50 1190 0101

Утвержден

РУСБ.10015-01-УД

Подп. и дата	
Инв. № дубл.	
Взам. инв. №	
Подп. и дата	
Инв. № подл.	

ОПЕРАЦИОННАЯ СИСТЕМА СПЕЦИАЛЬНОГО НАЗНАЧЕНИЯ «ASTRA LINUX SPECIAL EDITION»

Руководство по КСЗ. Часть 1

РУСБ.10015-01 97 01-1

Листов 357

2024

Литера $О_1$

АННОТАЦИЯ

Настоящий документ является первой частью руководства по комплексу средств защиты (КСЗ) программного изделия РУСБ.10015-01 «Операционная система специального назначения «Astra Linux Special Edition» (далее по тексту — ОС).

Документ предназначен для администраторов безопасности.

Руководство по КСЗ состоит из трех частей:

- РУСБ.10015-01 97 01-1 «Операционная система специального назначения «Astra Linux Special Edition». Руководство по КСЗ. Часть 1»;
- РУСБ.10015-01 97 01-2 «Операционная система специального назначения «Astra Linux Special Edition». Руководство по КСЗ. Часть 2»;
- РУСБ.10015-01 97 01-3 «Операционная система специального назначения «Astra Linux Special Edition». Руководство по КСЗ. Часть 3. Защищенная СУБД».

В первой части руководства приведены общие сведения о КСЗ, рассмотрены идентификация и аутентификация, дискреционное и мандатное управление доступом, защита памяти, изоляция процессов, защита среды виртуализации, маркировка документов, контроль подключения съемных машинных носителей информации, сопоставление пользователя с устройством, регистрация событий безопасности, надежное функционирование, фильтрация сетевого потока, контроль целостности, генерация КСЗ, режим ограничения действий пользователя Киоск-2, а также порядок запуска и применения ОС.

Во второй части руководства приведено описание тестов КСЗ.

В третьей части руководства приведено описание защищенной СУБД и порядок работы с ней.

Дополнительная информация о настройке компонентов и управлении программными пакетами, а также варианты реализации отдельных решений с использованием ОС приведены на официальном сайте `wiki.astralinux.ru`.

СОДЕРЖАНИЕ

1. ОБЩИЕ СВЕДЕНИЯ

КСЗ (подсистема безопасности PARSEC) предназначен для реализации функций ОС по защите информации от НСД и предоставления администратору безопасности информации средств управления функционированием КСЗ.

ВНИМАНИЕ! После установки ОС интерактивный вход в систему суперпользователя `root` по умолчанию заблокирован. Создаваемый при установке операционной системы пользователь включается в группу `astra-admin`. Пользователям, входящим в названную группу, через механизм `sudo` предоставляются права для выполнения действий по настройке ОС, требующих привилегий суперпользователя `root`. Далее по тексту такой пользователь именуется администратором.

1.1. Приемка ОС

При приемке экземпляра ОС необходимо провести следующие проверки:

1) при способе поставки BOX:
   а) проверка целостности упаковки;
   б) проверка комплектности в соответствии с формуляром из комплекта поставки;
   в) проверка маркировки — наличия в документе РУСБ.10015-01 30 02 «Операционная система специального назначения «Astra Linux Special Edition». Формуляр» уникального идентификатора ФСТЭК России[1)] или наличия знака СЗИ на упаковке и нерабочей поверхности установочного диска[2)];
   г) проверка контрольной суммы установочного диска и диска с документацией;
2) при способе поставки OEM:
   а) проверка целостности упаковки;
   б) проверка комплектности в соответствии с формуляром из комплекта поставки;
   в) проверка маркировки — наличия в документе РУСБ.10015-01 30 02 уникального идентификатора ФСТЭК России[1)] или наличия знака СЗИ на упаковке и нерабочей поверхности установочного диска[2)];
   г) проверка контрольной суммы установочного диска;
3) при способе поставки по сетям связи[1)]:
   а) проверка электронной подписи изготовителя образа установочного диска и документа РУСБ.10015-01 30 02 в формате PDF;
   б) проверка комплектности в соответствии с документом РУСБ.10015-01 30 02;
   в) проверка маркировки — наличия в разделе 6 документа РУСБ.10015-01 30 02 уникального идентификатора ФСТЭК России;

[1)] При поставке для применения в автоматизированных системах, находящихся в компетенции ФСТЭК России.
[2)] При поставке для применения в автоматизированных системах в защищенном исполнении, находящихся в компетенции Министерства обороны Российской Федерации.

г) проверка контрольной суммы образа установочного диска.

Подлинность и неизменность программного обеспечения изделия после установки на средство вычислительной техники подтверждаются средствами контроля целостности (`fly-admin-int-check`, `astra-int-check`) в соответствии с 10.3 путем вычисления и сравнения контрольных сумм исполняемых файлов и библиотек с эталонными значениями, хранящимися в файле `gostsums.txt`, размещенном на установочном диске и в обновлениях изделия.

Порядок контроля целостности при применении оперативных обновлений приведен в документе РУСБ.10015-01 31 01 «Операционная система специального назначения «Astra Linux Special Edition». Описание применения».

1.2. Состав КСЗ

В состав КСЗ входят следующие основные подсистемы:

- модули подсистемы безопасности PARSEC, входящие в состав ядра ОС;
- библиотеки;
- утилиты безопасности;
- подсистема протоколирования (регистрации);
- модули аутентификации;
- графическая подсистема;
- консольный вход в систему;
- средства контроля целостности;
- средства восстановления;
- средства разграничения доступа к виртуальным машинам[1)] ;
- средства контроля подключаемых устройств.

1.3. Контролируемые функции

КСЗ обеспечивает реализацию следующих функций ОС по защите информации от НСД:

1) в соответствии с требованиями безопасности информации к операционным системам[2)] :
   - идентификацию и аутентификацию;
   - управление доступом;
   - регистрацию событий безопасности;

[1)] Для процессоров, поддерживающих технологию виртуализации. Недоступно в режиме «Мобильный».

[2)] Требования безопасности информации к операционным системам утверждены приказом ФСТЭК России от 19 августа 2016 г. № 119.

- ограничение программной среды;
- изоляцию процессов;
- защиту памяти;
- контроль целостности;
- надежное функционирование;
- фильтрацию сетевого потока;
- маркирование документов;

2) в соответствии с требованиями по безопасности информации к средствам виртуализации[1)]:

- доверенную загрузку виртуальных машин;
- контроль целостности;
- регистрацию событий безопасности;
- управление доступом;
- резервное копирование;
- управление потоками информации;
- защиту памяти;
- ограничение программной среды;
- идентификацию и аутентификацию пользователей;
- централизованное управление образами виртуальных машин и виртуальными машинами;

3) в соответствии с требованиями по безопасности информации к средствам контейнеризации[2)]:

- изоляцию контейнеров;
- выявление уязвимостей в образах контейнеров;
- контроль целостности контейнеров и их образов;
- регистрацию событий безопасности;
- идентификацию и аутентификацию пользователей;

4) в соответствии с требованиями по безопасности информации к системам управления базами данных[3)]:

- управление доступом;
- идентификацию и аутентификацию пользователей;
- контроль целостности в системе управления базами данных;

[1)] Требования по безопасности информации к средствам виртуализации утверждены приказом ФСТЭК России от 27 октября 2022 г. № 187.

[2)] Требования по безопасности информации к средствам контейнеризации утверждены приказом ФСТЭК России от 4 июля 2022 г. № 118.

[3)] Требования по безопасности информации к системам управления базами данных утверждены приказом ФСТЭК России от 14 апреля 2023 г. № 64.

- регистрацию событий безопасности;
- резервное копирование и восстановление;
- обеспечение доступности;
- очистку памяти;
- производительность системы управления базами данных;
- ограничение программной среды.

Реализация функций безопасности ОС основана на следующих основных положениях:

1) при моделировании и описании управления доступом в ОС на основе ГОСТ Р 59453.1-2021 используются следующие термины:

а) субъект доступа — активный компонент ОС (например, процесс, запущенный от имени учетной записи пользователя), доступы которого регламентируются политиками управления доступом;

б) сущность (объект доступа) — пассивный компонент ОС, доступ к которому регламентируется политиками управления доступом, при этом используются следующие виды сущностей (объектов доступа):

- сущность-объект (объект) — пассивный компонент ОС (например, файл, сетевое соединение (файл-сокет), устройство (файл-устройство), файл-образ виртуальной машины или контейнера, как средства изоляции программной среды), доступ к которому регламентируется политиками управления доступом, к частям которого по отдельности управление доступом не осуществляется;

- сущность-контейнер (контейнер) — пассивный составной компонент ОС (например, каталог, том), доступ к которому регламентируется политиками управления доступом, состоящий из сущностей-объектов или сущностей-контейнеров, к которым по отдельности возможно осуществление управления доступом;

2) с каждым пользователем системы связан уникальный численный идентификатор — идентификатор пользователя (UID), который однозначно соотносится с записью в БД пользователей, содержащей информацию о пользователях, включая их реальные и системные имена. БД пользователей поддерживается и управляется системным администратором. UID является ярлыком субъекта (номинальный субъект), которым система пользуется для определения прав доступа. БД пользователей в ОС может быть как локальной для системы, так и являться частью ЕПП, функционирующего на основе протокола LDAP;

3) каждый пользователь входит в одну или более групп. Группа — это список пользователей системы, имеющий собственный идентификатор (GID). Поскольку группа объединяет несколько пользователей системы, в терминах политики безопасности она соответствует понятию «множественный субъект». GID является ярлыком мно-

жественного субъекта, которых у номинального субъекта может быть более одного. Таким образом, одному UID соответствует список GID;

4) роль действительного (работающего с сущностями) субъекта играет процесс. Каждому процессу присваивается единственный UID, являющийся идентификатором запустившего процесс номинального субъекта, т. е. пользователя. Процесс, порожденный некоторым процессом пользователя, наследует UID родительского процесса. Таким образом, все процессы, запускаемые пользователем, имеют его идентификатор. Все процессы, принадлежащие пользователю, образуют сеанс пользователя. Первый процесс сеанса пользователя порождается после прохождения процедур идентификации и аутентификации. При обращении процесса к сущности доступ предоставляется по результатам процедуры авторизации, т. е. обработки запроса на основе мандатных и дискреционных ПРД;

5) механизм ПРД реализован в ядре ОС, что обеспечивает его правильное функционирование при использовании любых компонентов, предоставляемых ОС. Реализация мандатного управления доступом затрагивает все подсистемы ядра, в которых реализовано дискреционное управление доступом. При этом оба вида управления доступом функционируют параллельно, не влияя на принятие решений друг друга (непротиворечивость). Доступ разрешается в том случае, если он возможен с учетом дискреционных и мандатных ПРД. Запрещается в случае, если доступ запрещен хотя бы одним из видов ПРД (дискреционным или мандатным).

1.4. Безопасная установка ОС

Безопасная установка ОС осуществляется в соответствии с РУСБ.10015-01 95 01-2 «Операционная система специального назначения «Astra Linux Special Edition». Руководство администратора. Часть 2. Установка и миграция».

1.5. Средства организации ЕПП

Организация ЕПП обеспечивает:

- сквозную аутентификацию в сети;
- централизацию хранения информации об окружении пользователей;
- централизацию хранения настроек системы защиты информации на сервере.

Сетевая аутентификация и централизация хранения информации об окружении пользователя подразумевает использование двух основных механизмов: поддержки кроссплатформенных серверных приложений для обеспечения безопасности (NSS) и подгружаемых аутентификационных модулей (PAM). Сквозная аутентификация в сети реализуется на основе протокола Kerberos с использованием службы каталогов LDAP в качестве источника данных для базовых системных служб на базе механизмов NSS и PAM. Подобный подход

обеспечивает централизацию хранения информации об окружении пользователей (в том числе предназначенную для обеспечения мандатного управления доступом):

- существующие в системе уровни и категории конфиденциальности;
- минимальные и максимальные уровни конфиденциальности, доступные пользователям при входе в систему;
- минимальные и максимальные наборы категорий конфиденциальности, доступные пользователям при входе в систему;
- члены привилегированных групп, которые могут получать из БД службы каталогов LDAP определенную информацию о пользователях.

Кроме того, с использованием СЗФС CIFS обеспечено централизованное хранение домашних каталогов пользователей.

Для снижения нагрузки на сеть и повышения производительности в ЕПП может применяться кэширование редко изменяемой информации в локальном кэше.

ВНИМАНИЕ! Измененная на сервере информация может попасть в локальный кэш с задержкой.

Более подробное описание ЕПП приведено в РУСБ.10015-01 95 01-1 «Операционная система специального назначения «Astra Linux Special Edition». Руководство администратора. Часть 1».

2. ИДЕНТИФИКАЦИЯ И АУТЕНТИФИКАЦИЯ

Идентификация и аутентификация пользователей в ОС выполняется с учетом требований ГОСТ Р 58833-2020 «Защита информации. Идентификация и аутентификация. Общие положения».

В общем случае идентификация и аутентификация охватывают:

1) первичную идентификацию, включающую подготовку, формирование и регистрацию информации о субъекте (объекте) доступа, а также присвоение субъекту (объекту) доступа идентификатора доступа и его регистрацию в перечне присвоенных идентификаторов;

2) хранение и поддержание актуального состояния (обновление) идентификационной и аутентификационной информации субъекта (объекта) доступа в соответствии с установленными правилами;

3) вторичную идентификацию, которая обеспечивает опознавание субъекта доступа, запросившего доступ к объекту доступа, по предъявленному идентификатору;

4) аутентификацию, включающую проверку подлинности субъекта (объекта) доступа и принадлежности ему предъявленных идентификатора и аутентификационной информации.

Функция идентификации и аутентификации пользователей в ОС основывается на использовании механизма PAM.

PAM представляют собой набор разделяемых библиотек (т. н. модулей), с помощью которых системный администратор может организовать процедуру аутентификации (подтверждение подлинности) пользователей прикладными программами. Каждый модуль реализует собственный механизм аутентификации. Изменяя набор и порядок следования модулей, можно построить сценарий аутентификации.

Подобный подход позволяет изменять процедуру аутентификации без изменения исходного кода и повторного компилирования PAM.

Сценарии аутентификации (т. е. работа этих функций) описываются в конфигурационном файле `/etc/pam.conf` и в конфигурационных файлов, расположенных в каталоге `/etc/pam.d/`. Сама аутентификация выполняется с помощью PAM. Модули располагаются в каталоге `/lib/security` в виде динамически загружаемых объектных файлов.

Если ЕПП не используется, аутентификация осуществляется с помощью локальной БД пользователей (файл `/etc/passwd`) и локальной БД пользовательских паролей (файл `/etc/shadow`). Подробную информацию о структуре этих файлов можно получить с помощью команд `man 5 passwd` и `man 5 shadow` соответственно. В ОС реализована возмож-

ность хранения аутентификационной информации пользователей, полученной с использованием хеш-функций по ГОСТ Р 34.11-2012 (ГОСТ Р 34.11-94).

При использовании ЕПП аутентификация пользователей осуществляется централизованно по протоколу Kerberos. Для защиты аутентификационной информации по умолчанию используются отечественные алгоритмы по ГОСТ 28147-89 и ГОСТ Р 34.11-2012.

В ЕПП в качестве источника данных для идентификации и аутентификации пользователей применяются службы каталогов LDAP. В результате вся служебная информация пользователей сети может располагаться на выделенном сервере в распределенной гетерогенной сетевой среде. Добавление новых сетевых пользователей в этом случае производится централизованно на сервере службы каталогов. Сетевые службы, поддерживающие возможность аутентификации пользователей (web, FTP, почта), могут вместо локальных учетных записей использовать тот же каталог LDAP проверки аутентификационной информации. Администратор сети может централизованно управлять конфигурацией сети, в т. ч. разграничивать доступ к сетевым службам.

Благодаря предоставлению информации LDAP в иерархической древовидной форме разграничение доступа в рамках службы каталогов LDAP может быть основано на введении доменов. В качестве домена в данном случае будет выступать поддерево службы каталогов LDAP. Сервисы LDAP позволяют разграничивать доступ пользователей к разным поддеревьям каталога, хотя по умолчанию в ОС реализуется схема одного домена.

Для управления пользователями, группами и их атрибутами используется графическая утилита `astra-systemsettings` («Параметры системы», см. электронную справку).

Примечание. При создании локальных пользователей или пользователей ЕПП необходимо обязательно устанавливать для них диапазоны допустимых уровней и категорий конфиденциальности: минимальный и максимальный уровни конфиденциальности, минимальный и максимальный наборы категорий конфиденциальности (см. раздел 4). Отсутствие установленных допустимых диапазонов мандатных атрибутов приводит к запрещению доступа при обращении к сетевым службам защищенных комплексов программ гипертекстовой обработки данных, электронной почты, СУБД и печати.

По умолчанию в сценарии `/etc/pam.d/common-auth`, содержащем общие для всех служб настройки и предоставляющем службу для входа в систему, используется PAM-модуль `pam_tally.so`. Данный PAM-модуль при начале процедуры аутентификации пользователя увеличивает счетчик неуспешных попыток аутентификации пользователя на единицу. Число неуспешных попыток аутентификации пользователя может быть просмотрено следующей командой:

```
faillog -u <имя_пользователя>
```

После успешного завершения попытки аутентификации пользователя счетчик неуспешных попыток аутентификации сбрасывается в ноль. Максимальное число неуспешных попыток аутентификации пользователя определяется на основе значения, заданного инструментом командной строки `faillog`, и значений параметров `deny` и `per_user`:

```
auth [success=ignore default=die] pam_tally.so per_user deny=10
```

Наличие параметра `per_user` означает, что если с помощью инструмента `faillog` было задано неравное `0` максимальное значение неуспешных попыток аутентификации для пользователя, то будет применяться оно. Иначе применяется значение, определяемое параметром `deny`. При отсутствии параметра `per_user` используется значение параметра `deny`.

Для сброса счетчика неуспешных попыток аутентификации для пользователя необходимо выполнить команду:

```
faillog -r -u <имя_пользователя>
```

Описание способов использования также приведено на справочных страницах `man faillog` и `man pam_tally`.

3. ДИСКРЕЦИОННОЕ УПРАВЛЕНИЕ ДОСТУПОМ

3.1. Общие сведения

Дискреционное управление доступом применяется к каждому субъекту и сущности. Механизм дискреционного управления доступом именованных субъектов (пользователей) к именованным сущностям (файлам, каталогам) обеспечивает создание для каждой пары субъект-сущность явного и недвусмысленного перечисления разрешенных типов доступа, формирующих правила разграничения доступа (ПРД). На защищаемые именованные сущности при их создании автоматически устанавливаются базовые дискреционные ПРД в виде:

- идентификаторов пользователя-владельца (UID) и группы-владельца (GID), которые вправе распоряжаться доступом к данной сущности;
- прав доступа данных субъектов к созданной сущности.

Сущностями доступа являются:

- файлы;
- каталоги;
- соединения (сокеты);
- сетевые пакеты;
- механизмы IPC (разделяемая память, очереди сообщений и др.).

3.1.1. Права доступа

Доступ к сущности определяется тремя видами операций:

- чтение (`read`, символьный код `r`, восьмеричный код `4`, битовая маска `100`);
- запись (`write`, символьный код `w`, восьмеричный код `2`, битовая маска `010`);
- исполнение (`execution`, буквенный код `x`, восьмеричный код `1`, битовая маска `001`).

В записях прав доступа на первом месте всегда операция чтения, на втором — операция записи, а на третьем — операция исполнения.

Права доступа к сущности определяют разрешенные с ней операции для каждого из трех классов субъектов:

- пользователь-владелец сущности (`user`, `u`);
- группа-владелец сущности (`group`, `g`);
- все остальные (`other`, `o`),

При отображении/указании прав доступа к сущности на первом месте записываются права доступа пользователя-владельца, на втором — группы-владельца, на третьем — прочих субъектов доступа.

Указываться права доступа могут символьной строкой, восьмеричными числами и битовыми масками:

1) при обозначении символьной строкой каждый тип операции обозначается соответствующим ему символьным кодом, а отсутствие права на операцию — символом «–» в соответствующей позиции, например:

а) `rwx` — разрешены все операции;

б) `r-x` — разрешены операции чтения и исполнения (`r` и `x`), а операция записи запрещена;

2) при обозначении восьмеричными числами используются суммы восьмеричных чисел разрешенных операций, например:

а) символьной строке `rwx` соответствует восьмеричная цифра `7` (4+2+1);

б) символьной строке `r-x` соответствует восьмеричная цифра `5` (4+0+1);

3) при обозначении битовыми масками суммируются битовые маски разрешенных операций:

а) символьной строке `rwx` соответствуют восьмеричная цифра `7` и битовая маска `111`;

б) символьной строке `r-x` соответствуют восьмеричная цифра `5` и битовая маска `101`.

Набор прав доступа к сущности для всех классов субъектов может быть представлен:

- символьной строкой из 9 символов, по три символа для каждого класса субъектов доступа: `rwxr-xr-x`;
- строкой из трех восьмеричных цифр, по одной цифре для каждого класса субъектов доступа: `755`;
- битовой маской, по три бита для каждого класса субъектов доступа: `111 101 101`.

При обращении субъекта к сущности (с запросом операции определенного вида, т. е. на чтение, запись или исполнение) система проверяет совпадение идентификаторов UID и GID процесса с идентификаторами UID и GID файла в определенном порядке, и, в зависимости от результата, применяет ту или иную группу прав.

Права доступа сущности могут быть изменены, если это разрешено (санкционировано) текущими правилами разграничения доступа.

Дополнительно для сущностей могут быть установлены следующие специальные биты:

- SUID (Set User ID) — бит, позволяющий запускать исполняемый файл с правами пользователя-владельца;
- SGID (Set Group ID) — бит, позволяющий запускать исполняемый файл с правами группы-владельца;
- Sticky — бит, ограничивающий удаление чужих файлов в каталоге.

Когда пользователь или процесс запускают исполняемый файл с установленными битами SUID и/или SGID, то файл запускается с правами своего владельца и/или группы (в зависимости от того, какой бит задан). Таким образом, пользователь может запускать файлы даже от имени root, не обладая его привилегиями. При установке на каталог бита SGID все вновь создаваемые в нем файлы будут наследовать группу каталога, а вновь создаваемые подкаталоги — и группу каталога, и SGID-бит (установка SUID-бита на каталог ни на что не влияет).

Sticky-бит используется для ограничения удаления объектов в каталоге. При его установке на каталог удалять или переименовывать сущности в этом каталоге сможет только их владелец или суперпользователь. Это полезно при совместной работе нескольких пользователей, так как без Sticky-бита любой пользователь с правами записи в каталог сможет удалять в нем любые файлы, включая созданные другими пользователями. Установить или снять Sticky-бит с каталога может владелец каталога или root. Установка Sticky-бита на файл ни на что не влияет.

Для просмотра прав, установленных на сущности в каталоге, выполнить из данного каталога команду:

```
ls -l
```

Вывод команды будет иметь следующий вид:

```
-rw-r--r-- 1 user user 485 авг 28 10:17 readme.txt
-rwSr-Sr-- 1 user user 20 авг 28 10:17 script.sh
drwxr-xr-x 2 user user 4096 авг 22 13:12 Видео
drwxr-xr-x 2 user user 4096 авг 22 13:12 Документы
drwxr-xr-t 2 user user 4096 авг 22 13:12 Музыка
```

В первом столбце вывода перечисляются установленные дискреционные ПРД сущностей. В таблице 1 приведена их расшифровка.

Таблица 1

Тип сущности	Права пользователя-владельца			Права группы-владельца			Права всех остальных			...	Имя сущности
-	r	w	S	r	-	S	r	-	-		script.sh
d	r	w	x	r	-	x	r	-	t		Музыка

Специальные обозначения:

- тип сущности: символ «–» — файл, `d` — каталог;
- символ `S` в правах пользователя-владельца и/или группы-владельца означает установленные `SUID` и/или `SGID` биты соответственно;
- символ `t` на последней позиции означает установленный `Sticky`-бит.

Изменение прав доступа к сущностям осуществляется с помощью инструмента `chmod` (см. 3.3.3). Смена пользователя-владельца и группы-владельца сущности возможна с помощью инструментов `chown` (см. 3.3.1) и `chgrp` (см. 3.3.2).

3.1.2. Списки контроля доступа

В дополнение к правам доступа к сущностям в ОС поддерживаются списки контроля доступа ACL (Access Control List), реализованные на основе расширенных атрибутов файловых систем. ACL состоит из набора записей, каждая из которых описывает права доступа. Использование ACL позволяет задавать права доступа индивидуально для любых пользователей и/или групп пользователей, дополняя и расширяя базовый механизм «пользователь-владелец — группа-владелец — все остальные». С помощью ACL доступ может быть предоставлен пользователю или группе независимо от доступа других пользователей, участия в группах и предоставления доступа к иным сущностям.

Дополнительно ACL предоставляет:

- механизм масок, позволяющий ограничивать права доступа;
- механизм наследования ACL (ACL по умолчанию), позволяющий определять порядок присвоения прав доступа вновь создаваемым сущностям.

Права доступа к сущности для пользователя-владельца, группы-владельца и всех остальных имеют соответствующее представление в ACL в виде отдельных записей. Таким образом, каждой сущности всегда сопоставляется минимальный ACL, включающий три записи: для пользователя-владельца, группы-владельца и всех остальных.

При использовании минимального ACL:

- стандартные права доступа для класса пользователь-владелец (например, `rwx`) отображаются в идентичные права доступа записи ACL типа пользователь-владелец (например, `user::rwx`);
- стандартные права доступа для класса группа-владелец (например, `rw-`) отображаются в идентичные права доступа записи ACL типа группа-владелец (например, `group::rw-`);
- стандартные права доступа для класса все остальные (например, `r--`) отображаются в идентичные права доступа записи ACL типа все остальные (например, `other::r--`).

В дополнение к минимальному ACL могут быть указаны записи для прав доступа других именованных субъектов, а также запись для маски доступа. ACL, содержащий более трех записей, называется расширенным ACL.

В общем случае расширенный ACL включает записи следующих типов:

- пользователь-владелец (текстовое представление `user::rwx`);
- именованный пользователь (текстовое представление `user:<пользователь>:rwx`);
- группа-владелец (текстовое представление `group::rwx`);
- именованная группа (текстовое представление `user:<группа>:rwx`);
- маска доступа (текстовое представление `mask::rwx`);
- все остальные (текстовое представление `other::rwx`).

Запись для маски доступа появляется добавлении прав для именованных пользователей и/или именованных групп в списке прав доступа. Если маска не была указана в явном виде, то в записи будет отображаться эффективная маска, представляющая собой сумму прав доступа группы-владельца, именованных пользователей и именованных групп.

Маска доступа, указанная в явном виде, может использоваться для ограничения прав доступа группы-владельца, именованных пользователей и именованных групп. На права пользователя-владельца и остальных пользователей маска не влияет. В маске указываются максимальные эффективные права, которые получат группа-владелец, именованные пользователи и именованные группы, независимо от того, какие именно права были для них заданы. Например, если в маске указано `r-x`, то группа-владелец, именованные пользователи и именованные группы будут иметь доступ на чтение и исполнение, но не получат доступ на запись, даже если он будет явно указан в их записях ACL.

Реализованная в механизме дискреционных ПРД проверка прав доступа субъекта к сущности (файлу/каталогу) выполняется в два этапа.

На первом этапе выбирается запись ACL, соответствующая субъекту. Дискреционные ПРД для сущностей просматриваются в следующем порядке, при этом решение о доступе принимается только на основе одной выбранной записи ACL:

- права пользователя-владельца;
- права именованных пользователей;
- права группы-владельца;
- права именованных групп;
- права для всех остальных.

На втором этапе проверяется, что запись ACL содержит права на вид доступа, который запрашивает субъект.

Алгоритм проверки прав доступа имеет следующий вид:

1) проверяется, является ли субъект доступа пользователем-владельцем сущности. Если это так, то проверяется, разрешен ли в соответствии с выбранной записью ACL запрошенный субъектом вид доступа. По результатам проверки доступ либо разрешается, либо запрещается. Если субъект доступа не является пользователем-владельцем сущности, проверка продолжается;

2) проверяется, является ли субъект доступа одним из именованных пользователей, указанных в записях ACL. Если это так, то проверяется, разрешен ли в соответствии с выбранной записью ACL и записью маски доступа ACL запрошенный субъектом вид доступа. По результатам проверки доступ либо разрешается, либо запрещается. Если субъект доступа не является именованным пользователем, указанным в записях ACL, проверка продолжается;

3) проверяется, является ли одна из групп субъекта доступа группой-владельцем сущности. Если это так, то проверяется, разрешен ли в соответствии с выбранной записью ACL запрошенный субъектом вид доступа. По результатам проверки доступ либо разрешается, либо запрещается. Если ни одна из групп субъекта доступа не является группой-владельцем сущности, проверка продолжается;

4) проверяется, является ли одна из групп субъекта доступа одной из именованных групп, указанных в записях ACL. Если это так, то проверяется, разрешен ли в соответствии с выбранной записью ACL и записью маски доступа ACL запрошенный субъектом вид доступа. По результатам проверки доступ либо разрешается, либо запрещается. Если ни одна из групп субъекта доступа не является именованной группой, указанной в записях ACL, проверка продолжается;

5) проверяется, разрешен ли в соответствии с записью ACL для всех остальных запрошенный субъектом вид доступа. По результатам проверки доступ либо разрешается, либо запрещается;

6) если доступ не был разрешен при проведении предыдущих проверок, то он запрещается.

Дискреционные ПРД сущностей, как стандартные для трех классов (пользователя-владельца, группы-владельца и всех остальных), так и указанные в записях ACL, могут изменяться следующими субъектами:

- суперпользователем `root` (пользователь с `uid=0`, по умолчанию заблокирован);
- администратором с использованием механизма `sudo`;
- пользователем-владельцем сущности.

Изменение стандартных прав доступа пользователя-владельца, группы-владельца и всех остальных автоматически приводит к изменению соответствующих прав в записи ACL. Аналогичным образом, смена прав в записях минимального ACL автоматически изменяет стандартные права.

При использовании для сущности расширенного ACL изменение дискреционных ПРД обеспечивается следующим образом:

1) при изменении стандартных прав доступа они идентично изменяются в соответствующих записях ACL;

2) при изменении в ACL прав доступа пользователя-владельца, группы-владельца и всех остальных они идентично изменяются в стандартных правах;

3) при задании в ACL прав доступа именованных пользователей и/или именованных групп в стандартных правах для группы-владельца будет отображаться эффективная маска ACL;

4) при явном задании маски ACL она будет отображаться в стандартных правах группы-владельца.

Если для сущности назначен расширенный ACL, то в выводе команды:

```
ls -l
```

в конце записи стандартных прав доступа присутствует символ «+».

Пример

```
-rw-r--r-x+ 1 user user 20 авг 28 11:44 1.txt
```

Каталогам дополнительно возможно добавить записи ACL, которые будут являться ACL по умолчанию. ACL по умолчанию применяется для назначения создаваемыми внутри каталога сущностям заданный ACL. ACL по умолчанию не применяется к самому каталогу и к уже существующим сущностям. Вновь создаваемые каталоги будут наследовать ACL по умолчанию.

ACL по умолчанию включает в себя тот же набор записей, что и обычный ACL:

- пользователь-владелец (текстовое представление `default:user::rwx`);
- именованный пользователь (текстовое представление `default:user:<пользователь>:rwx`);
- группа-владелец (текстовое представление `default:<группа>::rwx`);
- именованная группа (текстовое представление `default:group:<группа>:rwx`);
- маска доступа (текстовое представление `default:mask::rwx`);
- все остальные (текстовое представление `default:other::rwx`).

Маска доступа добавляется автоматически при наличии в ACL по умолчанию записей для именованных пользователей и/или групп.

Просмотреть записи ACL, назначенные сущности, можно с помощью инструмента `getfacl` (см. 3.3.5). Изменение записей выполняется инструментом `setfacl` (см. 3.3.6).

Таким образом, реализованный в ОС механизм, регулирующий принцип дискреционного управления доступом, предусматривает санкционированное изменение дискреционных ПРД, включая санкционированное изменение списка субъектов и списка защищаемых сущностей.

3.2. Linux-привилегии

Linux-привилегии используются для предоставления отдельным пользователям прав выполнения определенных административных действий и являются стандартными для системы Linux.

К Linux-привилегиям относятся: `CAP_CHOWN`, `CAP_DAC_OVERRIDE`, `CAP_DAC_READ_SEARCH`, `CAP_FOWNER`, `CAP_FSETID`, `CAP_KILL`, `CAP_SETGID`, `CAP_SETUID`, `CAP_SETPCAP`, `CAP_LINUX_IMMUTABLE`, `CAP_NET_BIND_SERVICE`, `CAP_NET_BROADCAST`, `CAP_NET_ADMIN`, `CAP_NET_RAW`, `CAP_IPC_LOCK`, `CAP_IPC_OWNER`, `CAP_SYS_MODULE`, `CAP_SYS_RAWIO`, `CAP_SYS_CHROOT`, `CAP_SYS_PTRACE`, `CAP_SYS_PACCT`, `CAP_SYS_ADMIN`, `CAP_SYS_BOOT`, `CAP_SYS_NICE`, `CAP_SYS_RESOURCE`, `CAP_SYS_TIME`, `CAP_SYS_TTY_CONFIG`, `CAP_MKNOD`, `CAP_LEASE`.

Linux-привилегии наследуются процессами от своих «родителей». Процессы, запущенные от имени учетной записи пользователя, наследуют его привилегии.

Процессы, запущенные от имени суперпользователя, независимо от наличия у них привилегий, обладают всеми предоставляемыми ими возможностями.

При запуске процесса с установленными привилегиями загрузчик динамических библиотек осуществляет сброс переменных среды окружения, позволяющих осуществлять загрузку динамических библиотек из нестандартных каталогов `LD_LIBRARY_PATH` и `LD_PRELOAD`. Таким образом, установка Linux-привилегий для пользователя может привести к невозможно-

сти запуска приложений, использующих динамическую загрузку библиотек из нестандартных каталогов (например, каталогов программного обеспечения Firefox, Thunderbird, LibreOffice, `fly-scan`).

Для администрирования ОС с включенным мандатным управлением доступом множество привилегий Linux расширено специальными PARSEC-привилегиями, описание которых приведено в 4.7. Для настройки КСЗ могут использоваться как Linux-, так и PARSEC-привилегии. Порядок управления привилегиями описан в 4.16.

3.3. Средства управления дискреционными ПРД

Для управления дискреционными ПРД используется графическая утилита `fly-fm` («Менеджер файлов», см. электронную справку).

Для управления Linux-привилегиями пользователей системы используется модуль «Пользователи» в разделе «Пользователи и группы» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_users
```

Описание инструментов управления дискреционными ПРД в режиме командной строки приведено в 3.3.1–3.3.6

3.3.1. chown

Инструмент `chown` позволяет изменять пользователя и/или группу, владеющих сущностями.

Синтаксис команды:

```
chown [параметр]... <ВЛАДЕЛЕЦ>[:[ГРУППА]] <ФАЙЛ>...
chown [параметр]... <:ГРУППА> <ФАЙЛ>...
chown [параметр]... --reference=RFILE <ФАЙЛ>...
```

Смена пользователя-владельца и/или группы-владельца производится согласно заданным параметрам, интерпретируемым в порядке указания следующим образом:

1) если задано только имя пользователя (или его числовой идентификатор), то данный пользователь становится владельцем каждого из указанных файлов, а группа-владелец этих файлов не изменяется;

2) если за именем пользователя через двоеточие следует имя группы (или числовой идентификатор группы) без пробелов между ними, то изменяется также и группа-владелец файлов;

3) если двоеточие или точка следует за именем пользователя, но группа не задана, то данный пользователь становится владельцем указанных файлов, а группа-владелец указанных файлов изменяется на основную группу пользователя;

4) если не указано имя пользователя, а двоеточие или точка вместе с группой заданы, то будет изменена только группа-владелец указанных файлов. В этом случае `chown` выполняет ту же функцию, что и `chgrp` (см. 3.3.2).

Описание основных параметров команды приведено в таблице 2.

Таблица 2

Параметр	Описание
`-c`, `--changes`	Подробно описывать только файлы, чей владелец изменяется. Аналогичен параметру `--verbose`
`--dereference`	Изменить владельца файла, на который указывает символьная ссылка, вместо самой символьной ссылки
`-h`, `--no-dereference`	Работать с самими символьными ссылками, а не с файлами, на которые они указывают. Данный параметр доступен, только если имеется системный вызов `lchown`
`--from=CURRENT_OWNER:CURRENT_GROUP`	Изменить владельца и/или группу каждого файла, только если текущий владелец и/или группа совпадает с `CURRENT_OWNER:CURRENT_GROUP`. Как группа, так и владелец могут не указываться, в этом случае совпадение для данного атрибута не обязательно
`-f`, `--silent`, `--quiet`	Не выводить сообщения об ошибках на файлы, чей владелец не может быть изменен
`--reference=RFILE`	Установить такого же владельца и группу, как и у файла-образца `RFILE`. Если `RFILE` является символической ссылкой, будут использованы данные файла, на который она ссылается
`-R`, `--recursive`	Рекурсивно изменять владельца каталогов и всего их содержимого
`-v`, `--verbose`	Подробно описывать только файлы, чей владелец изменяется
`--help`	Вывести справку и выйти
`--version`	Вывести информацию о версии и выйти

Примеры:

1. Назначить пользователя `user` и группу `admins` пользователем-владельцем и группой-владельцем всех сущностей в каталоге `/var/www`:

```
chown -R user:admins /var/www
```

2. Назначить группу `group1` группой-владельцем файла `example.txt`, не изменяя пользователя-владельца:

```
chown: group1 example.txt
```

3. Назначить пользователя `user` и его основную группу пользователем-владельцем и группой-владельцем каталога `/usr/bin`:

```
chown user: /usr/bin
```

3.3.2. chgrp

Инструмент `chgrp` изменяет группу, владеющую каждым из указанных файлов `FILE`, на указанную группу. Группа может быть задана именем или ее числовым идентификатором.

Синтаксис команды:

```
chgrp [параметр]... <ГРУППА> <ФАЙЛ>...
chgrp [параметр]... --reference=RFILE <ФАЙЛ>...
```

Описание параметров инструмента `chgrp` приведено в таблице 3.

Таблица 3

Параметр	Описание
`-c`, `--changes`	То же, что и `--verbose`, но выводить сообщение только тогда, когда действительно была изменена группа файла
`--dereference`	Изменить владельца файла, на который указывает символьная ссылка, вместо самой символьной ссылки
`-h`, `--no-dereference`	Изменить владельца символьной ссылки, а не владельца файла, на который указывает эта ссылка (доступна только на системах, имеющих системный вызов `lchown`)
`-f`, `--silent`, `--quiet`	Не выводить сообщения об ошибках
`--reference=RFILE`	Установить такую же группу, как и у файла-образца `RFILE`. Если `RFILE` является символической ссылкой, будут использованы данные файла, на который она ссылается
`-R`, `--recursive`	Рекурсивно изменять владельца каталогов и их содержимого
`-v`, `--verbose`	Подробно описывать изменение владельца для каждого файла
`--help`	Вывести справку и выйти
`--version`	Вывести информацию о версии и выйти

Пример

Изменить группу-владельца файла example.txt на группу admins:

```
chgrp admins example.txt
```

3.3.3. chmod

Инструмент chmod применяется для изменения прав доступа к сущностям.

Синтаксис команды:

```
chmod [параметр]... <РЕЖИМ>[,РЕЖИМ]... <ФАЙЛ>...
chmod [параметр]... <ОКТЕТ-РЕЖИМ>... <ФАЙЛ>...
chmod [параметр]... --reference=RFILE <ФАЙЛ>...
```

Описание параметров инструмента chmod приведено в таблице 4.

Таблица 4

Параметр	Описание
-c, --changes	То же, что и --verbose, но выводить сообщение только тогда, когда были произведены изменения
-f, --silent, --quiet	Не выдавать сообщения об ошибках на те файлы, чьи права не могут быть изменены
-v, --verbose	Подробно описывать измененные права доступа
--reference=RFILE	Изменить права доступа к файлу на те права, что имеет RFILE
-R, --recursive	Рекурсивное изменение прав доступа для каталогов и их содержимого
--help	Вывести справку и выйти
--version	Вывести информацию о версии и выйти

Инструмент chmod изменяет права доступа к сущности ФАЙЛ в соответствии с правами доступа, указанными в параметре РЕЖИМ (символьный вид) или в параметре ОКТЕТ-РЕЖИМ (восьмеричный вид), представляющего новые права доступа.

В таблице 5 представлены возможные варианты записи прав пользователя.

Таблица 5

Восьмеричная запись	Биты	Символьная запись	Права на файл	Права на каталог
0	000	---	Нет	Нет
1	001	--x	Исполнение	Чтение свойств файлов

Окончание таблицы 5

Восьмеричная запись	Биты	Символьная запись	Права на файл	Права на каталог
2	010	-w-	Запись	Нет
3	011	-wx	Запись и исполнение	Все права, кроме получения имен файлов
4	100	r--	Чтение	Чтение имен файлов
5	101	r-x	Чтение и исполнение	Чтение файлов и их свойств
6	110	rw-	Чтение и запись	Чтение имен файлов
7	111	rwx	Все права	Все права

Права доступа в восьмеричном виде (параметр OKTET-РЕЖИМ) задаются четырьмя восьмеричными цифрами от 0 до 7. Для установки комбинированного доступа (например, чтение и запись) необходимо сложить соответствующие цифры (4+2=6). Набор данных четырех слагаемых дают уникальное число, позволяющее однозначно определить набор прав доступа.

Порядок подстановки цифр прав доступа в команде:

1) первая цифра в параметре OKTET-РЕЖИМ определяет наличие специальных атрибутов:

а) бит пользователя SUID задается цифрой 4;

б) бит группы SGID задается цифрой 2;

в) Sticky-бит задается цифрой 1.

При комбинировании этих атрибутов значения складываются. Например, для установки Sticky-бита и SGID необходимо использовать цифру 3 (1+2). При их отсутствии указывается цифра 0 или не указывается ничего;

2) вторая цифра определяет права доступа для пользователя, владеющего данной сущностью. Цифра от 0 до 7 определяет точный набор предоставляемых прав;

3) третья цифра определяет права доступа для группы, владеющей сущностью. Права доступа задаются тем же образом, что и для пользователя-владельца сущности;

4) четвертая цифра задает права доступа для всех остальных пользователей, которые не входят в группу-владельца сущности. Права доступа задаются тем же образом, что и для пользователя-владельца сущности.

Команда:

```
chmod 0755 <имя_файла>
```

идентична команде:

```
chmod 755 <имя_файла>
```

Данные команды задают права, расшифровка которых приведена в таблице 6.

Таблица 6

Варианты записи	Специальный бит	Пользователь-владелец (u)	Группа-владелец (g)	Остальные (o)
Восьмеричная запись	0 или пропущен	7	5	5
Символьная запись	-	rwx	r-x	r-x

В восьмеричном виде файлу или каталогу в явном виде задаются права сразу для всех трех категорий субъектов доступа. В символьном виде можно установить права для отдельных категорий, не затрагивая остальные, а также добавить или отнять права относительно уже имеющихся.

Формат символьного вида параметра РЕЖИМ:

```
[ugoa][[+-=][rwxXstugo]...][,...]
```

Каждый аргумент — это список символьных команд изменения прав доступа, разделенных запятыми. Каждая такая команда может начинаться с комбинации букв u, g, o или a, которые указывают, чьи права доступа к файлу будут изменены:

- пользователя, владеющего файлом (u);
- группы, владеющей файлом (g);
- остальных пользователей, не входящих в данную группу (o);
- всех пользователей (a).

Буква a эквивалентна одновременному указанию u, g и o. Если не задана ни одна буква, то автоматически будет использоваться буква a, но биты, установленные в umask, не будут затронуты.

Оператор «+» добавляет указанные права доступа к уже имеющимся у каждого файла, оператор «-» удаляет эти права, а оператор «=» заменяет имеющиеся права доступа на указанные.

Описание параметров символьного вида РЕЖИМ приведено в таблице 7.

Таблица 7

Параметр	Описание
u	Пользователь (владелец файла) — от user (пользователь)
g	Группа — от group (группа)

Окончание таблицы 7

Параметр	Описание
o	Остальные пользователи — от other (остальные)
a	Все пользователи — от all (все)
+	Добавить разрешения к текущим правам доступа
-	Удалить разрешения из текущих прав доступа
=	Установить разрешения вне зависимости от текущих прав доступа

Буквы r, w, x, X, s, t, u, g и/или o, идущие после оператора «+», «-» или «=», устанавливают новые права доступа для субъекта доступа, заданного одной из букв u, g, o или a. Описание параметров символьной записи приведено в таблице 8.

Таблица 8

Параметр	Описание
r	Чтение начинается с нуля или более букв
w	Запись
x	Исполнение
X	Исполнение, только если файл является каталогом или уже имеет право на исполнение для какого-нибудь пользователя
s	SUID или SGID-биты
t	Sticky-бит
u	Установка таких же прав доступа, как и у пользователя-владельца
g	Установка таких же прав доступа, как и у группы-владельца
o	Установка таких же прав доступа, которые имеют остальные пользователи (не входящие в группу-владельца файла)

Примеры:

1. Снять бит SGID:

```
chmod g-s file
```

2. Установить биты SUID и SGID:

```
chmod ug+s file
```

3. Команда не выполни никаких действий (противоречивый набор параметров):

```
chmod o+s file
```

Инструмент chmod не изменяет права на символьные ссылки, т. к. этого не делает системный вызов chmod (права символьных ссылок в системе не используются). Если в параметрах командной строки указаны символьные ссылки, то chmod изменит права сущностей, на которые они ссылаются. В то же время символьные ссылки, встречающиеся при рекурсивной обработке каталогов, игнорируются.

3.3.4. umask

Инструмент umask представляет собой маски режима создания файлов.

Синтаксис команды:

```
umask [-p] [-S] [маска]
```

Пользовательская маска создания файла устанавливается равной значению маска. Если маска начинается с цифры, она интерпретируется как восьмеричное число, иначе — как маска в символьном формате, аналогичном используемому в команде chmod (см. 3.3.3).

Если маска не указана или задан параметр -S, выдается текущее значение маски. Параметр -S вызывает выдачу маски в символьном формате, по умолчанию выдается восьмеричное число.

Если указан параметр -p, а маска не задана, то результат выдается в виде, который можно использовать во входной команде.

Если маска была успешно изменена или не указана, команда завершается со статусом 0, в противном случае — со статусом 1.

Команды инструмента umask распознаются и выполняются оболочкой shell.

Команду инструмента umask целесообразно включить в пользовательский файл профиля ~/.profile. Тогда она будет автоматически вызываться при входе в систему и устанавливать нужный режим доступа к создаваемым файлам и каталогам.

3.3.5. getfacl

Инструмент getfacl служит для отображения списков контроля доступа сущностей. Для каждого файла getfacl выводит имя файла, пользователя-владельца, группу-владельца и ACL. Если каталог имеет ACL по умолчанию, то getfacl выводит также ACL по умолчанию. Файлы не могут иметь ACL по умолчанию.

Синтаксис команды:

```
getfacl [-dRLP] <файл> ...
```

Формат вывода:

```
1:  # file: somedir/
2:  # owner: lisa
3:  # group: staff
4:  user::rwx
5:  user:joe:rwx               #effective:r-x
6:  group::rwx                 #effective:r-x
7:  group:cool:r-x
8:  mask:r-x
9:  other:r-x
10:  default:user::rwx
11:  default:user:joe:rwx       #effective:r-x
12:  default:group::r-x
13:  default:mask:r-x
14:  default:other:---
```

Строки 4, 6 и 9 относятся к битам стандартных прав доступа к файлу, соответственно, прав пользователя-владельца, группы-владельца и всех остальных. Они представляют собой записи базового ACL. Строки 5 и 7 являются записями ACL для именованных пользователя и группы. Строка 8 — маска эффективных прав. Если маска задана в явном виде, она ограничивает права, предоставляемые всем группам и именованным пользователям (итоговые эффективные права отображаются с отметкой `#effective`). Если права для именованных пользователей и/или групп заданы без явного указания маски, она отображается равной сумме их прав доступа и ни на что не влияет. В таком случае отметка `#effective` напротив прав именованных пользователей и групп отсутствует. Маска не влияет на права для пользователя-владельца файла и всех других. Строки 10–14 показывают ACL по умолчанию, ассоциированный с данным каталогом.

Для двух и более файлов `getfacl` выводит ACL, разделенные для удобства пустыми строками. Вывод инструмента `getfacl` может использоваться как входные данные для инструмента `setfacl` (см. 3.3.6).

Описание параметров инструмента приведено в таблице 9.

Таблица 9

Параметр	Описание
`--access`	Вывести только ACL файла
`-d, --default`	Вывести только ACL по умолчанию
`--omit-header`	Не показывать заголовок (имя файла)
`--all-effective`	Показать все эффективные права
`--no-effective`	Не показывать эффективные права

Окончание таблицы 9

Параметр	Описание
`--skip-base`	Пропускать файлы, имеющие только основные записи
`-R`, `--recursive`	Для подкаталогов рекурсивно
`-L`, `--logical`	Следовать по символическим ссылкам, по умолчанию символические ссылки, не указанные в командной строке, игнорируются
`-P`, `--physical`	Не следовать по символическим ссылкам, даже если они указаны в командной строке
`--tabular`	Использовать табулированный формат вывода
`--numeric`	Показывать числовые значения пользователя/группы
`--absolute-names`	Не удалять ведущие «/» из пути файла
`--help`	Вывести справку и выйти
`--version`	Вывести информацию о версии и выйти

3.3.6. setfacl

Инструмент `setfacl` вносит изменения в ACL файлов и каталогов.

Синтаксис команды:

```
setfacl [-bkndRLP] [-mMxX ... ] <файл> ...
```

В команде после параметров можно указать несколько файлов и/или каталогов, после которых также возможно указать новые параметры и относящиеся к ним файлы и/или каталоги. Подобных блоков, состоящих из параметров и файлов/каталогов, может быть указано любое количество.

Описание параметров инструмента приведено в таблице 10.

Таблица 10

Параметр	Описание
`-m`, `--modify=acl`	Изменить текущий ACL для файла
`-M`, `--modify-file=file`	Прочитать записи ACL для модификации из файла
`-x`, `--remove=acl`	Удалить записи из ACL файла
`-X`, `--remove-file=file`	Прочитать записи ACL для удаления из файла
`-b`, `--remove-all`	Удалить все расширенные записи ACL
`-k`, `--remove-default`	Удалить ACL по умолчанию
`--set=acl`	Установить ACL для файла, заменив текущий ACL
`--set-file=file`	Прочитать записи ACL для установления из файла
`--mask`	Пересчитать маску эффективных прав

Окончание таблицы 10

Параметр	Описание
`-n`, `--no-mask`	Не пересчитывать маску эффективных прав. Обычно `setfacl` пересчитывает маску (если она не была задана явно) и устанавливает ее равной сумме всех прав группы-владельца, именованных пользователей и групп (т.е. всех субъектов, на права которых она влияет)
`-d`, `--default`	Применить ACL по умолчанию
`-R`, `--recursive`	Для подкаталогов рекурсивно
`-L`, `--logical`	Следовать по символическим ссылкам, по умолчанию ссылки, не указанные в командной строке, игнорируются
`-P`, `--physical`	Не следовать по символическим ссылкам, даже если они указаны в командной строке
`--restore=file`	Восстановить резервную копию прав доступа, созданную командой: `getfacl -R` или подобной. Все права доступа дерева каталогов восстанавливаются, используя этот механизм. Если вводимые данные содержат записи для пользователя-владельца или группы-владельца и инструмент `setfacl` запускается с правами `root`, то пользователь-владелец и группа-владелец всех файлов также восстанавливаются. Этот параметр не может использоваться совместно с другими параметрами, за исключением параметра `--test`
`--test`	Режим тестирования (ACL не изменяются)
`--help`	Вывести справку и выйти
`--version`	Вывести информацию о версии и выйти

При использовании параметров `--set`, `-m` и `-x` должны быть перечислены записи ACL в командной строке. Записи ACL разделяются запятыми.

При чтении ACL из файла при помощи параметров `--set-file`, `-M` и `-X` инструмент `setfacl` принимает множество записей в формате вывода инструмента `getfacl`. В строке обычно содержится только одна запись ACL.

Инструмент `setfacl` использует следующие форматы записей ACL:

1) права доступа отдельного пользователя:

```
[d:] [u:]uid [:perms]
```

Если не задан `uid`, то устанавливаются права доступа пользователя-владельца файла;

2) права доступа отдельной группы:

```
[d:] g:gid [:perms]
```

Если не задан `gid`, то устанавливаются права доступа группы-владельца;

3) маска эффективных прав:

```
[d:] m[:]  [:perms]
```

4) права доступа всех остальных:

```
[d:] o[:]  [:perms]
```

Значения `uid` и `gid` задаются именем или числом. Параметр `perms` может быть представлен в символьном (комбинация `r`, `w`, `x`, `-`) или восьмеричном (цифры от `0` до `7`) виде.

ВНИМАНИЕ! При перечислении записей ACL маска всегда указывается последней.

При создании прав доступа инструментом `setfacl` выполняются следующие правила:

1) если права доступа для именованного пользователя и/или группы добавлены в ACL, а маски прав не существует, то создается маска с правами доступа группы-владельца;

2) если создана запись ACL по умолчанию, а трех базовых записей не было, тогда делается их копия и они добавляются в ACL по умолчанию;

3) если ACL по умолчанию содержит какие-либо права доступа для именованного пользователя и/или группы и не содержит маску прав доступа по умолчанию, то при создании эта маска будет иметь те же права, что и группа по умолчанию.

Примеры:

1. Задать в ACL файла `1.txt` права `rwx` для именованного пользователя `steeve`, права `-w-` для именованной группы `admins` и права `r-x` для группы-владельца файла:

```
setfacl -m u:steeve:rwx,g:admins:w,g::r-x 1.txt
```

Результат выполнения команды:

```
getfacl 1.txt:

# file: 1.txt
# owner: user
# group: user
user::rw-
user:steeve:rwx
group::r-x
group:admins:-w-
mask::rwx
other::r-x
```

2. Задать в ACL каталога `reports` и всех его подкаталогов права `r-x` для пользователя-владельца и маску `rw` для ограничения максимальных прав группы-владельца и именованных пользователей/групп, заданных ранее (они, в итоге, не получат доступ на исполнение, так как маска разрешает только чтение и запись):

```
setfacl -R -m u::r-x,m:rw reports
```

Результат выполнения команды:

```
getfacl reports:

# file: reports
# owner: user
# group: user
user::r-x
user:steeve:rwx #effective:rw-
group::r-x #effective:r--
mask::rw-
other::r-x
```

4. МАНДАТНОЕ УПРАВЛЕНИЕ ДОСТУПОМ И МАНДАТНЫЙ КОНТРОЛЬ ЦЕЛОСТНОСТИ

4.1. Общие сведения

Мандатное управление доступом и мандатный контроль целостности (МКЦ) реализованы в ядре ОС и затрагивают следующие подсистемы:

- механизмы IPC;
- стек TCP/IP (IPv4, IPv6);
- ФС ext2/ext3/ext4/XFS;
- сетевые ФС CIFS, OCFS2, Ceph;
- ФС proc, tmpfs.

4.2. Мандатное управление доступом

При реализации политики мандатного управления доступом субъектам и сущностям присваиваются классификационные метки (уровни конфиденциальности и категории конфиденциальности, описание которых приведено в 4.2.1 и 4.2.2 соответственно).

Также могут быть присвоены дополнительные атрибуты сущностей для мандатного управления доступом, описание которых приведено в 4.2.3.

Для администрирования подсистем с мандатным управлением доступом множество привилегий Linux расширено специальными PARSEC-привилегиями, полное описание которых приведено в 4.7.

4.2.1. Уровень конфиденциальности

Иерархический уровень конфиденциальности (уровень конфиденциальности) определяет степень секретности сущности и соответствующий уровень доступа к ней, назначенный субъекту.

Уровень конфиденциальности представляет собой числовое значение от 0 до 255 (включительно). Каждой классификационной метке в каждый момент времени может быть назначен только один уровень конфиденциальности.

Числовые значения уровня конфиденциальности сравнимы между собой и технически реализованы как 8-битная беззнаковая величина (`uint8_t`). В пользовательских интерфейсах представляется десятичным значением или наименованием.

Описание применения правил мандатного управления доступом на основе результатов сравнения уровней конфиденциальности субъекта и сущности приведено в 4.6.

4.2.2. Категория конфиденциальности

Множество неиерархических категорий конфиденциальности (категории конфиденциальности) определяет набор категорий, доступных субъекту или назначенных сущности.

Категории конфиденциальности представляют собой битовую маску, состоящую из набора категорий конфиденциальности, каждая из которых представляется в виде отдельного разряда битовой маски. Значение разряда битовой маски, равное 1, означает наличие соответствующей категории конфиденциальности у сущности (субъекта), а равное 0 — отсутствие данной категории.

В ОС реализовано использование до 64 категорий конфиденциальности. Каждой классификационной метке в каждый момент времени может быть присвоен единственный набор категорий конфиденциальности. Набор категорий конфиденциальности может принимать значения от 0 до 0xFFFF FFFF FFFF FFFF (включительно), технически реализован как 64-битная маска, беззнаковая величина (`uint64_t`).

Категории конфиденциальности могут быть несравнимы между собой. В пользовательских интерфейсах представляются шестнадцатеричным значением или списком наименований категорий.

Описание применения правил мандатного управления доступом на основе результатов сравнения категорий конфиденциальности субъекта и сущности приведено в 4.6.

4.2.3. Дополнительные атрибуты сущностей для мандатного управления доступом

Дополнительные атрибуты сущностей для мандатного управления доступом позволяют уточнять или изменять правила мандатного управления доступом для тех или иных сущностей:

- `ccnr` — присваивается контейнерам. Определяет, что контейнер может содержать сущности с различными классификационными метками, но не большими, чем его собственная классификационная метка. Чтение содержимого такого контейнера разрешается субъекту вне зависимости от значения его классификационной метки, при этом субъекту доступна информация только про находящиеся в этом контейнере сущности с классификационной меткой не большей, чем его собственная классификационная метка, либо про сущности-контейнеры, также имеющие атрибут `ccnr`;
- `ehole` — присваивается объектам (файлам), имеющим минимальную классификационную метку и нулевую метку целостности. Приводит к игнорированию мандатных правил управления доступом при получении доступа на запись к данным объекта. Атрибут предназначен для объектов, из которых субъект не может прочитать данные, записанные в них субъектами с более высокой классификационной меткой, чем его собственная классификационная метка (например, `/dev/null`);

- `whole` — присваивается объектам (файлам), имеющим максимальную классификационную метку. Разрешает запись в них субъектам, имеющим более низкую классификационную метку (в обычном случае записывать «снизу вверх» запрещено).

Примечание. Атрибут `ccnri` более не используется в ОС для управления доступом, при этом штатное функционирование ОС соответствует функционированию с установленным атрибутом `ccnri`. Возможность установки и получения данного атрибута сохранена в ОС для обеспечения совместимости с системами и ПО, которые его использует. Атрибут `ccnra` в ОС приравнивается к атрибуту `ccnr` и также сохранен для обеспечения совместимости и не рекомендован к использованию.

Дополнительные атрибуты сущностей для мандатного управления доступом могут быть установлены (сняты) только субъектом с наличием привилегии `PARSEC_CAP_CHMAC` (см. 4.7) или субъектом с привилегией `root` и с максимальной меткой целостности (такой субъект обладает привилегией `PARSEC_CAP_CHMAC` по умолчанию).

4.3. Мандатный контроль целостности

При реализации политики мандатного контроля целостности субъектам и сущностям задаются метки целостности (неиерархический уровень (категория) целостности и иерархический (линейный) уровень целостности), описание метки целостности приведено в 4.3.1.

Также сущностям могут быть присвоены дополнительные атрибуты для мандатного контроля целостности, описание которых приведено в 4.3.2.

Для администрирования подсистем с мандатным контролем целостности множество привилегий Linux расширено специальными PARSEC-привилегиями, полное описание которых приведено в 4.7.

4.3.1. Метка целостности

Субъектам и сущностям задаются метки целостности, каждая из которых представляет собой пару: неиерархический уровень целостности (категория целостности) и иерархический уровень целостности (линейный уровень целостности).

Метка целостности сущности отражает степень критичности для программной среды ОС факта изменения этой сущности и, соответственно, степень защищенности сущности от изменений субъектами. Чем больше метка целостности сущности, тем более защищена сущность от изменений субъектами.

Метка целостности субъекта соответствует его полномочиям по доступу к сущности. Чем больше метка целостности у субъекта (пользователя или процесса), тем большими возможностями по изменению сущностей он обладает.

Категория целостности технически реализована как 32-битная маска, беззнаковая величина (`uint32_t`). В пользовательских интерфейсах представляется десятичным или шестнадцатеричным числом или наименованием.

В ОС по умолчанию выделены нулевая категория целостности, четыре ненулевых и несравнимых между собой (далее — изолированных) категорий целостности, а также максимальная категория целостности, которая не меньше всех остальных в системе.

При установке ОС по умолчанию предлагается максимальная категория целостности `max_ilev`, равная `63` (битовая маска 00111111), и минимальная категория целостности, равная нулю.

Линейный уровень целостности реализован в виде числа со знаком длиной 1 байт и диапазоном значений от -128 до 127. Значение по умолчанию равно `0`. Отрицательные значения введены для того, чтобы субъекты с линейным уровнем целостности `0` могли запускать недоверенные процессы с линейным уровнем целостности ниже собственного (отрицательным). В отличие от категорий целостности, которые задаются битовой маской и могут быть несравнимы между собой, любые два линейных уровня целостности сравнимы между собой, так как задаются целым числом.

В графическом интерфейсе системы и в консоли категория целостности обозначается термином «Уровень целостности» с двумя зарезервированными значениями: `Высокий` (`High`) для максимальной категории целостности и `Низкий` (`Low`) для минимальной категории целостности.

Метка целостности может быть назначена пользователю или группе пользователей. Метка целостности пользователя указывается в `/etc/parsec/micdb` (для локальной учетной записи) или в базе FreeIPA (для доменной учетной записи). Метка целостности для группы пользователей (локальной или доменной) указывается в `/etc/parsec/micgrdb`.

ВНИМАНИЕ! Если пользователю назначена метка целостности, то она имеет приоритет над меткой целостности группы, в которую входит данный пользователь (т. е. метка целостности группы в таком случае не применяется).

Если пользователю не назначена метка целостности, то при входе в сессию ему присваивается метка целостности группы, в которую он входит.

На одну группу может быть назначена только одна метка целостности, при этом пользователь может состоять в нескольких группах. В таком случае при входе пользователя в сессию вычисляется его эффективная метка целостности на основе меток целостности всех групп, в которых он состоит. Эффективная категория целостности представляет собой битовую маску, в которой каждый бит устанавливается равным 1, если он равен 1 в категории целостности хотя бы одной из групп (побитовое «ИЛИ»). Если в категориях целостности всех групп бит

равен нулю, то в эффективной категории целостности он также устанавливается равным нулю. Линейный уровень целостности всегда принимается равным нулю.

Непривилегированным пользователям, которым явно не назначена метка целостности и у которых отсутствует метка целостности группы, неявно присваивается нулевая метка целостности.

Администратору, создаваемому при установке ОС, присваивается максимальная категория целостности 63.

За системными службами, перечень и описание которых приведены в таблице 11, зарезервированы четыре изолированные категории целостности.

Таблица 11

Категория	Значение	Битовая маска	Описание
1	001	0000 0001	Категория задействована для сетевых служб
2	002	0000 0010	Категория задействована для виртуализации
3	004	0000 0100	Категория задействована для специального ПО
4	008	0000 1000	Категория задействована для графического сервера

После установки ОС максимальная категория целостности в системе может быть повышена. Максимальными категориями целостности в системе могут быть числа, у которых битовая маска включает битовые маски всех остальных используемых категорий целостности в системе, например, 63 (0x3F, битовая маска 00111111), 127 (0x7F, битовая маска 01111111), 191 (0xBF, битовая маска 10111111) и т. д.

Примечание. В текущей реализации, с учетом 32-битной маски, количество изолированных категорий целостности может быть увеличено до 32 при повышении максимальной категории целостности до 0xFFFF FFFF.

ВНИМАНИЕ! При повышении максимальной категории целостности выше значения 63, заданного при установке ОС, необходимо убедиться в повышении категории целостности администратора ОС.

Описание применения правил мандатного контроля целостности на основе результатов сравнения меток целостности субъекта и сущности приведено в 4.6.

4.3.2. Дополнительные атрибуты сущностей для мандатного контроля целостности

Дополнительные атрибуты сущностей для МКЦ позволяют уточнять или изменять правила МКЦ для тех или иных сущностей:

- `silev` (числовое значение `0x10`) — присваивается исполняемым бинарным файлам. Данный атрибут позволяет пользователю с низкой меткой целостности, который не является суперпользователем (`root`, `uid=0`), запускать процесс с высокой меткой целостности из файла с высокой меткой целостности. При запуске процесса из файла с атрибутом `silev` процессу присваивается метка целостности, равная наибольшему значению, которое одновременно меньше или равно значению метки целостности файла и значению максимальной метки целостности в системе (параметр командной строки ядра `parsec.max_ilev` и нулевой линейный уровень целостности). Например, если метка целостности файла меньше или равна максимальной метке целостности в системе, то процессу назначается метка целостности, равная метке целостности файла. В частности, атрибут `silev` нужен, чтобы пользователь с низкой меткой целостности мог запустить из файла `/usr/bin/passwd`, имеющего высокую метку целостности, процесс смены пароля с высокой меткой целостности.

  ВНИМАНИЕ! Использовать атрибут рекомендуется в исключительных случаях в соответствии с принятой политикой безопасности;

- `irelax` (числовое значение `0x20`) — присваивается каталогам. Данный атрибут определяет, что в каталог может осуществлять запись процесс с любой меткой целостности. Создаваемым в данном каталоге сущностям назначается метка целостности, равная наибольшему значению, которое одновременно меньше или равно значениям метки целостности создающего их процесса и метки целостности данного каталога (результирующая метка целостности вычисляется как совпадающие биты категорий целостности и минимальное значение из линейных уровней целостности каталога и процесса). Атрибут применяется при включенном расширенном режиме МКЦ (см. 4.5);

- `ssi` (числовое значение `0x40`) — присваивается файлам и каталогам для ограничения доступа на чтение и выполнение. При наличии данного атрибута у сущности доступ к ней на чтение и выполнение разрешается только субъектам (процессам) с метками целостности не ниже, чем у данной сущности. При этом если исполняемому бинарному файлу также присвоен атрибут `silev`, то ограничение доступа на выполнения к данному файлу не применяется;

- `iinh` (числовое значение `0x80`) — присваивается каталогам. Данный атрибут позволяет создаваемым в каталоге сущностям наследовать метку целостности родительского каталога (создаваемым в нем сущностям присваивается метка целостности каталога), при этом создаваемые в данном каталоге дочерние каталоги также наследуют атрибут `iinh`. Метка целостности создающего процесса должна

быть не меньше метки целостности каталога. Атрибут применяется при включенном расширенном режиме МКЦ (см. 4.5).

ВНИМАНИЕ! Если каталогу одновременно присвоены атрибуты `iinh` и `irelax`, то создаваемым в нем сущностям присваивается метка целостности по правилам работы атрибута `irelax`, т. к. `irelax` обладает приоритетом над `iinh`.

Дополнительный атрибут сущностей для МКЦ `silev` может быть установлен (снят) только субъектом с правами `root` и с максимальной меткой целостности.

Дополнительные атрибуты сущностей для МКЦ `irelax`, `ssi` и `iinh` могут быть установлены (сняты) субъектом с меткой целостности не ниже, чем у соответствующих сущностей.

4.4. Мандатный контекст безопасности

Мандатный контекст безопасности субъекта определяется его меткой безопасности совместно с назначенными ему привилегиями.

Мандатный контекст безопасности сущности определяется меткой безопасности сущности совместно с присвоенными ей дополнительными атрибутами сущностей для мандатного управления доступом и для МКЦ.

Метка безопасности состоит из классификационной метки и метки целостности:

1) классификационная метка, определяется:
   а) иерархическим уровнем конфиденциальности;
   б) неиерархическими категориями конфиденциальности;
2) метка целостности, определяется:
   а) иерархическим (линейным) уровнем целостности;
   б) неиерархическим уровнем (категорией) целостности.

Субъекты или сущности, которым явно не задан мандатный контекст безопасности, считаются имеющими нулевой мандатный контекст безопасности, т. е. имеют равный нулю уровень конфиденциальности, отсутствуют категории конфиденциальности, имеют равный нулю линейный уровень целостности и равную нулю категорию целостности, отсутствуют привилегии и дополнительные атрибуты.

Порядок присвоения мандатного контекста безопасности и правила принятия решения о предоставлении доступа на его основе описаны в 4.6.

4.5. Расширенный режим мандатного контроля целостности

При включенном МКЦ (без расширенного режима МКЦ) процесс при его непосредственном запуске наследует метку целостности процесса-родителя (процесс наследует метку целостности запустившего его пользователя).

В расширенном режиме МКЦ (strict mode) непосредственный запуск процесса запрещен в том случае, если исполняемый файл, из которого запускается процесс, имеет метку целостности меньше или несравнимую с меткой целостности процесса-родителя. В этом случае процесс возможно запустить только с использованием инструмента `sumic`, описание которого приведено в 4.15.10.

При создании сущности ей назначается нулевая категория целостности и линейный уровень целостности, равный минимальному значению из линейного уровня целостности родительского каталога и нуля.

При наличии у родительского каталога атрибутов `irelax` и/или `iinh` для назначения метки целостности создаваемой сущности используются правила, описанные в 4.3.2.

Если процесс, создающий сущность, обладает привилегией `PARSEC_CAP_INHERIT_INTEGRITY` (см. 4.7), то создаваемой сущности назначается метка целостности, равная метке целостности родительского каталога. При этом запрещено создавать сущность с меткой целостности выше или несравнимой с меткой целостности процесса, создающего данную сущность.

ВНИМАНИЕ! Если родительскому каталогу присвоен атрибут `irelax`, то создаваемым в нем сущностям присваивается метка целостности по правилам работы атрибута `irelax` (см. 4.3.2), т. к. `irelax` обладает приоритетом над привилегией `PARSEC_CAP_INHERIT_INTEGRITY`.

ВНИМАНИЕ! Расширенный режим МКЦ предназначен для усиления защиты ОС. Включение расширенного режима МКЦ необратимо, после включения он не может быть выключен. Вместе с тем его включение может привести к сбоям в работе некоторого прикладного ПО (особенно уже установленного). Поэтому перед включением расширенного режима МКЦ в ОС администратору рекомендуется провести тестирование используемого прикладного ПО с целью проверки его работоспособности при включенном расширенном режиме МКЦ и необходимости его дополнительной настройки.

Включение расширенного режима МКЦ осуществляется запуском от имени администратора команды:

```
astra-strictmode-control enable
```

Выключение расширенного режима МКЦ не предусмотрено.

После включения расширенного режима МКЦ будут выполнены необходимые настройки метки целостности сущностей, для параметра командной строки ядра `parsec.strict_mode` будет установлено значение `1`. Для активации расширенного режима МКЦ необходимо перезагрузить ОС.

Для проверки текущего состояния расширенного режима МКЦ (активен/неактивен) можно воспользоваться командой:

```
astra-strictmode-control status
```

Если после выполнения команды включения расширенного режима МКЦ не выполнялась перезагрузка ОС, то результат команды проверки состояния режима будет `НЕАКТИВНО`. Для получения информации о состоянии расширенного режима МКЦ, которое будет после перезагрузки ОС, выполнить команду:

```
astra-strictmode-control is-enabled
```

При включении расширенного режима МКЦ домашним каталогам (и их содержимому) пользователей, для которых задана ненулевая метка целостности, присваивается метка целостности, соответствующая максимально доступной метке целостности для данного пользователя. В дальнейшем при назначении пользователю ненулевой метки целостности также следует его домашнему каталогу назначить метку целостности, соответствующую максимально доступной метке целостности для данного пользователя. Если домашнему каталогу пользователя не будет назначена метка целостности, то после перезагрузки ОС она будет назначена автоматически.

При включенном расширенном режиме МКЦ применяются следующие изменения в работе ОС:

1) запрещен запуск процесса, если исполняемый файл, из которого запускается процесс, имеет метку целостности меньше или несравнимую с меткой целостности процесса-родителя (в данном случае для запуска необходимо использовать `sumic`, см. 4.15.10);

2) не применяется параметр командной строки ядра `parsec.ccnr_relax` (см. 4.18);

3) возможно применение привилегии `PARSEC_CAP_CCNR_RELAX` (см. 4.7);

4) возможно применение атрибутов `irelax` и `iinh` (см. 4.3.2).

4.6. Применение правил мандатного управления доступом и мандатного контроля целостности

Использование уровня и категорий конфиденциальности обеспечивает защиту от несанкционированного доступа к информации.

Использование метки целостности обеспечивает целостность информации путем запрета модификации сущностей с высокой меткой целостности недоверенными субъектами с более низкой меткой целостности.

Порядок задания мандатных атрибутов системы и присвоения мандатного контекста безопасности пользователям приведен в разделе 17.

Управление контекстом безопасности сущностей осуществляется с помощью инструмента `pdpl-file` в соответствии с 4.15.1.

Классификационная метка субъекта/сущности наследуется от создающего его/ее субъекта, т. е. процесс наследует классификационную метку процесса-родителя, а файл или каталог наследует классификационную метку процесса, создающего данный файл или каталог.

Классификационная метка сущности не может превышать значение классификационной метки контейнера, в котором она расположена.

Метка целостности субъекта наследуется от создающего его субъекта, т. е. процесс при его непосредственном запуске наследует метку целостности процесса-родителя. При этом при включенном расширенном режиме МКЦ субъект не может запустить процесс из файла, метка целостности которого меньше или несравнима с его собственной (см. 4.5).

Метка целостности сущности не наследуется от создающего ее субъекта. При создании сущности ей назначается нулевая категория целостности и линейный уровень целостности, являющийся минимальным значением из линейного уровня целостности каталога, в котором создается сущность, и нуля. При этом метка целостности субъекта, создающего сущность, должна быть не ниже метки целостности каталога, в котором создается сущность.

Порядок наследования меток целостности при запуске процессов или создании сущностей может быть изменен путем применения PARSEC-привилегий в соответствии с 4.7 и/или назначения дополнительных атрибутов сущностей для МКЦ в соответствии с 4.3.2.

ВНИМАНИЕ! Изменять классификационную метку сущности (т. е. изменять уровень конфиденциальности и/или категории конфиденциальности) может только субъект, имеющий метку целостности не ниже метки целостности этой сущности и обладающий привилегией `PARSEC_CAP_CHMAC` (см. 4.7), или субъект с правами `root` и с максимальной меткой целостности.

ВНИМАНИЕ! Понижать метку целостности сущности может только субъект с наличием привилегии `PARSEC_CAP_CHMAC` (см. 4.7) и имеющий метку целостности не ниже метки целостности этой сущности. Произвольно изменять метку целостности сущности может только субъект с правами `root` и с максимальной меткой целостности.

ВНИМАНИЕ! Субъект может изменять только собственную классификационную метку (т. е. изменять свой уровень конфиденциальности и/или категории конфиденциальности) при наличии у него привилегии `PARSEC_CAP_SETMAC` (см. 4.7).

ВНИМАНИЕ! Для процесса возможно только понизить метку целостности. Для выполнения этого безопасным способом, т. е. с закрытием наследуемых от родительского процесса файловых дескрипторов, рекомендуется использовать утилиту `sumic` (см. 4.15.10) и библиотечную функцию `pdp_set_pid_safe`.

Для создания в контейнере, имеющем атрибут `ccnr`, вложенной сущности с уровнем и категориями конфиденциальности меньшими, чем у контейнера, необходимо обладать привилегиями `PARSEC_CAP_IGNMACCAT` и `PARSEC_CAP_IGNMACLVL` (см. таблицу 12).

Мандатный контекст безопасности корневой файловой системы определяет максимальный мандатный контекст безопасности сущностей. Устанавливаемый по умолчанию мандатный контекст безопасности корневой файловой системы, а также мандатный контекст отдельных сущностей определены в сценарии `/usr/sbin/pdp-init-fs` (см. 4.15.3).

Принятие решения о запрете или разрешении доступа субъекта к сущности принимается на основе типа операции (чтение/запись/исполнение), мандатного контекста безопасности субъекта и мандатного контекста безопасности сущности.

Операции чтения разрешены, если:

1) уровень конфиденциальности субъекта не ниже уровня конфиденциальности сущности;

2) биты набора категорий конфиденциальности субъекта включают биты набора категорий конфиденциальности сущности;

3) метка целостности субъекта может быть любой.

Операции исполнения разрешены, если:

1) уровень конфиденциальности субъекта не ниже уровня конфиденциальности сущности;

2) биты набора категорий конфиденциальности субъекта включают биты набора категорий конфиденциальности сущности;

3) метка целостности субъекта:
   a) при отключенном расширенном режиме МКЦ — может быть любой;

б) при включенном расширенном режиме МКЦ — не выше метки целостности сущности.

Операция записи разрешена, если:

1) уровни и категории конфиденциальности субъекта и сущности совпадают;
2) метка целостности субъекта не ниже метки целостности сущности.

ВНИМАНИЕ! При переименовании или редактировании файла в текстовом формате с ненулевой меткой безопасности для сохранения значения метки на файле (предотвращения сброса в 0) необходимо использовать параметр ядра `parsec.rename_mask` (см. 4.18).

При сравнении меток целостности между собой используется следующее правило:

1) метка A равна метке B, если в категории целостности метки A установлены все те же биты, что и в категории целостности метки B, и линейный уровень целостности метки A равен линейному уровню целостности метки B;
2) метка A больше метки B, если в категории целостности метки A установлены все те же биты, что и в категории целостности метки B, а также в метке A установлены и другие биты, линейный уровень целостности метки A больше линейного уровня целостности метки B;
3) две метки целостности несравнимы, если у категорий целостности каждой метки есть биты, отсутствующие у другой метки.

Субъект не может получить доступ к управлению другим субъектом или сущностью, если его метка целостности ниже или несравнима с меткой целостности субъекта или сущности.

Порядок предоставления доступа на чтение, исполнения и запись может быть изменен путем применения PARSEC-привилегий в соответствии с 4.7, назначения дополнительных атрибутов сущностей для мандатного управления доступом в соответствии с 4.2.3 и/или дополнительных атрибутов сущностей для МКЦ в соответствии с 4.3.2.

4.7. PARSEC-привилегии

PARSEC-привилегии назначаются субъектам и позволяют изменить порядок назначения меток безопасности и предоставления доступа на чтение, исполнение и запись. PARSEC-привилегии, как и Linux-привилегии, приведенные в 3.2, наследуются процессами от процессов-родителей.

Процессы, запущенные от имени суперпользователя, имеющего максимальную (`Высокий`) категорию целостности по умолчанию, независимо от явного назначения им привилегий, имеют возможность осуществлять большинство привилегированных действий.

PARSEC-привилегии и их описание приведены в таблице 12.

Таблица 12

Привилегия Битовая маска	Описание
PARSEC_CAP_FILE_CAP 0x000001	Не используется. Для изменения меток безопасности файловых объектов использовать привилегию PARSEC_CAP_CHMAC. Для изменения меток безопасности процессов использовать PARSEC_CAP_SETMAC. Для изменения привилегий процессов использовать PARSEC_CAP_CAP
PARSEC_CAP_AUDIT 0x000002	Позволяет управлять политикой аудита
PARSEC_CAP_SETMAC 0x000004	Позволяет процессу изменять собственную классификационную метку (уровень и категории конфиденциальности)
PARSEC_CAP_CHMAC 0x000008	Позволяет субъекту изменять метку безопасности сущности, если метка целостности сущности меньше или равна метки целостности субъекта. Привилегия позволяет изменять атрибуты сущностей для мандатного управления доступом (см. 4.2.3). Привилегия позволяет только понижать метку целостности сущности. Привилегия не позволяет изменять атрибуты сущностей для МКЦ (см. 4.3.2)
PARSEC_CAP_IGNMACLVL 0x000010	Позволяет игнорировать мандатную политику по уровням конфиденциальности
PARSEC_CAP_IGNMACCAT 0x000020	Позволяет игнорировать мандатную политику по категориям конфиденциальности
PARSEC_CAP_SIG 0x000040	Позволяет посылать сигналы процессам, игнорируя мандатные права
PARSEC_CAP_UPDATE_ATIME 0x000080	Не используется. Позволяет изменять время доступа к файловым объектам
PARSEC_CAP_PRIV_SOCK 0x000100	Позволяет создавать новые сетевые сокеты процесса в привилегированном режиме (атрибут ehole). Привилегированный сокет позволяет осуществлять сетевое взаимодействие, игнорируя мандатную политику. Для точек соединения (UNIX-сокетов) использовать привилегию PARSEC_CAP_MAC_SOCK
PARSEC_CAP_READSEARCH 0x000200	Позволяет игнорировать мандатную политику при чтении и поиске файловых объектов (но не при записи)
PARSEC_CAP_CAP 0x000400	Позволяет процессу назначать себе любой непротиворечивый набор привилегий и читать привилегии, присвоенные процессам

Продолжение таблицы 12

Привилегия Битовая маска	Описание
PARSEC_CAP_MAC_SOCK 0x000800	Позволяет изменять метки безопасности точек соединения (UNIX-сокетов). Процесс с данной привилегией может изменять метку целостности сокета, у которого метка целостности не выше метки целостности процесса. При этом повышать метку целостности сокета возможно не выше метки целостности процесса. Для сетевых сокетов использовать привилегию PARSEC_CAP_PRIV_SOCK
PARSEC_CAP_UNSAFE_SETXATTR 0x001000	Позволяет устанавливать мандатные атрибуты объектов файловой системы без учета мандатных атрибутов родительского каталога. Также позволяет создавать жесткие ссылки без учета мандатных атрибутов родительского каталога. Привилегия используется для восстановления объектов файловой системы из резервных копий и только после установки значения 1 для параметра /parsecfs/unsecure_setxattr
PARSEC_CAP_IGNMACINT 0x002000	Позволяет игнорировать политику МКЦ. Устанавливать данную привилегию может только субъект с правами root и с максимальной меткой целостности в системе. **ВНИМАНИЕ!** Запрещено использование данной привилегии при штатной работе МКЦ и при разработке. Данная привилегия предназначена только для временного использования при устранении сбоев и отладке работы ОС или прикладного ПО
PARSEC_CAP_SUMAC 0x004000	Позволяет запускать процессы с другой классификационной меткой (уровнем и категорией конфиденциальности)
PARSEC_CAP_BYPASS_KIOSK 0x008000	Не используется. Позволяет игнорировать ограничения киоска (parsec-kiosk)
PARSEC_CAP_IPC_OWNER 0x010000	Отменяет мандатные ограничения при работе с сущностями IPC, такими как shared memory, message queue и т.д. (PARSEC-привилегия, аналогичная Linux-привилегии CAP_IPC_OWNER)
PARSEC_CAP_INHERIT_INTEGRITY 0x020000	При создании сущностей им назначается метка целостности, равная метке целостности родительского каталога. При этом запрещено создавать сущности с меткой целостности выше или несравнимой с меткой целостности процесса, создающего данную сущность. **ВНИМАНИЕ!** Если родительскому каталогу присвоен атрибут irelax, то создаваемым в нем сущностям присваивается метка целостности по правилам работы атрибута irelax (см. 4.3.2), т. к. irelax обладает приоритетом над данной привилегией. Используется для установщика пакетов (dpkg)
PARSEC_CAP_BYPASS_XATTR 0x040000	Отключает проверку подписи файлов в xattr
PARSEC_CAP_PROCFS 0x080000	Включает игнорирование уровней конфиденциальности и метки целостности при работе с /proc

Окончание таблицы 12

Привилегия Битовая маска	Описание
PARSEC_CAP_CCNR_RELAX 0x100000	Позволяет осуществлять в каталоге с установленным атрибутом ccnr действия (создание, удаление и др.) над вложенными файлами и каталогами с уровнями конфиденциальности не выше уровня конфиденциальности данного каталога. Привилегия применяется при включенном расширенном режиме МКЦ (см. 4.5)

Список доступных PARSEC-привилегий возможно просмотреть с помощью инструмента usercaps (см. 4.16.1).

Для настройки КСЗ могут использоваться как PARSEC-, так и Linux-привилегии. Порядок управления привилегиями описан в 4.16.

4.8. Включение и выключение мандатного управления доступом

4.8.1. Включение мандатного управления доступом

Мандатное управление доступом может быть включено в процессе установки ОС путем выбора соответствующего пункта программы установки ОС.

Включение мандатного управления доступом после установки ОС выполняется с помощью инструмента astra-mac-control, описанного в 16.6.31, или модуля «Мандатное управление доступом» в разделе «Управление доступом» графической утилиты astra-systemsettings («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_mac
```

При включении мандатного управления доступом для параметра командной строки ядра parsec.mac в загрузчике ОС устанавливается значение 1. При этом все доработанные для функционирования в условиях мандатного управления доступом сетевые службы, системы IPC и системы инициализации будут работать в режиме мандатного управления доступом. Также программам будет доступна возможность работать с файловыми объектами, имеющими ненулевую классификационную метку.

4.8.2. Выключение мандатного управления доступом

Выключение мандатного управления доступом целесообразно применять только для отладочных целей. Выполнять выключение в системе, введенной в эксплуатацию, не рекомендуется, поскольку выключение мандатного управления доступом может привести к

несогласованному состоянию объектов файловой системы с разными уровнями конфиденциальности.

Выключение мандатного управления доступом выполняется с помощью инструмента `astra-mac-control`, описанного в 16.6.31, или модуля «Мандатное управление доступом» в разделе «Управление доступом» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_mac
```

При выключении мандатного управления доступом для параметра командной строки ядра `parsec.mac` устанавливается значение `0`.

ВНИМАНИЕ! При выключенном мандатном управлении доступом службы, которые должны запускаться на ненулевом уровне конфиденциальности или в процессе работы повышают свой уровень конфиденциальности, будут запущены на нулевом уровне конфиденциальности или не будут запущены с выводом информации об ошибке.

После выключения мандатного управления доступом для программ будет отключена возможность работать с файловыми объектами, имеющими ненулевую классификационную метку.

ВНИМАНИЕ! Отключение возможности работы программ с файловыми объектами, имеющими ненулевую классификационную метку, не равносильно удалению таких объектов. Файловые объекты, имеющие ненулевую классификационную метку, после отключения возможности работы с ними сохраняются. Доступ к таким объектам не может быть получен штатными средствами ОС, но доступ к ним может быть получен при наличии неконтролируемого физического доступа к компьютеру и возможности использовать на нем нештатные средства.

4.9. Включение и выключение мандатного контроля целостности

Включение МКЦ может быть выполнено в процессе установки ОС путем выбора пункта «Мандатный контроль целостности» в программе установки ОС.

Включение и выключение МКЦ после установки ОС выполняется с помощью инструмента `astra-mic-control`, описанного в 16.6.27, или модуля «Мандатный контроль целостности» в разделе «Управление доступом» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_mic
```

При включении/выключении МКЦ автоматически включается/выключается МКЦ на файловой системе (см. 4.10).

При включении МКЦ для параметра командной строки ядра `parsec.max_ilev` в загрузчике ОС устанавливается значение максимальной категории целостности в системе. По умолчанию значение максимальной категории целостности в системе `63` (если при включении МКЦ не было указано другое значение, см. 16.6.27).

ВНИМАНИЕ! Графический сервер Xorg по умолчанию работает от имени служебной учетной записи `fly-dm` на выделенном уровне целостности 8. При отсутствии поддержки драйверами видеокарты KMS (Kernel Mode-Setting) запуск Xorg производится от имени учетной записи `root`.

При выключении МКЦ для параметра командной строки ядра `parsec.max_ilev` устанавливается значение `0`.

4.10. Мандатный контроль целостности на файловой системе

При включении МКЦ согласно 4.9 объектам файловой системы автоматически присваиваются метки целостности.

Присвоение меток целостности объектам файловой системы осуществляется в соответствии с конфигурационным файлом `/etc/parsec/fs-ilev.conf`. Объектам, не указанным в файле, метка целостности не назначается.

В конфигурационном файле перечислены объекты файловой системы и их метка целостности в формате:

```
<категория_целостности>[:<атрибут>:<линейный_уровень_целостности>] <путь>
```

где `<категория_целостности>` — категория целостности для объекта файловой системы, указанного в `<путь>`;
`<атрибут>` — дополнительный атрибут(ы) сущностей для МКЦ;
`<линейный_уровень_целостности>` — линейный уровень целостности для объекта файловой системы, указанного в `<путь>`;
`<путь>` — объект(ы) файловой системы или путь к нему.

Если в файле указаны несуществующие и неабсолютные пути, то они игнорируются. Корневому каталогу («/») метка целостности не назначается.

Значения, указываемые в конфигурационном файле в качестве категории целостности `<категория_целостности>`, приведены в таблице 13.

Таблица 13

Значение	Описание
<число>	Определенная категория целостности, заданная числовым значением. Может быть десятичным, восьмеричным, шестнадцатеричным или двоичным числом
high	Текущее значение max_ilev — максимальная категория целостности в ОС, заданная в параметре командной строки ядра parsec.max_ilev
max	Текущее значение max_ilev — максимальная категория целостности в ОС, заданная в параметре командной строки ядра parsec.max_ilev
low	То же, что и нулевая категория целостности
min	То же, что и нулевая категория целостности
exc	Игнорировать файл при проверке целостности. В качестве символа подстановки в конце пути можно использовать символ «*»

Пример

Конфигурационный файл /etc/parsec/fs-ilev.conf

```
exc     /etc/xdg/autostart/vboxclient.desktop
exc     /etc/X11/Xsession.d/98vboxadd-xclient
exc     /etc/ld.so.*
exc     /etc/resolv.conf
exc     /root/.config/*
exc     /root/.gnupg/gpg-agent.conf
max     /etc
max     /lib
max     /lib64
max     /lib32
max     /bin
max     /sbin
max     /boot
max     /root
max     /opt
max     /srv
max     /usr
max     /var/lib/dpkg/info
max     /var/lib/docker
max     /var/lib/polkit-1/localauthority
max     /etc/polkit-1
max     /usr/share/polkit-1
```

Для управления метками целостности на файловой системе используется инструмент командной строки set-fs-ilev.

После установки новых пакетов, а также в процессе работы ОС могут создаваться новые файлы в каталоге `/etc/`, которым метки целостности МКЦ автоматически не присваиваются. Чтобы привести объекты файловой системы в соответствие конфигурационному файлу `/etc/parsec/fs-ilev.conf`, необходимо выполнить команду:

```
sudo set-fs-ilev enable
```

Подробное описание инструмента `set-fs-ilev` приведено на справочной странице `man set-fs-ilev`.

Также для управления МКЦ на файловой системе может использоваться модуль «Мандатный контроль целостности» в разделе «Управление доступом» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_mic
```

Выключение МКЦ на файловой системе осуществляется автоматически при выключении МКЦ согласно 4.9.

Для обеспечения совместимости в ОС сохранен устаревший инструмент выключения МКЦ на файловой системе `unset-fs-ilev`.

4.11. Администрирование ОС при включенном МКЦ

Привилегированному пользователю (администратору) следует входить в систему с высокой категорией целостности (соответствует максимальной категории целостности ОС) и только для выполнения настроек ОС.

Для обычного (штатного) режима работы рекомендуется осуществлять вход в систему от имени непривилегированного пользователя с низкой категорией целостности (соответствует минимальной категории целостности).

Применяются следующие ограничения на сочетание классификационной метки и категории целостности для входа в сессию:

- если выбрана ненулевая классификационная метка (выбран ненулевой уровень конфиденциальности и/или категории конфиденциальности), то вход может быть выполнен только с нулевой категорией целостности (при этом в расширенном режиме МКЦ выбирается автоматически при выполнении входа);
- при включенном расширенном режиме МКЦ, если выбрана нулевая классификационная метка (или у пользователя отсутствуют уровень и категории конфиденциаль-

ности), то вход автоматически выполняется с максимальной категорией целостности, доступной пользователю.

ВНИМАНИЕ! При отключенном расширенном режиме МКЦ не рекомендуется выполнять вход в систему администратором с нулевой классификационной меткой и нулевой категорией целостности.

При консольном входе в систему администратор должен вручную выставлять категорию целостности (для высокой — `63`, для низкой — `0` или пропустить данный шаг).

При входе в сессию с высокой категорией целостности графический рабочий стол имеет красную тему оформления, которая не может быть изменена.

4.12. Запуск служб systemd с категорией целостности и уровнем конфиденциальности

Для запуска службы systemd с определенной меткой безопасности необходимо в конфигурационном файле (юните) соответствующей службы (`<имя_юнита>.service`) в разделе `[Service]` добавить следующий параметр (метку):

```
[Service]
PDPLabel=<Уровень>:<Категория_целостности>:<Категории>
```

Для управления юнитами пользователь должен иметь категорию целостности, включающую в себя категорию целостности управляемого юнита (заданную параметром `PDPLabel`, по умолчанию `0:63`). Попытки пользователя управлять юнитами, имеющими категорию целостности выше, чем его, регистрируются в системном журнале.

Категория целостности назначается только в пределах `max_ilev`.

Формат метки `PDPLabel` аналогичен принятому в системе PARSEC за исключением поля типа метки — метка службы не может иметь дополнительный атрибут сущности для мандатного управления доступом (наличие `ehole/whole` недопустимо). Более подробная информация о формате метки доступна в выводе команды:

```
pdpl-file --help
```

Для метки рекомендуется использовать числовые обозначения, так как при разрешении имен могут оказаться задействованы сетевые ресурсы, например, LDAP-каталоги, что может привести к ошибкам конфигурации, которые сложно диагностировать.

Для назначения службе PARSEC-привилегий, приведенных в 4.7, в соответствующем юните (<имя_юнита>.service) в разделе [Service] добавить параметр:

```
[Service]
CapabilitiesParsec=PARSEC_CAP_PRIV_SOCK ...
```

где через пробел могут быть перечислены PARSEC-привилегии.

После редактирования конфигурационного файла службы необходимо перезапустить systemd и соответствующую службу, выполнив команды:

```
systemctl daemon-reload
systemctl restart <имя_службы>.service
```

Для просмотра метки службы выполнить команду:

```
pdpl-ps <PID>
```

где <PID> — идентификатор службы.

Для просмотра установленных на службе PARSEC-привилегий выполнить команду:

```
pscaps <PID>
```

Для определения PID используется команда:

```
systemctl status <имя_службы>.service
```

4.13. Сетевое взаимодействие. Механизм privsock

В качестве основного протокола доступа к сетевой ФС используется протокол SMB, который поддерживает передачу расширенных атрибутов, содержащих информацию о мандатном контексте (метке безопасности и мандатных атрибутах управления доступом). Данный протокол широко распространен и работает в гетерогенных сетях (поддерживается различными операционными системами), а также поддерживает собственную аутентификацию и аутентификацию с использованием Kerberos.

Взаимодействие при помощи сетевого протокола IPv4 (IPv6) осуществляется через программный интерфейс объектов доступа, являющихся элементами межпроцессного и сетевого взаимодействия (например, сетевых сокетов), которые обеспечивают обмен данными между

процессами в рамках одной или нескольких ОС, объединенных в локальную вычислительную сеть.

Для поддержки мандатного управления доступом в сетевые пакеты протокола IPv4 (IPv6) внедряются классификационные метки. Порядок присвоения классификационных меток и их формат соответствует национальному стандарту ГОСТ Р 58256-2018. Прием сетевых пакетов подчиняется мандатным ПРД. Следует отметить, что метка безопасности сокета может иметь тип, позволяющий создавать сетевые службы, принимающие соединения с любыми уровнями секретности.

При необходимости для обеспечения целостности заголовка IP-пакетов, содержащего классификационную метку, допускается применение программного средства OpenVPN. Описание использования OpenVPN приведено в документе РУСБ.10015-01 95 01-1.

Отсутствие у сущности метки безопасности эквивалентно нулевой метке безопасности. Таким образом, ядро ОС, в которой все сущности и субъекты доступа имеют уровень секретности «несекретно», функционирует аналогично стандартному ядру операционной системы семейства Linux.

Для ряда сетевых служб (сервера LDAP, DNS, Kerberos и т. д.) необходимо обеспечить возможность их работы с клиентами, имеющими разный мандатный контекст безопасности, без внесения изменений в исходные тексты службы. Для предоставления данной возможности в подсистеме безопасности PARSEC реализован механизм запуска сетевых служб с использованием привилегированного сокета для ожидания входящих соединений — механизм `privsock`.

Механизм `privsock` предназначен для обеспечения функционирования системных сетевых служб, не осуществляющих обработку информации с использованием мандатного контекста, но взаимодействующих с процессами, работающими в мандатном контексте субъекта доступа.

Для его использования при функционировании сетевой службы необходимо отредактировать файл `/etc/parsec/privsock.conf`, добавив в него строку, содержащую полный путь к исполняемому файлу службы. Далее приведен пример строки из файла `/etc/parsec/privsock.conf` для запуска DNS-сервера с использованием механизма `privsock`.

Пример

```
/usr/sbin/named
```

Для использования механизма privsock необходимо, чтобы переменная PATH, используемая при запуске службы, содержала следующий путь:

```
/usr/lib/parsec/bin
```

Далее приведен пример задания требуемого пути в переменной окружения для запуска DNS-сервера.

Пример

Установка значения переменной окружения PATH может быть выполнена добавлением в файл /etc/default/bind9 следующей строки:

```
PATH=/usr/lib/parsec/bin:$PATH
```

Подробное описание механизма privsock приведено на справочной странице man privsock.

4.14. Шина межпроцессного взаимодействия D-Bus

D-Bus позволяет организовать взаимодействие процессов с использованием сообщений и общих шин. Для передачи сообщения между процессом и шиной используется механизм сокетов. Выделяют два типа шин: сессионные шины (session bus) и системная шина (system bus).

Сессионная шина создается для каждой пользовательской сессии при ее запуске. Она работает с меткой безопасности пользователя системы и все взаимодействующие через нее процессы также имеют данную метку безопасности. Взаимодействие процессов с иной меткой безопасности не происходит. Дополнительные сессионные шины могут быть созданы с использованием утилиты dbus-launch, которая позволяет запустить процесс одновременно с созданием новой сессионной шины. При этом имеется возможность указать, используя переменную DBUS_SESSION_BUS_ADDRESS, какую из сессионных шин данный процесс будет воспринимать как сессионную шину.

Пример

```
user@astra:~$ echo $DBUS_SESSION_BUS_ADDRESS
unix:abstract=/tmp/dbus-FMSYkkteW0,guid=2f874cb94fd70c984eff1d8857d24781
```

Аналогично функционирует системная шина. Данная шина создается только в одном экземпляре при старте службы D-Bus и через нее взаимодействуют процессы различных уровней.

Модуль dbus-daemon реализует и сессионные, и системную шины, поэтому механизм мандатного управления доступом используется во всех шинах.

4.14.1. Виды сообщений

D-Bus поддерживает следующие виды сообщений:

- method_call — вызов метода. Процесс требует вызова метода, реализованного другим процессом. Данное сообщение имеет конкретного адресата и содержит информацию об источнике сообщения;
- method_return — возврат. Как правило, после вызова метода данное сообщение используется для возврата значения. Адресатом сообщения является источник исходного сообщения;
- error — ошибка. Если при вызове произошла ошибка, вместо возврата может быть отправлено сообщение об ошибке. Адресатом сообщения является источник исходного сообщения;
- signal — сигнал. Данное сообщение имеет источник, но, как правило, не имеет конкретного адресата и рассылается шиной всем, кто подписан на данное сообщение (с использованием match-фильтра). Если указано имя службы, то сообщение направляется только определенному соединению (т. е. первичному владельцу данной службы).

Все сообщения являются однонаправленными, могут иметь назначение, а также могут подразумевать (но не требовать) ответ, например, после method_call можно не посылать method_return. Функционал отправителя и получателя должен быть согласован.

4.14.2. Процесс взаимодействия с шиной

Для идентификации соединения на шине используются параметры, приведенные в таблице 14.

Таблица 14

Параметр	Описание
unique connection name (UCN)	Уникальное имя соединения — идентификатор соединения
org.share.server	Общеизвестное имя (служба) соединения. Данное имя используется для приема сообщения от других клиентов (т. е. по этому имени клиенты могут найти данный процесс)
/org/share/server/object	Данный путь является именем объекта. Каждая служба может создавать несколько объектов
com.share.interface	Имя интерфейса. Соответствует некоторому набору методов (в том числе стандартным методам, общим для объектов), которые должны корректно обрабатываться объектом

Окончание таблицы 14

Параметр	Описание
`GetMyName`	Имя метода объекта или интерфейса, который можно вызвать

Для вызова метода необходимо:

- указать общеизвестное имя соединения, имя объекта, имя интерфейса и имя метода;
- сформировать набор аргументов;
- выполнить вызов.

Для обработки результата необходимо принять ответное сообщение (которое может быть как сообщением типа возврат, так и ошибкой). Также существуют широковещательные сообщения-сигналы, которые отправляются процессом на шину, где переадресовываются всем соединениям, которые на него подписаны.

Каждый процесс для работы через D-Bus подключается к заданной шине (при этом создается уникальное соединение с именем вида `:1.8`) с использованием функции `dbus_bus_get()`. При этом при подключении осуществляется механизм аутентификации клиента с использованием данных сокета. После этого клиент может работать с шиной с использованием сообщений.

4.14.3. Процесс соединения с системной шиной

После аутентификации клиент отправляет сообщение типа `method_call Hello` серверу D-Bus. Далее проверяется возможность доставки данного сообщения с использованием `pdp` и `selinux` функций. Для того, чтобы D-Bus мог обработать данный вызов с ненулевых уровней, необходимо для него установить соответствующие привилегии. В частности, это реализовано за счет установки привилегий `0x30` (Ignore Lev & Cat) на службу `org.freedesktop.Dbus`. Данные привилегии позволяют процессам (соединениям) с более высоким уровнем выполнять функции процессов с низким уровнем (`dbus`), что не предусматривает мандатное управление доступом, но разрешается привилегиями `0x30`, которые позволяют субъекту (`dbus`) игнорировать метку безопасности соединения, пытающегося выполнить его метод.

После этого формируется ответное сообщение типа `method_return`, которое возвращает ответный статус об успешном создании соединения с шиной. Одновременно происходит установка владельца службы. Под службой понимается имя на шине, которое соответствует некоторому соединению. При этом владельцами уникальных имен (вида `:1.6`) могут быть только сами соединения. Если используется пользовательское имя, например, `org.share.linux`, то его владельцем может быть соединение с некоторым уникальным именем. Другие процессы (соответственно с другими уникальными именами соединений)

также могут запросить доступ владения данным именем. В зависимости от настроек может произойти как замена владельца, так и установка его в очередь на владение. При этом, когда текущий владелец освободит данное имя (отключится от шины), владельцем станет следующее по списку соединение (если такие есть). Важно, что именно владелец, т. е. соединение и соответствующий ему процесс, будет обрабатывать адресованное данному имени (службе) сообщение. В дополнение к ответу источнику сообщения `method_call Hello` происходит формирование сигнала `signal`. D-Bus формирует сигнал `NameAquired` (т. е. источником является D-Bus), который сообщает, что имя (в данном случае UCN) получено.

Далее для получения владения некоторым именем (функция `dbus_bus_request_name`) формируется сообщение `method_call RequestName` серверу D-Bus. При этом могут использоваться следующие флаги:

- разрешить замену владельца. Если другое соединение потребует заменить владельца, то этот флаг разрешит выполнить смену с учетом метки безопасности;
- заменить владельца. Если разрешена замена, то он будет заменен у ранее созданной службы;
- не ставить в очередь, если владельца нельзя сейчас заменить. Иначе данное соединение встанет в конец очереди, и когда будут освобождены все владельцы — оно станет владельцем.

Если замена возможна, то вызывается драйвер-функция `RequestName`. После смены владельца службы формируется сигнал `NameOwnerChanged`. После освобождения имени формируется сигнал `NameLost`.

4.14.4. Объекты и субъекты системы

В рамках D-Bus объектами и субъектами являются сущности, сопоставленные с именами соединений и служб (т. е. уникальными именами и общеизвестными именами). Соответственно, каждое соединение имеет свою метку безопасности. Кроме этого, при создании службы (получения имени через `dbus_request_name`), а также создании уникального имени, происходит назначение им метки безопасности. Сведения о метках безопасности содержатся в хеш-таблице, где каждому имени соответствует метка безопасности. При освобождении имени, когда последний владелец уходит из очереди или клиент отсоединяется от шины, выполняется удаление соответствующей записи таблицы (при использовании `LOCKED`-настройки метка безопасности остается).

Проверка `pdp` доступа осуществляется при следующих событиях:

- принятие решения о доставке сообщения в зависимости от его типа;
- получение владения службой;
- выполнения методов службы D-Bus.

Каждое соединение при своем создании (получении уникального имени) получает соответствующую метку безопасности исходя из метки безопасности сокета.

4.14.5. Алгоритм проверки меток для различных сообщений и режимов работы

Взаимодействие по шине D-Bus можно разделить на два типа:

- сообщения между службами и соединениями;
- сообщения, адресованные службе D-Bus. Здесь представлены служебные сообщения, которые позволяют получать информацию о других службах и соединениях на шине. Т. е. в параметрах сообщения может быть указана заданная служба, информацию о которой необходимо получить.

Для сообщений между службами и соединениями используется функция `bus_pdplinux_allows_send()`. При широковещательной рассылке сигналов также происходит проверка каждого сообщения.

Для сообщений, адресованных службе D-Bus, используется сначала функция `bus_pdplinux_allows_send()`. Она определяет возможность отправки сообщения на D-Bus. Затем вызывается функция `bus_pdplinux_service_allows_connection_to_service` с заданными параметрами, которая определяет возможность выполнения тех или иных методов.

Для методов `service_exists`, `get_service_owner`, `list_queued_owners`, `get_connection_unix_user`, `get_connection_unix_process_id`, `get_adt_audit_session_data`, `get_connection_selinux_security_context`, `get_connection_pdplinux_security_context_helper`, `introspect`, `bus_driver_handle_get_id` и др. в зависимости от настроек проверяется возможность доступа `READ` (`TOS_READ`) к заданной службе.

Для метода `list_queued_owners` выводятся только те владельцы, метка безопасности соединения которых меньше, чем у соединения, создавшего данное сообщение (т. е. вызвавшего метод). Наличие привилегий может разрешать данный вызов в случае невыполнения последнего условия.

Для метода `list_services` выводится информация только по тем службам, метка безопасности которых меньше, чем у соединения, создавшего данное сообщение (т. е. вызвавшее метод). Наличие привилегий может разрешать данный вызов в случае невыполнения последнего условия.

Для получения владения службой в методах `acquire_service`, `add_owner`, `swap_owner` функция `bus_pdplinux_service_allows_connection_to_service` вызывается с указанием псевдодоступа `ETOS_ADDOWNER` (для `acquire_service` используется `ETOS_ADDOWNER_ACQUIRE_SERVICE`), которая в зависимости от

настроек параметра конфигурации <pdplinux_allow_different_owners> (BUS_CONTEXT_PDPLINUX_GET_pdplinux_allow_different_owners) проверяет наличие потенциальных владельцев в очереди с меткой безопасности, отличной от метки безопасности службы. Если параметр pdplinux_allow_different_owners установлен, то возможно получение владения службой.

Для непосредственной проверки возможности замены владельца службы модифицирована функция bus_service_get_allow_replacement(), которая выполняет проверку метки целостности и не позволяет проводить соответствующую замену в случае, если метка целостности службы выше метки целостности нового владельца.

При пересылке сообщений наряду с получателем сообщения может присутствовать соединение eavesdroping. При этом сообщение адресуется не только истинному получателю, но и перенаправляется на дополнительное соединение, которое может быть создано, например, программой dbus-monitor.

Для каждого типа сообщения формируется соответствующий вид доступа. Для method_call применяется доступ exec. Для этого сравниваются метки безопасности источника сообщения и соответствующей службы, совпадающей с ее текущим владельцем. Если метки безопасности равны, то доступ разрешается. Иначе вызывается функция bus_pdplinux_allows_execution(), которая разрешает низкоуровневым объектам запуск функций объекта более высокого уровня. Если запуск запрещен, то проверяются привилегии. Например, если служба, метод которой запускается, имеет привилегии 0x30, то запуск разрешается. Также существует возможность запуска метода, если соединение-получатель имеет дополнительный атрибут ehole в функции bus_pdplinux_allows_execution_helper().

Для method_return, error, signal получатель сообщения считывает данные из источника. Таким образом они меняются местами (получатель является субъектом, источник — объектом) и доступ назначается равным read. Проверка pdpl_permission().

4.14.6. Привилегии процесса dbus-daemon

Установка привилегий процесса dbus-daemon зависит от установки параметра сборки --enable-audit/--disable-audit. Это влечет за собой define: HAVE_LIBAUDIT (--enable-libaudit).

В случае, если HAVE_LIBAUDIT определен, то вследствие доработки функции dbus_change_to_daemon_user в файле audit.c в точку считывания привилегии CAP_AUDIT_WRITE добавлен код считывания PERMITTED (CAPNG_PERMITTED) привилегий процесса в переменные флагов have_cap_override, have_cap_ptrace, have_cap_admin (для соответствующих привилегий). Если значение данных переменных true, то производится установка CAP_DAC_OVERRIDE, CAP_SYS_PTRACE, CAP_SYS_ADMIN

привилегий для процесса dbus-daemon. Выполняется для обеспечения возможности доступа dbus-daemon к именам процессов.

В случае, если HAVE_LIBAUDIT не определен (--disable-audit), то производится обработка функции dbus_change_to_daemon_user в файле dbus-sysdeps-util-unix.c. При этом добавлен функционал, позволяющий скопировать все привилегии процесса из CAP_INHERITABLE в CAP_EFFECTIVE (при условии, что dbus запущен от имени пользователя root). Предполагается получение привилегий CAP_SYS_ADMIN CAP_DAC_OVERRIDE CAP_SYS_PTRACE для считывания имени процесса. Данные привилегии предварительно устанавливаются с использованием systemd в конфигурационном файле dbus.service:

```
# AmbientCapabilities=CAP_SYS_ADMIN CAP_DAC_OVERRIDE CAP_SYS_PTRACE
# SecureBits=keep-caps
```

В настоящий момент параметры AmbientCapabilities и SecureBits в dbus.service закомментированы, и используется HAVE_LIBAUDIT. Также при использовании нового systemd возможно использование PARSEC-привилегий для процесса dbus.socket:

```
CapabilitiesParsec=PARSEC_CAP_PRIV_SOCK PARSEC_CAP_IGNMACCAT
     PARSEC_CAP_IGNMACLVL
```

4.14.7. Расширенное управление политиками

4.14.7.1. Конфигурационный файл

Основными секциями конфигурационного файла D-Bus system.conf являются <busconfig> (корневая секция), <type> и <policy>.

Пример

Конфигурационный файл шины accessibility:

```
<!DOCTYPE busconfig PUBLIC "-//freedesktop//DTD D-Bus Bus Configuration
     1.0//EN" "http://www.freedesktop.org/standards/dbus/1.0/
     busconfig.dtd">
<busconfig>
  <type>accessibility</type>
<servicedir>/usr/share/dbus-1/accessibility-services</servicedir>
  <auth>EXTERNAL</auth>
  <listen>unix:tmpdir=/tmp</listen>

