Резервное копирование данных

Резервное копирование данных — это важный процесс для обеспечения безопасности и сохранности информации. В Системе для автоматизированного резервного копирования используются утилита rsync и таймеры (файлы с расширением .timer) systemd. Эта комбинация позволяет выполнять регулярные бэкапы файлов и каталогов с минимальной нагрузкой на Систему.

rsync — это мощный инструмент для синхронизации и копирования данных, который поддерживает локальные и удаленные копии, а также предоставляет функции инкрементного копирования. Утилита использует алгоритм, который минимизирует объем передаваемых данных, копируя только измененные файлы или части файлов.

Для организации эффективных процедур резервного копирования и восстановления данных рекомендуется:

* регулярно проверять и обновлять резервные копии;
* хранить резервные копии на разных физических носителях и в удаленных хранилищах;
* периодически тестировать восстановление данных для предотвращения проблем при реальных аварийных ситуациях;
* для хранения архивных копий конфиденциальных данных должны применяться организационные или технические средства предотвращения несанкционированного доступа.

Резервное копирование бывает нескольких видов в зависимости от объема данных, подлежащих сохранению, и особенностей процесса восстановления:

1. Полное резервное копирование (Full Backup). Этот тип копирования включает полное сохранение всех данных, независимо от того, изменялись они с последнего копирования или нет. Преимущества полного резервного копирования заключаются в простоте процесса восстановления, однако его выполнение требует больше времени и места на носителе.
2. Инкрементноерезервное копирование (Incremental Backup). Инкрементное копирование сохраняет только те данные, которые изменились после последнего резервного копирования (будь то полное или инкрементное). Это позволяет экономить место и время на создание копий, однако для восстановления всех данных потребуется несколько операций, так как нужно будет применить полную копию и все последующие инкрементные копии.
3. Дифференциальное резервное копирование (Differential Backup). Дифференциальное копирование сохраняет изменения, произошедшие с момента последнего полного резервного копирования. В отличие от инкрементного копирования, дифференциальное требует больше места по мере увеличения объема изменений, но процесс восстановления проще, так как требуется только полная копия и последняя дифференциальная копия.
4. Мирроринг (Mirroring).Это процесс создания точной копии данных в реальном времени на другом устройстве или сервере. Обычно используется для высоко доступных систем, где важно иметь постоянно актуальную резервную копию данных.
   1. **Планирование резервного копирования и восстановления данных**

При создании расписания резервного копирования важно определить, какие данные необходимо копировать, как часто это делать и на каких носителях будут храниться копии. План должен предусматривать возможность восстановления данных за любой период времени, например, для восстановления файла в состоянии, близком к текущему дню.

Для достижения этого могут применяться различные типы резервного копирования. Комбинация инкрементного и дифференциального резервного копирования позволяет достичь баланса между сохранением данных и эффективностью использования ресурсов. Полное копирование фиксирует все файлы системы, а обновляемое копирует только те, которые изменились с момента последнего сохранения.

При использовании многоуровневой схемы резервного копирования, например, полного копирования раз в месяц и инкрементного — раз в день, можно оптимизировать восстановление данных. Такая схема позволяет сократить время на ежедневное копирование, сохраняя при этом возможность точного восстановления данных.

Примечание – Расписание резервного копирования должно быть документировано и доступно пользователям, чтобы обеспечить прозрачность процесса и возможность восстановления.

Рекомендуется использовать схему резервного копирования "день–неделя–месяц" для сохранения трех копий данных (ежедневной, еженедельной и ежемесячной), что позволит избежать перезаписи поврежденных файлов.

План восстановления Системы — неотъемлемая часть плана резервного копирования. В случае аварийной ситуации важно знать, как быстро и полно можно восстановить Систему или отдельные файлы. В рамках плана следует определить местоположение носителей с резервными копиями, их доступность и устойчивость к повреждениям. Также важно предусмотреть, как резервные копии могут быть защищены от сбоев Системы, на которой они хранятся.

Для того чтобы план восстановления оставался надежным, необходимо регулярно проверять исправность носителей с резервными копиями. Это может включать выборочную проверку файлов или полное чтение копий после их создания. Также важно проводить тестирование восстановительных процедур, чтобы удостовериться, что в случае необходимости данные можно будет быстро и корректно восстановить.

Рекомендуется периодически тестировать носители и процедуру восстановления, чтобы убедиться в их работоспособности.

* 1. **Создание резервных копий**

Основные параметры утилиты rsync:

* -a (архивный режим): сохраняет структуру каталогов и права доступа.
* -v (verbose): выводит подробную информацию о процессе копирования.
* --delete: удаляет файлы в целевом каталоге, если они были удалены в исходном.
* -z: сжатие файлов при передаче;
* --progress: отображает прогресс копирования.

Пример команды для создания резервной копии каталога /home на внешний диск:

rsync -av --delete /home/ /mnt/backup/home/

где /mnt/backup/home/ — это целевой каталог, в который будут скопированы данные.

Автоматизация создания резервных копий в ОС осуществляется с использованием systemd. Для автоматического запуска задач резервного копирования используют планировщик systemd timers, который заменяет классический cron.

Для настройки заданий резервного копирования придерживайтесь следующей инструкции:

1. Создание unit-файл /etc/systemd/system/backup.service для выполнения задачи резервного копирования:

sudo nano /etc/systemd/system/backup.service

Содержимое файла должно быть следующим:

[Unit]

Description=Резервное копирование каталога /home

Wants=backup.timer

[Service]

ExecStart=/usr/bin/rsync -a --delete /home/ /mnt/backup/home/

[Install]

WantedBy=multi-user.target

где в строке Wants= прописывается имя таймер файла, который создается на следующем шаге. А в строке WantedBy= указывается уровень загрузки Системы, при котором будет работать данный .service файл.

1. Создание файла с расширением .timer и названием /etc/systemd/system/backup.timer для запуска сервиса:

sudo nano /etc/systemd/system/backup.timer

Содержимое файла должно быть следующим (для примера рассмотрим задание, в котором копирование будет запускаться ежедневно):

[Unit]

Description=Таймер для запуска резервного копирования

Requires=backup.service

[Timer]

OnCalendar=daily

Persistent=true

Unit=backup.service

[Install]

WantedBy=timers.target

где в строке Requires= указывается имя созданного выше .service файла.и регламентируется то, что таймер запускается только при его наличии. И в строке Unit= также указывается имя созданного файла .service который будет запускаться при срабатывании таймера.

Параметр Persistent=true должен быть включен в содержание файла, т.к. он позволяет выполнять задачу после пропуска, если Система была выключена.

1. Запуск и активация таймера:

sudo systemctl start backup.timer

sudo systemctl enable backup.timer

Теперь резервное копирование будет выполняться автоматически согласно заданному расписанию.

Подробно о создании и настройке расписания для файла .timer systemd см. раздел 3.4.2.1.

Дополнительные параметры для более гибкой настройки резервного копирования:

1. Инкрементное резервное копирование, которое позволяет копировать только изменённые файлы. Для этого используется флаг --link-dest. Для проведения инкрементного копирования настройте systemd сервис и таймер-файлы следующим образом:

[Service]

ExecStart=/usr/bin/rsync -av --link-dest=/mnt/backup/home/previous /home/ /mnt/backup/home/current/

Файл .timer, например, настройте для ежедневного инкрементного резервного копирования в 1:00:

[Timer]

OnCalendar=\*-\*-\* 01:00:00

Persistent=true

1. Ограничение пропускной способности при резервном копировании. Чтобы предотвратить перегрузку сети, можно ограничить скорость передачи данных при проведении резервного копирования. Для этого в файле .service укажите следующие параметры:

[Service]

ExecStart=/usr/bin/rsync -av --bwlimit=5000 /home/ /mnt/backup/home/

где 5000 означает ограничение в 5000 КБ/с (5 МБ/с). Это полезно для резервного копирования в рабочее время, когда важно сохранять пропускную способность сети.

1. Задание для копирования сжатых файлов с использованием опции -z:

[Service]

ExecStart=/usr/bin/rsync -avz /home/ /mnt/backup/home/

Дополнительно должен быть также задан .timer файл для данного задания.

1. Настройка уведомления по электронной почте о завершении задания через systemd service:

[Service]

ExecStart=/usr/bin/rsync -av --delete /home/ /mnt/backup/home/

ExecStartPost=/usr/sbin/sendmail admin@example.com < /var/log/backup.log

Дополнительно должен быть также задан .timer файл для данного задания.

1. Настройка резервного копирования с условием запуска задания только при подключении внешнего диска:

[Service]

ExecStart=/bin/bash -c 'if mount | grep /mnt/backup; then rsync -av --delete /home/ /mnt/backup/home/; fi'

Дополнительно должен быть также задан .timer файл для данного задания.

1. Комплексное задание с ограничением скорости, сжатием и уведомлением по электронной почте:

[Service]

ExecStart=/usr/bin/rsync -avz --bwlimit=5000 --delete /home/ /mnt/backup/home/

ExecStartPost=/usr/sbin/sendmail admin@example.com < /var/log/backup.log

Дополнительно должен быть также задан .timer файл для данного задания.

1. Отчёт по результатам выполнения задания. Чтобы сохранить лог выполнения задания в отдельный файл, можно добавить следующую команду:

[Service]

ExecStart=/usr/bin/rsync -av --delete /home/ /mnt/backup/home/

StandardOutput=append:/var/log/backup.log

Логи будут записываться в файл /var/log/backup.log после каждого выполнения задания.

1. Для контроля процесса резервного копирования и проверки резервных копий можно проверять логи systemd:

journalctl -u backup.service

Для проверки целостности данных можно использовать флаг --dry-run:

rsync -av --dry-run /home/ /mnt/backup/home/

Флаг --dry-run позволяет симулировать процесс копирования без фактического изменения данных, чтобы увидеть, какие файлы будут обновлены.

* 1. **Восстановление данных из резервной копии**

Восстановление данных происходит обратным копированием из резервного хранилища. Для примера рассмотрим копирование данных каталога /home:

rsync -av /mnt/backup/home/ /home/

Этот процесс гарантирует восстановление всей информации в ее актуальном виде.

1. Контроль целостности Системы
   1. Общие принципы работы AIDE

AIDE – программа контроля целостности Системы – используется для защиты от вредоносных программ, вирусов и обнаружения несанкционированных действий. AIDE создает базу данных с информацией о файлах и каталогах (контрольная сумма, время изменения) и периодически сравнивает их текущее состояние с этой базой.

Это позволяет выявить нарушения целостности Системы и принять меры для восстановления измененных файлов из резервной копии или с образа восстановления Системы. AIDE помогает сократить время расследования инцидентов, сосредоточившись на файлах, которые были изменены.

Дополнительные возможности AIDE:

* поддержка различных атрибутов файлов, в том числе тип файла, inode, uid, gid, разрешения, количество ссылок, mtime, ctime и atime;
* поддержка сжатия Gzip, SELinux, XAttrs, Posix ACL и атрибутов файловой системы;
* поддержка различных алгоритмов, включая md5, sha1, sha256, sha512, rmd160, crc32 и т. д.;
* отправка уведомлений по электронной почте.
  1. Установка и создание базы данных

Установите пакеты AIDE из репозиториев в Систему с помощью команды:

sudo dnf install aide

Далее необходимо создает базу данных AIDE, которая будет хранить контрольные суммы, права доступа и другие атрибуты файлов и директорий. Сравнивая текущие данные с ранее созданной базой, администратор может легко обнаружить любые отклонения, что позволяет оперативно реагировать на подозрительные изменения.

База данных в AIDE создается с помощью команды:

sudo aideinit

В ходе создания БД будет запрошен пароль от GPG-ключа, которым будет подписываться база AIDE для контроля ее целостности. Затем будет выполнено первичное заполнение базы /var/lib/aide/aide.db, т.е. зафиксировано заведомо корректное состояние Системы. Проверка целостности ОС и обновление базы данных.

Также необходимо своевременно проводить обновление базы данных AIDE после внесения легитимных изменений в Систему (например, при установке обновлений), чтобы актуализировать её состояние. Для этого выполните команду:

sudo aide --update

Затем замените старую базу на новую:

mv /var/lib/aide/aide.db.new.gz /var/lib/aide/aide.db.gz

* + 1. Настройка конфигурационного файла

Для проведения более корректных проверок AIDE необходимо правильно настроить его конфигурационный файл, который находится по пути /etc/aide.conf. Этот файл определяет, какие файлы и каталоги будут проверяться, какие атрибуты файлов отслеживаются, а также исключения из проверки. Вот основные шаги по настройке конфигурации AIDE:

Чтобы изменить настройки AIDE, откройте конфигурационный файл в текстовом редакторе:

sudo nano /etc/aide.conf

Далее задавайте необходимые атрибуты для проверки в тексте файла.

Для проверки всех атрибутов файла можно использовать следующую запись:

/etc p+i+n+u+g+s+m+c+md5

Для мониторинга только изменения прав доступа и содержания файлов:

/var/log p+c+md5

Проверка только атрибутов и прав доступа

NORMAL = p+i+n+u+g+s+m+S+sha256

Проверка всех атрибутов, включая контрольные суммы:

EXTENDED = p+i+n+u+g+s+m+S+sha256+md5+rmd160

где параметры:

* p - права доступа;
* i - inode номер;
* n - количество ссылок;
* u - идентификатор пользователя (UID);
* g - идентификатор группы (GID);
* s - размер файла;
* m - время последней модификации;
* S - время последнего изменения статуса файла;
* sha256, md5 - контрольные суммы файлов.

В отдельном разделе конфигурации также указываются файлы и каталоги, которые будут отслеживаться на предмет изменений. Пример конфигурации:

Проверка корневой файловой системы с полными атрибутами, определёнными в строчке NORMAL:

/ NORMAL

Проверка каталога /etc с расширенными атрибутами, определенными в строчке EXTENDED:

/etc EXTENDED

Чтобы избежать проверки определённых системных файлов или временных данных, вы можете добавлять исключения с помощью символа "!" перед каталогом или файлом.

!/tmp

!/var/tmp

После внесения всех необходимых изменений, сохраните файл и закройте редактор.

После настройки конфигурации вы можете протестировать её работу, запустив создание новой базы данных AIDE:

sudo aideinit

После этого AIDE создаст базу данных, которую можно будет использовать для последующих проверок.

Пример содержимого конфигурационного файлаAIDE:

database=file:/var/lib/aide/aide.db

database\_out=file:/var/lib/aide/aide.db.new

NORMAL = p+i+n+u+g+s+m+S+sha256

/etc EXTENDED

/bin NORMAL

/sbin NORMAL

!/proc

!/sys

!/var/tmp

!/var/cache

Этот пример включает основные правила для мониторинга критически важных директорий Системы и исключает временные и кешированные файлы.

* 1. Проверка целостности

После создания базы данных и настройки конфигурации можно запустить проверку целостности Системы. Это делается командой:

aide --check

Команда сравнит текущее состояние файлов с данными, сохраненными в базе. Результаты проверки будут выведены в терминал, где можно увидеть, какие файлы были изменены, добавлены или удалены.

Для запуска проверки немедленно можно воспользоваться командой:

sudo systemctl restart aidecjeck.service

Результаты проверок также записываются в системный журнал, который можно открыть и просмотреть по команде:

sudo journalctl -u aidecheck.service

При необходимости фиксации корректных изменений конфигурации ОС нужно использовать команду:

sudo aideupdate

при этом будет запрошен пароль доступа к ключу GPG.

Чтобы посмотреть статус последней проверки, нужно выполнить команду:

systemctl status aidecheck.service

* + 1. Автоматизация проверки

Для удобства администрирования Системы проверку целостности ОС можно автоматизировать с помощью использования таймера systemd. Подробная настройка файлов с расширением .timer рассмотрена в разделе 3.4.2.1.

Для настройки таймера через systemd выполняются следующие шаги для настройки задания:

1. Создание Unit-файла для автоматизации проверки AIDE, где будет указано как именно systemd должен запускать проверку. Создайте файл /etc/systemd/system/aide-check.service со следующим содержанием:

[Unit]

Description=AIDE integrity check

Documentation=man:aide(1)

[Service]

Type=oneshot

ExecStart=/usr/sbin/aide --check

1. Далее создается таймер-файл, который будет запускать проверку по расписанию. Например, проверка может выполняться ежедневно. Создайте файл /etc/systemd/system/aide-check.timer со следующим содержанием:

[Unit]

Description=Daily AIDE integrity check

[Timer]

OnCalendar=daily

Persistent=true

[Install]

WantedBy=timers.target

Таймер можно настроить на любое другое расписание, например, использовать параметр OnCalendar=weekly для еженедельной проверки.

1. Далее необходимо активировать таймер-файл командой:

sudo systemctl enable aide-check.timer

sudo systemctl start aide-check.timer

Команда enable добавляет таймер в автозапуск, а start запускает таймер сразу после выполнения команды.

1. Для того, чтобы провести проверку статуса таймера и убедиться, что он работает корректно, используется команда:

sudo systemctl status aide-check.timer

1. Формирование замкнутой программной среды
   1. Общие принципы

Замкнутая программная среда (далее – ЗПС) – такая организация ОС, при которой возможен запуск только доверенных приложений, а сторонние скрипты или приложения выполняться не будут. В ОС РОСА "ХРОМ" ЗПС реализуется посредством технологии IMA – путем подписывания всех исполняемых файлов Системы с помощью механизма электронной подписи.

Электронная подпись основана на двух ключах – закрытом, с помощью которого возможно подписать файлы, и открытом, необходимом для проверки подписанного. Специальным образом сконфигурированная Система при попытке запуска любого исполняемого файла или при загрузке библиотеки будет проверять с помощью открытого ключа их подлинность и в случае успеха разрешать запуск.

**Следует обратить внимание**, что доступ к закрытому ключу должен быть только у уполномоченных лиц, отвечающих за безопасность Системы. Закрытый ключ не должен храниться на компьютере, где осуществлялось подписывание файлов с его помощью.

Для работы с ЗПС в ОС должны быть установлены пакеты ima-evm-utils, libressl, ima-inspect**.**

* 1. Создание ключей

Для создание открытых и закрытых ключей выполняют следующие действия:

1. Открыть консоль администратора командой (с вводом пароля):

sudo -i

1. Создать временную папку для ключей и перейти в неё:

mkdir ~/ima

cd ~/ima

1. Создать файл конфигурации x509.conf**,** например, командой:

nano x509.conf

1. Задать содержимое файла следующими настройками:

[ req ]

distinguished\_name = req\_distinguished\_name

prompt = no

string\_mask = utf8only

x509\_extensions = myexts

[ req\_distinguished\_name ]

O = IMA

CN = Executable Signing Key

emailAddress = ivan@petrov.tld

[ myexts ]

basicConstraints=critical,CA:FALSE

keyUsage=digitalSignature

subjectKeyIdentifier=hash

authorityKeyIdentifier=keyid

Значения полей O, CN, emailAddress являются пользовательскими, нужно заменить их на необходимые;

1. Сохранить файл (если используется редактор nano, то последовательно нажать клавиши Ctrl+O, Enter и Ctrl+X);
2. Создать пару ключей командой:

libressl req -new -nodes -utf8 -batch -newkey gost2001 -pkeyopt dgst:streebog512 -pkeyopt paramset:A -streebog512 -days 109500 -x509 -config x509.conf -outform DER -out ima\_cert.der -keyout ima\_priv.pem

где:

–ima\_cert.der– открытый ключ;

–ima\_priv.pem – закрытый ключ;

1. Создать папку для открытых ключей ядра, скопировать туда открытый ключ:

mkdir -p /etc/keys/ima

cp -v ima\_cert.der /etc/keys/ima/ima\_cert.der

и дополнительно скопировать открытый ключ для работы механизма проверки:

cp -v ima\_cert.der /etc/keys/x509\_evm.der

* 1. Создание политики IMA

Для создания политики IMA, продолжая работать в консоли от имени root, необходимо редактировать файл политики IMA:

sudo nano /etc/sysconfig/ima-policy

и скопировать в него следующее содержимое:

# PROC\_SUPER\_MAGIC

dont\_measure fsmagic=0x9fa0

dont\_appraise fsmagic=0x9fa0

# SYSFS\_MAGIC

dont\_measure fsmagic=0x62656572

dont\_appraise fsmagic=0x62656572

# DEBUGFS\_MAGIC

dont\_measure fsmagic=0x64626720

dont\_appraise fsmagic=0x64626720

# TMPFS\_MAGIC

dont\_measure fsmagic=0x01021994

dont\_appraise fsmagic=0x01021994

# RAMFS\_MAGIC

dont\_appraise fsmagic=0x858458f6

# DEVPTS\_SUPER\_MAGIC

dont\_measure fsmagic=0x1cd1

dont\_appraise fsmagic=0x1cd1

# BINFMTFS\_MAGIC

dont\_measure fsmagic=0x42494e4d

dont\_appraise fsmagic=0x42494e4d

# SECURITYFS\_MAGIC

dont\_measure fsmagic=0x73636673

dont\_appraise fsmagic=0x73636673

# SELINUX\_MAGIC

dont\_measure fsmagic=0xf97cff8c

dont\_appraise fsmagic=0xf97cff8c

# CGROUP\_SUPER\_MAGIC

dont\_measure fsmagic=0x27e0eb

dont\_appraise fsmagic=0x27e0eb

# CGROUP2\_SUPER\_MAGIC

dont\_measure fsmagic=0x63677270

dont\_appraise fsmagic=0x63677270

# NSFS\_MAGIC

dont\_measure fsmagic=0x6e736673

dont\_appraise fsmagic=0x6e736673

measure func=MMAP\_CHECK mask=MAY\_EXEC

measure func=BPRM\_CHECK mask=MAY\_EXEC

appraise func=BPRM\_CHECK mask=MAY\_EXEC appraise\_type=imasig

appraise func=MMAP\_CHECK mask=MAY\_EXEC appraise\_type=imasig

Затем необходимо сохранить файл и выставить на него права командой:

chmod 0400 /etc/sysconfig/ima-policy

* 1. Подписывание файлов

Для подписания всех исполняемых файлов в Системе, в том числе скриптов и разделяемых библиотек \*.so\*, необходимо:

1. Создать файл скрипта для их подписи:

nano sign.sh

1. Скопировать нижеследующий текст в этот файл:

#!/bin/bash  
set -xe

FS="$(findmnt --output FSTYPE / | tail -n1)"

echo "$FS"

echo > failed.log

find / -fstype "$FS" -type f -executable | sort -u | while read -r line

do

if ! evmctl ima\_sign --hashalgo streebog512 --key ima\_priv.pem "$line" ; then

echo "$line" >> failed.log

fi

done

for D in /lib /lib64 /usr/lib /usr/lib64

do

find "$D" -fstype $FS -\! -executable -type f -name "\*.so\*" | sort -u | while read -r line

do

if ! evmctl ima\_sign --hashalgo streebog512 --key ima\_priv.pem "$line"; then

echo "$line" >> failed.log

fi

done

done

1. Сохранить файл скрипта и сделать его исполняемым:

chmod 755 sign.sh

1. Запустить скрипт и подписать файлы:

./sign.sh

После запуска (скрипт будет работать некоторое время) можно посмотреть журнал ошибок командой (он должен быть пуст):

cat failed.log

* 1. Подготовка файловой системы

Если используется ФС ext4, то необходимо добавить опцию монтирования. Для этого нужно открыть файл настроек монтирования командой:

sudo nano /etc/fstab

и добавить в строках с ext4 опцию iversion через запятую после defaults, как показано ниже (рисунок 23), после этого сохранить файл.

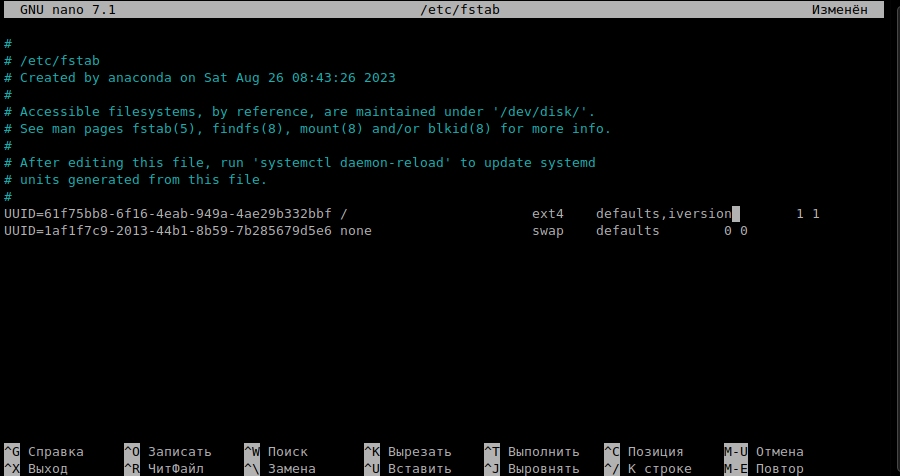


Рисунок 23 – Добавление опции монтирования для работы IMA

* 1. Добавление в образ начальной загрузки

Для добавления модуля integrity в initrd нужно выполнить команду:

sudo echo 'add\_dracutmodules+=" integrity "' | tee /etc/dracut.conf.d/10-ima.conf

и пересобрать образ начальной загрузки initrd командой:

sudo systemd-initramfs-gen

Следует обратить внимание, что в initrd должны попасть уже подписанные файлы, поэтому данный шаг выполняется после подписывания файлов в Системе.

* 1. Пробный запуск подписанной Системы

После подготовительных настроек ЗПС следует осуществить пробный запуск Системы с подписанными исполняемыми файлами. Для этого нужно добавить параметр загрузки ядра ima\_appraise=logв начальный загрузчик GRUB, аналогично описанному в п.3.8. В этом режиме Система не будет запрещать запуск неподписанных файлов, но будет записывать попытки их запуска.

После запуска Системы с вышеуказанным ключом можно попробовать проверить подпись системных файлов, например интерпретатора bash,командой:

sudo ima\_inspect /bin/bash

В выводе команды должна быть указана подпись и ее тип – streebog512, проверка подписи должна заканчиваться строчкой:

/bin/bash: verification is OK

* 1. Рабочий запуск и проверка работы

Для рабочего использования необходимо запустить Систему без параметра ima\_appraise=log в начальный загрузчик GRUB, то есть просто перезагрузить Систему. Система должна запуститься и работать штатно.

Далее необходимо проверить работу защиты от запуска неподписанных исполняемых файлов:

1. Войти в root-консоль с помощью команды:

sudo -i

1. Создать копию существующего исполняемого файла:

cp -v /bin/cat cat.copy

1. Предоставить права на исполнение для файла:

chmod +x cat.copy

1. Попробовать запустить файл командой:

./cat.copy

1. Получить в выводе ошибку:

-bash: ./cat.copy: Отказано в доступе

При этом в журнале аудита должна появиться запись о запрете запуска неподписанного файла.

Следует удалить из Системы закрытый ключ после проверки подписи файлов и штатной работоспособности ОС.

1. Управление идентификацией и аутентификацией с FreeIPA

Для централизованной аутентификации пользователей в локальной сети в ОС применяется технология IPA (FreeIPA). IPA – это интегрированное решение для управления идентификацией, аутентификацией и авторизацией ОС РОСА "ХРОМ" в сети. Этот сервер предоставляет централизованный контроль доступа и обеспечивает безопасность идентификации пользователей, компьютеров и служб в корпоративной среде.

* 1. Настройка сервера домена IPA

Для работы сервера IPA необходимо настроить сеть на сервере и имя хоста.

В качестве примера используются следующие данные для хоста IPA-сервера:

* имя хоста – dc1;
* имя домена – rosa.lan;
* IP-адрес IPA-сервера – 192.168.1.6;
* маска подсети, в которой будут использоваться IPA-сервер и рабочие станции – 255.255.255.0;
* шлюз по умолчанию – 192.168.1.1;
* DNS-сервер – 192.168.1.1;
* DNS-поиск домена – rosa.lan.

Алгоритм настройки сети:

1. установить имя сервера, например:

sudo hostnamectl dc1.rosa.lan

1. создать для сервера статический IP-адрес, используя интерфейс nmtui;
2. перезагрузить ОС, чтоб изменения вступили в силу.

После перезагрузки и входа в Систему необходимо установить пакет сервера IPA:

sudo dnf install ipa-server

и запустить его инсталляцию командой:

sudo ipa-server-install

В процессе установки сервера следует придерживаться следующих инструкций при запросе установщика:

1. о необходимости настройки DNS-сервера на IPA-сервере ответить YES;
2. о наименовании хоста, домена и realm ввести данные, которые были рассмотрены для примера в начале раздела, а именно: dc1.rosa.lan, rosa.lan и ROSA.LAN;
3. о логинах и паролях для сервера и административного аккаунта IPA ввести логин по умолчанию – admin, запрашиваемый пароль и запомнить их;
4. о DNS-сервере на вопросы:
   1. Do you want to configure DNS forwarders? – ответить YES, чтобы сконфигурировать DNS-серверы для форвардинга;
   2. Do you want to configure these servers as DNS forwarders? – ответить YES, чтобы сконфигурировать найденный DNS-серверы как сервер для форвардинга;
   3. Enter an IP address for a DNS forwarder , or press Enter to skip: – нажать клавишу Enter, чтобы не вписывать дополнительные DNS сервера для форвардинга., или вписать IP–адреса DNS-серверов для форвардинга;
   4. Do you want to search for missing reverse zone? – ответить YES, чтобы найти обратную зону DNS;
   5. Do you want to configure chrony with NTP server or pool address? – ответить NO, чтобы не конфигурировать сервер chrony, так как устраивает его конфигурация по умолчанию.
5. при выводе конфигурационной информации запросе о подтверждении правильности конфигурирования для дальнейшей установки IPA сервера ответить YES, если все правильно сконфигурировано.

