ООО ДиджиТекГруп
ОГРН: 1117746323892

Документация, содержащая информацию, необходимую для эксплуатации экземпляра программного обеспечения НАЙС ОС Z, предоставленного для проведения экспертной проверки

Описание и область применения операционной системы НАЙС ОС Z

НАЙС ОС Z является многопользовательской, многозадачной ОС, которая предоставляет платформу унифицированной функциональной универсальной доверенной среды для выполнения прикладного программного обеспечения. НАЙС ОС Z на уровне драйверов поддерживает широкий перечень оборудования актуальных версий, доступного на рынке средств вычислительной техники (СВТ), а также оборудования, снятого с производства, но поддерживаемого производителями. В НАЙС ОС Z поддерживается установка с оптических носителей информации, флешнакопителей, разделов локального жёсткого диска, а также установка по сети передачи данных.

Основные функции НАЙС ОС Z

НАЙС ОС Z может обеспечивать обслуживание от одного до нескольких пользователей одновременно. После успешного входа в систему пользователи имеют доступ в главную вычислительную среду, позволяющую запускать пользовательские приложения, создавать и получать доступ к файлам, задавать директивы пользователя на уровне оболочки командного процессора.

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

НАЙС ОС Z конфигурируется по умолчанию для работы в режиме дискреционного управления доступом (DAC). Любой пользователь с ролью, которая позволяет выполнять административные действия, считается административным пользователем. Кроме того, НАЙС ОС Z поддерживает типы, которые могут быть связаны с объектами, и домены, которые могут быть связаны с процессами.

Роли определяются доменами, к которым они имеют доступ. Предопределённый файл политики, который является частью конфигурации

НАЙС ОС Z, определяет правила между доменами и типами. С вышеизложенным определением ролей и прав доступа, подразумеваемых индивидуальными ролями, НАЙС ОС Z выполняет требования ролевого доступа. НАЙС ОС Z предназначена для работы в сетевом окружении с другими экземплярами НАЙС ОС Z, а также с иными совместимыми серверными и клиентскими системами одного и того же управляемого домена. Все эти системы должны конфигурироваться в соответствии с определённой общей политикой безопасности. НАЙС ОС Z разрешает использование многими пользователями одного или более процессоров, присоединённых внешних и запоминающих устройств для выполнения разнообразных функций, требующих управляемого распределенного доступа к данным, хранимым в системе. Такие инсталляции типичны для вычислительных систем рабочих групп или предприятий, к которым обращаются локальные пользователи, или компьютерных систем с иначе защищённым доступом.

Предполагается, что ответственность за сохранение данных, защищаемых НАЙС ОС Z, может делегироваться пользователям НАЙС ОС Z. Все данные находятся под управлением механизмов безопасности НАЙС ОС Z. Данные сохраняются в поименованных объектах, и НАЙС ОС Z может связать с каждым поименованным объектом описание прав доступа к этому объекту. Всем пользователям назначаются уникальные идентификаторы. Этот идентификатор пользователя используется вместе с атрибутами и ролями, назначенными пользователю, как основание для решений по управлению доступом.

НАЙС ОС Z подтверждает подлинность предъявленного идентификатора пользователя до того, как разрешать ему выполнять дальнейшие действия. НАЙС ОС Z внутри себя сопровождает ряд идентификаторов, связанных с процессами, которые получаются из уникального идентификатора пользователя, предъявляемого при входе в систему. Некоторые из этих идентификаторов могут изменяться во время выполнения процесса согласно политике, реализуемой НАЙС ОС Z.

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

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

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

- списки управления доступом;
- реализацию доменов и типов;
- журналируемая файловая система (ext4);
- подключаемые модули аутентификации (PAM);
- специализированная система аудита, которая позволяет учитывать критичные события безопасности и предоставляет административному пользователю инструментальные средства конфигурирования системы аудита и оценки записей аудита;
- базовые функции проверки комплекта оборудования позволяют по требованию административного пользователя проверять, правильно ли обеспечиваются основные функции безопасности аппаратных средств, на которые полагается объект оценки (ОО).

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

Состав НАЙС ОС Z

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

В структуре НАЙС ОС Z можно выделить следующие функциональные элементы:

- ядро ОС;
- системные библиотеки;
- встроенные средства защиты информации (КСЗ);
- системные приложения;
- программные серверы;
- прочие серверные программы;
- интерактивные рабочие среды и командные интерпретаторы;
- прочие системные приложения. Комплекс встроенных средств защиты информации является принадлежностью операционной среды НАЙС ОС Z и неотъемлемой частью ядра ОС и системных библиотек.

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

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

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

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

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

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

Интерактивные рабочие среды (ИРС) — программы (пакеты программ), предназначенные для работы пользователя в НАЙС ОС Z и предоставляющие ему удобный интерфейс для общения с ней.

Командные рабочие среды включают в свой состав командные интерпретаторы. Командные интерпретаторы — специальные программы (терминалы), предназначенные для выполнения различных команд, подаваемых пользователем при работе с НАЙС ОС Z.

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

В состав НАЙС ОС Z включены следующие дополнительные системные приложения:

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

Документация в составе

В составе НАЙС ОС Z представлена следующая документация:

- Электронная, контекстно-зависимая справочная система;

- Электронные справочники (man).

Требования к персоналу (администратору)

Администратор НАЙС ОС Z должен иметь:

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

Обязанности пользователей, определяемые предположениями безопасности

Доступ администраторов к НАЙС ОС Z должен осуществляться только из санкционированных точек доступа – рабочих мест, размещённых в контролируемой зоне, оборудованной средствами и системами физической защиты и охраны (контроля и наблюдения) и исключающей возможность бесконтрольного пребывания посторонних лиц. Для предотвращения несанкционированного доступа к системным компонентам пользователей НАЙС ОС Z администраторы обязаны предотвращать и выявлять установку и запуск встроенных программ отладки. Администраторы обязаны производить установку только штатных программных средств, не позволяющих осуществить несанкционированную модификацию ОО. При взаимодействии с внешними информационными системами администраторы, при помощи средств НАЙС ОС Z, должны осуществлять настройку взаимодействия только с доверенными системами, параметры безопасности (ПБ) которых скоординированы с ПБ рассматриваемой НАЙС ОС Z. При возникновении сбоев и отказов СВТ или НАЙС ОС Z администраторы обязаны предпринимать меры, направленные на восстановление безопасного состояния программного и аппаратного обеспечения НАЙС ОС Z в соответствии с настоящим руководством. Установка, конфигурирование и управление НАЙС ОС Z должны осуществляться администратором

НАЙС ОС Z в соответствии с настоящим руководством. Самостоятельные действия по установке, конфигурированию и управлению пользователям не доступны и ограничены правилами разграничения доступа НАЙС ОС Z.

Порядок обеспечения среды функционирования НАЙС ОС Z

Администраторы должны использовать функции, предоставляемые НАЙС ОС Z, в рамках выполнения своих должностных обязанностей, определенных в должностной инструкции соответствующих категорий пользователей. Администраторы обязаны производить настройку оборудования СВТ и предотвращать несанкционированную физическую модификацию аппаратного обеспечения, на котором выполняется НАЙС ОС Z. Права пользователей для получения доступа и выполнения обработки информации в НАЙС ОС Z основываются на одной или более ролях и назначаются администратором НАЙС ОС Z. Роли пользователей в НАЙС ОС Z отражают производственную функцию, обязанности, квалификацию и/или компетентность пользователей в рамках организации. По всем вопросам администрирования НАЙС ОС Z пользователь обязан обращаться к администраторам НАЙС ОС Z, которые являются компетентными, хорошо обученными и заслуживающими доверия. Предполагается наличие (одного или более) компетентных лиц (администраторов), которые назначаются для управления безопасностью НАЙС ОС Z и информации в ней.

Они несут ответственность за следующие функции:

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

Кроме того, эти лица (в качестве владельцев всех корпоративных данных), наряду с владельцами объекта, должны иметь возможность назначать и отменять права доступа ролей к объектам. Пользователи, в соответствии с назначенными в НАЙС ОС Z полномочиями и ролями, имеют права создавать новые объекты данных, владельцами которых они становятся. Персонал, ответственный за выполнение администрирования НАЙС ОС Z, должен пройти проверку на благонадёжность и в своей деятельности должен руководствоваться документацией на НАЙС ОС Z.

Уполномоченные пользователи обладают необходимым разрешением на доступ в НАЙС ОС Z, по крайней мере, к части информации, управляемой НАЙС ОС Z, и согласованно действуют в благоприятной среде. Администраторы в обязательном порядке должны быть ознакомлены с настоящим руководством и должны быть обучены применению функциональных возможностей безопасности, предоставляемых операционной системой. Администраторы должны выполнять группы задач, связанных со своими служебными полномочиями, в безопасной ИТ-среде с применением полного управления своими данными. Администраторы должны осуществлять регулярный контроль полноты и достаточности мер по обеспечению информационной безопасности на объектах, использующих НАЙС ОС Z.

Поддерживаемые файловые системы

В дистрибутиве поддерживаются следующие файловые системы (ФС):

- журналируемая файловая система ext2;
- журналируемая файловая система ext3;
- журналируемая файловая система ext4;
- журналируемая файловая система xfs;
- журналируемая файловая система btrfs;
- файловая система ISO 9660 для накопителей для оптических магнитных дисков.

Общие принципы работы НАЙС ОС Z

Работа с операционной средой заключается во вводе определенных команд (запросов) к операционной среде и получению на них ответов в виде текстового отображения. Диалог с ОС осуществляется посредством командных интерпретаторов с системных библиотек. Каждая системная библиотека представляет собой набор программ, динамически вызываемых операционной системой. Для удобства пользователей при работе с командными интерпретаторами используются интерактивные рабочие среды (ИРС), предоставляющие пользователю удобный интерфейс для работы с ОС. В самом центре ОС НАЙС ОС Z находится управляющая программа, называемая ядром. Ядро взаимодействует с компьютером и периферией (дисками, принтерами и т.д.), распределяет ресурсы и выполняет фоновое планирование заданий. Другими словами, ядро ОС изолирует пользователя от сложностей аппаратуры компьютера, командный интерпретатор от ядра, а ИРС от командного интерпретатора.

Процессы и файлы НАЙС ОС Z

НАЙС ОС Z является многопользовательской интегрированной системой. Это значит, что она разработана с расчётом на одновременную работу нескольких пользователей. Пользователь может либо сам работать в системе, выполняя некоторую последовательность команд, либо от его имени могут выполняться прикладные процессы. Пользователь взаимодействует с системой через командный интерпретатор, который представляет собой, как было сказано выше, прикладную программу, которая принимает от пользователя команды или набор команд и транслирует их в системные вызовы к ядру системы. Интерпретатор позволяет пользователю просматривать файлы, передвигаться по дереву файловой системы, запускать прикладные процессы. Все командные интерпретаторы имеют развитый командный язык и позволяют писать достаточно сложные программы, упрощающие процесс администрирования системы и работы с ней.

Процессы функционирования НАЙС ОС Z

Все программы, которые выполняются в текущий момент времени, называются процессами. Процессы можно разделить на два основных класса: системные процессы и пользовательские процессы. Системные процессы — программы, решающие внутренние задачи НАЙС ОС Z, например, организацию виртуальной памяти на диске или предоставляющие пользователям те или иные сервисы (процессы-службы).