  <policy context="default">
    <!-- Allow root to connect -->
```

```
    <allow user="root"/>
    <!-- Allow everything to be sent -->
    <allow send_destination="*" eavesdrop="true"/>
    <!-- Allow everything to be received -->
    <allow eavesdrop="true"/>
    <!-- Allow anyone to own anything -->
    <allow own="*"/>
    <deny send_interface="org.a11y.atspi.Text" send_member=
       "GetStringAtOffset"/>
    <deny send_interface="org.a11y.atspi.Text" send_member="GetText"/>
    <deny send_interface="org.a11y.atspi.Text" send_member=
       "GetTextBeforeOffset"/>
    <deny send_interface="org.a11y.atspi.Text" send_member=
       "GetTextAtOffset"/>
    <deny send_interface="org.a11y.atspi.Text" send_member=
       "GetTextAfterOffset"/>
    <deny send_interface="org.a11y.atspi.Text" send_member=
       "GetCharacterAtOffset"/>
    <deny send_interface="org.a11y.atspi.EditableText" send_member=
       "SetTextContents"/>
    <deny send_interface="org.a11y.atspi.EditableText" send_member=
       "InsertText"/>
    <deny send_interface="org.a11y.atspi.DeviceEventListener"
       send_member="NotifyEvent"/>
    <deny send_interface="org.a11y.atspi.DeviceEventController"
       send_member="RegisterKeystrokeListener"/>
    <deny send_interface="org.a11y.atspi.DeviceEventController"
       send_member="RegisterDeviceEventListener"/>
    <deny send_interface="org.a11y.atspi.DeviceEventController"
       send_member="GenerateKeyboardEvent"/>
    <deny send_interface="org.a11y.atspi.DeviceEventController"
    send_member="GenerateMouseEvent"/>
    <deny send_interface="org.a11y.atspi.Document" send_member=
       "GetAttributeValue"/>
  </policy>
```

Секция <policy> определяет политику безопасности, которая должна применяться к конкретному набору подключений к шине. Политика состоит из правил <allow> (разрешение) и <deny> (запрет). Политика, как правило, задается для системной шины и используется для разрешения либо запрета передачи сообщений.

Системная шина имеет политику по умолчанию, которая запрещает отправку сообщений типа method_call и получение владения службами на шине (own). Остальные действия, в

частности, ответы на сообщения (reply), получение ошибок (error) и сигналов (signal) по умолчанию в политике разрешены.

Таким образом, предпочтительнее реализовывать системные службы в виде небольших, целевых программ, которые работают в одном процессе и требуют одного общеизвестного имени (имени службы) на шине. В результате для определения политики достаточно будет создать <allow>-правила для разрешений типа own, чтобы позволить процессам создавать собственные службы, и добавить параметр send_destination, чтобы разрешить трафик от некоторых или всех пользователей к данным службам.

Секции <policy> назначается один из четырех параметров:

- context — возможные значения: default, mandatory;
- at_console — возможные значения: true, false;
- user — имя пользователя или его идентификатор;
- group — имя группы или ее идентификатор.

Политики применяются к соединениям в следующем порядке:

1) all context="default";
2) all group="connection's user's group";
3) all user="connection's auth user";
4) all at_console="true";
5) all at_console="false";
6) all context="mandatory".

Порядок применения политик изменяется в том случае, когда политики пересекаются по условиям применения. Несколько политик с тем же пользователем/группой/контекстом применяются в порядке их появления в конфигурационном файле.

Возможные параметры правил политик приведены в таблице 15. Параметры правил <deny> определяют необходимость запрета, указанного в параметрах сообщения.

Таблица 15

Параметр	Описание
send_interface	Имя интерфейса отправителя сообщения
send_member	Имя метода или сигнала отправителя сообщения
send_error	Имя ошибки
send_destination	В качестве значения указывается имя. Для правила <deny> параметр означает, что сообщения не могут быть отправлены владельцем данного имени, а не данной службой. То есть, если соединение владеет службами A, B, C, и отправка к A запрещена, то отправка к B или C также будет запрещена

Продолжение таблицы 15

Параметр	Описание
send_type	Возможные значения: method_call, method_return, signal и error. Задает тип исходящего сообщения
send_path	Путь вида /path/name отправителя сообщения
receive_interface	Имя интерфейса получателя сообщения
receive_member	Имя метода или сигнала получателя сообщения
receive_error	Имя ошибки
receive_sender	В качестве значения указывается имя. Для правила <deny> параметр означает, что сообщения не могут быть получены владельцем данного имени, а не данной службой
receive_type	Возможные значения: method_call, method_return, signal и error. Задает тип получаемого сообщения
receive_path	Путь вида /path/name получателя сообщения
send_requested_reply	Возможные значения: true, false. Параметр работает аналогично параметру eavesdrop: разрешает или запрещает (<allow> или <deny>) сообщение, полученное в качестве ожидаемого ответа (requested_reply), соответственно, предшествует этому сообщение вызова метода (method_call). Данный параметр актуален только для ответных сообщений (error и method_return) и игнорируется для других типов сообщений. Для правила <allow> значение параметра send_requested_reply="true" является значением по умолчанию и указывает на то, что только запрошенные ответы разрешены правилом. Значение параметра send_requested_reply="false" означает, что правило позволяет любой ответ, даже если он не ожидался. Для правила <deny> значение параметра send_requested_reply="false" является значением по умолчанию и указывает на то, что правило работает только когда ответ не был запрошен. Значение параметра send_requested_reply="true" указывает на то, что правило работает всегда, независимо от состояния ожидания ответа
receive_requested_reply	Аналогично параметру send_requested_reply

Окончание таблицы 15

Параметр	Описание
eavesdrop	Возможные значения: true, false. Прослушивание (Eavesdropping) возникает, когда приложение получает сообщение, адресованное службе, которой не владеет приложение, или ответ на такое сообщение. Для правил <allow> значение параметра eavesdrop="true" указывает на то, что правило будет применяться даже когда осуществляется прослушивание. Значение параметра eavesdrop="false" задано по умолчанию и означает, что правило разрешает те сообщения, которые доставляются только указанному получателю. Для правил <deny> значение параметра eavesdrop="true" указывает на то, что правило будет применяться только тогда, когда выполняется прослушивание. Значение параметра eavesdrop="false" задано по умолчанию и означает, что правило применяется всегда, даже когда не осуществляется прослушивание. Параметр eavesdrop можно комбинировать только с правилами типа приема и передачи (send_* и receive_*)
own	Имя
own_prefix	В качестве значения указывается имя. Правило <allow own_prefix="a.b"/> позволяет владеть именем a.b или любым именем, начинающимся с a.b, например это выполняется для a.b.c или a.b.c.d, но не выполняется для a.bc или a.c. Актуально, когда службы, например Telepathy и ReserveDevice, определяют значения служб для поддеревьев общеизвестных имен, например org.freedesktop.Telepathy.ConnectionManager.<произвольное_продолжение> и org.freedesktop.ReserveDevice1.<произвольное_продолжение>
user	В качестве значения указывается имя пользователя. Правила <deny> означают, что данный пользователь не может подключиться к шине сообщений
group	В качестве значения указывается имя группы. Действие параметра аналогично параметру user

Запрет пользователя и/или группы необходимо осуществлять в политике context="default" или context="mandatory", а не внутри секции <policy>.

Отдельное правило <deny> может содержать комбинации параметров. В таком случае запрет выполняется только тогда, когда все параметры соответствуют сообщению.

Пример

Правило запрета сообщений от пользователя john:

```
<deny user="john"/>
```

Нельзя включать одновременно параметры send_* и receive_* в одно правило.

Необходимо с осторожностью использовать параметры send_interface и receive_interface, т.к. поле интерфейса в сообщениях является необязательным и явное указание запрета интерфейса может привести к блокировке сообщений без интерфейса для всех служб. Всегда следует использовать send_interface и receive_interface совместно с другими параметрами.

Примеры:

1. <deny send_interface="org.foo.Bar"
send_destination="org.foo.Service"/>

2. <deny send_destination="org.freedesktop.Service"
send_interface="org.freedesktop.System" send_member="Reboot"/>

3. <deny send_destination="org.freedesktop.System"/>

4. <deny receive_sender="org.freedesktop.System"/>

5. <deny group="enemies"/>

4.14.7.2. Формирование клиентских политик из политик шины

Обработка файла конфигурации и считывание файлов настроек, в том числе политик в parent. Если в обработчике возникает ошибка, то функция возвращает NULL и, соответственно, ничего не считывается.

Пример

```
get_correct_parser ->
bus_context_new ->
bus_context_reload_config  ->
include_file  ->

bus_config_load (const DBusString      *file,
                 dbus_bool_t            is_toplevel,
                 const BusConfigParser *parent,
                 DBusError             *error)
```

Инициализация обработчика в функции XML_SetElementHandler:

```
expat_StartElementHandler -> bus_config_parser_start_element ->
   start_policy_child -> append_rule_from_element
```

При анализе вызовов формирование политик происходит следующим образом:

```
bus_driver_handle_hello ?  bus_connection_complete
process_config_every_time ?  bus_connections_reload_policy
bus_connections_reload_policy или bus_connection_complete ->
bus_context_create_client_policy ->  bus_policy_create_client_policy
```

Функция `bus_policy_create_client_policy` разбирает политики шины `BusPolicy *` и создает `BusClientPolicy*` исходя из соединения `DBusConnection *`. Таким образом `BusPolicy *` на основе `DBusConnection *` соответствует `BusClientPolicy*`.

4.15. Средства управления мандатными ПРД

Для управления мандатными ПРД в локальной системе используются следующие графические утилиты:

1) `fly-fm` («Менеджер файлов») — управление мандатными атрибутами файлов;

2) `astra-systemsettings` («Параметры системы») — управление привилегиями и мандатными атрибутами пользователей, работа с пользователями и группами.

Более подробное описание утилит см. в электронной справке.

Для управления мандатными ПРД в локальной системе из командной строки используются следующие инструменты:

1) `pdpl-file` — управление мандатными атрибутами файлов, описание приведено в 4.15.1;

2) `pdp-id` — отображение мандатных атрибутов сессии пользователя ОС, описание приведено в 4.15.2;

3) `pdp-init-fs` — сценарий инициализации мандатных атрибутов ФС, описание приведено в 4.15.3;

4) `pdp-ls` — вывод, аналогично стандартной команде `ls`, информации о файлах с отображением мандатных атрибутов, описание приведено в 4.15.4;

5) `pdpl-ps`— управление мандатными атрибутами процессов, описание приведено в 4.15.5;

6) `pdpl-user`— управление допустимыми уровнями и категориями конфиденциальности пользователей ОС, описание приведено в 4.15.6;

7) `pdpl-group` — управление меткой целостности группы пользователей ОС, описание приведено в 4.15.7;

8) `pdp-exec` — запуск процессов в заданном окружении, описание приведено в 4.15.8;

9) `sumac` — запуск процесса с заданными уровнем и категорией конфиденциальности в отдельной графической сессии, описание приведено в 4.15.9;

10) `sumic` — запуск процесса с пониженной (заданной) меткой целостности, описание приведено в 4.15.10;

11) `userlev` — изменение БД уровней конфиденциальности, описание приведено в 4.15.11;

12) `usercat` — изменение БД категорий конфиденциальности, описание приведено в 4.15.12.

Для управления мандатными ПРД в режиме ЕПП с развернутым доменом FreeIPA возможно использовать веб-интерфейс FreeIPA либо инструмент командной строки `ipa user-mod` (подробная информация по использованию инструмента командной строки доступна на справочной странице `ipa help user-mod`).

4.15.1. pdpl-file

Инструмент `pdpl-file` предназначен для управления мандатными атрибутами (меткой безопасности, дополнительными мандатными атрибутами управления доступом и дополнительными атрибутами для МКЦ) сущностей ОС.

Синтаксис команды:

```
pdpl-file [параметр[...]] [уровень_конфиденциальности][:категория_целостности
     [:категория_конфиденциальности[:дополнительный_атрибут1,...]
     [:линейный_уровень_целостности]]] <сущность>
```

Уровень конфиденциальности и категория целостности могут быть заданы именами или десятичными значениями. Линейный уровень целостности задается десятичным значением. Категория конфиденциальности может быть задана именем или шестнадцатеричным значением (см. 4.2.1, 4.2.2 и 4.3.1).

Пример

Рекурсивно для всех файлов каталога `/tmp` изменить уровень конфиденциальности на `Секретно` и категорию конфиденциальности на `Категория_A` (уровень и категория должны быть определены в системе):

```
pdpl-file -Rv Секретно:0:Категория_A /tmp
```

Для присвоения сущности одновременно всех категорий, которые определены в системе, можно использовать значение `-1` для `<категория_конфиденциальности>`.

Пример

```
pdpl-file 1:0:-1 /tmp
```

Дополнительные атрибуты для мандатного управления доступом `ccnr`, `ehole`, `whole` и дополнительные атрибуты для МКЦ `silev`, `irelax`, `ssi` и `iinh` могут быть заданы числовыми значениями или именами через запятую.

Пример

```
pdpl-file 2:0:0:ccnr,irelax /tmp
```

Для присвоения сущности линейного уровня целостности необходимо указать его значение после дополнительных атрибутов.

Примеры:

1. Установка отрицательного линейного уровня целостности на файл:

```
pdpl-file ::::-128 file.txt
```

2. Установка нулевого линейного уровня целостности на каталог вместе с атрибутом iinh:

```
pdpl-file :::iinh:0 /tmp
```

Описание параметров инструмента `pdpl-file` приведено в таблице 16.

Таблица 16

Параметр	Описание
`-f`, `--silent`, `--quiet`	Не выводить сообщений об ошибках
`-v`, `--verbose`	Выводить диагностические сообщения для каждого файла
`-c`, `--changes`	То же, что и `--verbose`, но сообщать только об изменениях
`-u`, `--unite`	Объединить текущую метку безопасности сущности с указанной в качестве аргумента

Окончание таблицы 16

Параметр	Описание
`-s, --subtract`	Вычесть из текущей метки безопасности сущности метку безопасности, указанную в качестве аргумента. При этом для итоговой метки безопасности значения задаются по следующим правилам: - уровень конфиденциальности — минимальное значение из текущей метки безопасности и указанной в качестве аргумента; - метка целостности — минимальное значение из текущей метки безопасности и указанной в качестве аргумента (должно быть указано в десятичном виде); - категории конфиденциальности — из текущей метки безопасности вычитаются категории, указанные в качестве аргумента; - дополнительные атрибуты — из текущей метки безопасности вычитаются дополнительные атрибуты, указанные в качестве аргумента
`-R, --recursive`	Применить рекурсивно
`-r, --reverse`	Сначала файлы в каталоге, потом каталог
`-h, --help`	Вывести справку и выйти
`--version`	Вывести информацию о версии и выйти

4.15.2. pdp-id

Инструмент `pdp-id` выводит мандатные атрибуты сессии пользователя ОС.

Синтаксис команды:

```
pdp-id [параметры]
```

Параметры и их описание приведены в таблице 17.

Таблица 17

Параметр	Описание
`-a`	Игнорируется, добавлен для совместимости с программой `id`
`-l,--level`	Вывести только уровень конфиденциальности
`-i,--ilevel`	Вывести только категорию целостности
`-d,--ilinear`	Вывести только линейный уровень целостности
`-c,--category`	Вывести только категории конфиденциальности
`-n,--name`	Для параметров `-l`, `-i` и `-c` выводить имена вместо числовых значений
`-h,--help`	Вывести справку и выйти
`--version`	Вывести информацию о версии и выйти

При отсутствии параметров инструмент выводит строку текущих мандатных атрибутов сессии.

Пример

Выполнить команду:

```
pdp-id
```

Вывод команды:

```
Уровень конф.=2(Секретно), Уровень целостности=0(Низкий),
     Линейная целостность=(), Категории=0x1(Категория_A)
```

Вывод команды показывает, что текущая сессия пользователя имеет уровень конфиденциальности `Секретно`, минимальную категорию целостности, нулевой линейный уровень целостности и категорию `Категория_A`.

4.15.3. pdp-init-fs

Сценарий инициализации мандатных атрибутов ФС `pdp-init-fs` вызывается при инициализации и перезапуске системы для установки корректных мандатных атрибутов на системные файловые объекты (файлы и каталоги), начиная с корня ФС.

Для запуска сценария следует выполнить в консоли:

```
pdp-init-fs
```

Сценарий располагается в каталоге `/usr/sbin` и доступен для правки только администратору.

ВНИМАНИЕ! Настоятельно не рекомендуется вносить изменения в указанный сценарий без необходимости. Изменения требуются только в случае изменения максимального уровня конфиденциальности или максимальной категории целостности в системе.

4.15.4. pdp-ls

Инструмент `pdp-ls` выводит, аналогично стандартной команде `ls`, информацию о файлах и каталогах (по умолчанию о текущем каталоге).

Синтаксис команды:

```
pdp-ls [параметры] [имя_файла]
```

Данный инструмент поддерживает все параметры инструмента `ls` и используется аналогичным образом.

Дополнительно поддерживает параметр `-M` для вывода информации об атрибутах мандатного управления доступом и МКЦ:

```
pdp-ls -M [имя_ƍайла]
```

Также данный инструмент имеет следующие дополнительные особенности:

- если на файле установлены ACL, то к ним добавляется символ «+»;
- если на файле установлена ненулевая метка безопасности, то к ACL добавляется `m`;
- если на файле установлены списки регистрации событий, то к ACL добавляется `a`.

Пример

```
pdp-ls -M
```

Вывод команды:

```
итого 4
drwxr-xr-x--- 2 user user Уровень_0:Низкий:Нет:0x0 common
-rw-r--r--m-- 1 user user Секретно:Низкий:Категория_А,Категория_Б:0x0
      file.txt
```

4.15.5. pdpl-ps

Инструмент `pdpl-ps` позволяет считать мандатный контекст безопасности процесса, заданного параметром (идентификатором процесса).

Синтаксис команды:

```
pdpl-ps [-nzhv] <PID>
```

Только администратор может считывать мандатный контекст безопасности произвольного процесса, обычный пользователь может считывать контекст только с собственного процесса, для этого параметр должен иметь нулевое значение.

Параметры инструмента приведены в таблице 18.

Таблица 18

Параметр	Описание
`-n`, `--numeric`	Вывести информацию о контексте безопасности в численном виде
`-z`, `--iszero`	Если метка безопасности нулевая, то завершиться с кодом `0`, иначе `1`. Если не удалось получить метку безопасности процесса, то завершить с кодом `74`
`-h`, `--help`	Вывести справку и выйти
`--version`	Вывести информацию о версии и выйти

4.15.6. pdpl-user

Инструмент `pdpl-user` отображает и изменяет метки безопасности пользователей ОС.

Синтаксис команды:

```
pdpl-user [-dzhv [-l минимальный:максимальный уровень конфиденциальности]
     [-i максимальная метка целостности] [-c минимальная категория:
     максимальная категория]] <пользователь>
```

Параметры инструмента приведены в таблице 19.

Таблица 19

Параметр	Описание
`-d`, `--delete`	Удалить строку пользователя из файла
`-z`, `--zero`	Обнулить значения уровней и категорий
`-l`, `--level`	Установить допустимые уровни конфиденциальности
`-i`, `--ilevel`	Установить максимальную метку целостности
`-m`, `--maclabel`	То же, что и `-l` (используется для совместимости)
`-c`, `--category`	Установить допустимые категории конфиденциальности
`-h`, `--help`	Вывести справку и выйти
`-v`, `--version`	Вывести информацию о версии и выйти

Если для параметров `-m` или `-c` указано одно значение или одно значение с предшествующим двоеточием, то это значение интерпретируется как максимальное значение, если одно значение с последующим двоеточием — как минимальное.

Инструмент `pdpl-user` при успешном выполнении всегда выводит значения установленных допустимых меток безопасности.

Чтобы просмотреть текущие допустимые метки безопасности, выполнить команду без ключей:

```
pdpl-user <пользователь>
```

ВНИМАНИЕ! Значения `Низкий` и `Высокий` в выводе команды обозначают метку целостности, т.е. совокупность категории целостности и линейного уровня целостности.

Пример

```
pdpl-user -l Уровень_0:Уровень_3 -i 0 -c 0:Категория_2 user1
```

Команда установит для пользователя `user1` следующие мандатные атрибуты:

- минимальный уровень конфиденциальности — `Уровень_0(0)`;
- максимальный уровень конфиденциальности — `Уровень_3(3)`;
- максимальную метку целостности — `Низкий(0)`;
- минимальную категорию конфиденциальности — `0x0(0)` (без категорий);
- максимальную категорию конфиденциальности — `0x2(2)`.

При этом уровни конфиденциальности `Уровень_0`, `Уровень_3`, категория конфиденциальности `Категория_2` должны быть определены в системе. Значения уровней и категорий могут быть заданы в числовой форме.

4.15.7. pdpl-group

Инструмент `pdpl-group` позволяет устанавливать и просматривать метку целостности группы пользователей ОС (см. 4.3.1).

Синтаксис команды:

```
pdpl-group [параметр] <группа>
```

Имя группы задается в текстовом формате. Описание параметров приведено в таблице 20.

Таблица 20

Параметр	Описание
`-d, --delete <группа>`	Удалить строку группы из файла
`-z, --zero <группа>`	Обнулить значение уровней и категорий целостности группы
`-i, --ilevel <метка_целостности> <группа>`	Установить максимальную метку целостности

Окончание таблицы 20

Параметр	Описание
`-h`, `--help`	Вывести справку по использованию инструмента
`-v`, `--version`	Вывести информацию о версии инструмента

Пример

Присвоить группе `group1` категорию целостности, равную `63`:

```
pdpl-group -i 63 group1
```

Результат выполнения команды:

```
minimal pdpl: Уровень_0:Низкий:Нет:0x0
0:0:0x0:0x0
maximal pdpl: Уровень_0:Высокий:Нет:0x0
0:63:0x0:0x0
```

4.15.8. pdp-exec

Инструмент `pdp-exec` позволяет администратору запускать процессы в заданном окружении:

- имя пользователя, от имени которого запускается процесс;
- метка безопасности процесса;
- PARSEC-привилегии.

При использовании инструмента `pdp-exec` следует учитывать, что возможен запуск процесса без применения мандатного контекста безопасности, поэтому использование `pdp-exec` должно быть регламентировано и ограничено.

Синтаксис команды:

```
pdp-exec [параметр[параметр...]] [--] <команда_запуска_процесса>
     [параметры_запуска_процесса]
```

В случае если с командой заданы параметры ее запуска, то указание символов «`--`» перед командой обязательно.

Описание параметров инструмента `pdp-exec` приведено в таблице 21.

Таблица 21

Параметр	Описание
-c <привилегии>, --capability=<привилегии>	Установить процессу указанные привилегии в качестве эффективных (текущих), наследуемых и разрешенных
-u <имя_пользователя>, --user=<имя_пользователя>	Запустить процесс от имени указанного пользователя
-l <метка_безопасности>, --label=<метка_безопасности>	Установить процессу указанную метку безопасности
-v, --version	Вывести информацию о версии и выйти
-h, --help	Вывести справку и выйти

Привилегии задаются в виде битовой маски (как правило, в шестнадцатеричном виде). Соответствие отдельных битов полномочиям приведено на справочной странице man parsec_capset, а также в таблице 12.

Метка безопасности задается в виде:

```
[уровень_конфиденциальности][:метка_целостности[:категория_конфиденциальности]]
```

Примеры:

1. Перезапустить службу dbus с PARSEC-привилегией PARSEC_CAP_PRIV_SOCK:

```
pdp-exec -c 0x100 -- /etc/init.d/dbus restart
```

2. Запустить оболочку bash от имени пользователя secretuser с PARSEC-привилегией PARSEC_CAP_SIG и с меткой безопасности 1:1:

```
pdp-exec -c 0x40 -u secretuser -l 1:1 -- bash
```

Более подробное описание pdp-exec приведено на справочной странице man pdp-exec.

4.15.9. sumac

Инструмент sumac используется для запуска процесса с заданными уровнем и категорией конфиденциальности в отдельной графической сессии с использованием виртуального графического сервера Xephyr. Пользователь может запускать процесс только в пределах разрешенных ему уровней и категорий.

Синтаксис команды:

```
sumac [параметры] <команда_запуска_процесса>
```

Параметры инструмента sumac приведены в таблице 22.

Таблица 22

Параметр	Описание
-l , --level=<значение>	Запустить процесс с указанным уровнем конфиденциальности
-c , --category=<значение>	Запустить процесс с указанной категорией конфиденциальности
-i , --stdin=<файл>	Перенаправить stdin запущенного процесса в указанный файл. Не используется
-o , --stdout=<файл>	Перенаправить stdout запущенного процесса в указанный файл. Не используется
-e , --stderr=<файл>	Перенаправить stderr запущенного процесса в указанный файл. Не используется
-x, --xauth	Попытаться создать запись в .Xauthority. В случае неудачи прервать выполнение процесса. Не используется
-h, --help	Вывести справку и выйти
-v, --version	Вывести информацию о версии и выйти

ВНИМАНИЕ! Для запуска процесса на уровне конфиденциальности, отличном от уровня текущей сессии, не должна стоять блокировка использования sumac в графической утилите astra-systemsettings («Параметры системы», см. электронную справку) и пользователю должна быть назначена привилегия PARSEC_CAP_SUMAC (см. 4.7).

ВНИМАНИЕ! Для использования sumac при включенном МКЦ также должно быть включено МКЦ на файловой системе.

Если указанный уровень конфиденциальности выше текущего, т. е. происходит увеличение уровня, то переменные окружения наследуются от текущего процесса. Текущие переменные окружения сбрасываются, чтобы избежать утечки информации. Также при порождении нового процесса закрываются все файловые дескрипторы, метка безопасности которых не совпадает с указанной в командной строке. В том числе закрываются stdin, stdout, stderr. Вследствие этого сообщения об ошибках, которые могут произойти при запуске процесса, не будут отображены. Параметры -x, -i, -o и -e ранее использовались для перенаправления stdin, stdout, stderr в файлы. В текущей версии они не выполняют полезных действий, но сохранены для совместимости с предыдущими версиями. Если в качестве значений параметров указаны имена файлов, то эти файлы создаются с нулевым размером.

ВНИМАНИЕ! Запуск процесса с понижением уровня конфиденциальности или с сокращением набора категорий конфиденциальности запрещен для предотвращения утечки информации на более низкие уровни.

Окно графического приложения, запущенного в отдельной сессии на другом уровне, будет иметь цветную рамку, цвет которой соответствует уровню конфиденциальности приложения. Описание цветовой индикации приведено в документе РУСБ.10015-01 93 01 «Операционная система специального назначения «Astra Linux Special Edition». Руководство пользователя».

Пример

Запуск графического приложения `xterm` с уровнем конфиденциальности 2 и категорией конфиденциальности `0xffff`:

```
sumac -l 2 -c 0xffff xterm
```

4.15.10. sumic

Инструмент `sumic` позволяет запускать процессы с меткой целостности ниже, чем метка целостности процесса-родителя, или несравнимой с ней. При этом запускаемый с использованием `sumic` процесс будет иметь метку целостности не выше метки целостности исполняемого файла, из которого он запущен. При запуске процесса с помощью `sumic` запрещается наследование открытых в процессе-родителе ресурсов, имеющих метку целостности, превышающую метку целостности создаваемого процесса. Запуск графической утилиты с использованием `sumic` выполняется в изолированном X-сервере.

Синтаксис команды:

```
sumic [параметры] [--] <команда_запуска_процесса>
     [параметры_запуска_процесса]
```

Описание параметров инструмента `sumic` приведено в таблице 23.

Таблица 23

Параметр	Описание
`-i <категория_целостности> [:линейный_уровень_целостности]`	Метка целостности, с которой будет запущен указанный процесс. В случае отсутствия параметра процесс будет запущен с нулевой меткой целостности
`-v, --version`	Вывести информацию о версии и выйти
`-h, --help`	Вывести справку и выйти

4.15.11. userlev

Инструмент `userlev` служит для просмотра и изменения в БД уровней конфиденциальности. Для просмотра всех уровней конфиденциальности инструмент следует запускать без параметров. Вносить изменения в БД уровней конфиденциальности может только администратор.

Синтаксис команды:

```
userlev [параметры] <уровень>
```

Параметры инструмента приведены в таблице 24.

Таблица 24

Параметр	Описание
`-d, --delete`	Удалить уровень конфиденциальности из БД
`-a, --add <значение>`	Добавить новый уровень конфиденциальности в БД
`-r, --rename <новое_имя>`	Переименовать существующий уровень конфиденциальности
`-m, --modify <новое_значение>`	Изменить значение уровня конфиденциальности
`-h, --help`	Вывести справку и выйти
`--version`	Вывести информацию о версии и выйти

4.15.12. usercat

Инструмент `usercat` служит для просмотра и изменения БД категорий конфиденциальности. Для просмотра всех категорий конфиденциальности команду следует запускать без параметров. Вносить изменения в БД категорий конфиденциальности может только администратор.

Синтаксис команды:

```
usercat [параметры] <категория_конфиденциальности>
```

Параметры инструмента приведены в таблице 25.

Таблица 25

Параметр	Описание
`-d, --delete`	Удалить категорию конфиденциальности из БД
`-a, --add <значение>`	Добавить новую категорию конфиденциальности в БД

Окончание таблицы 25

Параметр	Описание
`-r, --rename <новое_имя>`	Переименовать существующую категорию конфиденциальности
`-m, --modify <новое_значение>`	Изменить значение категории конфиденциальности
`-h, --help`	Вывести справку и выйти
`--version`	Вывести информацию о версии и выйти

4.16. Средства управления привилегиями пользователей и процессов

Для управления привилегиями пользователей и процессов используется графическая утилита `astra-systemsettings` («Параметры системы», см. электронную справку).

Также для управления привилегиями пользователей и процессов используются инструменты командной строки `usercaps`, `execaps` и `pscaps`, описание которых приведено, соответственно, в 4.16.1, 4.16.2 и 4.16.3.

ВНИМАНИЕ! Управление привилегиями пользователей выполняется только администратором с максимальной категорией целостности, установленной в ОС.

4.16.1. usercaps

Инструмент командной сроки `usercaps` позволяет просматривать и устанавливать привилегии пользователей и предназначен для использования администратором через механизм `sudo`.

Каждая привилегия имеет номер, шестнадцатеричное значение и название. Информация о привилегия отображается в выводе команд:

1) для Linux-привилегий:

```
usercaps -L
```

Пример

```
23  0x00800000  cap_sys_nice
```

2) для PARSEC-привилегий:

```
usercaps -M
```

Пример

```
17  0x020000  parsec_cap_inherit_integrity
```

Синтаксис команды:

```
usercaps [параметр] [пользователь]
usercaps [-l linux-привилегии] [-m PARSEC-привилеги] <пользователь>
```

Указать <linux-привилегии> и <PARSEC-привилегии> в команде можно тремя способами:

1) в виде названий привилегий в верхнем или нижнем регистре, разделенных запятой «,» или двоеточием «:»;

2) в виде номеров привилегий со знаками «+» или «-», разделенных запятой «,» или двоеточием «:»;

3) в виде шестнадцатеричной битовой маски, соответствующей набору привилегий.

Верхний регистр для названий и знак «+» для номеров добавляют пользователю соответствующие привилегии. Нижний регистр и знак «-» снимают соответствующие привилегии с пользователя.

При использовании битовой маски существующий набор привилегий пользователя заменяется на набор, соответствующий указанной битовой маске. Маска может вводиться в упрощенном виде: `150` будет интерпретировано как `0x000150`.

Описание параметров `usercaps` приведено в таблице 26.

Таблица 26

Параметр	Описание
`-d, --delete`	Удалить файл привилегий пользователя из каталога `/etc/parsec/capdb`
`-z, --zero`	Сбросить все привилегии пользователя
`-f, --full`	Присвоить все возможные привилегии пользователю
`-l <linux-привилегии>, --linux <linux-привилегии>`	Изменить Linux-привилегии пользователя
`-m <PARSEC-привилегии>, --parsec <PARSEC-привилегии>`	Изменить PARSEC-привилегии пользователя
`-L, --Linux`	Вывести список возможных Linux-привилегий
`-M, --PARSEC`	Вывести список возможных PARSEC-привилегий
`-n, --numeric`	Вывести привилегии пользователя в шестнадцатеричном формате

Окончание таблицы 26

Параметр	Описание
`-x <маска>`, `--hex <маска>`	Вывести PARSEC-привилегии, соответствующие указанной битовой маске. Маска может быть задана в форматах `dec`, `hex` или `oct` следующим образом: 1) `70` — `dec`; 2) `0x46` — `hex`; 3) `0106` — `oct`
`-h, --help`	Вывести справку и выйти
`-v, --version`	Вывести информацию о версии и выйти

Параметры `-f` и `-z` могут использоваться вместе с дополнительными параметрами `-L` или `-M` для присвоения или сброса у пользователя сразу всех Linux- или PARSEC-привилегий соответственно.

При вызове инструмента без параметров выводится список пользователей с указанием битовых масок их привилегий:

```
usercaps
```

Результат выполнения команды:

```
root -l 0x0 -m 0x0
user -l 0x0 -m 0x123
```

Примеры:

1. Вывести список возможных Linux-привилегий с номерами, шестнадцатеричными значениями и названиями:

```
usercaps -L
```

2. Вывести список возможных PARSEC-привилегий с номерами, шестнадцатеричными значениями и названиями:

```
usercaps -M
```

3. Вывести все привилегии пользователя:

```
usercaps <пользователь>
```

4. Сбросить все PARSEC-привилегии пользователя:

```
usercaps -z -M <пользователь>
```

5. Добавить пользователю привилегию `parsec_cap_audit` (номер привилегии — `1`, битовая маска — `0x000002`) и снять привилегию `parsec_cap_chmac` (номер привилегии — `3`, битовая маска — `0x000008`):

```
usercaps -m PARSEC_CAP_AUDIT,parsec_cap_chmac <пользователь>
```

или

```
usercaps -m +1,-3 <пользователь>
```

6. Установить пользователю набор привилегий, соответствующий битовой маске `0x000130`:

```
usercaps -m 130 <пользователь>
```

Результат выполнения команды:

```
PARSEC-привилегии пользователя '<пользователь>':
 4  0x000010  parsec_cap_ignmaclvl
 5  0x000020  parsec_cap_ignmaccat
 8  0x000100  parsec_cap_priv_sock
```

Описание `usercaps` также приведено на справочной странице `man usercaps`.

4.16.2. execaps

Инструмент командной строки `execaps` может быть использован администратором для запуска процесса с одновременной установкой выбранных PARSEC-привилегий.

Синтаксис команды:

```
execaps [параметр] [-c привилегии] [--] [команда]
```

Привилегии задаются в виде битовой маски (как правило, в шестнадцатеричном виде). Соответствие отдельных битов полномочиям приведено на справочной странице `man parsec_capset`, а также в таблице 12.

В качестве команды может быть задана программа с параметрами ее запуска. Если команда задается с параметрами запуска, то указание символов «--» перед командой обязательно.

Пример

Выполнить команду `pscaps` с привилегией `0x000008`:

```
execaps -c 0x000008 pscaps 0
```

Процесс `pscaps` запустится с заданными привилегиями и отобразит их в выводе:

```
00000008 00000008 00000008
```

Описание основных параметров `execaps` приведено в таблице 27.

Таблица 27

Параметр	Описание
`-c <привилегии>`, `--capability=<привилегии>`	Установить процессу указанные привилегии в качестве эффективных (текущих), наследуемых и разрешенных
`-f, --force`	Вызвать процесс, даже если не удалось установить привилегии
`-h, --help`	Вывести справку и выйти
`-v, --version`	Вывести информацию о версии и выйти

Более подробное описание `execaps` приведено на справочной странице `man execaps`.

4.16.3. pscaps

Инструмент командной строки `pscaps` применяется для просмотра и установки PARSEC-привилегий процесса.

Синтаксис команды:

```
pscaps <параметр>
pscaps <PID> [текущие_привилегии [разрешенные_привилегии
     [наследуемые_привилегии]]]
```

где `<PID>` — идентификатор процесса.

Если в качестве параметра указан только идентификатор процесса `<PID>`, то `pscaps` показывает набор PARSEC-привилегий (указанных в виде битовых масок привилегий) процесса.

При указании в качестве <PID> значение 0 будет показан набор привилегий процесса pscaps.

При указании в команде битовых масок привилегий (в десятичном или шестнадцатеричном виде) будут изменены привилегии процесса pscaps. В этом случае в качестве значения <PID> должно быть указано «0» или идентификатор процесса pscaps.

Описание параметров pscaps приведено в таблице 28.

Таблица 28

Параметр	Описание
-h, --help	Вывести справку и выйти
-v, --version	Вывести информацию о версии и выйти

Описание pscaps также приведено на справочной странице man pscaps.

4.17. Мандатное управление доступом в комплексах программ гипертекстовой обработки данных и электронной почты

Обеспечение мандатного управления доступом в комплексах программ гипертекстовой обработки данных и электронной почты реализовано на основе программного интерфейса библиотек подсистемы безопасности PARSEC.

На серверах комплексов программ гипертекстовой обработки данных и электронной почты при обработке запросов на соединение выполняется получение мандатного контекста соединения, унаследованного от субъекта (процесса). Сокет сервера, ожидающий входящих запросов на соединение, работает в контексте процесса, имеющего привилегию для приема соединений с любыми уровнями секретности.

После установки соединения и успешного прохождения процедуры идентификации и аутентификации пользователя процесс сервера, обрабатывающий запросы пользователя, переключается в контекст безопасности пользователя, сбрасывает привилегии, обрабатывает запросы пользователя и завершается.

В комплексе программ гипертекстовой обработки данных пользователь получает доступ к ресурсам, являющимся объектами ФС. Комплекс программ электронной почты использует технологию maildir, обеспечивающую хранение почтовых сообщений в виде отдельных объектов ФС. Создаваемые файлы почтовых сообщений маркируются метками безопасности, унаследованными от процесса-создателя. Таким образом, в обоих комплексах программ ресурсы, к которым осуществляется доступ от имени серверных процессов, обрабатывающих запросы пользователей, являются объектами ФС. Следовательно, доступ к защищаемым ресурсам при приеме и обработке запросов пользователей в процессе функционирования

серверов комплексов программ гипертекстовой обработки данных и электронной почты подчиняется мандатным ПРД.

4.18. Настройка параметров ядра в загрузчике GRUB 2

В загрузчике GRUB 2 возможно задать параметры командной строки ядра PARSEC, приведенные в таблице 29.

Таблица 29

Параметр	Описание
`parsec.max_ilev`	Задает максимальную категорию целостности (воспринимаемую модулями) в ОС на момент запуска. Допустимые значения от 0 до 255. При установке ОС задается по умолчанию значение 63. **ВНИМАНИЕ!** Если в системе присутствуют файлы с категорией целостности выше задаваемого в данный момент значения, это может вызвать отказ доступа к данным файлам (сокетам, каталогам и т. д.)
`parsec.ccnr_relax`	При заданном значении 1 позволяет непривилегированному пользователю выполнять запись файлов с разным уровнем конфиденциальности в контейнер (каталог) с установленным дополнительным мандатным атрибутом `ccnr`. Значение по умолчанию `0`. Параметр не применяется при включенном расширенном режиме МКЦ (см. 4.5)
`parsec.noload_files`	Запрет загрузки модулей ядра, встроенного программного обеспечения, образов `kexec`, ключей и сертификатов открытого ключа X.509 привилегированным пользователем с низкой категорией целостности (`root`, `uid=0`). Допустимые значения: `1` или `0` (`y` или `n`). Значение по умолчанию `1` (загрузка запрещена)
`parsec.ccnr_reject`	Запрет установки дополнительного мандатного атрибута `ccnr` привилегированным пользователем (`root`, `uid=0`). Допустимые значения: `1` или `0` (`y` или `n`). Значение по умолчанию `0` (установка разрешена)
`parsec.enable_exec_on_fuse`	Разрешение запуска сценариев и исполняемых файлов из файловых систем, смонтированных с помощью файловой системы FUSE. Допустимые значения: `1` или `0`. Значение по умолчанию `0` (запуск запрещен)

Окончание таблицы 29

Параметр	
parsec.rename_mask	Позволяет управлять механизмом сложения меток безопасности и дополнительных атрибутов файлов при их редактировании и/или переименовании текстовыми редакторами (предотвращает сброс меток безопасности и дополнительных атрибутов файлов). Значение параметра представляется в виде битовой маски, в которой каждый бит отвечает за установку соответствующего флага. В системе битовая маска задается соответствующим десятичным значением: - первый бит RENAME_MASK_LEV (двоичное значение 00001, десятичное значение 1) — объединение уровней конфиденциальности; - второй бит RENAME_MASK_CAT (двоичное значение 00010, десятичное значение 2) — объединение категорий конфиденциальности; - третий бит RENAME_MASK_ILEV (двоичное значение 00100, десятичное значение 4) — объединение меток целостности; - четвертый бит RENAME_MASK_FLAGS (двоичное значение 01000, десятичное значение 8) — объединение атрибута ssi; - пятый бит (двоичное значение 10000, десятичное значение 16) — определяет, будет ли добавлена запись аудита при объединении меток. Значение по умолчанию 0 (метка безопасности и дополнительные атрибуты сущности сбрасываются)

После установки ОС при необходимости изменение настроек загрузчика GRUB 2 осуществляется с использованием графической утилиты `fly-admin-grub2` (см. электронную справку).

Также параметры загрузчика GRUB 2 могут быть изменены в конфигурационном файле `/etc/default/grub`.

ВНИМАНИЕ! После внесения изменений в настройки загрузчика GRUB 2 необходимо в ОС от имени администратора выполнить команду:

```
sudo update-grub
```

5. ЗАЩИТА СРЕДЫ ВИРТУАЛИЗАЦИИ

ОС разработана с учетом применения встроенных средств защиты в виртуальной инфраструктуре и включает в свой состав ядро с поддержкой технологии виртуализации[1)] Kernel-based Virtual Machine (KVM, гипервизор II-типа).

ОС предоставляет возможность создания и защиты среды виртуализации с обеспечением выполнения следующих функций безопасности:

- доверенная загрузка виртуальных машин;
- контроль целостности;
- регистрация событий безопасности;
- управление доступом;
- резервное копирование;
- управление потоками информации;
- защита памяти;
- ограничение программной среды;
- идентификация и аутентификация пользователей;
- централизованное управление образами виртуальных машин и виртуальными машинами.

Основными средствами, необходимыми для создания среды виртуализации, являются:

- сервер виртуализации `libvirt`;
- программа эмуляции аппаратного обеспечения QEMU.

Управление средой виртуализации обеспечивается инструментом командной строки `virsh` с использованием программного интерфейса `libvirt`, который предоставляет средства создания и управления ВМ: настройка конфигурации и запуск, управление файлами-образами дисковых носителей, управление виртуальными сетевыми адаптерами и сетями, формирование контекста функционирования ВМ в виде процесса ОС.

5.1. Дискреционное управление доступом в среде виртуализации

Работа в среде виртуализации `libvirt` подчиняется правилам дискреционного управления доступом и возможна только после прохождения обязательной процедуры аутентификации.

Дискреционное управление доступом при работе с сервером виртуализации `libvirt` осуществляется совместным использованием модуля дискреционного управления доступом `dac` и специально разработанного модуля мандатного управления доступом `parsec`, взаимо-

[1)] Недоступно в режиме «Мобильный».

действующего с подсистемой безопасности PARSEC ОС. Модуль поддержки дискреционного управления доступом `dac` использует дискреционные атрибуты, состоящие из уникальных идентификаторов. Основанием для принятия решения о предоставлении доступа является сравнение дискреционных атрибутов ВМ и дискреционных атрибутов пользователя с учетом выполняемой операции, а также с учетом роли пользователя.

Дискреционные атрибуты ВМ могут быть динамическими и статическими:

- динамические генерируются в момент запуска ВМ на основе атрибутов запускающего пользователя (его уникального идентификатора);
- статические задаются администратором в конфигурации ВМ и определяют атрибуты ее запуска.

Все операции с виртуальной машиной разделяются на непривилегированные и привилегированные, например операции получения списка виртуальных машин или информации о конфигурации виртуальной машины являются операциями непривилегированными, а операции создания, удаления или изменения конфигурации виртуальных машин — привилегированными.

Привилегированные операции по изменению состава или конфигурации виртуальных машин требуют вхождения пользователя в локальную административную группу, имя которой задается в конфигурационном файле `/etc/libvirt/libvirtd.conf` в качестве значения параметра `admin_group`, значение по умолчанию `libvirt-admin`:

```
admin_group = "libvirt-admin"
```

Непривилегированные действия доступны пользователям группы `libvirt`.

5.2. Мандатное управление доступом к виртуальной машине

Работа в среде виртуализации `libvirt` подчиняется правилам мандатного управления доступом и возможна только после прохождения обязательной процедуры аутентификации.

Мандатное управление доступом при работе с сервером виртуализации `libvirt` выполняется модулем поддержки мандатного управления доступом `parsec`, разработанным с использованием прикладного программного интерфейса драйверов доступа `libvirt`. Модуль `parsec` использует метку безопасности подсистемы безопасности PARSEC ОС. Основанием для принятия решения о предоставлении доступа является сравнение меток безопасности ВМ и пользователя с учетом выполняемой операции.

Метка безопасности ВМ может быть динамической или статической:

- динамическая генерируется в момент запуска ВМ на основе метки безопасности запускающего пользователя;

- статическая задается администратором в конфигурации ВМ и определяет контекст безопасности ее запуска.

Все операции с ВМ разделяются на операции чтения и записи, например операции получения списка виртуальных машин или информации о конфигурации конкретной машины являются операциями чтения, а операции создания, удаления или изменения конфигурации ВМ — операциями записи.

Мандатное управление доступом осуществляется по следующим правилам:

1) создаваемая ВМ получает метку безопасности создающего ее пользователя;

2) образ ВМ не обладает меткой безопасности (если для него не задана статическая метка безопасности);

3) запущенная ВМ наследует метку безопасности запускающего пользователя (это отражается в виде наличия динамических меток безопасности как у процесса ВМ в ОС, так и у файлов-образов и устройств, принадлежащих ВМ);

4) доступ к функционирующей ВМ предоставляется только при равенстве меток безопасности пользователя и ВМ;

5) доступ к информации о ВМ и ее конфигурации предоставляется в соответствии с правилами мандатного управления доступом с учетом меток безопасности пользователя и ВМ.

В случае если операционная система ВМ не реализует мандатное управление доступом самостоятельно и/или не поддерживает классификационные метки по ГОСТ Р 58256-2018, запуск ВМ обеспечивается с классификационной меткой, соответствующей классификационной метке сессии пользователя. В таком случае мандатное управление доступом на основе классификационной метки процесса ВМ и соответствующей маркировки сетевых пакетов обеспечивается средствами ОС узла виртуализации.

Для исключения возможности установки вредоносного программного обеспечения и хранения защищаемых данных на виртуальном диске используется режим запуска ВМ «только чтение».

Режим запуска ВМ «только чтение» регламентируется дополнительными организационно-техническими мерами, состав которых согласуется с подразделением, ответственным за защиту информации.

В случае если операционная система ВМ реализует мандатное управление доступом и поддерживает классификационные метки по ГОСТ Р 58256-2018, запуск ВМ выполняется с учетом организационно-технических мер и в соответствии с политикой разграничения доступа на объекте информатизации одним из разрешенных способов:

1) в режиме «только чтение» с нулевой классификационной меткой в целях исключения влияния средств мандатного управления доступом ОС узла виртуализации

на маркировку сетевых пакетов, выполненную средствами защиты информации операционной системы ВМ;

2) в режиме «только чтение» при соответствии классификационной метки сессии пользователя, инициирующего запуск ВМ, и классификационной метки сессии в операционной системе ВМ так, чтобы средства ОС узла виртуализации заменяли значения классификационных меток, ранее установленные операционной системой ВМ, на значение классификационной метки текущей сессии в операционной системе ВМ;

3) без включения режима «только чтение» с нулевой классификационной меткой в целях исключения влияния средств мандатного управления доступом ОС узла виртуализации на маркировку сетевых пакетов, выполненную средствами защиты информации операционной системы ВМ.

Управление ВМ и доступ к файлу образа ВМ должен предоставляться уполномоченным пользователям только с помощью средств управления виртуализации.

Особенности настройки и применения любого из перечисленных способов приводятся в эксплуатационной документации на автоматизированную систему и/или отдельной инструкции по защите информации, подлежащей согласованию с подразделением, ответственным за защиту информации.

Примечания:

1. В случае использования в качестве гостевой системы ОС виртуальная машина не должна запускаться в мандатном контексте. Вместо этого необходимо выполнять удаленный вход с требуемым уровнем конфиденциальности средствами ОС.

2. Настоятельно рекомендуется использовать режим «только чтение» при запуске виртуальных машин в ненулевом мандатном контексте.

5.3. Ролевое управление доступом в среде виртуализации

В ОС ролевое управление доступом при работе с сервером виртуализации `libvirt` реализуется как с использованием драйвера доступа `polkit`, так и с использованием драйвера доступа `parsec`. Ролевое управление доступом обеспечивает разграничение возможностей выполнения привилегированных операций со средствами виртуализации.

Драйвер управления доступом `polkit` применим только в случае, если соединения с `libvirt` ограничены сокетом домена UNIX. Если соединения выполняются с сокетом TCP, идентифицирующая информация субъекта будет недоступна и доступ субъектов доступа к операциям с сервером виртуализации будет запрещен. Если субъект подключается по SSH, то будет идентифицирован локальный пользователь SSH.

Драйвер управления доступом parsec не зависит от сокета и позволяет осуществлять удаленные соединения. Проверки разрешений на использование полномочий ролей осуществляются на хосте сервера виртуализации.

Для реализации ролевой политики должен быть установлен пакет astra-kvm-secure.

5.3.1. Настройка ролей

Реализация ролевого метода управления доступом подразумевает разграничение доступа по ролям:

- администратор средства виртуализации;
- администратор безопасности средства виртуализации;
- разработчик виртуальной машины;
- администратор виртуальной машины.

Назначение ролей и полномочий осуществляется администратором системы с высокой меткой целостности.

Для работы в средстве виртуализации для ролей зарезервированы системные группы libvirt-dev, libvirt-adm, libvirt-admin, astra-audit и libvirt-admvm.

Роль администратора средства виртуализации позволяет создавать и управлять учетными записями пользователей средств виртуализации, назначать права доступа пользователям средства виртуализации к виртуальным машинам, создавать и удалять виртуальное оборудование, изменять конфигурации виртуального оборудования, управлять доступом виртуальных машин к физическому и виртуальному оборудованию, управлять квотами доступа виртуальных машин к физическому и виртуальному оборудованию, управлять перемещением виртуальных машин, удалять виртуальные машины, запускать и останавливать виртуальные машины, создавать снимки состояния виртуальных машин (включающих файлы конфигурации виртуальной машины, образа виртуальной машины и образа памяти виртуальной машины). Пользователю с данной ролью необходимо назначить возможность работы с высокой меткой целостности, а также:

1) при использовании драйвера polkit:

   а) включить его в группы libvirt-adm, libvirt-admin, libvirt, libvirt-qemu, kvm и astra-admin;

   б) добавить в /etc/sudoers для получения полномочий на применение механизма sudo:

   - открыть файл:

```
sudo visudo /etc/sudoers
```

- добавить строку:

```
%libvirt-adm ALL=(ALL:ALL) ALL
```

2) при использовании драйвера parsec:

а) включить его в группы libvirt-admin, libvirt, libvirt-qemu, kvm и astra-admin;

б) добавить в /etc/sudoers для получения полномочий на применение механизма sudo:

- открыть файл:

```
sudo visudo /etc/sudoers
```

- добавить строку:

```
%libvirt-admin ALL=(ALL:ALL) ALL
```

Роль администратора безопасности средства виртуализации позволяет иметь доступ на чтение к журналу событий безопасности средств виртуализации, формировать отчеты с учетом заданных критериев отбора, осуществлять выгрузку (экспорт) данных из журнала событий безопасности средства виртуализации. Пользователя с данной ролью необходимо включить в группу astra-audit и исключить из групп libvirt, libvirt-qemu, kvm.

Роль разработчика виртуальной машины позволяет создавать виртуальные машины и изменять конфигурации виртуальных машин. Пользователя с данной ролью необходимо включить в группы:

- при использовании драйвера polkit — в группы libvirt-dev, libvirt-admin, libvirt, libvirt-qemu и kvm;
- при использовании драйвера parsec — в группы libvirt-dev, libvirt, libvirt-qemu и kvm.

Роль администратора виртуальной машины позволяет осуществлять доступ пользователя средства виртуализации к виртуальной машине из интерфейса средства виртуализации. Пользователя с данной ролью необходимо включить в группу libvirt-admvm. Дополнительно для осуществления доступа к ВМ пользователю из группы libvirt-admvm должны быть назначены права доступа к данной ВМ путем настройки правил дискреционной политики доступа (ACL). Для этого необходимо в графической утилите virt-manager:

1) перейти в свойства ВМ;

2) выбрать раздел «Безопасность»;

3) во вкладке «Подробности» в таблице «Дискреционный контроль доступа» для пользователя установить разрешение «Использование» в соответствии с рис. 1.

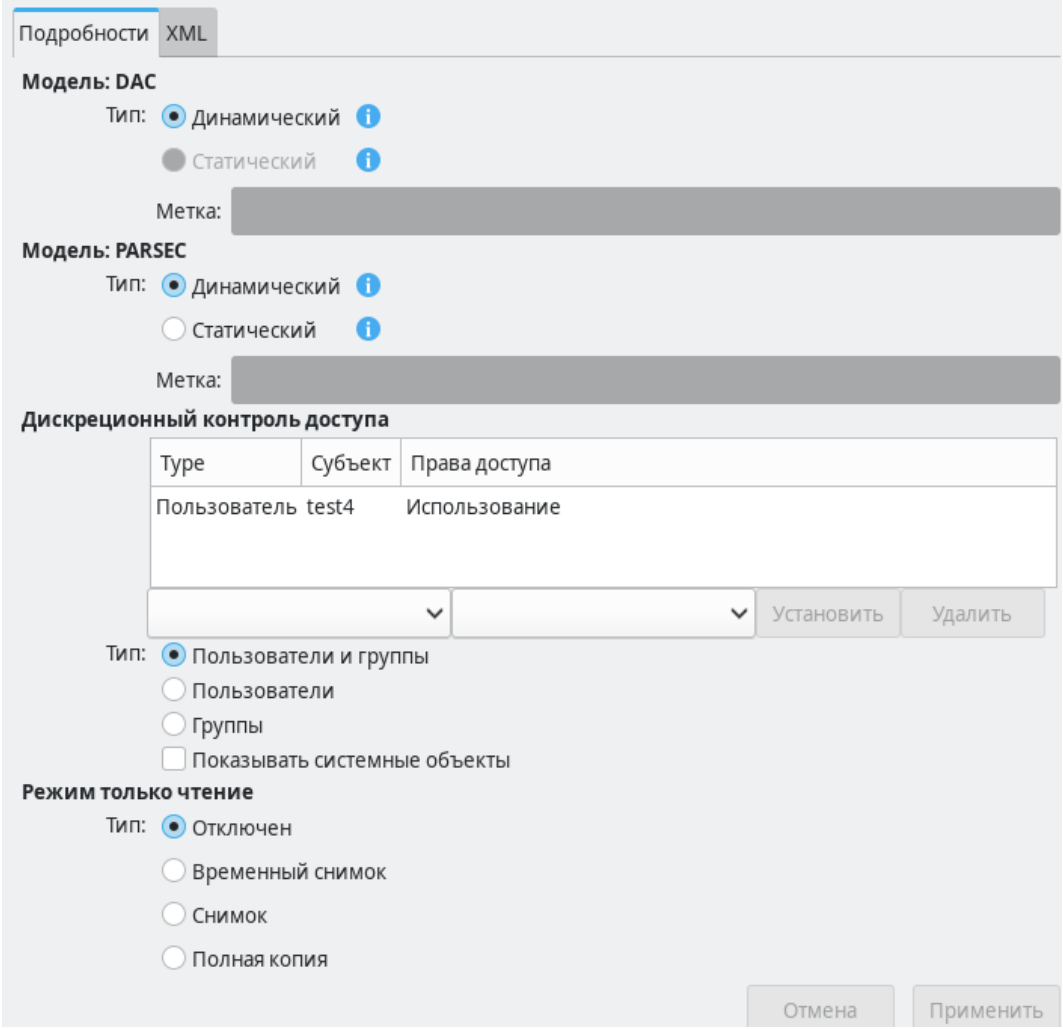

Рис. 1

Принятие решения о доступе пользователя к ВМ осуществляется на основе проверки вхождения пользователя в группу `libvirt-admvm` и наличия для пользователя соответствующего права доступа в ACL данной ВМ.

5.3.2. Настройка ролевого управления доступом с использованием драйвера доступа polkit

Для включения ролевого управления доступом на основе драйвера доступа `polkit` необходимо в конфигурационном файле `/etc/libvirt/libvirtd.conf` для параметра `access_drivers` задать значение `[ "polkit", "parsec" ]`:

```
access_drivers = ["polkit", "parsec" ]
```

Затем перезагрузить службу `libvirtd`, выполнив команды:

```
sudo systemctl reload libvirtd
sudo systemctl restart libvirtd
```

При установке сервера виртуализации libvirt в каталоге `/usr/share/polkit-1/actions/` создаются по умолчанию файлы с описанием возможных привилегированных операций:

1) файл `org.libvirt.unix.policy` — описывает два глобальных разрешения по доступу к виртуальной инфраструктуре:
   - `org.libvirt.unix.monitor` — позволяет выполнять мониторинг виртуальной инфраструктуры;
   - `org.libvirt.unix.manage` — позволяет выполнять управление виртуальной инфраструктурой;
2) файл `org.libvirt.api.policy` — содержит список операций (например, создание ВМ, удаление сети и т.д.), которые становятся доступны пользователям, если проверка по правилу `org.libvirt.unix.policy` пройдена.

После установки пакета `astra-kvm-secure` в каталоге `/var/lib/polkit-1/localauthority/` создаются правила, реализующие разрешения для групп администраторов в соответствии с их ролями.

В каталоге `/var/lib/polkit-1/localauthority/10-vendor.d` создается правило `60-libvirt.pkla`, которое позволяет пользователям групп `libvirt-dev`, `libvirt-adm` и `libvirt` осуществлять доступ к операциям по управлению виртуальной инфраструктурой, со следующим содержимым:

```
[Allow group libvirt-dev, libvirt-adm, libvirt managment permission]
Identity=unix-group:unix-group:libvirt;libvirt-dev;unix-group:libvirt-adm;
Action=org.libvirt.unix.manage
ResultAny=yes
ResultInactive=yes
ResultActive=yes
```

В каталоге `/var/lib/polkit-1/localauthority/15-libvirtd.d` создаются правила вида `<.org.libvirt.api.*.pkla>`, которые определяют разрешения для ролей.

Для редактирования правил используется графическая утилита `fly-admin-policykit-1` (описание утилиты приведено в электронной справке).

5.3.3. Настройка ролевого управления доступом с использованием драйвера доступа parsec

Для включения ролевого управления доступом на основе драйвера доступа parsec:

1) в конфигурационном файле `/etc/libvirt/libvirtd.conf` для параметра `access_drivers` должно быть задано значение `[ "parsec" ]`:

```
access_drivers = [ "parsec" ]
```

2) в конфигурационном файле /etc/libvirt/parsec_roledb.conf для параметра enable должно быть задано значение 1:

```
enable = 1
```

После выполненных действий необходимо перезагрузить сервис libvirtd:

```
sudo systemctl reload libvirtd
sudo systemctl restart libvirtd
```

Конфигурационный файл /etc/libvirt/parsec_roledb.conf содержит перечень всех привилегированных операций при работе с сервером виртуализации libvirt, к которым применяется политика доступа, и разрешения на выполнение данных операций для групп администраторов в соответствии с их ролями, определенные ролевой политикой.

Дополнительно перераспределение полномочий пользователей средства виртуализации в пределах назначенных им ролей осуществляется в конфигурационном файле /etc/libvirt/parsec_roledb.conf. Для редактирования конфигурационного файла возможно использовать инструмент virsh, для этого выполнить команду:

```
virsh -c qemu:///system config --edit-config  /etc/libvirt/parsec_roledb.conf
```

Пример

Запретить пользователю u_adm1, реализующему роль администратора средства виртуализации, удалять ВМ. Для этого необходимо отредактировать разрешения для операции domain_delete следующим образом:

```
domain_delete = ["g:libvirt-admin","-u:u_adm1"]
```

При внесении изменений с помощью virsh изменения применяются после сохранения конфигурационного файла и перезапуск службы libvirtd не требуется.

5.4. Режим «только чтение»

В целях исключения установки вредоносного программного обеспечения и хранения защищаемых данных в виртуальном диске используется режим запуска ВМ «только чтение».

Поддержка функционирования ВМ в режиме «только чтение» (запрета модификации ее образа) осуществляется специальными способами запуска ВМ, при которых основной образ защищается от записи. В зависимости от выбранного режима осуществляется создание физической копии или применяются различные варианты создания снимков образа с последующим их удалением после завершения работы виртуальной машины.

Данный режим работы называется «только чтение» и реализуется тремя способами:

1) временный снимок — способ запуска виртуальной машины, при котором основной образ защищается от записи, а все результаты работы пользователя фиксируются во временном снимке, существующем только в процессе функционирования виртуальной машины;

2) снимок — во время запуска виртуальной машины в заданном каталоге создается снимок, удаляемый после завершения функционирования виртуальной машины;

3) полная копия — во время запуска виртуальной машины в заданном каталоге создается полная копия образа, удаляемая после завершения функционирования виртуальной машины.

Создание полной копии может замедлять процесс запуска виртуальной машины по причине копирования образа большого размера, но данный вариант позволяет использовать возможности по сохранению состояния виртуальной машины для последующего его восстановления.

При использовании снимков в случае любого выключения виртуальной машины вследствие выключения или аппаратного сбоя сервера виртуализации все результаты работы виртуальной машины будут потеряны.

Каталог размещения временных файлов задается в конфигурационном файле `/etc/libvirt/qemu.conf` следующим конфигурационным параметром:

```
run_images_dir = "/var/lib/libvirt/runimages"
```

5.5. Идентификация и аутентификация пользователей в среде виртуализации

При доступе к виртуальным машинам различается доступ к серверу виртуализации libvirt для управления виртуальными машинами и доступ непосредственно к рабочему столу виртуальной машины.

Пользователь может выполнить запуск ВМ или получить доступ к ранее запущенной ВМ после прохождения процедуры идентификации и аутентификации в ОС. Доступ предоставляется в соответствии с установленными правилами разграничения доступа. Запущенная виртуальная машина представляет собой процесс ОС, который функционирует от имени учетной записи пользователя с его мандатными атрибутами безопасности.

Вход в систему осуществляется по идентификатору и паролю. При входе в систему осуществляется идентификация и проверка подлинности субъектов доступа процедурой аутентификации. По умолчанию в системе отключен режим автоматического входа по сохраненным учетным данным или без пароля. Настройка входа в систему осуществляется с использованием утилиты `fly-admin-dm`.

Установка пароля пользователя осуществляется администратором с помощью модуля «Пользователи» в разделе «Пользователи и группы» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_users
```

Ограничение количества неуспешных попыток входа и блокирования учетной записи и сеанса доступа пользователя при превышении числа неуспешных попыток аутентификации в системе устанавливается администратором с помощью инструментов управления политикой безопасности.