После этого шага начнется инсталляция IPA-сервера на ПК.

Если установка прошла успешно, установщик IPA-сервера выведет конечную информацию об установленном сервере и его сервисах.

Далее необходимо проверить работоспособность служб IPA командой:

ipactl status

Если в выводах команды везде фигурирует RUNNING, значит все сервисы запущены и работают.

Следующим и последним шагом установки необходимо инициализировать аккаунт администратора IPA – admin:

kinit admin

и ввести пароль, который был введен ранее при установке IPA.

Установка может занять значительное время. Для инсталляции и работы сервера IPA рекомендуется не менее 4 Гб оперативной памяти.

Далее управлять установленным IPA-сервером можно через web-интерфейс; в данном случае по адресу dc1.rosa.lan.

В открывшемся веб-интерфейсе FreeIPA-сервера нужно ввести логин (admin) и пароль, чтобы войти под учетной записью администратора.

После успешного входа откроется веб-интерфейс IPA-сервера, в котором будет производиться вся дальнейшая работа с сервером.

* + 1. Создание пользователей

Чтобы добавить пользователей на сервере IPA, перейдите в веб-интерфейс сервера в раздел "Идентификация → Активные пользователи", заполнить все необходимые параметры и нажать кнопку Добавить.

Следует обратить внимание, что в поле "Имя учетной записи" логин будущего пользователя должен быть введен латинскими буквами.

Информация, вводимая в поля "Имя" и "Фамилия" будущего пользователя, носит только справочную информацию и не влияет в дальнейшую работу.

В выпадающем списке поля "ID группы" необходимо выбрать группу editors.

Для добавления каждого нового пользователя необходимо пройти аналогичные шаги. Все вновь созданные пользователи появятся в разделе "Активные пользователи".

* + 1. Настройка сети

Для настройки сети IPA-сервера перейдите в веб-интерфейс управления сервером и далее в меню "Параметры Системы → Соединения", выбрать соединение и вкладку IPv4 (в правой части экрана) и задать следующие параметры:

* из выпадающего списка в параметре "Метод" выбрать "Вручную";
* в поле "DNS серверы" – IP-адрес IPA сервера 192.168.1.1;
* в поле "Домены поиска" – rosa.lan;
* добавить статические адреса:
* 192.168.1.58 – это IP-адрес рабочей станции;
* 255.255.255.0 – маска подсети;
* 192.168.1.1 – шлюз.

По завершении нужно нажать кнопку Применить.

Далее необходимо переподключиться к сети (отключить и включить снова) и проверить доступность FreeIPA-сервера по FQDN-имени:

ping dc1.rosa.lan

* 1. Настройка клиента домена IPA

Перед установкой клиента IPA необходимо настроить сеть:

1. установить имя рабочей станции в домене, например:

sudo hostnamectl station.rosa.lan

1. задать для станции шлюз и DNS-сервер как IP-адрес сервера IPA, используя интерфейс nmtui, а также указать домен поиска в rosa.lan;
2. перезагрузить ОС, чтоб изменения вступили в силу.

Если все сделано правильно, команда:

ping dc1

должна отрабатывать корректно, показывая соединение с сервером IPA.

После перезагрузки и входа в Систему необходимо установить пакет клиента IPA:

sudo dnf install ipa-client

Далее нужно запустить инсталляцию и вход в домен командой:

sudo ipa-client-install --mkhomedir

На этом этапе понадобится ввести пароль администратора домена. После перезагрузки можно входить доменным пользователем.

Для установки FreeIPA-клиента сначала нужно установить пакет ipa-client:

sudo dnf install ipa-client

Настроенный в соответствии с примером из п.10.1 IPA-сервер имеет хост и домен: dc1 / rosa.lan, соответственно клиент должен иметь такой же домен:

sudo hostnamectl set-hostname st1.rosa.lan

где:

* st1 – имя рабочей станции;
* rosa.lan – имя домена.

Следующим шагом необходимо ввести рабочую станцию в домен IPA и инициализировать пользователя IPA:

ipa-client-install –mkhomedir

На запросы инсталлятора следует дать следующие ответы:

* о перенастройке сервера времени chrony – NO;
* о продолжении установки с выбранными параметрами – YES;
* логин и пароль администратора IPA – admin и пароль администратора.

Если все прошло успешно, об этом будет сообщено в последней строке вывода на экран.

После этого следует войти на IPA-сервер и проверить присутствие рабочей станции в домене в списке "Узлы".

Затем нужно инициализировать пользователя IPA, например useripa1:

kinit useripa1

и ввести пароль для пользователя FreeIPA сервера – useripa1.

Далее необходимо задать новый пароль для пользователя взамен временного и подтвердить его.

* 1. Вход в домен

Для того чтобы пользователю войти в домен IPA, необходимо на экране входа в ОС в момент выбора пользователя Системы выбрать параметр "Нет в списке", после чего откроется поле ввода логина нового пользователя.

Необходимо ввести логин с доменом, например useripa1@ROSA.LAN (имя домена необходимо вводить заглавными буквами), и пароль, заданный в п.10.2 на предыдущем шаге.

После входа в рабочую станцию под доменным пользователем IPA следует запросить его ID. ID пользователя должно совпадать с его ID на IPA-сервере.

На этом вход в домен IPA завершен.

1. Аудит
   1. Общие сведения о auditd

Аудит (audit) – это механизм аудита безопасности в ядре Linux-подобных ОС, который позволяет отслеживать события в Системе. Аудит может использоваться для мониторинга доступа к файлам, сетевых подключений, а также других системных событий.

В пространстве пользователя работает сервис auditd, который получает от ядра ОС информацию о произошедших событиях и обрабатывает ее в соответствии с настройками.

Ведение логов аудита обычно решает две задачи:

* протоколирование событий для разбора в случае действий злоумышленника;
* протоколирование событий для отладки работы программ.
  1. Управление службой аудита

Служба auditd по умолчанию включена, однако при необходимости ее можно включить в автозагрузку при старте вручную:

sudo systemctl enable auditd

Выключить службу можно с помощью команды:

sudo systemctl mask auditd

Остановить работу службы можно с помощью команды:

sudo systemctl kill auditd

* 1. Настройка конфигурации аудита

Конфигурационный файл находится по адресу /etc/audit/auditd.conf. Его содержимое по умолчанию подходит для работы в Системе.

Некоторые из наиболее важных конфигурационных параметров:

* max\_log\_file – максимальный размер журнального файла аудита в мегабайтах;
* num\_logs – максимальное количество журнальных файлов аудита;
* log\_file – путь к журнальному файлу аудита;
* log\_group – группа, которой принадлежит журнальный файл аудита;
* write\_logs – записывать (опция yes) или не записывать (опция no) события аудита в файл /var/log/audit/audit.log.

Примечание – Если установлена опция no, то события аудита будут попадать в systemd-journald, но не будут дублироваться в еще один файл. Утилита ausearch читает именно этот файл, поэтому ей нельзя будет пользоваться.

Больше информации можно получить по справке командой:

man auditd.conf

* 1. Настройка правил аудита

Правила аудита — это инструкции для системы аудита, определяющие, какие события и действия пользователей или процессов должны подвергаться аудиту, т.е. логироваться. Они помогают отслеживать действия, связанные с безопасностью, и регистрировать важные изменения в Системе. Настройка правил аудита в ОС осуществляется с помощью утилиты auditctl, которая взаимодействует с сервисом auditd.

Правила задают условия и параметры для мониторинга определенных файлов, каталогов или системных вызовов. Эти правила могут быть установлены в реальном времени или загружены при запуске Системы из конфигурационных файлов.

Auditctl позволяет задать правила аудита в режиме реального времени. Эти правила действуют только до момента перезагрузки Системы. Чтобы обеспечить постоянную работу правил аудита при каждой загрузке ОС, их нужно сохранять в конфигурационных файлах.

Правила аудита хранятся в следующих файлах:

* /etc/audit/audit.rules
* /etc/audit/rules.d/\*.rules

Файлы содержат инструкции, каждая строка которых представляет собой аргумент для команды auditctl. При запуске auditd из этих файлов загружаются все правила, обеспечивая автоматическое применение аудита без необходимости ручного ввода команд при каждом запуске Системы.

Пример строки в файле правил:

-w /etc/passwd -p wa -k passwd-file

Эта строка аналогична следующей команде, которая будет выполнена утилитой auditctl:

sudo auditctl -w /etc/passwd -p wa -k passwd-file

В данном примере параметры означают следующее:

* -w /etc/passwd — указывает путь к файлу, который нужно отслеживать;
* -p wa — устанавливает права мониторинга: w (изменение) и a (добавление);
* -k passwd-file — добавляет метку (ключ) для поиска соответствующих событий в логах.

Рассмотрим основные типы параметров в правилах аудита:

* -w (watch) — указывает на файл или каталог, который нужно отслеживать;
* -p (permissions) — задает права доступа, за изменениями которых нужно следить. Доступные значения:
* r — чтение файла;
* w — запись в файл;
* x — выполнение файла;
* a — добавление информации в файл.
* -k (key) — метка для фильтрации и поиска событий по ключу;
* -S (syscalls) — отслеживание системных вызовов (например, open, execve);
* -F (fields) — добавление дополнительных условий, таких как идентификатор пользователя (UID) или группы (GID);
* -a (action) — определяет действие: добавление или удаление правила.

Для добавления правила аудита через командную строку можно использовать auditctl. Например, чтобы следить за изменениями в файле /etc/shadow, используется следующая команда:

sudo auditctl -w /etc/shadow -p wa -k shadow-file

Это правило будет активно до перезагрузки Системы.

Для удаления правил аудита используется параметр -d. Пример:

sudo auditctl -d -w /etc/shadow

* + 1. Использование ключа поиска событий

Метка, установленная через параметр -k, позволяет легко находить события, связанные с этим ключом, в логах. Чтобы просмотреть события, связанные с конкретным правилом, можно использовать команду:

ausearch -k shadow-file

Для получения подробной справки по утилите ausearch, включая ключи поиска и другие опции, используется следующая команда:

man ausearch

* 1. Выборка из логов аудита

Утилита ausearch – это инструмент для анализа журналов аудита, созданных с помощью системы аудита ОС. Журналы аудита содержат записи о действиях, совершаемых в Системе, таких как изменение файлов, запуск процессов или использование сетевых ресурсов. Утилита ausearch предназначена для поиска и анализа этих записей.

Утилита может выполнять поиск записей журнала аудита по различным критериям, таким как время, пользователь, тип события и др.

В качестве примера можно использовать следующую команду для поиска всех записей журнала аудита, связанных с изменением файла /etc/passwd:

sudo ausearch -f /etc/passwd

Результаты поиска будут выведены на экран консоли. Каждая запись журнала аудита будет содержать информацию о времени события, типе события, пользователе, который совершил действие, и другие подробности.

Одной из наиболее полезных функций утилиты ausearch является возможность использования ключей для выполнения расширенного поиска. Например, следующая команда выполняет поиск записей журнала аудита, связанных с запуском процесса с именем "httpd":

sudo ausearch -c httpd

Другим полезным ключом является ключ "-i", который позволяет просмотреть подробную информацию о конкретной записи журнала аудита. Например, следующая команда выводит подробную информацию о записи журнала аудита:

sudo ausearch -i 12345

где 12345 – это идентификатор записи журнала аудита.

Утилита ausearch также может использоваться для выполнения аудита безопасности. Например, можно настроить систему аудита таким образом, чтобы она записывала информацию обо всех неудачных попытках входа в Систему. Кроме того, можно использовать утилиту ausearch для поиска этих записей журнала аудита и выявления потенциальных угроз безопасности.

Также утилита ausearch поддерживает несколько форматов вывода, включая форматы текста и JSON. Это может быть полезно для автоматизации процесса анализа журналов аудита.

* 1. Примеры работы со службой аудита
     1. Отслеживание изменений в директории или файле

Для отслеживания всех изменений в директории /var/log нужно добавить следующее правило:

sudo auditctl -w /var/log -p w -k var\_log\_changes

Это правило отслеживает любые операции записи (опция w) в директорию /var/log и связывает их с ключевым словом (tag) var\_log\_changes.

Следует проверить, что правила аудита корректно добавлены командой, которая выводит список текущих правил аудита:

sudo auditctl -l

Для примера можно записать текст в файл внутри тестовой директории:

echo text | sudo tee /var/log/testfile

Можно просмотреть журнальные файлы аудита, используя утилиту ausearch. Чтобы просмотреть все записи аудита, связанные с директорией /var/log, следует использовать следующую команду:

sudo ausearch -k var\_log\_changes

Эта команда выводит все записи аудита, связанные с ключевым словом var\_log\_changes, в т.ч. событие записи в файл /var/log/testfile**.**

* + 1. Отслеживание запуска определенного приложения

В данном случае используется правило аудита следующего вида:

sudo auditctl -a exit,always -F path=/usr/bin/myapp -F perm=x -k myapp\_execution

Это правило отслеживает каждый выход (exit) из приложения /usr/bin/myapp с правами на выполнение (x) и связывает его с ключевым словом myapp\_execution.

Поиск событий осуществляется по аналогии с предыдущим пунктом:

sudo ausearch -k myapp\_execution

* + 1. Отслеживание системных вызовов

Правило аудита для отслеживания 64-битных системных вызовов execve() от имени root, т.е. отслеживания запуска программ от имени root:

sudo auditctl -a exit,always -F arch=b64 -S execve -F uid=0 -k authentication\_events

Это правило отслеживает каждый вызов функции execve (используется для запуска исполняемых файлов) с идентификатором пользователя (uid) равным 0 (root) и связывает его с ключевым словом authentication\_events. Т.к. 64-битные ОС поддерживают запуск и 32-битных программ, можно подвергнуть аудиту то же самое в 32-битном варианте:

sudo auditctl -a exit,always -F arch=b32 -S execve -F uid=0 -k authentication\_events

* + 1. Отслеживание доступа к конфиденциальным файлам

Для отслеживания доступа к конфиденциальным файлам используется правило аудита следующего вида:

sudo auditctl -a exit,always -F arch=b64 -S open -F dir=/etc/sensitive -F success=0 -k sensitive\_files\_access

Это правило отслеживает каждый вызов функции open для любого файла в директории /etc/sensitive, который заканчивается неудачей (success=0) и связывает его с ключевым словом sensitive\_files\_access.

* + 1. Отслеживание сетевых подключений

Для того, чтобы настроить правила аудита на отслеживание всех сетевых подключений в Системе используется правило следующего вида:

sudo auditctl -a exit,always -F arch=b64 -S bind -S connect -F success=0 -k network\_events

Это правило отслеживает каждый вызов функций bind() и connect() для любого сетевого подключения, которое заканчивается неудачей (success=0) и связывает его с ключевым словом network\_events.

* + 1. Запись в журнал аудита при подключении устройства USB

Для того, чтобы ОС фиксировала такие события как подключение USB-устройств используйте следующее правило аудита следующего вида:

sudo auditctl -w /dev/bus/usb -p rwxa -k usb

Теперь Система будет записывать информацию при подключении устройства USB в журнал аудита.

Чтобы найти связанные с подключением устройства USB записи журнала аудита, нужно выполнить следующую команду ausearch:

sudo ausearch -f /dev/bus/usb -k usb -i

Эта команда найдет все записи журнала аудита, связанные с ключом usb и событием OPEN, которое относится к подключению устройства USB.

* + 1. Централизованный сбор событий аудита

Централизованный сбор данных аудита в ОС необходим для долгосрочного хранения и анализа данных на внешних системах. Такой подход позволяет избежать потерь данных при повреждении локального диска и предоставляет возможность анализа журналов в централизованной точке. В данном контексте важны две основные задачи: архивирование событий и безопасная передача данных.

Для сохранения локальных журналов перед их отправкой на внешнее хранилище может использоваться команда tar, которая позволяет собрать все необходимые файлы с журналами событий в один архив, что упрощает передачу и сжатие данных. Например, чтобы архивировать все журналы аудита, можно использовать команду:

tar -czvf audit\_logs.tar.gz /var/log/audit

где параметры:

* -c — создает архив;
* -z — сжимает файлы с помощью gzip;
* -v — вывести процесс на экран;
* -f — указать имя файла.

Это архивирование важно для сохранения целостности данных до их передачи на внешние серверы.

Для отправки архивированных журналов событий на внешние хранилища используется служба systemd-journal-gatewayd. Эта служба предоставляет веб-интерфейс для просмотра и передачи логов по HTTP(S). Она может быть настроена для предоставления доступа как к локальным журналам, так и для передачи данных на внешние серверы.

Настройка этой службы выполняется через два файла конфигурации:

* /etc/systemd/journal-remote.conf — файл конфигурации, отвечающий за прием данных из внешних источников;
* /etc/systemd/journal-upload.conf — файл, используемый для настройки отправки данных на удаленные сервера.

Пример настройки конфигурации: в файле journal-upload.conf можно указать параметры для передачи данных на удаленный сервер:

[Upload]

URL=https://external-server:19532

Для обеспечения защищенной передачи данных аудита может использоваться шифрование с помощью SSL/TLS. Средство openssl позволяет создавать ключи для защиты передаваемых данных. Перед передачей, на сервере генерируются ключи и сертификаты:

1. Создание приватного ключа:

openssl genrsa -out private.key 2048

Генерация сертификата:

openssl req -new -x509 -key private.key -out server.crt -days 365

Эти ключи используются в конфигурации journal-upload.conf для обеспечения шифрования.

Эти меры безопасности позволяют гарантировать, что данные аудита будут переданы в защищенном виде и не будут перехвачены злоумышленниками.

1. Настройка и управление сетевыми сервисами
   1. Сервер DHCP

Сервер DHCP (Dynamic Host Configuration Protocol) используется для автоматического назначения IP-адресов и других сетевых параметров (например, шлюз по умолчанию, DNS-серверы) устройствам в сети. Это позволяет упростить процесс подключения устройств к сети, так как нет необходимости вручную настраивать параметры сетевого интерфейса для каждого устройства.

Когда устройство (клиент) подключается к сети, оно отправляет запрос на сервер DHCP. Сервер DHCP, в свою очередь, выделяет свободный IP-адрес из заранее настроенного диапазона и отправляет его клиенту, а также другие параметры конфигурации сети.

* + 1. Установка

Установка DHCP выполняется командой:

sudo dnf -y install dhcp

Следует обратить внимание, что обычно к Системе подключено более одного сетевого интерфейса. Если нужно, чтобы DHCP-сервер был запущен лишь на одном из интерфейсов, то необходимо выполнить следующие действия:

1. Открыть файл /etc/sysconfig/dhcpd;
2. Добавить имя конкретного интерфейса в список DHCPDARGS, например, если интерфейс eth0, то добавить:

DHCPDARGS=eth0

1. Сохранить этот файл и выйти.
   * 1. Настройка

Первым шагом установки DHCP-сервера является создание файла конфигурации dhcpd.conf. Основной файл конфигурации DHCP обычно располагается по пути /etc/dhcp/dhcpd.conf (изначально пуст) и хранит всю сетевую информацию, отправляемую клиентам.

Тем не менее существует другой пример файла конфигурации /usr/share/doc/dhcp\*/dhcpd.conf.sample, который является хорошей отправной точкой для настройки DHCP-сервера.

В файле конфигурации DHCP существует два типа операторов:

* parameters (параметры) – определяет, как выполнять задачу, выполнять ли задачу вообще и какие параметры конфигурации сети отправлять DHCP-клиенту;
* declarations (объявление) – определяет топологию сети, клиентов, предлагает адреса для клиентов и применяет группу параметров к группе объявлений.

Для настройки сервера DHCP выполняют следующие действия:

1. Скопировать пример файла конфигурации в качестве основного:

cp /usr/share/doc/dhcp-4.2.5/dhcpd.conf.example /etc/dhcp/dhcpd.conf

1. Открыть для редактирования основной файл конфигурации:

sudo nano /etc/dhcp/dhcpd.conf

1. Задать глобальные параметры DHCP-сервера, которые будут применяться ко всем подсетям:

option domain-name "rosa.lan";

option domain-name-servers ns1.rosa.lan, ns2.rosa.lan;

default-lease-time 3600;

max-lease-time 7200;

authoritative;

1. Определить подсеть; в этом примере сделана настройка DHCP для 192.168.56.0/24 LAN-сети:

subnet 192.168.56.0 netmask 255.255.255.0 {

option routers 192.168.56.1;

option subnet-mask 255.255.255.0;

option domain-search "rosa.lan";

option domain-name-servers 192.168.56.1;

range 192.168.56.10 192.168.56.100;

}

Все строки параметров в файле конфигурации заканчиваются точкой с запятой (;). Некоторые параметры могут иметь не одно значение, например domain-name-servers, у которого два IP-адреса, разделенные запятой. Строки, начинающиеся с "#", являются комментариями и не обрабатываются сервером DHCP.

Некоторые общие параметры сервера DHCP:

* subnet – объявляет подсеть (в вышеприведенном случае 192.168.38.0 с маской 255.255.255.0);
* range– диапазон выдаваемых адресов (от 192.168.56.10 до 192.168.56.100);
* option subnet-mask – маска сети (255.255.255.0);
* option broadcast-address–широковещательный адрес. (192.168.38.255);
* domain-name-servers – адреса серверов DNS (8.8.8.8, 8.8.4.4);
* option domain-name – доменное имя (workgroup);
* option routers – IP-адрес шлюза или точки выхода в сеть (192.168.56.1)

1. После внесения изменений в основной файл конфигурации открыть файл /etc/default/isc-dhcp-server и заменить параметр INTERFACESv4 на имя сетевого интерфейса, который смотрит внутрь сети. Чтобы узнать его имя, следует воспользоваться командами ipconfig или ip**.**
   * 1. Присвоение адреса

Для того чтобы присвоить статический IP-адрес конкретному клиентскому компьютеру в сети, нужно определить раздел в файле /etc/dhcp/dhcpd.conf, в котором необходимо указать MAC-адреса и фиксированный IP-адрес для назначения, например:

host rosa {

hardware ethernet 00:f0:m4:6y:89:0g;

fixed-address 192.168.56.105;

}

host rv-node {

hardware ethernet 00:4g:8h:13:8h:3a;

fixed-address 192.168.56.110;

}

Примечание – Узнать MAC-адрес в ОС РОСА "ХРОМ" можно, используя следующую команду:

ifconfig -a eth0 | grep HWaddr

После этого следует запустить службу DHCP и включить параметр для её автоматического запуска со следующей загрузки Системы, используя команды:

sudo systemctl start dhcpd

sudo systemctl enable dhcpd

Затем необходимо разрешить службу DHCP (DHCPD-служба прослушивает порт 67/UDP) командами:

nft add rule inet filter input udp dport 67 accept

или

nft add rule inet filter input tcp dport 67 ct state new accept

nft list ruleset > /etc/nftables.conf

И затем можно загрузить эти правила при старте:

nft -f /etc/nftables.conf

* + 1. Настройка клиента

Далее необходимо настроить клиентов в сети на автоматическое получение IP-адресов от DHCP-сервера. Для этого нужно войти на клиентский компьютер и перейти к редактированию файла конфигурации интерфейса Ethernet командой (имя/номер интерфейса приведены в качестве примера):

sudo nano /etc/sysconfig/network-scripts/ifcfg-eth0

В файл необходимо добавить следующие строки:

DEVICE=eth0

BOOTPROTO=dhcp

TYPE=Ethernet

ONBOOT=yes

Затем следует сохранить файл и выйти из редактора.

* 1. Сервер DNS

DNS является основным компонентом сетевой инфраструктуры, обеспечивающим преобразование доменных имен в IP-адреса и обратно. DNS является критическим элементом сетевой инфраструктуры, и его правильная настройка и управление играют важную роль в обеспечении работоспособности сетей.

В качестве сервера, реализующего DNS, в ОС используется сервер Bind.

* + 1. Установка

Перед установкой сервера необходимо обновить Систему:

dnf --refresh distro-sync

Далее перейдем к установке сервера Bind, исполняемый файл которого называется named. Следует обратить внимание, что все конфигурационные файлы/сервисы будут называться с префиксом named\*.

Далее для настройки DNS-сервера необходимо настроить параметр systemdresolvedв файле /etc/systmd/resolved.conf.

Перед настройкой также следует запросить команду:

lsof -i :53

Исправить нужно следующие строки:

DNS=127.0.0.1

FallbackDNS=

DNSSEC=no

LLMNR=resolve

DNSStubListener=no

где:

* DNS=127.0.0.1 – задает локальный IP-адрес, на котором будет работать будущий DNS-сервер;
* FallbackDNS= – данный параметр оставить пустым, чтобы systemd-resolved не переключался на fallback DNS-сервера;
* DNSSEC=no – отключает DNSSEC;
* LLMNR=resolve – перевести LLMNR в режим resolve;
* DNSStubListener=no – задать параметр no, чтобы systemd-resolved не прослушивал порт 53.

Остальные опции в этом файле следует оставить как есть и произвести перезапуск systemd-resolved:

sudo systemctl restart systemd-resolved

После запуска вновь требуется запросить команду (вывод команды должен быть пустой):

lsof -i :53

Если вывод действительно пустой, значит всё сделано корректно и можно приступать к настройке и запуску bind.

* + 1. Настройка

Основной конфигурационный файл сервиса bind – named.conf.

Для настройки нужно выполнить следующие действия:

1. Открыть конфигурационный файл named.conf и для минимальной работы DNS-сервера исправить следующие строки:

options {

listen-on port 53 { any; }; listen-on-v6 port 53 { any; }; . . . . .

allow-query { any; };

где:

* listen-on port 53 { any; }; – задать параметр any для прослушивания на всех IPv4 хоста;
* listen-on-v6 port 53 { any; }; – задать параметр any для прослушивания на всех IPv6 хоста;
* allow-query { any; }; – разрешить любые запросы к серверу.

1. Запустить bind:

sudo systemctl start named.service

1. Проверить, какие службы прослушивают порт 53:

lsof -i :53

Если в выводе прописан параметр named, то все настроено правильно.

1. Проверить работу DNS-сервера с помощью команды, например с адресом yandex.ru:

dig @127.0.0.1 yandex.ru

или

nslookup yandex.ru

Должны быть выведены IP-адреса yandex.ru.

1. Включить DNS-сервис в автозагрузку:

sudo systemctl enable named.service

* + 1. Переадресация

Чтобы настроить bind для переадресации (forward) запросов к другим DNS-серверам нужно выполнить следующие шаги:

1. Открыть конфигурационный файл named.conf и для минимальной работы DNS-сервера исправить следующие строки:

options {

. . . . .

recursion yes; . . . . .

forward only; forwarders {

77.88.8.8;

77.88.8.1;

};

где:

* forward – режим перенаправления:
* forward only – все запросы на настраиваемый DNS-сервер будут перенаправляться на другие DNS-сервера, прописанные в следующей опции "forwarders {}";
* forward first – все запросы на настраиваемый DNS-сервер будут перенаправляться на другие DNS-сервера, прописанные в следующей опции "forwarders {}", и если с помощью них не удастся разрешить запрос, то запрос будет пытаться разрешаться настраиваемым DNS-сервером локально;
* forwarders { 77.88.8.8; 77.88.8.1; } – список DNS-серверов, для перенаправления запросов.

1. Запустить DNS-сервер:

sudo systemctl start named.service

В результате DNS-сервер будет настроен для минимальной работы.

* 1. Сервер Samba

Samba представляет собой программный пакет с открытым исходным кодом, дающий сетевым администраторам возможность гибко и свободно настраивать, конфигурировать и выбирать системы и оборудование, то есть устанавливать на компьютерах с UNIX/Linux-подобными ОС имитации устройств с Windows. Это упрощает выполнение задач по обмену файлами (как файл-серверам) или задавать параметры печати в качестве принт-серверов.

Samba обеспечивает свободный доступ к:

* дискам Linux – к Windows-компьютерам;
* дискам Windows – к оборудованию с ПО Linux;
* принтерам Linux – к Windows-компьютерам;
* принтерам Windows – к Linux-системам.
  + 1. Настройка сети

Для обеспечения работы Samba используются протоколы SMBи CIFS.

В отличие от SMB**,** CIFS имеет более широкий функционал – этот протокол способен поддерживать символические и жесткие ссылки, а также файлы больших размеров.

Но в настоящее время CIFSпочти не используется, так как в последних версиях Windows (начиная с версии 8) используются протоколы SMB2 или SMB3, которые по сути являются обновленной версией CIFS.

В качестве примера ниже описывается создание лабораторной инфраструктуры, состоящей из одного сервера и одного клиента, находящихся в одной подсети 192.168.122.0/24. Все приведенные консольные команды выполняются от пользователя root, если не указано иное.

Важно правильно настроить сеть на клиенте и сервере. В простых случаях, в том числе при поднятии тестовых стендов, необходимо, чтобы клиент и сервер находились в одной подсети. Например, чтобы у ВМ с сервером был IP-адрес 192.168.10.2, а у ВМ с клиентом – 192.168.10.3.

Для настройки сети на сервере необходимо выполнить следующие шаги:

1. Установить полное доменное имя (FQDN) в формате <имя\_хоста>.<домен>.<зона>, например samba1.rosa.lan.

В этом случае именем хоста будет samba1, доменом – rosa, доменной зоной – lan, realm – rosa.lan.

Следует избегать использования зоны .local, или выключить ее, или перенастроить Avahi, чтобы использовать ее.