Пользовательские процессы — процессы, запускаемые пользователем из командного интерпретатора для решения задач пользователя или управления системными процессами. Фоновый режим работы процесса — режим, когда программа может работать без взаимодействия с пользователем. В случае необходимости интерактивной работы с пользователем (в общем случае) процесс будет «остановлен» ядром и работа его продолжится только после переведения его в «нормальный» режим работы.

Файловая система

НАЙС ОС Z В ОС использована файловая система, которая является единым деревом. Корень этого дерева — каталог, называемый root (рут), и обозначаемый /. Части дерева файловой системы могут физически располагаться в разных разделах разных дисков или вообще на других компьютерах, — для пользователя это прозрачно. Процесс присоединения файловой системы раздела к дереву называется монтированием, удаление — размонтированием. Например, файловая система CD-ROM в НАЙС ОС Z монтируется по умолчанию в каталог /mnt/cdrom (путь в НАЙС ОС Z обозначается с использованием /, а не \, как в DOS/Windows). Текущий каталог обозначается ./.

Организация файловой структуры

Система домашних каталогов пользователей помогает организовывать безопасную работу пользователей в многопользовательской системе. Вне

своего домашнего каталога пользователь обладает минимальными правами (обычно чтение и выполнение файлов) и не может нанести ущерб системе, например, удалив или изменив файл. Кроме файлов, созданных пользователем, в его домашнем каталоге обычно содержатся персональные конфигурационные файлы некоторых программ. Маршрут (путь) — это последовательность имён каталогов, представляющий собой путь в файловой системе к данному файлу, где каждое следующее имя отделяется от предыдущего наклонной чертой (слэшем). Если название маршрута начинается со слэша, то путь в искомый файл начинается от корневого каталога всего дерева системы. В обратном случае, если название маршрута начинается непосредственно с имени файла, то путь к искомому файлу должен начинаться от текущего каталога (рабочего каталога). Имя файла может содержать любые символы за исключением косой черты (/). Однако следует избегать применения в именах файлов большинства знаков препинания и непечатаемых символов. При выборе имен файлов рекомендуем ограничиться следующими символам:

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

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

Имена дисков и разделов

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

/dev/sda — первый диск;

/dev/sdb — второй диск и т.д.

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

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

Командные оболочки (интерпретаторы)

Как было сказано выше, для управления ОС используются командные интерпретаторы (shell). Зайдя в систему, пользователь увидит приглашение — строку, содержащую символ $ (далее этот символ будет обозначать командную строку). Программа ожидает ввода команд. Роль командного интерпретатора — передавать команды операционной системе.

При помощи командных интерпретаторов можно писать небольшие программы — сценарии (скрипты). В НАЙС ОС Z используется командная оболочка bash (Bourne Again Shell). Она ведёт историю команд и предоставляет возможность их редактирования.

Для проверки используемой оболочки необходимо выполнить команду:

```
echo $SHELL
```

У каждой командной оболочки свой синтаксис команд. В НАЙС ОС Z рекомендуется использовать командную оболочку **bash**. Дальнейшее описание и примеры будут приведены с использованием данной командной оболочки.

Командная оболочка bash

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

Ctrl+A — перейти на начало строки.

Ctrl+U — удалить текущую строку.

Ctrl+C — остановить текущую задачу.

Можно использовать «;», для того чтобы ввести несколько команд одной строкой.

Клавиши «вверх» и «вниз», позволяют перемещаться по истории команд. Для того чтобы найти конкретную команду в списке набранных, не пролистывая всю историю, нужно набрать: Ctrl + R

Команды, присутствующие в истории, отображаются в списке пронумерованными. Для того, чтобы запустить конкретную команду, необходимо выполнить:

! <номер_команды>

если будет введено:

!!

запустится последняя из набранных команд. Иногда имена программ и команд слишком длинны, но bash сам может завершать имена.

Нажав клавишу TAB, можно завершить имя команды, программы или каталога.

Предположим, необходимо использовать программу декомпрессии bunzip2. Для этого нужно выполнить: bu затем нажать TAB. Если ничего не происходит, то, вероятно, существует несколько возможных вариантов завершения команды.

Нажав клавишу TAB ещё раз, можно получить список имён, начинающихся с «bu». Например: bu buildhash builtin bunzip2 Если далее добавить «n» (bunzip2 — это единственное имя, третьей буквой которого является «n»), а затем нажать клавишу TAB, оболочка дополнит имя и остаётся лишь нажать Enter, чтобы запустить команду.

Заметим, что программу, вызываемую из командной строки, bash ищет в каталогах, определяемых в системной переменной PATH.

По умолчанию в этот перечень каталогов не входит текущий каталог, обозначаемый ./ (точка слэш). Поэтому для запуска программы prog из текущего каталога, необходимо выполнить команду: ./prog

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

Команда man

Для получения более подробной информации используется команда man. Например:

```
man ls
```

Команда su

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

Команда cd

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

```
cd /usr/bin/
```

Некоторые варианты команды:

- cd .. — позволяет сделать текущим родительский каталог;
- cd - — позволяет вернуться в предыдущий каталог;
- cd <без_параметров> — переводит в домашний каталог.

Команда ls

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

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

Команда rm

Команда rm используется для удаления файлов. rm <имя_файла> У данной программы существует ряд параметров.

Самые часто используемые:

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

Команды mkdir и rmdir

Команда mkdir позволяет создать каталог, тогда как rmdir удаляет каталог, при условии, что он пуст.

mkdir <имя_каталога>

rmdir <имя_каталога>

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

Команда less

Команда less позволяет постранично просматривать текст.

less <имя_файла>

Для выхода нужно нажать q.

Команда grep

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

grep <шаблон_поиска> <файл>

Команда ps

Команда ps отображает список текущих процессов. Колонка команд указывает имя процесса, колонка PID (идентификатор процесса) — номер процесса (этот номер используется для операций с процессом, например, чтобы принудительно завершить его командой kill).

ps <аргументы>

Аргументы:

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

Команда kill

Команда kill используется для принудительного завершения процесса, если программа перестала отвечать или зависла.

```
kill <PID_номер>
```

Иногда необходимо использовать команду вида kill -9 (когда обычная команда kill не даёт желательного эффекта). Номер PID выясняется при помощи команды ps.

Команда cat

Команда cat — утилита, выводящая последовательно указанные файлы (или устройства), таким образом объединяя их в единый поток.

Если вместо имени файла указывается «-», то читается стандартный ввод.

```
cat <имя_файла>
```

Команда sort

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

```
sort <опции> <файл>
```

Использование многозадачности

НАЙС ОС Z — это многозадачная система. Продемонстрируем на двух примерах, как это можно использовать. Первый пример — запуск программы в фоновом режиме. Для запуска программы в фоновом режиме необходимо набрать & после имени программы. После этого оболочка даёт возможность запускать другие приложения. Пользователь должен быть внимательным, т.к. некоторые программы интерактивны, и их запуск в фоновом режиме не имеет смысла (подобные программы просто остановятся, будучи запущенными в фоновом режиме).

Для того чтобы вернуть их в обычный режим, следует выполнить:

fg <имя_программы>

Второй метод представляет собой запуск нескольких независимых сеансов. В консоли нажмите Ctrl+Alt и одну из клавиш, находящихся в интервале от F1 до F6.

На экране появится новое приглашение системы, и можно открыть новый сеанс. Этот метод также позволяет работать на другой консоли, если консоль, которая была использована до этого, не отвечает, или необходимо остановить зависшую программу.

Режимы работы ОС

Диагностические режимы работы

С точки зрения функционирования ОС можно выделить 3 режима: нормальный (штатный), аварийный и режим восстановления. Обычно ОС нормально функционирует и выполняет возложенные на неё функции в нормальном режиме. В этом режиме пользователь получает ожидаемый отклик на свои действия от ОС (в этом разделе нормальный режим рассмотрен не будет, ему посвящены остальные разделы руководства). Однако в ряде случаев, если в работе системы возникают проблемы, ОС может выполнить загрузку в режиме восстановления или аварийном режиме с целью диагностики и исправления проблем.

Режим восстановления

Режим восстановления позволяет загрузить минимальное окружение ОС с дистрибутивного носителя вместо загрузки с жёсткого диска. Этот режим предусмотрен для восстановления в случае сбоя. В штатном режиме ОС использует файлы на жёстком диске компьютера для запуска программ, хранения информации и прочих операций. Однако не исключены ситуации, когда не получается полностью запустить ОС, чтобы иметь возможность обращения к файлам на жёстком диске. В режиме восстановления можно получить доступ к файлам, даже если не удалось запустить ОС с этого диска.

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

1. Начните загрузку ПК и дождитесь появления меню GRUB. Нажмите клавишу E для редактирования параметров загрузки.

2. Добавьте в конец строки, начинающейся с linux/boot/vmlinuz ..., следующую запись: systemd.unit=rescue.target Используйте сочетание клавиш Ctrl+A (для перехода в начало строки) и Ctrl+E (для перехода в конец строки).

3. Нажмите Ctrl+X для загрузки ОС с указанными параметрами. Загрузив систему, необходимо ответить на несколько простых вопросов, в

частности, выбрать используемый язык и расположение корректного образа восстановления. Если выбран образ восстановления, который не требует подключения к сети, будет предложено определить, есть ли необходимость установления сетевого подключения. Подключение к сети рекомендуется, если, например, нужно скопировать файлы на другой компьютер или установить какие-либо RPM-пакеты с общего сетевого ресурса. В режиме восстановления будет выполнена попытка найти установку ОС и подключить ее в /mnt/sysimage. После этого можно вносить необходимые изменения. Нажмите «Продолжить».

Также можно подключить файловые системы в режиме чтения вместо чтения-записи. Если это не удалось, нажмите кнопку «Пропустить» для перехода в командную оболочку. При выборе «Продолжить» система попытается подключить файловую систему в /mnt/sysimage. Если смонтировать раздел не удастся, появится соответствующее сообщение. При выборе варианта «только для чтения» будет предпринята попытка подключения файловой системы в /mnt/sysimage/ в режиме чтения.

Если будет выбран пункт «Пропустить», файловая система не будет подключена. Выберите «Пропустить», если считаете, что файловая система повреждена. Как только система загрузится в режиме восстановления, на виртуальных консолях появится приглашение (используйте Ctrl+Alt+Fx для перехода в нужную консоль).

Даже если файловая система подключена, в режиме восстановления корневым разделом по умолчанию становится временный раздел, а не тот, что используется при работе в обычном режиме.

Если файловая система была смонтирована успешно, можно сменить корневой раздел окружения режима восстановления на корневой раздел вашей файловой системы, выполнив команду: chroot /mnt/sysimage Это может пригодиться для выполнения команд, требующих, чтобы корневой раздел системы был подключен как / (таких как rpm).

Чтобы выйти из окружения chroot, выполните команду exit. При выборе «Пропустить» можно попытаться смонтировать раздел или логический том LVM2 вручную в режиме восстановления, создав каталог, к примеру, с именем /foo и выполнив следующую команду:

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

В приведённой выше команде /foo — созданный каталог, а /dev/mapper/ VolGroup00- LogVol02 — логический том LVM2, который необходимо смонтировать.