В ОС обеспечивается возможность смены установленного администратором ОС пароля пользователя после его первичной аутентификации.

Каждому пользователю в ОС по умолчанию присваивается уникальное имя и идентификатор. При попытке ввода неправильного значения идентификатора или пароля пользователя графической подсистемой ОС выводится сообщение о том, что вход в систему не выполнен, и пользователю повторно предлагается ввести правильный идентификатор и пароль.

Блокировка учетных записей пользователей выполняется автоматически по достижении заданного количества следующих подряд неудачных попыток аутентификации пользователя. Разблокировка учетной записи пользователя выполняется автоматически в соответствии с установленным администратором в политике учетных записей временным интервалом.

Защита пароля пользователя при его вводе обеспечивается за счет отображения вводимых символов условными знаками.

Хранение аутентификационной информации пользователей осуществляется с использованием хеш-функции по ГОСТ Р 34.11-94 и по ГОСТ Р 34.11-2012, что обеспечивает защиту аутентификационной информации.

Описание перечисленных выше процедур приведено в разделе 2.

Для аутентификации в условиях ЕПП и при удаленном доступе к серверу виртуализации используются методы SSH-, SASL- или SSL/TLS-аутентификации в соответствии с РУСБ.10015-01 95 01-1.

5.6. Доверенная загрузка виртуальных машин

Доверенная загрузка виртуальных машин обеспечивается с использованием средств динамического контроля целостности (создание замкнутой программной среды) в режиме контроля целостности файлов при их открытии на основе цифровой подписи в расширенных атрибутах ФС. При выявлении нарушения целостности конфигурации виртуального оборудо-

вания или целостности файлов виртуальной базовой системы ввода-вывода средствами ОС осуществляется блокировка запуска виртуальных машин. Описание применения замкнутой программной среды (ЗПС) приведено в 16.1.

Также в целях блокировки запуска виртуальной машины при выявлении нарушения целостности конфигурации виртуального оборудования данной виртуальной машины или нарушения целостности файлов виртуальной базовой системы ввода-вывода может применяться механизм контроля целостности «отпечаток конфигурации».

5.6.1. Применение режима контроля целостности файлов при их открытии

Блокировка запуска виртуальной машины при выявлении нарушения целостности конфигурации виртуального оборудования или файлов виртуальной базовой системы ввода-вывода осуществляется с использованием средств динамического контроля целостности в режиме запрета открытия поставленных на контроль файлов с неверной цифровой подписью в расширенных атрибутах ФС или без цифровой подписи в соответствии с 16.1.

Для возможности осуществления подписи файлов необходимо создать ключевую пару с помощью утилиты `gpg`, описание настройки модуля `digsig_verif`, создания ключевой пары и подписания файлов приведено в 16.1.

5.6.1.1. Контроль файлов конфигурации виртуального оборудования виртуальных машин

На контроль целостности файлов при их открытии на основе цифровой подписи устанавливаются конфигурационные файлы оборудования виртуальных машин в формате XML, расположенные в каталоге `/etc/libvirt/qemu/`.

Для внедрения цифровой подписи в расширенные атрибуты необходимо подписать на созданном ключе файлы конфигурации с использованием утилиты `bsign`.

Пример

Для внедрения цифровой подписи в файл конфигурации виртуального оборудования виртуальной машины `alse` необходимо выполнить команду:

```
sudo bsign --sign --xattr /etc/libvirt/qemu/alse.xml
```

Проверить наличие подписи можно командой:

```
sudo getfattr -dm- /etc/libvirt/qemu/alse.xml
```

или командой:

```
sudo bsign -w /etc/libvirt/qemu/alse.xml
```

Далее выполнить настройку записей имен файлов оборудования ВМ, на которые распространяется проверка цифровой подписи в расширенных атрибутах ФС. Для этого необходимо в файле `/etc/digsig/xattr_control` задать их список.

Пример

Для добавления в список контролируемого файла `/etc/libvirt/qemu/alse.xml` выполнить:

```
echo '/etc/libvirt/qemu/alse.xml' | sudo tee -a /etc/digsig/xattr_control
```

ВНИМАНИЕ! После внесения изменений в конфигурационный файл `/etc/digsig/xattr_control` необходимо от имени учетной записи администратора через механизм `sudo` выполнить команду:

```
sudo update-initramfs -u -k all
```

Также возможно внести имя контролируемого файла в `/etc/digsig/xattr_control` с использованием модуля «Замкнутая программная среда» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_digsig
```

Необходимо добавить имя файла в поле «Шаблоны имен файлов в расширенных атрибутах» (рис. 2).

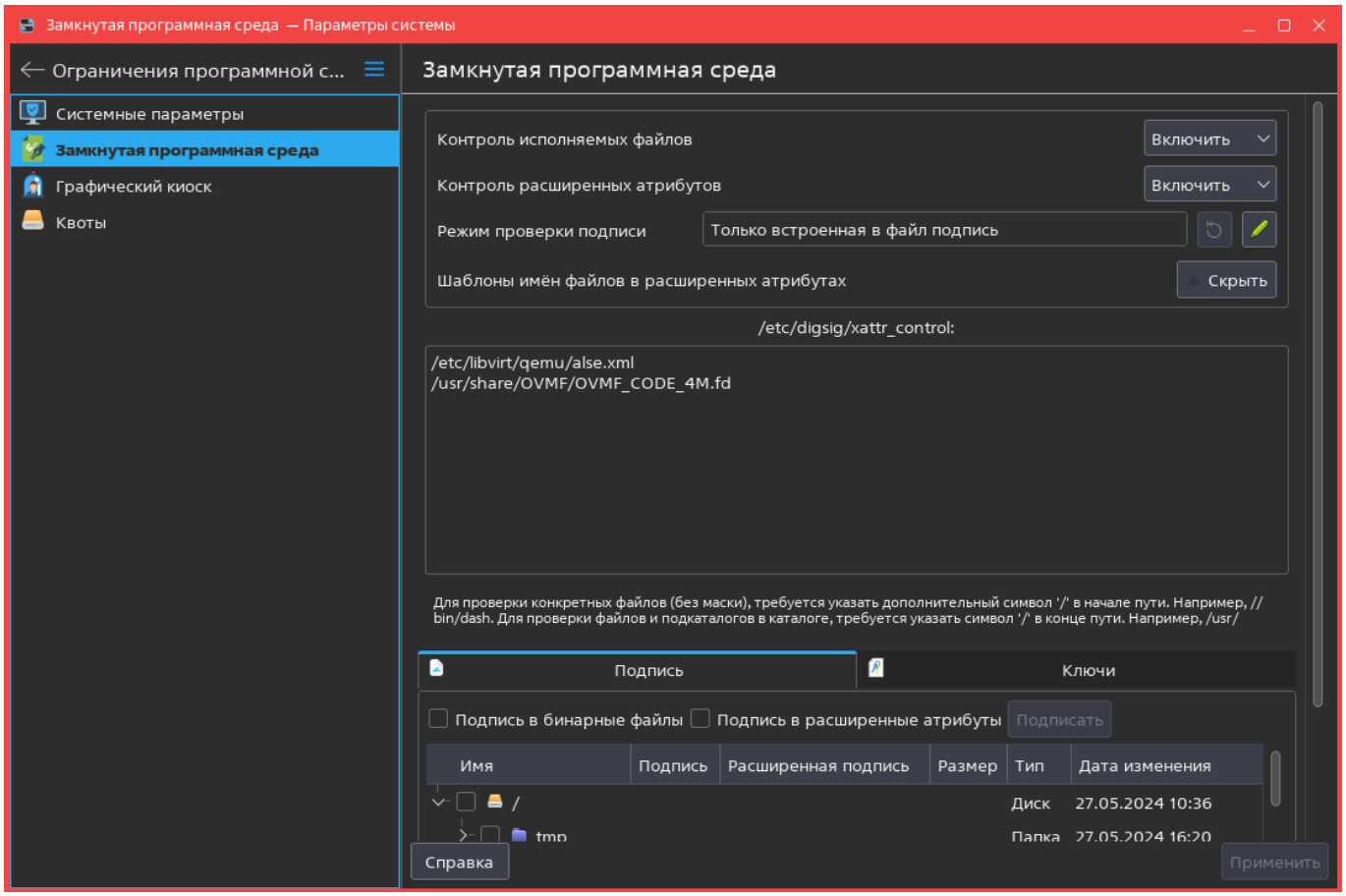

Рис. 2

5.6.1.2. Контроль файлов виртуальной базовой системы ввода-вывода (первичного загрузчика ВМ)

Загрузчик BIOS встроен в QEMU, исполняемый файл `qemu-system-x86_64` которого по умолчанию подписан для контроля запуска в ЗПС (при этом должен быть включен режим запрета запуска исполняемых файлов и разделяемых библиотек с неверной цифровой подписью, а также без цифровой подписи — значение параметра `DIGSIG_ELF_MODE=1`, см. 16.1).

В ОС входит пакет `ovmf` (должен быть установлен пакет `astra-kvm-secure`), который предоставляет поддержку загрузчика UEFI и позволяет использовать в ВМ загрузчик UEFI вместо BIOS. Загрузчик UEFI для ВМ необходимо выбрать на этапе создания ВМ. Затем для контроля запуска файлов в режиме ЗПС требуется подписать файлы загрузчика UEFI в расширенных атрибутах ФС.

Для использования в ВМ загрузчика UEFI необходимо в процессе создания ВМ с использованием `virt-manager` установить флаг «Проверить конфигурацию перед установкой» и нажать **[Готово]** в соответствии с рис. 3.

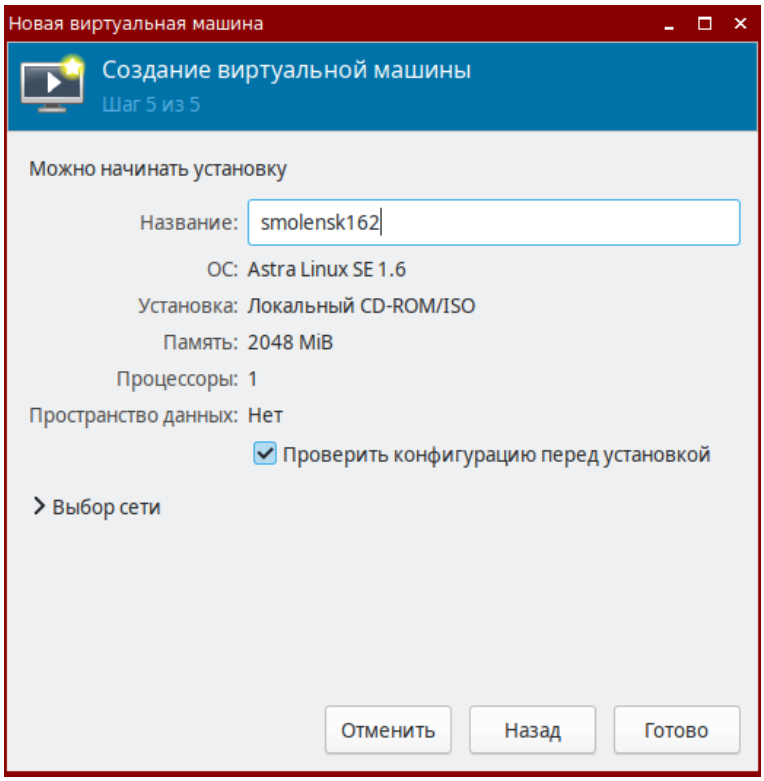

Рис. 3

В открывшемся окне конфигурации ВМ в разделе «Обзор» в секции «Свойства гипервизора» в раскрывающемся списке «Firmware» выбрать необходимую версию загрузчика UEFI, например «UEFI x86-64: /usr/share/OVMF/OVMF_CODE_4M.fd». Затем нажать **[Применить]** и **[Начать установку]**.

Проверить свойства гипервизора возможно в свойствах созданной ВМ в соответствии с рис. 4.

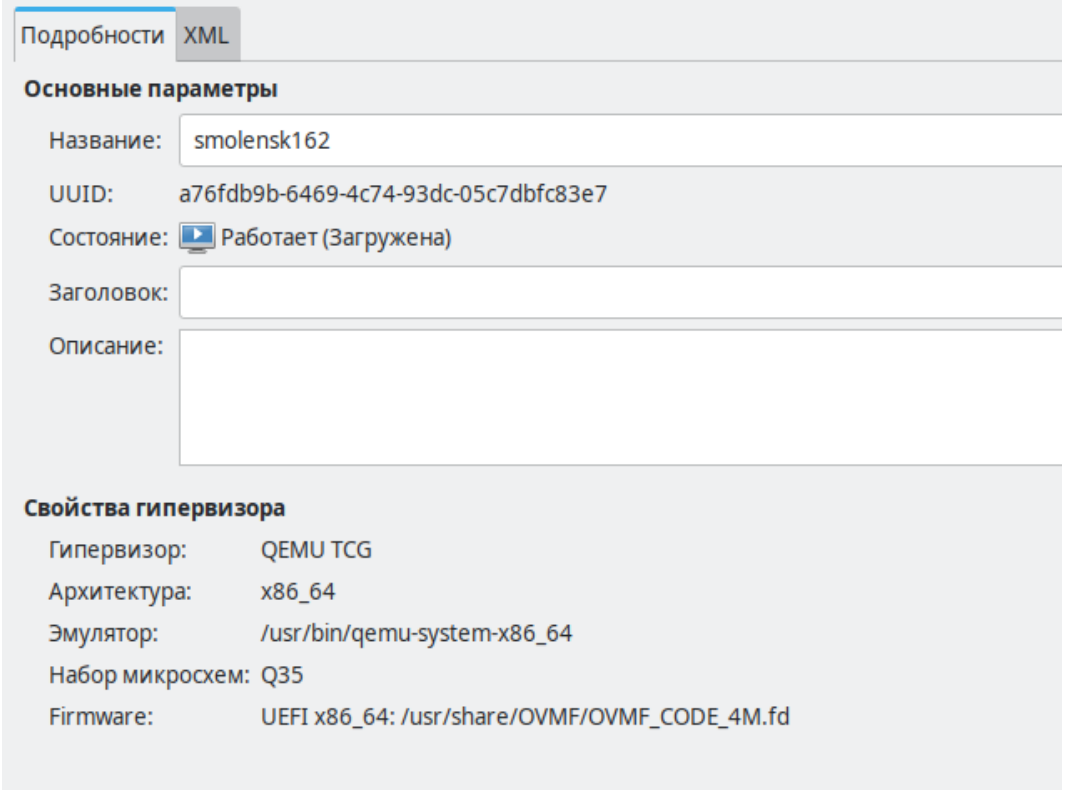

Рис. 4

Расположенные в каталоге `/usr/share/OVMF` файлы с расширением `.fd` должны быть установлены администратором на контроль целостности, для этого внедрить цифровую подпись в расширенные атрибуты ФС.

Подписывать файлы необходимо на созданном вторичном ключе с использованием утилиты `bsign`.

Пример

Для внедрения цифровой подписи в файл `/usr/share/OVMF/OVMF_CODE.fd` необходимо выполнить команду:

```
sudo bsign --sign --xattr /usr/share/OVMF/OVMF_CODE.fd
```

Проверить наличие подписи командой:

```
sudo getfattr -dm-  /usr/share/OVMF/OVMF_CODE.fd
```

Далее выполнить настройку записей имен файлов, на которые распространяется проверка цифровой подписи в расширенных атрибутах ФС. Для этого можно использовать модуль «Замкнутая программная среда» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Необходимо добавить в поле «Шаблоны имен файлов в расширенных атрибутах» путь к контролируемому файлу загрузчика в соответствии с рис. 2.

5.6.2. Применение механизма контроля целостности с использованием алгоритма работы с контрольными суммами («отпечатка конфигурации»)

Применение механизма контроля целостности обеспечивает динамическое создание хеша с использованием алгоритма подсчета контрольных сумм по стандарту ГОСТ Р 34.11-2012 для каждого легитимно загружаемого файла конфигурации и сохранение их в каталоге с высокой меткой целостности `/var/lib/libvirt/hash`.

ВНИМАНИЕ! Для обеспечения защиты базы эталонных контрольных сумм рекомендуется включить мандатный контроль целостности согласно 4.9.

Механизм контролирует следующий перечень конфигураций виртуальной инфраструктуры:

1) конфигурационные файлы ВМ формата XML;
2) конфигурационные файлы других объектов виртуальной инфраструктуры формата XML (`nwfilter`, `storage`, `cpu_map` и др.);
3) конфигурационные файлы средства виртуализации с расширением `*.conf`, расположенные в каталоге `/etc/libvirt/` и содержащие параметры настройки средства виртуализации;
4) файлы виртуальной базовой системы ввода-вывода (первичного загрузчика ВМ).

Управление конфигурациями виртуальной инфраструктуры возможно только в контексте доверенных процессов libvirt, доступ к которым, в соответствии с ролевой политикой доступа, разрешен только для пользователей, обладающих полномочиями роли администратора средства виртуализации (управление конфигурациями средства виртуализации) или роли разработчика виртуальных машин (управление конфигурациями виртуальных машин).

Для ограничения возможностей использования операций изменения конфигураций виртуальной инфраструктуры рекомендуется использовать ролевое управление доступом в соответствии с 5.3.

Для использования механизма должен быть установлен пакет `astra-kvm-secure`. Установка выполняется командой:

```
sudo apt install astra-kvm-secure
```

После установки пакета необходимо включить динамический контроль целостности, задав в конфигурационном файле `/etc/libvirt/libvirtd.conf` для параметра `integrity_control` значение `1`:

```
integrity_control = 1
```

Включать динамический контроль целостности рекомендуется только после всех настроек системы виртуализации.

Для применения настроек следует перезагрузить службу `libvirtd`:

```
sudo systemctl restart libvirtd
```

После включения механизма «отпечаток конфигурации» легитимное изменение конфигурационных файлов средства виртуализации с расширением `*.conf`, расположенных в каталоге `/etc/libvirt/`, возможно с использованием инструмента `virsh` путем выполнения команды:

```
virsh -c qemu:///system config --edit-config <конфигурационный_файл>
```

Пример

Легитимное изменение конфигурационного файла `/etc/libvirt/libvirtd.conf`:

```
virsh -c qemu:///system config --edit-config /etc/libvirt/libvirtd.conf
```

Применение `virsh` для редактирования конфигурационного файла обеспечивает автоматический пересчет контрольной суммы после сохранения.

Пересчет контрольной суммы конфигурационных файлов средства виртуализации libvirt с расширением `*.conf` также возможен с использованием команды:

```
virsh -c qemu:///system config --update-hash <конфигурационный_файл>
```

Пример

Пересчет контрольной суммы конфигурационного файла `/etc/libvirt/libvirtd.conf`:

```
virsh -c qemu:///system config --update-hash /etc/libvirt/libvirtd.conf
```

Легитимное изменение конфигураций объектов виртуальной инфраструктуры (`domain`, `nwfilter`, `storage` и др.) возможно путем корректировки XML-файла как с использованием инструмента `virsh`, выполнив команду:

```
virsh -c qemu:///system  edit <VM>
```

так и с помощью графической утилиты `virt-manager` — открыть свойства ВМ и в разделе «Общие» перейти во вкладку «XML».

Для включения возможности редактирования XML в `virt-manager` необходимо в главном окне программы перейти «Правка — Настройки» и в окне «Настройки» во вкладке «Общие» установить флаг «Включить редактирование XML».

При использовании `virsh` и `virt-manager` для редактирования конфигурации ВМ контрольные суммы файлов конфигурации автоматически пересчитываются после сохранения конфигурации, а изменения вступают в силу после перезагрузки ВМ.

Удаление объектов виртуальной инфраструктуры при использовании контроля целостности также необходимо выполнять с помощью `virsh` и `virt-manager`, обеспечивающих легитимную очистку базы эталонных контрольных сумм.

После включения механизма при каждой загрузке службы `libvirtd` будет выполняться сравнение контрольных сумм загружаемых в память конфигураций с эталонными. Если было осуществлено нелегитимное изменение конфигурации ВМ, вычисленное значение контрольной суммы не совпадет с эталонным и работа с этой ВМ будет заблокирована. При нарушении целостности конфигурации средства виртуализации блокируется доступ к средству виртуализации.

События нарушения целостности конфигурации ВМ и конфигурации средства виртуализации регистрируются подсистемой регистрации событий и записываются в журнал событий — записи «Нарушение целостности» с указанием имени файла и «Нарушение целостности конфигурации СВ» соответственно (см. 6.5).

5.6.3. Применение механизма контроля целостности файлов гостевой операционной системы

Применение механизма контроля целостности обеспечивает выборочную установку на контроль целостности файлов из состава гостевой операционной системы. Контроль осуществляется средством виртуализации путем вычисления контрольных сумм файлов в соответствии с ГОСТ Р 34.11-2012 (с длиной хеш-кода 256 или 512 бит) с последующим сравнением вычисленных значений с эталонными. Для использования механизма в системе должен быть установлен пакет `astra-kvm-secure`.

Подсчет контрольных сумм может осуществляться как в процессе загрузки гостевой операционной системы, так и периодически в процессе работы. Эталонные значения контрольных сумм хранятся в конфигурационных файлах контролируемых ВМ, что позволяет осуществлять контроль и при миграции ВМ. При выявлении нарушения целостности файлов гостевой операционной системы осуществляется блокировка ее запуска и регистрация в журнале подсистемы регистрации событий (см. 6.5) и в журнале `/var/log/syslog` фактов нарушения целостности объектов контроля.

Использование механизма периодического контроля целостности (в процессе работы гостевой операционной системы) возможно только при условии использования ОС в качестве гостевой операционной системы и установки в ней гостевого агента `qemu-guest-agent` из состава ОС, позволяющего считывать значения контрольных сумм объектов контроля. Установка агента выполняется командой:

```
sudo apt install qemu-guest-agent
```

В случае отсутствия в гостевой операционной системе ВМ установленного агента `qemu-guest-agent` периодическая проверка контроля целостности в процессе функционирования ВМ выполняться не будет, контроль целостности файлов может осуществляться только в процессе загрузки ВМ.

Настройка механизма контроля целостности файлов гостевой операционной системы осуществляется в конфигурационном файле `/etc/libvirt/libvirtd.conf`, при этом эти настройки применяются ко всем ВМ. Настройку рекомендуется выполнять с использованием инструмента `virsh`. Для редактирования конфигурационного файла выполнить на хосте гипервизора команду:

```
virsh -c qemu:///system config --edit-config /etc/libvirt/libvirtd.conf
```

В конфигурационном файле `/etc/libvirt/libvirtd.conf` доступны следующие настройки:

1) для включения механизма контроля целостности файлов гостевой операционной системы в процессе ее функционирования необходимо для параметра `file_integrity_check_period_s` задать значение периода проверки (в секундах):

```
file_integrity_check_period_s = 30
```

При этом будет обеспечиваться только регистрация фактов нарушения целостности объектов контроля без блокировки работы ВМ;

2) для принудительного выключения функционирующей ВМ в случае нарушения целостности установленных на контроль файлов необходимо для параметра `file_integrity_check_shutdown_domain` задать значение `1`:

```
file_integrity_check_shutdown_domain = 1
```

При этом будет осуществляться регистрация фактов нарушения целостности объектов контроля с последующим принудительным выключением ВМ, если хотя бы у одного установленного на контроль файла гостевой операционной системы значение контрольной суммы не совпадет с эталонным значением, хранящимся в конфигурационном файле ВМ;

3) для включения механизма контроля целостности файлов гостевой операционной системы в процессе ее загрузки необходимо для параметра `file_integrity_on_startup_VM` задать значение `1`:

```
file_integrity_on_startup_VM = 1
```

При этом будет осуществляться регистрация фактов нарушения целостности объектов контроля и блокировка запуска ВМ, если хотя бы у одного установленного на контроль файла гостевой операционной системы значение контрольной суммы не совпадет с эталонным значением, хранящимся в конфигурационном файле ВМ.

Для применения настроек следует перезапустить службу `libvirtd`:

```
sudo systemctl restart libvirtd
```

Перед установкой на контроль файлов гостевой операционной системы необходимо для каждого файла определить:

1) полный путь к контролируемому файлу в гостевой операционной системе;
2) наименование диска (раздела) хранения контролируемого файла гостевой операционной системы;
3) точку монтирования диска ВМ.

На хосте гипервизора данные значения задаются для файлов, подвергаемых контролю.

Для определения наименования раздела и точки монтирования в операционной системе ВМ возможно использовать инструмент `lsblk` для просмотра таблицы разделов.

Пример

Вывод команды `lsblk`:

```
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sr0     11:0    1 1024M  0 rom
vda    252:0    0   20G  0 disk
--vda1 252:1    0 18,6G  0 part /
--vda2 252:2    0 512M  0 part /boot/efi
--vda3 252:3    0  976M  0 part [SWAP]
vdb    252:16   0   10G  0 disk
```

Файловая система расположена на диске `vda1`, соответствующая точка монтирования указана в столбце `MOUNTPOINT`: раздел `vda1` монтируется в «/».

Для установки на контроль целостности файла гостевой операционной системы требуется на хосте гипервизора выполнить команду:

```
virsh -c qemu:///system file-integrity <domain> --path-add <путь_к_файлу> \
     --mount-point <точка_монтирования> --disk-name <имя_диска>
```

где <domain> — наименование ВМ;
<путь_к_файлу> — полный путь к файлу, подлежащему контролю;
<точка_монтирования> — точка монтирования раздела, в котором хранится файл;
<имя_диска> — наименование диска, на котором хранится файл в гостевой ОС.

Пример

```
virsh -c qemu:///system file-integrity alse --path-add /bin/afick \
     --mount-point / --disk-name vda1
```

Установку на контроль целостности файлов гостевой операционной системы должен выполнять администратор средства виртуализации с нулевой классификационной меткой при выключенной ВМ.

После установки на контроль файлов они будут указаны в виде записей блока `fileintegrity` конфигурационного файла ВМ.

Пример

Блок `fileintegrity` в конфигурационном файле ВМ с добавленным для контроля файлом в гостевой операционной системе, контрольная сумма файла не рассчитана:

```
<fileintegrity check="-1">
<item mount_point="/" disk_name="vda1" alg="gost512" path="/bin/afick">
</fileintegrity>
```

Параметр `check` в блоке `fileintegrity` отображает состояние механизмов контроля целостности файлов гостевой операционной системы и может принимать следующие значения:

1) значение `-1`:
   а) первичное состояние периодического контроля целостности, когда в гостевой операционной системе еще не определено наличие агента `qemu-guest-agent`. После загрузки ВМ и успешно выполненной проверки наличия агента значение `-1` автоматически изменится на `1`;

б) агент qemu-guest-agent не найден, периодический контроль в процессе функционирования ВМ не будет выполняться;

в) отключен периодический контроль целостности в процессе функционирования (file_integrity_check_period_s = 0);

2) значение 1 — агент найден и периодический контроль целостности файлов гостевой операционной системы для данной ВМ выполняется;

3) значение 0 — может быть установлено только администратором для принудительного отключения периодического контроля целостности и контроля целостности при загрузке для конкретной ВМ.

ВНИМАНИЕ! Значения параметра check=-1 или check=1 отображают только состояние периодического контроля целостности файлов в процессе функционирования ВМ и не влияют на контроль целостности при загрузке ВМ. Значение параметра check=o отображает состояние периодического контроля целостности и контроля целостности при загрузке ВМ.

После настройки контроля целостности и постановки на контроль файлов гостевой операционной системы перед первой загрузкой ВМ будет выполнена проверка наличия в гостевой операционной системе файлов, поставленных на контроль. Если проверка пройдена успешно, то рассчитываются и указываются в конфигурационном файле ВМ эталонные значения контрольных сумм контролируемых файлов.

ВНИМАНИЕ! Если после настройки контроля целостности и постановки файлов на контроль первый запуск ВМ будет осуществляться с ненулевым уровнем конфиденциальности, то расчет эталонных значений контрольных сумм контролируемых файлов должен быть выполнен до первого запуска ВМ с помощью команды:

```
virsh -c qemu:///system file-integrity <domain> --update-hash <путь_к_файлу>
```

Пример

Блок fileintegrity в конфигурационном файле ВМ с рассчитанной контрольной суммой файла в гостевой операционной системе:

```
<fileintegrity check="1">
<item mount_point="/" disk_name="vda1" alg="gost512" path="/bin/afick"
hash="67f6...">
</fileintegrity>
```

Примечание. Для item возможно автоматическое добавление параметров disk_num (порядковый номер диска в конфигурации ВМ), part_num (порядковый номер раздела на диске) и alg (длина ключа алгоритма подсчета контрольных сумм). Значения для данных параметров формируются автоматически и необходимы для корректной работы механизма.

При необходимости установить на контроль целостности новые файлы гостевой операционной системы администратору средства виртуализации следует при выключенной ВМ и с нулевой классификационной меткой повторить действия по постановке на контроль файлов с помощью команды:

```
virsh -c qemu:///system file-integrity <domain> --path-add <путь_к_файлу> \
    --mount-point <точка_монтирования> --disk-name <имя_диска>
```

Перерасчет контрольных сумм в случае нарушения контроля целостности объектов контроля выполняется с использованием команды:

```
virsh -c qemu:///system file-integrity <domain> --update-hash <путь_к_файлу>
```

Перерасчет контрольных сумм рекомендуется выполнять при выключенной ВМ.

Выключение контроля целостности файлов гостевой операционной системы выполняется командой:

```
virsh -c qemu:///system file-integrity <domain> --stop
```

Более подробная информация об использовании команды `virsh file-integrity` приведена на странице помощи:

```
virsh help file-integrity
```

5.6.4. Контроль целостности образов ВМ

Механизм контроля целостности образов ВМ позволяет блокировать запуск ВМ из образов, расположенных в файловых хранилищах системы виртуализации, целостность которых была нарушена. Применение механизма возможно только при включенном механизме контроля целостности с использованием алгоритма работы с контрольными суммами (см. 5.6.2).

Примечание. Механизм эффективно работает с небольшими дисками в формате QCOW2, при работе с дисками в формате RAW возможно увеличение затрат времени и ресурсов.

Для включения контроля целостности образов ВМ необходимо в файле `/etc/libvirt/libvirtd.conf` для параметра `integrity_image_control` задать значение `1`:

```
integrity_image_control = 1
```

Для применения настроек требуется перезапустить службу `libvirtd`:

```
sudo systemctl restart libvirtd
```

После включения контроля целостности образов ВМ при первом запуске ВМ рассчитывается контрольная сумма (хеш) образа и сохраняется в каталоге `/var/lib/libvirt/hash`. При выключении ВМ рассчитывается контрольная сумма измененного образа ВМ и перезаписывается. При каждом последующем включении ВМ выполняется проверка контрольной суммы образа. Если вычисленная контрольная сумма не соответствует сохраненной в `/var/lib/libvirt/hash`, то запуск ВМ блокируется. Также выполняется перерасчет контрольной суммы при создании снимков, сохранении и перезагрузке ВМ.

При установке новой ВМ после включения механизма контроля целостности контрольная сумма образа вычисляется сразу при создании образа.

При включенном контроле целостности образов запуск ВМ выполняется дольше, поэтому для более быстрого вычисления рекомендуется использовать алгоритм `xxhash128` вместо `gost512`. Для выбора алгоритма необходимо в `/etc/libvirt/libvirtd.conf` для параметра `hash_type` указать соответствующее значение:

```
hash_type = "xxhash128"
```

5.7. Контроль целостности в среде виртуализации

Для осуществления контроля целостности в среде виртуализации используются следующие механизмы:

1) механизм контроля запуска исполняемых файлов в условиях ЗПС в соответствии с 5.7.1;

2) механизм регламентного контроля целостности Another File Integrity Checker (AFICK) в соответствии с 5.7.2;

3) механизм контроля целостности «отпечаток конфигурации» в соответствии с 5.6.2;

4) механизм контроля целостности исполняемых файлов гостевой операционной системы в соответствии с 5.6.3;

5) механизм контроля целостности образов ВМ в соответствии с 5.6.4;

6) механизм контроля целостности областей памяти ВМ (по запросу из гостевой операционной системы) в соответствии с 5.12.

Контроль целостности осуществляется для следующих типов объектов: конфигурации виртуального оборудования виртуальных машин, конфигураций объектов виртуальной инфраструктуры, исполняемых файлов и параметров настройки средств виртуализации, файлов

виртуальной базовой системы ввода-вывода (первичного загрузчика виртуальной машины), исполняемых файлов гостевой операционной системы виртуальной машины, областей памяти виртуальной машины.

Регистрация событий безопасности, связанных с целостностью средств виртуализации и виртуальных машин, осуществляется подсистемой регистрации событий в соответствии с 6.5.

5.7.1. Применение динамического контроля целостности в режиме ЗПС

Инструменты ЗПС предоставляют возможность внедрения цифровой подписи в исполняемые файлы и в расширенные атрибуты ФС, обеспечивая таким образом контроль целостности как в процессе загрузки средства виртуализации, так и динамически в процессе функционирования.

5.7.1.1. Режим контроля неизменности и подлинности загружаемых исполняемых файлов

Для обеспечения контроля целостности исполняемых файлов средства виртуализации должен быть включен и настроен режим запрета запуска исполняемых файлов и разделяемых библиотек с неверной цифровой подписью или без цифровой подписи в соответствии с 16.1.

Исполняемые файлы средства виртуализации подписаны по умолчанию для контроля запуска в ЗПС.

ВНИМАНИЕ! Для контроля целостности исполняемых файлов гостевой операционной системы рекомендуется в качестве гостевой операционной системы использовать ОС с включенным режимом запрета загрузки на исполнение файлов и разделяемых библиотек с неверной цифровой подписью, а также без цифровой подписи.

5.7.1.2. Режим контроля целостности файлов при их открытии

Для обеспечения динамического контроля целостности применяется режим запрета открытия поставленных на контроль файлов с неверной цифровой подписью или без цифровой подписи в расширенных атрибутах в соответствии с 5.6.1.

Контролю подлежат следующие файлы:

1) конфигурационные файлы libvirt с расширением `.conf`, расположенные в каталоге `/etc/libvirt/`;

2) конфигурационные файлы виртуальных машин в формате XML, расположенные в каталоге `/etc/libvirt/qemu/`;

3) файлы виртуальной базовой системы ввода-вывода с расширением `.fd`, расположенные в каталоге `/usr/share/OVMF/`.

5.7.2. Применение регламентного контроля целостности AFICK

Организация регламентного контроля целостности объектов контроля обеспечивается программным средством AFICK путем вычисления контрольных сумм файлов и соответствующих им атрибутов расширенной подсистемы безопасности PARSEC (мандатных атрибутов и атрибутов расширенной подсистемы протоколирования) с последующим сравнением вычисленных значений с эталонными в соответствии с 10.4.

Возможность проведения периодического контроля осуществляется с использованием системного планировщика заданий `cron`. Системный планировщик заданий `cron` также позволяет выполнять проверку целостности объектов контроля в процессе загрузки ОС.

Для контроля целостности необходимо в конфигурационном файле `/etc/afick.conf` задать список файлов и каталогов, которые будет контролировать AFICK.

Контролю подлежат следующие файлы:

1) конфигурационные файлы libvirt с расширением `.conf`, расположенные в каталоге `/etc/libvirt/`;
2) конфигурационные файлы виртуальных машин в формате XML, расположенные в каталоге `/etc/libvirt/qemu/`;
3) файлы виртуальной базовой системы ввода-вывода с расширением `.fd`, расположенные в каталоге `/usr/share/OVMF/`.

5.8. Регистрация событий безопасности в среде виртуализации

5.8.1. Настройка регистрации событий безопасности, связанных с функционированием средства виртуализации

В среде виртуализации обеспечивается регистрация событий безопасности, связанных с функционированием средств виртуализации, и оповещение администратора безопасности средства виртуализации о событиях безопасности. Состав регистрируемой информации соответствует ГОСТ Р 59548-2022.

Регистрация событий безопасности, настройка реагирования системы на события и информирование администратора осуществляется подсистемой регистрации событий из состава ОС. Описание подсистемы регистрации событий и журнала событий приведено в 6.5.

Также регистрацию событий осуществляет служба `libvirtd`. Запись событий осуществляется в журнал подсистемы регистрации событий и в системный журнал `/var/log/syslog`.

Для каждой записи в журнале регистрируются номер (уникальный идентификатор), дата, время и тип события безопасности.

Просмотр системного журнала осуществляется администратором с использованием инструментов командной строки (`ausearch`, `aureport`, `aulast` и `auvirt`) и графической утилиты `ksystemlog`.

Инструмент `auvirt` используется для поиска в журнале аудита записей, созданных libvirt, чтобы вывести список сеансов ВМ. Также он выполняет поиск событий, связанных с гостевыми системами (остановка хостовой системы, отказ в доступе), и событий, связанных с QEMU-процессами.

5.8.2. Механизм централизованного сбора журналов с удаленных хостов виртуализации

Для решения задач централизованного сбора журналов с удаленных хостов виртуализации возможно применение встроенных механизмов подсистемы регистрации событий (см. 6.5).

Служба `syslog-ng` из состава подсистемы регистрации событий предоставляет возможность централизованного сбора событий безопасности, связанных с функционированием средства виртуализации, из журналов событий безопасности.

При установке пакета `astra-kvm-secure` создаются конфигурационные файлы для удаленного сбора данных о событиях безопасности.

Для настройки отправки событий с удаленного хоста необходимо в конфигурационном файле `/etc/syslog-ng/conf.d/mod-astra-libvirtd-remote.conf` в соответствующей строке установить значение IP-адреса сервера сбора журналов (вместо указанного по умолчанию `"127.0.0.1"`) и раскомментировать параметр `destination(libvirtd_remote_dst)`:

```
destination libvirtd_remote_dst {
network(
"127.0.0.1" #ip for remote host
port(2222) # for example
transport(tcp) # or udp
);
};

log {
source(s_src);
filter(libvirtd_syslog_filter);
destination(libvirtd_remote_dst);
};
```

Для осуществления возможности приема пакетов с удаленных хостов необходимо в конфигурационном файле сервера сбора журналов

/etc/syslog-ng/conf.d/mod-astra-libvirtd-remote-server.conf раскомментировать параметр destination(astra_json_dst):

```
source libvirtd_remote_src {
network(
port(2222) # for example
transport(tcp) # or udp
);
};

log {
source(libvirtd_remote_src);
filter(libvirtd_syslog_filter);
parser(libvirtd_syslog_parser);
parser(astra_libvirtd_parser);
destination(astra_json_dst);
};
```

5.9. Резервное копирование в среде виртуализации

Резервное копирование образов виртуальных машин и конфигурации виртуального оборудования виртуальных машин, а также параметров настройки средств виртуализации и сведений о событиях безопасности реализуется с использованием встроенных в средства виртуализации ОС механизмов резервного копирования (инструментов командной строки virsh backup-begin, virsh dumpxml и virsh snapshot-create), а также встроенных в ОС средств резервного копирования.

ПО резервного копирования и восстановления из состава ОС включает инструменты командной строки и распределенные системы управления хранилищами данных:

1) комплекс программ Bacula;
2) инструмент копирования rsync;
3) инструменты архивирования и копирования tar, cpio, gzip, cp.

Описание указанных инструментов приведено в РУСБ.10015-01 95 01-1.

5.9.1. Резервное копирование образов виртуальных машин

Полное резервное копирование текущего состояния образа виртуальной машины выполняется от имени администратора средства виртуализации.

Для резервного копирования используется следующая команда, при этом ВМ должна быть включена:

```
virsh -c qemu:///system backup-begin <domain>
```

где <domain> — наименование ВМ. Допускается указывать идентификатор ВМ, присвоенный средой виртуализации (ID), или всемирно уникальный идентификатор ВМ (UUID).

После выполнения резервного копирования копии образов сохраняются в каталог /var/lib/libvirt/images/.

Пример

Для выполнения резервного копирования образа ВМ alse необходимо от имени администратора средства виртуализации:

1) вывести перечень имеющихся ВМ:

```
virsh -c qemu:///system list --all
```

Вывод команды:

```
 ID Имя Состояние
----------------------------
 - alse выключен
```

2) запустить ВМ alse:

```
virsh -c qemu:///system start alse
```

3) запустить резервное копирование образа ВМ alse:

```
virsh -c qemu:///system backup-begin alse
```

4) для проверки статуса задания, запущенного в отношении ВМ alse, выполнить команду:

```
virsh -c qemu:///system domjobinfo alse
```

Для отображения информации о завершенных заданиях, запущенных в отношении ВМ alse, выполнить команду:

```
virsh -c qemu:///system domjobinfo alse --completed
```

Вывод команды:

```
Тип задания: Завершено
```

```
Операция: Резервное копирование
Прошло: 63740 ms
Обработано файлов: 19,531 GiB
Осталось файлов: 0,000 B
Всего файлов: 19,531 GiB
```

5) проверить наличие созданной резервной копии можно командой:

```
sudo ls -lash /var/lib/libvirt/images/
```

Более подробная информация об использовании команды резервного копирования образов ВМ приведена на странице помощи:

```
virsh help backup-begin
```

5.9.2. Резервное копирование конфигурации виртуального оборудования виртуальных машин

Выполнение резервного копирования конфигурации виртуального оборудования созданных виртуальных машин (существующего XML-файла ВМ) осуществляется от имени администратора средства виртуализации командой:

```
virsh -c qemu:///system dumpxml <domain> > <имя_копии>
```

где <domain> — наименование ВМ, допускается указывать идентификатор ВМ, присвоенный средой виртуализации (ID), или всемирно уникальный идентификатор ВМ (UUID);
<имя_копии> — имя файла или полный путь к файлу, в который необходимо сохранить копию конфигурации виртуального оборудования. Если не указан полный путь к файлу, то он будет сохранен в текущем каталоге (в котором выполнялась команда).

```
virsh dumpxml <domain-id|domain-name|domain-uuid> <имя_копии>
```

Пример

Для выполнения резервного копирования конфигурации ВМ `alse` необходимо от имени администратора средства виртуализации выполнить команду:

```
virsh -c qemu:///system dumpxml alse > alse_backup.xml
```

Более подробная информация об использовании команды резервного копирования конфигурации ВМ приведена на странице помощи:

```
virsh help dumpxml
```

5.9.3. Резервное копирование параметров настройки средства виртуализации

Выполнение резервного копирования параметров настройки средства виртуализации осуществляется с использованием инструментов архивирования и копирования из состава ОС: `tar`, `cpio`, `gzip`, `cp`.

Пример

Для выполнения резервного копирования конфигурационных файлов в каталоге `/etc/libvirt/` выполнить команды:

```
cd /etc/libvirt
tar --xattrs --acls -czv conf_backup.tar.gz /etc/libvirt/libvirtd.conf \
    /etc/libvirt/libvirt.conf /etc/libvirt/libvirt-admin.conf
```

5.9.4. Создание снимков текущего состояния машины

Снимок состояния ВМ содержит снимки состояния диска и оперативной памяти ВМ (ВМ должна быть включена). Создание снимка состояния ВМ выполняется от имени администратора средства виртуализации с помощью команды:

```
virsh -c qemu:///system snapshot-create <domain> [--disk-only]
```

где `<domain>` — наименование ВМ, допускается указывать идентификатор ВМ, присвоенный средой виртуализации (ID), или всемирно уникальный идентификатор ВМ (UUID);
`[–disk-only]` — необязательный параметр, указывает на необходимость сохранения в снимке состояния ВМ только снимка состояния диска, без снимка оперативной памяти ВМ.

Более подробная информация об использовании команды создания снимков приведена на странице помощи:

```
virsh help snapshot-create
```

Созданные снимки ВМ сохраняются в каталог `/var/lib/libvirt/qemu/snapshot/<ВМ>/`.

Для просмотра существующих снимков ВМ выполнить команду:

```
virsh -c qemu:///system snapshot-list <domain>
```

Наименование снимка ВМ задается меткой времени создания снимка в формате Unix.

Если необходимо задать собственное наименование снимка ВМ, то для создания снимка можно воспользоваться командой:

```
virsh -c qemu:///system snapshot-create-as <domain> [наименование_снимка] \
    [описание_снимка] [--disk-only]
```

Более подробная информация об использовании команды приведена на странице помощи:

```
virsh help snapshot-create-as
```

Для восстановления ВМ из снимка выполнить команду:

```
virsh -c qemu:///system snapshot-revert <domain> [наименование_снимка]
```

Более подробная информация об использовании команды восстановления из снимков приведена на странице помощи:

```
virsh help snapshot-revert
```

Для управления снимками ВМ в утилите `virt-manager` необходимо:

1) в главном окне утилиты выбрать ВМ;
2) нажать **[Открыть]**;
3) в открывшемся окне нажать **[Управление снимками]**.

5.9.5. Резервное копирование и полная очистка журналов

Резервное копирование сведений о событиях безопасности реализуется с использованием встроенных в ОС средств ротации и архивирования журналов `logrotate`. Настройка ротации возможна с использованием утилиты `fly-admin-events` («Настройка регистрации системных событий»), описание утилиты приведено в электронной справке.

Запустить ротацию журнала возможно вручную, без ожидания установленного в cron правила, для этого выполнить команду:

```
sudo logrotate /etc/logrotate.d/syslog-ng-mod-astra
```

5.9.6. Экспорт и импорт виртуальных машин

В ОС реализован экспорт имеющегося экземпляра ВМ с использованием библиотек сервера виртуализации libvirt. В результате экспорта будет сформирован архив, содержащий копии образов дисков, снимков состояний и файла конфигурации виртуального оборудования заданной ВМ. Полученный архив может быть использован для формирования новой (импорта) ВМ с идентичной конфигурацией виртуального оборудования, имя импортируемой ВМ будет соответствовать имени архива. При этом значения параметров, которые должны быть уникальными, будут автоматически переопределены для исключения коллизий.

Экспортируемая ВМ должна быть выключена.

Экспорт ВМ выполняется от имени администратора средства виртуализации с использованием инструмента командной строки virt-export:

```
virt-export --connect qemu:///system --original <имя_ВМ> --name <имя_архива> \
    --path <каталог>
```

где <имя_ВМ> — наименование экспортируемой ВМ;
<имя_архива> — наименование копии экспортируемой ВМ;
<каталог> — каталог сохранения архива с копией экспортируемой ВМ.

Более подробная информация об использовании инструмента командной строки virt-export приведена на справочной странице:

```
man virt-export
```

Для экспорта имеющегося экземпляра ВМ с помощью графической утилиты virt-manager необходимо в главном окне утилиты открыть контекстное меню ВМ и выбрать «Экспортировать». В открывшемся окне в поле «Укажите путь к пространству хранения» задать каталог сохранения архива с копией экспортируемой ВМ и нажать **[Экспортировать]**.

Для импорта экземпляра ВМ можно воспользоваться инструментом командной строки virt-import:

```
virt-import --connect qemu:///system --original <имя_архива> --pool <пул>
```

где `<имя_архива>` — файл архива копии ВМ, которую необходимо импортировать. Если файл архива не размещен в текущем каталоге, то необходимо указать путь к нему;
`<пул>` — наименование пула, в котором необходимо разместить файлы образов дисков новой ВМ.

Более подробная информация об использовании инструмента командной строки `virt-import` приведена на справочной странице странице:

```
man virt-import
```

Для импорта экземпляра ВМ с помощью графической утилиты `virt-manager` необходимо в главном окне утилиты открыть контекстное меню подключения «QEMU/KVM» и выбрать «Импортировать». В открывшемся окне в поле «Укажите путь к пространству хранения» указать путь к архиву копии ВМ, которую необходимо импортировать, и нажать **[Импортировать]**.

5.10. Управление потоками информации в среде виртуализации

Управление потоками информации между виртуальными машинами и информационными (автоматизированными) системами на канальном и сетевом уровнях реализуется штатными средствами ОС, реализующими технологию виртуальных локальных сетей (VLAN) стандарта IEEE 802.1q.

Для управления потоками информации в виртуальной инфраструктуре применяются следующие механизмы, обеспечивающие сетевую фильтрацию:

1) драйвер виртуальных сетей libvirt — предоставляет изолированное мостовое устройство (без подключения физических сетевых карт), к которому подключены гостевые TAP-устройства. Для ВМ разрешается обмен сетевым трафиком между собой и с хостом. Для виртуальной сети возможны три конфигурации:

а) «Изолированный» (isolated) — весь внешний трафик полностью блокируется;

б) NAT — исходящий трафик во внешнюю сеть разрешен, но замаскирован;

в) «Маршрутизация» (forward) — исходящий трафик в локальную сеть разрешен;

2) драйвер сетевых фильтров libvirt — обеспечивает полностью настраиваемую сетевую фильтрацию трафика на гостевых сетевых картах с использованием сетевых фильтров `nwfilter`. Наборы правил для управления трафиком определяются на уровне хоста. Затем наборы правил связываются с определенными гостевыми сетевыми картами;

3) изоляция сетей с помощью VLAN — программный многоуровневый коммутатор Open vSwitch (OVS) для виртуальных сетей обеспечивает изоляцию сети с помощью VLAN путем тегирования портов, а также фильтрацию сетевого трафика. Описание настройки и работы OVS приведено в РУСБ.10015-01 95 01-1.

5.10.1. Управление сетевыми фильтрами nwfilter

Контроль взаимодействия виртуальных машин между собой реализуется с использованием сетевых фильтров `nwfilter`. Сетевые фильтры предоставляют возможность настраивать и применять правила фильтрации сетевого трафика на виртуальных машинах, а также управлять параметрами входящего и исходящего сетевого трафика. Поскольку правила фильтрации нельзя обойти внутри виртуальной машины, это делает их обязательными при использовании виртуальной машины.

Управление фильтрами выполняется как автономными объектами верхнего уровня и осуществляется администратором средства виртуализации с использованием интерфейсов управления средствами виртуализации libvirt. В сетевых фильтрах могут быть настроены правила фильтрации сетевого трафика ВМ отдельно для каждого сетевого интерфейса через конфигурационный XML-файл виртуальных машин. Правила применяются на хосте при запуске виртуальной машины и могут быть изменены во время работы виртуальной машины (путем изменения XML-описания сетевого фильтра).

Несколько виртуальных машин могут использовать один и тот же сетевой фильтр. При изменении такого фильтра обновляются правила фильтрации сетевого трафика всех запущенных виртуальных машин, в которых применяется данный сетевой фильтр.

Также возможно настраивать правила фильтрации сетевого трафика на отдельных сетевых интерфейсах, настроенных для определенных типов сетевых конфигураций виртуальных машин (network, ethernet в режиме моста, bridge).

Сетевые фильтры задаются в формате XML и располагаются в каталоге `/etc/libvirt/nwfilter/`.

Пример

```
<filter name='no-spamming' chain='XXXX'>
<uuid>d217f2d7-5a04-0e01-8b98-ec2743436b74</uuid>
<rule ...>
....
</rule>
<filterref filter='XXXX'/>
</filter>
```

Каждый фильтр имеет имя и UUID, которые служат уникальными идентификаторами. Фильтр может содержать `<rule>`, которые используются для фактического определения сетевых элементов управления. Элемент `<rule>` имеет параметры `action` (действие), `direction` (направление) и необязательный `priority` (приоритет).

Пример