Для установки имени хоста на контроллере домена используется команда:

sudo hostnamectl set-hostname samba1.rosa.lan

Посмотреть IP-адрес можно с помощью команды:

/sbin/ip a

Пример вывода:

[root@server1 ~]# ip a

1: lo: <LOOPBACK,UP,LOWER\_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000

link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00 inet 127.0.0.1/8 scope host lo

valid\_lft forever preferred\_lft forever inet6 ::1/128 scope host

valid\_lft forever preferred\_lft forever

2: host0@if6: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000

link/ether 32:e2:0b:ba:17:c5 brd ff:ff:ff:ff:ff:ff link-

netnsid 0

inet 169.254.212.10/16 brd 169.254.255.255 scope link host0\ valid\_lft forever preferred\_lft forever

inet 192.168.122.12/24 brd 192.168.122.255 scope global\ dynamic

host0

valid\_lft 2577sec preferred\_lft 2577sec inet6\ fe80::30e2:bff:feba:17c5/64 scope link valid\_lft forever\ preferred\_lft forever

В данном примере IPv4-адрес – 192.168.122.12.

1. Добавить в файл /etc/hosts строку такого вида, чтобы и FQDN, и краткое имя выводились в IP-адрес контроллера домена:

192.168.122.12 samba1.rosa.lan samba1

1. Проверить корректность записей в файле /etc/hosts командой:

ping -c3 samba1.rosa.lan ping -c3 samba1

**Следует обратить внимание**, что сеть должна запускаться до входа пользователя в Систему. Если используется подключение к сети по WiFi, то необходимо настроить его соответствующим образом.

* + 1. Запуск контроллера домена

Контроллер домена – сервер, управляющий доступом к сетевым ресурсам в рамках одного домена (группы хостов или сетей, которые объединены общими политиками безопасности).

Для запуска контроллера домена нужно выполнить следующие шаги:

1. Установить необходимые для работы пакеты:

sudo dnf install samba-server /bin/ps /usr/bin/xargs\ /usr/bin/nslookup

1. Отключить лишние службы из состава набора программ Samba, так как их обособленная работа не требуется и будет мешать контроллеру домена:

sudo systemctl disable --now smb nmb winbind

1. Убедиться, что они действительно выключены, используя команду, вывод которой должен быть пустым:

ps ax | grep -E "samba|smbd|nmbd|winbindd" | grep -v grep

1. Удалить старые файлы настроек, оставив их резервные копии:

mv -v /etc/samba/smb.conf /etc/samba/smb.conf.old mv -v\ /etc/krb5.conf /etc/krb5.conf.old

1. Определить старые БД, путь к которым можно узнать командой:

smbd -b | grep -E "LOCKDIR|STATEDIR|CACHEDIR|PRIVATE\_DIR"

1. Использовать для очистки старых БД команду:

smbd -b | grep -E "LOCKDIR|STATEDIR|CACHEDIR|PRIVATE\_DIR" |\ awk

'{print $NF}' | xargs -I'{}' rm -fv '{}'/\*.{tdb,ldb}

Если эта команда ничего не выдала, то БД от прошлых запусков контроллера домена не было.

1. Отключить службу systemd-resolved на порту 53, которая может препятствовать работе DNS-сервера, встроенного в Samba или основанного на BIND. Чтобы устранить конфликт, необходимо отключить механизм DNS Stub Listener командой:

mkdir -p /etc/systemd/resolved.conf.d

echo -e '[Resolve]\nDNSStubListener=no' >\ /etc/systemd/resolved.conf.d/no53port.conf

sudo systemctl try-restart systemd-resolved

После перезапуска systemd-resolved порт 53 освободится и станет доступен для использования контроллером домена.

1. Запустить интерактивную настройку домена:

samba-tool domain provision --use-rfc2307 --interactive

Ниже приведен ее типовой вывод. В квадратных скобках ([ ]) указывается значение по умолчанию. Следует нажать клавишу Enter, чтобы согласиться с предложенным, или ввести иное значение. Если сеть и имя хоста были настроены верно, то нет необходимости изменять значения по умолчанию:

[root@server1 ~]# samba-tool domain provision --use-rfc2307\ --interactive

Realm [ROSA.LAN]:

Domain [ROSA]:

Server Role (dc, member, standalone) [dc]:

DNS backend (SAMBA\_INTERNAL, BIND9\_DLZ, BIND9\_FLATFILE, NONE) [SAMBA\_INTERNAL]:

DNS forwarder IP address (write 'none' to disable forwarding) [192.168.122.1]:

Administrator password:

Retype password:

На последнем шаге необходимо ввести пароль доменного пользователя Administrator; при вводе пароль не отображается; пароль должен быть сложным.

Примечание:

* При выборе типа "DNS backend" рекомендуется использовать BIND9\_DLZ, если планируется интеграция с внешним DNS-сервером BIND.
* Использование BIND9\_FLATFILE не поддерживается и будет исключено из будущих версий Samba.
  + 1. Перенаправление

DNS Forwarder – это адрес сервера, к которому перенаправляются DNS-запросы, на которые сам контроллер домена не может ответить, например, запрос DNS yandex.ru будет направлен в него.

Для настройки перенаправления нужно провести следующие действия:

1. Выполнить команду:

mv -v /var/lib/samba/private/krb5.conf /etc/krb5.conf

1. Добавить службу контроллера домена в автозапуск:

sudo systemctl enable samba

1. Запустить контроллер домена:

sudo systemctl restart samba

1. Посмотреть лог контроллера домена и убедиться, что он запустился без ошибок:

systemctl status samba

Статус службы должен быть active (запущена), но в конце лога может быть ошибка, например:

dnsupdate\_nameupdate\_done: Failed DNS update with exit code 29

1. Убедиться, что сведения об ошибке не являются критическими, запустив команду:

sudo samba\_dnsupdate --verbose

В подробном логе будет запись про ошибки "WERR\_DNS\_ERROR\_RECORD\_ALREADY\_EXISTS", которые означают, что добавляемая запись DNS уже существует. Она была создана при создании домена. Запись "Failed update of 29 entries" означает, что такая ситуация возникла 29 раз, что и стало кодом возврата samba\_dnsupdate.

1. Проверить работу DNS-сервера на другом компьютере, например, на будущем клиенте домена, выполнив команду:

nslookup samba1.rosa.lan 192.168.122.12

Для корректного выполнения необходимо подставить в пример реальные IP- адреса.

Пример успешного ответа:

$ nslookup samba1.rosa.lan 192.168.122.12

Server: 192.168.122.12

Address: 192.168.122.12#53

Name: samba1.rosa.lan

Address: 192.168.122.12

Для настройки клиентов будет полезно знать рабочую группу NT с помощью команды:

[root@server1 ~]# cat /etc/samba/smb.conf | grep workgroup =\ SAMBA

Следует убедиться, что этот сервер способен выдать DNS требуемого домена, выполнив на будущем клиенте домена (не на контроллере) команду, например:

nslookup yandex.ru 192.168.122.1

После первоначальной настройки контроллера его можно заменить в файле /etc/samba/smb.conf, перезапустив контроллер после изменения настроек командой:

sudo systemctl restart samba

После успешной настройки контроллера домена рекомендуется создать хотя бы одного пользователя домена. Это позволит проверить корректность функционирования службы аутентификации и в дальнейшем использовать учётные записи для входа клиентов в домен.

Для создания нового пользователя следует выполнить следующую команду:

samba-tool user create <имя\_пользователя>

Пример:

samba-tool user create IvanovVP

После ввода команды будет запрошен пароль для новой учётной записи. Пароль должен соответствовать установленной политике безопасности и быть достаточно сложным.

* + 1. Использование внешнего DNS-сервера BIND

Контроллер домена Samba может использовать встроенный механизм обработки DNS-запросов (SAMBA\_INTERNAL) или быть настроен для работы с внешним DNS-сервером BIND. В ряде случаев рекомендуется использовать BIND с поддержкой DLZ-модуля.

Для начала необходимо установить пакет bind:

sudo dnf install bind

По умолчанию служба BIND настроена на прослушивание только интерфейса loopback (127.0.0.1). Чтобы разрешить обработку запросов на всех сетевых интерфейсах, следует изменить конфигурацию:

sudo sed -i '/listen-on\|allow-query/ s|{.\+}|{ any; }|' /etc/named.conf

Затем необходимо включить использование DLZ-модуля:

echo 'include "/var/lib/samba/bind-dns/named.conf";' >> /etc/named.conf

Служба systemd-resolved может занимать порт 53, используемый DNS-сервером BIND. Чтобы устранить конфликт, необходимо отключить DNS Stub Listener:

mkdir -p /etc/systemd/resolved.conf.d

echo -e '[Resolve]\nDNSStubListener=no' > /etc/systemd/resolved.conf.d/no53port.conf

sudo systemctl try-restart systemd-resolved

* + - 1. Запуск службы BIND

Для того чтобы добавить BIND в автозагрузку и запустить службу, используется команда:

sudo systemctl enable --now named

Для проверки статуса службы:

systemctl status named

В ходе выполнения команды samba-tool domain provision необходимо выбрать тип DNS-сервера, при этом:

* при использовании BIND следует указать BIND9\_DLZ в качестве типа DNS backend.
* не рекомендуется использовать BIND9\_FLATFILE, так как он не поддерживается и будет удалён в будущих версиях Samba.
  + 1. Подключение клиента ROSA к домену

Для полноценной работы доменной инфраструктуры необходимо подключить клиентскую машину с ОС РОСА "ХРОМ" к созданному домену. Это позволяет пользователям аутентифицироваться через контроллер домена, а также использовать единые политики безопасности.

Для настройки сети на клиенте следует:

1. убедиться, что клиент находится в одной подсети с контроллером домена и имеет доступ к нему по сети;
2. добавить IP-адрес контроллера домена в список DNS-серверов;
3. добавить имя домена (realm) в список доменов поиска одним из следующих способов:

* через графический интерфейс утилиты NetworkManager необходимо перейти в "Параметры системы ⟶ блок "Сеть и связь" ⟶ Соединения":
  1. открыть настройки сетевого подключения;
  2. перейти во вкладку "IPv4" (рисунок 24);
  3. в поле "DNS-серверы" указать IP-адрес контроллера домена (например, 192.168.122.12);
  4. в поле "Домены поиска" указать имя домена (например, samba.loc);
  5. сохранить изменения;

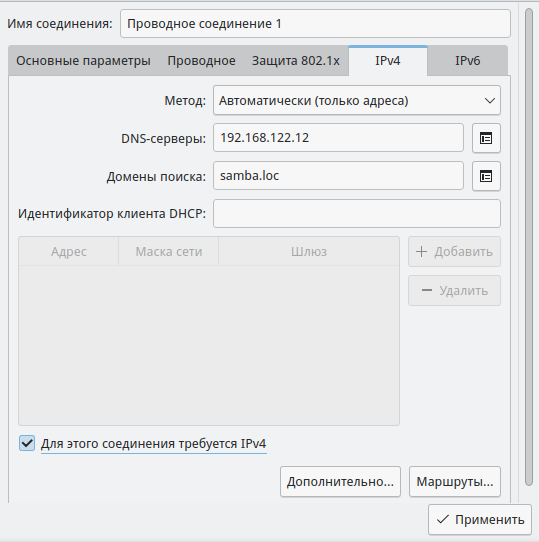


Рисунок 24 – Настройка сети в графическом интерфейсе утилиты NetworkManager

* через утилиту nmtui или вручную:
* в ROSA 2019.05 и новее можно использовать утилиту nmtui;
* при отсутствии NetworkManager можно вручную отредактировать файл /etc/resolv.conf, добавив строки:

nameserver 192.168.122.12

search samba.loc

* при использовании resolv.conf вручную убедиться, что он не перезаписывается DHCP-клиентом;

проверить сетевую доступность контроллера:

ping server1.samba.loc

на клиентской машине установить следующие пакеты:

sudo dnf install drakxtools samba-server samba-client samba-winbind nss\_ldap libnss-role pam\_krb5 lib64sasl2-plug-gssapi urpmi perl-URPM

Для подключения к домену с помощью drakauth необходимо:

1. запустить утилиту drakauth командой:

drakauth

Если переменная окружения $DISPLAY установлена, откроется графический интерфейс, в противном случае будет доступен псевдографический режим;

выбрать способ аутентификации – "Домен Windows" (рисунок 25);

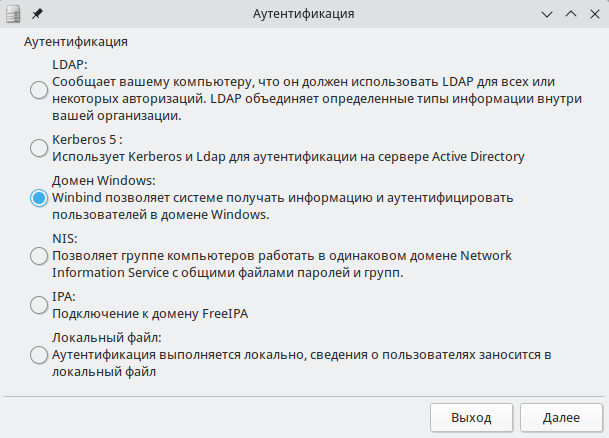


Рисунок 25 – Выбор способа аутентификации в графическом интерфейсе drakauth

указать тип домена — "Active Directory" (рисунок 26);

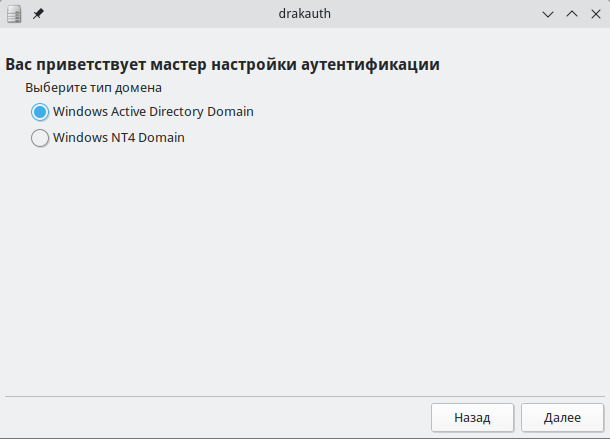


Рисунок 26 – Выбор типа домена в графическом интерфейсе drakauth

убедиться, что поля домена и сервера заполнены автоматически (рисунок 27). При необходимости внести корректировки;

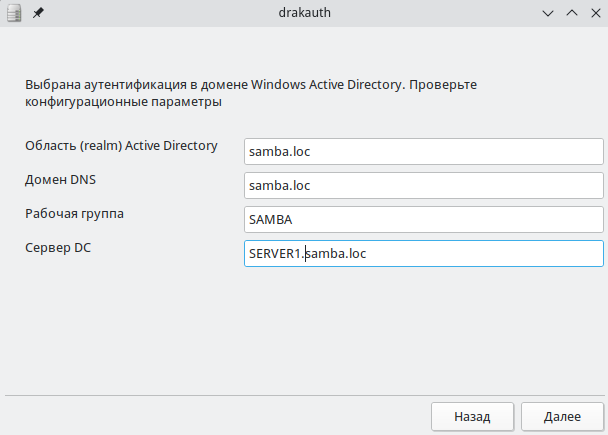


Рисунок 27 – Проверка параметров конфигурации в графическом интерфейсе drakauth

указать имя NetBIOS, например client1; поле "Описание компьютера" заполнять необязательно (рисунок 28);

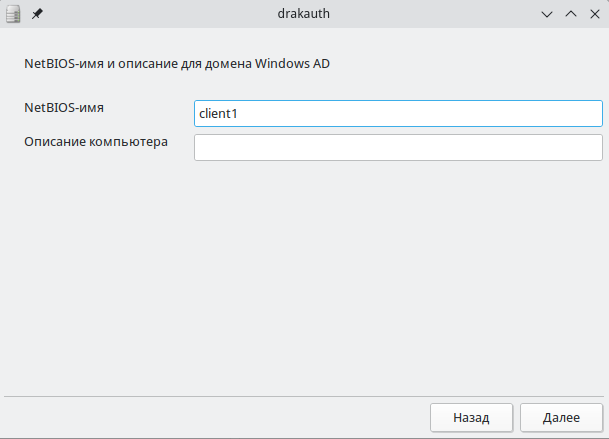


Рисунок 28 – Добавление описания для домена в графическом интерфейсе drakauth

ввести логин и пароль администратора домена (рисунок 29);

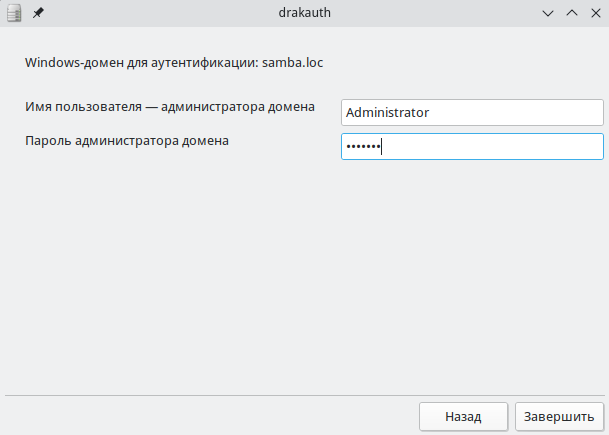


Рисунок – Назначение логина и пароля для администратора домена в графическом интерфейсе drakauth

В случае успешного ввода в домен будет предложено перезагрузить Систему; в случае возникновения ошибок при подключении домена на экран будет выведено соответствующее окно с сообщением и предложением проверить установленные настройки;

после перезагрузки можно проверить подключение командой:

wbinfo -u

Команда должна отобразить список доменных пользователей. Также можно проверить вход в Систему под доменной учётной записью.

* 1. Веб-серверы
     1. Apache

Для обеспечения работы веб-сервера используются следующие программные средства:

* Apache HTTP-сервер – веб-сервер с открытым исходным кодом;
* PHP – скриптовый язык программирования, созданный для генерации HTML-страниц на веб-сервере и работы с БД;
* MySQL – свободная система управления базами данных (СУБД);
* PhpMyAdmin – инструмент для визуальной работы с БД MySQL.
  + - 1. Установка Apache

Для установки веб-сервера Apache из консоли нужно установить один пакет, который по зависимостям установит еще порядка 40 пакетов, необходимых для нормальной и полноценной работы сервера.

Основные команды для работы с веб-сервером Apache:

* установка пакетов с Apache:

sudo dnf install apache-base

* запуск сервера Apache:

sudo systemctl start httpd

* добавление сервера Apache в автозапуск при старте Системы:

sudo systemctl enable httpd

* остановка сервера:

sudo systemctl stop httpd

После каких-либо изменений в конфигурационных файлах сервер следует перезапустить:

sudo systemctl restart httpd

* + - 1. Установка PHP

Для работы с PHP нужно установить всего 3 пакета, выполнив команду:

sudo dnf install php php-mysql apache-mod\_php

* + - 1. Установка MySQL

Для установки MySQL-сервера нужно выполнить команду:

sudo dnf install mysql-server

Запуск сервера MySQL:

sudo systemctl start mysqld

Включение запуска сервера при загрузке компьютера:

sudo systemctl enable mysqld

Перезапуск сервера:

sudo systemctl restart mysqld

* + - 1. Установка PhpMyAdmin

Перед установкой PhpMyAdmin необходимо проверить корректность предыдущих установок и настроек.

Для безопасности работы с БД MySQL нужно поменять пароль администратора:

mysqladmin -u root password новый\_пароль

Сначала необходимо запустить сервера в правильной последовательности:

sudo systemctl start mysqld

sudo systemctl start httpd

Теперь можно проверить работоспособность локального сервера, для чего набрать в адресной строке браузера адрес <http://localhost/>.

Если все прошло удачно, то страница браузера будет отображать сообщение с текстом об успешной работе (рисунок 30).

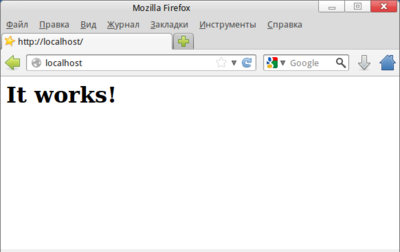


Рисунок 30 –Окно браузера при успешной установке сервера

Затем нужно проверить работу PHP, для этого в папке /var/www/htmlсоздать файлinfo.php с одной строчкой:

<?php phpinfo(); ?>

Теперь можно проверить работоспособность PHP на локальном сервере, набрав в адресной строке браузера адрес <http://localhost/info.php>. Окно браузера должно выглядеть так, как показано ниже (рисунок 31).

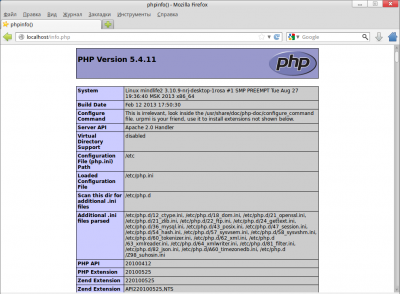


Рисунок 31 – Проверка работоспособности PHP

Далее следует прокрутить эту страницу ниже и убедиться в работоспособности MySQL. В разделе MySQL должен быть установлен статус Enabled (рисунок 32).

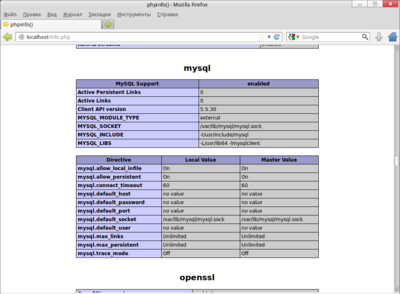


Рисунок 32 – Проверка статуса MySQL

После всех проверок установленных программных средств можно установить PhpMyAdmin командой:

sudo dnf install phpmyadmin

Затем необходимо заменить содержимое установленного по умолчанию файла /etc/httpd/conf/webapps.d/phpmyadmin.conf на следующее:

Alias /phpmyadmin /usr/share/phpmyadmin

<Directory /usr/share/phpmyadmin>

Options none

AllowOverride Limit

Require all granted

</Directory>

Чтобы проверить работу PhpMyAdmin, нужно набрать в адресной строке браузера адрес http://localhost/phpmyadmin/.

Если все сделано правильно, то в браузере появится примерное содержание, как ниже (рисунок 33).

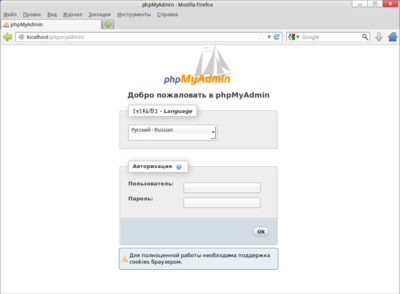


Рисунок 33 – Окно входа в PhpMyAdmin

* + 1. Веб-сервер Nginx

Nginx — это один из самых популярных веб-серверов, который обеспечивает работу множества высоконагруженных сайтов с большим объёмом трафика. По сравнению с Apache Nginx, как правило, использует ресурсы более эффективно и может применяться как полноценный веб-сервер, так и в роли обратного прокси-сервера.

Возможности сервера Nginx:

* обработка статических запросов, индексных файлов; автоматическое формирование списка файлов в директориях; кеширование дескрипторов открытых файлов;
* высокопроизводительное обратное проксирование с возможностью кеширования; поддержка балансировки нагрузки и механизмов отказоустойчивости;
* поддержка FastCGI, uwsgi, SCGI и memcached-серверов с кешированием, возможностью распределения нагрузки и организации отказоустойчивости;
* модульная архитектура, включая фильтры обработки контента (сжатие gzip, поддержка byte-ranges, chunked-ответов, XSLT-фильтров, SSI-фильтров, преобразование изображений);
* параллельная обработка подзапросов внутри одной страницы через SSI-фильтр с возможностью передачи запросов к сторонним FastCGI- или прокси-серверам;
* поддержка SSL и расширения TLS SNI;
* конфигурация виртуальных серверов по имени хоста или IP-адресу;
* поддержка keep-alive- и pipelined-соединений;
* гибкая система конфигурации;
* возможность изменения настроек и обновления исполняемого файла без прерывания обслуживания клиентов;
* настройка форматов журналов, буферизация и ротация логов;
* отображение специальных страниц ошибок (3xx–5xx);
* модуль rewrite с поддержкой регулярных выражений для перенаправлений;
* ограничение доступа по IP-адресам, HTTP Basic-аутентификация;
* проверка заголовка HTTP referer;
* поддержка методов HTTP: PUT, DELETE, MKCOL, COPY, MOVE;
* потоковая передача мультимедиа (FLV и MP4);
* ограничение скорости ответов и числа одновременных подключений;
* встроенная поддержка Perl-скриптов.

Nginx может использоваться как прокси-сервер для почтовых протоколов с проверкой пользователей через внешний HTTP-сервер аутентификации. Поддерживаются протоколы POP3, IMAP и SMTP с методами авторизации USER/PASS, APOP, AUTH LOGIN/PLAIN/CRAM-MD5. Также реализована поддержка SSL, STARTTLS и STLS для организации защищённых соединений при работе с почтовыми сервисами.

Архитектура Nginx построена на модели одного главного и нескольких рабочих процессов, где рабочие процессы запускаются под непривилегированным пользователем. Для обработки большого количества одновременных соединений используются высокопроизводительные механизмы событий, включая kqueue (FreeBSD 4.1+), epoll (Linux 2.6+), rt signals (Linux 2.2.19+), /dev/poll (Solaris 7+), event ports (Solaris 10), select и poll.

В Nginx поддерживаются дополнительные функции событийных моделей, например: EV\_CLEAR, EV\_DISABLE, NOTE\_LOWAT, EV\_EOF и другие; реализована поддержка механизмов передачи файлов (sendfile, sendfile64, sendfilev), асинхронного ввода-вывода (AIO) и режимов DIRECTIO; кроме того, возможно применение accept-фильтров (FreeBSD 4.1+, NetBSD 5.0+) и TCP\_DEFER\_ACCEPT (Linux 2.4+).

Сервер Nginx демонстрирует низкое потребление памяти (порядка 2,5 МБ на 10000 неактивных keep-alive-соединений) и минимизацию операций копирования данных при передаче запросов, что обеспечивает его высокую производительность и масштабируемость.

Официальный сайт проекта: <https://nginx.org/ru/>.

* + - 1. Установка и структура файлов Nginx

Для установки веб-сервера Nginx необходимо выполнить следующую команду:

sudo dnf install nginx

После установки требуется запустить сервис Nginx:

sudo systemctl start nginx.service

Чтобы обеспечить автоматический запуск сервиса Nginx при старте ОС, необходимо выполнить команду:

sudo systemctl enable nginx.service

Структура файлов и директорий Nginx включает в себя:

* Каталог веб-контента /usr/share/nginx/html. Данная директория содержит веб-контент, доступный по умолчанию сразу после установки Nginx. В неё входит демонстрационная страница, которую выводит установленный по умолчанию веб-сервер. При необходимости размещения собственного контента или веб-приложений следует отредактировать конфигурацию и указать другую директорию.
* Каталоги конфигураций сервера:
* /etc/nginx/ – основной каталог, содержащий все файлы конфигурации Nginx;
* /etc/nginx/nginx.conf – главный конфигурационный файл, определяющий глобальные параметры работы веб-сервера. Для изменения параметров, применяемых ко всем виртуальным хостам, необходимо вносить правки именно в этот файл;
* /etc/nginx/conf.d/ – каталог, содержащий конфигурационные файлы для отдельных виртуальных хостов. Рекомендуется создавать для каждого сайта отдельный конфигурационный файл с расширением .conf, совпадающий по названию с доменным именем, например example.ru.conf;
* Каталоги журналов работы:
* /var/log/nginx/access.log – все запросы к веб-серверу фиксируются в данном файле, если в конфигурации Nginx не указано иное;
* /var/log/nginx/error.log – все ошибки и сообщения о сбоях работы веб-сервера регистрируются в этом журнале.
  + - 1. Настройка Nginx

Основным конфигурационным файлом Nginx является /etc/nginx/nginx.conf. При необходимости изменения настроек следует открыть данный файл и внести требуемые параметры.

После запуска сервиса Nginx рекомендуется проверить его работоспособность, открыв веб-браузер и введя адрес:

http://<IP-адрес>

где <IP-адрес> — это адрес сервера, на котором запущен Nginx.