Если раздел имеет тип ext2 или ext3, замените ext4 на ext2 или ext3. Если вы не знаете названий всех физических разделов, для их просмотра используйте команду:

```
fdisk -l
```

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

```
pvdisplay vgdisplay lvdisplay
```

В строке приглашения можно выполнить множество полезных команд, включая следующие: ssh, scp и ping – если сеть запущена; dump и restore – если вы используете ленточные накопители; parted и fdisk – для управления разделами; rpm – для установки и обновления программного обеспечения; vi – для редактирования текстовых файлов.

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

Аварийный режим

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

1. Начните загрузку ПК и дождитесь появления меню GRUB. Нажмите клавишу E для редактирования параметров загрузки.

2. Добавьте в конец строки, начинающейся с linux/boot/vmlinuz ..., следующую запись: systemd.unit=emergency.target Используйте сочетание клавиш Ctrl+A (для перехода в начало строки) и Ctrl+E (для перехода в конец строки).

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

Управление ПО

Введение: пакеты, зависимости и репозитории В НАЙС ОС Z огромное число общих ресурсов, которыми пользуются сразу несколько программ: разделяемых библиотек, содержащих стандартные функции, исполняемых файлов, сценариев и стандартных утилит и т.д.

Удаление или изменение версии одного из составляющих систему компонентов может повлечь неработоспособность других, связанных с ним компонентов, или даже вывести из строя всю систему.

В контексте системного администрирования проблемы такого рода называют нарушением целостности системы. Задача администратора — обеспечить наличие в системе согласованных версий всех необходимых программных компонентов (обеспечение целостности системы).

Для установки, удаления и обновления программ и поддержания целостности системы используются менеджеры пакетов. С точки зрения менеджера пакетов программное обеспечение представляет собой набор компонентов — программных пакетов.

Такие компоненты содержат в себе набор исполняемых программ и вспомогательных файлов, необходимых для корректной работы программного обеспечения.

Менеджеры пакетов облегчают установку программ: они позволяют проверить наличие необходимых для работы устанавливаемой программы компонент подходящей версии непосредственно в момент установки, а также производят необходимые процедуры для регистрации программы во всех операционных средах пользователя: сразу после установки программа может быть доступна пользователю из командной строки и — если это предусмотрено — появляется в меню всех графических оболочек.

Важно! Благодаря менеджерам пакетов, пользователю обычно не требуется непосредственно обращаться к установочным процедурам отдельных программ или непосредственно работать с каталогами, в которых установлены исполняемые файлы и компоненты программ (обычно это /usr/bin, /usr/share/<имя_пакета>) — всю работу делает

менеджер пакетов. Поэтому установку, обновление и удаление программ обычно называют управлением пакетами.

Часто компоненты, используемые различными программами, выделяют в отдельные пакеты и помечают, что для работы ПО, предоставляемого пакетом A, необходимо установить пакет B. В таком случае говорят, что пакет A зависит от пакета B или что между пакетами A и B существует зависимость.

Отслеживание зависимостей между такими пакетами представляет собой серьёзную задачу — некоторые компоненты могут быть взаимозаменяемыми: может обнаружиться несколько пакетов, предлагающих затребованный ресурс.

Задача контроля целостности и непротиворечивости установленного в системе ПО ещё сложнее. Представим, что некие программы A и B требуют наличия в системе компоненты C версии 1.0. Обновление версии пакета A, требующее обновления компоненты C до новой, использующей новый интерфейс доступа, версии (скажем, до версии 2.0), влечёт за собой обязательное обновление и программы B. Однако менеджеры пакетов оказались неспособны предотвратить все возможные коллизии при установке или удалении программ, а тем более эффективно устранить нарушения целостности системы.

Особенно сильно этот недостаток сказывается при обновлении систем из централизованного репозитория пакетов, в котором последние могут непрерывно обновляться, дробиться на более мелкие и т.п. Этот недостаток и стимулировал создание систем управления программными пакетами и поддержания целостности системы. Для автоматизации этого процесса в НАЙС ОС Z применяется система управления программными пакетами tdnf и ее аналог Ttdnf.

Основы работы с RPM

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

- унифицированная работа с разными пакетами;
- отслеживание зависимостей между пакетами выполняется автоматически;
- непротиворечивость между разными пакетами.
- Если вызвать rpm без параметров, то он покажет краткий список ключей. Обычно же формат вызова rpm следующий:
    - rpm -<ключ_режима> <дополнительные_ключи> <параметры>

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

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

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

Команда rpm -q позволяет получать следующую информацию о пакете:

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

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

Команда rpm -qi (info) выдаёт сводку информации о пакете — название, версию, объем и т.д., плюс краткую аннотацию. Для получения списка файлов используется ключ -l (list). Поскольку некоторые пакеты содержат очень большое количество файлов, то стоит отправлять вывод rpm -ql команде less. Для получения «полной» информации о пакете (аннотации и списка файлов) можно указать ключи -i и -l одновременно. Часто

возникает необходимость узнать, какому пакету принадлежит какой-то файл (например, чтобы знать, где искать к нему документацию). Для этого можно воспользоваться ключом -f (file). При этом надо указывать полное имя файла — с директорией. Кроме того, если к файлу есть «несколько путей» (из-за символьных ссылок на директории), то следует указывать «основной» (обычно тот, который без символьных ссылок), иначе rpm не сможет предоставить ответ.

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

Назначение tdnf

Фактически, tdnf представляет собой оболочку для rpm, обеспечивающую работу с репозиториями. Утилита tdnf — это менеджер пакетов, который умеет запрашивать информацию о пакетах, получать пакеты из репозиториев, устанавливать и удалять их, используя автоматическое разрешение зависимостей, а также обновлять целиком систему до последних версий пакетов.

tdnf выполняет автоматическое разрешение зависимостей для пакетов, которые обновляются, устанавливаются или удаляются, и, таким образом, позволяет автоматически определять, получать и устанавливать все доступные по зависимостям пакеты.

Для tdnf можно настроить новые, дополнительные репозитории, или, по-другому, источники пакетов, кроме того, для него доступны многие дополнения, которые улучшают и расширяют его возможности. tdnf позволяет выполнять многие из задач, которые выполняет RPM; кроме того, многие из опций командной строки у него также подобны опциям RPM. Утилита tdnf обеспечивает простое и легкое управление пакетами на одной машине или же на группе машин.

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

Источники программ (репозитории)

Репозитории, с которыми работает tdnf, отличаются от обычного набора пакетов наличием мета информации — индексов пакетов, содержащихся в репозитории, и сведений о них. Поэтому, чтобы получить всю информацию о репозитории, tdnf достаточно получить его индексы. tdnf может работать с любым количеством репозиториев одновременно, формируя единую информационную базу обо всех содержащихся в них пакетах. При установке пакетов tdnf обращает внимание только на название пакета, его версию и зависимости, а расположение в том или ином репозитории не имеет значения. Если потребуется, tdnf в рамках одной операции установки группы пакетов может пользоваться несколькими репозиториями.

tdnf позволяет взаимодействовать с репозиторием с помощью различных протоколов доступа. Наиболее популярные — HTTP и FTP, однако существуют и некоторые дополнительные методы. Для того чтобы tdnf мог использовать тот или иной репозиторий, информацию о нем необходимо поместить в папку /etc/yum.repos.d/.

После редактирования списка репозиториев в sources.list необходимо обновить локальную базу данных tdnf о доступных пакетах. Сделать это можно командой: tdnf update При выборе пакетов для установки tdnf руководствуется всеми доступными репозиториями вне зависимости от способа доступа к ним.

Так, если в репозитории, доступном по сети Интернет, обнаружена более новая версия программы, чем на компакт-диске, то tdnf начнёт загружать данный пакет из сети Интернет.

Для создания репозитория необходимо выполнить следующий ряд действий:

• Установить пакет createrepo

```
tdnf install createrepo
```

• Скопировать все пакеты в один каталог, например: /mnt/local_repo

• Запустить команду createrepo --database, указав этот каталог: createrepo --database /mnt/local_repo

Поиск пакетов

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

Команда вида:

```
tdnf search <подстрока>
```

позволяет найти все пакеты, в именах или описании которых присутствует указанная подстрока. Для того чтобы подробнее узнать о каждом из найденных пакетов и прочитать его описание, можно воспользоваться командой tdnf info, которая покажет информацию о пакете из репозитория: tdnf info gdm-libs

Установка или обновление пакета

Установка пакета с помощью tdnf выполняется командой:

```
tdnf install <имя_пакета>
```

tdnf позволяет устанавливать в систему пакеты, требующие для работы другие, пока ещё не установленные. В этом случае он определяет, какие

пакеты необходимо установить, и устанавливает их, пользуясь всеми доступными репозиториями.

Команда

```
tdnf install <имя_пакета>
```

используется также для обновления уже установленного пакета или группы пакетов. В этом случае tdnf дополнительно проверяет, не обновилась ли версия пакета в репозитории по сравнению с установленным в системе. При помощи tdnf можно установить и отдельный бинарный rpm-пакет, не входящий ни в один из репозиториев (например, полученный из Интернет).

Для этого достаточно выполнить команду:

```
tdnf install <путь_к_файлу.rpm>
```

При этом tdnf проведёт стандартную процедуру проверки зависимостей и конфликтов с уже установленными пакетами. Иногда, в результате операций с пакетами без использования tdnf, целостность системы нарушается, и tdnf отказывается выполнять операции установки, удаления или обновления. В этом случае необходимо повторить операцию, задав опцию -f, заставляющую tdnf исправить нарушенные зависимости, удалить или заменить конфликтующие пакеты. В этом случае необходимо внимательно следить за сообщениями, выдаваемыми tdnf. Любые действия в этом режиме обязательно требуют подтверждения со стороны пользователя.

Удаление установленного пакета

Для удаления пакета используется команда вида:

```
tdnf remove <имя_пакета>
```

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

Обновление всех установленных пакетов

Для проверки доступных обновлений всех системных пакетов необходимо выполнить:

```
tdnf check-update
```

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

```
tdnf upgrade
```

Система безопасности

Программа sudo

sudo — это программа, разработанная в помощь системному администратору и позволяющая делегировать те или иные привилегированные ресурсы пользователям с ведением протокола работы.
Основная идея — предоставить пользователям как можно меньше прав, но при этом ровно столько, сколько необходимо для решения поставленных задач.
Команда sudo предоставляет возможность пользователям выполнять команды от имени root либо других пользователей.
Правила, используемые sudo для принятия решения о предоставлении доступа, находятся в файле /etc/sudoers. Кроме того, пример правил, предоставляющих пользователям, являющимися членами группы rpm, возможность устанавливать, обновлять и удалять пакеты в системе, приведён в файле /usr/share/doc/sudo-<версия>/sample.sudoers. Для редактирования файла /etc/sudoers следует использовать программу visudo, которая проверяет синтаксис и тем самым позволяет избежать ошибок в правилах. В большинстве случаев грамотная настройка sudo делает работу от имени суперпользователя ненужной.