```
<rule action='drop' direction='out' priority='500'>
```

В правилах используются элементы сопоставления с конкретным протоколом. Поддерживаемые протоколы: `mac`, `arp`, `rarp`, `ip`, `ipv6`, `tcp/ip`, `icmp/ip`, `igmp/ip`, `udp/ip`, `udplite/ip`, `esp/ip`, `ah/ip`, `sctp/ip`, `tcp/ipv6`, `icmp/ipv6`, `igmp/ipv6`, `udp/ipv6`, `udplite/ipv6`, `esp/ipv6`, `ah/ipv6`, `sctp/ipv6`. Каждый протокол определяет, что допустимо указывать внутри элемента `<rule>`. Запись имеет вид:

```
<protocol match='yes|no' attribute1='value1' attribute2='value2'/>
```

Пример

Запись для протокола TCP с портами 0-1023:

```
<tcp match='yes' srcportstart='0' srcportend='1023'/>
```

Набор правил по умолчанию можно посмотреть, выполнив команду:

```
virsh -c qemu:///system nwfilter-list
```

Перечень фильтров, автоматически устанавливаемых вместе с libvirt, приведен в таблице 30.

Таблица 30

Фильтр	Описание
`no-arp-spoofing`	Запретить виртуальной машине подделку ARP-трафика. Фильтр разрешает только сообщения запросов и ответов ARP и обеспечивает, чтобы эти пакеты содержали MAC-адреса и IP-адреса виртуальной машины
`allow-arp`	Разрешить трафик ARP в обоих направлениях
`allow-ipv4`	Разрешить трафик IPv4 в обоих направлениях
`allow-ipv6`	Разрешить трафик IPv6 в обоих направлениях
`allow-incoming-ipv4`	Разрешить входящий трафик IPv4
`allow-incoming-ipv6`	Разрешить входящий трафик IPv6
`allow-dhcp`	Разрешить виртуальной машине запрашивать IP-адрес через DHCP (с любого DHCP-сервера)
`allow-dhcpv6`	Аналогично `allow-dhcp`, но для DHCPv6

Окончание таблицы 30

Фильтр	Описание
`allow-dhcp-server`	Разрешить виртуальной машине запрашивать IP-адрес с указанного DHCP-сервера. Десятичный IP-адрес DHCP-сервера с точками должен быть указан в ссылке на этот фильтр. Имя переменной должно быть `DHCPSERVER`
`allow-dhcpv6-server`	Аналогично `allow-dhcp-server`, но для DHCPv6
`no-ip-spoofing`	Запретить виртуальной машине отправлять пакеты IPv4 с исходным IP-адресом, отличным от адреса в пакете
`no-ipv6-spoofing`	Аналогичен `no-ip-spoofing`, но для IPv6
`no-ip-multicast`	Запретить виртуальной машине отправлять многоадресные IP-пакеты
`no-ipv6-multicast`	Аналогичен `no-ip-multicast`, но для IPv6
`clean-traffic`	Запретить виртуальной машине подделку MAC, IP и ARP. Данный фильтр ссылается на остальные фильтры
`clean-traffic-gateway`	Запретить трафик между ВМ на основе их MAC-адресов. Данный фильтр ссылается на другие фильтры
`no-arp-ip-spoofing`	Запретить ВМ подмену ARP-трафика. Фильтр разрешает отправку сообщений запросов и ответов ARP только в том случае, если пакеты содержат IP-адрес ВМ. Данный фильтр используется фильтром `no-arp-spoofing`
`no-arp-mac-spoofing`	Запретить ВМ подмену ARP-трафика. Фильтр разрешает отправку сообщений запросов и ответов ARP только в том случае, если пакеты содержат MAC-адрес ВМ. Данный фильтр используется фильтром `no-arp-spoofing`
`no-mac-broadcast`	Запретить ВМ отправлять широковещательные кадры (MAC-адрес назначения `ff:ff:ff:ff:ff:ff`)
`no-mac-spoofing`	Запретить подмену MAC-адреса ВМ
`no-other-l2-traffic`	Запретить весь трафик канального уровня (L2), кроме трафика, для которого указаны правила (т.е. запретить весь трафик, кроме ARP и IPv4/v6)
`no-other-rarp-traffic`	Позволяют ВМ определить собственный IP-адрес с помощью протокола RARP, но блокируют весь остальной трафик RARP
`qemu-announce-self-rarp`	
`qemu-announce-self`	

Большинство фильтров по умолчанию являются составными (используют другие фильтры по умолчанию). При этом фильтр `clean-traffic` объединяет все фильтры в один фильтр, который затем можно связать с сетевой картой ВМ. Такое применение позволяет предотвратить самые распространенные атаки: подделка MAC, IP и ARP.

Посмотреть конфигурацию фильтра можно, выполнив команду:

```
virsh -c qemu:///system nwfilter-dumpxml <filter_name>|<UUID>
```

Для обновления фильтров рекомендуется использовать инструмент `virsh nwfilter-define`, что гарантирует корректное применение правил `iptables/ebtables` для ВМ.

Для управления сетевыми фильтрами используется набор инструментов `virsh`:

1) `virsh nwfilter-define` — позволяет определить или обновить сетевой фильтр из XML-файла;

2) `virsh nwfilter-undefine` — позволяет отменить определение сетевого фильтра;

3) `virsh nwfilter-dumpxml` — позволяет вывести информацию о сетевом фильтре в формате XML;

4) `virsh nwfilter-list` — выводит список сетевых фильтров;

5) `virsh nwfilter-edit` — позволяет изменить конфигурацию XML для сетевого фильтра.

Для применения правила фильтрации требуется в XML-файл ВМ добавить ссылку на фильтр.

Пример

Связать фильтр `clean-traffic` с ВМ. Для этого необходимо отредактировать XML-файл ВМ, включив в `<interface>` ссылку на фильтр `<filterref>`, а также указав IP-адрес, который разрешено использовать ВМ:

```
<interface type='bridge'>
<mac address='52:54:00:56:44:32'/>
<source bridge='br1'/>
<ip address='10.33.8.131'/>
<target dev='vnet0'/>
<model type='virtio'/>
<filterref filter='clean-traffic'/>
</interface>
```

Если IP-адрес не указан, драйвер сетевого фильтра активирует `learning mode` (режим обучения). Для этого используется модуль `libpcap`, который отслеживает сетевой трафик, отправляемый ВМ, и пытается определить первый используемый им IP-адрес. В результате будет заблокирован трафик на этот адрес.

Драйвер сетевого фильтра использует комбинацию `ebtables`, `iptables` и `ip6tables` в зависимости от того, на какие протоколы ссылается фильтр. Правила фильтрации `clean-traffic` требуют использования только `ebtables`. При указании протоколов `tcp`, `udp` и др. (например, если добавить новый фильтр, блокирующий порт 25, для предотвращения спама по электронной почте) будет также использоваться `iptables`.

Пример

Фильтр сетевого трафика, предотвращающий подмену IP-адреса со стороны ВМ:

```
<filter name='no-ip-spoofing' chain='ipv4-ip' priority='-710'>
<uuid>64f897b5-4ca1-46ee-b459-68fe2df325c1</uuid>
<rule action='return' direction='out' priority='100'>
<ip srcipaddr='0.0.0.0' protocol='udp'/>
</rule>
<rule action='return' direction='out' priority='500'>
<ip srcipaddr='$IP'/>
</rule>
<rule action='drop' direction='out' priority='1000'/>
</filter>
```

Данный фильтр реализует отбрасывание трафика, если IP-адрес (указанный через значение переменной IP) в исходящем IP-пакете не соответствует ожидаемому. Исключением являются исходящие UDP-пакеты с IP-адресом 0.0.0.0 (пакет с начальным запросом к DHCP-серверу). Пакеты, соответствующие правилам, будут возвращены в исходный фильтр.

5.10.2. Реализация собственных правил

На хосте вся фильтрация трафика, осуществляемая подсистемой сетевых фильтров libvirt, сначала проходит через поддержку фильтрации, реализованную `ebtables`, и только затем через фильтры `iptables` или `ip6tables`. Если в дереве фильтров есть правила с протоколами `mac`, `stp`, `vlan arp`, `rarp`, `ipv4` или `ipv6`, то правила `ebtables` будут созданы автоматически.

Роль атрибута цепочки в XML-файле сетевого фильтра заключается в том, что внутри создается определяемая пользователем таблица `ebtables`, которая затем, например если указана цепочка ARP, получает весь трафик ARP, исходящий от ВМ или направляющийся к ней. Далее в корневой цепочке интерфейса генерируется правило, которое направляет весь IPv4-трафик в заданную пользователем цепочку. Следовательно, все правила трафика ARP затем должны быть помещены в фильтры, определяющие эту цепочку. Этот тип ветвления в пользовательские таблицы поддерживается только при фильтрации на уровне `ebtables`.

Можно создавать несколько цепочек для одного и того же протокола. Для этого имя цепочки должно иметь префикс одного из ранее перечисленных протоколов. Чтобы создать дополнительную цепочку для обработки ARP-трафика, можно указать цепочку с именем `arp-test`.

Например, можно фильтровать UDP-трафик по исходным и конечным портам, используя фильтр IP-протокола и указывая атрибуты для протокола, исходного и конечного IP-адресов

и портов UDP-пакетов, которые должны быть приняты. Это позволяет фильтровать UDP-трафик с помощью `ebtables`. Однако после того, как пакет IP или IPv6 (например, UDP) прошел уровень `ebtables` и если в дереве фильтров есть хотя бы одно правило, которое создает экземпляры правил `iptables` или `ip6tables`, необходимо наличие правила, разрешающего прохождение пакета UDP. Этого можно добиться с помощью правила, содержащего соответствующий узел фильтрации трафика `udp` или `udp-ipv6`.

Пример

Для ВМ `test` с интерфейсом `eth0` создать фильтр `test-eth0`, отвечающий следующим требованиям:

1) предотвращает подделку MAC, IP и ARP;
2) открывает на ВМ только порты 22 и 80 протокола TCP;
3) позволяет ВМ отправлять `ping` с интерфейса, но не позволяет получать `ping` на интерфейсе;
4) позволяет ВМ выполнять поиск DNS (UDP к порту 53).

Требование по предотвращению подделки выполняется существующим фильтром `clean-traffic`, поэтому возможно ссылаться на фильтр libvirt из создаваемого пользовательского фильтра.

Чтобы включить трафик для TCP-портов 22 и 80, необходимо добавить два правила для включения этого типа трафика. Чтобы разрешить виртуальной машине отправлять `ping`, необходимо добавить правило для ICMP-трафика. Возможно разрешить инициировать общий ICMP-трафик с ВМ, а не только эхо-запросы ICMP и ответные сообщения. Чтобы в дальнейшем запретить остальной исходящий и входящий трафик для ВМ, потребуется добавить правило, которое отбрасывает весь другой трафик.

Для создания фильтра необходимо:

1) создать файл фильтра:

```
touch test-eth0.xml
```

Файл будет создан в каталоге выполнения команды;

2) добавить в созданный файл XML-код:

```
<filter name='test-eth0'>
<!-- reference the clean traffic filter to prevent
MAC, IP and ARP spoofing. By not providing
and IP address parameter, libvirt will detect the
IP address the VM is using. -->
<filterref filter='clean-traffic'/>
<!-- enable TCP ports 22 (ssh) and 80 (http) to be reachable -->
<rule action='accept' direction='in'>
```

```
<tcp dstportstart='22'/>
</rule>
<rule action='accept' direction='in'>
<tcp dstportstart='80'/>
</rule>
<!-- enable general ICMP traffic to be initiated by the VM;
this includes ping traffic -->
<rule action='accept' direction='out'>
<icmp/>
</rule>
<!-- enable outgoing DNS lookups using UDP -->
<rule action='accept' direction='out'>
<udp dstportstart='53'/>
</rule>
<!-- drop all other traffic -->
<rule action='drop' direction='inout'>
<all/>
</rule>
</filter>
```

Правила в добавляемом XML-коде не содержат IP-адрес ВМ в качестве исходного или конечного адреса, однако фильтрация трафика будет работать корректно, т.к. внутренняя интерпретация правил выполняется для каждого интерфейса, и правила интерпретируются на основе информации о том, какой интерфейс (TAP) отправил или получит пакет, а не на том, какой IP-адрес источника или получателя может быть;

3) загрузить (применить) фильтр:

```
virsh -c qemu:///system nwfilter-define test-eth0.xml
```

Файл с фильтром будет скопирован в каталог `/etc/libvirt/nwfilter/`;

4) в конфигурации ВМ в секции описания сетевого интерфейса в качестве значения параметр `filterref filter` указать фильтр:

```
<interface type='bridge'>
<source bridge='mybridge'/>
<filterref filter='test-eth0'/>
</interface>
```

5) включить ВМ и проконтролировать, что правило применилось, для этого выполнить команду:

```
virsh -c qemu:///system nwfilter-binding-list
```

Результат выполнения команды:

```
Устройство порта   Фильтр
```

```
-------------------------------
vnet195            test-eth0
```

Чтобы более строго контролировать трафик ICMP и гарантировать, что с ВМ могут отправляться только эхо-запросы ICMP, а ВМ может получать только эхо-ответы ICMP, приведенное в пункте перечисления 2) правило ICMP можно заменить следующими двумя правилами:

```
<!-- enable outgoing ICMP echo requests-->
<rule action='accept' direction='out'>
<icmp type='8'/>
</rule>
<!-- enable incoming ICMP echo replies-->
<rule action='accept' direction='in'>
<icmp type='0'/>
</rule>
```

5.11. Защита памяти в среде виртуализации

Для очистки остаточной информации в памяти средства вычислительной техники используются механизмы, которые обеспечивают очищение неиспользуемых блоков ФС непосредственно при их освобождении (распределении) и очищение активных разделов страничного обмена. Описание механизмов приведено в 9.1.

Управление механизмом очистки неиспользуемых блоков ФС непосредственно при их освобождении осуществляется с использованием инструмента `astra-secdel-control`. Управление механизмом очистки активных разделов страничного обмена осуществляется с использованием инструмента `astra-swapwiper-control`.

Защита задач ядра и процессов пользователей при доступе к страницам оперативной памяти обеспечивается архитектурой и параметрами ядра ОС (см. 9.2).

Для предупреждения несанкционированных изменений модулей ядра в составе ОС применяется модуль lkrg (см. 16.6.15), который обеспечивает мониторинг угроз и блокирование несанкционированных изменений в ядре ОС. Таким образом, целостность всей области памяти ВМ при использовании lkrg обеспечивается по умолчанию. Для обработки критически важных данных рекомендуется на хосте использовать ОС с включенным модулем lkrg и ЗПС (см. 16.1), а в гостевых операционных системах применять ОС на усиленном или максимальном уровне защищенности и включенным ЗПС.

Для изоляции и управления виртуальными гостевыми машинами используется технология KVM, которая включает специальный модуль ядра KVM и средство создания виртуального аппаратного окружения QEMU для изоляции и управления виртуальными гостевыми машинами. KVM, используя загруженный в память модуль ядра, с помощью драйвера поль-

зовательского режима (который представляет собой модифицированный драйвер от QEMU) эмулирует слой аппаратного обеспечения, в среде которого могут создаваться и запускаться виртуальные машины.

В архитектуре KVM виртуальная машина выполняется как обычный процесс ОС, на который распространяется действие мер по изоляции процессов. Каждая гостевая операционная система в рамках своей виртуальной машины на уровне хостовой операционной системы является обычным процессом, выполняющимся с копией процесса `qemu`. Для каждого виртуального процессора создается поток. Эти процессы и потоки не отличаются от аналогичных в хостовой операционной системе. В части изоляции и разграничения доступа процессы виртуальных машин будут рассматриваться как обычные процессы операционной системы и для них будут применяться стандартные правила разграничения доступа и контроля ресурсов системы. Такие правила гарантируют, что программное обеспечение, запущенное в виртуальной машине, не сможет снизить производительность или нарушить работу других виртуальных машин или гипервизора.

Решение задачи изоляции адресных пространств процессов основано на архитектуре ядра ОС (см. раздел 7). Изоляция областей памяти виртуальных машин обеспечивается путем применения механизма управления памятью аппаратной платформы IOMMU (input/output memory management unit), который отвечает за трансляцию виртуальных адресов, видимых аппаратным устройством, в физические адреса и позволяет задавать ограничения операций ввода-вывода для изоляции при использовании виртуализации.

Блок управления памятью для операций ввода-вывода (IOMMU) — это тип блока управления памятью (MMU), который подключает шину расширения с поддержкой прямого доступа к памяти (DMA) к основной памяти. Он расширяет системную архитектуру, добавляя поддержку виртуализации адресов памяти, используемых периферийными устройствами. Кроме того, он обеспечивает изоляцию и защиту памяти, позволяя системному программному обеспечению контролировать, к каким областям физической памяти может обращаться устройство ввода-вывода. Это также помогает фильтровать и переназначать прерывания от периферийных устройств. Устройство может получить доступ только к областям памяти, которые сопоставлены для него. Следовательно, неисправные и/или вредоносные устройства не могут повредить память. Изоляция памяти позволяет безопасно назначать устройства виртуальной машине без ущерба для хоста и других гостевых ОС.

Для применения механизма управления памятью аппаратной платформы необходимо включить аппаратную поддержку виртуализации в настройках UEFI для процессоров Intel/AMD и включить вложенную виртуализацию для ВМ. Для этого следует:

1) убедиться, что в настройках BIOS аппаратной платформы включена поддержка виртуализации, процессор AMD-Vi/Intel VT-d поддерживается и включен в настройках BIOS;

2) создать файл /etc/modprobe.d/kvm.conf с содержимым в зависимости от архитектуры:

```
kvm_intel nested=1
```

или

```
kvm_amd nested=1
```

3) выполнить перезагрузку хостовой машины. После загрузки хостовой машины проверить активацию nested виртуализации можно командой:

```
sudo virt-host-validate
```

4) для включения IOMMU на хосте гипервизора необходимо добавить в конфигурационный файл GRUB /etc/default/grub параметр intel_iommu=on или amd_iommu=on (в зависимости от архитектуры):

```
GRUB_CMDLINE_LINUX_DEFAULT="parsec.mac=0 quiet net.ifnames=0
intel_iommu=on"
```

Затем обновить конфигурацию загрузчика:

```
sudo update-grub
```

5) перезагрузить систему и выполнить проверку наличия модуля IOMMU в системе, выполнив команду:

```
sudo virt-host-validate
```

Результат выполнения команды при успешной проверке:

```
"QEMU: Checking if IOMMU is enabled by kernel"
```

6) в конфигурацию ВМ в секцию <features> добавить строку <ioapic driver="qemu"/>:

```
<features>
<acpi/>
<apic/>
<vmport state="off"/>
<ioapic driver="qemu"/>
</features>
```

7) в конфигурацию ВМ в секцию <devices> добавить следующие строки (указав для параметра iommu model в качестве значения intel или amd в зависимости от архитектуры):

```
<iommu model="intel">
<driver intremap="on" caching_mode="on"/>
</iommu>
```

8) загрузить ВМ и в гостевой операционной системе в конфигурационном файле GRUB `/etc/default/grub` также указать загрузку IOMMU:

```
GRUB_CMDLINE_LINUX_DEFAULT="parsec.mac=0 quiet net.ifnames=0
intel_iommu=on"
```

Затем обновить конфигурацию загрузчика:

```
sudo update-grub
```

9) перезагрузить гостевую операционную систему;

10) загрузить ВМ и в гостевой операционной системе выполнить команду:

```
sudo dmesg | grep -e DMAR -e IOMMU
```

В выводе проверить наличие информации об активации IOMMU:

```
[    0.008446] ACPI: DMAR 0x000000007FFE2961 0000E8 (v01 BOCHS  BXPC
00000001 BXPC 00000001)
[    0.008457] ACPI: Reserving DMAR table memory at [mem
0x7ffe2961-0x7ffe2a48]
[    0.021407] DMAR: IOMMU enabled
```

11) в гостевой операционной системе для проверки наличия модуля IOMMU выполнить команду:

```
sudo virt-host-validate
```

Результат выполнения команды при успешной проверке:

```
QEMU: проверка для поддержки сопоставления устройств IOMMU: OK
QEMU: проверка если блок IOMMU включен в ядре: OK
```

5.12. Применение механизма контроля целостности областей памяти по запросу из гостевой операционной системы

Механизм контроля целостности областей памяти ВМ базируется на создании канала приема-передачи данных между ВМ и средством виртуализации с использованием механизма передачи запросов из гостевой операционной системы посредством служебного сокета типа `vsock`.

Контроль осуществляется средством виртуализации путем вычисления контрольных сумм заданных областей памяти в соответствии с ГОСТ Р 34.11-2012 с последующим сравнением вычисленных значений с эталонными. Подсчет контрольных сумм осуществляется периодически в процессе работы гостевой операционной системы. При миграции ВМ данные об эталонных значениях установленных на контроль областей памяти будут также пере-

даваться вместе с другой служебной информацией о ВМ для возможного последующего выполнения контроля целостности на другом хосте.

Для приема запросов на осуществление контроля целостности областей памяти гостевой операционной системы используется служба `vsockd` на хосте гипервизора, которая открывает слушающий сокет VSOCK. Виртуальная машина передает в сокет сообщение с параметрами `start_addr` (физический адрес начала блока памяти гостевой операционной системы) и `end_addr` (физический адрес конца блока памяти гостевой операционной системы). Служба `vsockd` на хосте принимает сообщение, получает адрес CID VSOCK и отправляет запрос в средство виртуализации (libvirt) для последующего вычисления контрольной суммы по алгоритму ГОСТ Р 34.11-2012.

Эталонные значения контрольных сумм областей памяти хранятся в памяти средства виртуализации до выключения ВМ. При выявлении нарушения целостности областей памяти гостевой операционной системы возможна блокировка работы ВМ и регистрация фактов нарушения целостности объектов контроля в журнал подсистемы регистрации событий (см. 6.5) и в журнал `/var/log/syslog`.

Для использования механизма в системе должен быть установлен пакет `astra-kvm-secure`.

Настройка механизма контроля целостности файлов гостевой операционной системы осуществляется в конфигурационном файле `/etc/libvirt/libvirtd.conf`. Настройку рекомендуется выполнять с использованием `virsh` путем выполнения команды:

```
virsh -c qemu:///system config --edit-config /etc/libvirt/libvirtd.conf
```

Для включения механизма контроля целостности областей памяти гостевой операционной системы необходимо в файле `/etc/libvirt/libvirtd.conf` для параметра `memory_integrity_check_period_s` задать значение периода проверки (в секундах):

```
memory_integrity_check_period_s = 30
```

При этом будет обеспечиваться только регистрация фактов нарушения целостности объектов контроля без блокировки работы ВМ.

Для принудительного выключения функционирующей ВМ в случае нарушения целостности установленных на контроль областей памяти необходимо в файле `/etc/libvirt/libvirtd.conf` для параметра `memory_integrity_check_shutdown_domain` задать значение `1`:

```
memory_integrity_check_shutdown_domain = 1
```

При этом будет обеспечиваться регистрация фактов нарушения целостности областей памяти с последующим принудительным выключением ВМ, если значение контрольной суммы хотя бы одной установленной на контроль области памяти гостевой операционной системы не совпадет с эталонным значением, хранящимся в конфигурационном файле ВМ.

Для применения настроек следует перезагрузить службу `libvirtd`:

```
sudo systemctl restart libvirtd
```

К ВМ необходимо подключить сокет VirtIO VSOCK для связи гостевой операционной системы с хостом. Для этого необходимо в `virt-manager`:

1) открыть свойства ВМ и нажать **[Добавить оборудование]**;
2) в окне «Добавление виртуального оборудования» выбрать «VirtIO VSOCK»;
3) во вкладке «Подробности» для параметра «CID гостевой системы» установить флаг «Авто»;
4) нажать **[Готово]**.

В результате в конфигурационном файле ВМ будет сформирована запись вида:

```
<vsock model="virtio">
<cid auto="yes" address="3"/>
<address type="pci" domain="0x0000" bus="0x07" slot="0x00" function="0x0"/>
</vsock>
```

Каждому сокету автоматически присваивается метка CID гостевой операционной системы, которая обозначает сквозной идентификатор каждой ВМ.

После добавления сокета необходимо выполнить запуск гостевой операционной системы.

Использование механизма контроля целостности возможно при условии использования ОС в качестве гостевой операционной системы и установки в ней гостевого агента `qemu-guest-agent` из состава ОС.

Установку на контроль областей памяти обеспечивает инструмент `/sbin/memcontrol_VM`, принимающий в качестве аргументов данные об идентификаторе контролируемого процесса (pid) и диапазон виртуальных адресов блока памяти в шестнадцатеричном формате.

Синтаксис команды:

1) в случае использования виртуальных адресов процесса с pid:

```
memcontrol_VM --virt <pid> <address_start> <address_end>
```

2) в случае использования физических адресов памяти гостевой операционной системы для постановки на контроль:

```
memcontrol_VM --phys <address_start> <address_end>
```

Для определения диапазона виртуальных адресов блока памяти ВМ в гостевой операционной системе используется инструмент `ps`, выводящий идентификатор процесса по его названию, и файл `/proc/<pid>/maps`, содержащий адреса областей памяти, которые используются процессом в данный момент, с указанием прав доступа к ним.

Пример

Для установки на контроль областей памяти, занимаемых процессом `syslog-ng`, необходимо:

1) определить идентификатор процесса командой:

```
ps aux | grep syslog-ng
```

Вывод команды:

```
root  394  0.5  1.8  624928  38180  ?      Ssl  16:20  0:27
/usr/sbin/syslog-ng -F --no-caps
u    1438  0.0  0.0    6228    872  pts/2  S+   17:44  0:00 grep
syslog-ng
```

В выводе команды pid процесса равен `394`;

2) выполнить поиск адресов областей памяти, которые используются процессом с pid, равным `394`, в данный момент:

```
sudo cat /proc/394/maps
```

Вывод команды:

```
00400000-00402000 r--p 00000000 08:01 1320884 /usr/sbin/syslog-ng
00402000-00403000 r-xp 00002000 08:01 1320884 /usr/sbin/syslog-ng
00403000-00404000 r--p 00003000 08:01 1320884 /usr/sbin/syslog-ng
00404000-00405000 r--p 00003000 08:01 1320884 /usr/sbin/syslog-ng
00405000-00406000 rw-p 00004000 08:01 1320884 /usr/sbin/syslog-ng
00c12000-01165000 rw-p 00000000 00:00 0 [heap]
731f6c000000-731f6c070000 rw-p 00000000 00:00 0
731f6c070000-731f70000000 ---p 00000000 00:00 0
731f737ff000-731f73800000 ---p 00000000 00:00 0
731f73800000-731f74000000 rw-p 00000000 00:00 0
731f74000000-731f74023000 rw-p 00000000 00:00 0
731f74023000-731f78000000 ---p 00000000 00:00 0
731f78000000-731f78021000 rw-p 00000000 00:00 0
731f78021000-731f7c000000 ---p 00000000 00:00 0
731f7c000000-731f7c021000 rw-p 00000000 00:00 0
731f7c021000-731f80000000 ---p 00000000 00:00 0
731f80000000-731f80021000 rw-p 00000000 00:00 0
```

```
731f80021000-731f84000000 ---p 00000000 00:00 0
731f84000000-731f84032000 rw-p 00000000 00:00 0
731f84032000-731f88000000 ---p 00000000 00:00 0
731f88000000-731f88021000 rw-p 00000000 00:00 0
731f88021000-731f8c000000 ---p 00000000 00:00 0
731f8c000000-731f8c025000 rw-p 00000000 00:00 0
731f8c025000-731f90000000 ---p 00000000 00:00 0
```

Сегмент с правами доступа `r-xp` указывает на хранение исполняемого кода `syslog-ng`. Соответствующая ему область памяти имеет диапазон адресов `00402000-00403000` (`0x402000-0x403000` в шестнадцатеричном формате);

3) из гостевой операционной системы выполнить запрос постановки на контроль целостности выбранной области памяти для процесса с pid, равным `394`:

```
sudo memcontrol_VM --virt 394 0x402000 0x403000
```

В выводе команды будут указаны физические адреса начала и конца блока поставленной на контроль памяти гостевой операционной системы и рассчитанное для этой области значение эталонной контрольной суммы.

В результате контрольная сумма области памяти будет записана в памяти средства виртуализации с целью дальнейшего контроля целостности.

Для управления режимом контроля целостности областей памяти для отдельной ВМ используется команда:

```
virsh -c qemu:///system memory-integrity <domain> [--clear] [--start] [--stop]
```

где `<domain>` — наименование ВМ, допускается указывать идентификатор ВМ, присвоенный средой виртуализации (ID), или всемирно уникальный идентификатор ВМ (UUID).

Более подробная информация об использовании команды приведена на странице помощи:

```
virsh help memory-integrity
```

5.13. Ограничение программной среды в среде виртуализации

Контроль за запуском компонентов программного обеспечения, обеспечивающий выявление и блокировку запуска компонентов программного обеспечения, не включенных в перечень (список) компонентов, разрешенных для запуска, осуществляется штатными средствами ОС.

Механизм динамического контроля целостности (режим ЗПС) обеспечивает выявление и блокировку запуска компонентов программного обеспечения, целостность которого нарушена. В исполняемые файлы и разделяемые библиотеки, входящие в состав ПО, внедряется цифровая подпись. При включенном режиме ЗПС выполняться будут только подписанные файлы. Запуск файлов без цифровой подписи или с неверной подписью будет блокироваться. Описание ЗПС приведено в 16.1.

Применение режима Киоск-2 ограничивает права пользователей на запуск программ в ОС. Степень этих ограничений задается маской киоска, которая накладывается на права доступа к исполняемым файлам при любой попытке пользователя получить доступ. Описание Киоск-2 приведено в 16.2.

5.14. Централизованное управление

В среде виртуализации реализовано создание, модификация, хранение, получение и удаление (в т. ч. централизованное) образов виртуальных машин в информационной (автоматизированной) системе.

Для централизованного хранения образов ВМ используется хранилище данных — пул (pool). Хранилище определяет физическое расположение файлов-образов и может быть различных типов. Хранилище должно быть доступно для подключения на серверах виртуализации.

Для распределенного хранилища следует использовать кластерную файловую систему ocfs2 или блочное устройство ceph/rbd. Образы виртуальных машин помещаются в данное хранилище и становятся доступными для всех узлов виртуализации.

ОС поддерживает возможность миграции виртуальных машин между серверами виртуализации. Миграция позволяет перенести работу виртуальной машины с одного физического хоста (сервера виртуализации) на другой без остановки ее работы. При этом образ диска виртуальной машины доступен на обоих серверах виртуализации. Для этого используется распределенное хранилище данных: на обоих серверах виртуализации хранилище должно монтироваться в одно и то же место и образ виртуальной машины должен быть расположен в данном хранилище.

Управление перемещением виртуальных машин реализуется с использованием интерфейсов управления средствам виртуализации libvirt в соответствии с установленными правилами сетевого взаимодействия.

Для решения задач централизованного протоколирования и анализа журналов аудита используется Zabbix.

Для решения задач централизованного сбора журналов с удаленных хостов виртуализации возможно применение встроенных механизмов сбора информации о событиях `syslog-ng`. Вместе с установкой пакета `astra-kvm-secure` обеспечивается создание конфигурацион-

ных файлов для удаленного сбора данных о событиях безопасности, связанных с функционированием средства виртуализации, из журналов событий безопасности.

5.14.1. Пример организации распределенного хранилища

Организация распределенного хранилища для серверов виртуализации на примере двух серверов виртуализации и сервера хранения данных (СХД):

1) СХД — имя компьютера `astra-storage`, IP-адрес `172.16.1.20`;
2) первый сервер виртуализации — имя компьютера `astra1`, IP-адрес `172.16.1.21`;
3) второй сервер виртуализации — имя компьютера `astra2`, IP-адрес `172.16.1.22`.

Компьютеры должны быть объединены в локальную сеть и доступны по имени компьютера (в `/etc/hosts` на каждом компьютере должны быть выполнены соответствующие настройки).

Для создания сетевого хранилища данных на СХД применяется технология SAN, позволяющая монтировать на компьютере сетевое блочное устройство как локальное блочное устройство. Доступ к сетевому блочному устройству настраивается по протоколу iSCSI.

5.14.1.1. Создание и подключение сетевого хранилища

Для создания SAN-хранилища (сетевого блочного устройства) необходимо выполнить следующие действия на `astra-storage`:

1) вывести перечень имеющихся блочных устройств командой:

```
lsblk
```

Пример вывода команды:

```
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda      8:0    0 465,8G  0 disk
sdb      8:16   0 298,1G  0 disk
|__sdb1  8:17   0   512M  0 part /boot/efi
|__sdb2  8:18   0 296,7G  0 part /
|__sdb3  8:19   0   977M  0 part [SWAP]
sdc      8:32   0 465,8G  0 disk
|__sdc1  8:33   0 465,8G  0 part
sr0      11:0   1  1024M  0 rom
```

Из вывода команды определить блочное устройство, на основе которого будет создано сетевое блочное устройство (например, `sda`);

2) установить консоль управления Linux-IO Target командой:

```
sudo apt install targetcli-fb
```

3) запустить консоль управления Linux-IO Target командой:

```
sudo targetcli
```

4) в консоли управления Linux-IO Target вывести текущую конфигурацию командой:

```
ls
```

Пример вывода команды:

```
o- / ..................................................... [...]
o- backstores ........................................... [...]
| o- block ..................................... [Storage Objects: 0]
| o- fileio .................................... [Storage Objects: 0]
| o- pscsi ..................................... [Storage Objects: 0]
| o- ramdisk ................................... [Storage Objects: 0]
o- iscsi ..................................................... [Targets: 0]
o- loopback .................................................. [Targets: 0]
o- vhost ..................................................... [Targets: 0]
o- xen-pvscsi ................................................ [Targets: 0]
```

5) создать (зарегистрировать) сетевое блочное устройство в разделе `/backstores/block` командой:

```
/backstores/block create <имя_устройства> /dev/<блочное_устройство>
```

где `<имя_устройства>` — наименование, которое будет присвоено создаваемому сетевому блочному устройству;

`<блочное_устройство>` — выбранное блочное устройство для создания сетевого блочного устройства.

Пример команды для текущей конфигурации:

```
/backstores/block create san_storage /dev/sda
```

Результат выполнения команды:

```
Created block storage object san_storage using /dev/sda
```

6) проверить создание сетевого блочного устройства `san_storage` командой:

```
ls
```

Результат выполнения команды:

```
o- / ..................................................... [...]
o- backstores ........................................... [...]
| o- block ..................................... [Storage Objects: 1]
| | o- san_storage ...... [/dev/sda (465.8GiB) write-thru deactivated]
| |   o- alua ....................................... [ALUA Groups: 1]
| |     o- default_tg_pt_gp ........... [ALUA state: Active/optimized]
| o- fileio .................................... [Storage Objects: 0]
| o- pscsi ..................................... [Storage Objects: 0]
```

```
| o- ramdisk ................................ [Storage Objects: 0]
o- iscsi ........................................... [Targets: 0]
o- loopback ........................................ [Targets: 0]
o- vhost ........................................... [Targets: 0]
o- xen-pvscsi ...................................... [Targets: 0]
```

7) создать цель для сетевого блочного устройства (iSCSI-target) в разделе `/iscsi` командой:

```
/iscsi create
```

Результат выполнения команды:

```
Created target iqn.2003-01.org.linux-iscsi.astra-storage.x8664:
sn.727a2d3719d4.
Created TPG 1.
Global pref auto_add_default_portal=true
Created default portal listening on all IPs (0.0.0.0), port 3260
```

Для сетевого блочного устройства была создана цель `iqn.2003-01.org.linux-iscsi.astra-storage.x8664:sn.727a2d3719d4`;

8) проверить создание цели командой:

```
ls
```

Результат выполнения команды:

```
o- / ..................................................... [...]
o- backstores ........................................... [...]
| o- block ..................................... [Storage Objects: 1]
| | o- san_storage ..... [/dev/sda (465.8GiB) write-thru deactivated]
| |   o- alua ....................................... [ALUA Groups: 1]
| |     o- default_tg_pt_gp ........... [ALUA state: Active/optimized]
| o- fileio .................................... [Storage Objects: 0]
| o- pscsi ..................................... [Storage Objects: 0]
| o- ramdisk ................................... [Storage Objects: 0]
o- iscsi ........................................... [Targets: 1]
| o- iqn.2003-01.org.linux-iscsi.astra-storage.x8664:sn.727a2d3719d4
[TPGs: 1]
|   o- tpg1 ................................... [no-gen-acls, no-auth]
|     o- acls ........................................... [ACLs: 0]
|     o- luns ........................................... [LUNs: 0]
|     o- portals ..................................... [Portals: 1]
|       o- 0.0.0.0:3260 ...................................... [OK]
o- loopback ........................................ [Targets: 0]
o- vhost ........................................... [Targets: 0]
o- xen-pvscsi ...................................... [Targets: 0]
```

9) создать LUN (Logical Unit Number — номер логического устройства) на основе сетевого блочного устройства (зарегистрированного в разделе `/backstores/block`) командой:

```
/iscsi/<цель>/tpg1/luns/ create /backstores/block/<имя_устройства>
```

где `<цель>` — цель для сетевого блочного устройства, созданная согласно пункту перечисления 7);

`<имя_устройства>` — наименование сетевого блочного устройства, заданное при его создании согласно пункту перечисления 5).

Пример команды для текущей конфигурации:

```
/iscsi/iqn.2003-01.org.linux-iscsi.astra-storage.x8664:sn.727a2d3719d4/\
     tpg1/luns/ create /backstores/block/san_storage
```

Результат выполнения команды:

```
Created LUN 0
```

10) проверить результат создания LUN командой:

```
ls
```

Результат выполнения команды:

```
o- / ..................................................... [...]
o- backstores ........................................... [...]
| o- block ..................................... [Storage Objects: 1]
| | o- san_storage ........ [/dev/sda (465.8GiB) write-thru activated]
| |   o- alua ....................................... [ALUA Groups: 1]
| |     o- default_tg_pt_gp ........... [ALUA state: Active/optimized]
| o- fileio .................................... [Storage Objects: 0]
| o- pscsi ..................................... [Storage Objects: 0]
| o- ramdisk ................................... [Storage Objects: 0]
o- iscsi ................................................ [Targets: 1]
| o- iqn.2003-01.org.linux-iscsi.astra-storage.x8664:sn.727a2d3719d4
TPGs: 1]
|   o- tpg1 ................................... [no-gen-acls, no-auth]
|     o- acls ............................................. [ACLs: 0]
|     o- luns ............................................. [LUNs: 1]
|     | o- lun0 .... [block/san_storage (/dev/sda) (default_tg_pt_gp)]
|     o- portals ........................................ [Portals: 1]
|       o- 0.0.0.0:3260 ......................................... [OK]
o- loopback ............................................. [Targets: 0]
o- vhost ................................................ [Targets: 0]
o- xen-pvscsi ........................................... [Targets: 0]
```

11) в данном примере не используется авторизация и аутентификация, поэтому для отключения контроля доступа к цели необходимо последовательно выполнить команды:

```
cd /iscsi/<цель>/tpg1
set attribute generate_node_acls=1
set attribute demo_mode_write_protect=0
```

Примеры команд для текущей конфигурации:

а) `cd /iscsi/iqn.2003-01.org.linux-iscsi.astra-storage.x8664:\`
`sn.727a2d3719d4/tpg1`

б) `set attribute generate_node_acls=1`

Результат выполнения команды:

```
Parameter generate_node_acls is now '1'
```

в) `set attribute demo_mode_write_protect=0`

Результат выполнения команды:

```
Parameter demo_mode_write_protect is now '0'
```

12) сохранить конфигурацию сетевого блочного устройства командой:

```
/ saveconfig
```

Результат выполнения команды:

```
Configuration saved to /etc/rtslib-fb-target/saveconfig.json
```

Конфигурация сетевого блочного устройства сохранена в файл `/etc/rtslib-fb-target/saveconfig.json`;

13) выйти из консоли управления Linux-IO Target командой:

```
exit
```

Результат выполнения команды:

```
Global pref auto_save_on_exit=true
Last 10 configs saved in /etc/rtslib-fb-target/backup/.
Configuration saved to /etc/rtslib-fb-target/saveconfig.json
```

5.14.1.2. Подключение на серверах виртуализации сетевого блочного устройства

Созданное на `astra-storage` сетевое блочное устройство (iSCSI-target) необходимо подключить в качестве локального блочного устройства на серверах виртуализации (iSCSI-initiator).

Данные настройки должны быть выполнены на всех компьютерах, на которых настраивается доступ к сетевому блочному устройству. В данном примере настройки выполняются на

серверах виртуализации astra1 и astra2. Для подключения сетевого блочного устройства к серверу виртуализации необходимо:

1) установить пакет open-iscsi командой:

```
sudo apt install open-iscsi
```

2) настроить автоматическое подключение LUN при перезагрузке сервера виртуализации. Для этого в конфигурационном файле /etc/iscsi/iscsid.conf для параметра node.startup установить значение automatic:

```
node.startup = automatic
```

3) запустить службу iscsi командой:

```
sudo systemctl start iscsi
```

После первого запуска службы iscsi будет сгенерирован уникальный идентификатор инициатора (iSCSI-initiator), который можно просмотреть в файле /etc/iscsi/initiatorname.iscsi, выполнив команду:

```
sudo cat /etc/iscsi/initiatorname.iscsi
```

Результат выполнения команды на astra1:

```
## DO NOT EDIT OR REMOVE THIS FILE!
## If you remove this file, the iSCSI daemon will not start.
## If you change the InitiatorName, existing access control lists
## may reject this initiator.  The InitiatorName must be unique
## for each iSCSI initiator.  Do NOT duplicate iSCSI InitiatorNames.
InitiatorName=iqn.1993-08.org.debian:01:694dac4a9eba
```

Результат выполнения команды на astra2:

```
## DO NOT EDIT OR REMOVE THIS FILE!
## If you remove this file, the iSCSI daemon will not start.
## If you change the InitiatorName, existing access control lists
## may reject this initiator.  The InitiatorName must be unique
## for each iSCSI initiator.  Do NOT duplicate iSCSI InitiatorNames.
InitiatorName=iqn.1993-08.org.debian:01:7c60f380d294
```

4) выполнить поиск доступных целей (сетевых блочных устройств, iSCSI-target):

```
sudo iscsiadm -m discovery -t st -p <IP-адрес>
```

где <IP-адрес> — IP-адрес компьютера, на котором зарегистрировано сетевое блочное устройство.

Пример команды для текущей конфигурации:

```
sudo iscsiadm -m discovery -t st -p 172.16.1.20
```

Результат выполнения команды:

```
172.16.1.20:3260,1 iqn.2003-01.org.linux-iscsi.astra-storage.x8664:
sn.727a2d3719d4
```

5) автоматически подключить все найденные цели:

```
sudo iscsiadm -m node -l
```

Результат выполнения команды:

```
Logging in to [iface: default, target:
iqn.2003-01.org.linux-iscsi.astra-storage.x8664:sn.727a2d3719d4,
portal: 172.16.1.20,3260]
Login to [iface: default, target:
iqn.2003-01.org.linux-iscsi.astra-storage.x8664:sn.727a2d3719d4,
portal: 172.16.1.20,3260] successful.
```

6) проверить, что на сервере виртуализации было добавлено новое блочное устройство:

```
lsblk
```

Результат выполнения команды на `astra1`:

```
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda      8:0    0 931,5G  0 disk
|_sda1   8:1    0   512M  0 part /boot/efi
|_sda2   8:2    0    28G  0 part /
|_sda3   8:3    0  18,6G  0 part [SWAP]
|_sda4   8:4    0 884,5G  0 part /home
sdb      8:16   0 465,8G  0 disk
sr0     11:0    1  1024M  0 rom
```

На `astra1` добавлено новое блочное устройство `/dev/sdb`.
Результат выполнения команды на `astra2`:

```
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda           8:0    0 465,8G  0 disk
nvme0n1     259:0    0 238,5G  0 disk
|_nvme0n1p1 259:1    0   512M  0 part /boot/efi
|_nvme0n1p2 259:2    0    28G  0 part /
|_nvme0n1p3 259:3    0   7,5G  0 part [SWAP]
|_nvme0n1p4 259:4    0 202,6G  0 part /home
```

На `astra2` добавлено новое блочное устройство `/dev/sda`.

5.14.1.3. Настройка кластера

Для использования сетевого хранилища серверами виртуализации требуется создать кластер и разметить на нем кластерную файловую систему OCFS2.

Создание кластера

Для создания кластера необходимо на одном из узлов кластера (например, astra1) выполнить следующие действия:

1) установить пакет ocfs2-tools командой:

```
sudo apt install ocfs2-tools
```

2) создать ocfs2-кластер командой:

```
sudo o2cb add-cluster <имя_кластера>
```

Например:

```
sudo o2cb add-cluster ocfs2cluster
```

В результате выполнения команды будет создан файл конфигурации кластера /etc/ocfs2/cluster.conf;

3) добавить описание узлов кластера, выполнив команду для каждого узла кластера:

```
sudo o2cb add-node <имя_кластера> <сетевое_имя_узла> --ip <IP-адрес_узла>
```

где <имя_кластера> — имя созданного кластера;
<сетевое_имя_узла> — имя компьютера, добавляемого в качестве узла кластера;
<IP-адрес_узла> — IP-адрес компьютера, добавляемого в качестве узла кластера.

Имя узла кластера должно соответствовать имени компьютера, указанному в /etc/hostname, также имена компьютеров и их IP-адреса должны быть указаны в /etc/hosts на других узлах кластера.

Пример команд для текущей конфигурации:

```
sudo o2cb add-node ocfs2cluster astra1 --ip 172.16.1.21
sudo o2cb add-node ocfs2cluster astra2 --ip 172.16.1.22
```

Данные о добавленных узлах кластера записываются в /etc/ocfs2/cluster.conf.

Проверить содержимое файла конфигурации кластера возможно с помощью команды:

```
cat /etc/ocfs2/cluster.conf
```

Результат выполнения команды:

```
cluster:
heartbeat_mode = local
```

```
node_count = 2
name = ocfs2cluster

node:
number = 0
cluster = ocfs2cluster
ip_port = 7777
ip_address = 172.16.1.21
name = astra1

node:
number = 1
cluster = ocfs2cluster
ip_port = 7777
ip_address = 172.16.1.22
name = astra2
```

На всех остальных узлах кластера требуется выполнить следующие действия для настройки кластера:

1) установить пакет `ocfs2-tools` командой:

```
sudo apt install ocfs2-tools
```

2) скопировать в локальный каталог `/etc/ocfs2/` файл конфигурации кластера:

```
sudo scp \
     <локальный_администратор>@<IP-адрес_узла>:/etc/ocfs2/cluster.conf \
     /etc/ocfs2/
```

где `<IP-адрес_узла>` — IP-адрес узла кластера, на котором была выполнена настройка кластера и с которого копируется файл конфигурации;
`<локальный_администратор>` — имя локального администратора узла кластера, на котором была выполнена настройка кластера и с которого копируется файл конфигурации.

В ходе выполнения команды требуется:

а) на запрос пароля для команды `sudo` — ввести пароль локального администратора узла кластера, на который копируется файл;

б) на запрос установки соединения ответить «yes» («Да»);

в) ввести пароль локального администратора узла кластера, с которого копируется файл конфигурации.

Пример команды для текущей конфигурации, выполняется на `astra2`:

```
sudo scp admin1@172.16.1.21:/etc/ocfs2/cluster.conf /etc/ocfs2
```

Результат выполнения команды:

```
[sudo] пароль для admin2:
The authenticity of host ’172.16.1.21 (172.16.1.21)’ can’t be established.
ECDSA key fingerprint is SHA256:mONkRKlGRAc1ZEywS/sYRVdZINN2PAHJIKRwLFzMGSM.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added ’172.16.1.21’ (ECDSA) to the list of known hosts.
admin1@172.16.1.21’s password:
cluster.conf 100% 371 197.3KB/s 00:00
```

На узле `astra2` проверить полученный файл конфигурации:

```
cat /etc/ocfs2/cluster.conf
```

Файл конфигурации кластера на узле `astra2` `/etc/ocfs2/cluster.conf` должен быть идентичен файлу конфигурации кластера на узле `astra1`.

На каждом узле настроить кластер OCFS2:

1) запустить мастер настройки кластера OCFS2 командой:

```
sudo dpkg-reconfigure ocfs2-tools
```

2) в мастере настройки кластера OCFS2:
   а) разрешить запускать кластер OCFS2 (O2CB) во время загрузки, нажав кнопку **[Да]**;
   б) задать имя кластера (например, `ocfs2cluster`) и нажать кнопку **[Ok]**;
   в) для остальных параметров выбрать значения, установленные по умолчанию;
3) перезапустить службу `o2cb` командой:

```
sudo systemctl restart o2cb
```

4) проверить успешный запуск службы `o2cb`. Для этого просмотреть в журнале регистрации события, относящиеся к службе:

```
sudo journalctl -u o2cb
```

Пример вывода команды на узле `astra1`:

```
-- Logs begin at Thu 2023-06-01 13:49:19 MSK, end at Thu 2023-06-01
14:12:52 MSK. --
июн 01 14:10:38 astra1 systemd[1]: Starting Load o2cb Modules...
июн 01 14:10:38 astra1 o2cb[3293]: checking debugfs...
июн 01 14:10:39 astra1 o2cb[3293]: Loading stack plugin "o2cb": OK
июн 01 14:10:39 astra1 o2cb[3293]: Loading filesystem "ocfs2_dlmfs": OK
июн 01 14:10:39 astra1 o2cb[3293]: Mounting ocfs2_dlmfs filesystem at /dlm: OK
июн 01 14:10:39 astra1 o2cb[3293]: Setting cluster stack "o2cb": OK
июн 01 14:10:39 astra1 o2cb[3293]: Registering O2CB cluster "ocfs2cluster": OK
```

```
июн 01 14:10:39 astra1 o2cb[3293]: Setting O2CB cluster timeouts : OK
июн 01 14:10:39 astra1 o2hbmonitor[3342]: Starting
июн 01 14:10:39 astra1 systemd[1]: Started Load o2cb Modules.
июн 01 14:12:34 astra1 systemd[1]: Stopping Load o2cb Modules...
июн 01 14:12:34 astra1 o2cb[3418]: Clean userdlm domains: OK
июн 01 14:12:34 astra1 o2cb[3418]: Stopping O2CB cluster ocfs2cluster:
Unregistering O2
июн 01 14:12:34 astra1 o2cb[3418]: Unmounting ocfs2_dlmfs filesystem: OK
июн 01 14:12:34 astra1 o2cb[3418]: Unloading module "ocfs2_dlmfs": OK
июн 01 14:12:34 astra1 o2cb[3418]: Unloading module "ocfs2_stack_o2cb": OK
июн 01 14:12:34 astra1 systemd[1]: o2cb.service: Succeeded.
июн 01 14:12:34 astra1 systemd[1]: Stopped Load o2cb Modules.
июн 01 14:12:34 astra1 systemd[1]: o2cb.service: Consumed 185ms CPU time.
июн 01 14:12:34 astra1 systemd[1]: Starting Load o2cb Modules...
июн 01 14:12:34 astra1 o2cb[3485]: checking debugfs...
июн 01 14:12:34 astra1 o2cb[3485]: Loading stack plugin "o2cb": OK
июн 01 14:12:34 astra1 o2cb[3485]: Loading filesystem "ocfs2_dlmfs": OK
июн 01 14:12:34 astra1 o2cb[3485]: Mounting ocfs2_dlmfs filesystem at /dlm: OK
июн 01 14:12:34 astra1 o2cb[3485]: Setting cluster stack "o2cb": OK
июн 01 14:12:34 astra1 o2cb[3485]: Registering O2CB cluster "ocfs2cluster": OK
июн 01 14:12:34 astra1 o2cb[3485]: Setting O2CB cluster timeouts : OK
июн 01 14:12:34 astra1 o2hbmonitor[3531]: Starting
июн 01 14:12:34 astra1 systemd[1]: Started Load o2cb Modules.
```

В журнале должны отсутствовать сообщения об ошибках запуска службы. Для выхода из просмотра журнала нажать клавишу **<q>**.

Разметка кластерной файловой системы

Разметка подключенного сетевого блочного устройства для использования кластерной файловой системы OCFS2 выполняется на одном из узлов кластера.

Для разметки необходимо выполнить команду:

```
sudo mkfs.ocfs2 -T vmstore <блочное_устройство>
```

где `-T` — запустить автоматическую тонкую настройку параметров файловой системы;

`vmstore` — производить тонкую настройку для обеспечения максимальной производительности при размещении на блочном устройстве файлов образов виртуальных машин;

`<блочное_устройство>` — сетевое блочное устройство, подключенное согласно 5.14.1.2.

Для текущей конфигурации выполнить разметку сетевого блочного устройства на узле `astra1` командой:

```
sudo mkfs.ocfs2 -T vmstore /dev/sdb
```

Вывод команды:

```
[sudo] пароль для admin1:
mkfs.ocfs2 1.8.5
Cluster stack: classic o2cb
Filesystem Type of vmstore
Label:
Features: sparse extended-slotmap backup-super unwritten inline-data
strict-journal-super xattr indexed-dirs refcount discontig-bg append-dio
Block size: 4096 (12 bits)
Cluster size: 1048576 (20 bits)
Volume size: 500107837440 (476940 clusters) (122096640 blocks)
Cluster groups: 15 (tail covers 25356 clusters, rest cover 32256 clusters)
Extent allocator size: 188743680 (45 groups)
Journal size: 134217728
Node slots: 8
Creating bitmaps: done
Initializing superblock: done
Writing system files: done
Writing superblock: done
Writing backup superblock: 5 block(s)
Formatting Journals: done
Growing extent allocator: done
Formatting slot map: done
Formatting quota files: done
Writing lost+found: done
mkfs.ocfs2 successful
```

В результате на блочном устройстве `/dev/sdb` (сетевое блочное устройство `san_storage`) была создана файловая система OCFS2.

Для просмотра доступных сетевых блочных устройств с кластерной файловой системой OCFS2 выполнить команду на любом из узлов кластера:

```
sudo mounted.ocfs2 -d
```

Результат выполнения команды на узле astra1 текущей конфигурации:

```
Device    Stack  Cluster  F  UUID                              Label
/dev/sdb  o2cb              E42202DFA47B42DB9B7DE6778555B4A4
```

Размеченное сетевое блочное устройство необходимо примонтировать на каждом узле кластера, для этого требуется:

1) создать точку монтирования:

```
sudo mkdir -p <точка_монтирования>
```

2) примонтировать сетевое блочное устройство:

```
sudo mount <сетевое_блочное_устройство> <точка_монтирования>
```

Пример команд на узле astra1 для текущей конфигурации:

1) создать точку монтирования:

```
sudo mkdir -p /mnt/ocfs2-storage
```

2) примонтировать сетевое блочное устройство /dev/sdb:

```
sudo mount /dev/sdb /mnt/ocfs2-storage/
```

3) для проверки монтирования сетевого блочного устройства выполнить команду:

```
lsblk
```

Результат выполнения команды:

```
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda      8:0    0 931,5G  0 disk
|_sda1   8:1    0   512M  0 part /boot/efi
|_sda2   8:2    0    28G  0 part /
|_sda3   8:3    0  18,6G  0 part [SWAP]
|_sda4   8:4    0 884,5G  0 part /home
sdb      8:16   0 465,8G  0 disk /mnt/ocfs2-storage
sr0     11:0    1  1024M  0 rom
```

Для просмотра примонтированных сетевых блочных устройств с кластерной файловой системой OCFS2 необходимо выполнить команду на любом из узлов кластера:

```
sudo mounted.ocfs2 -f
```

Результат выполнения команды на узле astra1 текущей конфигурации:

```
Device    Stack  Cluster  F  Nodes
/dev/sdb  o2cb              astra1, astra2
```

В выводе команды отображается сетевое блочное устройство и список узлов кластера, на которых оно примонтировано.

5.14.1.4. Автоматическое монтирование

Для обеспечения автоматического монтирования сетевого блочного устройства при загрузке узла необходимо:

1) определить UUID блочного устройства, выполнив команду на любом из узлов кластера:

```
sudo blkid
```

Результат выполнения команды на узле astra1 текущей конфигурации:

```
/dev/sda1: UUID="503E-AC24" TYPE="vfat"
PARTUUID="8dabfe8a-031e-4da6-b5be-33a8cba97322"
/dev/sda2: UUID="68de36df-e90e-4fb9-8bba-47dd0b5f4dca"
TYPE="ext4" PARTUUID="e2aa5324-2f15-4962-98f2-744ca4e8f844"
/dev/sda3: UUID="b2f367e4-3e3f-4c58-a159-0e222ebdc422"
TYPE="swap" PARTUUID="f2e75fdf-db3e-40ee-8a9f-ab634363c6fe"
/dev/sda4: UUID="abc173bd-a889-499e-9424-96ce8f8c2b9b"
TYPE="ext4" PARTUUID="f5de7620-7636-4bb0-afe2-1c4192a5c1aa"
/dev/sdb: UUID="e42202df-a47b-42db-9b7d-e6778555b4a4" TYPE="ocfs2"
```

2) на каждом узле кластера отредактировать файл /etc/fstab, добавив строку вида:

```
UUID=e42202df-a47b-42db-9b7d-e6778555b4a4       /mnt/ocfs2-storage
ocfs2   _netdev,x-systemd.requires=o2cb.service   0       0
```

где e42202df-a47b-42db-9b7d-e6778555b4a4 — UUID сетевого блочного устройства;
/mnt/ocfs2-storage — точка монтирования;
ocfs2 — тип файловой системы OCFS2;
_netdev,x-systemd.requires=o2cb.service — параметры монтирования, задающие монтирование после загрузки сетевых служб и при запущенной службе o2cb.

5.14.1.5. Централизованное управление в среде виртуализации

Для создания централизованного хранилища образов (pool, пул) необходимо на каждом сервере виртуализации, объединенном в кластер (astra1, astra2), выполнить добавление пула:

1) на узле astra1 войти в ОС под учетной записью администратора, входящего в группу libvirt-admin. Если в системе включено ролевое управление доступом в

среде виртуализации (см. 5.3), то необходимо выполнить настройки в соответствии с 5.3.1;

2) запустить `virt-manager`, в главном окне программы выбрать строку подключения «QEMU/KVM» и в меню выбрать «Правка — Свойства подключения»;

3) в окне «QEMU/KVM — сведения о подключении» перейти во вкладку «Пространство данных» и нажать кнопку **[+]** (**[Добавить пул]**);

4) в окне «Добавление пространства»:

а) в поле «Название» задать наименование пула, например «ocfs2-pool» (название пула на всех узлах может быть одинаковым для удобства);

б) в поле «Путь к цели» указать точку монтирования сетевого блочного устройства (на узле `astra1` для текущей конфигурации `/mnt/ocfs2-storage`);

в) нажать **[Готово]**;

5) проверить наличие подключенного хранилища — пул должен отображаться во вкладке «Пространство данных» в соответствии с рис. 5;

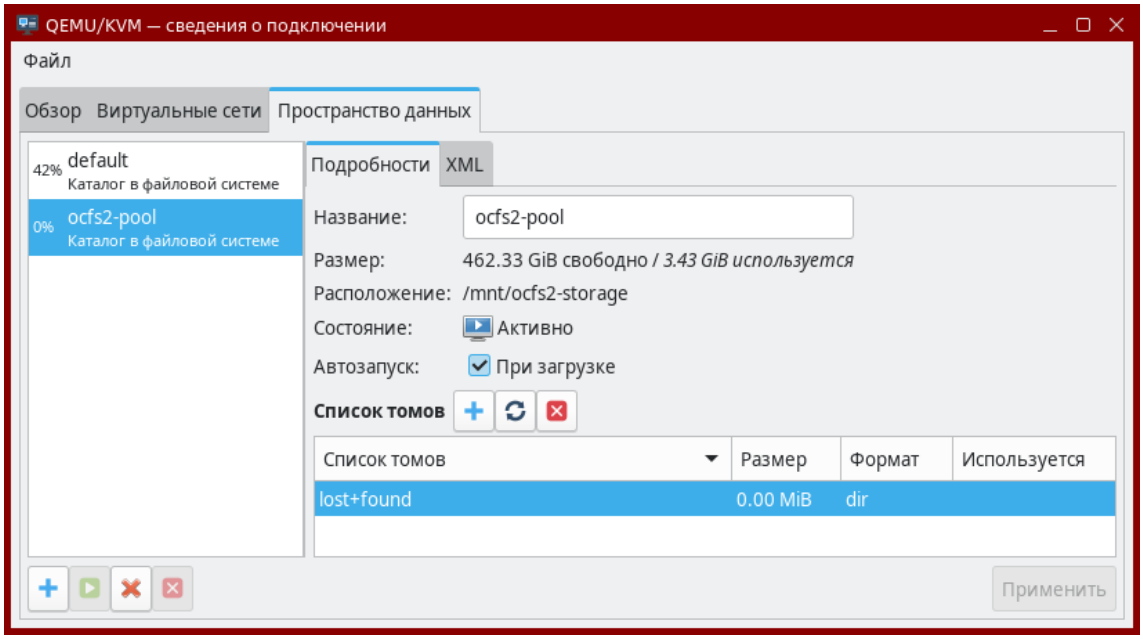

Рис. 5

Для проверки совместного использования пула необходимо:

1) на одном из узлов кластера (например, `astra2`) выполнить копирование образа ВМ в хранилище командой:

```
sudo cp ~/alse-vanilla-1.7.2-qemu-max-mg8.0.0.qcow2 /mnt/ocfs2-storage
```

В общем случае копирование выполняется командой вида:

```
sudo cp <путь_к_образу> <точка_монтирования_сетевого_блочного_устройства>
```

2) на каждом из узлов кластера проверить отображение образа во вкладке «Пространство данных» в списке томов (предварительно требуется нажать кнопку обновления списка томов) в соответствии с рис. 6.

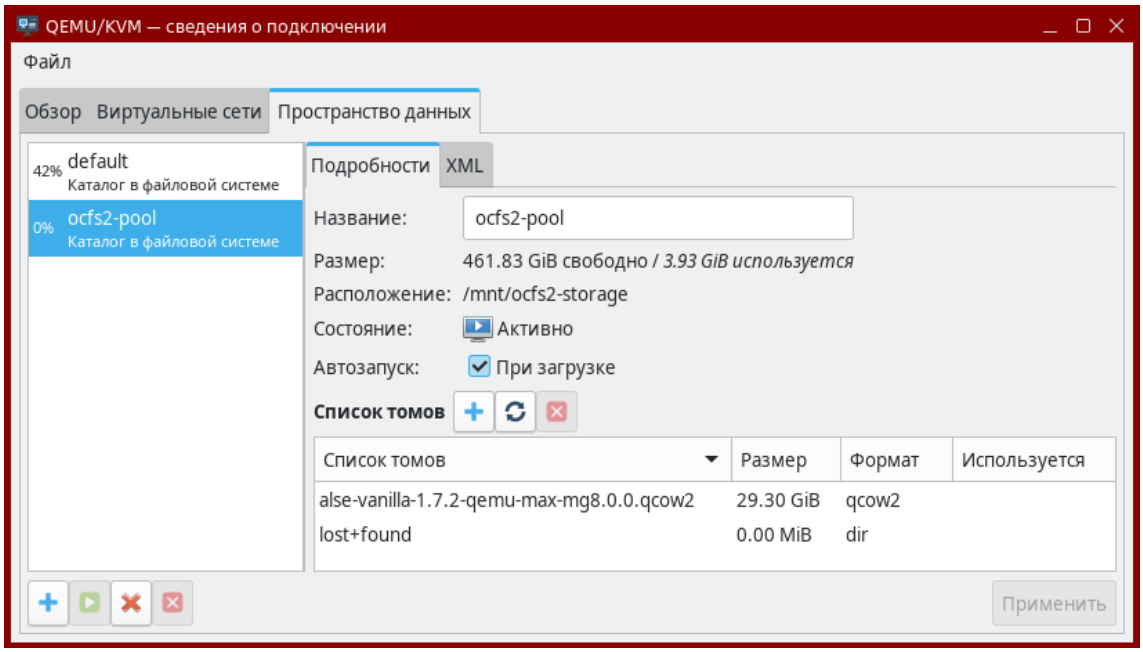

Рис. 6

5.14.2. Пример миграции виртуальных машин

Миграция запущенной ВМ с одного сервера виртуализации на другой сервер виртуализации возможна при использовании серверами виртуализации распределенного хранилища.

Миграция выполняется по SSH, поэтому предварительно на каждом сервере виртуализации должна быть настроена служба `ssh` (описание настройки `ssh` см. в РУСБ.10015-01 95 01-1) и на одном из серверов виртуализации должна быть создана ВМ.

На сервере виртуализации, с которого будет выполняться миграция запущенной ВМ (`astra1`), необходимо подключить по SSH второй сервер виртуализации (`astra2`). Для этого:

1) войти в ОС на узле `astra1` и запустить `virt-manager`;

2) в меню выбрать «Файл — Добавить соединение» и добавить подключение по SSH к узлу `astra2` в соответствии с рис. 7, указав в поле «Имя пользователя» учетную запись администратора узла `astra2`;

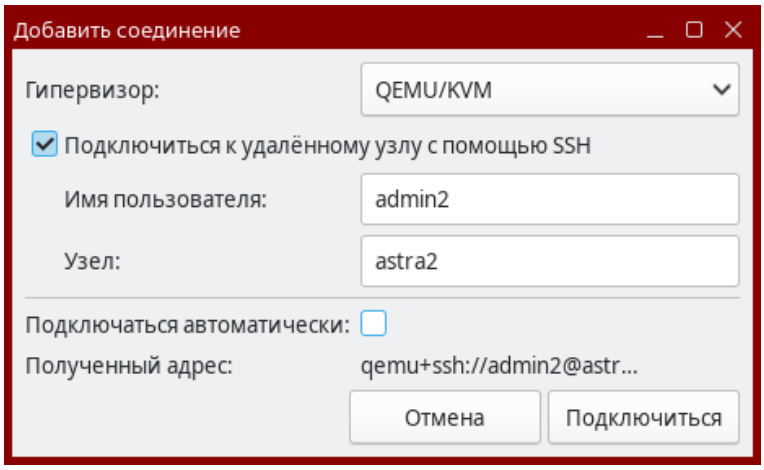

Рис. 7

3) в окне подтверждения соединения нажать **[Да]**;

4) в окне аутентификации ввести пароль от учетной записи администратора на узле `astra2`;

5) в главном окне `virt-manager` в списке соединений будет добавлено подключение к узлу `astra2` в соответствии с рис. 8.

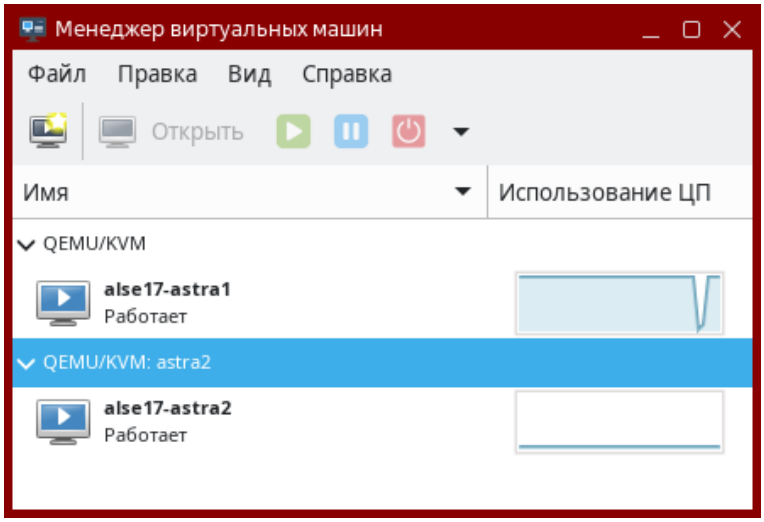

Рис. 8

Для создания ВМ на сервере виртуализации `astra1`, например путем импорта образа диска ВМ, добавленного в хранилище согласно 5.14.1.5, необходимо:

1) запустить `virt-manager`;

2) выбрать подключение «QEMU/KVM» и в меню выбрать «Файл — Создать виртуальную машину»;

3) в открывшемся окне «Новая виртуальная машина» выбрать «Импорт образа диска» и нажать **[Вперед]**;

4) на следующем шаге:

   а) в поле «Укажите путь к пространству хранения» указать в качестве источника образ ВМ в хранилище (например, `ocfs2-pool`);

б) в поле «Выберите операционную систему для установки» выбрать название операционной системы импортируемой ВМ в соответствии с рис. 9;

в) для перехода к следующему шагу нажать **[Вперед]**;

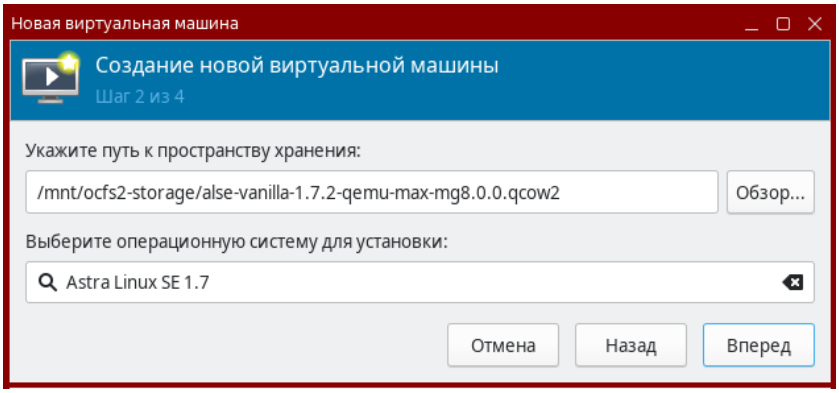

Рис. 9

5) на следующем шаге, при необходимости, задать значения памяти и количества процессоров, затем нажать **[Вперед]**;

6) на следующем шаге в поле «Название» указать имя ВМ, например `alse-astra1`. При необходимости проверить или изменить конфигурацию ВМ — установить флаг «Проверить конфигурацию перед установкой». Затем для окончания настройки, создания и запуска импортированной ВМ нажать **[Готово]**.

Для выполнения миграции запущенной ВМ с одного сервера виртуализации на другой сервер виртуализации необходимо в главном окне `virt-manager` открыть контекстное меню ВМ и выбрать «Миграция». В открывшемся окне из раскрывающегося списка «Новый узел» выбрать сервер виртуализации, на который необходимо переместить ВМ, и нажать **[Миграция]** в соответствии с рис. 10.

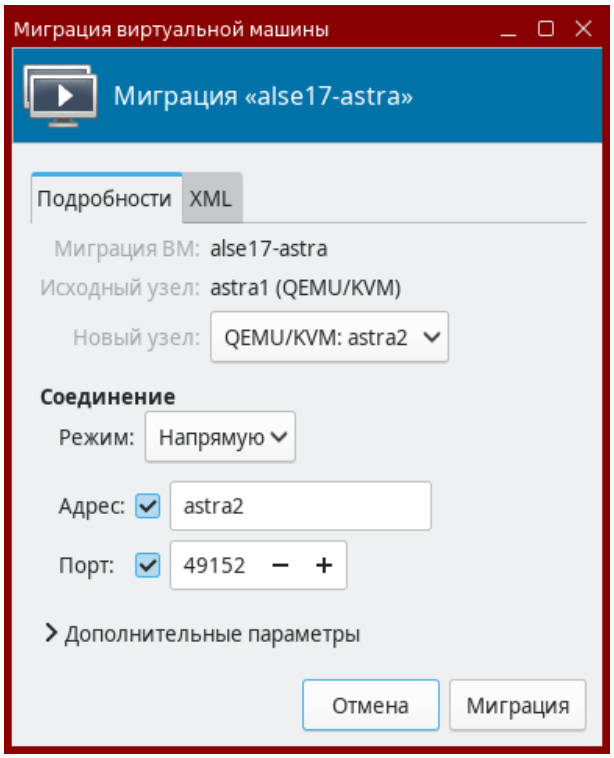

Рис. 10

В результате выполнения миграции виртуальная машина `alse-astra1` мигрирует с узла `astra1` на `astra2` (см. рис. 11).

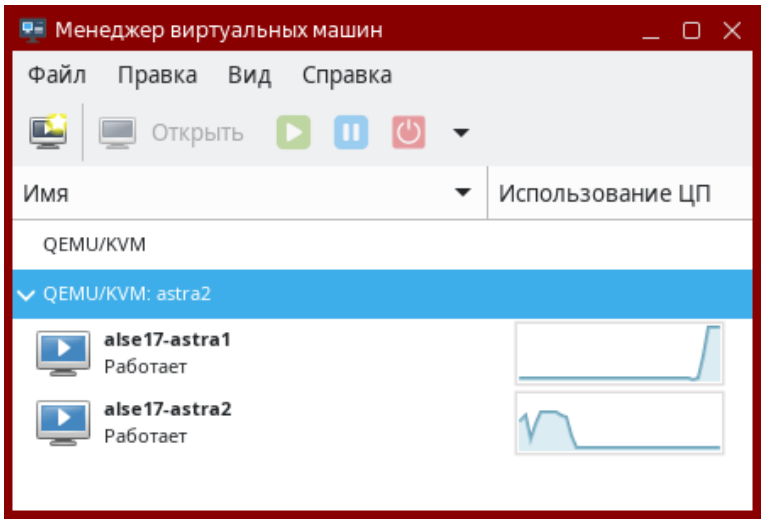

Рис. 11

6. РЕГИСТРАЦИЯ СОБЫТИЙ БЕЗОПАСНОСТИ

В ОС регистрация событий безопасности выполняется с учетом требований ГОСТ Р 59548-2022 «Защита информации. Регистрация событий безопасности. Требования к регистрируемой информации».

Регистрация событий безопасности реализуется использованием службы `auditd` и подсистемы регистрации событий из состава ОС, описание которой приведено в 6.5.

Служба `auditd` выполняет регистрацию событий объектов файловой системы (аудит файлов) и пользователей (аудит процессов) согласно заданным правилам. Работа с правилами аудита описана в 6.1 и 6.2. Регистрация событий осуществляется в журнал аудита, описание которого приведено в 6.3. Описание настройки параметров аудита приведено в 6.4.

Применение настроенных параметров аудита процессов осуществляется PAM-модулем `pam_parsec_aud`. По умолчанию регистрация событий аудита процессов включена в PAM-сценарии: `fly-dm`, `fly-dm-np`, `login`, `su`, `sshd`, `sumac.xauth`. Для регистрации событий аудита процессов пользователя, проходящего аутентификацию через другие PAM-сценарии, необходимо включить в соответствующие сценарии строку следующего вида:

```
session required pam_parsec_aud.so
```

В библиотеках подсистемы безопасности PARSEC реализован программный интерфейс для регистрации событий с использованием службы регистрации событий безопасности ОС, применяемый для регистрации событий в СУБД (описание приведено в РУСБ.10015-01 97 01-3) и комплексе программ электронной почты.

6.1. Правила регистрации событий

Регистрация событий осуществляется в соответствии с правилами аудита, правила делятся на два типа:

1) временные — действуют до перезагрузки системы. Данные правила относятся к подсистеме ядерного аудита: они задаются посредством инструмента `auditctl` и начинают выполняться при запуске службы `auditd`;

2) постоянные — действуют всегда, даже после перезагрузки системы. Такие правила задаются в файлах формата `.rules`, располагающихся в каталоге `/etc/audit/rules.d/`.

Подробное описание правил аудита, а также синтаксис использования инструмента `auditctl` приведены на справочной странице `man auditctl`.

Примеры:

1. Регистрировать все системные вызовы от процесса с идентификатором (pid) 1005:

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

2. Регистрировать все файлы, открытые пользователем аудита с идентификатором (auid) 510:

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

При добавлении постоянных правил аудита в файлах используется синтаксис инструмента auditctl без указания имени инструмента.

Пример

Регистрировать все системные вызовы от процесса с идентификатором (PID) 1005:

```
-a exit,always -S all -F pid=1005
```

Подсистема безопасности PARSEC предоставляет дополнительный способ выбора событий для регистрации — PARSEC-аудит. Он использует списки правил регистрации событий, назначаемые на процессы и файлы с помощью инструментов командной строки setfaud (см. 6.4.3), useraud (см. 6.4.4), psaud (см. 6.4.5).

Для работы PARSEC-аудита в файлах /etc/audit/rules.d/10-parsec.rules (аудит процессов и файлов) и /etc/audit/rules.d/10-parsec-nw.rules (сетевой аудит) заданы следующие постоянные правила:

1) правила аудита процессов (необходимы для работы useraud и psaud):

```
-a always,exit -F subj_type=psaud -F arch=<архитектура> -S \
     <системные_вызовы> -k parsec-p
```

где -a always,exit — регистрировать события в журнале, добавить правило в список exit (события, происходящие при выходе из системного вызова);
-F subj_type=psaud — обрабатывать правила, заданные с помощью useraud и psaud;
-F arch=<архитектура> — аппаратная платформа;
-S <системные_вызовы> — перехватывать события при вызовах, указанных в <системные_вызовы> (список вызовов, разделенных запятой);
-k parsec-p — присвоить метку фильтрации parsec-p событиям по данному правилу;

2) правила аудита файлов (необходимы для работы `getfaud` и `setfaud`):

```
-a always,exit -F obj_type=faud -F arch=<архитектура> -S \
     <системные_вызовы> -k parsec-f
```

где `-a always,exit` — регистрировать события в журнале, добавить правило в список `exit` (события, происходящие при выходе из системного вызова);
`-F obj_type=faud` — обрабатывать правила, заданные с помощью `setfaud`;
`-F arch=<архитектура>` —- аппаратная платформа;
`-S <системные_вызовы>` — перехватывать события при вызовах, указанных в `<системные_вызовы>` (список вызовов, разделенных запятой);
`-k parsec-f` — присвоить метку фильтрации `parsec-f` событиям по данному правилу;

3) правила сетевого аудита (анализируют работу системных вызовов, связанных со взаимодействием по сети):

```
-a always,exit -F subj_type=psaud -F arch=<архитектура> -S \
     <сетевые_системные_вызовы> -k parsec-p
```

где `-a always,exit` — регистрировать события в журнале, добавить правило в список `exit` (события, происходящие при выходе из системного вызова);
`-F subj_type=psaud` — обрабатывать правила, заданные с помощью `useraud` и `psaud`;
`-F arch=<архитектура>` — аппаратная платформа;
`-S <сетевые_системные_вызовы>` — перехватывать события при вызовах, указанных в `<сетевые_системные_вызовы>` (список вызовов, разделенных запятой);
`-k parsec-p` — присвоить метку фильтрации `parsec-p` событиям по данному правилу.

Включение и выключение правил PARSEC-аудита процессов и файлов выполняется с помощью инструмента командной строки `astra-audit-control` (описание приведено в 16.6.33). Включение и выключение правил сетевого PARSEC-аудита выполняется с помощью инструмента командной строки `astra-audit-network-control` (описание приведено в 16.6.34).

В дополнение к постоянным правилам PARSEC-аудита (файлов, процессов и сетевого) можно задавать собственные постоянные правила аудита. Правила рекомендуется добавлять в файл `/etc/audit/rules.d/audit.rules`. При необходимости возможно создать в каталоге `/etc/audit/rules.d/` новый файл с произвольным именем и расширением `*.rules` и задать в нем необходимые правила. Файл `/etc/audit/rules.d/audit.rules` можно редактировать вручную или с помощью графической утилиты графической утилиты

system-config-audit («Конфигурация аудита», описание утилиты приведено в электронной справке). Другие файлы правил можно редактировать только вручную.

6.2. Регистрация событий на основе меток безопасности

В ОС поддерживается регистрация событий на основе меток безопасности субъектов и объектов. Для этого используются параметры subj_type и obj_type и операции «=» (равно) и «!=» (не равно) с ними.

Метка имеет следующий вид (при этом любое из значений может быть пустым):

```
<Уровень_конфиденциальности>:<Категория_целостности>:
     <Категории_конфиденциальности>:<Дополнительные_мандатные_атрибуты>
```

Уровень конфиденциальности и категория целостности задаются в виде десятичных чисел, а категории конфиденциальности и дополнительные мандатные атрибуты — в виде шестнадцатеричных чисел без префикса 0x. Числовые значения дополнительных мандатных атрибутов приведены в 4.3.2.

Примеры:

1. Регистрация событий доступа к объектам каталога субъектами нулевого уровня конфиденциальности:

```
auditctl -a always,exit -S all -F dir=/var/level1_dir -F perm=rwxa -F \
    subj_type=0:::
```

2. Регистрация событий доступа к сущностям с категорией целостности 63 и дополнительным мандатным атрибутом ssi:

```
auditctl -a exit,always -S all -F obj_type=0:63:0:40
```

6.3. Журнал аудита

Служба auditd регистрирует события безопасности в журнале аудита /var/log/audit.

Также в журнале аудита /var/log/audit регистрируются действия с журналом событий подсистемы регистрации событий (см. 6.5) — удаление, переименование, перемещение, ротация файла журнала событий.

Для просмотра журнала аудита используется графическая утилита ksystemlog, описание утилиты приведено в электронной справке.

Действия с журналом аудита `/var/log/audit` (удаление, переименование, перемещение файла журнала аудита) регистрируются подсистемой регистрации событий и указываются в журнале событий (см. 6.5.3).

6.4. Средства управления аудитом

6.4.1. Графические утилиты

Для управления аудитом могут использоваться следующие графические утилиты (описание утилит доступно в электронной справке):

- `astra-systemsettings` («Параметры системы») — управление аудитом, привилегиями и мандатными атрибутами пользователей, работа с пользователями и группами;
- `system-config-audit` («Конфигурация аудита») — включение и выключение регистрации событий, настройка службы `auditd`, настройка журнала аудита, а также добавление, удаление и редактирование правил аудита;
- `fly-admin-events` («Настройка регистрации системных событий») — утилита из состава подсистемы регистрации событий (см. 6.5), с помощью которой возможно настроить регистрацию событий запуска и остановки службы `auditd`, регистрацию событий добавления и удаления правил `auditd`, регистрацию действий с журналом аудита. Дополнительно утилита позволяет добавлять правила аудита.

6.4.2. getfaud

Инструмент командной строки `getfaud` служит для получения списков правил регистрации событий файлов и каталогов.

Синтаксис команды:

```
getfaud [параметры] <файлы_и/или_каталоги>
```

Описание параметров `getfaud` приведено в таблице 31.

Таблица 31

Параметр	Описание
`-d`, `--default`	Вывести список регистрируемых событий по умолчанию для каталога. Данный список устанавливается на создаваемые в каталоге файлы и подкаталоги
`-R`, `--recursive`	Рекурсивно вывести список регистрируемых событий для подкаталогов
`-L`, `--logical`	Следовать по символическим ссылкам

Окончание таблицы 31

Параметр	Описание
`-P`, `--physical`	Не следовать по символическим ссылкам
`-n`, `--numeric`	Вывести флаги событий в числовом формате
`-l`, `--long`	Вывести полные имена флагов событий
`-p`, `--absolute-names`	Абсолютные имена
`-c`, `--omit-header`	Не показывать заголовок (имя файла)
`-s`, `--skip-empty`	Пропускать файлы с пустыми атрибутами
`-h, --help`	Вывести справку и выйти
`-v, --version`	Вывести информацию о версии и выйти

Следующие события доступны для регистрации:

1) `o`, `open` — открытие файла;
2) `c`, `create` — создание файла;
3) `x`, `exec` — исполнение файла;
4) `u`, `delete` — удаление файла (в каталоге);
5) `r`, `acl` — смена ACL;
6) `n`, `chown` — смена владельца;
7) `m`, `mac` — изменение метки;
8) `y`, `modify` — изменение файла;
9) `a`, `audit` — изменение списка регистрируемых событий файла;
10) `d`, `chmod` — изменение прав доступа к файлу.

Информация о списках посылается на стандартный вывод и может являться входными данными для инструмента `setfaud` (см. 6.4.3).

Описание инструмента приведено на справочной странице `man getfaud`.

6.4.3. setfaud

Инструмент командной строки `setfaud` позволяет устанавливать на файлы и каталоги списки правил регистрации событий. Правила могут быть переданы непосредственно (с помощью параметров `-s`, `-m`) или считаны из файлов (с помощью параметров `-S`, `-M`, `-B`). Файлы могут быть сформированы с помощью перенаправления вывода инструмента `getfaud` (см. 6.4.2).

Список регистрируемых событий приведен в 6.4.2. Описание инструмента и список регистрируемых событий также приведены на справочной странице `man setfaud`.

Синтаксис команды:

```
setfaud [параметры] [правила_регистрации] [файлы_и/или_каталоги]
```

Описание параметров `setfaud` приведено в таблице 32.

Таблица 32

Параметр	Описание
`-s, --set`	Установить список регистрируемых событий, передав его непосредственно в команде
`-b, --remove`	Удалить все элементы списка регистрируемых событий
`-m, --modify`	Назначить или изменить существующее правило регистрации событий
`-d, --default`	Изменить список регистрируемых событий по умолчанию для каталога. Данный список будет устанавливаться на создаваемые в каталоге объекты. Параметр всегда должен быть первым. Несовместим с параметром `-R`
`-S, --set-file`	Установить список регистрируемых событий из файла
`-X, --remove-all`	Удалить все списки регистрируемых событий
`-M, --modify-file`	Изменить или добавить элементы списка регистрируемых событий из файла
`-B, --restore`	Восстановить атрибуты из файла
`-R, --recursive`	Используется для каталогов. Назначить список регистрации событий рекурсивно
`-L, --logical`	Следовать по символическим ссылкам
`-P, --physical`	Не следовать по символическим ссылкам
`-h, --help`	Вывести справку и выйти
`-v, --version`	Вывести информацию о версии и выйти

Правила регистрации задаются в виде:

```
[u:<пользователь>:<флаги_регистрации>][,g:<группа>:<флаги_регистрации>]
     [,o:<флаги_регистрации>]
```

где `<пользователь>` и `<группа>` — символические или численные идентификаторы пользователя и группы;

`u:` — правила для пользователя;

`g:` — правила для группы;

`o:` — правила для всех остальных.

Флаги регистрации задаются в виде:

```
<флаги_успешных_событий>[:<флаги_неуспешных_событий>]
```

ВНИМАНИЕ! Если в команде указаны только флаги успешных событий, то они также будут применены и к неуспешным событиям.

Можно указать один или несколько флагов событий без разделителей и пробелов между ними (как успешных, так и неуспешных), при этом они могут иметь один из следующих видов:

1) флаг добавления или отключения регистрации события:

```
<+|-><имя_события>
```

Пример

Установка списка правил регистрации успешного события выполнения файла, регистрации неуспешного удаления файла и удаление правила регистрации неуспешного открытия файла пользователем `user`:

```
setfaud -m u:user:+exec:+delete-open start.sh
```

2) флаг только добавления регистрации события:

```
<сокращенное_имя_события>
```

Примеры:

1. Регистрация успешных и неуспешных событий открытия и удаления файла (и никаких других) пользователями группы `group`:

```
setfaud -s g:group:ou file.pdf
```

2. Регистрация всех успешных и неуспешных событий (кроме изменения) с файлом, выполняемых остальными пользователями:

```
setfaud -m o:oxudnarmc:oxudnarmc file.pdf
```

Для каталогов, наряду с обычным списком правил регистрации событий, возможно установить список правил регистрации событий по умолчанию. Если на каталог установлен список правил регистрации по умолчанию, то на все файлы и дочерние каталоги, которые будут созданы в данном каталоге, будут установлены списки регистрации событий из списка по умолчанию.

Для внесения изменений в список по умолчанию используется параметр `-d`. Данный параметр взаимоисключаем с параметром `-R`. При использовании параметра `-R` изменяются рекурсивно списки правил регистрации событий для каталога и всех уже существующих файлов и дочерних каталогов в нем. При использовании нескольких параметров параметр `-d` должен быть первым (`-ds` — правильно, `-sd` — не будет работать).

Примеры:

1. Регистрация успешных и неуспешных событий удаления файлов и дочерних каталогов в каталоге `/opt/test`, выполняемых пользователем `username`. Правила регистрации будут назначаться всем объектам, создаваемым в `/opt/test`. Правила для уже существующих объектов изменены не будут.

```
setfaud -ds u:username:+delete:+delete /opt/test
```

2. Регистрация успешных и неуспешных событий запуска файлов на исполнение в каталоге `/opt/test`, выполняемых пользователем `username`. Правила регистрации будут назначены данному каталогу и уже существующим объектам в данном каталоге. Создаваемые в каталоге `/opt/test` объекты не будут иметь назначенных правил регистрации.

```
setfaud -Rs u:username:+exec:+exec /opt/test
```

Для отключения регистрации всех событий в качестве флага регистрации событий используется `0`.

Пример

Отключение регистрации всех успешных и неуспешных событий действий пользователя `user` с файлом:

```
setfaud -m u:user:0:0 file.pdf
```

При необходимости добавления или удаления правил регистрации событий одного типа, не затрагивая существующие правила для событий другого типа, следует для неизменяемых правил в качестве флага регистрации событий использовать `+0`.

Примеры:

1. Отключение регистрации неуспешного события удаления файла пользователем `user`, не меняя правила регистрации успешных событий:

```
setfaud -m u:user:+0:-delete file.pdf
```

2. Отключение регистрации успешного события удаления файла пользователем `user`, не меняя правила регистрации неуспешных событий:

```
setfaud -m u:user:-delete:+0 file.pdf
```

Для сброса назначенных на файлы/каталоги правил регистрации событий используется команда:

```
setfaud -b <файлы_и/или_каталоги>
```

6.4.4. useraud

Инструмент командной строки `useraud` позволяет назначать, просматривать и изменять правила регистрации событий для пользователей и групп пользователей.

При назначении правил регистрации событий создаются в каталоге `/etc/parsec/auddb/` файлы с именами вида `user:<UID>`, `group:<GID>` и `other:0` для правил пользователей, групп и всех остальных, соответственно. Если пользователю не назначены индивидуальные правила регистрации событий (отсутствует файл вида `/etc/parsec/auddb/user:<UID>`), то для него применяются правила регистрации для группы. Если они также отсутствуют, то применяются правила регистрации событий для остальных пользователей (при их наличии).

Список регистрируемых событий приведен в 6.4.2. Описание инструмента и список регистрируемых событий также приведены на справочной странице `man useraud`.

Синтаксис команды:

```
useraud [параметр] [имя_пользователя(группы)] [--] [флаги_регистрации_событий]
```

Описание параметров инструмента `useraud` приведено в таблице 33.

Таблица 33

Параметр	Описание
`-d, --delete`	Сбросить правила регистрации событий
`-n, --numeric`	Вывести флаги событий в числовом формате
`-l, --long`	Вывести полные имена флагов событий
`-g, --group`	Правила регистрации событий для группы
`-o, --other`	Правила регистрации событий для остальных (любой пользователь)
`-m, --modify`	Назначить или изменить существующее правило регистрации событий
`-h, --help`	Вывести справку и выйти

Окончание таблицы 33

Параметр	Описание
`-v, --version`	Вывести информацию о версии и выйти

Для просмотра индивидуальных правил регистрации событий, назначенных локальному пользователю, используется команда:

```
useraud <имя_пользователя>
```

Для просмотра правил регистрации событий, назначенных одной или нескольким группам, используется команда:

```
useraud -g [имя_группы]
```

Если имя группы не задано, выводятся все группы с назначенными правилами.

Если правила регистрации событий для пользователя или группы не назначены (отсутствует соответствующий файл настроек в каталоге `/etc/parsec/auddb`), то при попытке их просмотра выводится сообщение:

```
useraud: невозможно прочесть ф̧айл настроек аудита: Нет такого ф̧айла или каталога
```

Для просмотра правил регистрации событий, назначенных всем остальным, используется команда:

```
useraud -o
```

Флаги регистрации событий задаются в виде:

```
<ф̧лаги_успешных_операций>[:<ф̧лаги_неуспешных_операций>]
```

ВНИМАНИЕ! Если в команде указаны только флаги успешных событий, то они также будут применены и к неуспешным событиям.

Пример

При добавлении флага регистрации успешного события выполнения файла:

```
useraud -m user1 +exec
```

также будет добавлен флаг регистрации неуспешного события выполнения файла.

Можно указать один или несколько флагов событий без разделителей и пробелов между ними (как успешных, так и неуспешных), при этом они могут иметь один из следующих видов:

1) флаг добавления или отключения регистрации события:

<+|-><имя_события>

Если первый флаг события в строке отключает регистрацию, то перед флагами необходимо использовать символы «--».

Примеры:

1. Регистрация успешного события выполнения файла и отключение регистрации неуспешного события открытия:

```
useraud -m user1 +exec:-open
```

2. Отключение регистрации успешного и неуспешного событий открытия файла, включение регистрации успешного события выполнения файла:

```
useraud -m user2 -- -open+exec:-open
```

2) флаг только добавления регистрации события:

<сокращенное_имя_события>

Пример

Регистрация успешных событий удаления и создания файла и неуспешных событий выполнения файла:

```
useraud -m user3 uc:x
```

Для отключения регистрации всех событий для пользователя/группы в качестве флага регистрации событий используется 0.

Пример

Отключение регистрации успешных и неуспешных событий для группы:

```
useraud -m -g group1 0:0
```

При необходимости добавления или удаления правил регистрации событий одного типа, не затрагивая существующие правила для событий другого типа, следует для неизменяемых правил в качестве флага регистрации событий использовать +0.

Примеры:

1. Отключение регистрации неуспешного события удаления файлов и/или каталогов, не меняя правил регистрации успешных событий:

```
useraud -m user1 +0:-delete
```

2. Отключение регистрации всех успешных событий для пользователя, не меняя правила регистрации неуспешных событий:

```
useraud -m user2 0:+0
```

Для сброса назначенных пользователю правил регистрации событий используется команда:

```
useraud -d <имя_пользователя>
```

В результате выполнения данной команды удаляется файл настроек регистрации /etc/parsec/auddb/user:<UID>, и к пользователю начинают применяться настройки регистрации событий для групп или для всех остальных (при их наличии). Удаление файла настроек вручную аналогично выполнению данной команды.

ВНИМАНИЕ! Отключение регистрации событий для пользователя командой:

```
useraud -m <имя_пользователя> 0:0
```

не приводит к удалению файла /etc/parsec/auddb/user:<UID>, поэтому к пользователю не будут применяться имеющиеся настройки для групп или всех остальных.

Изменения правил регистрации событий применяются к пользователю при его новом входе в систему.

6.4.5. psaud

Правила регистрации событий наследуются порожденными процессами. Инструмент командной строки psaud позволяет изменить или считать правила регистрации событий указанного процесса.

Список регистрируемых событий приведен в 6.4.2. Описание инструмента и список регистрируемых событий также приведены на справочной странице man psaud.

Синтаксис команды:

```
psaud [параметры] <идентификатор_процесса> [--] [флаги_регистрации]
```

Описание параметров `psaud` приведено в таблице 34.

Таблица 34

Параметр	Описание
`-d, --delete`	Сбросить правила регистрации событий
`-n, --numeric`	Выводить информацию о правилах регистрации событий в числовом формате
`-l, --long`	Выводить информацию о правилах регистрации событий в полной форме
`-h, --help`	Вывести справку и выйти
`--version`	Вывести информацию о версии и выйти

Если в команде указаны флаги регистрации, то инструмент устанавливает указанные флаги регистрации событий на процесс. Если флаги регистрации событий не указаны, то инструмент считывает их с процесса, заданного параметром (идентификатор процесса).

Флаги регистрации задаются в виде:

```
<флаги_успешных_событий>[:<флаги_неуспешных_событий>]
```

ВНИМАНИЕ! Если в команде указаны только флаги успешных событий, то они также будут применены и к неуспешным событиям.

Можно указать один или несколько флагов событий без разделителей и пробелов между ними (как успешных, так и неуспешных), при этом они могут иметь один из следующих видов:

1) флаг добавления или отключения регистрации события:

```
<+|-><имя_события>
```

Если первый флаг события в строке отключает регистрацию, то перед флагами необходимо использовать символы «`--`».

Примеры:

1. Регистрация успешного события выполнения файла:

```
psaud 10234 +exec
```

2. Отключение регистрации успешного события открытия файла и включение регистрации успешного события выполнения файла:

```
psaud 10234 -- -open+exec
```

2) флаг только добавления регистрации события:

```
<сокращенное_имя_события>
```

Пример

Регистрация успешных событий открытия и удаления файла:

```
psaud 10234 ou
```

Для отключения регистрации всех событий для процесса в качестве флага регистрации событий используется `0`.

Пример

Отключение регистрации успешных и неуспешных событий для процесса:

```
psaud 10234 0:0
```

При необходимости добавления или удаления правил регистрации событий одного типа, не затрагивая существующие правила для событий другого типа, следует для неизменяемых правил в качестве флага регистрации событий использовать `+0`.

Примеры:

1. Отключение регистрации неуспешного события удаления для процесса, не меняя правил регистрации успешных событий:

```
psaud 10234 +0:-delete
```

2. Отключение регистрации успешного события удаления для процесса, не меняя правила регистрации неуспешных событий:

```
psaud 10234 -- -delete:+0
```

Для сброса назначенных процессу правил регистрации событий используется команда:

```
psaud -d <идентификатор_процесса>
```

6.4.6. ausearch

Инструмент командной строки `ausearch` предназначен для просмотра файлов журнала регистрации событий ядра, а также событий пользователя.

Синтаксис команды:

```
ausearch [параметры]
```

Описание параметров `ausearch` приведено в таблице 35.

Таблица 35

Параметр	Описание
`-a, --event`	Поиск событий с заданным идентификатором события. Сообщения обычно начинаются с записи вида `msg=audit(1116360555.329:2401771)`. Идентификатор события — число после «:». Все события аудита, связанные с одним системным вызовом, имеют одинаковый идентификатор
`-c, --comm`	Искать события с заданным именем исполняемого файла задачи
`--debug`	Вывести информацию о некорректных событиях, не попавших в стандартный вывод ошибок
`-e, --exit`	Искать события по заданному коду завершения системного вызова или коду последней ошибки
`-f, --file`	Искать события с заданным именем файла
`-ga, --gid-all`	Искать события с заданным эффективным или обычным идентификатором группы
`-ge, --gid-effective`	Искать события с заданным эффективным идентификатором группы или именем группы
`-gi, --gid`	Искать события с заданным идентификатором группы или именем группы
`-h, --help`	Вывести справку и выйти
`-hn, --host`	Искать события с заданным именем узла. Имя узла может быть именем узла, полным доменным именем или сетевым адресом
`-i, --interpret`	Транслировать числовые значения в текстовые. Например, идентификатор пользователя будет транслирован в имя пользователя. Трансляция выполняется с использованием данных с той машины, где запущен `ausearch`
`-if, --input`	Использовать указанный файл или каталог вместо журнала аудита. Может быть полезно при анализе журнала аудита с другой машины или при анализе частично сохраненного журнала
`--input-logs`	Использовать расположение файла журнала, указанное в `auditd.conf`, в качестве входных данных для поиска. Это необходимо при использовании `ausearch` в задаче `cron`
`--just-one`	Искать до первого совпадения

Продолжение таблицы 35

Параметр	Описание
-k, --key	Искать события с заданным ключевым словом
-l, --line-buffered	Очистить вывод каждой строки. Может быть полезно, когда стандартный вывод конвейерно перенаправляется и использование стратегии буферизации блоков по умолчанию нежелательно
-m, --message	Искать события с заданным типом. Возможно указать список значений, разделенных запятыми. Для получения всех сообщений системы аудита указать тип ALL. Если указать данный параметр без значения, будет выведен список допустимых типов. Тип сообщения может быть строкой или числом
-n, --node	Вывести события с указанной машины. Возможно указать несколько машин, тогда выводятся события для каждой из указанных машин
-o, --object	Искать события с заданным контекстом (объектом)
-p, --pid	Искать события с заданным идентификатором процесса
-pp, --ppid	Искать события с заданным идентификатором родительского процесса
-r, --raw	Необработанный вывод. Используется для извлечения записей для дальнейшего анализа
-sc, --syscall	Искать события с заданным системным вызовом. Можно указать его номер или имя. Если указано имя, оно будет проверено на машине, где запущен ausearch
-se, --context	Искать события с заданным контекстом (scontext/subject или tcontext/object)
-su, --subject	Искать события с заданным контекстом scontext (subject)
-sv, --success	Искать события с заданным флагом успешного выполнения. Допустимые значения: yes (успешно) и no (неуспешно)
-te, --end	Искать события, которые произошли в указанное время или раньше. Формат даты и времени зависит от региональных настроек. Если дата не указана, то подразумевается текущий день (today). Если не указано время, то подразумевается текущий момент (now). Используется 24-часовая нотация времени. Например, дата может быть задана как 10/24/2005, а время — 18:00:00
-ts, --start	Искать события, которые произошли в указанное время или позже. Формат даты и времени зависит от региональных настроек. Если дата не указана, то подразумевается текущий день (today). Если не указано время, то подразумевается полночь (midnight). Используется 24-часовая нотация времени. Например, дата может быть задана как 10/24/2005, а время — 18:00:00
-tm, --terminal	Искать события с заданным терминалом. Некоторые службы (например, cron и atd) используют имя службы как имя терминала
-ua, --uid-all	Искать события, у которых идентификатор пользователя, эффективный идентификатор пользователя или loginuid (auid) совпадают с заданным идентификатором пользователя
-ue, --uid-effective	Искать события с заданным эффективным идентификатором пользователя

Окончание таблицы 35

Параметр	Описание
`-ui, --uid`	Искать события с заданным идентификатором пользователя
`-ul, --loginuid`	Искать события с заданным идентификатором пользователя. Все программы, которые его используют, должны использовать `pam_loginuid`
`-uu, --uuid`	Искать события с заданным `uuid`
`-v, --version`	Показать версию и выйти
`-vm, --vm-name`	Совпадение с полным словом. Поддерживается для имени файла, имени узла, терминала и контекста
`-w, --word`	Поиск по заданной строке с полным совпадением. Поддерживается для имени файла, имени узла, терминала и контекста
`-x, --executable`	Искать события с заданным именем исполняемой программы

Описание инструмента приведено на справочной странице `man ausearch`.

Примеры:

1. Отобразить события процессов и пользователей

```
ausearch -i -ts "11:35:35" -te "11:36:00" -k parsec-p
```

2. Регистрация событий с файлами

```
ausearch -i -ts "11:35:35" -te "11:36:00" -k parsec-f
```

3. Протоколирование событий, связанных с мандатным управлением доступом и аудитом

```
ausearch -i -ts "11:35:35" -te "11:36:00" -m avc
```

4. Протоколирование событий, приходящих от пользователя

```
ausearch -i -ts "11:35:35" -te "11:36:00" -m user_avc
```

6.4.7. Параметры регистрации событий

При необходимости возможно отключить регистрацию событий в журнал, но правила PARSEC-аудита все равно продолжат обрабатываться и загружать ресурсы ОС.

Отключение регистрации набора системных вызовов возможно выполнить одним из следующих способов:

1) отключение регистрации системных вызовов, не используемых для мандатного управления доступом — выполнить команду:

```
echo 1 > /parsecfs/disable-non-mac-audit
```

Для проверки состояния регистрации выполнить команду:

```
cat /parsecfs/disable-non-mac-audit
```

Если вывод команды равен 1, то регистрация системных вызовов, не используемых для мандатного управления доступом, отключена;

2) отключение регистрации всех системных вызовов — выполнить команду:

```
echo 1 > /parsecfs/disable-all-audit
```

Для проверки состояния регистрации выполнить команду:

```
cat /parsecfs/disable-all-audit
```

Если вывод команды равен 1, то регистрация всех системных вызовов отключена;

3) отключение регистрации запретов доступа (на основе мандатного управления доступом) — выполнить команду:

```
echo 1 > /parsecfs/disable-denied-audit
```

Для проверки состояния регистрации выполнить команду:

```
cat /parsecfs/disable-denied-audit
```

Если вывод команды равен 1, то регистрация запретов доступа отключена.

Для оптимизации работы и уменьшения нагрузки на ОС рекомендуется выполнять отключение регистрации событий с помощью инструмента командной строки `astra-audit-control` (описание приведено в 16.6.33). В таком случае будет отключена как регистрация событий в журнал, так и сама обработка правил PARSEC-аудита.

6.5. Подсистема регистрации событий

6.5.1. Регистрация событий и уведомление о событиях

В ОС реализована подсистема регистрации событий, которая собирает информацию о событиях, в том числе о событиях безопасности, из различных источников и предоставляет инструменты для просмотра собранных данных и реагирования на события.

Подсистема регистрации событий включает следующие основные компоненты:

1) менеджер и маршрутизатор событий `syslog-ng` — основная служба подсистемы регистрации событий, которая реализована в соответствии со стандартом Syslog. Служба `syslog-ng` принимает информацию о событиях из различных источников (события от `auditd`, собственные подключаемые модули, файлы, прикладное ПО и др.), выполняет фильтрацию и обработку полученных данных, регистрирует события в журнал, а также, в зависимости от конфигурации, сохраняет в файл, отправляет по сети и т.д.;

2) модуль `syslog-ng-mod-astra` — модуль для `syslog-ng`, выполняющий дополнительную обработку и фильтрацию событий;

3) `astra-event-watcher` — службы уведомления пользователя о событиях, обработанных менеджером `syslog-ng`;

4) `astra-event-diagnostics` — инструмент диагностики подсистемы регистрации событий, описание инструмента приведено в 6.5.4.

Работа модуля `syslog-ng-mod-astra` настраивается в следующих конфигурационных файлах:

1) `/etc/astra-syslog.conf` — список регистрируемых событий;

2) `/usr/share/syslog-ng-mod-astra/event-settings/` — каталог с файлами настроек по умолчанию для каждого события.

Для настройки регистрации событий и регистрируемых параметров событий используются:

1) графическая утилита `fly-admin-events` («Настройка регистрации системных событий»), описание утилиты приведено в электронной справке;

2) инструмент командной строки `astra-admin-events`.

Для просмотра с помощью инструмента `astra-admin-events` списка регистрируемых событий необходимо выполнить команду без параметров:

```
astra-admin-events
```

В выводе списка событий значение «`1`» — событие регистрируется, значение «`0`» — событие не регистрируется. Порядок использования инструмента приведен на странице помощи:

```
astra-admin-events -h
```

Модуль `syslog-ng-mod-astra` информацию о событиях регистрирует в следующих файлах:

1) `/parsec/log/astra/events` — журнал событий в формате JSON, описание журнала приведено в 6.5.3;

2) `/var/log/astra/prevlogin-<имя_пользователя>` — журнал в формате JSON сводной статистики предыдущих входов в систему пользователя `<имя_пользователя>`. Включает данные о последней завершенной сессии данного пользователя, а также количество успешных и неуспешных входов данного пользователя со времени начала ведения статистики. Доступен для чтения только пользователю `<имя_пользователя>`.

Параметры уведомлений службы `astra-event-watcher` определены в файле `/usr/share/knotifications5/astra-event-watcher.notifyrc`. Передача данных выполняется с использованием D-Bus. Реализация действий осуществляется модулем KNotifications. Служба получает информацию только о событиях, для которых настроено реагирование. Служба, запущенная от имени определенного пользователя, реагирует только на те события, которые настроены для данного пользователя. Для этого служба отслеживает сигнал, передаваемый `syslog-ng` по D-Bus всем заинтересованным приложениям. В сигнале также передается список имен пользователей и имен групп, для которых требуется реакция на событие.

Параметры уведомлений для определенного пользователя могут быть переопределены в файле `/home/<имя_пользователя>/.config/astra-event-watcher.notifyrc`.

Информирование (оповещение) о событиях безопасности осуществляется с помощью графической утилиты `fly-notifications` («Центр уведомлений»), описание утилиты приведено в электронной справке.

6.5.2. Конфигурационный файл службы syslog-ng

Работа службы `syslog-ng` настраивается в конфигурационном файле по умолчанию `/etc/syslog-ng/syslog-ng.conf` и в пользовательских конфигурационных файлах вида `/etc/syslog-ng/conf.d/<имя_файла>.conf`. Описание структуры и синтаксиса конфигурационного файла приведено на справочной странице `man syslog-ng.conf`.

В конфигурационном файле `/etc/syslog-ng/syslog-ng.conf` не допускается удалять или отключать комментированием следующие параметры:

1) источники по умолчанию:

```
source s_src {
    system();
    internal();
};
```

2) инструкции `@include` для включения других конфигурационных файлов;

3) объекты `filter`.

Без указания в конфигурационном файле любого из данных элементов служба `syslog-ng` не сможет запуститься. Это приведет к невозможности работы подсистемы регистрации событий.

При необходимости исключить стандартные фильтры регистрации событий можно закомментировать их непосредственно внутри цепочек `log` или закомментировать полностью использующие их цепочки `log`.

6.5.3. Журнал событий

Служба `syslog-ng` выполняет регистрацию событий в журнал `/parsec/log/astra/events`. В журнале событий регистрируются попытки запуска неподписанных файлов, успешная и неуспешная аутентификация, успешная и неуспешная авторизация, данные о пользовательских сессиях и другие события безопасности, в соответствии с настройками регистрации событий (см. 6.5.1). Также в журнале событий регистрируются действия с журналом аудита `/var/log/audit` службы `auditd` (см. 6.3) — удаление, переименование, перемещение файла журнала аудита.

Каждая запись в журнале событий состоит из набора параметров события, для каждого параметра указано его значение.

Файл журнала событий доступен для чтения только пользователям из группы `astra-audit`.

Файл журнала событий доступен для чтения и записи в конец файла только администратору. Разрешение на запись в конец файла задано установкой на файл атрибута `a` (с помощью `chattr`).

При включенном МКЦ файл журнала событий имеет высокую метку целостности. Процесс, осуществляющий регистрацию событий в файле журнала, также имеет высокую метку целостности. Таким образом, изменение файла журнала событий доступно только администратору с высокой меткой целостности.

Для просмотра журнала событий может использоваться:

1) графическая утилита `fly-event-viewer` («Журнал системных событий»), описание утилиты приведено в электронной справке;

2) инструмент командной строки `astra-event-viewer`, порядок использования инструмента приведен на странице помощи:

```
astra-event-viewer -h
```

Как инструмент командной строки, так и графическая утилита по умолчанию отображают только часть значений параметров событий, записанных в журнале событий `/parsec/log/astra/events`.

Для вывода всех значений параметров событий инструментом командной строки `astra-event-viewer` выполнить:

```
astra-event-viewer --detailed
```

При этом будут выведены все значения параметров событий, записанные в журнале событий `/parsec/log/astra/events`.

В графической утилите `fly-event-viewer` записи из журнала событий `/parsec/log/astra/events` отображаются в виде таблицы, где строки соответствуют событиям, а столбцы — их параметрам (имя события, тип события, критичность и другие). Параметры событий, для которых не предусмотрены отдельные столбцы, отображаются в столбце «Нераспознанная информация». Чтобы добавить столбцы для параметров событий, необходимо:

1) в каталоге `/etc/fly-event-viewer/emp/` создать конфигурационный файл с произвольным именем и расширением `*.conf` (если путь не существует, то предварительно создать соответствующие каталоги);

2) для каждого параметра добавить в созданный конфигурационный файл запись в формате JSON вида:

```
[
{ "name": "<имя_параметра>", "English": "<описание параметра на
     английском языке>", "Russian": "<описание параметра на
     русском языке>" }
]
```

Пример

```
[
{ "name": "domainvm", "English": "Virtual machine ID", "Russian":
     "Идентификатор виртуальной машины" },
{ "name": "user", "English": "User", "Russian": "Пользователь" },
{ "name": "file", "English": "File", "Russian": "Файл" },
{ "name": "hostip", "English": "Hypervisor network address",
     "Russian": "Сетевой адрес гипервизора" }
]
```

ВНИМАНИЕ! Указанное имя параметра должно точно соответствовать имени параметра в журнале событий.

Действия с журналом событий (удаление, переименование, перемещение, ротация файла журнала событий) регистрируются подсистемой регистрации событий и указываются первой записью в журнале событий:

- удаление журнала событий регистрируется событием «Журнал событий удален»;
- переименование или перемещение журнала событий регистрируется событием «Журнал событий переименован или перемещен»;
- ротация журнала событий регистрируется событием «Журнал событий ротирован»;
- действия с журналом событий недоверенными процессами (всеми процессами, кроме процессов `syslog-ng` и `logrotate`) регистрируются событием «Журнал событий изменен недоверенным процессом».

Также действия с журналом событий регистрируются службой `auditd` и указываются в журнале аудита `/var/log/audit` (см. 6.3).

6.5.4. Самодиагностика подсистемы регистрации событий

Инструмент `astra-event-diagnostics` обеспечивает самодиагностику подсистемы регистрации событий.

Самодиагностика выполняется:

1) автоматически при загрузке ОС после запуска служб `syslog-ng` и `auditd`;
2) периодически в процессе функционирования ОС, периодичность самодиагностики настраивается в юните `astra-event-diagnostics.timer`.

Также провести диагностику возможно, выполнив команду:

```
sudo astra-event-diagnostics -v
```

При выполнении самодиагностики производятся следующие проверки:

1) запуск и работоспособность служб `syslog-ng` и `auditd`;
2) наличие необходимых модулей из состава `syslog-ng-mod-astra`;
3) корректность конфигурационных файлов:
   а) `/etc/audit/rules.d/astra-syslog.rules`;
   б) `/etc/astra-syslog.conf`;
   в) `/etc/syslog-ng/conf.d/mod-astra.conf`;
   г) некоторых из `/usr/share/syslog-ng-mod-astra/event-settings/`.

Возможно настроить автоматическое блокирование учетной записи пользователя, если в результате самодиагностики выявлены ошибки. Для этого необходимо запустить команду самодиагностики с параметром `-b`:

```
sudo astra-event-diagnostics -b
```

Описание параметров команды приведено на странице помощи:

```
sudo astra-event-diagnostics -h
```

Разблокировка учетной записи, заблокированной после выявления ошибок при самодиагностике, может быть выполнена с использованием модуля «Пользователи» в разделе «Пользователи и группы» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_users
```

Регистрация событий о результатах самодиагностики осуществляется:

1) в журнал `/parsec/log/astra/astra-event-diagnostics`;
2) в журнал `/var/log/syslog`;
3) в журнал событий `/parsec/log/astra/events` (в случае, если подсистема регистрации событий частично работоспособна);
4) с использованием `astra-sosreport`.