Ниже приведён пример базовой конфигурации /etc/nginx/nginx.conf по умолчанию:

#user nobody;

worker\_processes 1;

#error\_log logs/error.log;

#error\_log logs/error.log notice;

#error\_log logs/error.log info;

#pid logs/nginx.pid;

events {

worker\_connections 1024;

}

http {

include mime.types;

default\_type application/octet-stream;

#log\_format main '$remote\_addr - $remote\_user\ [$time\_local] "$request" '

# '$status $body\_bytes\_sent "$http\_referer"\ '

# '"$http\_user\_agent"\ "$http\_x\_forwarded\_for"';

#access\_log logs/access.log main;

sendfile on;

#tcp\_nopush on;

#keepalive\_timeout 0;

keepalive\_timeout 65;

#gzip on;

server {

listen 80;

server\_name localhost;

#charset koi8-r;

#access\_log logs/host.access.log main;

location / {

root html;

index index.html index.htm;

}

#error\_page 404 /404.html;

# redirect server error pages to the static page\ /50x.html

#

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root html;

}

# proxy the PHP scripts to Apache listening on\ 127.0.0.1:80

#

#location ~ \.php$ {

# proxy\_pass http://127.0.0.1;

#}

# pass the PHP scripts to FastCGI server listening on\ 127.0.0.1:9000

#

#location ~ \.php$ {

# root html;

# fastcgi\_pass 127.0.0.1:9000;

# fastcgi\_index index.php;

# fastcgi\_param SCRIPT\_FILENAME\ /scripts$fastcgi\_script\_name;

# include fastcgi\_params;

#}

# deny access to .htaccess files, if Apache's\ document root

# concurs with nginx's one

#

#location ~ /\.ht {

# deny all;

#}

}

# another virtual host using mix of IP-, name-, and port-based configuration

#

#server {

# listen 8000;

# listen somename:8080;

# server\_name somename alias another.alias;

# location / {

# root html;

# index index.html index.htm;

# }

#}

# HTTPS server

#

#server {

# listen 443 ssl;

# server\_name localhost;

# ssl\_certificate cert.pem;

# ssl\_certificate\_key cert.key;

# ssl\_session\_cache shared:SSL:1m;

# ssl\_session\_timeout 5m;

# ssl\_ciphers HIGH:!aNULL:!MD5;

# ssl\_prefer\_server\_ciphers on;

# location / {

# root html;

# index index.html index.htm;

# }

#}

}

Примечания по параметрам конфигурации:

* user — определяет пользователя, от имени которого запускается процесс Nginx. В дистрибутиве ОС данный параметр закомментирован, поскольку сервер запускается под пользователем nginx по умолчанию;
* listen — задаёт порт, на котором принимает соединения сервер (например, 80 для HTTP);
* server\_name — определяет имя сервера; по умолчанию используется localhost;
* location / — определяет обработку запросов к корневому адресу сайта;
* root — путь к каталогу, из которого осуществляется выдача файлов сайта;
* index — список файлов, которые будут открываться по умолчанию при отсутствии указания конкретного имени файла в запросе.

**Настройка виртуальных хостов**

Для организации обслуживания нескольких сайтов (виртуальных хостов) рекомендуется выполнять следующие шаги (рассмотрен простой пример для двух сайтов):

1. подготовить структуру каталогов. В каталоге /var/www рекомендуется создать отдельные подкаталоги для каждого сайта, а внутри — директории html для хранения файлов сайта. Например:

mkdir -p /var/www/rosaserver1/html

mkdir -p /var/www/rosaserver2/html

установить владельца каталогов для текущего пользователя с использованием переменной окружения $USER:

chown -R $USER:$USER /var/www/rosaserver1/html

chown -R $USER:$USER /var/www/rosaserver2/html

для проверки корректности настройки рекомендуется создать простые страницы. Для редактирования содержимого первого сайта открыть файл командой:

nano /var/www/rosaserver1/html/index.html

добавить в файл следующий код:

<html>

<head>

<title>Welcome to RosaServer1!</title>

</head>

<body>

Success! The RosaServer1 block is working!

</body>

</html>

для редактирования содержимого второго сайта открыть файл командой:

nano /var/www/rosaserver2/html/index.html

добавить в файл следующий код:

<html>

<head>

<title>Welcome to RosaServer2!</title>

</head>

<body>

Success! The RosaServer2 block is working!

</body>

</html>

файлы необходимо сохранить и закрыть.

Организация структуры виртуальных хостов

Для хранения конфигураций виртуальных хостов требуется создать директории:

sudo mkdir /etc/nginx/sites-available

sudo mkdir /etc/nginx/sites-enabled

После этого необходимо указать в конфигурационном файле Nginx, что активные блоки server находятся в каталоге sites-enabled. Для этого следует открыть файл /etc/nginx/nginx.conf и в конце блока "http {}" добавить:

include /etc/nginx/sites-enabled/\*.conf;

**Создание файлов конфигурации виртуальных хостов**

В качестве шаблона может быть использован файл /etc/nginx/conf.d/virtual.conf. Рекомендуется скопировать его в каталог sites-available для каждого сайта:

cp /etc/nginx/conf.d/virtual.conf /etc/nginx/sites-available/rosaserver1.conf

cp /etc/nginx/conf.d/virtual.conf /etc/nginx/sites-available/rosaserver2.conf

После копирования необходимо открыть соответствующие файлы и отредактировать их. Например, для первого сайта:

sudo nano /etc/nginx/sites-available/rosaserver1.conf

Пример содержимого файла:

server {

listen 80;

server\_name rosaserver1;

location / {

root /var/www/rosaserver1/html;

index index.html index.htm;

}

}

Для второго сайта аналогично, изменяя параметры server\_name и root:

server {

listen 80;

server\_name rosaserver2;

location / {

root /var/www/rosaserver2/html;

index index.html index.htm;

}

}

Файлы необходимо сохранить и закрыть.

**Включение виртуальных хостов**

Для активации виртуальных хостов требуется создать символические ссылки из каталога sites-available в sites-enabled:

ln -s /etc/nginx/sites-available/rosaserver1.conf /etc/nginx/sites-enabled/rosaserver1.conf

ln -s /etc/nginx/sites-available/rosaserver2.conf /etc/nginx/sites-enabled/rosaserver2.conf

Необходимо выполнить перезапуск сервиса Nginx, чтобы применить новые настройки:

sudo systemctl restart nginx.service

Чтобы проверить работу виртуальных хостов, рекомендуется открыть веб-браузер и проверить доступность созданных сайтов по адресам:

http://rosaserver1/

http://rosaserver2/

При корректной настройке должны отобразиться созданные ранее демонстрационные страницы.

* + - 1. Настройка HTTPS/TLS в Nginx

Поддержка защищённых соединений с использованием протоколов SSL/TLS является одной из важнейших функций веб-сервера Nginx. Реализация HTTPS позволяет обеспечить конфиденциальность и целостность данных при обмене между пользователем и сервером.

Nginx предоставляет гибкие возможности для настройки TLS, включая выбор версий протокола, шифров, поддержку OCSP Stapling и других расширений безопасности.

Для тестовых целей или при отсутствии центра сертификации может использоваться самоподписанный сертификат. Для его создания необходимо выполнить следующую команду:

openssl req -x509 -nodes -days 365 -newkey rsa:2048 \

-keyout /etc/ssl/private/server.key \

-out /etc/ssl/certs/server.crt

В результате будет создан сертификат server.crt и соответствующий ему закрытый ключ server.key. Необходимо обеспечить надёжные права доступа к закрытому ключу, например:

chmod 600 /etc/ssl/private/server.key

Пример базовой конфигурации TLS в Nginx:

server {

listen 443 ssl;

server\_name example.local;

ssl\_certificate /etc/ssl/certs/server.crt;

ssl\_certificate\_key /etc/ssl/private/server.key;

ssl\_protocols TLSv1.2 TLSv1.3;

ssl\_ciphers HIGH:!aNULL:!MD5;

ssl\_prefer\_server\_ciphers on;

ssl\_session\_cache shared:SSL:10m;

ssl\_session\_timeout 10m;

location / {

root /var/www/html;

index index.html index.htm;

}

}

где:

* ssl\_protocols — перечень протоколов, разрешённых для использования. В целях безопасности рекомендуется ограничиться TLSv1.2 и TLSv1.3;
* ssl\_ciphers — определяет допустимые наборы шифров. Рекомендуется использовать HIGH с исключением слабых алгоритмов, например HIGH:!aNULL:!MD5;
* ssl\_prefer\_server\_ciphers — включает приоритет серверных шифров при согласовании алгоритма между клиентом и сервером;
* ssl\_session\_cache — определяет параметры кеширования сессий TLS, что уменьшает время повторных подключений;
* ssl\_session\_timeout — задаёт время жизни кешированной сессии.

После изменения конфигурации необходимо выполнить проверку синтаксиса командой:

sudo nginx -t

и перезапустить сервис для применения изменений:

sudo systemctl reload nginx

* + - 1. Настройка обратного проксирования в Nginx

Веб-сервер Nginx нередко используется не только для раздачи статического контента, но и в качестве высокопроизводительного обратного прокси-сервера (reverse proxy), который перенаправляет запросы на другие приложения, например: Apache, Tomcat, Node.js или другие веб-сервисы. Такой подход позволяет централизовать обработку SSL, уменьшить нагрузку на сервер приложений, реализовать балансировку нагрузки и повысить отказоустойчивость.

Пример, при котором Nginx принимает запросы на порт 80 и перенаправляет их на внутренний сервер Apache, работающий на порту 8080:

server {

listen 80;

server\_name example.local;

location / {

proxy\_pass http://127.0.0.1:8080;

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For\ $proxy\_add\_x\_forwarded\_for;

}

}

В данной конфигурации:

* все запросы отправляются на 127.0.0.1:8080 (Apache);
* передаются заголовки с исходным IP-адресом пользователя и именем хоста для правильной обработки логики приложения;
* сохраняется возможность ведения журналов с реальными IP пользователей.

Директива proxy\_pass указывает, куда именно будет перенаправляться запрос. Поддерживается схема http://, https:// или даже unix:/path/to/socket.

Например, следующая команда означает, что все запросы из данного блока location перенаправляются на указанный адрес:

proxy\_pass http://backend-app:8000;

При наличии нескольких серверов приложений можно использовать блок upstream для балансировки запросов, например:

upstream backend {

server 192.168.1.10;

server 192.168.1.11;

}

server {

listen 80;

location / {

proxy\_pass http://backend;

}

}

В этом примере Nginx будет распределять запросы по серверам 192.168.1.10 и 192.168.1.11 по умолчанию методом round-robin.

Для более сложных сценариев доступны опции sticky-сессий, настройки fail\_timeout и другие механизмы управления доступностью бэкендов.

Рекомендуется явно задать тайм-ауты для взаимодействия с проксируемыми серверами, чтобы избежать зависания соединений:

proxy\_connect\_timeout 5s;

proxy\_send\_timeout 30s;

proxy\_read\_timeout 30s;

send\_timeout 30s;

Эти параметры позволяют ограничить время установления соединения, время отправки и получения данных от сервера приложений, что существенно повышает стабильность работы при сетевых сбоях или ошибках на стороне бэкенда.

После внесения изменений необходимо проверить корректность конфигурации с помощью команды:

sudo nginx -t

и перезапустить веб-сервер:

sudo systemctl reload nginx

* + - 1. Механизмы защиты Nginx

Для повышения уровня безопасности веб-сервера Nginx рекомендуется применять ряд механизмов, которые помогут защитить сервис от перегрузок, злоупотреблений и атак типа DoS. К таким механизмам относятся:

* Ограничение числа одновременных подключений и скорости:
* Настройка ограничения числа подключений:

limit\_conn\_zone $binary\_remote\_addr zone=addr:10m;

limit\_conn addr 20;

* Настройка ограничения скорости запросов:

limit\_req\_zone $binary\_remote\_addr zone=req\_zone:10m\ rate=5r/s;

limit\_req zone=req\_zone burst=10 nodelay;

Такая конфигурация позволит производить не более 20 одновременных соединений с одного IP-адреса, а также не более 5 запросов в секунду.

* **Защита от слишком больших запросов**, которые могут вызвать отказ сервиса. Рекомендуется задать максимальный размер тела запроса:

client\_max\_body\_size 2m;

Значение может быть указано в мегабайтах или килобайтах в зависимости от предполагаемой нагрузки.

* Интеграция с внешними инструментами фильтрации, такими как fail2ban.

Fail2ban — это утилита, позволяющая отслеживать в журналах сервера подозрительные попытки входа или злоупотребления и автоматически блокировать IP-адреса злоумышленников через брандмауэр.

Для защиты Nginx обычно используется фильтр, анализирующий ошибки в файле журнала. Для установки и настройки фильтра необходимо выполнить следующие действия:

1. установить пакет fail2ban;
2. создать фильтр, например /etc/fail2ban/filter.d/nginx-http-auth.conf, с выражениями для поиска атак;
3. настроить jail (правило) в /etc/fail2ban/jail.local, например:

[nginx-http-auth]

enabled = true

filter = nginx-http-auth

port = http,https

logpath = /var/log/nginx/error.log

maxretry = 5

findtime = 600

bantime = 3600

После этого fail2ban будет блокировать IP-адреса, с которых происходят слишком частые ошибки аутентификации или другие подозрительные действия.

В комплексе с настройками fail2ban стоит использовать внутренние директивы Nginx, которые позволяют реализовать базовую защиту от DoS-атак:

* limit\_conn ограничивает число одновременных соединений на одного клиента;
* limit\_req ограничивает частоту запросов.

Они могут комбинироваться, создавая эффективный барьер против перегрузок сервиса и злоупотреблений.

После внесения изменений рекомендуется проверить корректность конфигурации:

sudo nginx -t

и применить её:

sudo systemctl reload nginx

* + - 1. Логирование и анализ в Nginx

Настроенные журналы логирования позволяют выявлять проблемы, отслеживать попытки несанкционированного доступа и анализировать нагрузку.

Рекомендуется выполнить комплексную настройку логирования по следующим этапам:

1. **Настройка формата** access.log**.** По умолчанию журнал запросов access.log ведётся с базовым форматом, который может быть расширен под конкретные требования. Для этого в конфигурации определяется директива log\_format:

log\_format main '$remote\_addr - $remote\_user [$time\_local]\ "$request" '

'$status $body\_bytes\_sent "$http\_referer" '

'"$http\_user\_agent" "$http\_x\_forwarded\_for"';

Эта конструкция позволит фиксировать IP-адрес клиента, дату, строку запроса, статус ответа, реферер и заголовки агента.

**Использование пользовательских форматов журналирования (custom log format).** Если требуется регистрировать дополнительные данные, например время ответа или значения cookie, можно добавить их в шаблон log\_format:

log\_format custom '$remote\_addr [$time\_local] "$request" '

'status=$status bytes=$body\_bytes\_sent '

'rt=$request\_time cookie="$http\_cookie"';

После чего необходимо указать данный формат в директиве access\_log:

access\_log /var/log/nginx/access.log custom;

**Ротация логов с использованием** logrotate**.** Для автоматического ограничения размера журналов и предотвращения переполнения файловой системы используется система logrotate. В дистрибутивах ОС конфигурация логирования nginx обычно хранится в файле /etc/logrotate.d/nginx. Пример типичной настройки:

/var/log/nginx/\*.log {

daily

missingok

rotate 7

compress

delaycompress

notifempty

create 0640 nginx adm

sharedscripts

postrotate

sudo systemctl reload nginx.service > /dev/null 2>&1 || true

endscript

}

Такая конфигурация обеспечивает ежедневную ротацию, хранение семи архивных копий, сжатие старых журналов и автоматическую перезагрузку службы после ротации.

В примере типичной настройки:

* daily — выполняется ротация журналов ежедневно;
* rotate 7 — хранить 7 предыдущих архивных копий;
* compress — при ротации старые файлы сжимаются с помощью gzip;
* delaycompress — откладывает сжатие на один цикл ротации, чтобы не сжимать самый новый архив сразу;
* postrotate ... endscript — блок команд, который выполняется после ротации; в данном случае перезапускает конфигурацию сервиса Nginx для корректного использования нового файла журнала.

**Интеграция с централизованной системой журналирования.** При необходимости можно перенаправить журналы Nginx в syslog или системный журнал journald. Для этого в конфигурации сервера указывается:

access\_log syslog:server=unix:/dev/log,facility=local7,tag=nginx\_access;

error\_log syslog:server=unix:/dev/log,facility=local7,tag=nginx\_error warn;

* + - 1. Резервное копирование и восстановление Nginx

Для обеспечения стабильной и безопасной работы веб-сервера Nginx необходимо предусмотреть процедуры регулярного резервного копирования его конфигурационных файлов, а также описать порядок их восстановления.

Рекомендуется выполнять следующие действия:

1. **Определить критически важные файлы и каталоги для резервного копирования.** В резервные копии должны обязательно включаться:

* основной файл конфигурации /etc/nginx/nginx.conf;
* все файлы виртуальных хостов, расположенные в каталогах /etc/nginx/conf.d/, /etc/nginx/sites-available/ и /etc/nginx/sites-enabled/;
* пользовательские SSL-сертификаты и ключи в /etc/ssl/certs/ и /etc/ssl/private/;
* при необходимости скрипты и шаблоны мониторинга или логирования, интегрированные с Nginx.

1. Создавать сжатую копию конфигурации и сертификатов не реже одного раза в неделю с сохранением нескольких предыдущих версий архива, используя команду:

sudo tar czf /backup/nginx-config-$(date +%Y%m%d).tar.gz /etc/nginx /etc/ssl

Такой архив может храниться на отдельном сервере резервного копирования, в том числе с применением средств шифрования при передаче или хранении.

1. Для восстановления после сбоя необходимо выполнить:

* распаковку резервного архива в стандартные каталоги:

sudo tar xzf nginx-config-20250620.tar.gz -C /

* проверку корректности прав доступа и владельцев файлов;
* проверку конфигурации веб-сервера командой:

sudo nginx -t

* перезагрузку службы для применения восстановленной конфигурации:

sudo systemctl reload nginx

При использовании системы управления версиями (например, Git) возможно организовать дополнительный контроль изменений и отслеживать историю правок в конфигурационных файлах.

* + - 1. Обновление и управление версиями Nginx

Для контроля актуальности установленной версии рекомендуется:

* проверять текущую версию командой nginx -v;
* при необходимости уточнять доступные обновления в репозитории с помощью:

dnf info nginx

Процесс обновления Nginx выполняется стандартным менеджером пакетов dnf:

sudo dnf update nginx

После обновления либо изменения конфигурационных файлов необходимо выполнить безопасную перезагрузку службы с сохранением активных соединений:

sudo nginx -s reload

Этот вариант позволяет без прерывания обслуживания клиентов применить новую конфигурацию или загрузить обновлённый бинарный файл.

* + 1. Веб-сервер Angie

Веб-сервер Angie представляет собой современный, эффективный и масштабируемый программный продукт, реализованный как форк Nginx. Angie создан бывшими разработчиками исходного проекта с целью развития решения в новом направлении. При этом Angie сохраняет совместимость с конфигурацией и модулями Nginx, что позволяет использовать его в качестве замены без необходимости доработки существующих настроек.

Angie включает все возможности версии Nginx 1.25.3 и расширяет их рядом дополнительных функций, среди которых:

* Поддержка HTTP/3 как для клиентских соединений, так и для соединений с проксируемыми серверами с возможностью раздельного использования различных протоколов (HTTP/1.x, HTTP/2, HTTP/3) на разных направлениях.
* Упрощение конфигурации за счёт возможности в директиве location задавать несколько строк сопоставления и объединять блоки с одинаковыми настройками.
* Получение базовой информации о веб-сервере, его конфигурации, а также статистики по клиентским соединениям, проксируемым серверам, зонам разделяемой памяти и другим параметрам через REST-подобный API-интерфейс в формате JSON.
* Возможность экспорта метрик и статистики в формате Prometheus с гибкими шаблонами.
* Интеграция с визуальной консолью мониторинга Console Light, доступной через веб-браузер (онлайн-пример консоли доступен по адресу: <https://console.angie.software/>).
* Автоматическое обновление списка проксируемых серверов по доменным именам, включая возможность получения данных из DNS-записей SRV.
* Режим привязки сессий, позволяющий направлять все запросы в рамках одной сессии на один и тот же проксируемый сервер.
* Реализация плавного ввода проксируемого сервера после сбоя с использованием опции slow\_start директивы server.
* Ограничение скорости отдачи MP4-файлов с учётом их битрейта, что снижает нагрузку на сеть.
* Поддержка директивы mqtt\_preread в модуле stream, расширяющая возможности балансировки и аутентификации по протоколу MQTT.
* Наличие готовых бинарных пакетов для интеграции со многими сторонними модулями.

Официальный сайт проекта: <https://angie.software/>

* + - 1. Установка Angie

Для установки веб-сервера Angie необходимо выполнить следующие действия:

1. установитьпакет Angie из репозитория:

sudo dnf install angie

1. запустить сервис веб-сервера:

sudo systemctl start angie.service

1. включить автоматический запуск при загрузке:

sudo systemctl enable angie.service

* + - 1. Настройка и управление веб-сервером Angie

По умолчанию структура каталогов Angie совпадает с Nginx:

* /usr/share/angie/html — директория веб-контента;
* /etc/angie/ — основной каталог конфигурации;
* /etc/angie/angie.conf — главный конфигурационный файл;
* /etc/angie/conf.d/ — каталог для виртуальных хостов;
* /var/log/angie/access.log — журнал запросов;
* /var/log/angie/error.log — журнал ошибок.

Настройка веб-сервера Angie осуществляется по тем же принципам, что и настройка Nginx, благодаря полной совместимости конфигурационного синтаксиса. При этом могут использоваться как стандартные подходы конфигурации, так и расширенные возможности, добавленные в Angie.

После установки рекомендуется проверить работу веб-сервера. Для этого следует открыть браузер и перейти по адресу сервера (например, http://<IP-адрес>). При корректном запуске будет отображена стандартная страница Angie.

Основным конфигурационным файлом Angie является /etc/angie/angie.conf. В этом файле задаются глобальные параметры работы сервера.

Рекомендуется придерживаться следующего подхода:

* все изменения в глобальных настройках вносить в angie.conf;
* файлы конфигурации отдельных виртуальных хостов хранить в каталоге /etc/angie/conf.d/;
* при большом числе сайтов структурировать конфигурацию с помощью отдельных директорий sites-available и sites-enabled аналогично практике Nginx.

Для активации виртуальных хостов может быть добавлена директива include /etc/angie/sites-enabled/\*.conf; в блок "http {}" основного конфигурационного файла.

Ниже приведён пример минимальной конфигурации виртуального хоста для Angie:

server {

listen 80;

server\_name example.local;

location / {

root /var/www/example.local/html;

index index.html index.htm;

}

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root /usr/share/angie/html;

}

}

В приведённой конфигурации задаётся имя сервера, корневая директория сайта и базовая обработка ошибок.

После изменения конфигурационных файлов необходимо произвести проверку корректности синтаксиса:

sudo angie -t

В случае отсутствия ошибок конфигурацию рекомендуется загрузить без прерывания активных соединений:

sudo systemctl reload angie.service

* + - 1. Дополнительные функции Angie

Angie расширяет традиционные функции Nginx следующими механизмами, которые могут быть востребованы при настройке:

* slow\_start — позволяет плавно вводить проксируемый сервер в работу после сбоя. Настраивается в блоке upstream:

upstream backend {

server 192.168.1.100;

server 192.168.1.101;

slow\_start 30s;

}

В данном примере повторное включение сервера будет происходить в течение 30 секунд с постепенным увеличением доли запросов.

* mqtt\_preread — директива для протокола MQTT, применяемая в модуле stream:

stream {

mqtt\_preread on;

server {

listen 1883;

proxy\_pass mqtt\_backend;

}

}

mqtt\_preread используется для расширенной аутентификации и балансировки MQTT-соединений.

* HTTP/3 — поддержка нового протокола включается в секции listen:

server {

listen 443 quic reuseport;

http3 on;

ssl\_certificate /etc/ssl/certs/server.crt;

ssl\_certificate\_key /etc/ssl/private/server.key;

...

}

Здесь параметр quic включает использование UDP и HTTP/3, при этом по-прежнему возможен fallback на HTTP/1.x или HTTP/2.

* + - 1. Мониторинг и API

Angie предоставляет REST-подобный API для получения данных о состоянии сервера, текущей конфигурации, статистике по зонам разделяемой памяти, соединениям и проксируемым серверам.

В ответах API используется формат JSON, что позволяет легко интегрировать Angie с системами мониторинга. Также поддерживается экспорт метрик Prometheus с гибкой настройкой шаблонов.

Для удобного визуального контроля может использоваться консоль Console Light. Её работа не требует дополнительной установки — необходимо лишь активировать соответствующий блок в конфигурации и открыть веб-интерфейс по адресу сервера.

* + - 1. Настройка HTTPS/TLS в Angie

Angie поддерживает все основные механизмы TLS/SSL, применяемые в современных веб-серверах. При этом полностью сохраняется совместимость с синтаксисом Nginx.

Для включения защищённых соединений необходимо выполнить следующие шаги:

* сгенерировать ключ и сертификат (например, с помощью openssl);
* разместить их в защищённой директории, доступной только пользователю, от имени которого работает Angie;
* описать соответствующую секцию server с параметрами listen и ssl\_certificate.

Пример минимальной конфигурации HTTPS:

server {

listen 443 ssl;

server\_name example.local;

ssl\_certificate /etc/ssl/certs/server.crt;

ssl\_certificate\_key /etc/ssl/private/server.key;

ssl\_protocols TLSv1.2 TLSv1.3;

ssl\_ciphers HIGH:!aNULL:!MD5;

location / {

root /var/www/example.local/html;

index index.html index.htm;

}

}

В данном примере задаются:

* сертификат сервера (ssl\_certificate);
* закрытый ключ (ssl\_certificate\_key);
* допустимые протоколы tls (рекомендуется ограничить tlsv1.2 и tlsv1.3);
* набор безопасных шифров (ssl\_ciphers).

Рекомендуется придерживаться следующих принципов для безопасной настройки веб-сервера:

* сертификаты и ключи следует хранить в директории с ограниченными правами доступа;
* не использовать ssl-протоколы ниже tlsv1.2;
* при работе с публичными центрами сертификации (ca) важно настроить ocsp stapling для повышения надёжности;
* желательно активировать параметр ssl\_prefer\_server\_ciphers on, чтобы сервер контролировал выбор шифра.

После внесения всех изменений в конфигурационные файлы необходимо выполнить проверку синтаксиса:

sudo angie -t

После проверки необходимо перезапустить службу без обрыва соединений:

sudo systemctl reload angie.service

* + - 1. Настройка обратного проксирования в Angie

Веб-сервер Angie, как и Nginx, может использоваться в качестве обратного прокси (reverse proxy) для передачи запросов к другим серверам приложений, например: Apache, Tomcat, Node.js и др. Данный механизм позволяет централизовать обработку SSL, балансировать нагрузку и распределять трафик.

Для настройки обратного проксирования необходимо создать блок server, в котором используется директива proxy\_pass. Ниже приведён минимальный пример:

server {

listen 80;

server\_name proxy.example.local;

location / {

proxy\_pass http://127.0.0.1:8080;

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

}

}

В данной конфигурации все запросы, поступающие на порт 80, перенаправляются на внутренний сервер по адресу 127.0.0.1:8080. Дополнительно устанавливаются заголовки для передачи исходного IP-адреса клиента и имени хоста.

Балансировка нагрузки

Angie поддерживает балансировку между несколькими серверами приложений с помощью блока upstream, например:

upstream backend {

server 192.168.1.10 max\_fails=3 fail\_timeout=30s;

server 192.168.1.11 max\_fails=3 fail\_timeout=30s;

}

server {

listen 80;

location / {

proxy\_pass http://backend;

}

}

В данной конфигурации создаётся пул из двух серверов с балансировкой по умолчанию (round-robin) и базовым механизмом проверки доступности через параметры max\_fails и fail\_timeout.

Привязка сессий

Для случаев, когда требуется сохранять сессию пользователя на одном и том же бэкенде, используется механизм привязки сессий (sticky sessions**)**, например:

upstream backend {

server 192.168.1.10;

server 192.168.1.11;

sticky cookie srv\_id expires=1h domain=.example.local path=/;

}

server {

listen 80;

location / {

proxy\_pass http://backend;

}

}

При данной настройке создаётся cookie srv\_id, позволяющее направлять все запросы пользователя в течение часа на один и тот же сервер.

Тайм-ауты и ограничения

Рекомендуется дополнительно настроить параметры тайм-аутов, чтобы защитить сервер от зависших соединений:

proxy\_connect\_timeout 5s;

proxy\_send\_timeout 30s;

proxy\_read\_timeout 30s;

send\_timeout 30s;

Также могут использоваться лимиты на количество соединений с одного адреса или ограничение скорости:

limit\_conn\_zone $binary\_remote\_addr zone=addr:10m;

limit\_conn addr 10;

limit\_req\_zone $binary\_remote\_addr zone=req:10m rate=1r/s;

limit\_req zone=req burst=5;

После внесения всех изменений в конфигурационные файлы необходимо выполнить проверку синтаксиса:

sudo angie -t

После проверки необходимо перезапустить службу без обрыва соединений:

sudo systemctl reload angie.service

* + - 1. Ограничения и защита в Angie

В целях повышения безопасности и устойчивости веб-сервера Angie рекомендуется настроить ряд механизмов ограничения ресурсов и фильтрации запросов. Это позволяет минимизировать риск DoS-атак, защитить сервис от чрезмерной нагрузки и контролировать активность пользователей:

* Ограничение количества подключений от одного IP-адреса директивой limit\_conn:

limit\_conn\_zone $binary\_remote\_addr zone=addr:10m;

limit\_conn addr 20;

где для IP-адреса создаётся зона размером 10 МБ и допускается не более 20 одновременных подключений.

* Ограничение скорости передачи данных к клиентам с помощью limit\_rate:

server {

location / {

limit\_rate 500k;

}

}

где задан параметр "limit\_rate 500k", который позволит не превышать скорость 500 КБ в секунду для одного соединения.

* Ограничение скорости запросов с помощью limit\_req:

limit\_req\_zone $binary\_remote\_addr zone=req\_zone:10m rate=5r/s;

limit\_req zone=req\_zone burst=10 nodelay;

где задается не более 5 запросов в секунду на один IP с возможностью коротких всплесков до 10 запросов подряд.

* Фильтрация доступа к ресурсу по IP-адресам можно использовать директивы allow и deny:

server {

location / {

allow 192.168.1.0/24;

deny all;

}

}

В данном случае доступ будет разрешён только из подсети 192.168.1.0/24, остальные подключения будут отклоняться.

* Защита от медленных атак типа slowloris с помощью настройки тайм-аутов:

client\_body\_timeout 10s;

client\_header\_timeout 10s;

send\_timeout 30s;

keepalive\_timeout 15s;

Эти параметры ограничивают время ожидания данных от клиента, уменьшая вероятность блокировки рабочих процессов медленными соединениями.