Брандмауэр iptables

Дизайн НАЙС ОС Z акцентирует внимание на безопасности. В минимальной и полной версиях НАЙС ОС Z, политика безопасности по умолчанию включает брандмауэр и отбрасывает пакеты с внешних интерфейсов и приложений. В результате вам может потребоваться добавить правила в iptables для разрешения пересылки, разрешения протоколов, таких как HTTP, и открытия портов. Вы должны настроить брандмауэр под свои приложения и требования. По умолчанию все порты, кроме 22 закрыты. Политики установлены в DROP. Для просмотра правил файрвола, введите команду:

```
iptables —list
```

Chain INPUT (policy DROP)

target prot opt source destination

ACCEPT all -- anywhere anywhere

ACCEPT all -- anywhere anywhere ctstate RELATED,ESTABLISHED

ACCEPT tcp -- anywhere anywhere tcp dpt:ssh

Chain FORWARD (policy DROP)

target prot opt source destination

Chain OUTPUT (policy DROP)

target prot opt source destination

ACCEPT all -- anywhere anywhere

Права доступа к файлам и каталогам в НАЙС ОС Z

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

В основе механизмов разграничения доступа лежат имена пользователей и имена групп пользователей. В НАЙС ОС Z каждый пользователь имеет уникальное имя, под которым он входит в систему (логируется).

Кроме того, в системе создаётся некоторое число групп пользователей, причём каждый пользователь может быть включён в одну или несколько

групп. Создаёт и удаляет группы суперпользователь, он же может изменять состав участников той или иной группы.

Члены разных групп могут иметь разные права по доступу к файлам, например, группа администраторов может иметь больше прав, чем группа программистов. В индексном дескрипторе каждого файла записаны имя так называемого владельца файла и группы, которая имеет права на этот файл. Первоначально, при создании файла его владельцем объявляется тот пользователь, который этот файл создал. Точнее — тот пользователь, от чьего имени запущен процесс, создающий файл.

Группа тоже назначается при создании файла — по идентификатору группы процесса, создающего файл. Владельца и группу файла можно поменять в ходе дальнейшей работы с помощью команд chown и chgrp. Выполним команду ls -l. Но зададим ей в качестве дополнительного параметра имя конкретного файла, например, файла, задающего саму команду ls.

```
ls -l /bin/ls

-rwxr-xr-x 1 root root 170584 мар 20 09:39 /bin/ls
```

В данном случае владельцем файла является пользователь root и группа root. Но нас в выводе этой команды больше интересует первое поле, определяющее тип файла и права доступа к файлу. Это поле в приведённом примере представлено цепочкой символов -rwxr-xr-x.

Эти символы можно условно разделить на 4 группы. Первая группа, состоящая из единственного символа, определяет тип файла. Этот символ в соответствии с возможными типами файлов может принимать такие значения:

- - — обычный файл;
- d — каталог;
- b — файл блочного устройства;

- c — файл символьного устройства;
- s — доменное гнездо (socket);
- p — именованный канал (pipe);
- l — символическая ссылка (link).

Далее следуют три группы по три символа, которые и определяют права доступа к файлу соответственно для владельца файла, для группы пользователей, которая сопоставлена данному файлу, и для всех остальных пользователей системы. В нашем примере права доступа для владельца определены как rwx, что означает, что владелец (root) имеет право читать файл (r), производить запись в этот файл (w) и запускать файл на выполнение (x).

Замена любого из этих символов прочерком будет означать, что пользователь лишается соответствующего права. В том же примере мы видим, что все остальные пользователи (включая и тех, которые вошли в группу root) лишены права записи в этот файл, т. е. не могут файл редактировать и вообще как-либо изменять.

Права доступа и информация о типе файла хранятся в индексных дескрипторах в отдельной структуре, состоящей из двух байтов, т. е. из 16 бит. Четыре бита из этих 16-ти отведены для кодированной записи о типе файла. Следующие три бита задают особые свойства исполняемых файлов. И, наконец, оставшиеся 9 бит определяют права доступа к файлу. Эти 9 бит разделяются на 3 группы по три бита.

Первые три бита задают права пользователя, следующие три бита — права группы, последние 3 бита определяют права всех остальных пользователей (т. е. всех пользователей, за исключением владельца файла и группы файла). При этом, если соответствующий бит имеет значение «1», то право предоставляется, а если он равен «0», то право не

предоставляется. В символьной форме записи прав единица заменяется соответствующим символом (r, w или x), а 0 представляется прочерком.

Право на чтение (r) файла означает, что пользователь может просматривать содержимое файла с помощью различных команд просмотра, например, командой more или с помощью любого текстового редактора. Но, отредактировав содержимое файла в текстовом редакторе, вы не сможете сохранить изменения в файле на диске, если не имеете права на запись (w) в этот файл. Право на выполнение (x) означает, что вы можете загрузить файл в память и попытаться запустить его на выполнение как исполняемую программу. K

онечно, если в действительности файл не является программой (или скриптом shell), то запустить этот файл на выполнение не удастся, но, с другой стороны, даже если файл действительно является программой, но право на выполнение для него не установлено, то он тоже не запустится.

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

```
drwxr-xr-x  4 root root      4096 мар 20 13:34 boot
```

Естественно, что по отношению к каталогам трактовка понятий «право на чтение», «право на запись» и «право на выполнение» несколько изменяется.

Право на чтение по отношению к каталогам легко понять, если вспомнить, что каталог — это просто файл, содержащий список файлов в данном каталоге. Следовательно, если вы имеете право на чтение каталога, то вы можете просматривать его содержимое (этот самый список файлов в каталоге).

Право на запись тоже понятно — имея такое право, вы сможете создавать и удалять файлы в этом каталоге, т. е. просто добавлять в каталог или удалять из него запись, содержащую имя какого-либо файла и соответствующие ссылки.

Право на выполнение в данном случае означает право переходить в этот каталог. Если вы, как владелец, хотите дать доступ другим пользователям на просмотр какого-то файла в своём каталоге, вы должны дать им право доступа в каталог, т. е. дать им «право на выполнение каталога».

Более того, надо дать пользователю право на выполнение для всех каталогов, стоящих в дереве выше данного каталога. Поэтому в принципе для всех каталогов по умолчанию устанавливается право на выполнение как для владельца и группы, так и для всех остальных пользователей. И, уж если вы хотите закрыть доступ в каталог, то лишите всех пользователей (включая группу) права входить в этот каталог.

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

Если вы имеете право доступа в каком-то из подкаталогов этого каталога, то вы можете перейти в него (командой cd), но, как говорится «вслепую», по памяти, потому что списка файлов и подкаталогов текущего каталога вы не увидите. Алгоритм проверки прав пользователя при обращении к файлу можно описать следующим образом. Система вначале проверяет, совпадает ли имя пользователя с именем владельца файла. Если эти имена совпадают (т. е. владелец обращается к своему файлу), то проверяется, имеет ли владелец соответствующее право доступа: на чтение, на запись или на выполнение (не удивляйтесь, суперпользователь может лишить некоторых прав и владельца файла). Если право такое есть, то соответствующая операция разрешается.

Если же нужного права владелец не имеет, то проверка прав, предоставляемых через группу или через группу атрибутов доступа для остальных пользователей, уже даже не проверяются, а пользователю выдаётся сообщение о невозможности выполнения затребованного действия. Если имя пользователя, обращающегося к файлу, не совпадает с именем владельца, то система проверяет, принадлежит ли владелец к группе, которая сопоставлена данному файлу (далее будем просто называть ее группой файла). Если принадлежит, то для определения возможности доступа к файлу используются атрибуты, относящиеся к группе, а на атрибуты для владельца и всех остальных пользователей внимания не обращается. Если же пользователь не является владельцем файла и не входит в группу файла, то его права определяются атрибутами для остальных пользователей.

Таким образом, третья группа атрибутов, определяющих права доступа к файлу, относится ко всем пользователям, кроме владельца файла и пользователей, входящих в группу файла.

chmod

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

```
chmod wXp <имя_файла>
```

где:

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

- вместо подставляется: – либо + (предоставить право); – либо – (лишить соответствующего права); – либо = (установить указанные права вместо имеющихся).

- вместо - подставляется символ, обозначающий соответствующее право: – r (чтение); – w (запись); – x (выполнение). Далее приведено несколько примеров использования команды chmod:

- Предоставление всем пользователям системы прав на выполнение данного файла:

```
chmod a+x <имя_файла>
```

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

```
chmod go-rw <имя_файла>
```

- Установка всем прав на чтение, запись и выполнение:

```
chmod ugo+rwx <имя_файла>
```

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

```
chmod a+x <имя_файла>
```

- можно записать просто:

```
chmod +x <имя_файла>
```

- Второй вариант задания команды chmod (он используется чаще) основан на цифровом представлении прав.

Для этого мы кодируем символ r цифрой 4, символ w — цифрой 2, а символ x — цифрой 1. Для того чтобы предоставить пользователям какой-либо набор прав, надо сложить соответствующие цифры. Получив, таким образом, нужные цифровые значения для владельца файла, для

группы файла и для всех остальных пользователей, задаём эти три цифры в качестве аргумента команды chmod (ставим эти цифры после имени команды перед вторым аргументом, который задаёт имя файла). Например, если надо дать все права владельцу (4+2+1=7), право на чтение и 100 запись — группе (4+2=6), и не давать никаких прав остальным, то следует дать такую команду:

chmod 760 <имя_файла>

Если вы знакомы с двоичным кодированием восьмеричных цифр, то вы поймёте, что цифры после имени команды в этой форме ее представления есть ни что иное, как восьмеричная запись тех самых 9 бит, которые задают права для владельца файла, группы файла и для всех пользователей. Выполнять смену прав доступа к файлу с помощью команды chmod может только сам владелец файла или суперпользователь. Для того чтобы иметь возможность изменить права группы, владелец должен дополнительно быть членом той группы, которой он хочет дать права на данный файл. Надо рассказать ещё о трёх возможных атрибутах файла, устанавливаемых с помощью той же команды chmod. Это те самые атрибуты для исполняемых файлов, которые в индексном дескрипторе файла в двухбайтовой структуре, определяющей права на файл, занимают позиции 5-7, сразу после кода типа файла. Первый из этих атрибутов — так называемый «бит смены идентификатора пользователя».

Смысл этого бита состоит в следующем. Обычно, когда пользователь запускает некоторую программу на выполнение, эта программа получает те же права доступа к файлам и каталогам, которые имеет пользователь, запустивший программу. Если же установлен «бит смены идентификатора пользователя», то программа получит права доступа к файлам и каталогам, которые имеет владелец файла программы (таким образом, рассматриваемый атрибут лучше называть «битом смены идентификатора владельца»). Это позволяет решать некоторые задачи, которые иначе было бы трудно выполнить.

Самый характерный пример — команда смены пароля passwd. Все пароли пользователей хранятся в файле /etc/passwd, владельцем которого является суперпользователь root. Поэтому программы, запущенные обычными пользователями, в том числе команда passwd, не могут производить запись в этот файл. А значит пользователь как бы не может менять свой собственный пароль. Но для файла /usr/bin/passwd установлен «бит смены идентификатора владельца», каковым является пользователь root. Следовательно, программа смены пароля passwd запускается с правами root и получает право записи в файл /etc/passwd (уже средствами самой программы обеспечивается то, что пользователь может изменить только одну строку в этом файле).