Если в результате самодиагностики были выявлены ошибки в работе подсистемы регистрации событий, то события о результатах регистрируются с уровнем важности «Критический».

Уведомление о результатах самодиагностики обеспечивает служба `astra-event-watcher`.

Информирование (оповещение) о событиях осуществляется с помощью утилиты `fly-notifications` («Центр уведомлений»). Описание утилиты приведено в электронной справке.

6.6. Средства централизованного аудита и протоколирования

В состав ОС входит программное средство Zabbix, обеспечивающее функционал для настройки, сбора данных и мониторинга состояния сети, а также жизнеспособности и целостности ресурсов сети. Описание установки и настройки Zabbix приведено в документе РУСБ.10015-01 95 01-1.

7. ИЗОЛЯЦИЯ ПРОЦЕССОВ

Ядро ОС обеспечивает для каждого процесса в системе собственное изолированное адресное пространство. Данный механизм изоляции основан на страничном механизме защиты памяти, а также механизме трансляции виртуального адреса в физический, поддерживаемый модулем управления памятью. Одни и те же виртуальные адреса (с которыми и работает процессор) преобразуются в разные физические для разных адресных пространств. Процесс не может несанкционированным образом получить доступ к пространству другого процесса, т. к. непривилегированный пользовательский процесс лишен возможности работать с физической памятью напрямую.

Механизм разделяемой памяти является санкционированным способом получить нескольким процессам доступ к одному и тому же участку памяти и находится под контролем дискреционных и мандатных ПРД.

Адресное пространство ядра защищено от прямого воздействия пользовательских процессов с использованием механизма страничной защиты. Страницы пространства ядра являются привилегированными и доступ к ним из непривилегированного кода вызывает исключение процессора, которое обрабатывается корректным образом ядром ОС. Единственным санкционированным способом доступа к ядру ОС из пользовательской программы является механизм системных вызовов, который гарантирует возможность выполнения пользователем только санкционированных действий.

8. СОЗДАНИЕ И ЗАЩИТА ИЗОЛИРОВАННЫХ ПРОГРАММНЫХ СРЕД (КОНТЕЙНЕРОВ)

ОС содержит программные средства (средства контейнеризации), реализующие создание и функционирование изолированных программных сред, с обеспечением выполнения следующих функций безопасности:

- изоляция контейнеров;
- выявление уязвимостей в образах контейнеров;
- проверка корректности конфигурации контейнеров;
- контроль целостности контейнеров и их образов;
- регистрация событий безопасности;
- идентификация и аутентификация пользователей.

Средства контейнеризации реализуют функциональные возможности по созданию образов контейнеров, формированию среды выполнения контейнеров и обеспечения выполнения их процессов, запуску контейнера, а также позволяют управлять данным контейнером. Также возможно создание групп контейнеров («подов»), в которых контейнеры выполняются совместно, разделяя между собой общие ресурсы.

В состав ОС входит программное обеспечение Docker и Podman для автоматизации развертывания и управления приложениями в средах с поддержкой контейнеризации. Описание установки, настройки и работы с контейнерами Docker и Podman приведено в РУСБ.10015-01 95 01-1.

8.1. Изоляция контейнеров и пространств

В ОС реализованы механизмы изоляции уровня ядра Linux, описание которых приведено в разделе 7. Механизмы изоляции реализуют функции изоляции контейнеров, включая изоляцию пространств идентификаторов процессов контейнеров и пространств имен.

Изоляция доступа контейнеров к ресурсам аппаратной платформы достигается путем применения механизма групп управления `control groups` (`cgroups`), используемого для ограничения в ресурсах группы процессов, выполняющихся в контейнере, а также для ограничения других групп, например групп процессов, принадлежащих какой-либо службе или пользовательской сессии.

Различают следующие контроллеры:

1) `cpu` — накладывает ограничения на долю процессорного времени (cpu share);
2) `cpuacct` — подсчитывает потребленные ресурсы процессора;
3) `cpuset` — привязывает процессы к конкретным процессорам;
4) `memory` — ограничивает потребление памяти;

5) `devices` — ограничивает доступ к файлам устройств;

6) `freezer` — позволяет приостанавливать и возобновлять работу групп процессов;

7) `net_cls` и `net_prio` — назначают сетевому трафику процессов класс и приоритет обработки для использования сетевыми фильтрами и планировщиками;

8) `blkio` — ограничивает количество запросов к блочным (дисковым) устройствам при помощи планировщика CFQ;

9) `pids` — накладывает ограничение на количество процессов в группе.

Изоляция доступа контейнеров к системным вызовам ядра хостовой операционной системы реализуется механизмом ядра Linux SecComp, позволяющим процессам определять системные вызовы, которые они будут использовать. Если злоумышленник получит возможность выполнить произвольный код, SecComp не даст ему использовать системные вызовы, которые не были заранее определены.

Дополнительно для изоляции среды функционирования контейнера в ОС предусмотрена возможность запуска контейнера Docker с пониженной (выделенной) категорией целостности 2.

По умолчанию функция запуска контейнеров на пониженном уровне МКЦ выключена. Управление функцией производится с помощью инструмента командной строки `astra-docker-isolation`, описание которого приведено в 16.6.13.

Также в ОС сохранен для совместимости инструмент командой строки `docker-isolation`. Для включения функции запуска контейнеров на пониженном уровне МКЦ с использованием этого инструмента следует выполнить:

```
sudo /usr/share/docker.io/contrib/parsec/docker-isolation on
sudo systemctl restart containerd
sudo systemctl restart docker
```

Для выключения функции следует выполнить:

```
sudo /usr/share/docker.io/contrib/parsec/docker-isolation off
sudo systemctl restart containerd
sudo systemctl restart docker
```

8.2. Выявление уязвимостей в образах контейнеров

8.2.1. Общие сведения

В состав ОС входит программное обеспечение OpenSCAP, обеспечивающее поиск уязвимостей в образах контейнеров. OpenSCAP использует библиотеку `libopenscap` для

сканирования. Сканирование уязвимостей выполняется с использованием файла базы данных уязвимостей, рекомендуемого ФСТЭК России и содержащего сведения об уязвимостях банка данных угроз безопасности информации ФСТЭК России. В качестве файла базы данных уязвимостей используется файл формата OVAL `/usr/share/oval/db.xml`.

Проверка образов и контейнеров на наличие уязвимостей выполняется автоматически при следующих событиях:

- создание образа из Dockerfile или из контейнера;
- загрузка образа из архива или потока ввода;
- создание файловой системы образа из архива;
- скачивание образа из реестра;
- запуск или перезапуск контейнера.

Периодическая проверка образов контейнеров на наличие известных уязвимостей осуществляется автоматически один раз в неделю.

При обнаружении уязвимостей дальнейшее использование образа контейнера запрещено. Соответствующее ограничение по эксплуатации приведено в 18.2.

Для блокировки запуска контейнера, в образе которого обнаружена уязвимость, применяется глобальный параметр `astra-sec-level` в конфигурационных файлах средств контейнеризации. В качестве значения параметра задается число от 1 до 6, которое определяет класс защиты:

1) 1-5 классы защиты — при обнаружении уязвимости в контейнере его запуск блокируется;

2) 6 класс защиты — отладочный режим, при обнаружении уязвимости в контейнере выводится соответствующее предупреждение, при этом запуск контейнера не блокируется.

В случае если класс защиты не задан или задан не из диапазона 1-6, то при обнаружении уязвимости в контейнере автоматически задается 1 класс защиты с выводом соответствующего сообщения в журнал и запуск контейнера блокируется.

События безопасности, связанные с выявлением уязвимостей в контейнере, регистрируются в соответствии с описанием 8.5.

Для устранения обнаруженной уязвимости и последующего запуска контейнера без блокировки, необходимо:

1) запустить средство контейнеризации в режиме отладки (6 класс защиты);
2) запустить контейнер и устранить уязвимость;
3) запустить средство контейнеризации с требуемым классом защиты.

8.2.2. Настройка класса защиты в контейнерах Docker

Для программного обеспечения Docker задать класс защиты можно одним из способов:

1) с помощью конфигурационного файла, для этого необходимо выполнить следующие действия:

а) создать каталог `/etc/docker`, если он не был создан ранее, командой:

```
sudo mkdir -p /etc/docker
```

б) создать конфигурационный файл `/etc/docker/daemon.json`, если он не был создан ранее, и указать в нем следующие параметры:

```
{
"debug" : true,
"astra-sec-level" : <класс_защиты>
}
```

Пример

```
{
"debug" : true,
"astra-sec-level" : 4
}
```

2) с помощью параметров запуска службы, для этого необходимо выполнить следующие действия:

а) открыть редактор для добавления параметра запуска службы командой:

```
sudo systemctl edit docker
```

б) в редакторе добавить следующие строки:

```
[Service]
Environment="DOCKER_OPTS=--astra-sec-level <класс_защиты>"
```

Пример

```
[Service]
Environment="DOCKER_OPTS=--astra-sec-level 4"
```

в) сохранить изменения и закрыть редактор.

ВНИМАНИЕ! Для указания класса защиты контейнеров Docker допускается использовать только один из способов. Одновременное использование конфигурационного файла и параметров запуска службы `docker` для указания класса защиты приведет к ошибке запуска службы.

ВНИМАНИЕ! После внесения изменений необходимо перезапустить службу `docker` командой:

```
sudo systemctl restart docker
```

Кроме того, возможен разовый запуск службы `docker` с заданным классом защиты, отличным от указанного в настройках службы. Для этого необходимо остановить службу `docker`, а затем выполнить команду:

```
sudo dockerd --astra-sec-level <класс_защиты>
```

Пример

Остановить службу `docker` командой:

```
sudo systemctl stop docker
```

затем запустить службу `docker`, указав шестой класс защиты:

```
sudo dockerd --astra-sec-level 6
```

Изменения класса защиты не будут сохранены. После перезагрузки служба `docker` будет запущена с исходными настройками.

8.2.3. Настройка класса защиты в контейнерах Podman

Для программного обеспечения Podman задать класс защиты можно с помощью конфигурационного файла `/etc/podman.conf`. Для этого необходимо создать конфигурационный файл `/etc/podman.conf`, если он не был создан ранее, и указать в нем значение параметра `astra-sec-level`:

```
{
"astra-sec-level" : <класс_защиты>
}
```

Пример

```
{
"astra-sec-level" : 4
}
```

Для просмотра настроек программного обеспечения Podman выполнить команду:

```
podman astra-config
```

Пример вывода в результате выполнения команды:

Пример

Вывод команды просмотра настроек программного обеспечения Podman:

```
{
"astra-sec-level": 4,
"oscap-report-dir": "/home/astra/.podman/scanoval/reports",
"oscap-exec": "/usr/bin/oscap",
"oscap-db-xml": "/usr/share/oval/db.xml"
}
```

8.3. Обеспечение корректности конфигурации контейнеров

Обеспечение корректности конфигурации контейнеров, в том числе ограничение прав прикладного программного обеспечения, выполняемого внутри контейнера, на использование периферийных устройств, вычислительных ресурсов хостовой операционной системы, устройств хранения данных и съемных машинных носителей информации (блочных устройств); а также монтирование корневой файловой системы хостовой операционной системы в режиме «только для чтения» и ограничение доступа прикладного программного обеспечения, выполняемого внутри контейнера, к системным вызовам ядра хостовой операционной системы, осуществляется средствами Docker и Podman.

8.3.1. Ограничение прав на использование ресурсов хостовой машины для Docker

Для ограничения прав прикладного программного обеспечения, выполняемого внутри контейнера Docker, на использование вычислительных ресурсов (оперативной памяти, операций ввода-вывода за период времени) хостовой операционной системы необходимо установить ограничение на объем памяти путем последовательного выполнения следующих команд:

1) установить ограничение:

```
sudo docker run -it --memory=<объем>m ubuntu bash
```

где <объем>m — объем памяти в МБ;

2) ограничить операции ввода-вывода:

```
sudo docker run -it --memory=<объем>m --device-write-bps \
     /dev/sda:<объем>mb ubuntu bash
```

Для настройки монтирования корневой файловой системы хостовой операционной системы в режиме «только для чтения» необходимо запустить `docker` с параметром `readonly`, указанным через запятую «,» после точки монтирования внутри контейнера:

```
docker run -d --name=nginxtest --mount \
    source=nginx-vol,destination=/usr/share/nginx/html,readonly \
    nginx:latest
```

Ограничение доступа к системным вызовам ядра хостовой операционной системы реализуется с помощью профиля `seccomp`, в котором возможно перечислить запрещенные или разрешенные системные вызовы. Профиль представляет собой файл формата JSON.

Пример

Разрешить все системные вызовы, кроме `chmod` и `chown`:

```
{
"defaultAction": "SCMP_ACT_ALLOW",
"architectures": [
"SCMP_ARCH_X86_64",
"SCMP_ARCH_X86",
"SCMP_ARCH_X32"
],
"syscalls": [
{
"name": "chmod",
"action": "SCMP_ACT_ERRNO",
"args": []
},
{
"name": "chown",
"action": "SCMP_ACT_ERRNO",
"args": []
}
]
}
```

Для запуска контейнера Docker с использованием профиля `seccomp` необходимо использовать параметр `--security-opt` и указать путь к файлу профиля:

```
sudo docker run --security-opt seccomp=/etc/default/profile.json ubuntu bash
```

8.3.2. Ограничение прав на использование ресурсов хостовой машины для Podman

Ограничение прав прикладного программного обеспечения, выполняемого внутри контейнера Podman, на использование вычислительных ресурсов и файловой системы хостовой операционной системы возможно только при отключенном МКЦ. Для корректной работы Podman необходимо перед отключением МКЦ установить пакет `dbus-user-session`. Управление режимом МКЦ описано в 4.9.

Управление ограничениями для Podman на использование ресурсов хостовой операционной системы выполняется в соответствии с 8.3.1.

8.4. Контроль целостности контейнеров и их образов

Контроль целостности объектов контроля (образов контейнеров и исполняемых файлов контейнеров, параметров настройки средства контейнеризации) реализуется сертифицированными средствами регламентного контроля целостности AFICK из состава ОС (см. 10.4).

Для контроля целостности объектов контейнеризации необходимо добавить в конфигурационный файл `/etc/afick.conf` в секции `file section` соответствующие строки:

1) для контроля целостности образов контейнеров Docker и исполняемых файлов контейнеров Docker необходимо получить образ контейнера и поставить на контроль целостности каталог с образами и каталог с кэшем, добавив строки:

```
##############
# file section
##############

/var/lib/docker/image ETC
/var/lib/docker/overlay2 ETC
```

2) для контроля целостности параметров настройки Docker (файл `/etc/docker/daemon.json`) необходимо добавить строку:

```
##############
# file section
##############

/etc/docker/ ETC
```

3) для контроля целостности образов контейнеров Podman и исполняемых файлов контейнеров Podman добавить строки:

```
##############
# file section
```

```
##############

/var/lib/containers/storage/overlay-containers/ ETC
/var/lib/containers/storage/overlay-images/ ETC
/home/<user>/.local/share/containers/storage/overlay-containers/ ETC
/home/<user>/.local/share/containers/storage/overlay-images/ ETC
```

4) для контроля целостности параметров настройки средства контейнеризации Podman добавить строки:

```
##############
# file section
##############
...
/etc/podman.conf
/etc/containers/ ETC
...
```

Системный планировщик заданий `cron` позволяет выполнять проверку целостности объектов контроля в процессе загрузки операционной системы.

Локальное реагирование на события в автозапуске для сессии пользователя, регистрация событий безопасности и оповещение администратора о событиях безопасности осуществляется подсистемой регистрации событий (см. 6.5).

Целостность образов контейнеров и параметров настройки средства контейнеризации также контролируется путем применения цифровой подписи — создания замкнутой программной среды в соответствии с 16.1. Для выполнения контроля необходимо от имени учетной записи администратора:

1) настроить замкнутую программную среду (см. 16.1);
2) получить идентификатор контейнера:

```
root@astra-test:/var/lib/docker# docker inspect nginx | grep Id
```

3) подписать файл конфигурации образа:

```
/var/lib/docker# bsign --sign --xattr
./image/overlay2/imagedb/content/<id>
```

где <id> — идентификатор контейнера;

4) добавить имя контролируемого файла в `/etc/digsig/xattr_control`:

```
sudo echo "var/lib/docker/image/overlay2/imagedb/content/<id>
```

где <id> — идентификатор контейнера;

5) выполнить команду:

```
sudo update-initramfs -u -k all
```

6) выполнить перезагрузку.

При нарушении целостности запуск образа контейнера блокируется.

8.5. Регистрация событий безопасности, связанных с контейнерами

Регистрация событий безопасности, связанных с образами и контейнерами Podman и Docker, осуществляется подсистемой регистрации событий (см. 6.5).

Событиям безопасности, связанным с образами и контейнерами Docker, присваиваются метки `dockerd_audit`. Событиям безопасности, связанным с образами и контейнерами Podman, присваиваются метки `podman_audit`. Записи в журнале имеют следующий формат:

```
podman.audit | user ; uid | событие | результат | дополнительная информация
```

Регистрация событий безопасности, связанных с образами и контейнерами Docker, также осуществляется штатными средствами Docker в системные журналы.

При регистрации событий, связанных с образами и контейнерами, указывается UID и имя пользователя ОС, от имени которого выполнено данное действие. Если действие было выполнено удаленно, то вместо UID и имени пользователя указывается IP-адрес компьютера, с которого выполнено действие. Если инициатор действия в той же ОС, что и служба `dockerd`, но не удалось определить UID пользователя, то вместо UID и имени пользователя указывается символ «@». Также для события указывается результат — успешно или неуспешно. В случае неуспешного события указывается причина ошибки.

События аутентификации пользователя, в т.ч. неуспешные попытки аутентификации, регистрируются:

1) в журнале подсистемы регистрации событий (см. 6.5);
2) в журнале `/var/log/auth.log`.

Регистрация событий безопасности, связанных с выявлением уязвимостей в образе и в контейнере, осуществляется в журнал подсистемы регистрации событий (см. 6.5), а также в системные журналы `/var/log/daemon.log` и `/var/log/syslog.log`. Регистрируются следующие события:

1) запуск сканирования (start) с указанием пути и ID сканируемого контейнера/образа;

2) результат сканирования:

а) если уязвимость не обнаружена, то сообщение об окончании сканирования (complete) с указанием пути и ID сканируемого контейнера/образа;

б) если уязвимость обнаружена, то сообщение о неуспешном сканировании (fail) с указанием версии ПО, операционной системы, обнаруженной уязвимости, адреса отчета сканирования. Если при этом запуск контейнера был заблокирован (класс защиты 1-5), то также регистрируется сообщение о блокировке запуска контейнера.

Регистрация действий с образами и контейнерами (события создания, модификации и удаления образов контейнеров; получения доступа к образам контейнеров; запуска и остановки контейнеров с указанием причины остановки; модификации запускаемых контейнеров) осуществляется в журнале подсистемы регистрации событий (см. 6.5) и в журнале `/var/log/syslog.log`.

Настройка событий, которые подлежат регистрации, и реагирование на них системы осуществляется с помощью утилиты `fly-admin-events` («Настройка регистрации системных событий»). Оповещение администратора безопасности средства контейнеризации о событиях безопасности осуществляется с использованием утилиты `fly-notifications` («Центр уведомлений»). Описание утилит приведено в электронной справке.

8.6. Идентификация и аутентификация пользователей

Первичная идентификация пользователей средства контейнеризации осуществляется администратором безопасности средства контейнеризации. Идентификация и аутентификация пользователей в средстве контейнеризации осуществляется с учетом требований разделов ГОСТ Р 58833-2020. При входе в систему осуществляется идентификация и проверка подлинности субъектов доступа с помощью процедуры аутентификации. Описание процедуры приведено в разделе 2.

Хранение аутентификационной информации пользователей осуществляется с использованием хеш-функции по ГОСТ Р 34.11-94 и по ГОСТ Р 34.11-2012, что обеспечивает ее защиту.

Установка пароля пользователя осуществляется администратором с помощью модуля «Пользователи» в разделе «Пользователи и группы» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_users
```

Пароль пользователя должен содержать не менее 8 символов при алфавите пароля не менее 70 символов. Максимальное количество неуспешных попыток аутентификации (ввода неправильного пароля) до блокировки — 4.

Запрет запуска средством контейнеризации в хостовой операционной системе процессов, обладающих привилегиями администратора информационной (автоматизированной) системы и администратора безопасности информационной (автоматизированной) системы, осуществляется с помощью утилиты `unshare`, позволяющей реализовать:

1) запрет запуска процессов от имени суперпользователя `root`;
2) запрет просмотра процессов хостовой машины;
3) невозможность какого-либо влияния на процессы хостовой машины и других пространств;
4) невлияние монтирования и размонтирования ФС на состав дерева каталогов хостовой машины.

8.7. Работа с Docker в непривилегированном режиме с ненулевыми метками безопасности

8.7.1. Принцип функционирования

Контейнеры Docker в непривилегированном (rootless) режиме (описание режима приведено в РУСБ.10015-01 95 01-1) могут быть запущены от имени любого непривилегированного пользователя с ненулевой меткой безопасности контейнера. Для корректной работы программного интерфейса (API) PARSEC при запуске контейнера необходимо примонтировать каталог `/parsecfs/` хостовой ОС в одноименный каталог внутри контейнера, например с помощью команды:

```
docker run -it -v /parsecfs:/parsecfs astra
```

Метка безопасности контейнера всегда задается четырьмя десятичными неотрицательными числами, разделенными двоеточием:

```
0:63:0:0
```

где первое число — иерархический уровень конфиденциальности;
второе число — неиерархическая категория целостности;
третье число — неиерархические категории конфиденциальности;
четвертое число — зарезервировано для задания мандатных атрибутов файловых объектов и в работе с контейнерами не применяется (следует всегда использовать значение `0`).

Запуск контейнера при работе в расширенном режиме МКЦ (strict mode) невозможен, так как в этом режиме не работает Docker.

При запуске контейнера в непривилегированном режиме с ненулевой меткой безопасности применяются следующие ограничения:

1) контейнер может быть запущен с меткой безопасности не выше метки безопасности пользователя;

2) запуск процессов внутри контейнера возможен только с одинаковой для всех процессов меткой безопасности, равной метке безопасности контейнера (задается при запуске контейнера);

3) в метке безопасности контейнера ненулевой может быть либо только классификационная метка, либо только метка целостности (при этом отрицательная метка целостности также является ненулевой меткой);

4) для системных файлов внутри контейнера (исполняемых, конфигурационных и т.д.) всегда используется ненулевая категория целостности (например, `0:2:0:0` или `0:63:0:0`) и, соответственно, всегда используется нулевая классификационная метка;

5) процессы (субъекты), работающие внутри контейнера в непривилегированном режиме, не могут изменять метки безопасности файловых объектов в файловой системе своего контейнера;

6) если внутри контейнера необходима работа с файлами с ненулевой классификационной меткой, то данные файлы должны группироваться в специально созданных каталогах:

```
/var/lib/rootlessdocker/<имя_пользователя>/l<уровень_конфиденциальности>
i<иерархический_уровень_целостности>c<категории_конфиденциальности>t0x0
```

Пример

```
/var/lib/rootlessdocker/astra/l0i2c0x0t0x0
```

При этом метки безопасности таких каталогов могут сочетать ненулевые классификационные метки и ненулевые метки целостности. Классификационные метки данных каталогов должны быть не выше классификационной метки контейнера.

Для хостовой ОС контейнер, запущенный в непривилегированном режиме, является группой процессов:

1) имеющих метку безопасности субъекта, запустившего контейнер;

2) не имеющих прав администратора;

3) работающих с набором файловых объектов, расположенных в файловой системе контейнера и имеющих собственные метки безопасности.

При этом процессы внутри контейнера:

1) запущены от имени администратора;
2) имеют неограниченный доступ к объектам файловой системы контейнера (за исключением возможности изменять метки безопасности).

Файловые объекты в файловой системе контейнера (rootFS) создаются с меткой безопасности, в которой:

1) классификационная метка равна классификационной метке субъекта, запустившего контейнер;
2) метка целостности всегда нулевая.

К процессам (субъектам) внутри контейнера, запущенного в непривилегированном режиме с ненулевой меткой безопасности, будут применяться общие правила мандатного управления доступом и мандатного контроля целостности хостовой ОС. Например, процессы контейнера в непривилегированном режиме, имеющего метку безопасности `1:0:0:0`, смогут изменять файлы с меткой безопасности `1:0:0:0`, читать содержимое файлов с меткой безопасности `1:0:0:0` или `0:0:0:0`).

Описание работы с контейнерами Docker в непривилегированном режиме с ненулевыми метками безопасности также приведено на справочной странице `man rootless-helper-astra` `man rootlessenv`.

8.7.2. Управление запуском контейнера с ненулевой меткой безопасности

Для работы с контейнерами в непривилегированном режиме от имени пользователя с ненулевой меткой безопасности следует запустить для этого пользователя непривилегированную службу Docker с указанием соответствующей метки безопасности.

Запуск службы для работы с ненулевой классификационной меткой выполняется командой:

```
sudo systemctl start rootless-docker@$(systemd-escape <имя_пользователя>@\
     <метка_безопасности>)
```

Запуск службы для работы с различными классификационными метками выполняется командой:

```
sudo systemctl start rootless-docker@$(systemd-escape <имя_пользователя>@\
     0:0:0:0@privsock)
```

при этом использование флага `privsock` запускает непривилегированную службу Docker с привилегией `PARSEC_CAP_PRIV_SOCK`, позволяющей выполнять команды для сетевых подключений (например, `docker pull`) игнорируя мандатные ограничения (см. 4.7).

Для настройки автоматического запуска службы после перезагрузки можно выполнить команду:

```
sudo systemctl enable rootless-docker@$(systemd-escape \
     <имя_пользователя>@<метка_безопасности>@privsock)
```

После запуска для пользователя непривилегированной службы Docker возможно выполнение команд Docker от имени данного пользователя (например, копирование образов в пользовательские репозитории образов или запуск контейнеров).

8.7.3. Копирование образа в репозиторий пользователя

Для запуска контейнера пользователем в сессии с ненулевой классификационной меткой требуется создать пользовательский репозиторий с соответствующей меткой, содержащий образы.

Для создания пользовательских копий образов Docker необходимо:

1) экспортировать образ:

```
rootlessenv docker save -o /tmp/<имя_архива>.tar <имя_образа>
```

Экспорт рекомендуется выполнять в каталог `/tmp/`, доступный для чтения всем пользователям;

2) разрешить чтение экспортированного образа:
   а) всем пользователям:

```
chmod o+r /tmp/<имя_архива>.tar
```

   б) только указанным пользователям:

```
setfacl -m u:<имя_пользователя>:r /tmp/<имя_архива>.tar
```

3) импортировать образ с помощью инструмента `pdp-exec`, указав в команде имя пользователя с нужными метками безопасности (при этом должны быть запущены непривилегированные службы Docker для указанного пользователя с соответствующими метками безопасности):

```
sudo pdp-exec -u <имя_пользователя> -l <метка_безопасности> \
     -- rootlessenv docker load -i /tmp/<имя_архива>.tar
```

8.7.4. Выполнение команд и запуск контейнеров в непривилегированном режиме от имени пользователя

Команды Docker, выполняемые пользователем (или администратором от имени пользователя) выполняются с меткой безопасности сессии пользователя (или указанной администратором):

1) для запуска контейнера пользователем из своей сессии (контейнер будет запущен из репозитория образов пользователя, имеющего метку безопасности, равную метке безопасности сессии пользователя, и унаследует данную метку безопасности):

```
rootlessenv docker run <имя_образа>
```

2) для запуска администратором оболочки командой строки контейнера, из которой можно выполнять команды Docker от имени указанного пользователя, выполнить команду:

```
sudo pdp-exec -u <имя_пользователя> -l <метка_безопасности> \
     -- rootlessenv
```

3) для запуска контейнера администратором от имени указанного пользователя выполнить команду:

```
sudo pdp-exec -u <имя_пользователя> -l <метка_безопасности> \
     -- rootlessenv docker run --rm -ti <имя_образа>
```

Пользователь (администратор) может со стороны хостовой ОС присваивать метки безопасности файловым объектам в файловой системе контейнера (например, `1:0:0:0` для конфиденциальных пользовательских файлов и `0:2:0:0` для системных файлов контейнера с высокой категорией целостности).

Местонахождение файловой системы контейнера на хостовой ОС можно узнать командой:

```
docker inspect <имя_контейнера> | egrep "(Lower|Upper)Dir"
```

При работе администратора с контейнером пользователя данная команда должна выполняться от имени пользователя с указанием нужной метки безопасности:

```
sudo pdp-exec -u <имя_пользователя> -l <метка_безопасности> \
     -- rootlessenv docker inspect <имя_контейнера> | egrep "(Lower|Upper)Dir"
```

8.8. Работа с Podman в непривилегированном режиме с ненулевыми метками безопасности

Работа с контейнерами Podman в непривилегированном режиме с ненулевыми метками безопасности доступна только администратору с высокой меткой целостности. Запуск контейнеров Podman с пониженной (выделенной) категорией целостности не поддерживается.

9. ЗАЩИТА ПАМЯТИ

9.1. Очистка памяти

Ядро ОС гарантирует, что обычный непривилегированный процесс не получит данные чужого процесса, если это явно не разрешено ПРД. Это означает, что средства IPC контролируются с помощью ПРД, и процесс не может получить неочищенную память (как оперативную, так и дисковую).

Дополнительно ядро обеспечивает возможности по очистке остаточной информации:

- очистку остаточной информации в ядерном стеке (STACKLEAK);
- очистку остаточной информации в ядерной куче (PAGE_POISONING).

В ОС реализован механизм очистки освобождаемой внешней памяти. Механизм может быть включен при установке ОС путем выбора соответствующего пункта. После установки ОС включение механизма осуществляется в соответствии с 16.6.29.

Реализованный в ОС механизм очистки освобождаемой внешней памяти очищает неиспользуемые блоки ФС непосредственно при их освобождении, а также очищает разделы страничного обмена. Работа данного механизма снижает скорость выполнения операций удаления и усечения размера файла.

Данные любых удаляемых/усекаемых файлов в пределах заданной ФС предварительно очищаются предопределенной или псевдослучайной маскирующей последовательностью. Механизм является настраиваемым и позволяет обеспечить работу ФС (ext2/ext3/ext4/XFS) в одном из следующих режимов:

1) очистка осуществляется путем перезаписи каждого байта в освобождаемой области посредством четырех сигнатур следующего вида: `11111111`, `01010101`, `10101010`, `00000000`. Использование режима включается параметром `secdel` в конфигурационном файле `/etc/fstab` для раздела ФС, на котором требуется очищать блоки памяти при их освобождении (например, `/dev/sda1`). В список параметров монтирования добавляется параметр `secdel`.

Пример

```
/dev/sda1 /home ext4 acl,defaults,secdel 0 2
```

2) очистка осуществляется путем перезаписи каждого байта в освобождаемой области посредством четырех сигнатур следующего вида: `11111111`, `01010101`, `10101010`, `00000000`. Количество перезаписей определяется администратором. Использование режима включается установкой значения параметра `secdel` в конфигурационном файле `/etc/fstab` для раздела ФС, на котором требуется очищать

блоки памяти при их освобождении (например, /dev/sda1). При установке числа перезаписей больше четырех сигнатуры используются повторно. Например, при установке числа перезаписей, равного 6, последовательность сигнатур, используемых для перезаписи, имеет следующий вид: 11111111, 01010101, 10101010, 00000000, 11111111, 01010101. В список параметров монтирования добавляется параметр secdel=6.

Пример

```
/dev/sda1 /home ext4 acl,defaults,secdel=6 0 2
```

3) очистка осуществляется путем перезаписи каждого байта в освобождаемой области посредством четырех псевдослучайных сигнатур. Использование режима включается параметром secdelrnd в конфигурационном файле /etc/fstab для раздела ФС, на котором требуется очищать блоки памяти при их освобождении (например, /dev/sda1). В список параметров монтирования добавляется параметр secdelrnd.

Пример

```
/dev/sda1 /home ext4 acl,defaults,secdelrnd 0 2
```

4) очистка осуществляется посредством перезаписи каждого байта в освобождаемой области посредством псевдослучайных сигнатур. Количество перезаписей определяется администратором. Использование режима включается установкой значения параметра secdelrnd в конфигурационном файле /etc/fstab для раздела ФС, на котором требуется очищать блоки памяти при их освобождении (например, /dev/sda1). Например, при установке числа перезаписей, равного 6, в список параметров монтирования добавляется параметр secdelrnd=6.

Пример

```
/dev/sda1 /home ext4 acl,defaults,secdelrnd=6 0 2
```

Включение механизма очистки блоков памяти при их освобождении в первом режиме может быть выполнено с использованием инструмента astra-secdel-control, описанного в 16.6.29.

Для включения очистки активных разделов страничного обмена необходимо установить в конфигурационном файле /etc/parsec/swap_wiper.conf для параметра ENABLED значение Y.

Пример

```
ENABLED=Y
```

Включение очистки активных разделов страничного обмена может быть выполнено с использованием инструмента `astra-swapwiper-control`, описанного в 16.6.30.

Для задания списка разделов страничного обмена, для которых не выполняется очистка, может быть использован параметр `IGNORE`, значение которого является списком перечисленных через пробел игнорируемых разделов страничного обмена.

Пример

```
IGNORE="/dev/sdz10 /dev/sdz11"
```

Установка параметра монтирования для очистки блоков памяти при их освобождении, а также настройка очистки разделов страничного обмена при выключении системы может быть выполнена с использованием модуля «Политика очистки памяти» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_policy_clean_memory
```

9.2. Средства ограничения прав доступа к страницам памяти

Средства ограничения прав доступа к страницам памяти реализованы на основе возможностей ядра и параметров ядра ОС.

Включенные параметры ядра ОС предоставляют защиту задачам ядра и процессам пользователей при доступе к страницам оперативной памяти:

- запрет записи в область памяти, помеченную как исполняемая;
- запрет создания исполняемых областей памяти;
- запрет создания исполняемого стека;
- случайный выбор (рандомизацию) адресного пространства процесса.

Предотвращение выполнения произвольного кода обеспечивается путем контроля доступа к страницам памяти по типам: чтение, запись, исполнение или их комбинации.

Применяются различные механизмы защиты памяти, которые основаны на эмуляции или аппаратной реализации NX-бита (бита исполнения на страницах памяти) и PCID (Processor-Context ID — идентификатор контекста выполнения), а также технологии ASLR (случайный выбор расположения виртуального адресного пространства) и KASLR (случайный выбор адресного пространства ядра). При наличии аппаратной поддержки NX-бита и PCID на процессорах x86-64 производительность вычислительной системы не ухудшается.

Ядро ОС имеет архитектуру, обеспечивающую невозможность его перемещения в физическом адресном пространстве, при этом технология KASLR задет случайное смещение начального адреса выполнения ядра при каждой его загрузке.

Технология ASLR обеспечивает случайный характер (рандомизацию) смещений сегментов кода и данных (в том числе стека и кучи) при использовании системного вызова отображения в память `mmap()`.

Гарантия того, что адреса с произвольным доступом не будут одновременно доступны на запись и выполнение, реализуется использованием возможностей системных вызовов `mmap()` и `mprotect()`.

10. КОНТРОЛЬ ЦЕЛОСТНОСТИ

Для обеспечения контроля целостности (в т. ч. контроля целостности КСЗ) в ОС реализованы:

- средство подсчета контрольных сумм файлов и оптических дисков (10.1);
- средство подсчета контрольных сумм файлов в deb-пакетах (10.2);
- средство контроля соответствия дистрибутиву (10.3);
- средства регламентного контроля целостности (10.4);
- средства создания и проверки ЭП (10.5);
- средства создания замкнутой программной среды (16.1).

Для решения задач контроля целостности используется библиотека `libgost`, в которой для вычисления контрольных сумм реализованы функции хеширования в соответствии с ГОСТ Р 34.11-2012 с длиной хеш-кода 256 бит и ГОСТ Р 34.11-2012 с длиной хеш-кода 512 бит и ГОСТ Р 34.11-94. По умолчанию используются функции хеширования в соответствии с ГОСТ Р 34.11-2012. Использование функции в соответствии с ГОСТ Р 34.11-94 сохранено для совместимости.

Библиотека `libgost` используется в средствах подсчета контрольных сумм файлов и оптических дисков, контроля соответствия дистрибутиву и регламентного контроля целостности, модулях аутентификации.

10.1. Подсчет контрольных сумм файлов и оптических дисков

Для подсчета контрольных сумм файлов и оптических дисков в состав ОС включен инструмент командной строки `gostsum`. Для вывода информации о синтаксисе `gostsum` необходимо выполнить команду:

```
gostsum -h
```

Синтаксис инструмента:

```
gostsum [параметр] [файл]
```

Параметры инструмента `gostsum` приведены в таблице 36.

Таблица 36

Параметр	Описание
`--gost-94`	Рассчитать контрольную сумму по стандарту ГОСТ Р 34.11-94

Окончание таблицы 36

Параметр	Описание
`--gost-2012`	Рассчитать контрольную сумму по стандарту ГОСТ Р 34.11-2012 с длиной хеш-кода 256 бит (используется по умолчанию, если не задан другой параметр)
`--gost-2012-512`	Рассчитать контрольную сумму по стандарту ГОСТ Р 34.11-2012 с длиной хеш-кода 512 бит
`-b`	Установить размер блоков, которыми будет считываться файл. По умолчанию размер блока равен 512 байтам
`-o`	Задать имя файла для вывода контрольной суммы (по умолчанию стандартный поток вывода)
`-d`	Задать имя файла устройства чтения оптических дисков (файла с образом оптического диска) для подсчета контрольной суммы (для форматов ISO9660 или UDF)
`-e`	Рассчитать контрольные суммы файлов из deb-пакета
`-r`	Выполнять рекурсивно
`-E`	Исключить все файлы, кроме deb-пакетов
`-H`	Отключить отображение индикатора прогресса
`-t`	Протестировать алгоритмы подсчета контрольных сумм
`-h`	Показать эту справку и выйти

Пример

Подсчет контрольной суммы оптического диска:

```
gostsum -d /dev/cdrom
```

10.2. Подсчет контрольных сумм файлов в deb-пакетах

Для подсчета контрольных сумм файлов в deb-пакетах в состав ОС включен инструмент командной строки `gostsum_from_deb`.

Инструмент устарел и вместо него рекомендуется использовать `gostsum` (см. 10.1)

Для вывода информации о синтаксисе `gostsum_from_deb` необходимо выполнить команду:

```
gostsum_from_deb -h
```

Синтаксис инструмента:

```
gostsum_from_deb [gostsum аргументы] [-d каталог] [-p deb-пакет]
```

Параметры инструмента gostsum_from_deb приведены в таблице 37.

Таблица 37

Параметр	Описание
--gost-94	Устанавливает, что будет использован алгоритм ГОСТ Р 34.11-94
--gost-2012	Устанавливает, что будет использован алгоритм ГОСТ Р 34.11-2012 с длиной хеш-кода 256 бит (по умолчанию)
--gost-2012-512	Устанавливает, что будет использован алгоритм ГОСТ Р 34.11-2012 с длиной хеш-кода 512 бит
gostsum arguments	Аргументы утилиты gostsum
-d каталог	Задает имя каталога, содержащего deb-пакеты, для файлов которых вычисляются контрольные суммы
-p deb-пакет	Задает имя deb-пакета, для файлов которого вычисляются контрольные суммы

10.3. Контроль соответствия дистрибутиву

Для обеспечения контроля целостности объектов ФС (в т.ч. СЗИ) в состав ОС входят средства, предоставляющие возможность контроля соответствия дистрибутиву объектов ФС установленной ОС.

В состав дистрибутива входит файл gostsums.txt со списком файлов установленной ОС, которые не должны изменяться в процессе эксплуатации, и их контрольными суммами. Контроль целостности может выполняться для файлов, указанных в gostsums.txt, или на основе шаблонов. Шаблоны проверки представляют собой текстовые файлы, содержащие списки файлов для проверки целостности, по умолчанию в системе присутствуют следующие шаблоны:

1) /usr/share/gostsum/templates/gostsum_minimal.txt — список минимального состава файлов ОС;

2) /usr/share/gostsum/templates/gostsum_fb.txt — список файлов, реализующих функции безопасности или поддерживающих их выполнение.

В файле gostsums.txt контрольные суммы рассчитаны по ГОСТ Р 34.11-2012 с длиной хеш-кода 256 бит.

Также в ОС возможно проверять MD5-суммы установленных в системе deb-пакетов и сравнивать их с эталонными значениями, указанным в файлах *.md5sums в каталоге /var/lib/dpkg/info/. Проверка может выполняться как на основе всех файлов *.md5sums, так и на основе шаблонов.

Вычисление контрольных сумм файлов и проверка соответствия полученных значений эталонным контрольным суммам выполняются с помощью графической утилиты `fly-admin-int-check` или инструмента командной строки `astra-int-check`.

Для вычисления и проверки контрольных сумм с помощью инструмента командной строки используется команда:

```
astra-int-check [параметры]
```

Параметры инструмента `astra-int-check` приведены в таблице 38.

Таблица 38

Параметр	Описание
`--templates`, `-t <файлы>`	Указать шаблоны проверки. Файлы перечисляются через запятую с указанием полного пути. Если для параметра указано пустое значение `-t ""`, то проверка аналогична проверке с параметром `-n`. Если параметр не указан и отсутствует параметр `-n`, то проверка будет выполняться по всем доступным шаблонам
`--no-templates`, `-n`	Отключить все шаблоны. Будут проверяться все файлы, указанные в файле `gostsums.txt` дистрибутива ОС или, при использовании без параметра `-p`, все файлы, для которых есть контрольные суммы по MD5 в каталоге `/var/lib/dpkg/info/`
`--list-templates`, `-l`	Показать доступные шаблоны
`--mount-point`, `-p <каталог>`	Точка монтирования диска с дистрибутивом ОС. Если данный параметр не указан, будет проводиться проверка MD5-сумм установленных deb-пакетов
`--add`, `-a "<фильтр>"`	Фильтры для проверки дополнительных файлов и/или каталогов в виде регулярных выражений, перечисляемых через запятую
`--ignore`, `-i "<фильтр>"`	Фильтры для исключения файлов и/или каталогов из проверки в виде регулярных выражений, перечисляемых через запятую
`--txt <файл>`	Сохранить отчет о проверке в виде текстового файла
`--html <файл>`	Сохранить отчет о проверке в формате HTML
`--xml <файл>`	Сохранить отчет о проверке в формате XML
`--check-directory`, `-d <каталог>`	Проверить содержимое произвольного каталога на соответствие файлу с контрольными суммами
`--gost <файл>`	Файл с контрольными суммами ГОСТ Р 34.11-2012
`--md5 <файл>`	Файл с контрольными суммами MD5
`--tui`	Запуск псевдографического интерфейса пользователя, использование которого аналогично графической утилите `fly-admin-int-check`
`--verbose`	Подробный вывод
`-h`, `--help`	Вывод справки по параметрам командной строки
`-v`, `--version`	Показать сведения о версии

Примеры:

1. Проверка содержимого каталога `/home/user/docs` на соответствие контрольным суммам по ГОСТ Р 34.11-2012, перечисленным в файле `gost.txt`, с подробным выводом процесса проверки в консоль:

```
astra-int-check -d /home/user/docs --gost /home/user/gost.txt --verbose
```

Если в файле с контрольными суммами используются относительные пути, то они будут добавлены к указанному каталогу. Если в файле используются абсолютные пути, то проверяться будут только те пути, которые начинаются с указанного каталога.

2. Проверка целостности файлов, перечисленных в шаблоне `gostsum_fb.txt`, за исключением содержимого каталога `/usr/lib`, с созданием отчета о проверке в текстовом формате:

```
astra-int-check -t /usr/share/gostsum/templates/gostsum_fb.txt -p \
     /media/cdrom0 -i "/usr/lib/*" --txt /home/user/report.txt
```

Описание использования графической утилиты `fly-admin-int-check` см. в электронной справке.

10.4. Регламентный контроль целостности

Организация регламентного контроля целостности ОС, прикладного ПО и СЗИ обеспечивается набором программных средств на основе Another File Integrity Checker, представленного пакетом `afick`. В указанном наборе реализована возможность для проведения периодического (с использованием системного планировщика заданий `cron`) вычисления контрольных сумм файлов и соответствующих им атрибутов расширенной подсистемы безопасности PARSEC (мандатных атрибутов и атрибутов расширенной подсистемы протоколирования) с последующим сравнением вычисленных значений с эталонными. В указанном наборе программных средств реализовано использование библиотеки `libgost`, обеспечивающей подсчет контрольных сумм в соответствии с ГОСТ Р 34.11-2012 с длиной хеш-кода 256 бит.

Эталонные значения контрольных сумм и атрибутов файлов хранятся в БД. Данная БД контрольных сумм и атрибутов может быть создана при помощи команды:

```
afick -i
```

Созданный в результате выполнения команды файл `/var/lib/afick/afick.db`. Файл содержит БД с форматом данных perl Storable. В файле содержится набор строк, в каждой строке указано имя файла и через пробел его атрибуты и сигнатуры. БД защищается системой разграничения доступа.

Для вычисления контрольных сумм могут использоваться алгоритмы MD5-Digest и ГОСТ Р 34.11-2012.

Настройка регламентного контроля целостности выполняется в конфигурационном файле `/etc/afick.conf`, в котором указываются пути к файлам/каталогам, подвергаемым контролю целостности, и правила контроля целостности, применяемые к файлам и каталогам. Подробное описание файла приведено на справочной странице `man afick.conf`.

По умолчанию в файле `/etc/afick.conf` используются следующие правила контроля целостности:

1) правило `PARSEC`:

```
PARSEC = p+d+i+n+u+g+s+b+md5+m+e+t
```

где `p+d+i+n+u+g+s+b+md5+m` — слежение за всеми стандартными атрибутами файла и использование хеш-функции MD5-Digest для слежения за целостностью содержимого файлов;
`+e+t` — контроль расширенных атрибутов: метки безопасности и флагов аудита, соответственно.

Контроль ACL осуществляется при установке флага `+g`;

2) правило `GOST`:

```
GOST = p+d+i+n+u+g+s+b+gost+m+e+t
```

где `p+d+i+n+u+g+s+b+gost+m` — слежение за всеми стандартными атрибутами файла и использование хеш-функции ГОСТ Р 34.11-2012 с длиной хеш-кода 256 бит для слежения за целостностью содержимого файлов;
`+e+t` — контроль расширенных атрибутов: метки безопасности и флагов аудита, соответственно.

Контроль ACL осуществляется при установке флага `+g`;

В файле `/etc/afick.conf` в отдельной строке задается путь к файлу или каталогу и применяемое к нему правило контроля.

Пример

```
/boot             GOST
/bin              GOST
/etc/security     PARSEC
/etc/pam.d        PARSEC
/etc/fstab        PARSEC
/lib/modules      PARSEC
/lib64/security   PARSEC
/lib/security     PARSEC
/sbin             PARSEC
```

```
/usr/bin        PARSEC
/usr/lib        PARSEC
/usr/sbin       PARSEC
```

В конфигурационном файле также представлен ряд дополнительных путей с правилами контроля. Соответствующие строки помечены знаком комментария «#» и могут быть активированы удалением этого знака.

При запуске `afick` автоматически будет установлено ежедневное задание для `cron`. Файл с заданием находится в `/etc/cron.daily/afick_cron/`.

Параметр `report_url:=stdout` задает местоположение файла-отчета.

В конфигурационном файле применен простой язык макросов, который используется при определении переменных для заданий системного планировщика заданий `cron`.

Рекомендуется утилитой `afick` и/или средствами аппаратно-программного модуля доверенной загрузки обеспечить контроль целостности следующих объектов:

1) файлов образов ядра ОС:

`/boot/vmlinuz-*`

2) файлов образов временной файловой системы, используемой ядром ОС при начальной загрузке (добавляются в список контроля целостности после выполнения всех необходимых операций по настройке, требующих обновления образов временной файловой системы):

`/boot/initrd.img-*`

3) конфигурационного файла меню загрузчика GRUB 2:

`/boot/grub/grub.cfg`

4) конфигурационного файла, определяющего используемый по умолчанию графический дисплейный менеджер:

`/etc/X11/default-display-manager`

5) конфигурационного файла настройки файловых систем, доступных для монтирования через NFS:

`/etc/exports`

6) конфигурационного файла, содержащего информацию о различных устройствах хранения и файловых системах:

`/etc/fstab`

7) файла, содержащего перечень локальных групп ОС (добавляется в список контроля целостности после создания всех необходимых групп):

```
/etc/group
```

8) сценариев для запуска служб в каталоге:

```
/etc/init.d/
```

9) конфигурационного файла, определяющего параметры работы первого процесса пользовательского режима `init`:

```
/etc/inittab
```

10) конфигурационных файлов, определяющих порядок работы PAM-модулей:

```
/etc/pam.conf
/etc/pam.d/chfn
/etc/pam.d/chsh
/etc/pam.d/common-account
/etc/pam.d/common-account.pam-old
/etc/pam.d/common-auth
/etc/pam.d/common-auth.pam-old
/etc/pam.d/common-password
/etc/pam.d/common-password.pam-old
/etc/pam.d/common-session
/etc/pam.d/common-session.pam-old
/etc/pam.d/cron
/etc/pam.d/cups
/etc/pam.d/cvs
/etc/pam.d/dovecot
/etc/pam.d/fly-dm
/etc/pam.d/fly-dm-np
/etc/pam.d/login
/etc/pam.d/other
/etc/pam.d/passwd
/etc/pam.d/polkit
/etc/pam.d/samba
/etc/pam.d/sshd
/etc/pam.d/su
/etc/pam.d/sumac.xauth
```

11) файла, содержащего перечень локальных пользователей ОС (добавляется в список контроля целостности после создания всех необходимых пользователей):

```
/etc/passwd
```

12) символических ссылок на сценарии для запуска служб в каталогах:

```
/etc/rc*
```

13) конфигурационного файла, определяющего перечень терминалов, с которых суперпользователь `root` может регистрироваться в системе:

```
/etc/securetty
```

14) конфигурационного файла, определяющего перечень регистрируемых оболочек в ОС:

```
/etc/shells
```

15) конфигурационного файла, содержащего значения параметров командной строки ядра:

```
/etc/sysctl.conf
```

16) модулей ядра, входящих в подсистему безопасности PARSEC:

```
/lib/modules/*/misc/digsig_verif.ko
/lib/modules/*/misc/parsec.ko
/lib/modules/*/misc/parsec-cifs.ko
```

С помощью команд `lsmod` и `modinfo` можно определить перечень модулей ядра, подлежащих контролю целостности средствами АПМДЗ;

17) PAM-модулей в каталоге:

```
/lib/security/
```

18) исполняемых файлов в каталогах (в случае, если в ОС не используется замкнутая программная среда (см. 16.1)):

```
/bin/
/sbin/
/usr/bin/
/usr/sbin/
```

В режиме «Мобильный» в каталоге `/usr/sbin/` доступны сценарии регламентного контроля целостности:

1) `astra-mobile-create-int-db` — создание базы данных эталонного состояния системы. Если включена функция проверки целостности системы при запуске, то сценарий будет выполняться при каждом обновлении списка пакетов (установке/удалении пакетов);

2) `astra-mobile-int-check` — проверка целостности системы. Сценарий будет выполняться при запуске системы, если функция проверки целостности включена. В случае неудачной проверки делается соответствующая запись в файле `/tmp/astra-mobile-int-check.log`.

Описание настройки регламентного контроля целостности с помощью `afick` в графическом интерфейсе режима «Мобильный» приведено в электронной справке («Документация — Графический интерфейс — Режим «Мобильный»).

10.5. Сервис электронной подписи

Комплекс программ из состава ОС предоставляет сервис электронной подписи (СЭП), обеспечивающий интеграцию с дополнительно устанавливаемыми сертифицированными ФСБ России средствами криптографической защиты информации[1)] (СКЗИ) в целях создания и проверки усиленной квалифицированной электронной подписи[2)] .

ВНИМАНИЕ! СЭП предоставляется программами, функционирующими в условиях политики разграничения доступа, не допускающей их применение совместно с СКЗИ в режиме обработки сведений, составляющих государственную тайну.

ВНИМАНИЕ! Эксплуатация СКЗИ в составе информационных систем должна осуществляться в соответствии с правилами пользования СКЗИ и указаниями, определенными в формуляре (или иных эксплуатационных документах) на СКЗИ.

СЭП обеспечивает создание и проверку электронной подписи электронных документов в соответствии с ГОСТ Р 34.10-2012 средствами сертифицированных СКЗИ.

Дополнительная информация о настройке ОС и СКЗИ для совместного использования, а также варианты реализации отдельных решений приведены на официальном информационном ресурсе разработчика ОС `wiki.astralinux.ru`.

10.5.1. Принцип функционирования

СЭП[3)] обеспечивается совокупностью совместно функционирующих программных компонентов ОС:

1) `astra-csp-cryptopro` — программный модуль, обеспечивающий предоставление СЭП из командной строки и программного интерфейса. Реализует функции создания и проверки ЭП без взаимодействия в графическом интерфейсе;

2) `fly-csp-cryptopro` — программный модуль, обеспечивающий предоставление СЭП из основного меню, графического интерфейса менеджера файлов `fly-fm` и командной строки. Предоставляет функции создания и проверки ЭП в графическом интерфейсе пользователя;

3) `libreoffice-astracsp-plugin` — расширение, обеспечивающее предоставление СЭП из графического интерфейса пакета офисных программ LibreOffice в процессе работы с электронными документами;

[1)] Не допускается применение для защиты информации, содержащей сведения, составляющие государственную тайну.

[2)] Электронная подпись (ЭП) (в соответствии с № 63-ФЗ от 06.04.2011) — информация в электронной форме, которая присоединена к другой информации в электронной форме (подписываемой информации) или иным образом связана с такой информацией и которая используется для определения лица, подписывающего информацию (является юридически значимой).

[3)] Здесь и далее в качестве примера приведено описание взаимодействия с СКЗИ «КриптоПро CSP».

4) fly-csp-preview — расширение, обеспечивающее графическое представление на электронном документе отметки об ЭП;

5) менеджер файлов fly-fm;

6) пакет офисных программ LibreOffice.

Непосредственное взаимодействие с СКЗИ (формирование управляющих команд, получение и отображение результата работы СКЗИ с использованием программы cryptcp) осуществляет программный модуль astra-csp-cryptopro.

Программный модуль fly-csp-cryptopro в целях решения функциональных задач СЭП взаимодействует с программным модулем astra-csp-cryptopro и не взаимодействует непосредственно с СКЗИ.

Менеджер файлов fly-fm, расширение пакета офисных программ LibreOffice libreoffice-astracsp-plugin и расширение для добавления отметки об ЭП fly-csp-preview в целях решения функциональных задач СЭП взаимодействуют с программным модулем fly-csp-cryptopro и не взаимодействуют непосредственно с СКЗИ.

Пакет офисных программ LibreOffice взаимодействует с расширением пакета офисных программ LibreOffice libreoffice-astracsp-plugin и не взаимодействует непосредственно с СКЗИ.

В процессе функционирования СЭП осуществляется вызов перечисленных в 10.5.4 программ из состава ОС для предварительного просмотра электронных документов, формируются управляющие команды для СКЗИ с целью вызова функций проверки или создания ЭП, осуществляется отображение получаемого после выполнения вызываемых функций результата.