* Для дополнительной защиты возможно использование fail2ban, который анализирует журналы ошибок Angie и блокирует IP-адреса с подозрительной активностью. Рекомендуется настроить fail2ban с фильтром для error.log и указанием параметров бана в файле /etc/fail2ban/jail.conf или отдельном jail-файле, например:

[angie]

enabled = true

filter = angie

port = http,https

logpath = /var/log/angie/error.log

maxretry = 5

findtime = 600

bantime = 3600

* + - 1. Логирование и ротация журналов в Angie

Веб-сервер Angie обладает гибкими возможностями по ведению журналов запросов и ошибок, которые позволяют контролировать его работу, проводить аудит и своевременно реагировать на возникающие сбои или аномалии.

По умолчанию Angie сохраняет журналы обращений в файл /var/log/angie/access.log, а ошибки — в файл /var/log/angie/error.log. Форматы записей можно изменять в соответствии с требованиями эксплуатации, например, добавив в них дополнительную информацию о заголовках, времени ответа, реферере или пользовательском агенте. Для этого используется директива log\_format. Пример расширенного шаблона:

log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '

'$status $body\_bytes\_sent "$http\_referer" '

'"$http\_user\_agent" "$http\_x\_forwarded\_for"';

access\_log /var/log/angie/access.log main;

Уровень детализации сообщений об ошибках регулируется директивой error\_log. Например, для вывода предупреждений и ошибок достаточно указать уровень warn:

error\_log /var/log/angie/error.log warn;

При эксплуатации нескольких виртуальных хостов рекомендуется организовать отдельные журналы для каждого сайта, что позволит быстрее анализировать проблемы и распределять нагрузку при аудите, например:

server {

server\_name site1.local;

access\_log /var/log/angie/site1\_access.log;

error\_log /var/log/angie/site1\_error.log warn;

}

Также Angie поддерживает вывод журналов через системный журнал journald или через syslog-сервер. Для этого в настройках access\_log и error\_log можно указать:

access\_log syslog:server=unix:/dev/log,facility=local7,tag=angie\_access;

error\_log syslog:server=unix:/dev/log,facility=local7,tag=angie\_error warn;

Это обеспечивает централизованный сбор журналов для последующего анализа средствами системного журнала.

Ротация журналов

Для управления размерами файлов журналов и автоматического их сжатия рекомендуется применять системную утилиту logrotate. При стандартной установке создаётся конфигурация в каталоге /etc/logrotate.d/angie, которая может быть настроена, например, следующим образом:

/var/log/angie/\*.log {

daily

missingok

rotate 7

compress

delaycompress

notifempty

create 0640 nginx adm

sharedscripts

postrotate

sudo systemctl reload angie.service > /dev/null 2>&1 || true

endscript

}

В приведённой конфигурации ротация выполняется ежедневно с хранением семи предыдущих архивов, причём старые логи сжимаются для экономии места. После ротации автоматически отправляется команда перезагрузки конфигурации сервера, чтобы Angie корректно переключился на новый лог-файл.

* + - 1. Мониторинг и диагностика в Angie

Для эффективного администрирования веб-сервера Angie необходимо обеспечить постоянный мониторинг его состояния и использование встроенных инструментов диагностики.

Angie предоставляет расширенные возможности мониторинга за счёт встроенного REST API и визуальной консоли Console Light. REST API позволяет в реальном времени получать сведения о конфигурации, состоянии клиентских соединений, статистике зон разделяемой памяти, а также о параметрах работы проксируемых серверов. Данные возвращаются в формате JSON, что упрощает интеграцию с внешними системами мониторинга, например Zabbix или Prometheus.

Для Prometheus в Angie реализован собственный экспорт метрик с возможностью настройки шаблонов. Это позволяет агрегировать показатели нагрузки и работы сервера в стандартных панелях мониторинга, а также формировать предупреждения при выходе параметров за заданные пределы.

Кроме того, Angie поддерживает визуальную консоль Console Light, которая позволяет в браузере наблюдать за ключевыми показателями веб-сервера. Console Light не требует отдельной установки и доступна сразу после включения в конфигурации. Для её активации достаточно добавить в основной конфигурационный файл соответствующий блок и открыть указанный интерфейс по адресу сервера.

Мониторинг и диагностика могут дополняться средствами анализа логов. Рекомендуется:

* отслеживать изменения в журналах ошибок и доступа с помощью системного журнала или logrotate;
* контролировать ключевые тайм-ауты и показатели использования памяти;
* проверять своевременность применения обновлений конфигурации с помощью команды angie -t;
* анализировать производительность с помощью отчётов Prometheus или других инструментов.
  + - 1. Резервное копирование и восстановление конфигурации Angie

Для обеспечения надёжной и стабильной работы веб-сервера Angie важно организовать корректное резервное копирование его конфигурационных файлов, а также предусмотреть процедуры быстрого восстановления при сбоях или ошибках конфигурации.

В первую очередь необходимо включить в резервную копию следующие элементы:

* Основной файл конфигурации /etc/angie/angie.conf.
* Все файлы виртуальных хостов, например, в каталогах /etc/angie/conf.d/, /etc/angie/sites-available/ и /etc/angie/sites-enabled/.
* Пользовательские сертификаты и ключи в каталогах /etc/ssl/certs/ и /etc/ssl/private/.
* Файлы журналов конфигурационных шаблонов мониторинга (если используются).

Рекомендуется использовать утилиты tar или rsync, позволяющие быстро создать резервную копию всей структуры конфигурации вместе с правами доступа (подробнее о работе с утилитами рассмотрено в разделе 7). Пример команды:

sudo tar czf /backup/angie-config-$(date +%Y%m%d).tar.gz /etc/angie /etc/ssl

Архив, созданный таким образом, можно переместить на внешний носитель или в защищённое хранилище.

При необходимости восстановления достаточно выполнить распаковку архива в исходное место и проверить корректность прав доступа:

sudo tar xzf angie-config-20250620.tar.gz -C /

После восстановления рекомендуется:

* убедиться в целостности всех файлов конфигурации;
* проверить синтаксис конфигурации командой angie -t;
* выполнить перезапуск или перезагрузку конфигурации angie:

sudo systemctl reload angie.service

Для повышения надёжности рекомендуется дополнительно вести журнал изменений конфигурации, фиксируя дату, причину и перечень изменений. Это позволит избежать ошибок при откате и быстрее выявлять некорректные правки.

Также следует рассмотреть возможность интеграции конфигурационных файлов Angie в систему управления версиями (например, Git), что обеспечит контроль версий и удобное отслеживание изменений между разными этапами эксплуатации.

Таким образом, регулярное резервное копирование и чётко прописанные процедуры восстановления позволяют минимизировать риски при эксплуатации веб-сервера Angie и быстро возвращать его к работоспособному состоянию в случае сбоев.

* + - 1. Миграция с Nginx на Angie

Поскольку Angie является форком Nginx и сохраняет полную совместимость с его конфигурационным синтаксисом, процесс миграции обычно не вызывает значительных трудностей и может быть выполнен с минимальными изменениями.

Angie поддерживает все основные директивы и модули, используемые в Nginx, включая виртуальные хосты, обратное проксирование, балансировку нагрузки, TLS/SSL, фильтры, а также возможности журналирования. Поэтому в большинстве случаев достаточно скопировать существующие файлы конфигурации Nginx в директории конфигурации Angie.

При выполнении миграции с Nginx на Angie рекомендуется придерживаться следующего порядка:

1. Подготовка и анализ конфигурации:

* проверить актуальность версий Nginx и Angie;
* необходимо убедиться, что установленные модули Nginx не используют закрытые проприетарные расширения, отсутствующие в Angie;
* сохранить резервную копию всех конфигурационных файлов Nginx;

1. Установка Angie:

* установка пакета Angie через менеджер пакетов dnf;
* проверка доступности службы и наличия необходимых зависимостей;
* отключение службу Nginx для исключения конфликта портов;

1. Перенос конфигурации:

* копирование данных из /etc/nginx в /etc/angie;
* проверка всех include-директив, которые должны указывать корректные пути;
* убедиться в корректности путей к логам и каталогам виртуальных хостов;

1. Тестирование:

* проверка синтаксис конфигурации Angie командой angie -t;
* локальное тестирование на стенде для проверки работоспособности всех виртуальных хостов, TLS, проксирования и других функций;
* при необходимости корректировка параметров, добавление новых возможностей Angie (например, http3, slow\_start).

1. Ввод в эксплуатацию:

* запуск службу Angie;
* убедиться в корректном функционировании всех сайтов и сервисов;
* настройка мониторинга и журналирование для оперативного контроля состояния.

1. Рекомендации для выполнения успешной миграции:

* обязательно сохранять резервные копии всех конфигурационных файлов до начала миграции;
* тестировать работы новых функций Angie сначала в тестовом окружении;
* при работе с динамическими модулями необходимо проверить их совместимости;
* вести журнал всех действий по миграции и фиксировать возможные проблемы для последующего анализа.
  1. Веб-интерфейс управления Cockpit

Cockpit представляет собой инструмент для администрирования устройств под управлением ОС РОСА "ХРОМ" и других ОС на основе Linux, основанный на удобном веб-интерфейсе. Cockpit позволяет выполнять основные задачи системного администрирования с помощью любого современного веб-браузера, обеспечивая простое и централизованное управление сервером без необходимости подключаться к нему через консоль напрямую.

С помощью Cockpit можно:

* просматривать состояние ресурсов сервера (процессор, память, диски);
* управлять службами и процессами;
* контролировать состояние сети и настраивать сетевые интерфейсы;
* отслеживать журналы системы;
* управлять пользователями и группами;
* обновлять пакеты;
* подключаться к консоли системы прямо из браузера.

Cockpit может служить важным элементом инфраструктуры администрирования, особенно в распределённых или изолированных средах.

Для установки Cockpit вместе с пакетом расширенного мониторинга PCP следует выполнить команду:

sudo dnf install cockpit cockpit-pcp

После установки необходимо активировать сокет Cockpit, чтобы сервис запускался при старте Системы:

sudo systemctl enable cockpit.socket

Для немедленного запуска службы используется команда:

sudo systemctl start cockpit.socket

После запуска веб-интерфейс Cockpit будет доступен по адресу:

http://<ip-адрес сервера>:9090/

где <ip-адрес сервера> — это IP-адрес или полное доменное имя хоста, на котором установлен Cockpit.

Cockpit поддерживает установку дополнительных модулей, расширяющих его возможности. Чтобы получить список доступных модулей в репозитории, используется команда:

dnf search cockpit-

Основные модули Cockpit:

* cockpit-machines — управление виртуальными машинами (libvirt/KVM), включая создание, удаление и настройку ВМ;
* cockpit-storaged — управление дисками, разделами, томами LVM и RAID-массивами;
* cockpit-networkmanager — удобное управление сетевыми интерфейсами через NetworkManager;
* cockpit-sosreport — сбор отчётов о состоянии Системы для последующего анализа;
* cockpit-podman — работа с контейнерами Podman, включая их запуск и мониторинг;
* cockpit-kubernetes — управление кластерами Kubernetes и отслеживание их состояния;
* cockpit-packagekit — обновление и установка программного обеспечения через интерфейс PackageKit;
* cockpit-selinux — просмотр и базовое управление политиками SELinux;
* cockpit-composer — создание и генерация собственных ISO-образов и готовых для развёртывания систем;
* cockpit-pcp — расширенный мониторинг производительности с помощью Performance Co-Pilot (PCP);
* cockpit-navigator — интеграция с командной строкой в стиле Midnight Commander для удобного управления файлами.

Для установки необходимого модуля нужно выполнить:

sudo dnf install cockpit-<модуль>

где <модуль> — это имя выбранного расширения.

После установки Cockpit рекомендуется:

* ограничить доступ к порту 9090 только с доверенных IP-адресов через брандмауэр;
* при необходимости настроить HTTPS для защиты передаваемых данных, указав путь к TLS-сертификату и приватному ключу в конфигурации Cockpit;
* ограничить список пользователей, имеющих право входа, через стандартные механизмы контроля групп (например, группа wheel).
  1. Сервер журналирования
     1. Общие сведения

Централизованный сервер журналирования может помочь с анализом событий на компьютерах в ИТ-инфраструктуре, объединенной в систему. Настроить их сбор с журналов можно с помощью journald.

В ОС используется сервис journald. Journald входит в состав systemd.

В systemd предусмотрены специальные компоненты для решения задачи логирования: systemd-journal-remote, systemd-journal-upload и systemd-journal-gatewayd.

* + 1. Установка сервера

Для работы нужно установить пакет systemd-journal-gateway на всех трех хостах, перечисленных ниже, командой:

sudo dnf install systemd-journal-gateway

В качестве примера в пп.12.6.3–12.6.6 будет использовано три разных хоста:

* rosaserver – сервер логирования journald;
* rosaserver1 – первый клиент (сервер/десктоп);
* rosaserver2 – второй клиент (сервер/десктоп).
  + 1. Настройка сервера

Настройка состоит из двух частей. В первой части настраивается сам сервер с именем хоста rosaserver.

Во второй части настраиваются два клиента (сервера/десктопа) c именами хостов/IP: rosaserver1/192.168.1.111 и rosaserver2/192.168.1.232.

В первую очередь осуществляется настройка systemd-journal-remote на сервере. В ходе настройки необходимо совершить несколько шагов:

1. Прописать в файле /etc/hosts клиентов:

192.168.1.111 rosaserver1

192.168.1.232 rosaserver2

1. Создать директорию /var/log/journal/remote, в которой будут храниться все журналы с клиентов:

mkdir -p /var/log/journal/remote

1. Предоставить права на эту директорию:

chown systemd-journal-remote:systemd-journal-remote /var/log/journal/remote

Для дальнейшей настройки сервера следует выполнить следующие действия:

1. Создать директорию для сертификата:

sudo mkdir /etc/pki/tls/ca

sudo ln -s /etc/pki/tls/ca /etc/ssl/ca

Остальные директории /etc/ssl/certs и /etc/ssl/private уже созданы.

1. Сгенерировать ключ сервера:

openssl req -newkey rsa:2048 -days 3650 -x509 -nodes -out /etc/ssl/ca/caserver.pem -keyout /etc/ssl/ca/caserver.key -subj '/CN=Certificate authority/'

1. Сгенерировать сертификаты для сервера (опция CN-SERVER должна совпадать с именем хоста сервера; в данном случае /CN=rosaserver):

openssl req -newkey rsa:2048 -nodes -out /etc/ssl/certs/rosaserver.csr -keyout /etc/ssl/certs/rosaserver.key -subj "/CN=rosaserver/"

openssl x509 -req -in /etc/ssl/certs/rosaserver.csr -CA /etc/ssl/ca/caserver.pem -CAkey /etc/ssl/ca/caserver.key -out /etc/ssl/private/rosaserver.pem

1. Сгенерировать сертификаты для первого клиента (опция CN- CLIENT должна совпадать с именем хоста клиента, в данном случае /CN=rosaserver1):

openssl req -newkey rsa:2048 -nodes -out /etc/ssl/certs/rosaserver1.csr -keyout /etc/ssl/certs/rosaserver1.key -subj "/CN=rosaserver1/"

openssl x509 -req -in /etc/ssl/certs/rosaserver1.csr -CA /etc/ssl/ca/caserver.pem -CAkey /etc/ssl/ca/caserver.key -out /etc/ssl/private/rosaserver1.pem

1. Генерировать сертификаты для второго клиента (опция CN-CLIENT должна совпадать с именем хоста клиента, в данном случае /CN=rosaserver2):

openssl req -newkey rsa:2048 -nodes -out /etc/ssl/certs/rosaserver2.csr -keyout /etc/ssl/certs/rosaserver2.key -subj "/CN=rosaserver2/"

openssl x509 -req -in /etc/ssl/certs/rosaserver2.csr -CA /etc/ssl/ca/caserver.pem -CAkey /etc/ssl/ca/caserver.key -out /etc/ssl/private/rosaserver2.pem

1. Сгенерированные ключи и сертификаты для сервера уже размещены в нужно месте; требуется скопировать ключи и сертификаты для клиентов.
2. На сервере выдать нужные права:

chmod 644 /etc/ssl/certs/rosaserver.key

chmod 644 /etc/ssl/private/rosaserver.pem

1. Отредактировать файл конфигурации /etc/systemd/journal-remote.conf к следующему виду:

[Remote]

Seal=false

SplitMode=host

ServerKeyFile=/etc/ssl/certs/rosaserver.key

ServerCertificateFile=/etc/ssl/private/rosaserver.pem

TrustedCertificateFile=/etc/ssl/ca/caserver.pem

1. Сохранить файл и выйти.
   * 1. Настройка клиента

Для настройки клиентов нужно выполнить следующие действия:

1. Прописать сервер в файле /etc/hosts на каждом из клиентов:

192.168.1.15 rosaserver

1. Создать директорию для сертификата:

sudo mkdir /etc/pki/tls/ca

sudo ln -s /etc/pki/tls/ca /etc/ssl/ca

Остальные директории и ссылки /etc/ssl/certs и /etc/ssl/private уже созданы.

1. Войти на сервер и скопировать ключи и сертификаты для первого клиента по SSH:

scp /etc/ssl/ca/caserver.pem root@192.168.1.111:/etc/ssl/ca/

scp /etc/ssl/private/rosaserver1.pem root@192.168.1.111:/etc/ssl/private/

scp /etc/ssl/certs/rosaserver1.key root@192.168.1.111:/etc/ssl/certs/

1. Скопировать ключи и сертификаты для второго клиента по ssh:

scp /etc/ssl/ca/caserver.pem root@192.168.1.232:/etc/ssl/ca/

scp /etc/ssl/private/rosaserver2.pem root@192.168.1.232:/etc/ssl/private/

scp /etc/ssl/certs/rosaserver2.key root@192.168.1.232:/etc/ssl/certs/

1. Предоставить нужные права на первом клиенте:

chmod 644 /etc/ssl/certs/rosaserver1.key

chmod 644 /etc/ssl/private/rosaserver1.pem

1. Предоставить нужные права на втором клиенте:

chmod 644 /etc/ssl/certs/rosaserver2.key

chmod 644 /etc/ssl/private/rosaserver2.pem

1. Привести файл конфигурации первого клиента к виду:

[Upload]

URL=https://rosaserver:19532

ServerKeyFile=/etc/ssl/certs/rosaserver1.key

ServerCertificateFile=/etc/ssl/private/rosaserver1.pem

TrustedCertificateFile=/etc/ssl/ca/caserver.pem

1. Привести файл конфигурации второго клиента к виду:

[Upload]

URL=https://rosaserver:19532

ServerKeyFile=/etc/ssl/certs/rosaserver2.key

ServerCertificateFile=/etc/ssl/private/rosaserver2.pem

TrustedCertificateFile=/etc/ssl/ca/caserver.pem

* + 1. Запуск логирования journald

Для запуска логирования нужно выполнить запуск:

1. На сервере:

sudo systemctl start systemd-journal-gateway.service

sudo systemctl start systemd-journal-remote.service

1. На всех клиентах:

sudo systemctl start systemd-journal-gateway.service

sudo systemctl start systemd-journal-upload.service

* + 1. Проверка работоспособности

Для чтения собственных логов самого сервера используется команда:

journalctl

Для чтения логов с удаленных клиентов используется команда:

sudo journalctl -D /var/log/journal/remote

Для проверки лога с удаленного сервера необходимо выполнить следующие шаги:

1. Войти на второй клиент, т.е. на rosaserver2, и выполнить команду:

logger -p syslog.debug "### TEST-2 MESSAGE from client rosaserver2 ###"

1. Войти на сервер и посмотреть, как данная запись с удаленного клиента rosaserver2 отразилась в журнале на сервере, с помощью команды:

sudo journalctl -D /var/log/journal/remote/ | grep TEST

Примерный результат приведен ниже (рисунок 34).

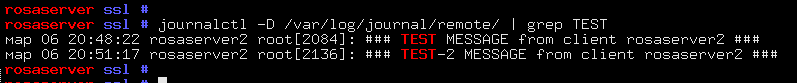


Рисунок 34 – Проверка отображения записи с сервера

Логи с удаленных серверов будут храниться в директории /var/log/journal/remote по имени хоста (рисунок 35):

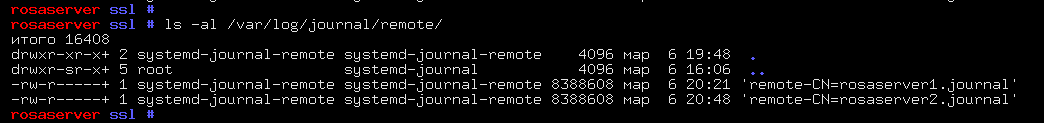


Рисунок 35 – Хранение логов удаленных серверов

* 1. Сервер печати CUPS

Сервер печати в ОС основан на системе CUPS (Common UNIX Printing System), которая является стандартным решением для управления печатью в UNIX-подобных системах. CUPS поддерживает различные принтеры и позволяет организовать печать как для локальных, так и для сетевых пользователей.

Настройка сервера печати может потребоваться в сетевой среде, где необходимо централизованное управление печатью для нескольких клиентов.

Сервер CUPS уже установлен в Системе, но если по каким-то причинам он отсутствует, его можно установить с помощью команды:

sudo dnf install cups

После установки необходимо запустить службу CUPS и добавить её в автозагрузку:

sudo systemctl start cups

sudo systemctl enable cups

Если доступ к серверу CUPS ограничен только локальной машиной, то для настройки доступа других клиентов к серверу печати, необходимо внести изменения в конфигурационный файл /etc/cups/cupsd.conf.

Для редактирования конфигурационного файла /etc/cups/cupsd.conf нужно:

1. открыть файл в текстовом редакторе:

sudo nano /etc/cups/cupsd.conf

1. найти и изменить параметры в блоках Listen и Allow для разрешения удаленного доступа:

Listen \*:631

<Location />

Order allow,deny

Allow all

</Location>

1. после изменений сохранить файл и перезапустить службу CUPS:

sudo systemctl restart cups

* + 1. Добавление и настройка принтеров

Для добавления принтера может использоваться веб-интерфейс CUPS, который доступен по адресу http://localhost:631. Этот интерфейс позволяет управлять принтерами, просматривать очередь печати и настраивать параметры сервера (рисунок 36).

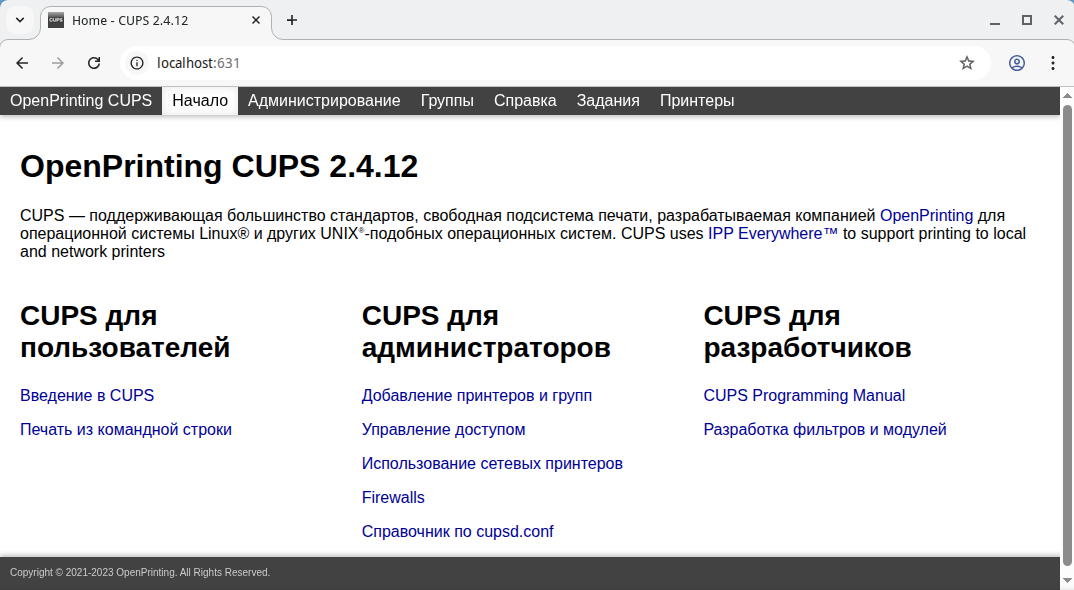


Рисунок 36 – Веб-интерфейс управления сервером CUPS

Для добавления принтера через веб-интерфейс CUPS необходимо:

1. Перейти в веб-интерфейс, перейти в раздел Администрирование (рисунок 37). Для входа необходимо ввести логин и пароль учетной записи Системы.

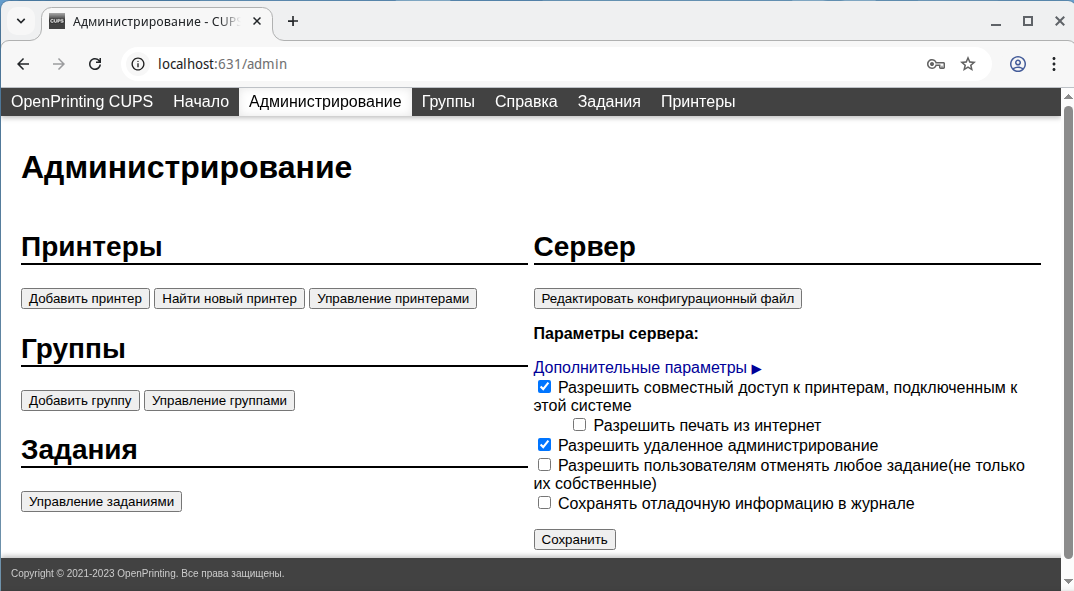


Рисунок 37 – Панель администрирования веб-интерфейса CUPS

1. Нажать "Добавить принтер" и следовать инструкциям мастера настройки.

В большинстве случаев CUPS автоматически определяет модель принтера и подбирает подходящий драйвер. Если ваш принтер не поддерживается автоматически, необходимо выбрать драйвер вручную или установить его из дополнительных пакетов.

Если принтер больше не используется, его можно удалить через веб-интерфейс или команду:

sudo lpadmin -x имя\_принтера

* + - 1. Настройка сетевого принтера (IPP)

Настройка сетевого принтера с использованием IPP требует выполнения ряда шагов через веб-интерфейс CUPS, а также некоторых изменений через командную строку, если это необходимо. Для настройки сетевого принтера необходимо:

1. Открыть веб-интерфейс CUPS по адресу http://localhost:631;
2. Перейти на вкладку Администрирование и выбрать опцию "Добавить принтер". Для доступа к разделу Администрирование необходимо ввести учетные данные администратора для подтверждения действий;
3. В списке доступных принтеров выберите "Internet Printing Protocol (IPP)" или "IPP Everywhere" (в зависимости от модели принтера);
4. Ввод данных о принтере:

* в поле "Подключение" следует ввести IP-адрес или URL принтера;
* Система попытается определить доступные параметры принтера автоматически. Также необходимо убедиться, что настройки IPP верны.

Выбор драйвера и настройка параметров:

* в процессе настройки CUPS предложит выбрать подходящий драйвер. В большинстве случаев он будет выбран автоматически. Если целевой принтер не был найден автоматически, необходимо выбрать драйвер из списка вручную.
* при необходимости настроить дополнительные параметры: формат бумаги, разрешение, цветность и другие.

Завершение настройки:

* после добавления принтера перейти в раздел "Принтеры" для проверки статуса устройства;
* печать пробной страницы для проверки правильности конфигурации.

Некоторые параметры также можно настроить напрямую через командную строку с использованием команды lpadmin:

sudo lpadmin -p office-printer -E -v ipp://192.168.1.100/printers/office-printer -m everywhere

где:

* -p указывает имя принтера,
* -v — URL-адрес подключения,
* -m — драйвер.

Для включения принтера и активации его в Системе:

sudo cupsenable office-printer

sudo cupsaccept office-printer

Для просмотра состояния задания печати и устранения неполадок можно использовать как веб-интерфейс, так и команды:

Просмотр очереди заданий:

lpstat -o

Отмена задания:

cancel job-id

Для диагностики проблем с печатью проверьте логи CUPS, расположенные в /var/log/cups/. В случае возникновения проблем с доступом к принтеру или драйверами, убедитесь, что все необходимые пакеты установлены и службы работают корректно.

systemctl status cups

Если возникают проблемы с подключением или печатью, рекомендуется проверить логи CUPS, которые хранятся в /var/log/cups/error\_log.