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

```
chmod +s <имя_файла>
```

Аналогичным образом работает «бит смены идентификатора группы». Еще один возможный атрибут исполняемого файла — это «бит сохранения задачи» или «sticky bit» (дословно — «бит прилипчивости»). Этот бит указывает системе, что после завершения программы надо сохранить ее в оперативной памяти. Удобно включить этот бит для задач, которые часто вызываются на выполнение, так как в этом случае экономится время на загрузку программы при каждом новом запуске. Этот атрибут был необходим на старых моделях компьютеров. На современных быстродействующих системах он используется редко. Если используется цифровой вариант задания атрибутов в команде chmod, то цифровое значение этих атрибутов должно предшествовать цифрам, задающим права пользователя:

```
chmod 4775 <имя_файла>
```

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

• 4 — «бит смены идентификатора пользователя»;

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

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

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

umask

umask (от англ. user file creation mode mask — маска режима создания пользовательских файлов) — функция среды POSIX, изменяющая права доступа, которые присваиваются новым файлам и директориям по умолчанию. Права доступа файлов, созданных при конкретном значении umask, вычисляются при помощи следующих побитовых операций (umask обычно устанавливается в восьмеричной системе счисления): побитовое «И» между унарным дополнением аргумента (используя побитовое «НЕ») и режимом полного доступа.

Фактически, umask указывает, какие биты следует сбросить в выставляемых правах на файл — каждый установленный бит umask

запрещает выставление соответствующего бита прав. Исключением из этого запрета является бит исполняемости, который для обычных файлов зависит от создающей программы (трансляторы ставят бит исполняемости на создаваемые файлы, другие программы — нет), а для каталогов следует общему правилу. umask 0 означает, что следует (можно) выставить все биты прав (rwxrwxrwx), umask 777 запрещает выставление любых прав. Допустим, что значение umask равняется 174, тогда каждый новый файл будет иметь права доступа 602, а каждая новая директория 603.

chown

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

ACL

ACL (Access Control List — Список Контроля Доступа) предоставляет расширенный и более гибкий механизм распределения прав файловых систем. Он предназначен для расширения прав доступа к файлам. ACL позволяет устанавливать разрешения любым пользователям или группам для различных файловых ресурсов. Чтобы включить ACL, файловая система должна быть смонтирована с опцией acl. Используйте fstab для постоянного монтирования с данной опцией. Для изменения прав ACL используйте команду setfacl. Добавить разрешения для пользователя (здесь имя пользователя или его ID):

setfacl -m «u::permissions»

Systemd – управление компонентами ОС

Systemd – менеджер системы и сервисов в операционной системе НАЙС ОС Z. Systemd реализует концепцию юнитов systemd. Юниты представлены конфигурационными файлами, размещёнными в одной из директорий:

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

Этот каталог приоритетнее каталога юнитов, созданных в рантайме. Юниты содержат информацию о системных сервисах, прослушиваемых сокетах, сохраненных снапшотах состояний системы и других объектах, относящихся к системе инициализации.

Типы юнитов systemd:

- .service – системный сервис;
- .target — группа юнитов systemd;
- .automount – точка автомонтирования файловой системы;
- .device – файл устройства, распознанного ядром;
- .mount – точка монтирования файловой системы;
- .path – файл или директория в файловой системе;
- .scope – процесс, созданный извне;
- .slice – группа иерархически организованных юнитов, управляющая системными процессами;
- .snapshot – сохранённое состояние менеджера systemd;

- .socket – сокет межпроцессного взаимодействия;
- .swap – swap-устройство или swap-файл (файл подкачки);
- .timer – таймер systemd.

Во время загрузки systemd прослушивает сокеты для всех системных сервисов, поддерживает этот тип активации и передаёт сокеты этим сервисам сразу после старта сервисов. Это позволяет systemd не только запускать сервисы параллельно, но также предоставляет возможность перезапускать сервисы без потери любых отправленных им сообщений, пока сервисы были недоступны. Соответствующий сокет остаётся доступным и все сообщения выстраиваются в очередь. Системные сервисы, использующие D–Bus для межпроцессного взаимодействия, могут быть запущены по требованию, когда клиентское приложение пытается связаться с ними.

Системные сервисы, поддерживающие активацию, основанную на устройствах, могут быть запущены, когда определённый тип оборудования подключается или становится доступным. Системные сервисы могут поддерживать этот вид активации, если изменяется состояние папки или директории. Система может сохранять состояние всех юнитов и восстанавливать предыдущее состояние системы.

Systemd отслеживает и управляет точками монтирования и автомонтирования. Агрессивная параллелизация Systemd запускает системные сервисы параллельно из-за использования активации, основанной на сокетах. В комбинации с сервисами, поддерживающими активацию по требованию, параллельная активация значительно уменьшает время загрузки системы. До активации и деактивации юнитов systemd вычисляет их зависимости, создает временную транзакцию и проверяет целостность этой транзакции. Если транзакция не целостная, systemd автоматически пытается исправить ее и удалить не требующиеся задания до формирования сообщения об ошибке. SystemD полностью поддерживает скрипты инициализации SysV, как описано в спецификации Linux Standard Base (LSB), что упрощает переход на systemd. По способу использования сервисные юниты .service напоминают скрипты

инициализации. Для просмотра, старта, остановки, перезагрузки, включения или выключения системных сервисов используется команда systemctl. При использовании systemctl указывать расширение файла не обязательно.

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

SELinux

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

Введение

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

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

Используемые библиотеки SELinux добавляют в операционную систему поддержку RBAC.

Установка

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

```
dnf install selinux-policy
```

После перезагрузки SELinux проиндексирует содержимое жёсткого диска, это может занять некоторое время. Для настройки SELinux можно использовать различные текстовые редакторы, чтобы настроить вручную его файл конфигурации /etc/selinux/config.

Утилиты

Утилита audit2allow

Утилита audit2allow создает разрешающие правила политики SELinux из файлов журналов, содержащих сообщения о запрете операций. Эта утилита сканирует журналы в поиске сообщений, появляющихся, когда система не дает разрешения на операцию. Далее утилита генерирует ряд правил, которые, будучи загруженными в политику, могли бы разрешить эти операции. Однако, данная утилита генерирует только разрешающие правила Type Enforcement (TE). Некоторые отказы в использовании разрешений могут потребовать других изменений политики. Например, добавление атрибута в определение типа, для разрешения существующего ограничения (constraint), добавления разрешающего правила для роли или модификации ограничения (constraint). В случае сомнений для диагностики можно попробовать использовать утилиту audit2why. Следует с осторожностью работать с выводом данной утилиты, убедившись, что разрешаемые операции не представляют угрозы безопасности. Обычно бывает лучше определить новый домен и/или тип или произвести другие структурные изменения.

Лучше избирательно разрешить оптимальный набор операций вместо того, чтобы вслепую применить иногда слишком широкие разрешения, рекомендованные этой утилитой. Некоторые запреты на использование

разрешений бывают не принципиальны для приложения. В таких случаях вместо использования разрешительного правила («allow» rule) лучше просто подавить журналирование этих запретов при помощи правила dontaudit.

Утилита secon

Утилита secon — позволяет просмотреть контекст SELinux для файла, программы или ввода пользователя. Просматривает часть контекста. Контекст берётся из файла, идентификатора процесса, ввода пользователя или контекста, в котором была запущена утилита secon.

Синтаксис команды имеет вид:

secon [-hVurtscmPRfLp] [CONTEXT] [--file] [--link] [—pid]

Если не было задано опций, то для того чтобы secon получил контекст из другого источника, может быть задан дополнительный аргумент CONTEXT. Если этим аргументом является знак дефиса (-), то контекст будет прочитан из стандартного ввода.

Если аргументов не было заданно, то secon будет пытаться прочитать контекст со стандартного ввода, но только в том случае, если стандартный ввод не терминал (tty). В этом случае secon будет вести себя как будто была передана опция --self.

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

Утилита audit2why

Утилита audit2why — позволяет определить из сообщения аудита SELinux причину запрета доступа. Эта утилита обрабатывает сообщения аудита SELinux, принятые со стандартного ввода, и сообщает, какой компонент политики вызвал каждый из запретов.

Если задана опция -p, то используется указанная этой опцией политика, в противном случае используется активная политика.

Есть три возможные причины запрета доступа:

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

В первом случае разрешительное TE правило может присутствовать в политике, но было отключено через булевы атрибуты. Если же разрешительного правила не было, то оно может быть создано при помощи утилиты audit2allow. Во втором случае могло произойти нарушение ограничения. Просмотрите policy/ constraints или policy/mls для того, чтобы определить искомое ограничение. Обычно проблема может быть решена добавлением атрибута типа к домену. В третьем случае была произведена попытка сменить роль, при том что не существует разрешительного правила для участвующей при этом пары ролей. Проблема может быть решена добавлением в политику разрешительного правила для этой пары ролей.

PAM

PAM или Pluggable Authentication Modules (подключаемые модули аутентификации) — это модульный подход к системе аутентификации. Они позволяют сторонним службам предоставлять модуль аутентификации для обеспечения доступа к службе для систем с поддержкой PAM.

Службы, использующие PAM для аутентификации, могут использовать их сразу же, без необходимости дополнительной пересборки. На сервере PAM могут использоваться для управления аутентификацией (как часть управления предоставления доступа). При использовании PAM сервисам нет необходимости поддерживать собственную систему аутентификации. Вместо этого они полагаются на модули PAM, доступные в системе.

Любой сервис при необходимости может использовать собственную конфигурацию PAM, хотя в большинстве случаев аутентификация выполняется одинаково во множестве сервисов. Вызывая модули PAM,

сервисы могут поддерживать двухфакторную аутентификацию «из коробки», сразу же использовать централизованные хранилища аутентификационных средств и многое другое.

PAM предоставляют гибкую модульную архитектуру для следующих сервисов:

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

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

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

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

Последним принципом, являющимся также частью названия PAM, является «подключаемая архитектура». Когда необходимо интегрировать новый бэк-энд, всё, что нужно сделать администратору — это установить

библиотеку для этого бэк-энда (большинство модулей используют один файл настроек).

Начиная с этого момента модуль становится доступен для использования приложениями.

Администраторы могут настроить аутентификацию для использования этого бэк-энда и просто перезапустить приложение. П

риложения, для которых необходимо использование PAM, линкуются с библиотекой PAM (libpam) и могут вызывать нужные функции работы с указанными выше службами.

Кроме этого, в приложении не нужно ничего реализовывать специфичного для работы с этими сервисами, так как эту задачу на себя берёт PAM. И когда пользователь захочет аутентифицироваться, скажем, в веб-приложении, то это приложение вызывает PAM (передавая ему идентификатор и, возможно, пароль или запрос) и проверяет возвращаемые данные, чтобы принять решение, аутентифицировался ли пользователь и имеет ли он доступ к приложению.