* + - 1. Управление доступом к принтеру

CUPS позволяет настраивать доступ к принтерам для отдельных пользователей или групп. Это можно сделать как через веб-интерфейс, так и в конфигурационных файлах.

Управление доступом и правами через веб-интерфейс:

1. В разделе Принтеры выбрать нужный принтер;
2. Перейдите в меню "Настроить разрешенных пользователей" для ограничения доступа к принтеру;
3. Здесь можно указать конкретных пользователей или группы, которые имеют право на печать.

Чтобы ограничить доступ к принтеру через консоль, добавьте в конфигурацию следующие строки:

<Location /printers/имя\_принтера>

Order deny,allow

Deny from all

Allow from 192.168.1.0/24

</Location>

где параметр Order deny,allow задает порядок обработки правил. Сначала применяются правила deny, а затем правила allow. Это значит, что сначала доступ запрещается для всех, а затем разрешается только для указанных источников.

После внесения изменений сохраните их и перезапустите CUPS:

sudo systemctl restart cups

* 1. Почтовый сервер OpenSMTPD

OpenSMTPD представляет собой почтовый сервер, развиваемый проектом OpenBSD и предназначенный для простой, безопасной и надежной замены таких решений, как Sendmail. Данный сервер обеспечивает минимальную конфигурацию при сохранении всех необходимых функций.

Для установки OpenSMTPD необходимо выполнить следующую команду:

sudo dnf install opensmtpd

Рекомендуется ознакомиться с составом пакета, включая список доступных руководств (man-страниц):

rpm -ql opensmtpd

Запуск службы осуществляется командой:

sudo systemctl start opensmtpd

Для проверки состояния службы используется:

sudo systemctl status opensmtpd

Журнал работы службы просматривается командой:

sudo journalctl -u opensmtpd

Для проверки используемых интерфейсов и портов следует выполнить:

sudo ss -ntulp | grep smtpd

По умолчанию служба прослушивает порт 25 на локальном интерфейсе (127.0.0.1).

При успешном запуске службы необходимо добавить её в автозапуск:

sudo systemctl enable opensmtpd

* + 1. Настройка OpenSMTPD

Для корректной работы почтового сервера следует задать его доменное имя, например:

sudo hostnamectl set-hostname mail.rosa.loc

В приведённом примере в качестве имени хоста используется mail.rosa.loc, что подразумевает обслуживание почтовых адресов вида user@rosa.loc (где rosa.loc — домен второго уровня).

Файл конфигурации OpenSMTPD расположен по адресу /etc/opensmtpd/smtpd.conf.

Для редактирования файла конфигурации нужно выполнить:

sudo nano /etc/opensmtpd/smtpd.conf

По умолчанию в конфигурации задано значение, которое определяет, что служба smtpd будет слушать только соединения, приходящие с локального компьютера:

listen on localhost

Данный параметр регламентирует:

* входящие подключения из внешней сети будут недоступны;
* почта может отправляться и приниматься только локально, например, между системными службами или пользователями на этом же сервере.

Если предполагается доступность почтового сервера из внешней сети, следует заменить эту опцию на следующую:

listen on 0.0.0.0

Для указания домена для обслуживания в файле конфигурации необходимо над строкой:

match for local action "local"

добавить строку, определяющую домен:

match from any for domain "rosa.loc" action "local"

где rosa.loc — это доменное имя, соответствующее ранее заданному имени хоста.

После внесения изменений в конфигурацию необходимо:

1. перезапустить службу командой:

sudo systemctl restart opensmtpd

1. убедиться в её успешном запуске:

sudo systemctl status opensmtpd

* + 1. Проверка работы OpenSMTPD

После настройки конфигурации почтового сервера необходимо убедиться в его работоспособности. По умолчанию OpenSMTPD осуществляет доставку писем в формате Maildir, где каждое сообщение хранится в отдельном файле в каталоге ~/Maildir домашней директории пользователя-получателя. Этот каталог требуется создать заранее.

Для осуществления проверки доставки почты выполняют следующие шаги:

1. Создание нового пользователя-получателя:

sudo useradd ivanov

1. Создание каталога для хранения почты:

sudo -u ivanov mkdir -p /home/ivanov/Maildir

1. Определение IP-адреса сервера командой:

ip a

Например, адрес сервера: 192.168.122.12.

1. Проверка доставки письма с другого ПК на сервер через службу Telnet:

telnet 192.168.122.12 25

Ожидаемый ответ сервера:

220 mail.rosa.loc ESMTP OpenSMTPD

Последовательное выполнение команд SMTP-протокола (с обязательным соблюдением синтаксиса):

helo dom.loc

Ожидаемый ответ:

250 mail.rosa.loc Hello dom.loc [192.168.122.1], pleased to meet you

Следующей командой указывается адрес отправителя письма (следует обратить внимание на необходимость строгого соблюдения RFC 2822 и указания адреса в знаках < >):

mail from:<vasya@domain.loc>

Ожидаемый ответ:

250 2.0.0 Ok

Указывается получатель письма:

rcpt to:<ivanov@rosa.loc>

Ожидаемый ответ:

250 2.1.5 Destination address valid: Recipient ok

Ввод:

data

В ответ приходит приглашение начать ввода письма. Необходимо ввести текст:

Message-ID: <x01@dom.loc>

Subject: Test

Privet

.

Ожидаемый ответ:

250 2.0.0 <message\_id> Message accepted for delivery

Проверка логов и доставки сообщения:

* 1. Просмотр системного журнала командой:

journalctl -u opensmtpd

* 1. Убедиться, что письмо сохранено в директории пользователя:

cat /home/ivanov/Maildir/new/\*

Пример содержимого доставленного сообщения:

Return-Path: <vasya@domain.loc>

Delivered-To: ivanov@rosa.loc

Received: from dom.loc (hp-xfce [192.168.122.1])

by mail.rosa.loc (OpenSMTPD) with SMTP id df9164e0

for <ivanov@rosa.loc>;

Sat, 10 Dec 2022 20:13:31 +0000 (UTC)

Message-ID: <x01@dom.loc>

Subject: Test

Privet

Почтовый сервер определяет домашний каталог пользователя через механизм PAM. Конфигурация PAM для службы OpenSMTPD расположена в файле:

/etc/pam.d/opensmtpd

Домашний каталог пользователя ivanov был получен через PAM. Конфигурация PAM для службы OpenSMTPD расположена в файле:

/etc/pam.d/opensmtpd

Это позволяет использовать как локальные, так и доменные (например, LDAP) учетные записи для получения почты.

* + 1. Интеграция OpenSMTPD с Dovecot

Для организации полноценного почтового сервера, обеспечивающего как приём сообщений (SMTP), так и их выдачу пользователям (IMAP), целесообразно использовать связку OpenSMTPD и Dovecot. В данной конфигурации OpenSMTPD будет принимать почту и сохранять её в формате Maildir, а Dovecot — предоставлять доступ к этой почте по IMAP.

Для установки Dovecot с необходимыми зависимостями следует выполнить команду:

sudo dnf install dovecot

Для настройки Dovecot необходимо выполнить следующие шаги:

1. Открыть и перейти в основной конфигурационный файл:

sudo nano /etc/dovecot/conf.d/10-mail.conf

1. В файле необходимо найти строку:

#mail\_location =

и заменить её на следующую:

mail\_location = maildir:~/Maildir

Внесенные опции укажут Dovecot на использование формата хранения Maildir в домашнем каталоге каждого пользователя.

1. После внесения изменений необходимо запустить и активировать службу Dovecot командой:

sudo systemctl enable --now dovecot.socket

После выполненных настроек почта будет работать следующим образом:

* OpenSMTPD принимает сообщения по протоколу SMTP и сохраняет их в директорию ~/Maildir пользователя;
* Dovecot предоставляет пользователю доступ к этим сообщениям по протоколу IMAP.

Таким образом можно подключаться к почтовому ящику как по SMTP (для исходящей почты), так и по IMAP (для входящей почты), например, используя удобный почтовый клиент.

По умолчанию как OpenSMTPD, так и Dovecot используют системных пользователей Linux и их пароли для аутентификации. Это обеспечивает интеграцию с PAM и возможность управления учетными записями через стандартные механизмы ОС.

* + 1. Дополнительные рекомендации по безопасности и интеграции

Для обеспечения безопасной работы почтового сервера OpenSMTP также рекомендуется выполнять следующие настройки Системы:

1. Настройка TLS/SSL. Для обеспечения защищённой передачи писем рекомендуется настроить TLS/SSL на OpenSMTPD и Dovecot.

В smtpd.conf можно активировать STARTTLS или слушать SMTPS (порт 465), задав путь к сертификатам и ключам:

listen on 0.0.0.0 port 587 tls pki mail.rosa.loc

где pki mail.rosa.loc — определение сертификата в конфигурации OpenSMTPD.

Можно сгенерировать самоподписанные сертификаты либо использовать сервис Let’s Encrypt (центр сертификации, предоставляющий бесплатные криптографические сертификаты X.509 для шифрования передаваемых через интернет данных HTTPS и других протоколов, используемых серверами в Интернете).

Для IMAP рекомендуется настроить Dovecot на порт 993 (IMAPS), в файле /etc/dovecot/conf.d/10-ssl.conf указав:

ssl = required

ssl\_cert = </etc/letsencrypt/live/mail.rosa.loc/fullchain.pem

ssl\_key = </etc/letsencrypt/live/mail.rosa.loc/privkey.pem

В результате пароли пользователей будут передаваться по защищённому каналу.

Для защиты пользователей почты рекомендуется интеграция со спам-фильтрами и антивирусными решениями. В качестве антиспам-решения может быть установлен rspamd или SpamAssassin. Для проверки вложений на наличие вирусов можно интегрировать ClamAV.

В цепочке обработки писем OpenSMTPD настраивается фильтр с помощью filter-директив, перенаправляющий почту через rspamd или amavis. Эти компоненты могут быть развернуты на том же сервере или вынесены на отдельные узлы.

Настройка SPF, DKIM и DMARC:

* SPF настраивается через публикацию DNS-записи типа TXT, в которой перечисляются IP-адреса отправителей, например:

v=spf1 ip4:192.168.122.12 -all

* DKIM предполагает установку пакета opendkim и генерацию пары ключей, после чего публичный ключ публикуется в DNS;
* DMARC задаётся также через TXT-запись в DNS и позволяет указать, как обрабатывать письма, не прошедшие SPF/DKIM.

Эти механизмы совместно снижают риск попадания писем в папки спама у получателей.

Настройка ротации логов. Журналы работы OpenSMTPD и Dovecot могут со временем занимать значительный объём, для их ротации используется система logrotate. В стандартных пакетах уже предусмотрены конфигурации в каталоге /etc/logrotate.d/. При необходимости можно настроить собственные параметры, например: частоту ротации, количество архивных копий и алгоритм сжатия. Регулярная ротация позволяет не переполнить файловую систему логами.

Рекомендуется регулярно выполнять резервное копирование:

* конфигурационных файлов (/etc/opensmtpd/, /etc/dovecot/);
* почтовых ящиков пользователей (~/Maildir);
* базы пользователей (если используется внешняя БД).

Для автоматизации можно применять rsync, borg, tar в связке с cron.

Настройка межсетевого экранирования (Nftables) для почтовых сервисов.

Для обеспечения доступности SMTP и IMAP/IMAPS из внешней сети необходимо настроить правила в системе сетевого экранирования Nftables, разрешив входящие соединения на соответствующие порты:

* 25 — SMTP (OpenSMTPD);
* 143 — IMAP (Dovecot);
* 993 — IMAPS (если используется шифрование).

Пример добавления правил дляOpenSMTP в Nftables:

1. Открыть файл конфигурации Nftables:

sudo nano /etc/nftables.conf

1. Добавить правила в таблицу inet (если такая уже создана) или в существующую filter:

table inet filter {

chain input {

type filter hook input priority 0;

# Разрешить входящие подключения к почтовым сервисам

tcp dport {25, 143, 993} accept

}

}

1. Применить изменения:

sudo nft flush ruleset

sudo nft -f /etc/nftables.conf

1. Проверить применённые правила:

sudo nft list ruleset

Также рекомендуется выполнить следующие меры для обеспечения безопасности подключений почтового сервера:

* ограничить доступ к портам 25/143/993 только с доверенных IP-адресов (например, только адреса организации или VPN);
* при необходимости использовать шифрование для SMTP (порт 465 или STARTTLS на 25) и IMAP (порт 993);
* убедиться, что правила Nftables применяются при каждом запуске Системы командой

systemctl enable nftables

1. Удаленный доступ к серверу

Часть из рассматриваемых далее способов удаленного подключения к Системе работает только в случае, если установлен GUI-интерфейс.

* 1. Доступ по OpenSSH
     1. Общие сведения об OpenSSH

OpenSSH – это бесплатный SSH-сервер, дающий возможность интерактивного управления сервером.

В большинстве дистрибутивов OpenSSH-сервер уже присутствует в ОС, и его установка не требуется. В случае отсутствия OpenSSH следующая команда выполнит установку необходимых пакетов:

sudo dnf openssh

Далее следует добавить SSH-сервер в автозагрузку. При следующем запуске сервера ОС выполнит автоматический запуск SSH-сервера. Как и в случае с другими сервисами, systemd позволяет управлять параметрами запуска, автозагрузки и рестарта демона OpenSSH. Автозапуск включается командой:

sudo systemctl enable ssh

Работоспособность утилиты проверяется командой:

ssh localhost

* + 1. Настройка SSH

Настройка SSH необходима для улучшения защищенности Системы. Например, можно отключить возможность входа от имени пользователя root или изменить порт подключения со стандартного 22 на произвольный. Лучше использовать порты из верхнего диапазона (50000-65000).

Примечание – В стеке протоколов TCP/IP доступно 65536 портов.

Настройка выполняется в конфигурационном файле. Перед его модификацией создают резервную копию.

# sudo cp /etc/ssh/sshd\_config

/etc/ssh/sshd\_config.factory-defaults

Все изменения конфигурации SSH проводятся в файле /etc/ssh/sshd\_config. Для его редактирования нужно выполнить команду:

sudo nano /etc/ssh/sshd\_config

В файле нужно раскомментировать строку "Port 22" и изменить значение на 55555 (рисунок 38).

Следует учитывать, что автоматизированные системы, такие как боты, зачастую приоритетно сканируют порты с одинаковыми цифрами. Поэтому в производственных и корпоративных сетях рекомендуется применять порты с различными цифрами для повышения уровня безопасности.

йИзображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 38 – Изменение стандартного порта подключения

Далее нужно отключить возможность входа на сервер учетной записи суперпользователя (root) и добавить возможность входа через ключи. Для этого необходимо изменить значения параметров PermitRootLogin на no и PubkeyAuthentication на yes (рисунок 39).

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Рисунок 39 – Изменение параметров PermitRootLogin и PubkeyAuthentication

После этого следует перезагрузить демон SSH. Соединение при этом будет разорвано и подключиться можно будет через новый порт и пользовательскую учетную запись (она должны быть предварительно создана):

sudo systemctl restart ssh

Далее нужно переподключиться от обычной учетной записи и по другому порту:

ssh -p 55555 ansible@192.168.0.104

После успешного подключения можно продолжать работу с сервером.

* + 1. Пары ключей

Еще один способ аутентификации на сервере – пара ключей RSA: открытый и закрытый. Открытый хранится на сервере, к которому будет выполняться подключение, а закрытый – на удаленном компьютере (или другом сервере), с которого выполняется подключение.

Далее (рисунок 40) в качестве примера приведена схема безопасного обмена ключами между "Отправителем" и "Получателем". "Злоумышленник" может читать сообщения, если они не зашифрованы. Здесь "Отправитель" или "Получатель" шифруют сообщение при помощи открытого ключа принимающей стороны, которая его дешифрует при помощи своего закрытого ключа.

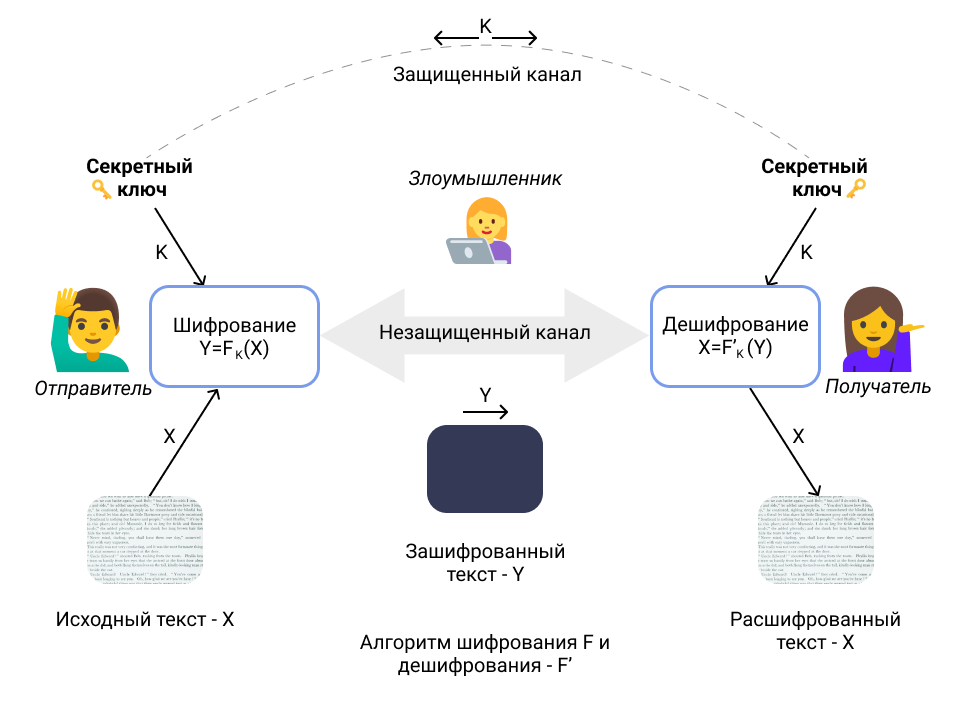


Рисунок 40 – Безопасный обмен ключами

Чтобы сгенерировать такую пару ключей, достаточно выполнить команду:

ssh-keygen -t rsa

Команду нужно выполнять на рабочей станции от имени пользователя, который будет в дальнейшем подключаться к удаленному компьютеру. Путь к хранению ключей можно оставить по умолчанию.

После создания ключей можно переходить к следующему шагу – копированию открытого ключа на удаленный сервер, выполнив следующую команду:

ssh-copy-id -i ~/.ssh/id\_rsa.pub rosa@192.168.0.104

В этом примере 192.168.0.104 – это IP-адрес удаленного сервера. После ввода пароля ключ копируется в папку .ssh домашней директории пользователя.

Сразу же после выполнения копирования следует проверить доступ при помощи созданной пары ключей:

ssh rosa@192.168.0.104

Для отключения возможности входа по паролю необходимо в файле /etc/ssh/sshd\_config отредактировать значение PasswordAuthentication и присвоить no (рисунок 41).

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Рисунок 41 – Редактирование параметра PasswordAuthentication

После изменения настроек нужно перезагрузить службу SSH:

sudo systemctl restart ssh

После проведенных настроек при попытке подключения пользователем, для которого не определена пара ключей, будет выдаваться ошибка подключения.

При этом подключение при помощи ключа будет успешным.

Отключение доступа по паролю значительно повышает безопасность сервера.

* 1. Терминальный доступ с использованием XRDP

Терминальный сервер XRDP предназначен для организации удалённого доступа пользователей к графическому сеансу рабочего стола через протокол RDP. После подключения пользователь проходит аутентификацию и получает полноценную рабочую среду, аналогичную локальному сеансу.

Для установки терминального сервера XRDP необходимо выполнить установку следующих пакетов с помощью команды:

sudo dnf install xrdp task-x11

После установки требуется включить и запустить службу xrdp командой:

sudo systemctl enable --now xrdp

Для корректной работы терминального сеанса необходимо работать в одном из доступных графических окружений:

* **Минимальное KDE Plasma 5; устанавливается командой**:

sudo dnf install task-plasma5-minimal

* **Полное KDE Plasma 5 с базовым набором программ; устанавливается командой**:

sudo dnf install task-plasma5

* **KDE Plasma 5, соответствующее десктопному образу ОС; устанавливается командой:**

sudo dnf install task-iso-plasma5

* **Минимальное XFCE; устанавливается командой**:

sudo dnf install task-xfce-minimal

* **XFCE с настройками РОСА и дополнительным ПО; устанавливается командой:**

sudo dnf install task-xfce

* **Окружение MATE; устанавливается командой**:

sudo dnf install task-mate

* **GNOME, соответствующее десктопному образу ОС; устанавливается командой**:

sudo dnf install task-iso-gnome

* + 1. Настройка локальных пользователей

Для предоставления доступа по RDP локальным пользователям необходимо:

1. создать локальную учётную запись:

sudo useradd <имя\_пользователя>

1. задать пароль пользователю:

passwd <имя\_пользователя>

1. добавить пользователя в группу tsusers, обладающую правом на запуск XRDP-сессий:

usermod -a -G tsusers <имя\_пользователя>

* + 1. Настройка доменных пользователей

Для обеспечения подключения доменных пользователей по протоколу RDP через XRDP необходимо настроить сопоставление доменных групп с локальной группой tsusers, используемой XRDP для контроля доступа:

1. Необходимо убедиться, что сервер введён в домен и Система успешно распознает доменные учетные записи. Проверить наличие информации о домене можно командой:

realm list

Если сервер введён в домен, в ответ будет показан список обнаруженных доменов.

Подробнее о вводе в домен см. раздел 12.3.5.

1. На контроллере домена добавить пользователей, которым требуется доступ по XRDP, в одну общую доменную группу, например RDP-Users.
2. На сервере с ОС РОСА "ХРОМ" выполнить команду для сопоставления этой доменной группы с локальной группой tsusers:

sudo roleadd "Доменная\_группа" tsusers

Например:

sudo roleadd "DOMAIN\\RDP-Users" tsusers

После выполнения этой команды все члены доменной группы RDP-Users будут автоматически считаться членами локальной группы tsusers и смогут подключаться к серверу через XRDP без необходимости вручную добавлять каждого пользователя в локальные группы.

1. Проверить корректность настроек командой:

id 'доменный\_пользователь'

Вывод команды должен содержать группу tsusers.

* + 1. Подключение с клиента

Для подключения к терминальному серверу можно использовать любой RDP-клиент, поддерживающий стандартный протокол. В качестве примера — клиент Remmina, доступный в официальном репозитории РОСА. Для установки Remmina в терминале нужно выполнить следующие действия:

1. обновить список пакетов:

sudo dnf makecache

1. установить пакет Remmina от имени пользователя root:

sudo dnf install remmina

1. после завершения установки запустить Remmina из меню приложений (категория Интернет) или выполнить в терминале:

remmina

Клиентское подключение в Remmina производится по IP-адресу терминального сервера с вводом имени пользователя и пароля, соответствующих разрешённой локальной или доменной учётной записи (рисунок 42).

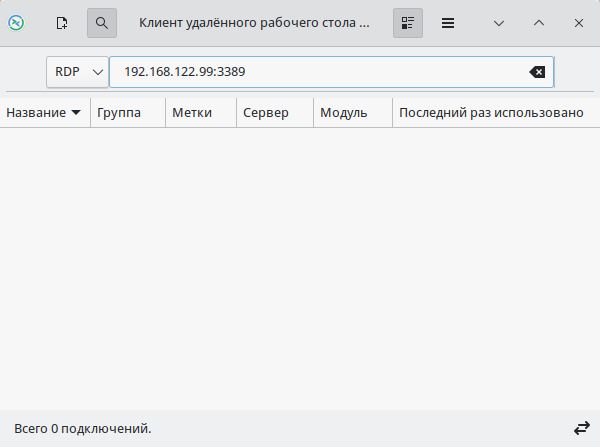


Рисунок 42 – Подключение по RDP с помощью клиента Remmina

1. Конфигурация системы для защиты информации
   1. Общее описание

Для использования ОС в задачах, требующих повышенной защиты информации, применяются более строгие настройки файлов конфигурации Системы и более подробные настройки журналирования событий. ОС имеет возможность использования таких настроек установкой пакетов sconfigs, при этом будут установлены пакеты, добавляющие в ОС настройки, обеспечивающие повышенную защиту информации и которые можно выбрать в зависимости от текущего профиля защиты.

ОС может быть сконфигурирована для безопасной работы используя настройки, описанные в пп.14.2–14.9.

* 1. Настройки службы SSH

Безопасные настройки службы удаленного доступа SSH – sshd –обеспечиваются установкой пакета sconfigs-sshd**.**

При этом устанавливаются ключи настроек в файле /etc/ssh/sshd\_config.d/sshd\_hardening.conf.

Все настройки в файле подробно документированы и при необходимости могут быть изменены опытным администратором. Более подробно настройка удаленного доступа к Системе через SSH описана в разделе 13.

* 1. Отключение файловых систем без атрибутов безопасности

Для безопасности рекомендуется отключать возможность загрузки редко используемых ФС, не поддерживающих атрибуты безопасности, с помощью команды modprobe.

Эти настройки могут быть выполнены с помощью установки пакета sconfigs-disable-filesystems-modulesи связанных с ним служебных пакетов.

Документированный список отключаемых ФС настраивается в файле /etc/modprobe.d/restrict\_fs\_modules.conf.

* 1. Отключение редких сетевых протоколов

Отключение в целях безопасности неиспользуемых сетевых протоколов может быть реализовано с помощью установки пакета sconfigs-disable-network-modules с зависимыми пакетами. После его установки в Систему добавляется файл настроек //etc/modprobe.d/restrict\_net\_modules.conf с документированными настройками отключения таких протоколов.

* 1. Отключение дампов оперативной памяти

ОС по умолчанию сохраняет дампы оперативной памяти (coredumps) при сбоях Системы, что позволяет искать и исправлять ошибки. Но такая настройка снижает безопасность Системы, т.к. в дампах могут содержаться открытые пароли и другие важные данные. Рекомендуется отключать возможность создания дампов памяти путем установки пакета sconfigs-disable-coredumps.

При установке пакета в файлах конфигурации //etc/security/limits.d/90-disable-coredumps.conf и //etc/sysctl.d/96-disable-coredumps.conf будут установлены необходимые для этого отключения настройки.

* 1. Настройка переменных ядра

Для повышения безопасности функционирования ядра используется множество настроек (например, отключение системных вызовов по клавише SysRq). Установить всю совокупность этих настроек можно установкой пакета sconfigs-harden-kernel. Эти настройки перечислены и документированы в файле /etc/sysctl.d.95-kernel-hardening.conf.

* 1. Настройки аудита

Необходимым условием защиты информации в ОС является настройка и функционирование службы аудита. В репозиториях предлагается два шаблона настроек аудита – сокращенный sconfigs-basic-audit и полный sconfigs-detailed-audit. Эти пакеты содержат, соответственно, файлы настроек аудита /etc/sconfigs/audit-rules/20-basic-audit.rules и /etc/sconfigs/audit-rules/30-detailed-audit.rules, в которых подробно документированы настройки службы аудита и установлены их рекомендованные значения.

* 1. Отключение настроек блокировки экрана

По умолчанию пользователь в KDE Plasma может отключить блокировку экрана по наступлении периода бездействия или при выходе из ждущего режима, что позволяет неавторизованному пользователю получить доступ к Системе. Файл /etc/xdg/kscreenlockerrc из пакета sconfigs-kscreenlocker блокирует такое поведение, делая настройки по умолчанию недоступными для изменения.

* 1. Настройка автоблокировки консольного режима

Для работы в консольном режиме tty применяется программа tmux, сконфигурированная (файл /etc/tmux.conf) следующим образом:

set -g lock-command "vlock -c"

set -g lock-after-time 300

bind L lock-session

Таким образом включается автоблокировка консоли по истечении 5 минут бездействия командой vlock.

Замена стандартной консоли tty на tmux осуществляется в файле конфигурации /etc/bashrc скриптом:

if [ -n "$PS1" ] &&   
 [ "$XDG\_SESSION\_TYPE" = tty ] &&   
 [ -z "$TMUX" ] &&   
 ! [[ "$TERM" =~ screen ]] &&   
 ! [[ "$TERM" =~ tmux ]] &&   
 command -v tmux >/dev/null 2>&1

then   
 exec tmux

fi

1. Системы управления базами данных

Система управления базами данных (далее – СУБД) – это комплекс программно-языковых средств, позволяющих создавать базы данных и управлять данными.

* 1. Доступные СУБД

В ОС доступны для использования три СУБД:

* MySQL – реляционная клиент-серверная СУБД; для проектов небольшого и среднего размера; легкая, гибкая и довольно простая в использовании; бесплатная; хорошо подходит для обучения и веб-проектов;
* SQLITE – небольшая и легкая встраиваемая СУБД, которая обычно применяется в локальных проектах;
* PostgreSQL – объектно-реляционная СУБД клиент-серверного типа, которую иногда называют бесплатным аналогом Oracle; масштабируемая; для высоконагруженных проектов; содержит огромное количество функций и распространяется бесплатно.
  1. PostgreSQL

Свободная объектно-реляционная PostgresSQL присутствует в репозиториях ОС. Для установки используют следующую команду:

sudo dnf install postgresql-server

PostgreSQL использует концепцию ролей для выполнения аутентификации и авторизации клиента. В некоторых аспектах они напоминают обычные учетные записи в ОС, но PostgresSQL не делает различий между пользователями и группами и предпочитает использовать более гибкий термин – роль.

После установки PostgresSQL настроена на использование аутентификации ident, это означает выполнение привязки ролей PostgresSQL к системной учетной записи ОС – postgres. Если роль существует внутри PostgresSQL, пользователь ОС с тем же именем может выполнить вход в качестве этой роли.