Внутренней задачей PAM является определение, где необходимо аутентифицировать пользователя. Сильной стороной PAM является то, что любой желающий может создать модули PAM для интеграции с любым поддерживающим PAM сервисом или приложением. Если какая-нибудь компания выпускает новый сервис для аутентификации, всё, что нужно будет сделать, — это предоставить для взаимодействия с этим сервисом модуль PAM, после чего любое использующее PAM приложение сможет незамедлительно работать с этим сервисом: нет необходимости что-то пересобирать или улучшать. Другой важной особенностью PAM является то, что они поддерживают объединение в цепочки нескольких модулей. Пример конфигурационного файла PAM для некоего сервиса:

```
# Аутентификация

auth required pam_env.so

auth required pam_ldap.so

# Управление учётными записями

account required pam_ldap.so

# Управление паролями

password required

pam_ldap.so

# Управление сеансами

session optional pam_loginuid.so

session required pam_selinux.so

close session required pam_env.so

session required pam_log.so level=audit

session required pam_selinux.so open multiple

session optional pam_mail.so
```

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

Каждый из этих разделов в файле вызывает один или несколько модулей PAM. Например, pam_env.so устанавливает переменные среды, которые могут быть использованы последующими модулями.

Код, возвращаемый модулем PAM, вместе с управляющими директивами (в данном примере — required или optional) позволяет PAM решать, что делать дальше.

PAM поддерживают следующие управляющие директивы:

- required — указанный модуль PAM должен вернуть код успеха для того, чтобы весь сервис (например, аутентификация) была успешен. Если модуль PAM вернёт код неудачи, остальные модули будут всё равно вызваны (хотя уже точно известно, что сам сервис будет недоступен);

- requisite — указанный модуль PAM должен вернуть код успеха для того, чтобы весь сервис был доступен. В отличие от required, если модуль PAM вернёт код неудачи, директива сразу же завершится, и сам сервис будет недоступен;

- sufficient — если указанный модуль PAM вернёт код успеха, весь сервис будет разрешён. Оставшиеся модули PAM не будут проверяться. Однако, если модуль PAM вернёт код неудачи, оставшиеся модули пройдут проверку, а неудача данного модуля не будет приниматься во внимание;

- optional — код успеха или неудачи указанного модуля PAM будет иметь значение, если это единственный модуль в стеке. Цепочки модулей позволяют выполнить множественную аутентификацию, выполнить несколько задач в процессе создания сеанса и тому подобное. Так как конфигурационные файлы PAM управляют процессом аутентификации в приложении, очень важно правильно с ними взаимодействовать. Файлы обычно располагаются в каталоге /etc/pam.d/.

В больших организациях или в требовательных к безопасности системах любая модификация конфигурационных файлов PAM должна подвергаться соответствующему аудиту. Это же относится к каталогу, где располагаются модули PAM (/lib/security или /lib64/security). Помимо

файлов-сценариев для некоторых модулей могут использоваться дополнительные файлы конфигурации. Все они расположены в каталоге /etc/security и каждый файл предназначен для конкретной группы настроек.

Файл time.conf — предназначен для ограничения времени доступа пользователей с различных терминалов к различным сервисам. Например, запретить вход в систему с первой виртуальной консоли администратору во время выходных. Эти настройки обслуживает модуль pam_time и, соответственно, если необходимо, чтобы ограничения вступили в силу, модуль должен быть прописан в соответствующем сценарии.

Файл pam_env.conf — предназначен для ограничения возможности изменения отдельных переменных среды пользователями. Работает под руководством модуля pam_env.

Файл limits.conf — предназначен для индивидуального или группового ограничения: размера core-файла, максимально допустимого размера файла, максимального количества открытых файлов, запущенных процессов, количества одновременных входов в систему и т.д. Руководящий модуль pam_limits.

Файл access.conf — так как PAM имеет средства аутентификации по сети, то подобные настройки являются полезными, т.к. контролируется не только «кто может зайти» или «не зайти», но и «откуда». Контролируется pam_access.

Файл group.conf — указывает, какой группе будет принадлежать служба, запущенная определенным пользователем, в определённое время с определённого терминала. Контролируется pam_time и pam_group.

Файл console.perms — определет права, получаемые привилегированными пользователями к консоли во время входа в систему и возвращаемые при выходе. Модуль pam_console.

Список модулей:

- pam_cracklib: Тип password. Проверяет пароль на стойкость. Примечание. Это не обязательно при использовании модуля pam_unix. Полезен для программ, задающих пароли. Полезные параметры:

- – retry=N — даётся N попыток на исправление ошибки,

- – diffok=N — должно быть изменено минимум N символов при смене пароля,

- – minlen=N — минимальный размер пароля,

- – dcredit=N ucredit=N lcredit=N ocredit=N — в пароле должно присутствовать минимум N цифр, строчных, прописных букв и других символов.

- pam_deny: Тип любой. Всегда перекрывает доступ.

- pam_env: Тип auth. Контролирует сохранность переменных среды. Полезный параметр conffile=S — задаёт альтернативное название файла конфигурации.

- pam_ftp: Тип auth. Предназначен для организации анонимного доступа. То есть получив имя пользователя anonymous, ждёт в качестве пароля что-то похожее на его почтовый адрес. Полезные параметры: – ignore — не обращать внимание, похож ли пароль на почтовый адрес; – users=XXX,YYY — позволяет анонимный вход для пользователей из этого списка.

- pam_group: Тип auth. Предназначение ясно из описания конфигурационного файла group.conf.

- pam_lastlog: Тип auth. Сообщает о времени и месте входа в систему. Обновляет файл /var/log/wtmp. Полезные параметры: – nodate, noterm, nohost, silent — не выводить в сообщении дату, терминал, имя машины или вообще ничего, – never — если пользователь никогда ранее не появлялся, то его приветствуют.

- pam_limits: Тип session. Предназначение указано выше при описании файла limits.conf. Полезный параметр: conf=S — альтернативное имя конфигурационного файла.

- pam_listfile: Тип auth. Предназначен для организации доступа на основе конфигурационных файлов-списков например /etc/ftpaccess. Для примера смотрите соответствующие файлы сценариев. Возможные параметры: – onerr=succeed|fail — задаёт возвращаемое значение в случае неудачного поиска; – sence=allow|deny — задаёт возвращаемое значение в случае удачного поиска; – file=filename — имя файла со списком; – item=user|tty|rhost|ruser|group|shell apply=user|@group — дополнительные ограничения, если тип объявлен tty, rhost или shell.

- pam_mail: Тип auth. Сообщается о наличии почты, если таковая имеется. Полезные параметры: – dir=S — путь к каталогу почтовых очередей; – noenv — не устанавливать переменную среды MAIL; – close — сообщать, если есть почта у пользователей с аннулированными бюджетами; – nopen — не печатать какую-либо почтовую информацию, если пользовательский бюджет только что заведён.

- pam_nologin: Тип auth. Стандартная реакция на наличие файла /etc/nologin. Когда он присутствует, в систему может войти только root, а остальным будет выдано на экран содержимое этого файла;

- pam_permit: Тип любой. Использование данного модуля ОПАСНО и применимо только в критических ситуациях. Всегда даёт допуск.

- pam_pwdb: Тип любой. Замещает модули серии pam_unix.... Использует интерфейс библиотеки libpwdb (пользовательские базы данных), что повышает независимость системы аутентификации от способа хранения пользовательских данных (NIS или passwd или shadow). Полезные параметры: – nullok — можно использовать пустые пароли; – md5, shadow, bigcrypt — различные способы шифрования пароля.

- pam_radius: Тип session. Позволяет осуществлять аутентификацию через RADIUS сервер.

- pam_rhosts_auth: Тип auth. Механизм работы этого модуля основывается на анализе содержимого файлов hosts.equiv и .rhosts, используемых для аутентификации таких служб как rlogin и rsh. Полезные параметры: – no_hosts_equiv — игнорировать содержимое файла hosts.equiv; – no_rhosts — игнорировать содержимое файлов .rhosts; – suppres s — охраняет системные журналы от потока малозначимых сообщений, в частности, когда используется контрольный флаг sufficient.

- pam_root_ok: Тип auth. Используется в случае, когда администратору необходимо получить доступ к сервису без введения пароля. Этот модуль допускает пользователя к сервису только если его uid равен 0. • pam_securetty: Включает в проверку файл /etc/securitty. Суперпользователь сможет войти только на указанных терминалах.

- pam_time: Тип account. Смотри описание устройства конфигурационного файла time.conf;

- pam_warn: Тип auth и password. Просто ведёт записи в системных журналах, например, при смене пароля.

- pam_weel: Тип auth. Права суперпользователя может использовать только пользователь группы wheel (группа root). Полезные параметры: – group=XXX — использовать указанную группу вместо стандартной нулевой, – deny — инвертирование действия алгоритма, запрещается получать права суперпользователя пользователям указанной группы, используется вместе с предыдущим параметром, – trust — избавляет пользователей группы wheel от необходимости вводить пароль при смене uid на 0. Для отслеживания попыток неуспешной аутентификации необходимо от имени администратора root отредактировать файлы system-auth и password-auth: vi /etc/pam.d/system-auth vi /etc/pam.d/ password-auth Добавить в секцию auth две строки, устанавливающие блокировку учётной записи на время не менее 15 минут в случае ввода не более 4 неправильных паролей, и 1 строку, подключающую нужный модуль.

Rsyslog

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

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

Все программы ведут лог путём отправки сообщений об ошибках или своём состоянии с помощью сокета syslog или просто записывая все сообщения в файл, который будет находиться в каталоге /var/log/. Но важное значение имеет уровень подробности логирования.

Доступна настройка подробностей в каждой отдельной программе, или же с помощью syslog. Это поможет уменьшить использование дискового пространства на хранение логов. Но тут нужно найти компромисс между количеством информации и использованием диска. Например, ядро ОС определяет такие уровни логов, или как мы будем называть их ниже — приоритеты:

- KERN_EMERG — система неработоспособна;
- KERN_ALERT — нужно немедленно принять меры;
- KERN_CRIT — критическая ошибка;
- KERN_ERR — обычная ошибка;
- KERN_WARNING — предупреждение;
- KERN_NOTICE — замечание;

- KERN_INFO — информационное сообщение;
- KERN_DEBUG — сообщения отладки. Подобные уровни лога поддерживаются в большинстве программ, которые ведут логи.

Все настройки Rsyslog находятся в файле /etc/rsyslog.conf и других конфигурационных файлах из /etc/rsyslog.d.

Проверить наличие данных файлом можно командой:

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

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

В этих файлах могут содержаться дополнительные настройки, например, аутентификация на Rsyslog-сервере. В главном конфигурационном файле содержится очень много полезных настроек.

Обычно он обеспечивает управление локальными логами по умолчанию, но для работы через сеть нужно добавить настройки. Синтаксис конфигурационного файла очень прост: <переменная> <значение> Все директивы начинаются со знака доллара, содержат имя переменной, а дальше связанное с ней значение. Так выглядит каждая строка конфигурационного файла. В его первой части размещены общие настройки программы и загрузка модулей. Во второй — ваши правила сортировки и фильтрации лог-файлов.

Afick - верификация целостности

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

Afick контролирует изменения в файловой системе и сразу сообщает о них пользователю, тем самым предоставляя возмоожность применения или отклонения внесенных изменений.

Данная информация может помочь в расследовании инцидента, когда необходимо определить, какие были произведены изменения в системе в результате взлома. В процессе установки

Afick формирует базу данных файлов, каталогов и соответствующих им контрольных сумм. Файлы и каталоги, включенные в эту базу данных, выбираются, исходя из входных данных в файле конфигурации Afick, называемом afick.conf, после добавления этого файла в каталог /etc.