Чтобы использовать PostgreSQL, можно войти в эту учетную запись. Но сначала нужно установить для нее свой пароль:

passwd postgres

Чтобы PostgresSQL запускалась с загрузкой ОС, нужно включить сервис:

sudo systemctl enable postgresql12.service

Чтобы запустить PostgreSQL, необходимо запустить соответствующий сервис:

sudo systemctl start postgresql12.service

Для проверки факта запуска PostgreSQL и его версии выполняют команду:

sudo -u postgres psql -c "SELECT version();"

В результате запуска можно получить доступ к командной строке PostgreSQL, а затем свободно взаимодействовать с СУБД с помощью команды:

sudo -u postgres psql

Например, с помощью команд можно:

* Создать роль testuser:

postgres=# create user testuser;

* Создать пароль для роли testuser:

postgres=# \password testuser

* Создать базу данных testdb:

postgres=# create database testdb;

* Предоставить разрешения пользователю testuser в базе данных testdb:

postgres=# grant connect, create on database testdb to testuser;

postgres=# grant all privileges on database testdb to testuser;

* Просмотреть базы данных:

postgres=# \l

* Просмотреть роли:

postgres=# \du

* Выйти из консоли psql:

postgres=# \q

* 1. MySQL

MySQL — это одна из самых популярных систем управления реляционными базами данных, основанная на языке запросов SQL (Structured Query Language). MySQL отличается высокой производительностью, надежностью и широким применением в серверных ОС. Она поддерживает модель клиент-сервер, что позволяет эффективно управлять данными в многопользовательских системах.

Основное преимущество MySQL заключается в поддержке многопоточности, позволяющей работать с большими объемами данных и обрабатывать несколько запросов одновременно. MySQL используется для веб-сайтов, облачных платформ, а также для крупных корпоративных приложений благодаря совместимости с различными ОС и интеграции с популярными веб-серверами, такими как Apache.

Установка СУБД MySQL описана в п.12.4.1.3.

* 1. SQLITE

SQLite — это легковесная встраиваемая система управления базами данных, которая не требует отдельного сервера для работы и хранит все данные в одном файле. Она широко применяется для локальных баз данных или приложений, которые не нуждаются в сложной многопользовательской архитектуре.

SQLite работает с минимальными ресурсами и подходит для приложений с низкой нагрузкой на БД, таких как десктопные программы, мобильные приложения, а также для разработки и тестирования. SQLite является ACID-совместимой (атомарность, согласованность, изолированность, долговечность), поддерживает основные SQL-запросы и транзакции. Однако, в отличие от MySQL или PostgreSQL, она не предназначена для масштабных многопользовательских систем, так как не поддерживает полноценную многопоточность и функции клиент-серверной архитектуры.

Для установки SQLite в ОС используйте команду:

sudo dnf install sqlite

Для получения справки по командам и функциям, связанным с SQLite, можно использовать встроенную в утилиту команду:

sqlite3 --help

Также можно обратиться к официальной документации SQLite или прочитать справку в формате man-страницы:

man sqlite3

Эти команды предоставят подробное руководство по использованию SQLite, включая основные команды для работы с базами данных.

1. Автоматизация инфраструктуры с Ansible

Ansible — это инструмент для автоматизации управления конфигурациями, развёртывания программ и выполнения сложных задач. Он помогает Администраторам управлять большим количеством серверов и хостов централизованно, обеспечивая простоту и эффективность процессов. Ansible интегрируется в архитектуру ОС и поддерживает автоматизацию множества операций.

Ansible использует архитектуру без агента, работая по SSH-протоколу, что упрощает настройку и использование. Все операции описываются в виде "плейбуков" на языке YAML, что делает управление конфигурациями прозрачным и легко читаемым.

Основные компоненты Ansible:

* Модули — это основные рабочие единицы Ansible, которые выполняют определённые задачи, такие как управление пакетами, работой с файлами и сетевыми интерфейсами.
* Инвентарь (Inventory) - список узлов (хостов), к которым Ansible будет обращаться для выполнения задач. Инвентарь может быть статическим (описан в текстовом файле) или динамическим (автоматически генерируется).
* Плейбуки (Playbooks) - наборы инструкций, написанных на YAML, определяющие, какие задачи нужно выполнить на целевых хостах.
* Роли (Roles) - механизм, который группирует плейбуки, переменные и задачи для более эффективного и модульного управления инфраструктурой.
  1. Установка и подключение хостов

Для установки Ansible нужно выполнить команду от имени root:

sudo dnf install ansible

Для подключения хостов к Системе автоматизации следует выполнить следующие шаги:

1. В файле /etc/ansible/hosts прописать все хосты, на которые будет распространяться конфигурация. Хосты можно разделить по группам, а также, если есть домен, автоматически экспортировать список из домена. Можно прописывать как IP-адреса, так и имена хостов, если они видны DNS-сервером в сети. Для тестового примера можно прописать два хоста:

[test]

10.10.1.111

10.10.1.74

1. Подключение к хостам осуществляется по протоколу SSH с помощью RSA-ключей; сгенерировать серверный ключ можно командой (в процессе ее выполнения следует на все запросы нажимать клавишу Enter):

ssh-keygen -C "$(whoami)@$(hostname)-$(date -I)"

1. Распространить ключ на все подключенные хосты командой:

ssh-copy-id root@server

где:

* root – пользователь, от имени которого будут выполняться плейбуки;
* server – IP-адрес хоста.

Пример: root@10.10.1.74.

* 1. Проверка подключения

Для проверки подключения удаленных хостов с помощью Ansible нужно выполнить команду:

ansible test -m ping

где test – это группа хостов, указанная в файле hosts.

В выводе команды для каждого хоста должен присутствовать успешный результат:

test-1 | success >> {

"changed": false, U

"ping": "pong"

}

test-2 | success >> {

"changed": false,

"ping": "pong"

}

* 1. Описание сценариев Ansible

Для работы с Ansible на клиенте достаточно SSH-соединения. Задача Ansible – выполнить одновременно ряд операций на нескольких удаленных рабочих станциях. Все операции, которые выполняются последовательно, записываются в некий Ansible-скрипт - плейбук, основанный на собственном наборе команд. Записи ведутся в простом формате и построены на YAML-разметке. Пример записи сценария:

- name: "First step"

hosts: localhost

tasks:

- taskA

- taskB

где:

* name: "First step" – имя группы задач;
* hosts: localhost – хосты, на которых будут выполнены задачи;
* параметры, перечисленные после команды tasks – перечень задач, которые необходимо выполнить.

В одном плейбуке может быть несколько блоков с задачами, для чего необходимо добавить дополнительные параметры name после завершения описания предыдущей группы задач.

Команды программы – tasks – являются модулями Ansible. Таким образом, задавая параметры task, вызывается модуль Ansible с его параметрами. Записать task можно в двух видах: кратком и полном.

Пример команды установки curl при помощи dnf-модуля в кратком виде:

-dnf: name=curl state=latest

Тогда плейбук с этим task будет выглядеть следующим образом:

- name: "First step"

hosts: localhost

tasks:

- dnf: name=curl state=latest

Возможность более полно описать команду task – присвоить ей имя (name). Тогда получившийся текст выглядит как подзадача основного блока задач:

- name: "First step"

hosts: localhost

tasks:

- name: "Add curl package"

dnf: name=curl state=latest

В этом случае сразу видно, что есть блок задач "First step", в нем существует одна задача – установка пакета curl.

* 1. Создание сценариев

Плейбуки являются сценариями, выполняемыми на удаленных хостах. В качестве примера можно создать плейбук, устанавливающий программы, выполнением следующих действий:

1. Создать каталог для хранения плейбуков, которые пишутся на языке YAML:

sudo mkdir /etc/ansible/playbooks

1. В каталоге /etc/ansible/playbooks создать файл install\_programm.yml:

sudo touch /etc/ansible/playbooks/install\_programm.yml

со следующим содержимым:

---

- hosts: all

tasks:

- name: Install programm

dnf:

name: "{{ packages }}"

vars:

packages:

- vim

- zstd

В данном примере для установки на всех хостах используется модуль dnf и массив packages, который содержит подлежащие установке программы.

Для определения хостов, на которых будет проведена установка, можно воспользоваться командой:

ansible-playbook /etc/ansible/playbooks/install\_programm.yml --list-host

Созданный набор инструкций можно запустить командой:

ansible-playbook /etc/ansible/playbooks/install\_programm.yml

* 1. Выполнение одиночных команд

С помощью Ansible возможно сразу на всех клиентах выполнить команду bash без создания плейбуков. Например, чтобы просмотреть информацию об использовании оперативной памяти на удаленных хостах, нужно выполнить команду:

ansible test -a "free -h"

1. Контейнеризация

Контейнеризация – это метод виртуализации на уровне ОС, который позволяет запускать приложения и их зависимости в изолированных контейнерах. Контейнеры включают в себя все необходимые для работы приложения компоненты, такие как библиотеки, зависимости и настройки, но используют ядро ОС хоста, что делает их легковесными по сравнению с виртуальными машинами.

Контейнеризация является отличной альтернативой аппаратной виртуализации. Преимущества использования контейнеров:

* Изоляция окружения: контейнеры гарантируют, что приложения не будут конфликтовать друг с другом по зависимостям и настройкам;
* Упрощение развертывания: контейнеры можно развертывать одинаково в разных средах, что снижает риск ошибок, связанных с различиями в окружениях;
* Использование контейнеров позволяет легко наращивать или сокращать количество работающих экземпляров приложения;
* Контейнеры используют общие ресурсы ОС, что снижает нагрузку на оборудование.

В Системе используются такие популярные технологии контейнеризации, как Docker и Kubernetes. Далее рассмотрим основные принципы работы с данным ПО.

* 1. Docker

Одним из наиболее популярных инструментов для программной виртуализации является программа Docker – автоматизированное средство управления виртуальными контейнерами. Он решает множество задач, связанных с созданием контейнеров, размещением в них приложений, управлением процессами, а также тестированием ПО и его отдельных компонентов.

Docker – программное обеспечение с открытым исходным кодом, применяемое для разработки, тестирования, доставки и запуска веб-приложений в средах с поддержкой контейнеризации. Он нужен для более эффективного использования Системы и ресурсов, быстрого развертывания готовых программных продуктов, а также для их масштабирования и переноса в другие среды с гарантированным сохранением стабильной работы.

Docker-контейнеры работают в разных средах: локальном центре обработки информации, облаке, персональных компьютерах и т. д.

Docker представляет собой набор нескольких основных компонентов:

* Docker-демон (Docker-daemon) – сервер контейнеров, входящий в состав программных средств Docker. Демон управляет Docker-объектами (сети, хранилища, образы и контейнеры). Демон также может связываться с другими демонами для управления сервисами Docker.
* Docker-клиент (Docker-client / CLI) – интерфейс взаимодействия пользователя с Docker-демоном. Клиент и Демон – важнейшие компоненты "движка" Докера (Docker Engine). Клиент Docker может взаимодействовать с несколькими демонами.
* Docker-образ (Docker-image) – файл, включающий зависимости, сведения, конфигурацию для дальнейшего развертывания и инициализации контейнера.
* Docker-файл (Docker-file) – описание правил по сборке образа, в котором первая строка указывает на базовый образ. Последующие команды выполняют копирование файлов и установку программ для создания определенной среды для разработки.
* Docker-контейнер (Docker-container) – это легкий, автономный исполняемый пакет программного обеспечения, который включает все необходимое для запуска приложения: код, среду выполнения, системные инструменты, системные библиотеки и настройки.
* Том (Volume) – эмуляция ФС для осуществления операций чтения и записи. Она создается автоматически с контейнером, поскольку некоторые приложения осуществляют сохранение данных.
* Реестр (Docker-registry) – зарезервированный сервер, используемый для хранения docker-образов. Примеры реестров:
* Центр Docker – реестр, используемый для загрузки docker-image. Он обеспечивает их размещение и интеграцию с GitHub и Bitbucket.
* Контейнеры Azure – предназначен для работы с образами и их компонентами в директории Azure (Azure Active Directory).
* Доверенный реестр Docker или DTR – служба docker-реестра для инсталляции на локальном компьютере или сети компании.
* Docker-хаб (Docker-hub) или хранилище данных – репозиторий, предназначенный для хранения образов с различным ПО. Наличие готовых элементов влияет на скорость разработки.
* Docker-хост (Docker-host) – машинная среда для запуска контейнеров с программным обеспечением.
* Docker-сети (Docker-networks) – применяются для организации сетевого интерфейса между приложениями, развернутыми в контейнерах.
* Docker Engine ("движок" Docker) – ядро механизма Docker. "Движок" отвечает за функционирование и обеспечение связи между основными Docker-объектами (реестром, образами и контейнерами).

Сервер выполняет инициализацию демона (фоновой программы), который применяется для управления и модификации контейнеров, образов и томов.

REST API – механизм, отвечающий за организацию взаимодействия Docker-клиента и Docker -демона.

Клиент – позволяет пользователю взаимодействовать с сервером при помощи команд, набираемых в интерфейсе (CLI).

Установка и запуск сервиса docker:

sudo dnf install docker docker-containerd docker-compose -y

sudo systemctl start docker

sudo systemctl enable docker

* 1. Kubernetes
     1. Общие сведения

Kubernetes – это система с открытым исходным кодом для развертывания, масштабирования и управления контейнезированными приложениями.

Kubernetes по сути является не просто системой оркестрации. Технически оркестрация это про выполнение определенного рабочего процесса: сначала сделай A, затем B, затем C. Kubernetes же устраняет прямую необходимость в этом. В нем есть процессы управления, по факту независимые и компонуемые. Главная задача процессов управления – перевести текущее состояние к нужному состоянию. В этом случае неважно какой будет маршрут от А до С, что исключает централизованный контроль. Благодаря этому Kubernetes проще в использовании, мощная, надежная, а также устойчивая и расширяемая.

Контейнеры позволяют поделить приложения на более мелкие части с четким разделением задач. Уровень абстракции, предоставляемый для отдельного образа контейнера, позволяет понять, как строятся распределенные приложения. Такой модульный подход дает возможность быстрее осуществлять разработку с помощью небольших и более целенаправленных групп, каждая из которых отвечает за определенные контейнеры. Это также позволяет изолировать зависимости и шире использовать компоненты меньшего размера. Сделать это только с помощью контейнеров не получится. А вот в Kubernetes это можно достичь с помощью Pods (подов). Pod (под) – это группа из одного или более контейнеров с общим хранилищем/сетевыми ресурсами и спецификацией запуска контейнеров или отдельный инстанс (экземпляр объекта) приложения. Размещая контейнеры таким образом, Kubernetes устраняет соблазн втиснуть слишком много функций в один образ контейнера. Концепция Service (Сервисы) в Kubernetes используется для группирования нескольких подов, которые выполняют те же функции. Сервисы легко настраиваются для таких целей, как обнаружение, горизонтальное масштабирование и балансировка нагрузки. Kubernetes согласно официальной документации так же сможет предоставить:

* Используя имя DNS или собственный IP-адрес мониторинг сервисов и распределение нагрузки, Kubernetes может обнаружить контейнер. При высоком трафике в нем Kubernetes сбалансирует нагрузку и распределит сетевой трафик так, что развертывание будет стабильным.
* Система хранения по выбору (например, локальное хранилище, провайдеры общедоступного облака и многое другое) может быть автоматически смонтирована с помощью оркестрации хранилища Kubernetes.
* Автоматическое развертывание и откаты. Kubernetes через описание желаемого состояния развернутых контейнеров (манифесты, которые пишутся на YAML) может изменить фактическое состояние на желаемое. То есть создание новых контейнеров для развертывания, удаления существующих контейнеров и распределения всех их ресурсов в новый контейнер в Kubernetes можно автоматизировать.
* Автоматическое распределение нагрузки. Kubernetes сам размещает контейнеры на узлах так, чтобы наиболее эффективно использовать ресурсы. Администратору остается только указать сколько ЦП и ОЗУ требуется каждому контейнеру и предоставить кластер узлов, на котором будут запущены контейнеры.
* Самоконтроль. Если в работе контейнеров что-то пошло не так, то Kubernetes сам перезапускает, заменяет и завершает работу контейнеров, которые не проходят проверку работоспособности.
* Управление конфиденциальной информацией и конфигурацией. Пароли, OAuth-токены и ключи SSH могут храниться и управляться Kubernetes без изменений образов контейнеров и без раскрытия конфиденциальной информации в конфигурации стека.
  + 1. Установка kubernates master-node

Установка kubernates master-node проводится следующим образом:

1. Установка всех зависимостей kubernates:

sudo dnf install -y kubelet kubeadm kubectl docker

1. Запуск и добавление в автозагрузку сервисов:

sudo systemctl enable docker.service

sudo systemctl enable containerd.service

sudo systemctl start docker.service

sudo systemctl start containerd.service

sudo systemctl enable kubelet

sudo systemctl start kubelet

1. Добавление узлов в hosts (master, worker):

sudo hostnamectl set-hostname master-node

vi /etc/hosts

10.45.4.58 master-node

10.45.4.59 worker-node

1. Отключение swap:

sudo sed -i '/swap/d' /etc/fstab

sudo swapoff -a

1. Инициализация kubernates:

kubeadm init

1. Выдача установщиком токена подключения к master node:

kubeadm join 10.45.4.58:6443 --token ncb63c.pugf9xehkjakptqp --discovery-token-ca-cert-hash sha256:73cb40755a49b40f2a724984392f32fd7ff607a437da524619d4aba43a2c31b6

1. Создание каталогов, необходимых для kubernates:

mkdir -p $HOME/.kube

cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

chown $(id -u):$(id -g) $HOME/.kube/config

export kubever=$(kubectl version | base64 | tr -d '\n'

1. Добавление сети для работоспособности kubernates:

kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-version=$kubever"

1. Перезагрузка kubernates:

sudo systemctl restart kubelet

* + 1. Установка kubernates worker-nodes (pods-контейнеры)

Установка kubernates worker-nodes выполняется следующими шагами:

1. Установка docker:

sudo dnf install docker

1. Запуск и добавление в автозагрузку сервисов docker:

sudo systemctl start docker

sudo systemctl enable docker

1. Установка всех зависимостей kubernates:

sudo dnf install -y kubelet kubeadm kubectl

1. Добавление автозагрузку kubernates:

sudo systemctl enable kubelet

1. Изменение имени виртуального узла:

sudo hostnamectl set-hostname worker-node

1. Добавление узлов в hosts:

vi /etc/hosts

10.45.4.58 master-node

10.45.4.59 worker-node

1. Отключение swap:

sudo sed -i '/swap/d' /etc/fstab

sudo swapoff -a

1. Присоединение worker-node к master (токен – из предыдущего этапа):

kubeadm join 10.45.4.58:6443 --token ncb63c.pugf9xehkjakptqp --discovery-token-ca-cert-hash sha256:73cb40755a49b40f2a724984392f32fd7ff607a437da524619d4aba43a2c31b6

1. Перезапуск сервиса:

sudo systemctl restart kubelet.service

1. Создание deployment-плана:

vi deploy.yml

apiVersion : apps/v1 #версия api

kind: Deployment #Принципал deployment

metadata: #Метаданные

name: my-web-deployment

labels:

app : my-k8s-application

spec: #Описание контейнера

selector:

matchLabels:

project: kgb #Проект

template: #Шаблон для контейнера

metadata:

labels:

project: kgb

spec:

containers: #Контейнер

- name : kgb-web #Имя контейнера

image: Debian #Имидж на котором будет основываться контейнер

ports: #Порт который нужно пробросить в хост машину

- containerPort: 80

1. Применение deployment плана:

#kubectl apply –f deploy.yml

1. Cбор отчетов о производительности
   1. Общие сведения о Zabbix

Для эффективного мониторинга Системы рекомендуется использовать службу Zabbix. Это мощное решение для мониторинга, которое способствует обеспечению стабильной работы Системы и своевременному обнаружению возможных проблем.

Пакеты Zabbix должны быть установлены администратором Системы самостоятельно, так как они не входят в сборку ОС.

Службы Zabbix являются универсальными инструментами по сбору отчетов и могут адаптироваться под различные масштабы и спецификации Систем: от небольших до крупных сред, позволяя контролировать множество устройств и сервисов.

С помощью Zabbix возможно выполнение следующего функционала в ОС:

* Отслеживание разнообразных метрик Системы, таких как использование процессора, память, дисковое пространство, сетевую активность и много других параметров, что позволяет получать полное представление о состоянии Системы;
* Настройка свои триггеров и уведомлений, что обеспечивает возможность мгновенной реакции на возникающие инциденты. Например, настройка уведомления о превышении лимитов загрузки CPU или исчерпании дискового пространства;
* Предоставление данных и отчетности по производительности Системы в графическом режиме.

Система Zabbix также предоставляет RESTful API для интеграции с другими системами или создания собственных автоматизированных процессов.

Компоненты службы Zabbix:

* Zabbix Server - основной компонент, который отвечает за сбор данных, хранение информации и обработку триггеров.
* Zabbix Agent - устанавливается на целевых системах и собирает данные о производительности и состоянии системных ресурсов, отправляя их на сервер.
* Zabbix Frontend - веб-интерфейс через веб-сервер Apache HTTPD, который позволяет администраторам и пользователям взаимодействовать с системой, осуществляя настройку, просмотр статистики и управление уведомлениями.
  1. Установка и настройка
     1. Установка компонентов Zabbix

Процесс установки и настройки Zabbix:

1. Установка сервера и агента:
   1. Сначала необходимо добавить пакеты с Zabbix в Систему:

sudo dnf install zabbix-server zabbix-server-mysql zabbix-web zabbix-web-mysql mariadb locales-ru

Это установит сервер Zabbix, веб-интерфейс Zabbix, компоненты для использования MySQL в качестве базы данных и саму базу данных — MariaDB (форк MySQL).

* 1. Установка имени хоста (это делать не обязательно, но рекомендуется), например:

sudo hostnamectl set-hostname\ zabbix.infrastructure.имя\_хоста.ru

1. Настройка базы данных:
   1. После установки MariaDB необходимо настроить базу данных для Zabbix. Для этого необходимо подключиться к MariaDB и создать базу данных и пользователя:

sudo mysql -uroot -p

* 1. Ввести SQL-запрос для создания базы данных и пользователя:

create database zabbix character set utf8 collate utf8\_bin;

create user zabbix@localhost identified by 'пароль';

grant all privileges on zabbix.\* to zabbix@localhost;

quit;

* 1. Заменить параметр "пароль" на пароль, который будет использоваться для БД.

1. Импорт начальных схем и данных в базу данных Zabbix:

zcat /usr/share/doc/zabbix-server-mysql\*/create.sql.gz | mysql -uzabbix -p zabbix

1. Настройка Zabbix Server:
   1. Настройка файла конфигурации Zabbix Server, с указанием параметров подключения к базе данных. Для этого необходимо открыть и отредактировать файл /etc/zabbix/zabbix\_server.conf:

sudo nano /etc/zabbix/zabbix\_server.conf

* 1. Найти и изменить строки следующим образом:

DBName=zabbix

DBUser=zabbix

DBPassword=password

где необходимо заменить параметр "password" на пароль, который был задан ранее для пользователя базы данных Zabbix.

1. Запуск службы Zabbix Server и Zabbix Agent и настройка их на автоматический запуск при старте Системы:

sudo systemctl restart zabbix-server zabbix-agent apache2

sudo systemctl enable zabbix-server zabbix-agent apache2

* + 1. Настройка веб-интерфейса

Рассмотрим шаги по настройке веб-интерфейса Zabbix:

1. Настройка PHP:

Откройте файл /etc/zabbix/apache.conf и настройте параметры PHP для Zabbix:

sudo nano /etc/zabbix/apache.conf

Обновите следующие значения параметров:

php\_value date.timezone Europe/Moscow

Убедитесь, что установлен корректный часовой пояс, соответствующий вашему региону.

1. Перезапустите веб-сервер Apache, чтобы применить изменения:

sudo systemctl restart apache2

1. Первоначальная настройка через веб-интерфейс. Откройте веб-браузер и перейдите по адресу http://your\_server\_ip/zabbix, где вместо «your\_server\_ip» укажите ip-адрес вашего сервера. Следуйте инструкциям на экране, чтобы завершить установку.

В ходе настройки потребуется ввести имя базы данных, пользователя и пароль, которые вы настроили ранее.

* + 1. Установка Zabbix Agent на целевых машинах

Для подключения целевых машин к Zabbix Agent необходимо:

1. Установить Zabbix Agent на целевые машины:

sudo apt-get install zabbix-agent

1. Настройка Zabbix Agent, для этого открыть на редактирование файл конфигурации /etc/zabbix/zabbix\_agentd.conf:

sudo nano /etc/zabbix/zabbix\_agentd.conf

Найдите и измените следующие строки:

Server=your\_zabbix\_server\_ip

ServerActive=your\_zabbix\_server\_ip

Hostname=your\_target\_system\_hostname

Заменить параметр "your\_zabbix\_server\_ip" на IP-адрес вашего сервера Zabbix и "your\_target\_system\_hostname" на имя хостовой машины.

1. Запуск Zabbix Agent и добавление его на автоматический запуск при старте целевой системы:

sudo systemctl restart zabbix-agent

sudo systemctl enable zabbix-agent

* + 1. Завершение настройки

Далее можно вернуться в веб-интерфейс Zabbix и добавить новые хосты для мониторинга. В разделе "Configuration → Hosts" добавляются целевые системы, на которых установлен Zabbix Agent, и настраиваются необходимые параметры мониторинга.

1. Кластеризация и высокая доступность
   1. Общие сведения

Pacemaker и Corosync являются компонентами отказоустойчивой кластерной среды с открытым исходным кодом, предназначенной для обеспечения высокой доступности сервисов.

Pacemaker представляет собой менеджер ресурсов кластера (Cluster Resource Manager, CRM), отвечающий за управление состоянием ресурсов в кластере и их автоматическое переключение при отказах. Corosync обеспечивает кластерное членство и обмен сообщениями между узлами, выступая в роли коммуникационного уровня.

Кластерная архитектура Pacemaker и Corosync условно делится на три уровня:

* CIB (Cluster Information Base) — уровень, независимый от кластера. Содержит конфигурацию ресурсов и агентов управления. CIB представляет собой XML-базу, реплицируемую между всеми узлами кластера. Управление осуществляется с любого узла;
* Cluster Resource Manager (CRMd) — уровень управления ресурсами, реализуемый демоном pacemaker. Отвечает за реакцию на события, происходящие в кластере (отказы, переключения узлов, действия администратора), расчет оптимального состояния и реализацию необходимых операций;
* PEngine (Policy Engine) — информационный уровень. Обеспечивает сетевое взаимодействие, обработку кворума, распределение команд управления и реализацию политик работы кластера. Этот уровень опирается на Corosync.

Ключевые компоненты кластера:

* DC (Designated Coordinator) — основной управляющий узел, выбранный путём голосования. При его недоступности выполняется автоматическая переизбрание;
* LRMd (Local Resource Manager daemon) — демон управления локальными ресурсами. Отвечает за исполнение команд через RA (Resource Agent) — скрипты, управляющие жизненным циклом ресурса;
* STONITHd (Shoot The Other Node In The Head) — компонент защиты от порчи данных. Обеспечивает физическое отключение узла в случае критического сбоя, в том числе через IPMI, BMC или внешние устройства.

С точки зрения Pacemaker ресурсом считается любой объект, управляемый через скрипты: от IP-адреса и службы Apache до систем с репликацией и режимами master/slave. Управление ресурсами должно осуществляться исключительно через кластер, исключая ручной запуск или автоматический запуск при загрузке ОС.

* + 1. **Типы ресурсов**

В Pacemaker реализованы следующие типы ресурсов:

* Primitive — базовый тип, представляющий собой один экземпляр ресурса;
* Clone — ресурс, который может быть запущен на нескольких узлах. Делится на типы: Anonymous, Globally Unique и Stateful;
* Multi-state — расширение Clone-ресурса с поддержкой состояний Master и Slave. Применяется, например, для DRBD.

Дополнительно ресурсы могут объединяться в Group, которая обеспечивает запуск на одном узле в заданном порядке.

Поддерживаются механизмы:

* Migration — "живой" перенос ресурса между узлами (при поддержке агентом операций migrate\_to и migrate\_from);
* Colocation — указание на необходимость совместного или раздельного размещения ресурсов;
* Ordering — определение порядка запуска и остановки ресурсов;
* Location — указание предпочтений по размещению ресурсов на узлах.
  + 1. Политики управления

Pacemaker позволяет управлять поведением ресурсов при отказах и восстановлении узлов с помощью параметров:

* resource-stickiness — желательность оставаться на текущем узле;
* migration-threshold — количество отказов, после которых ресурс перемещается;
* failure-timeout — интервал, после которого считается, что сбой более неактуален;
* multiple-active — поведение при запуске ресурса на нескольких узлах;
* monitor — периодическая проверка состояния ресурса (указывается через параметр interval).

Уровень кворума определяет возможность функционирования кластера. По умолчанию кластер считается работоспособным при наличии более 50% работоспособных узлов. Поведение при потере кворума задаётся параметром no-quorum-policy.

Pacemaker поддерживает разнообразные схемы отказоустойчивости: Active/Passive, Active/Active, N+1, N-to-1 и другие.

* 1. Развёртывание и настройка Pacemaker и Corosync

Данный раздел содержит пошаговую инструкцию по развёртыванию кластера высокой доступности на двух серверах под управлением ОС РОСА "ХРОМ" с использованием компонентов Pacemaker и Corosync. В качестве примера рассматривается настройка кластера с веб-сервером Nginx:

1. Подготовка Системы.