Файл конфигурации afick.conf имеет простую синтаксическую структуру. При необходимости пользователь может быстро добавить или удалить типы файлов, каталоги и т.д.

chrony

Chrony – клиент и сервер протокола сетевого времени NTP.

Chrony может быстрее синхронизировать системные часы с лучшей точностью времени, и он может быть особенно полезен для систем, которые не работают в сети все время.

Основные преимущества:

- эффективная работа в среде, где доступ к временной связке прерывистый;
- быстрая синхронизация времени с большой точностью;
- адаптация к внезапным изменениям частоты тактовых импульсов; •
- работа в перегруженной сети, даже если перегрузка продолжается длительное время;
- время в конфигурации по умолчанию никогда не изменяется, чтобы не нарушать работу других запущенных программ;
- использование меньшего объема памяти и запуск процессов только при необходимости в целях экономии энергии. Действия по настройке требуется производить с правами пользователя root или пользователя с правами администратора.

Chrony доступен по умолчанию в репозитории НАЙС ОС Z. Для синхронизации времени необходимо добавить службу chronyd.service в автозагрузку и запустить ее командой:

```
systemctl enable --now chronyd.service
```

Клиенты NTP должны знать, с какими серверами NTP они должны связаться, чтобы получить текущее время. Серверы NTP могут быть указаны в директиве server в файле конфигурации NTP.

Отказоустойчивый кластер

Corosync — это программа с открытым исходным кодом, которая предоставляет возможности кластерного членства и обмена сообщениями, часто называемые уровнем обмена сообщениями, на клиентские серверы.

Pacemaker — это менеджер ресурсов кластера с открытым исходным кодом (CRM), который координирует ресурсы и службы, которые управляются и становятся доступными кластеру.

По сути, Corosync позволяет серверам связываться как кластер, в то время как Pacemaker предоставляет возможность управлять тем, как ведёт себя кластер. Архитектура pacemaker состоит из трёх уровней:

- Кластеронезависимый уровень — на этом уровне располагаются сами ресурсы и их скрипты, которыми они управляются, и локальный демон, который скрывает от других уровней различия в стандартах, использованных в скриптах.

- Менеджер ресурсов (Pacemaker), который представляет из себя мозг. Он реагирует на события, происходящие в кластере: отказ или присоединение узлов, ресурсов, переход узлов в сервисный режим и другие административные действия. Pacemaker, исходя из сложившейся ситуации, делает расчёт наиболее оптимального состояния кластера и даёт команды на выполнение действий для достижения этого состояния (остановка/перенос ресурсов или узлов).

- Информационный уровень — на этом уровне осуществляется сетевое взаимодействие узлов, т.е. передача сервисных команд (запуск/остановка ресурсов, узлов и т.д.), обмен информацией о полноте состава кластера (quorum) и т.д. На этом уровне работает Corosync.

Для наглядности рассмотрим создание простого кластера с плавающим IP-адресом. В случае если работающий сервер отказывает, второй автоматически запускается и начинает работать вместо первого. Если плользователю требуется иметь доступ к настройкам кластера через доменное имя, необходимо настроить DNS-адресацию серверов. Если у домена нет имени для использования, можно использовать плавающий IP-адрес для доступа к настройкам.

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

Corosync использует UDP-транспорт между портами 5404 и 5406. Если используется брандмауэр, необходимо убедиться, что между этими серверами разрешена связь. Установка Corosync и Pacemaker производится командой:

```
dnf install corosync pcs pacemaker
```

Изменение приоритета процесса

Утилита nice — программа, предназначенная для запуска процессов с изменённым приоритетом nice.

Приоритет nice (целое число) процесса используется планировщиком процессов ядра ОС при распределении процессорного времени между процессами.

Приоритет nice — число, указывающее планировщику процессов ядра ОС приоритет, который пользователь хотел бы назначить процессу. Утилита nice, запущенная без аргументов, выводит приоритет nice, унаследованный от родительского процесса. nice принимает аргумент «смещение» в диапазоне от -20 (наивысший приоритет) до +19 (низший приоритет).

Если указать смещение и путь к исполняемому файлу, утилита nice получит приоритет своего процесса, изменит его на указанное смещение и использует системный вызов семейства exec() для замещения кода своего процесса кодом из указанного исполняемого файла. Команда nice сделает то же, но сначала выполнит системный вызов семейства fork() для запуска дочернего процесса (англ. sub-shell).

Если смещение не указано, будет использовано смещение +10. Привилегированный пользователь (root) может указать отрицательное смещение. Приоритет nice и приоритет планировщика процессов ядра ОС — разные числа. Число nice — приоритет, который пользователь хотел бы назначить процессу.

Приоритет планировщика — действительный приоритет, назначенный процессу планировщиком. Планировщик может стремиться назначить процессу приоритет, близкий к nice, но это не всегда возможно, так как в системе может выполняться множество процессов с разными приоритетами. Приоритет nice является атрибутом процесса и, как и другие атрибуты, наследуется дочерними процессами.

В выводе утилит top, ps, htop и др. приоритет nice называется NI — сокращение от nice, а приоритет планировщика — PRI — сокращение от

priority. Обычно, NI = PRI - 20, но это верно не всегда. По умолчанию NI=0, соответственно PRI=20. Планировщик процессов ядра ОС поддерживает приоритеты от 0 (реальное время) до 139 включительно.

Приоритеты -20...+19 утилиты или команды nice соответствуют приоритетам 100...139 планировщика процессов. Другие приоритеты планировщика процессов можно установить командой chrt из пакета util-linux. Для изменения приоритета уже запущенных процессов используется утилита renice.

Управление дисковыми квотами

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

Система ограничений (quota system) является частью ядра ОС.

Команда:

quota

позволяет просмотреть любые ограничения дискового пространства для каждого пользователя.

Доступны два типа ограничений, которые могут быть наложены на пользователя. Обычно, если используется одно из ограничений, то и второе тоже будет использоваться.

Ограничение может быть установлено как на все дисковое пространство пользователя, которое используется этим пользователем, так и на число файлов (inodes), которыми он может владеть.

Quota обеспечивает информацию на ограничения, которые были установлены системным администратором, на каждую из областей, которые используются в данный момент.

Ограничения по inodes и block накладываются как на uid (идентификатор пользователя), так и на gid (идентификатор группы).

Так, если пользователь входит в группу, которая превысила наложенное на неё ограничение, то он не сможет использовать дисковое пространство, даже если он все ещё может использовать его как пользователь. Существуют четыре числа для каждого ограничения:

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

«Мягкое» ограничение определяет число блоков размером 1 Кбайт, которое пользователь может немного превысить.

«Жёсткое» ограничение не может быть превышено ни каким образом. Если пользователь пытается превысить данное число, то он получает сообщение о невозможности сделать это. При этом ядро возвращает код ошибки EDQUOT.

После того как пользователь превысит доступное для него «мягкое» ограничение (softlimit) устанавливается время, после истечения которого «мягкое» ограничение становится «жёстким» (hardlimit).

Обычно срок этого периода истекает после 7 дней (1 неделя). В этот период времени пользователь может удалить ненужные ему файлы, после чего он вновь может использовать «мягкое» ограничение до момента истечения указанного промежутка времени.

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

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

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

В первую очередь необходимо решить, на какую файловую систему необходимо наложить ограничения (quotas). Чаще всего ограничения накладываются на файловую систему, в которой располагаются домашние каталоги пользователей или на файловую систему, которая смонтирована в каталог /usr, и пользователи имеют право записывать на неё информацию.

Для того чтобы разрешить ограничения на дисковое пространство, на необходимой файловой системе пользователю необходимо отредактировать файл /etc/fstab, добавив к указанной системе опции для ограничения дискового пространства (как для пользователей, так и для групп).

Блокирование файлов

В ряде случаев бывает необходимо защитить файлы, открытые кем-либо и используемые в текущий момент, от удаления любым другим

пользователем, даже если он имеет на это права. Для поддержки этого функционала в НАЙС ОС Z необходимо установить пакет fileprotect:

```
tdnf install fileprotect
```

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

Резервирование данных

В состав дистрибутива ОС входит утилита резервного копирования rsync. rsync (англ. Remote Synchronization) — программа, которая выполняет синхронизацию файлов и каталогов в двух местах с минимизированием трафика, используя сжатие данных при необходимости. Важным отличием rsync от многих других программ/- протоколов является то, что зеркалирование осуществляется одним потоком в каждом направлении (а не по одному или несколько потоков на каждый файл).

rsync может копировать или отображать содержимое каталога и копировать файлы, опционально используя сжатие и рекурсию. rsyncd — демон, реализующий протокол rsync. По умолчанию использует TCPпорт 873. Базовый синтаксис утилиты имеет вид:

```
rsync <опции> <источник> <место_назначения>
```

Лимиты ресурсов

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

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

Монтирование файловых систем

mount — утилита командной строки для монтирования файловых систем. Использование утилиты: mount /dev/cdrom /mnt/cdrom

Устройство /dev/cdrom монтируется в каталог /mnt/cdrom, если он существует. Начиная от момента монтирования и пока пользователь не отмонтирует файловую систему (или туда не будет смонтировано что-то иное), в каталоге /mnt/cdrom будет содержаться дерево каталогов устройства /dev/cdrom; те файлы и подкаталоги, которые раньше находились в /mnt/cdrom, сохранятся, но будут недоступны до размонтирования устройства /dev/cdrom.

Для размонтирования достаточно указать точку монтирования или имя устройства, например: umount /dev/cdrom В случае необходимости, при выполнении команды mount, можно указать дополнительные параметры монтирования: -t <Тип файловой системы>

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

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

Управление пользователями

Общая информация

НАЙС ОС Z – многопользовательская операционная система. Также имеются группы пользователей, основное предназначение которых — облегчить управление большим количеством пользователей, а также более точно распределить права доступа к различным объектам системы.

Пользователи и группы внутри системы обозначаются цифровыми идентификаторами — UID и GID, соответственно. Пользователь может входить в одну или несколько групп. По умолчанию он входит в группу, совпадающую с его именем. Чтобы узнать, в какие ещё группы входит пользователь, введите команду id, вывод ее может быть примерно следующим: uid=1000(test) gid=1000(test) группы=1000(test),16(rpm) Такая запись означает, что пользователь test (цифровой идентификатор 1000) входит в группы test и rpm. Разные группы могут иметь разный уровень доступа к тем или иным каталогам; чем в большее количество групп входит пользователь, тем больше прав он имеет в системе.

Утилита passwd

Утилита passwd поддерживает традиционные опции passwd и утилиту shadow.

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

Добавления нового пользователя

Для добавления нового пользователя используйте команды useradd и passwd. В примере в качестве первоначально введённого пароля указана последовательность символов 123, затем введён надёжный пароль:

useradd test1

passwd test1

Смена пароля для пользователя test1:

Новый пароль:

НЕУДАЧНЫЙ ПАРОЛЬ: СЛИШКОМ короткий

НЕУДАЧНЫЙ ПАРОЛЬ: СЛИШКОМ простой

Повторите ввод нового пароля:

passwd: все токены проверки подлинности успешно обновлены.

В результате описанных действий в системе появился пользователь test1 с некоторым паролем.