На всех узлах, планируемых к объединению в кластер, необходимо внести записи в файл /etc/hosts, чтобы обеспечить корректное разрешение имён.

Пример содержимого файла /etc/hosts:

10.45.4.58 webserver-01

10.45.4.59 webserver-02

Редактирование может быть выполнено с использованием любого доступного текстового редактора.

1. Установка и первичная настройка Nginx.

На каждом сервере устанавливается веб-сервер Nginx с помощью менеджера пакетов. После установки служба добавляется в автозагрузку и запускается командами:

sudo dnf install nginx

sudo systemctl enable nginx

sudo systemctl start nginx

Для визуального различения узлов в кластере рекомендуется изменить содержимое страницы по умолчанию. На каждом сервере в файл /usr/share/nginx/html/index.html добавляется уникальная метка.

На сервере webserver-01:

echo "<h1>webserver-01</h1>" >\ /usr/share/nginx/html/index.html

На сервере webserver-02:

echo "<h1>webserver-02</h1>" >\ /usr/share/nginx/html/index.html

1. Установка компонентов Pacemaker и Corosync.

Для настройки кластера устанавливаются пакеты corosync, pacemaker и pcsd. Указанные службы добавляются в автозагрузку и запускаются.

sudo dnf install corosync pacemaker

sudo systemctl enable corosync

sudo systemctl enable pacemaker

sudo systemctl enable pcsd

sudo systemctl start corosync

sudo systemctl start pacemaker

sudo systemctl start pcsd

В процессе установки создаётся системный пользователь hacluster, необходимый для аутентификации между узлами. На всех узлах кластера для пользователя hacluster задаётся одинаковый пароль.

passwd hacluster

1. Аутентификация и создание кластера.

После установки и настройки компонентов необходимо выполнить аутентификацию между узлами с использованием утилиты pcs:

sudo pcs cluster auth webserver-01 webserver-02

Затем создаётся кластер с именем rosacluster, включающий оба сервера:

sudo pcs cluster setup --force --name rosacluster webserver-01 webserver-02

Для корректного функционирования кластера рекомендуется повторно включить службы в автозагрузку и перезагрузить Систему:

sudo systemctl enable corosync

sudo systemctl enable pacemaker

sudo systemctl enable pcsd

reboot

1. Проверка состояния кластера после перезагрузки с помощью команды:

pcs status

* 1. Управление ресурсами и политиками в кластере
     1. Отключение STONITH

STONITH (Shoot-The-Other-Node-In-The-Head) представляет собой механизм аппаратного ограждения, применяемый для исключения повреждения общих ресурсов в случае отказа узла. В производственной среде рекомендуется включать STONITH и использовать совместимые устройства управления питанием или IPMI. Однако в демонстрационных или тестовых средах при отсутствии таких устройств функцию STONITH допускается отключить.

При отсутствии настроенных устройств STONITH при выполнении команды pcs status в выводе будет отображаться предупреждение.

Для отключения STONITH в кластере выполняется команда:

sudo pcs property set stonith-enabled=false

* + 1. Отключение контроля кворума

По умолчанию кластер прекращает управление ресурсами, если теряет кворум (менее половины активных узлов). Для тестовой конфигурации, в частности двухузлового кластера, может быть необходимо отключить реакцию на потерю кворума. Это осуществляется с помощью настройки no-quorum-policy.

Отключение реакции на потерю кворума выполняется следующей командой:

sudo pcs property set no-quorum-policy=ignore

Текущие свойства кластера можно просмотреть командой:

pcs property list

* + 1. Добавление ресурсов в кластер

Одним из ключевых элементов высокой доступности является плавающий IP-адрес — адрес, который может автоматически перемещаться между узлами в пределах одной сети. В приведённом примере используется IP-адрес 10.45.4.60.

Для регистрации IP-адреса в виде ресурса с именем v\_ip используется следующая команда:

sudo pcs resource create v\_ip ocf:heartbeat:IPaddr2 ip=10.45.4.60 cidr\_netmask=24 op monitor interval=20s

Далее добавляется ресурс nginx, представляющий службу веб-сервера, управляемую через systemd:

sudo pcs resource create nginx systemd:nginx

Состояние ресурсов проверяется с помощью команды:

sudo pcs status resources

Если в выводе отображаются ресурсы v\_ip и nginx, это означает, что плавающий IP-адрес и веб-сервер были успешно добавлены в кластер.

* + 1. Настройка политики совместного размещения ресурсов

Для обеспечения корректной работы веб-сервера Nginx необходимо, чтобы он всегда запускался на том же узле, где активен плавающий IP-адрес. Для этого используется политика коллокации.

Настройка коллокации выполняется командой:

sudo pcs constraint colocation add nginx v\_ip INFINITY

Также необходимо определить порядок запуска ресурсов: сначала должен быть активирован IP-адрес, затем — веб-сервер.

Настройка порядка запуска ресурсов:

sudo pcs constraint order v\_ip then nginx

Проверка текущего состояния ресурсов кластера выполняется командой:

sudo pcs status

* 1. Проверка работоспособности кластера

Для проверки корректности настройки отказоустойчивого кластера необходимо убедиться, что ресурсы (веб-сервер Nginx и плавающий IP-адрес) автоматически переключаются между узлами в случае отказа одного из них.

Для выполнения проверки необходимо:

1. Открыть веб-браузер и перейти по адресу:

http://10.45.4.60

На открывшейся странице можно увидеть индексную страницу Nginx, размещённую на первом узле webserver-01.

1. Для эмуляции отказа текущего ведущего узла выполняется остановка кластера на webserver-01:

sudo pcs cluster stop webserver-01

1. После остановки кластера на первом узле при повторном обращении к адресу http://10.45.4.60 будет отображена индексная страница, размещённая на узле webserver-02. Это означает, что ресурсы кластера автоматически переместились на доступный узел, обеспечив непрерывность сервиса.

Проверка выполнения всех шагов позволяет подтвердить успешную настройку отказоустойчивого кластера с использованием Pacemaker и Corosync.

* 1. Мониторинг и управление отказами

Для обеспечения корректной работы ресурсов в кластере необходимо настроить их регулярную проверку. В Pacemaker для этого используется операция monitor.

Операция мониторинга добавляется в момент создания ресурса с указанием интервала:

sudo pcs resource create nginx systemd:nginx op monitor interval=30s

Если ресурс уже создан, параметр можно добавить отдельно:

sudo pcs resource op add nginx monitor interval=30s

Pacemaker отслеживает результат работы проверки и реагирует согласно установленной политике. Рекомендуется также задать параметры, определяющие поведение кластера при сбоях:

* migration-threshold — количество сбоев, после которого ресурс переносится на другой узел;
* failure-timeout — интервал (в секундах), через который сбой считается неактуальным и может быть повторно допущен;
* on-fail — политика действия при сбое (restart, fence, standby, block и др.).

Пример команды:

sudo pcs resource update nginx migration-threshold=2 failure-timeout=60 on-fail=restart

Для отслеживания статуса ресурсов рекомендуется использовать команду:

sudo pcs status resources

Логирование событий выполняется службой Pacemaker и доступно через системный журнал:

journalctl -u pacemaker

* 1. Управление ресурсами и диагностика

В процессе эксплуатации кластера может потребоваться ручное вмешательство: перемещение ресурсов между узлами, временное отключение ресурса, снятие ограничений.

Перемещение ресурса на конкретный узел:

sudo pcs resource move nginx webserver-02

Запрет запуска ресурса на узле:

sudo pcs resource ban nginx webserver-01

Очистка всех ограничений, связанных с ресурсом:

sudo pcs resource clear nginx

Временное отключение ресурса:

sudo pcs resource disable nginx

Повторное включение ресурса:

sudo pcs resource enable nginx

Просмотр состояния кластера с детализацией:

sudo pcs status

sudo pcs status resources

sudo pcs status nodes

Дополнительно может использоваться команда для получения краткого отчёта по текущему состоянию кластера:

sudo crm\_mon -1

Рекомендуется регулярно сохранять конфигурацию кластера:

sudo pcs config backup rosacluster-backup.tar.gz

Для восстановления используется команда:

sudo pcs config restore rosacluster-backup.tar.gz

1. Локальная виртуализация с использованием qemoo

Утилита qemoo представляет собой обёртку для запуска виртуальных машин с использованием эмулятора QEMU. Скрипт предназначен для упрощения командной строки QEMU и автоматизации типовых операций, таких как:

* запуск ISO-образов и виртуальных дисков;
* автоматическое создание дисков в формате qcow2;
* эмуляция загрузки в режиме EFI;
* проброс физических устройств и каталогов;
* работа через systemd и поддержка демон-режима SPICE;
* настройка сетевых интерфейсов виртуальных машин;
* управление конфигурацией виртуальных машин через конфигурационные файлы.

Утилита входит в состав дистрибутива ОС и предназначена для использования администраторами при отладке, тестировании, развёртывании и сопровождении виртуальных инсталляций ОС.

Для запуска виртуальной машины с использованием qemoo достаточно передать путь к ISO-образу, блочному устройству или файлу виртуального диска. Примеры:

qemoo Rosa.iso

qemoo /dev/sda

qemoo ./Rosa.qcow2

При необходимости загрузки в режиме EFI используется ключ -e:

qemoo -e Rosa.qcow2

Дополнительные ключи и параметры позволяют расширить функциональность скрипта и адаптировать его под конкретные задачи виртуализации. Все параметры передаются в одной командной строке, а конфигурация может задаваться как явно, так и через конфигурационные файлы, размещённые локально или глобально.

* 1. Запуск виртуальных машин с помощью qemoo

Для запуска виртуальных машин с использованием qemoo необходимо передать в качестве аргумента путь к ISO-образу, блочному устройству или образу виртуального диска в формате .qcow2. Утилита самостоятельно определяет параметры запуска на основе переданного файла и применяет настройки по умолчанию или из конфигурационных файлов.

Примеры запуска:

qemoo Rosa.iso

qemoo /dev/sda

qemoo ./Rosa.qcow2

* + 1. Загрузка в режиме EFI

При необходимости загрузки в режиме EFI используется ключ -e:

qemoo -e Rosa.qcow2

Также может быть применена комбинированная команда:

qemoo -i -e /dev/sdb

В этом случае будет произведена установка с ISO в EFI-режиме на указанный диск или образ.

* + 1. Добавление дополнительных устройств и параметров

Для подключения дополнительных устройств (например, второго диска или USB-накопителя) используется ключ -a. Пример запуска ISO-образа с подключением второго устройства:

qemoo Rosa.iso -a /dev/sdb

Параметры, поддерживаемые эмулятором QEMU, могут передаваться напрямую после двойного дефиса "--". Например, передача параметра количества виртуальных процессоров:

qemoo -e Rosa.iso -- -smp 4

* + 1. Загрузка в режиме, имитирующем запись ISO-образа на носитель

Скрипт поддерживает режим, в котором ISO-образ загружается так, как если бы он был записан побайтово на флеш-накопитель. Для этого используется ключ -l:

qemoo -l Rosa.iso

qemoo -l -e Rosa.iso

* + 1. Проброс USB-устройств

Для проброса физических USB-устройств в виртуальную машину используется ключ -L, за которым следует путь к устройству:

qemoo -L /dev/sdb

Такая схема может быть использована, например, для загрузки виртуальной машины с USB-модема или токена, содержащего ОС.

* 1. Установка ОС на виртуальный диск

Скрипт qemoo поддерживает автоматизированную установку ОС с ISO-образа на виртуальный диск в формате .qcow2. Для этого используется ключ -i.

При запуске с данным ключом в текущем каталоге создаётся файл виртуального диска, который автоматически подключается к виртуальной машине при запуске.

Примеры установки:

qemoo -i Rosa.iso

В режиме EFI:

qemoo -i -e Rosa.iso

Также возможно указание целевого устройства (например, реального диска /dev/sdb):

qemoo -i -e /dev/sdb

После завершения установки возможно использование ранее созданного .qcow2-образа для загрузки, передав его в qemoo как аргумент:

qemoo \_qemoo1\_ROS.qcow2

Имена создаваемых образов формируются автоматически, но могут быть заданы и вручную.

Также поддерживается установка с ISO-образа на физический диск. В этом случае используется ключ -a, указывающий дополнительное устройство для записи:

qemoo Rosa.iso -a /dev/sdb

Во всех режимах установки применяется типовая конфигурация QEMU, достаточная для корректной загрузки и установки ОС РОСА "ХРОМ".

* 1. Сетевые режимы и проброс устройств

При запуске виртуальных машин с использованием qemoo автоматически настраивается сетевое подключение, обеспечивающее взаимодействие гостевой системы с хостом и другими машинами в сети.

* + 1. Автоматическое подключение к подсети virbr0

Если на хост-системе установлена и запущена служба libvirtd, виртуальные машины, запущенные с помощью qemoo, автоматически получают IP-адрес из подсети 192.168.122.0/24, связанной с виртуальным интерфейсом virbr0.

Такая конфигурация обеспечивает полноценную сетевую связность и доступ к интернету через NAT. В случае отсутствия интерфейса virbr0 (например, если libvirtd не установлен или не активен) для каждой виртуальной машины создаётся собственная NAT-сеть, что может ограничивать взаимодействие между гостевыми системами и внешней средой.

* + 1. Настройка собственного сетевого моста

Для получения большей гибкости возможно создание и использование собственного сетевого моста с именем qemoobr0. При наличии такого моста утилита qemoo игнорирует интерфейс virbr0 и использует qemoobr0 для подключения виртуальных машин.

Настройка моста производится средствами Системы. Конкретные действия зависят от используемой сетевой конфигурации хоста.

* + 1. Проброс каталогов в гостевую систему

При запуске виртуальной машины с помощью qemoo может быть автоматически проброшен каталог с хоста внутрь гостевой системы. По умолчанию пробрасывается текущий каталог (./), из которого выполняется запуск. При старте виртуальной машины в консоль выводится инструкция по подключению каталога внутри гостевой ОС на базе Linux.

* + 1. Проброс USB-устройств

Для проброса USB-устройства в виртуальную машину используется ключ -L. Указывается путь к устройству:

qemoo -L /dev/sdb

Этот режим может быть использован, например, для загрузки с 4G-модема с SD-картой, токена с предустановленной Системой или иных внешних накопителей.

* 1. Работа в демон-режиме (SPICE)

Утилита qemoo поддерживает запуск виртуальной машины в виде фонового демона с выводом графического интерфейса через протокол SPICE. Это позволяет подключаться к экрану гостевой ОС с помощью внешних клиентов, таких как virt-viewer или Remmina.

* + 1. Запуск в SPICE-режиме

Для запуска виртуальной машины в демон-режиме используется ключ -d. При этом qemoo автоматически выделяет свободный порт и сообщает его в вывод консоли:

qemoo -d Rosa.iso

Вывод будет содержать номер порта, на который можно подключиться из клиента SPICE. Подключение выполняется вручную, указав IP-адрес хоста и полученный порт.

* + 1. Управление виртуальной машиной через systemd

При запуске виртуальной машины с ключом -i (установка ОС) qemoo создаёт рядом с образом файл конфигурации с расширением .conf. Этот файл содержит параметры запуска и может использоваться для интеграции с systemd.

Пример содержимого конфигурационного файла:

ACTION=run

RAM="auto"

ADD=""

EFI="-bios /usr/share/OVMF/OVMF\_CODE.fd"

PORT=""

REDIRUSB=""

LOSETUP=""

SPICE=""

SHARE="/home/user/Downloads"

QEMOOADD=""

Для запуска виртуальной машины от имени пользователя root через systemd применяется следующая команда:

sudo systemctl start qemoo@$(systemd-escape\ /путь/к/образ.qcow2)

Проверка статуса виртуальной машины:

systemctl status qemoo@$(systemd-escape /путь/к/образ.qcow2)

Для включения автозапуска виртуальной машины при старте хост-системы:

sudo systemctl enable qemoo@$(systemd-escape\ /путь/к/образ.qcow2)

Отключение автозапуска:

sudo systemctl disable qemoo@$(systemd-escape\ /путь/к/образ.qcow2)

Запуск от имени текущего пользователя с использованием user-юнитов:

sudo systemctl --user enable qemoo@$(systemd-escape\ /путь/к/образ.qcow2)

Команда systemd-escape используется для корректного экранирования символов "/" в путях файловой системы при создании unit-имен.

* 1. Конфигурация и приоритет параметров

Скрипт qemoo поддерживает использование конфигурационных файлов, в которых могут задаваться параметры запуска виртуальных машин. Эти параметры могут быть как общими для всех запусков, так и специфичными для отдельных образов. Это позволяет централизованно управлять поведением qemoo без необходимости передавать параметры вручную в командной строке при каждом запуске.

При запуске qemoo учитываются конфигурационные файлы в следующем порядке приоритета (от меньшего к большему):

1. глобальный файл – /etc/qemoo.cfg. Используется по умолчанию, если переменная окружения $QEMOOCFG не задана;
2. локальный файл в текущем каталоге – ./qemoo.cfg;
3. индивидуальный конфигурационный файл для образа: если виртуальная машина запускается из файла образа example.qcow2, то приоритет будет иметь файл example.qcow2.conf, расположенный рядом с ним;
4. конфигурационный файл, явно указанный в параметре: при запуске с опцией "--config – /путь/к/файлу.cfg".

Если в конфигурационных файлах заданы одинаковые параметры, приоритет имеет значение из файла с более высоким уровнем.

* + 1. Типовые параметры конфигурации

В конфигурационном файле могут быть заданы следующие переменные:

* ACTION — тип действия (run, install и др.);
* RAM — объём оперативной памяти, выделяемой виртуальной машине;
* ADD — путь к дополнительному устройству (например, дополнительный диск);
* EFI — параметры для запуска в режиме EFI;
* PORT — порт для SPICE-подключения;
* SPICE — активация вывода видео через SPICE (yes / no);
* REDIRUSB — проброс USB-устройств;
* LOSETUP — использование losetup при работе с образом;
* SHARE — каталог, пробрасываемый в гостевую ОС;
* QEMOOADD — дополнительные параметры командной строки для qemu-system-\*.

Пример конфигурационного файла:

ACTION=run

RAM=2048

ADD=/dev/sdb

EFI="-bios /usr/share/OVMF/OVMF\_CODE.fd"

SPICE=yes

PORT=6001

SHARE=/home/user/Projects

1. Создание собственного репозитория пакетов

Репозиторий пакетов представляет собой хранилище RPM-файлов, дополненное служебными файлами метаданных, обеспечивающими установку и обновление пакетов с использованием менеджера пакетов dnf. Подробнее о работе с dnf см. раздел 3.6.1.

Создание собственного репозитория может быть необходимо для решения следующих задач:

* централизованное размещение сторонних пакетов, используемых в организации, для их установки и обновления из единого источника;
* создание зеркала (точной копии) официальных репозиториев РОСА в пределах локальной сети предприятия.

Примечание – Все команды, требующие прав суперпользователя, должны выполняться с использованием sudo. В остальных случаях выполнение команд производится от имени непривилегированного пользователя.

* 1. Зеркалирование репозиториев

Зеркалирование репозитория заключается в создании и регулярном обновлении локальной копии репозиториев, размещённых на официальных серверах РОСА. Такое зеркало позволяет сократить внешнее сетевое взаимодействие и ускорить установку и обновление пакетов внутри корпоративной сети.

Для зеркалирования рекомендуется использовать утилиту rsync. Инструкция по созданию зеркала с использованием rsync:

1. Если утилита rsync не установлена, необходимо выполнить установку командой:

sudo dnf install rsync

1. Создание каталогов для хранения копий репозиториев:

sudo mkdir -p /srv/mirror/x86\_64 /srv/mirror/i686

1. Запуск синхронизации с официальным сервером:

sudo rsync \

-av --progress \

--exclude SRPMS \

--exclude 'debug\_\*' \

--exclude \_\_\_REMOVED \

--exclude '\*-rpm-backup' \

rsync://mirror.rosalab.ru/rosa/rosa2021.1/repository/{x86\_64,i686} /srv/mirror/

В приведённой команде синхронизируются репозитории для архитектур x86\_64 и i686.

Каталоги, содержащие пакеты с отладочной информацией (debug\_\*) и SRPM-исходники (SRPMS), исключаются из копирования для экономии дискового пространства.

Регулярное выполнение задачи синхронизации рекомендуется автоматизировать с использованием планировщика systemd timer. Подробная инструкция по созданию планируемых задач приведена в разделе 3.4.2.1.

* 1. Создание собственного репозитория

Если необходимо использовать сторонние RPM-пакеты, можно организовать собственный репозиторий для централизованного хранения и распространения этих пакетов.

Для создания репозитория необходимо выполнить следующие действия:

1. установить утилиту для формирования метаданных репозитория:

sudo dnf install createrepo\_c

1. создать каталог для хранения пакетов:

sudo mkdir -p /srv/repo/x86\_64

1. поместить все необходимые RPM-пакеты в созданный каталог /srv/repo/x86\_64.
2. создать или обновить метаданные репозитория:

sudo createrepo\_c /srv/repo/x86\_64

для обновления репозитория после добавления или удаления пакетов необходимо повторно выполнить команду на шаге **г)**;

1. для проверки содержимого репозитория используется следующая команда:

dnf --repofrompath=local,/srv/repo/x86\_64 --disablerepo='\*' --enablerepo=local list

* 1. Подключение локального репозитория

Для локального подключения репозитория необходимо создать конфигурационный файл в каталоге /etc/yum.repos.d/. В файле указывается путь к репозиторию на локальной Системе.

Пример файла myrepo.repo:

[myrepo]

name=My Local Repository

baseurl=file:///path/to/my/repo

enabled=1

gpgcheck=0

где /path/to/my/repo — путь к каталогу, содержащему RPM-пакеты и созданные метаданные.

После создания конфигурационного файла пакеты из локального репозитория становятся доступными для установки с помощью dnf наравне с другими подключёнными репозиториями.

* 1. Доступ к репозиторию по сети

Организовать сетевой доступ к локальному репозиторию пакетов можно с использованием HTTP или NFS.

* + 1. Доступ по HTTP

Для предоставления доступа по протоколу HTTP достаточно использовать любой веб-сервер, способный отдавать статические файлы. В качестве примера рассмотрена настройка веб-сервера angie (об этом подробнее см. раздел 12.4.3).

Для настройки HTTP-сервера нужно выполнить следующие шаги:

1. Установить веб-сервер:

sudo dnf install angie

1. Открыть основной конфигурационный файл на редактирование:

sudo nano /etc/angie/angie.conf

1. Привести файл к необходимому виду. Ниже приведён минимальный пример:

worker\_processes 1;

events {

worker\_connections 1024;

}

http {

include mime.types;

default\_type application/octet-stream;

sendfile on;

keepalive\_timeout 65;

server {

listen 80;

server\_name localhost;

location / {

root /srv;

autoindex on;

index index.html index.htm;

}

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root html;

}

}

}

Для выхода и сохранения изменений в редакторе nano использовать комбинации клавиш: Ctrl+X, затем Enter и Ctrl+O.

1. Запустить веб-сервер и добавить его в автозапуск:

sudo systemctl enable --now angie

1. Проверить доступ к репозиторию, открыв в браузере адрес http://<IP-адрес\_сервера>:80. Должно отобразиться содержимое каталога /srv.

Примечание – Если требуется только временный или тестовый доступ, можно использовать встроенный модуль Python:

python3 -m http.server

По умолчанию сервер запускается на порту 8000. В конфигурационном файле репозитория указать адрес в формате:

baseurl=http://192.168.1.100:8000

Для настройки HTTP-клиента необходимо:

1. В файлах конфигурации /etc/yum.repos.d/\*.repo указать адрес собственного сервера. Пример:

Было:

baseurl=http://mirror.rosalab.ru/rosa/rosa$releasever/repository/x86\_64/main/release/

Стало:

baseurl=http://192.168.1.100/repo/x86\_64/rosa$releasever/repository/x86\_64/main/release/

При необходимости можно указать несколько репозиториев через запятую — в случае недоступности первого будет использован следующий.

1. Проверить работу подключённого репозитория:

sudo dnf makecache

* + 1. Доступ по NFS

Для организации доступа по сети с использованием NFS необходимо настроить NFS-сервер и клиент.

Для настройки NFS-сервера необходимо:

1. Установить утилиты NFS:

sudo dnf install nfs-utils

1. Открыть файл конфигурации сервера:

sudo nano /etc/exports

1. Добавить строку с настройками общего доступа. Пример:

/srv 192.168.1.1/255.255.255.0(ro,insecure,nohide,all\_squash,anonuid\=1000,anongid=1000,no\_subtree\_check)

В приведённом примере доступ предоставляется только на чтение для сети с адресами от 192.168.1.1 до 192.168.1.255.

1. Запустить сервер и добавить его в автозапуск:

sudo systemctl enable --now nfs-server

Для настройки NFS-клиента необходимо:

1. Создать каталог для монтирования:

sudo mkdir -p /mnt/nfs

1. Открыть файл /etc/fstab для редактирования:

sudo nano /etc/fstab

1. Добавить строку монтирования NFS-ресурса:

192.168.1.100:/srv /mnt/nfs nfs4 auto,\_netdev,bg,nofail

где 192.168.1.100 — адрес NFS-сервера.

1. Выполнить монтирование для проверки:

sudo mount /mnt/nfs

1. Указать путь к локальному репозиторию в файле /etc/yum.repos.d/\*.repo:

Заменить содержимое строки:

baseurl=http://mirror.rosalab.ru/rosa/rosa$releasever/repository/x86\_64/main/release/

на следующее содержимое:

baseurl=file:///mnt/nfs/srv/repo/x86\_64/rosa$releasever/repository/x86\_64/main/release/

1. Проверить работу подключённого репозитория:

sudo dnf makecache

Вывод должен представлять собой список репозиториев, подключённых в Системе и успешно обработанных пакетным менеджером dnf.

Перечень терминов и сокращений

| Сокращение | Определение |
| --- | --- |
| АПМДЗ | Аппаратно-программный модуль доверенной загрузки |
| БД | База данных |
| ВМ | Виртуальная машина |
| ЗПС | Замкнутая программная среда |
| ИТ | Информационные технологии |
| ОС | Операционная система |
| ПК | Персональный компьютер |
| ПО | Программное обеспечение |
| СДЗ | Средство доверенной загрузки |
| СУБД | Система управления базой данных |
| ЦС | Центр сертификации |
| ФС | Файловая система |
| ACL | Access Control List – список контроля доступа |
| AIDE | Advanced Intrusion Detection Environment – усовершенствованная система обнаружения вторжений |
| API | Application programming interface – программный интерфейс приложения |
| **BIOS** | **Basic Input/Output System** – базовая подсистема ввода-вывода |
| CA | Certification authority – центр сертификации (удостоверяющий центр) |
| CIFS | Common Internet File System – протокол, являющийся частью протокола SMB, для удаленного подключения нескольких платформ – Windows, Linux или macOS, для идентификации и чтения файлов системы NTFS в Windows и HFS в macOS, чтобы совместно использовать эти файлы и принтеры между машинами с Windows и Linux |
| CSV | Comma separated values – текстовый формат, предназначенный для представления табличных данных |
| DHCP | Dynamic host configuration protocol – протокол динамической настройки сетевой конфигурации хоста |
| DNS | Domain name system – система доменных имен |
| ENC | External node classifier – классификатор внешних хостов |
| ESR | Extended support release – релиз с расширенной (долговременной) поддержкой |
| Ext4 | Fourth Extended Filesystem – журналируемая файловая система |
| Ext3 | Third Extended Filesystem – журналируемая файловая система |
| GUI | Graphical User Interface — это графический пользовательский интерфейс, |
| HTTP | Hypertext transfer protocol – протокол передачи гипертекста |
| HTTPS | Hypertext transfer protocol secure – защищенная версия протокола передачи гипертекста |
| IP | Internet protocol – протокол межсетевого взаимодействия |
| IPv4 | Internet Protocol Version 4 – версия 4 интернет-протокола |
| IPv6 | Internet Protocol Version 6 – версия 6 интернет-протокола |
| IPP | Internet Printing Protocol – Интернет-протокол печати |
| IPA | Identity, policy and audit – система идентификации, политик и аудита (СИПА) |
| LDAP | Lightweight directory access protocol – протокол доступа к каталогам |
| LDAPS | Lightweight directory access protocol secure – защищенная версия протокола доступа к каталогам |
| LUKS | Linux Unified Key Setup – система шифрования дисков, которая хранит данные в зашифрованном физическом разделе |
| MAC | Media access control – уникальный идентификатор сетевого оборудования |
| MBR | Мaster boot record – главная загрузочная запись |
| MTA | Mail transfer agent – агент передачи почтовых сообщений |
| NTFS | New Technology File System — это проприетарная файловая система, разработанная компанией Microsoft. |
| NTP | Network time protocol – протокол сетевого времени |
| SMB | Server Message Block – протокол соединения с устройствами со старыми версиями ОС Microsoft для подключения к серверам и обмена файлами |
| SMTP | Simple mail transfer protocol – протокол передачи почтовых сообщений |
| SSH | Secure shell – защищенная оболочка |
| SSL | Secure sockets layer – уровень защищенных сокетов |
| TCP | Transmission control protocol – протокол управления передачей данных |
| TFTP | Trivial file transfer protocol – протокол передачи файлов |
| UDP | User datagram protocol – протокол пользовательских датаграмм |
| URL | Uniform resource locator – сетевой адрес ресурса |
| USB | Universal Serial Bus – стандарт для подключения и передачи данных между ПК и периферийными устройствами |
| XFS | журналируемая файловая система |
| YAML | Yet another markup language – язык разметки |
|  |  |