Если пароль оказался слишком слабым с точки зрения системы, она об этом предупредит (как в примере выше).

Пользователь в дальнейшем может поменять свой пароль при помощи команды passwd –, но если он попытается поставить слабый пароль, система откажет ему (в отличие от root) в изменении.

В дистрибутивах ОС для проверки паролей на слабость используется модуль PAM passwdqc.

Программа useradd имеет множество параметров, которые позволяют менять ее поведение по умолчанию. Например, можно принудительно указать, какой будет UID или какой группе будет принадлежать пользователь.В результате описанных действий в системе появился пользователь test1 с некоторым паролем.

Если пароль оказался слишком слабым с точки зрения системы, она об этом предупредит (как в примере выше). Пользователь в дальнейшем

может поменять свой пароль при помощи команды passwd –, но если он попытается поставить слабый пароль, система откажет ему (в отличие от root) в изменении.

В дистрибутивах ОС для проверки паролей на слабость используется модуль PAM passwdqc. Программа useradd имеет множество параметров, которые позволяют менять ее поведение по умолчанию. Например, можно принудительно указать, какой будет UID или какой группе будет принадлежать пользователь.

Модификация уже имеющихся пользовательских записей

Для модификации уже имеющихся пользовательских записей применяется утилита

usermod: usermod -G audio,rpm,test1 test1

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

usermod -l test2 test1

Будет произведена смена имени пользователя с test1 на test2.

usermod -L test2

и

usermod -U test2

соответственно временно блокируют возможность входа в систему пользователю test2 и возвращают все на свои места. Изменения вступят в силу только при следующем входе пользователя в систему. При неинтерактивной смене или задании паролей для целой группы пользователей используйте утилиту chpasswd.

На стандартный вход ей следует подавать список, каждая строка которого будет выглядеть как:

<имя>:<пароль>

Удаление пользователей

Для удаления пользователей используйте userdel. Команда:

```
userdel test2
```

удалит пользователя test2 из системы.

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

Утилиты vigr и T используются для ручного редактирования файлов /etc/passwd и /etc/group, в которых хранятся основные записи о пользователях и группах в системе. Не рекомендуется создавать пользователей с правами сверх необходимых. Предпочтительнее создать серию новых групп и включить в них требуемого пользователя. А для данных групп установить соответствующие права на объектах файловой системы (утилиты chmod и chown).

Пароли пользователей

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

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

Принцип:

```
login : password : UID : GID : GECOS : home : shell
```

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

- регистрационное имя или логин;

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

Основным назначением /etc/passwd является сопоставление логина и идентификатора пользователя (UID). Изначально поле пароля содержало хеш пароля и использовалось для аутентификации. Однако, в связи с ростом вычислительных мощностей процессоров появилась серьёзная угроза применения простого перебора для взлома пароля.

Поэтому все пароли были перенесены в специальные файлы, такие как /etc/shadow. Эти файлы недоступны для чтения обычным пользователям. Такой подход называется механизмом скрытых паролей. Регистрационные имена должны быть уникальными и представлять собой строки не длиннее 32 символов (любые, кроме двоеточия и символа новой строки).

По сути, имя пользователя — это его короткий и легко запоминаемый псевдоним, который используется при входе в систему и часто включается в адреса электронной почты.

Идентификатор пользователя — это число от 0 до 232-1. Пользователь с идентификатором «0» (обычно root) называется суперпользователем и имеет право на выполнение любых операций в системе. Принято соглашение о выделении «специальным» пользователям (bin, daemon), назначение которых — только запуск определённых программ, маленьких идентификаторов (меньше 100). Пользователь может принадлежать к одной или нескольким группам, которые используются для задания прав более чем одного пользователя на тот или иной файл.

Список групп с их участниками задаётся в /etc/group. В файле же /etc/passwd указывается идентификатор группы по умолчанию. Всем файлам, созданным пользователем после регистрации в системе, будет автоматически присвоен этот номер группы (исключение — если для каталога, в котором создаётся файл, установлен в правах бит SGID, то будет присвоена такая же группа, как у самого каталога).

/etc/group содержит записи обо всех группах в системе. Каждая его строка содержит: • символьное имя группы; • пароль группы — устаревшее поле, сейчас не используется. В нём обычно стоит «x»; • идентификатор группы, или GID; • список имён участников, разделённых запятыми. Пример записи: bin:x:1:root,bin,daemon Здесь сообщается, что группа bin имеет GID=1, а входят в неё пользователи root, bin и daemon. Поле GECOS хранит вспомогательную информацию о пользователе (номер телефона, адрес, полное имя и так далее). Оно не имеет чётко определённого синтаксиса. Тем не менее, демон fingerd предполагает, что в нём содержатся следующие элементы, разделённые запятыми: • полное имя; • адрес офиса или домашний адрес; • рабочий телефон; • домашний телефон. С помощью утилиты chfn можно изменять эту информацию, а с помощью finger — узнать, например, полное имя любого пользователя в системе (или даже на другом Глава 7. Управление пользователями 180 компьютере сети). Пример строки с заполненным полем GECOS: tester:x:210:8:Edward Chernenko,Marx Street 10,4554391,5454221: /home/ed:/bin/bash После входа в систему пользователь оказывается в своём домашнем каталоге. Исторически сложилось так, что домашний каталог пользователя root называется /root, а остальные имеют вид /home/<имя_пользователя>. Если на момент входа в систему домашний каталог отсутствует, то система выдаёт сообщение об ошибке и отказывается допустить пользователя к командной строке. Это можно изменить посредством установки параметра в файле /etc/login.defs в значение no. Следует отметить, что при использовании графического интерфейса пользователь не увидит предупреждения или сообщения об ошибке, но просто будет выведен из системы безо всяких объяснений (так как оконный менеджер не сможет выполнить запись в нужный каталог, такой как ~/.gnome). В поле регистрационной оболочки задаётся shell, то есть интерпретатор командной строки. Здесь может быть указана любая

программа, и пользователь может сам выбирать для себя наиболее подходящую при помощи команды chsh. Тем не менее некоторые системы в целях безопасности требуют, чтобы суперпользователь явно разрешил использовать приложение в качестве интерпретатора командной строки. Для этого используется специальный файл /etc/shells, содержащий список «допустимых» оболочек. vipw — запускает текстовый редактор, указанный в переменной среды EDITOR (или редактор по умолчанию, обычно vi), загружая в него копию файла /etc/passwd. После закрытия редактора переносит временную копию в сам файл. Не позволяет двум пользователям выполнять редактирование одновременно. В файле /etc/shadow хранятся хеши паролей всех пользователей в системе. Процессы суперпользователя могут читать его напрямую, а для остальных создана специальная библиотека PAM. Она позволяет непривилегированным приложениям спрашивать у неё, правильный ли пароль ввёл пользователь, и получать ответ. Библиотека PAM, как правило, действует с привилегиями вызвавшего процесса. Таким образом, хеш не попадает «в чужие руки». Пароль шифруется с MD5-хешированием или blowfish-хешированием (bcrypt), MD5-хеши всегда записываются после префикса 1. Перед хешированием к паролю добавляются случайные символы — salt (соль, от англ. add salt to something — сделать что-либо более интересным; в русскоязычных источниках иногда используется термин «затравка»). Salt также приписывается к началу полученного хеша. Благодаря salt нельзя при простом просмотре файла обнаружить пользователей с одинаковыми паролями. Кроме имени (первое поле каждой строки) и хеша (второе поле) в файле /etc/shadow также хранятся: • дата последнего изменения пароля; • через сколько дней можно будет поменять пароль; • через сколько дней пароль устареет; • за сколько дней до того, как пароль устареет, начать напоминать о необходимости смены пароля; • через сколько дней после того, как пароль устареет, заблокировать учётную Глава 7. Управление пользователями 181 запись пользователя; • дата, при достижении которой учётная запись блокируется; • зарезервированное поле. Даты обозначаются как число дней с 1 января 1970 года. ОС поддерживает управление качеством используемых паролей. Рассмотрим настройку различных параметров используемых паролей. Время действия пароля (по истечении указанного

времени пользователь должен будет сменить пароль). Необходимо в конфигурационном файле /etc/login.defs изменить параметр. Обратите внимание, что данное требование будет работать только для вновь создаваемых пользователей, для уже существующих нужно использовать команду: chage -M Если пользователю необходимо выдавать предупреждение за несколько дней до окончания срока действия пароля, необходимо использовать параметр.

Средство контейнеризации

Общие сведения о контейнеризации

Контейнеризация — метод виртуализации, при котором ядро операционной системы поддерживает несколько изолированных экземпляров пространства пользователя вместо одного.

Эти экземпляры (обычно называемые контейнерами) с точки зрения выполняемых в них процессов идентичны отдельному экземпляру операционной системы. Ядро обеспечивает полную изолированность контейнеров, поэтому программы из разных контейнеров не могут воздействовать друг на друга. В отличие от аппаратной виртуализации, при которой эмулируется аппаратное окружение и может быть запущен широкий спектр гостевых операционных систем, в контейнере может быть запущен экземпляр операционной системы только с тем же ядром, что и у хостовой операционной системы (все контейнеры узла используют общее ядро).

При этом при контейнеризации отсутствуют дополнительные ресурсные накладные расходы на эмуляцию виртуального оборудования и запуск полноценного экземпляра операционной системы, характерные при аппаратной виртуализации.

В НАЙС ОС Z в качестве средства контейнеризации используется контейнерная платформа Docker. Docker — программное обеспечение для автоматизации развертывания и управления приложениями в средах с поддержкой контейнеризации. Позволяет "упаковать" приложение со всем его окружением и зависимостями в контейнер, который может быть развернут на любой Linux-подобной системе с поддержкой контрольных групп в ядре, а также предоставляет набор команд для управления этими контейнерами. Для экономии пространства хранения проект использует файловую систему aufs с поддержкой технологии каскадно-объединенного монтирования: контейнеры используют образ базовой операционной системы, а изменения записываются в отдельную область. Также поддерживается размещение контейнеров в файловой системе btrfs с включенным режимом копирования при записи. В состав

программных средств входит демон — сервер контейнеров, клиентские средства, позволяющие из интерфейса командной строки управлять образами и контейнерами, а также REST API, позволяющий управлять контейнерами программно. Демон обеспечивает полную изоляцию запускаемых на узле контейнеров на уровне файловой системы (у каждого контейнера собственная корневая файловая система), на уровне процессов (процессы имеют доступ только к собственной файловой системе контейнера, а ресурсы разделены средствами containerd), на уровне сети (каждый контейнер имеет доступ только к привязанному к нему сетевому пространству имен и соответствующим виртуальным сетевым интерфейсам).

Набор клиентских средств позволяет запускать процессы в новых контейнерах, останавливать и запускать контейнеры, приостанавливать и возобновлять процессы в контейнерах. Набор команд позволяет осуществлять мониторинг запущенных процессов. Новые образы возможно создавать из специального файла, также возможно записать все изменения, произведенные в контейнере, в новый образ. Все команды могут работать как с docker-сервисом локальной системы, так и с любым сервером Docker, доступным по сети. Кроме того, в интерфейсе командной строки встроены возможности по взаимодействию с различными централизованными хранилищами образов, в которых размещены предварительно собранные образы контейнеров.