Обращение к программному модулю fly-csp-cryptopro осуществляется:

1) менеджером файлов fly-fm (с использованием соответствующего пункта контекстного меню);

2) пользователем путем интерактивного запуска приложения с использованием основного меню ОС («Утилиты — Электронная подпись КриптоПро»);

3) расширением пакета офисных программ LibreOffice, запуск которого инициируется путем вызова пользователем соответствующего пункта меню графического интерфейса пакета офисных программ LibreOffice.

Обращение к программному модулю astra-csp-cryptopro осуществляется пользователем посредством интерфейса командной строки и прикладным программным обеспечением путем запуска на выполнение программного модуля с соответствующими параметрами.

10.5.2. Условия применения

В качестве СКЗИ разрешается использовать только КриптоПро CSP версии 4.0 R4 (исполнения 1-Base и 2-Base), КриптоПро CSP версии 5.0 (исполнения 1-Base и 2-Base) и КриптоПро CSP версии 5.0 R2 (исполнения 1-Base и 2-Base).

СКЗИ КриптоПро CSP в исполнении 2-Base должно использоваться с аппаратно-программным модулем доверенной загрузки.

При эксплуатации СКЗИ необходимо соблюдать требования и рекомендации эксплуатационной документации на СКЗИ, в частности требования по защите от несанкционированного доступа и по криптографической защите, а также требования по поддерживаемым СКЗИ аппаратно-программным платформам. В частности, при использовании СЭП со встроенным СКЗИ необходимо проведение проверки программного обеспечения BIOS ЭВМ, на которых предполагается функционирование СКЗИ и СЭП, на соответствие методическим документам ФСБ России в области исследований программного обеспечения BIOS.

Контроль целостности СКЗИ и СЭП должен выполняться с использованием механизма замкнутой программной среды ОС или с использованием стандартных средств контроля целостности КриптоПро CSP.

Из числа компонентов СЭП контролю целостности подлежит программный модуль `astra-csp-cryptopro`.

Перечень компонентов СКЗИ, подлежащих контролю целостности, приведен в эксплуатационной документации на СКЗИ.

Подробный порядок настройки механизма контроля целостности описан в эксплуатационной документации на ОС и используемое СКЗИ.

10.5.3. Установка

Менеджер файлов `fly-fm` и пакет офисных программ LibreOffice из состава ОС устанавливаются в соответствии с эксплуатационной документацией ОС.

Для использования СЭП дополнительно в системе должны быть установлены следующие программные компоненты (описание компонентов приведено в 10.5.1) из состава ОС:

1) программный модуль `astra-csp-cryptopro`;
2) программный модуль `fly-csp-cryptopro`;
3) расширение `libreoffice-astracsp-plugin`.

Для установки программного модуля и расширения необходимо в терминале последовательно выполнить следующие команды от имени администратора:

```
apt update
apt install astra-csp-cryptopro
apt install fly-csp-cryptopro
apt install libreoffice-astracsp-plugin
```

При установке расширения `libreoffice-astracsp-plugin` проводится проверка наличия в информационной системе установленного программного модуля `fly-csp-cryptopro` и пакета офисных программ LibreOffice.

При запуске программных модулей `fly-csp-cryptopro` и `astra-csp-cryptopro` проводится проверка наличия в информационной системе установленного программного обеспечения СКЗИ и пакета офисных программ LibreOffice.

Дополнительно возможно установить расширение `fly-csp-preview` (описание расширения приведено в 10.5.1), выполнив в терминале команду от имени администратора:

```
apt install fly-csp-preview
```

Программные модули `astra-csp-cryptopro` и `fly-csp-cryptopro` разработаны на языке программирования Python версии 3 и включают в свой состав:

1) основной файл программы `/usr/bin/astra-csp-cryptopro`;
2) файл графической программы `/usr/bin/fly-csp-cryptopro`;
3) дополнительные файлы и библиотеки в каталогах `/usr/lib/python3/dist-packages/astra_csp_cryptopro/` и `/usr/lib/python3/dist-packages/astra_csp-0.1.0.egg-info/`;
4) файлы с указанием значений параметров СЭП, задаваемых по умолчанию:
   а) `/usr/share/astra-csp/astra-csp.yaml` — общие параметры программных модулей;
   б) `/usr/share/astra-csp/fly-csp.yaml` — параметры программного модуля `fly-csp-cryptopro`;
   в) `/usr/share/astra-csp/cryptopro/astra-csp-cryptopro.yaml` — порядок взаимодействия программного модуля `astra-csp-cryptopro` с СКЗИ;
   г) `/usr/share/astra-csp/cryptopro/fly-csp-cryptopro.yaml` — порядок взаимодействия программного модуля `fly-csp-cryptopro` с СКЗИ;
5) ярлык для запуска `/usr/share/flyfm/actions/fly-csp-cryptopro.desktop`.

Файл /usr/share/astra-csp/astra-csp.yaml содержит следующие параметры и их значения по умолчанию:

1) максимально допустимый для предоставления СЭП уровень конфиденциальности:

```
max_mac_level: 1
```

2) максимальное время ожидания выполнения запросов к СКЗИ:

```
max_csp_time_waiting: 30000
```

3) перечень расширений файлов электронных документов, для которых разрешено предоставление СЭП:

```
file_extensions_filter:
...
extensions:  ['.pdf', '.png', '.jpg', '.html', '.xml', '.txt', '.odt',
'.ods', '.odp', '.odg', '.doc', '.xls', '.ppt', '.docx', '.xlsx',
'.pptx']
```

4) отключение (false) или включение (true) действия перечня, заданного в пункте 3) перечисления, по умолчанию false:

```
file_extensions_filter:
enable: false
```

Установка значения false допускается только для целей тестирования и апробации технических решений с применением СЭП в информационной системе.

В файле /usr/share/astra-csp/fly-csp.yaml приведен перечень mime-типов файлов электронных документов, для просмотра которых будет проведена предварительная конвертация во временный файл формата PDF:

```
pdf_conversion_mime_types:
- application/vnd.oasis.opendocument.text
- application/vnd.oasis.opendocument.spreadsheet
- application/vnd.oasis.opendocument.presentation
- application/vnd.oasis.opendocument.graphics
- application/msword
- application/vnd.ms-word
- application/vnd.ms-excel
- application/vnd.ms-powerpoint
- application/vnd.openxmlformats-officedocument.wordprocessingml.document
- application/vnd.openxmlformats-officedocument.spreadsheetml.sheet
- application/vnd.openxmlformats-officedocument.presentationml.presentation
- text/plain
```

В случае, если mime-тип выбранного файла электронного документа включен в список, определенный в файле /usr/share/astra-csp/fly-csp.yaml, будет проведена предварительная конвертация файла электронного документа в файл формата PDF.

Файл /usr/share/astra-csp/cryptopro/astra-csp-cryptopro.yaml содержит параметр, задающий адрес источника доверенного времени:

```
time_address: 'http://testca2012.cryptopro.ru/tsp/tsp.srf'
```

Данный источник допустимо использовать только для выполнения тестирования. При эксплуатации СЭП источник доверенного времени должен быть задан в файле /etc/astra-csp/astra-csp-cryptopro.yaml (если файл отсутствует, то необходимо создать его) в качестве значения параметра time_address:

```
time_address: <адрес_источника_доверенного_времени>
```

ВНИМАНИЕ! Внесение изменений в любые файлы из состава СЭП, расположенные в каталоге /usr/, в т.ч. в /usr/share/astra-csp/, запрещено. При необходимости переопределить значения параметров, заданных по умолчанию в файлах в каталоге /usr/share/astra-csp/, требуется изменить значения параметров в файлах с аналогичными именами, расположенных в каталоге /etc/astra-csp/ (если файл с таким же именем отсутствует, то его следует создать). Переопределять значения параметров разрешено только администратору.

ВНИМАНИЕ! Добавление в конфигурационный файл mime-типов, отличных от перечисленных, без проведения дополнительных исследований запрещено.

Для изменения параметров, указанных в файлах /usr/share/astra-csp/astra-csp.yaml и /usr/share/astra-csp/fly-csp.yaml, необходимо создать файлы /etc/astra-csp/astra-csp.yaml и /etc/astra-csp/fly-csp.yaml, в которых требуется указать только изменяемые параметры с новыми значениями. Для остальных параметров будут использоваться значения, заданные по умолчанию в файлах /usr/share/astra-csp/astra-csp.yaml и /usr/share/astra-csp/fly-csp.yaml.

Также возможно изменить значения некоторых параметров, указанных в файлах /usr/share/astra-csp/astra-csp.yaml и /usr/share/astra-csp/fly-csp.yaml, с использованием графической утилиты fly-csp-cryptopro (см. электронную справку). При этом файлы /etc/astra-csp/astra-csp.yaml и /etc/astra-csp/fly-csp.yaml с измененными значениями параметров будут созданы автоматически.

10.5.4. Просмотр электронного документа

Просмотр документа при создании и проверке электронной подписи при работе СЭП осуществляется с использованием программных средств, включенных в состав ОС. Доступен предварительный просмотр электронных документов, хранящихся в файлах следующих форматов:

1) PDF — для просмотра используется программа Okular;
2) PNG, JPG — для просмотра используется программа gwenview;
3) HTML, XML — для просмотра используется браузер из состава ОС, заданный в качестве браузера по умолчанию.

Для предварительного просмотра электронных документов, хранящихся в файлах форматов TXT, ODT, ODS, ODP, ODG, DOC, XLS, PPT, DOCX, XLSX и PPTX, осуществляется их автоматическая предварительная конвертация в формат PDF путем вызова средств из состава пакета офисных программ LibreOffice и отображение пользователю.

Перечень mime-типов файлов, предполагающих для просмотра предварительную конвертацию в формат PDF, задается в конфигурационном файле в соответствии с описанием в 10.5.3 и приведен в таблице 39.

Таблица 39

Mime-тип файла	Описание
text/plain	Текстовый документ TXT
application/vnd.oasis.opendocument.text	Текстовый документ ODT
application/vnd.oasis.opendocument.spreadsheet	Электронная таблица ODS
application/vnd.oasis.opendocument.presentation	Презентация ODP
application/vnd.oasis.opendocument.graphics	Графический документ ODG
application/msword	Текстовый документ DOC
application/vnd.ms-word	Текстовый документ DOC
application/vnd.ms-excel	Электронная таблица XLS
application/vnd.ms-powerpoint	Презентация PPT
application/vnd.openxmlformats-officedocument.wordprocessingml.document	Текстовый документ DOCX
application/vnd.openxmlformats-officedocument.spreadsheetml.sheet	Электронная таблица XLSX
application/vnd.openxmlformats-officedocument.presentationml.presentation	Презентация PPTX

10.5.5. Проверка уровня конфиденциальности сессии

В целях исключения использования СКЗИ при обработке информации, содержащей сведения, составляющие государственную тайну, перед началом работы выполняется проверка допустимого для предоставления СЭП уровня конфиденциальности путем сравнения уровня конфиденциальности сессии текущего пользователя с максимально допустимым значением, заданным в конфигурационном файле.

Максимально допустимый для предоставления СЭП уровень конфиденциальности задается в конфигурационном файле `/usr/share/astra-csp/astra-csp.yaml` параметром `max_mac_level`, а также в графической утилите `fly-csp-cryptopro` (см. электронную справку). По умолчанию установлено значение `max_mac_level: 1` и может быть изменено администратором безопасности. Политикой управления доступом и организационно-распорядительными документами по защите информации объекта информатизации (объекта вычислительной техники) должен быть предусмотрен явный запрет внесения изменений в конфигурационный файл любыми пользователями, за исключением ответственного за защиту информации администратора безопасности, зарегистрированного в ОС как привилегированный пользователь.

В случае, если уровень конфиденциальности текущей сессии пользователя превышает уровень, заданный в конфигурационном файле, при запуске программы `fly-csp-cryptopro` или `astra-csp-cryptopro` будет выведено сообщение об ошибке и программа завершит свою работу.

11. НАДЕЖНОЕ ФУНКЦИОНИРОВАНИЕ

11.1. Восстановление ОС после сбоев и отказов

Основными причинами нарушения процесса функционирования СЗИ ОС являются сбои оборудования, приведшие к различным повреждениям ФС. К таковым относятся: сбои электропитания, повреждения носителей информации (жестких дисков), повреждения соединительных кабелей.

11.1.1. Восстановление файловой системы

В процессе перезагрузки после сбоя ОС автоматически выполнит программу проверки и восстановления ФС — `fsck`. Если повреждения ФС окажутся незначительными, то ее выполнения достаточно для обеспечения целостности ФС.

В случае обнаружения серьезных повреждений ФС данная программа может предложить перезагрузить компьютер в однопользовательский режим и произвести запуск программы `fsck` вручную. Администратор, контролирующий процесс загрузки ОС, после сбоя должен следовать инструкциям, выдаваемым программой `fsck`. Описание программы приведено на справочной странице `man fsck`.

После завершения загрузки ОС следует проверить целостность файлов с помощью программы контроля целостности. Если в результате проверки найдутся поврежденные или измененные файлы, особенно в каталоге `/etc` и его подкаталогах, то следует восстановить поврежденные файлы из резервной копии. Инструменты создания резервных копий и восстановления из них описаны в документе РУСБ.10015-01 95 01-1, а также в 11.1.3 и 11.1.4.

Если сбой привел к выходу из строя жестких дисков, то следует заменить вышедшее из строя оборудование и переустановить ОС с DVD-диска с дистрибутивом, а пользовательские данные восстановить из резервной копии.

Если восстановление ФС не привело к работоспособности ОС или из-за серьезности сбоя нет возможности использовать инструмент восстановления ФС, то возможно запустить ОС в режиме восстановления в соответствии с 11.1.2.

11.1.2. Режим восстановления ОС

После серьезного повреждения ФС, когда компьютер невозможно перезагрузить, существует возможность восстановления без переустановки ОС. Для этого необходимо:

1) загрузиться с установочного DVD-диска или USB-носителя;
2) в процессе загрузки нажать клавишу **<C>** для входа в меню программы установки;

3) в окне программы установки нажать **<F10>**, в открывшемся меню выбрать «Выйти в консоль» и нажать **<Enter>**. Будет выполнен переход в режим командной строки под управлением ядра, загруженного с DVD-диска или USB-носителя;

4) определить имя раздела, в который была установлена ОС, для этого выполнить команду:

```
blkid
```

На экране монитора отобразится информация о разделах жесткого диска (если в результате ввода команды на экране монитора нет информации о разделах диска, то повреждения слишком серьезны и необходима полная переустановка системы).

Пример

Вывод после выполнения команды `blkid`:

```
/dev/sr0: BLOCK_SIZE="2048" UUID="2024-03-25-20-02-17-00"
LABEL="Astra 1.8_x86-64 amd64" TYPE="iso9660" PTTYPE="dos"
/dev/loop0: TYPE="squashfs"

/dev/sda2: LABEL="system" UUID=
     "4064eb0c-d416-4bca-8b0c-34e2f1729cad"
BLOCK_SIZE="4096" TYPE="ext4"
PARTUUID="edc7b1f6-2aa3-456f -8cad-435cac7d0cs8"

/dev/sda1: UUID="FBEE-2F9B" BLOCK_SIZE="512" TYPE="vfat"
PARTUUID="332f2bb3-afce-457b-9c6b-cca8d3cea818"
```

В приведенном примере ОС была установлена в раздел `/dev/sda2`;

5) запустить автоматическую проверку и восстановление ФС, выполнив команду:

```
fsck.ext4 -p -f -c /dev/<имя раздела>
```

Пример

Команда для проверки раздела `/dev/sda2`:

```
fsck.ext4 -p -f -c /dev/sda2
```

Пример вывода после выполнения команды:

```
system: Updating bad block inode.

system: 318177/2297456 files (0.2\% non-contiguous),
4157309/9186816 blocks
```

6) по окончании проверки извлечь DVD-диск (USB-носитель) с дистрибутивом ОС и перезагрузить компьютер. Для этого можно воспользоваться командой `reboot`.

11.1.3. Комплекс программ Bacula

Bacula обеспечивает поддержку сохранения расширенных атрибутов каталогов и файлов и, при необходимости, их последующее восстановление.

Описание комплекса программ Bacula, а также его установки и настройки приведено в документе РУСБ.10015-01 95 01-1.

ВНИМАНИЕ! Для восстановления объектов ФС с установленными мандатными атрибутами необходимо запустить консоль управления Bacula с PARSEC-привилегией `0x1000` (`PARSEC_CAP_UNSAFE_SETXATTR`). Привилегия может быть получена с использованием утилиты `execaps`:

```
sudo execaps -c 0x1000 -- bconsole
```

ВНИМАНИЕ! После восстановления объектов ФС с установленными мандатными атрибутами необходимо выполнить перемонтирование ФС, в которой восстанавливались объекты, или перезагрузить ОС.

11.1.4. Инструмент командной строки `tar`

ВНИМАНИЕ! Работа с мандатными атрибутами и атрибутами аудита при использовании различных инструментов командной строки для создания резервных копий требует использования параметров сохранения расширенных атрибутов (как правило вида `--xattrs`).

ВНИМАНИЕ! Перед восстановлением мандатных атрибутов файлов из резервных копий необходимо от имени учетной записи администратора выполнить команду:

```
echo 1 | sudo tee /parsecfs/unsecure_setxattr
```

ВНИМАНИЕ! Для восстановления мандатных атрибутов файлов из резервных копий процесс должен иметь PARSEC-привилегию `0x1000` (`PARSEC_CAP_UNSAFE_SETXATTR`). Привилегия может быть получена с использованием инструмента командной строки `execaps`:

```
sudo execaps -c 0x1000 -- tar
```

ВНИМАНИЕ! Восстановление расширенных атрибутов файлов с использованием `unsecure_setxattr` возможно только в случае, если атрибуты восстанавливаются с помощью системного вызова `setxattr` путем установки атрибута `security.PDPL`. Использование `unsecure_setxattr` не влияет на возможность изменения мандатных атрибутов файлов системными вызовами `pdpl_set_path`, `pdpl_set_fd`.

После восстановления из резервных копий файлов с мандатными атрибутами необходимо выполнить команду:

```
echo 0 | sudo tee /parsecfs/unsecure_setxattr
```

Пример

Использование `tar`, при котором будут восстановлены расширенные атрибуты каталогов и файлов, вложенных в архив.

До использования утилиты должен быть создан пользователь `user1`, для которого заданы мандатные атрибуты, и пользователь должен выполнить вход в систему.

Создать от имени учетной записи администратора архив домашнего каталога пользователя с помощью команды:

```
sudo tar --xattrs --acls -cvzf /opt/home.tgz /home/.pdp/user1
```

где `--xattrs` — включает поддержку расширенных атрибутов;
`--acls` — включает поддержку POSIX ACL;
`-cvzf` — обеспечивают, соответственно, создание архива (`create`), включение режима отображения обрабатываемых файлов (`verboze`), применение метода сжатия (`gzip`), указание файла (`file`);
`/opt/home.tgz` — задает место расположения созданного архива и его имя;
`/home/.pdp/user1` — определяет каталоги или файлы для добавления в архив.

Перед восстановлением необходимо выполнить следующую команду, устанавливающую для параметра защиты файловой системы `/parsecfs/unsecure_setxattr` значение «1», разрешающее применять привилегию `PARSEC_CAP_UNSAFE_SETXATTR`:

```
echo 1 | sudo tee /parsecfs/unsecure_setxattr
```

Выполнить восстановление с помощью команды:

```
sudo execaps -c 0x1000 -- tar --xattrs \
    --xattrs-include=security.{PDPL,AUDIT,DEF_AUDIT} \
    --acls -xvf /opt/home.tgz -C /opt/home2/
```

где `--xattrs-include=security.PDPL,AUDIT,DEF_AUDIT` — определяет подключаемый шаблон восстановления расширенных атрибутов (мандатных атрибутов, атрибутов аудита и атрибутов аудита по умолчанию) для параметра `xattrs`;

`-xvf` — обеспечивает, соответственно, извлечение из архива (`extract`), включение режима отображения обрабатываемых файлов (`verbose`), указание файла (`file`).

После восстановления необходимо выполнить команду, устанавливающую запрет на применение привилегии `PARSEC_CAP_UNSAFE_SETXATTR` (настройка ОС по умолчанию):

```
echo 0 | sudo tee /parsecfs/unsecure_setxattr
```

Дополнительная информация по команде `tar` приведена в РУСБ.10015-01 95 01-1.

11.2. Восстановление в режиме «Мобильный»

При функционировании ОС в режиме «Мобильный» в случае сбоя возможно восстановление ОС с сохранением данных в каталогах `/opt` и `/home` путем выполнения обновления с установочного USB-носителя или из образа в формате IMG (см. РУСБ.10015-01 95 01-1).

В случае ошибки или отказа ОС возможно сбросить ее настройки к значениям по умолчанию (на состояние после установки ОС), при этом все данные на устройстве будут удалены.

Для сброса настроек необходимо выполнить команду:

```
sudo astra-mobile-factory-reset
```

Также возможно выполнить сброс настроек в графическом интерфейсе:

1) открыть панель быстрого доступа и нажать кнопку **[Настройки]**;
2) выбрать пункт «Обновление системы»;
3) нажать кнопку **[Сбросить к заводским настройкам]** и затем в предупреждении нажать **[Я уверен]**;
4) в окне аутентификации ввести пароль администратора и нажать **[Да]**;
5) в окне программы начальной настройки:
   а) нажать **[Начать]**;
   б) в окне «Пароль администратора» задать пароль для учетной записи `administrator` и перейти к следующей настройке;

в) в окне «Уровни защищенности» из выпадающего списка выбрать уровень защищенности (см. РУСБ.10015-01 95 01-1) и при необходимости включить функции безопасности, затем перейти к следующей настройке;

г) для применения настроенной конфигурации ОС нажать **[Сохранить]** и подтвердить действие в открывшемся диалоговом окне — после применения настроек ОС будет перезапущена. Если необходимо изменить выбранные настройки, следует нажать **[Назад]**. Для отмены настроек и закрытия программы начальной настройки нажать **[Отменить]** — при этом устройство будет выключено, а при следующем включении устройства будет запущена программа начальной настройки.

12. ФИЛЬТРАЦИЯ СЕТЕВОГО ПОТОКА

12.1. Включение фильтрации сетевого потока

Фильтрация сетевого потока и контроль сетевого трафика, проходящего через компьютер, осуществляется с помощью фильтра сетевых пакетов.

Запуск фильтрации пакетов обеспечивается межсетевым экраном ufw. Межсетевой экран позволяет создавать правила фильтрации используя технологию `iptables`.

Управление межсетевым экраном осуществляется с помощью графической утилиты `gufw` или с помощью инструмента командной строки `astra-ufw-control`.

Для включения межсетевого экрана с помощью инструмента командной строки выполнить команду:

```
astra-ufw-control enable
```

Для выключения межсетевого экрана с помощью инструмента командной строки выполнить команду:

```
astra-ufw-control disable
```

Проверка состояния межсетевого экрана выполняется с помощью команды:

```
ufw status
```

Результат выполнения команды:

- `active` — межсетевой экран включен;
- `inactive` — межсетевой экран выключен.

12.2. Фильтр сетевых пакетов `iptables`

Фильтр сетевых пакетов реализован на основе технологии `iptables` и позволяет выполнять следующие задачи:

1) фильтрацию пакетов — это механизм, который на основе заданных правил, разрешает или запрещает передачу информации, проходящей через него, с целью ограждения подсети от внешнего доступа, или, наоборот, для недопущения выхода наружу. Фильтр пакетов может определять правомерность передачи информации только на основе заголовков IP-пакетов, а также может анализировать и их содержимое, т. е. использовать данные протоколов более высокого уровня;

2) трансляцию сетевых адресов (т. н. «маскарадинг») — это подмена некоторых параметров в заголовках IP-пакетов. Используется для сокрытия реальных IP-адресов компьютеров защищаемой подсети, а также для организации доступа из подсети с компьютерами, не имеющими реальных IP-адресов, к глобальной сети;

3) прозрачное проксирование — это переадресация пакетов на другой порт компьютера. Обычно используется для того, чтобы пользователи из подсети использовали proxy-сервер маршрутизатора без дополнительного конфигурирования их клиентских программ.

Настройка рассмотренных механизмов (фильтрация пакетов, трансляция сетевых адресов и прозрачное проксирование) выполняется командой `iptables`.

12.3. Формирование правил

Каждое правило — это строка, содержащая критерии, определяющие применимость данного правила к пакету, и действие, которое необходимо выполнить в случае выполнения критерия.

Правила имеют следующий синтаксис:

```
iptables [-t table] command [match] [target/jump]
```

Если в правило не включается спецификатор таблицы `[-t table]`, то по умолчанию предполагается использование таблицы `filter`. Если необходимо использовать другую таблицу, то это требуется указать явно. Спецификатор таблицы можно указывать в любом месте строки правила, однако принято указывать в начале правила.

Непосредственно после имени таблицы должна стоять команда. Если спецификатора таблицы нет, то команда всегда должна стоять первой. Команда определяет действие `iptables`, например, вставить, добавить в конец цепочки или удалить правило.

В разделе `match` указываются критерии проверки, по которым определяется применимость данного правила к пакету. Возможно указать в качестве критерия IP-адрес источника пакета или сети, сетевой интерфейс и т. п.

Параметр `target` указывает, какое действие должно быть выполнено при условии выполнения критериев в правиле. Возможны такие действия как: передача пакета ядром в другую цепочку правил, «сбросить» пакет и забыть про него, выдать на источник сообщение об ошибке и т. п.

Созданные правила фильтрации сохраняются и применяются только до перезагрузки ОС, после перезагрузки ОС требуется повторное формирование правил.

Для обеспечения возможности восстановления правил фильтрации после перезагрузки ОС необходимо:

1) создать файл `/etc/iptables.rules` для сохранения правил фильтрации:

```
sudo touch /etc/iptables.rules
```

2) установить для созданного файла ограничения на чтение, так как из него можно получить информацию об открытых сетевых портах, которая может быть использована для атак на компьютер:

```
sudo chmod 640 /etc/iptables.rules
```

3) сохранить существующие правила фильтрации в файле `/etc/iptables.rules`:

```
sudo iptables-save | sudo tee /etc/iptables.rules > /dev/null
```

4) создать сценарий `/etc/network/if-pre-up.d/iptables` со следующим содержимым:

```
iptables-restore < /etc/iptables.rules
exit 0
```

5) создать сценарий `/etc/network/if-post-down.d/iptables` со следующим содержимым:

```
touch /etc/iptables.rules
chmod 640 /etc/iptables.rules
iptables-save > /etc/iptables.rules
exit 0
```

6) сделать созданные сценарии исполняемыми:

```
sudo chmod +x /etc/network/if-post-down.d/iptables \
     /etc/network/if-pre-up.d/iptables
```

Примечание. Для сохранения правил фильтрации рекомендуется выполнять пункты 1)–3) перечисления на стр. 239 после каждого изменения правил. Сценарий, приведенный в пункте 5) перечисления, сохраняет текущую версию правил автоматически только при штатном выключении компьютера. При нештатном выключении компьютера или сбое изменения в правилах фильтрации могут быть утеряны.

12.4. Порядок прохождения таблиц и цепочек

Когда пакет приходит на сетевой фильтр, то он сначала попадает на сетевое устройство, перехватывается соответствующим драйвером и далее передается в ядро. Затем пакет проходит несколько таблиц, прежде чем он будет передан далее. На каждом этапе пакет

может быть остановлен. После прохождения таблиц пакет передается либо локальному приложению, либо перенаправляется на другой компьютер.

В таблице `filter` цепочку `FORWARD` проходят все пакеты, которые движутся через сетевой фильтр. Порядок следования пакета приведен в таблице 40.

Таблица 40

Шаг	Таблица	Цепочка	Описание
1	=	=	Кабель
2	=	=	Сетевой интерфейс (например, `eth0`)
3	`mangle`	`PREROUTING`	Используется для внесения изменений в заголовок пакета, например, для изменения битов `TOS` и пр.
4	`nat`	`PREROUTING`	Используется для трансляции сетевых адресов `DNAT`. `SNAT` выполняется позднее, в другой цепочке. Любого рода фильтрация в этой цепочке может производиться только в исключительных случаях
5	=	=	Принятие решения о дальнейшей маршрутизации, т. е. в этой точке решается, куда пойдет пакет — локальному приложению или на другой узел сети
6	`filter`	`FORWARD`	Попадают только те пакеты, которые идут на другой компьютер. Вся фильтрация транзитного трафика должна выполняться здесь. Через эту цепочку проходит трафик в обоих направлениях, поэтому обязательно учитывать это обстоятельство при написании правил фильтрации
7	`nat`	`POSTROUTING`	Предназначена в первую очередь для `SNAT`. Не использовать для фильтрации без особой необходимости. Здесь же выполняется и маскарадинг
8	=	=	Выходной сетевой интерфейс (например, `eth1`)
9	=	=	Кабель

Пакеты, предназначенные локальному процессу/приложению, проходят через цепочку `INPUT`, а не через `FORWARD`. В таблице 41 представлен порядок движения пакета.

Таблица 41

Шаг	Таблица	Цепочка	Описание
1	=	=	Кабель
2	=	=	Входной сетевой интерфейс (например, `eth0`)
3	`mangle`	`PREROUTING`	Обычно используется для внесения изменений в заголовок пакета, например, для установки битов `TOS` и пр.
4	`nat`	`PREROUTING`	Преобразование адресов `DNAT`. Фильтрация пакетов здесь допускается только в исключительных случаях
5	=	=	Принятие решения о маршрутизации

Окончание таблицы 41

Шаг	Таблица	Цепочка	Описание
6	filter	INPUT	Фильтрация входящего трафика. Все входящие пакеты, адресованные локальному приложению, проходят через эту цепочку, независимо от того, с какого интерфейса они поступили
7	=	=	Локальный процесс/приложение

В таблице 42 представлен порядок движения пакетов, созданных локальными процессами.

Таблица 42

Шаг	Таблица	Цепочка	Описание
1	=	=	Локальный процесс
2	mangle	OUTPUT	Внесение изменений в заголовок пакета. Фильтрация, выполняемая в этой цепочке, может иметь негативные последствия
3	filter		
4	=	=	Принятие решения о маршрутизации. Здесь решается — куда пойдет пакет дальше
5	nat	POSTROUTING	Здесь выполняется SNAT. Не следует в этой цепочке производить фильтрацию пакетов во избежание нежелательных побочных эффектов. Однако и здесь можно останавливать пакеты, применяя политику по умолчанию — DROP
6	=	=	Сетевой интерфейс (например, eth0)
7	=	=	Кабель

mangle

В таблице mangle не следует производить фильтрацию, маскировку или преобразование адресов (DNAT, SNAT), в ней допускается выполнять действия, приведенные в таблице 43.

Таблица 43

Действие	Описание
TOS	Выполняет установку битов поля TOS. Это поле используется для назначения сетевой политики обслуживания пакета, т. е. задает желаемый вариант маршрутизации
TTL	Используется для установки значения поля TTL пакета
MARK	Устанавливает специальную метку на пакет, которая затем может быть проверена другими правилами в iptables или другими программами, например, iproute2. С помощью меток можно управлять маршрутизацией пакетов, ограничивать трафик и т. п.

Таблица mangle имеет следующие цепочки:

- PREROUTING — используется для внесения изменений на входе в сетевой фильтр перед принятием решения о маршрутизации;

- INPUT — для пакетов, поступающих в саму систему;
- FORWARD — для изменения пакетов, маршрутизирующихся через систему;
- OUTPUT — для внесения изменений в пакеты, поступающие от приложений внутри сетевого фильтра;
- POSTROUTING — для изменения пакетов после маршрутизации и перед их отправкой.

nat

Только первый пакет из потока проходит через цепочки таблицы nat. Трансляция адресов или маскировка применяются ко всем последующим пакетам в потоке автоматически. Для таблицы nat характерны действия, приведенные в таблице 44.

Таблица 44

Действие	Описание
DNAT	Производит преобразование адресов назначения в заголовках пакетов. Другими словами, этим действием перенаправляются пакеты на другие адреса, отличные от указанных в заголовках пакетов
SNAT	Используется для изменения исходных адресов пакетов. С помощью этого действия можно скрыть структуру локальной сети
MASQUERADE	Применяется в тех же целях, что и SNAT, но в отличие от последней дает более сильную нагрузку на систему. Происходит это потому, что каждый раз, когда требуется выполнение этого действия, производится запрос IP-адреса для указанного в действии сетевого интерфейса, в то время как для SNAT IP-адрес указывается непосредственно. Однако благодаря такому отличию, MASQUERADE может работать в случаях с динамическим IP-адресом

Таблица nat имеет следующие цепочки:

- PREROUTING — используется для внесения изменений в пакеты на входе в сетевой фильтр;
- INPUT — используется для изменения пакетов, предназначенных для локальных сокетов;
- OUTPUT — используется для преобразования пакетов, созданных приложениями внутри сетевого фильтра, перед принятием решения о маршрутизации;
- POSTROUTING — используется для изменения пакетов после маршрутизации и перед их отправкой.

filter

В таблице filter содержатся наборы правил для выполнения фильтрации пакетов. Пакеты могут пропускаться далее либо отвергаться в зависимости от их содержимого.

В таблице filter можно выполнить DROP, LOG, ACCEPT или REJECT. Таблица filter имеет следующие цепочки:

- INPUT — используется для прохождения пакетов, которые предназначены локальным приложениям (сетевому фильтру);
- FORWARD — используется для фильтрации пакетов, идущих транзитом через сетевой фильтр;
- OUTPUT — используется для фильтрации исходящих пакетов, сгенерированных приложениями на самом сетевом фильтре.

12.5. Механизм трассировки соединений

Механизм трассировки соединений является частью сетевого фильтра iptables и устроен так, чтобы netfilter мог получить информацию о состоянии конкретного соединения. Наличие этого механизма позволяет создавать более надежные наборы правил.

В пределах iptables соединение может иметь одно из четырех базовых состояний: NEW, ESTABLISHED, RELATED и INVALID. Для управления пакетами на основе их состояния используется критерий --state. Трассировщик определяет четыре основных состояния каждого TCP- или UDP-пакета и некоторые дополнительные характеристики. Для TCP- и UDP-пакетов — это IP-адреса отправителя и получателя, порты отправителя и получателя.

Трассировка производится в цепочке PREROUTING. Это означает, что iptables производит все вычисления, связанные с определением состояния, в пределах этой цепочки. Когда отправляется инициирующий пакет в потоке, то ему присваивается состояние NEW, а когда возвращается пакет ответа, то состояние соединения изменяется на ESTABLISHED и т. д.

Таблица трассировки

Таблица трассировщика записана в файле /proc/net/ip_conntrack. В ней содержится список всех активных соединений.

Если модуль ip_conntrack загружен, то команда cat /proc/net/ip_conntrack должна вывести:

```
tcp 6 117 SYN_SENT src=192.168.1.6 dst=192.168.1.9 sport=32775 dport=22
[UNREPLIED] src=192.168.1.9 dst=192.168.1.6 sport=22 dport=32775 use=2
```

В примере вывода команды содержится вся информация, которая известна трассировщику по конкретному соединению.

В начале указано название протокола, в данном случае — tcp. Далее следует некоторое число в обычном десятичном представлении. После него следует число, определяющее

«время жизни» записи в таблице (т. е. количество секунд, через которое информация о соединении будет удалена из таблицы). В приведенном примере запись в таблице будет храниться еще 117 с, если через это соединение более не проследует ни одного пакета, в противном случае это значение будет установлено в значение по умолчанию для заданного состояния. Это число уменьшается на 1 каждую секунду. Далее следует фактическое состояние соединения. В примере это состояние имеет значение `SYN_SENT`. Внутреннее представление состояния несколько отличается от внешнего. Значение `SYN_SENT` говорит о том, что через данное соединение проследовал единственный пакет TCP `SYN`. Далее расположены адреса отправителя и получателя, порты отправителя и получателя. Здесь же указано ключевое слово, которое сообщает о том, что ответного трафика через это соединение еще не было. Далее приводится дополнительная информация по ожидаемому пакету — это IP-адреса и номера портов отправителя и получателя для ожидаемого ответного пакета (те же данные, только поменявшиеся местами).

После получения ответа трассировщик снимет флаг `[UNREPLIED]` и заменит его флагом `[ASSURED]`. Этот флаг сообщает, что соединение установлено, и эта запись не будет стерта по достижении максимально возможного количества трассируемых соединений.

Максимальное количество записей, которое может содержаться в таблице, зависит от значения по умолчанию, которое может быть установлено вызовом функции `ipsysctl`.

Для объема ОЗУ 256 МБ значение соответствует 16376 записям. Можно посмотреть и изменить это значение через:

```
/proc/sys/net/ipv4/ip_conntrack_max
```

Состояния

Сетевые пакеты могут иметь несколько различных состояний в пределах ядра в зависимости от типа протокола. Однако вне ядра имеется только четыре состояния, как было сказано выше. Параметры, описывающие состояние пакета, используются в критерии `--state`. Допустимыми являются: `NEW`, `ESTABLISHED`, `RELATED` и `INVALID`.

В таблице 45 подробно рассмотрено каждое из возможных состояний и приведены необходимые комментарии.

Таблица 45

Состояние	Описание
`NEW`	Сообщает о том, что пакет является первым для данного соединения. Это означает, что это первый пакет в данном соединении, который увидел модуль трассировщика

Окончание таблицы 45

Состояние	Описание
ESTABLISHED	Говорит о том, что это не первый пакет в соединении. Для перехода в состояние ESTABLISHED необходимо, чтобы один компьютер передал пакет и получил на него ответ от другого компьютера. После получения ответа признак соединения NEW будет заменен на ESTABLISHED
RELATED	Появляется, если данное соединение связано с другим соединением, имеющим признак ESTABLISHED, т. е. соединение инициировано из уже установленного соединения, имеющего признак ESTABLISHED
INVALID	Говорит о том, что пакет не может быть идентифицирован и поэтому не может иметь определенного статуса. Это может происходить по разным причинам, например, при нехватке памяти или при получении ICMP-сообщения, которое не соответствует какому-либо известному соединению. Наилучшим вариантом было бы применение действия DROP к таким пакетам

Таблицы

Параметр `-t` указывает на используемую таблицу. По умолчанию используется таблица `filter`.

Команды

В таблице 46 приводится список команд, которые используются в `iptables`, и правила их использования. Посредством данных команд `iptables` узнает, что необходимо выполнить. Обычно предполагается одно из двух действий — это добавление нового правила в цепочку или удаление существующего правила из той или иной таблицы.

Таблица 46

Команда	Использование
`-A, --append`	Добавляет новое правило в конец заданной цепочки. Пример `iptables -A INPUT ...`

Продолжение таблицы 46

Команда	Использование

`-D, --delete`	Удаляет правило из цепочки. Команда имеет два формата записи: первый — когда задается критерий сравнения с параметром `-D`, второй — порядковый номер правила. Если задается критерий сравнения, то удаляется правило, которое имеет в себе этот критерий, если задается номер правила, то будет удалено правило с заданным номером. Счет правил в цепочках начинается с единицы.
	
	Пример
	
	`iptables -D INPUT --dport 80 -j DROP, iptables \`
	`-D INPUT 1`

`-E, --rename-chain`	Выполняет переименование пользовательской цепочки. В примере цепочка `allowed` будет переименована в цепочку `disallowed`. Эти переименования не изменяют порядок работы. Пример `iptables -E allowed disallowed` Команда должна быть указана всегда
`-F, --flush`	Сброс (удаление) всех правил из заданной цепочки (таблицы). Если имя цепочки и таблицы не указывается, то удаляются все правила во всех цепочках. Пример `iptables -F INPUT`
`-I, --insert`	Вставляет новое правило в цепочку. Число, следующее за именем цепочки, указывает номер правила, перед которым следует вставить новое правило. В примере выше указывается, что данное правило должно быть первым в цепочке `INPUT`. Пример `iptables -I INPUT 1 --dport 80 -j ACCEPT`

Продолжение таблицы 46

Команда	Использование
`-L, --list`	Вывод списка правил в заданной цепочке. В нижеприведенном примере предполагается вывод правил из цепочки `INPUT`. Если имя цепочки не указывается, то выводится список правил для всех цепочек. Формат вывода зависит от наличия дополнительных ключей в команде, например `-n`, `-v` и др. Пример `iptables -L INPUT`
`-N, --new-chain`	Создается новая цепочка с заданным именем в заданной таблице. В нижеприведенном примере создается новая цепочка с именем `allowed`. Имя цепочки должно быть уникальным и не должно совпадать с зарезервированными именами цепочек и действий (`DROP`, `REJECT` и т. п.). Пример `iptables -N allowed`
`-P, --policy`	Определяет политику по умолчанию для заданной цепочки. Политика по умолчанию определяет действие, применяемое к пакетам, не попавшим под действие ни одного из правил в цепочке. В качестве политики по умолчанию допускается использовать `DROP`, `ACCEPT` и `REJECT`. Пример `iptables -P INPUT DROP`
`-R, --replace`	Данная команда заменяет одно правило другим. В основном она используется во время отладки новых правил. Пример `iptables -R INPUT 1 -s 192.168.0.1 -j`
`-X, --delete-chain`	Удаление заданной цепочки из заданной таблицы. Удаляемая цепочка не должна иметь правил и не должно быть ссылок из других цепочек на удаляемую цепочку. Если имя цепочки не указано, то будут удалены все цепочки, определенные командой `-N` в заданной таблице. Пример `iptables -X allowed`

Окончание таблицы 46

Команда	Использование
`-Z, --zero`	Обнуление всех счетчиков в заданной цепочке. Если имя цепочки не указывается, то подразумеваются все цепочки. При использовании ключа `-v` совместно с командой `-L` на вывод будут поданы и состояния счетчиков пакетов, попавших под действие каждого правила. Допускается совместное использование команд `-L` и `-Z`. В этом случае будет выдан сначала список правил со счетчиками, а затем произойдет обнуление счетчиков

Ключи

Некоторые команды могут использоваться совместно с дополнительными ключами. Перечень ключей и их описание приведены в таблице 47.

Таблица 47

Ключ	Описание
`c, --set-counters`	Используется вместе с командами `--insert`, `--append` и `--replace` при создании нового правила для установки счетчиков пакетов и байт в заданное значение. Например, ключ `--set-counters 20 4000` установит счетчик пакетов в 20, а счетчик байт в 4000
`--line-numbers`	Используется вместе с командой `--list`, включает режим вывода номеров строк при отображении списка правил командой `--list`. Номер строки соответствует позиции правила в цепочке
`--modprobe`	Может использоваться с любой командой, определяет команду загрузки модуля ядра. Данный ключ используется в случае, если команда `modprobe` находится вне пути поиска
`n, --numeric`	Используется вместе с командой `--list`. Заставляет `iptables` выводить IP-адреса и номера портов в числовом виде, предотвращая попытки преобразовать их в символические имена
`-v, --verbose`	Используется вместе с командами `--list`, `--append`, `--insert`, `--delete` и `--replace` для повышения информативности вывода. В случае использования с командой `--list` в вывод этой команды включаются также имя интерфейса, счетчики пакетов и байт для каждого правила. Формат вывода счетчиков предполагает вывод, кроме цифр числа, еще и символьных множителей `K` (x1000), `M` (x1,000,000) и `G` (x1,000,000,000). Чтобы команда `--list` выводила полное число (без употребления множителей), требуется применять ключ `-x`, который описан ниже. При использовании с другими командами на вывод будет подан подробный отчет о произведенной операции
`-x, --exact`	Используется вместе с командой `--list`. Для всех чисел в выходных данных выводятся их точные значения без округления и без применения множителей `K`, `M`, `G`. Важно то, что данный ключ используется только с командой `--list` и не применяется с другими командами

12.6. Критерии выделения пакетов

Выделяются следующие критерии:

1) общие — критерии, которые допустимо употреблять в любых правилах. Они не зависят от типа протокола и не требуют подгрузки модулей расширения. В эту группу добавлен критерий `--protocol`, несмотря на то, что он используется в некоторых специфичных от протокола расширениях. Например, при использовании TCP-критерия необходимо использовать и критерий `--protocol`, которому в качестве дополнительного ключа передается название протокола — TCP. Однако `--protocol` сам по себе является критерием, который используется для указания типа протокола;

2) неявные — это критерии, которые подгружаются неявно и становятся доступны, например, при указании критерия `--protocol`. Существует три автоматически подгружаемых расширения: TCP-, UDP- и ICMP-критерии. Загрузка этих расширений может производиться и явным образом с помощью ключа `-m`, `--match`, например:

```
-m tcp
```

3) перед использованием вышеописанных расширений они должны быть загружены явно, с помощью ключа `-m` или `--match`. Так, например, если использовать критерии `--state`, то следует явно указать это в строке правила `-m state` левее используемого критерия. Все отличие между явными и неявными критериями заключается только в том, что первые необходимо подгружать явно, а вторые подгружаются автоматически.

12.7. Действия и переходы

Действия и переходы сообщают правилу, что необходимо выполнить, если пакет соответствует заданному критерию. Чаще всего употребляются действия `ACCEPT` и `DROP`.

Описание переходов в правилах выглядит точно так же, как и описание действий, т. е. ставится ключ `-j` и указывается название цепочки правил, на которую выполняется переход. На переходы накладывается ряд ограничений, первое — цепочка, на которую выполняется переход, должна находиться в той же таблице, что и цепочка, из которой этот переход выполняется; второе — цепочка, являющаяся целью перехода, должна быть создана до того, как на нее будут выполняться переходы.

Пример

Создать цепочку `tcp_packets` в таблице `filter` с помощью команды:

```
iptables -N tcp_packets
```

Выполнять переходы на эту цепочку:

```
iptables -A INPUT -p tcp -j tcp_packets
```

т. е., встретив пакет протокола TCP, `iptables` произведет переход на цепочку `tcp_packets` и продолжит движение пакета по этой цепочке.

Если пакет достиг конца цепочки, то он будет возвращен в вызывающую цепочку (в примере — это цепочка `INPUT`), и движение пакета продолжится с правила, следующего за правилом, вызвавшим переход. Если к пакету во вложенной цепочке будет применено действие `ACCEPT`, то автоматически пакет будет считаться принятым и в вызывающей цепочке и уже не будет продолжать движение по вызывающим цепочкам. Однако пакет пойдет по другим цепочкам в других таблицах.

Действие — это предопределенная команда, описывающая действие, которое необходимо выполнить, если пакет совпал с заданным критерием. Например, можно применить действие `DROP` или `ACCEPT` к пакету. В результате выполнения одних действий пакет прекращает свое прохождение по цепочке, например `DROP` и `ACCEPT`; в результате других, после выполнения неких операций, продолжает проверку, например `LOG`; в результате работы третьих — даже видоизменяется, например `DNAT` и `SNAT`, `TTL` и `TOS`, но также продолжает продвижение по цепочке.

ACCEPT

Если над пакетом выполняется действие `ACCEPT`, то пакет прекращает движение по цепочке (и всем вызвавшим цепочкам, если текущая цепочка была вложенной) и считается принятым, тем не менее, пакет продолжит движение по цепочкам в других таблицах и может быть отвергнут там. Действие задается с помощью ключа `-j ACCEPT`. Дополнительных ключей не имеет.

DNAT

`DNAT` используется для преобразования адреса места назначения в IP-заголовке пакета. Если пакет подпадает под критерий правила, выполняющего `DNAT`, то этот пакет и все последующие пакеты из этого же потока поменяют адрес назначения и будут переданы на требуемое устройство, компьютер или сеть.

Может выполняться только в цепочках `PREROUTING` и `OUTPUT` таблицы `nat` и во вложенных подцепочках.

Для действия `DNAT` предназначен ключ `--to-destination`.

Пример

```
iptables -t nat -A PREROUTING -p tcp -d 15.45.23.67 \
     --dport 80 -j DNAT --to-destination 192.168.1.1-192.168.1.10
```

Этот ключ указывает, какой IP-адрес должен быть подставлен в качестве адреса места назначения. В вышеприведенном примере во всех пакетах, пришедших на адрес `14.45.23.67`, адрес назначения будет изменен на один из диапазона от `192.168.1.1` до `192.168.1.10`. Все пакеты из одного потока будут направляться на один и тот же адрес, а для каждого нового потока будет выбираться один из адресов в указанном диапазоне случайным образом. Можно также определить единственный IP-адрес. Можно дополнительно указать порт или диапазон портов, на который (которые) будет перенаправлен трафик. Для этого после IP-адреса через двоеточие указать порт, например:

```
--to-destination 192.168.1.1:80
```

для указания диапазона портов:

```
--to-destination 192.168.1.1:80-100
```

Синтаксис действий `DNAT` и `SNAT` во многом схож. Указание портов допускается только при работе с протоколом `TCP` или `UDP`, при наличии параметра `--protocol` в критерии.

DROP

`DROP` сбрасывает пакет и `iptables` забывает о его существовании. Сброшенные пакеты прекращают свое движение полностью, т. е. они не передаются в другие таблицы, как это происходит в случае с действием `ACCEPT`. Следует помнить, что данное действие может иметь негативные последствия, поскольку может оставлять незакрытые сокеты как на стороне сервера, так и на стороне клиента, наилучшим способом защиты будет использование действия `REJECT`, особенно при защите от сканирования портов.

LOG

`LOG` служит для журналирования отдельных пакетов и событий. В журнал могут заноситься заголовки IP-пакетов и другая интересующая информация. Информация из журнала может быть прочитана с помощью `dmesg` или `syslogd`, либо с помощью других команд.

Ключи действия `LOG` приведены в таблице 48.

Таблица 48

Ключ	Описание
--log-level	Используется для задания уровня журналирования. Можно задать следующие уровни: debug, info, notice, warning, warn, err, error, crit, alert, emerg и panic. Ключевое слово error означает то же самое, что и err, warn — warning и panic — emerg. Приоритет определяет различия в том, как будут заноситься сообщения в журнал. Все сообщения заносятся в журнал средствами ядра. Если установить строку kern.=info /var/log/iptables в файле syslog.conf, то все сообщения из iptables, использующие уровень info, будут заноситься в файл /var/log/iptables. Однако в этот файл попадут и другие сообщения, поступающие из других подсистем, которые используют уровень info. Пример `iptables -A FORWARD -p tcp -j LOG --log-level debug`

--log-prefix	Задает префикс, который будет стоять перед всеми сообщениями iptables. Сообщения со специфичным префиксом затем легко можно найти, к примеру, с помощью grep. Префикс может содержать до 29 символов, включая пробелы.
	
	Пример
	
	`iptables -A INPUT -p tcp -j LOG --log-prefix \`
	`"INPUT packets"`

--log-tcp-sequence	Позволяет заносить в журнал номер TCP Sequence-пакета. Номер TCP Sequence идентифицирует каждый пакет в потоке и определяет порядок сборки потока. Этот ключ потенциально опасен для безопасности системы, если системный журнал разрешает доступ «на чтение» всем пользователям. Как и любой другой журнал, содержащий сообщения от iptables. Пример `iptables -A INPUT -p tcp -j LOG --log-tcp-sequence`
--log-tcp-options	Позволяет заносить в системный журнал различные сведения из заголовка TCP-пакета. Такая возможность может быть полезна при отладке. Этот ключ не имеет дополнительных параметров. Пример `iptables -A FORWARD -p tcp -j LOG --log-tcp-options`

Окончание таблицы 48

Ключ	Описание
`--log-ip-options`	Позволяет заносить в системный журнал различные сведения из заголовка IP-пакета. Во многом схож с ключом `--log-tcp-options`, но работает только с IP-заголовком. Пример `iptables -A FORWARD -p tcp -j LOG --log-ip-options`

MARK

`MARK` используется для установки меток для определенных пакетов. Это действие может выполняться только в пределах таблицы `mangle`. Установка меток обычно используется для нужд маршрутизации пакетов по различным маршрутам, для ограничения трафика и т. п. Метка пакета существует только в период времени, пока пакет не покинул брандмауэр, т. е. метка не передается по сети. Если необходимо как-то пометить пакеты, чтобы использовать маркировку на другом компьютере, то можно манипулировать битами поля `TOS`.

Ключ `--set-`для действия `MARK` устанавливает метку на пакет. После ключа `--set-mark` должно следовать целое число.

Пример

```
iptables -t mangle -A PREROUTING -p tcp --dport 22 -j MARK --set-mark 2
```

MASQUERADE

Маскарадинг подразумевает получение IP-адреса от заданного сетевого интерфейса, вместо прямого его указания, как это делается с помощью ключа `--to-source` в действии `SNAT`.

Действие `MASQUERADE` может быть использовано вместо `SNAT`, даже если имеется постоянный IP-адрес.

`MASQUERADE` допускается указывать только в цепочке `POSTROUTING` таблицы `nat`, так же как и действие `SNAT`. `MASQUERADE` имеет ключ, использование которого необязательно.

Ключ `--to-ports` для действия `MASQUERADE` используется для указания порта источника или диапазона портов исходящего пакета. Можно указать один порт, например:

```
--to-ports 1025
```

или диапазон портов:

```
--to-ports 1024-3100
```

Этот ключ можно использовать только в правилах, где критерий содержит явное указание на протокол TCP или UDP с помощью ключа `--protocol`.

Пример

```
iptables -t nat -A POSTROUTING -p TCP -j MASQUERADE --to-ports \
     1024-31000
```

REDIRECT

`REDIRECT` выполняет перенаправление пакетов и потоков на другой порт того же самого компьютера. К примеру, можно пакеты, поступающие на HTTP-порт, перенаправить на порт HTTP-прокси. Действие `REDIRECT` очень удобно для выполнения прозрачного проксирования (transparent proxying), когда компьютеры в ЛВС даже не подозревают о существовании прокси.

`REDIRECT` может использоваться только в цепочках `PREROUTING` и `OUTPUT` таблицы `nat`, а также выполняться в подцепочках.

Ключ `--to-ports` для действия `REDIRECT` определяет порт или диапазон портов назначения. Без указания ключа `--to-ports` перенаправления не происходит, т. е. пакет идет на тот порт, куда и был назначен.

Для указания одного порта назначения ввести:

```
iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-ports 8080
```

Если необходимо указать диапазон портов:

```
iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-ports 8080-8090
```

Этот ключ можно использовать только в правилах, где критерий содержит явное указание на протокол TCP или UDP с помощью ключа `--protocol`.

REJECT

`REJECT` используется, как правило, в тех же самых ситуациях, что и `DROP`, но в отличие от `DROP`, команда `REJECT` выдает сообщение об ошибке на компьютер, передавший пакет. Действие `REJECT` работает только в цепочках `INPUT`, `FORWARD` и `OUTPUT` (и во вложенных

в них цепочках). Пока существует только единственный ключ, управляющий поведением команды REJECT.

Ключ --reject-with для действия REJECT указывает, какое сообщение необходимо передать в ответ, если пакет совпал с заданным критерием. При применении действия REJECT к пакету сначала на компьютер-отправитель будет отослан указанный ответ, а затем пакет будет сброшен. Допускается использовать следующие типы ответов: icmp-net-unreachable, icmp-host-unreachable, icmp-port-unreachable, icmp-proto-unreachable, icmp-net-prohibited и icmp-host-prohibited. По умолчанию передается сообщение port-unreachable. Все вышеуказанные типы ответов являются ICMP error messages (сообщениями об ошибках). Тип ответа tcp-reset используется только для протокола TCP. Если указано значение tcp-reset, то действие REJECT передаст в ответ пакет TCP RST, который используется для закрытия TCP-соединения.

Пример

```
iptables -A FORWARD -p TCP --dport 22 -j REJECT --reject-with tcp-reset
```

RETURN

RETURN прекращает движение пакета по текущей цепочке правил и производит возврат в вызывающую цепочку, если текущая цепочка была вложенной, или, если текущая цепочка лежит на самом верхнем уровне (например, INPUT), к пакету будет применена политика по умолчанию. В качестве политики по умолчанию назначают действия ACCEPT или DROP.

SNAT

SNAT используется для преобразования сетевых адресов, т. е. изменения исходящего IP-адреса в IP-заголовке пакета. SNAT допускается выполнять только в таблице nat, в цепочке POSTROUTING. Другими словами, только здесь допускается преобразование исходящих адресов. Если первый пакет в соединении подвергся преобразованию исходящего адреса, то все последующие пакеты из этого же соединения будут преобразованы автоматически и не пойдут через эту цепочку правил.

Ключ --to-source для действия SNAT используется для указания адреса, присваиваемого пакету.

Пример

```
iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to-source \
     194.236.50.155-194.236.50.160:1024-32000
```

Указывается IP-адрес, который будет подставлен в заголовок пакета в качестве исходящего. Если необходимо перераспределить нагрузку между несколькими брандмауэрами, то можно

указать диапазон адресов, где начальный и конечный адреса диапазона разделяются дефисом, например:

```
194.236.50.155-194.236.50.160
```

Тогда конкретный IP-адрес будет выбираться из диапазона случайным образом для каждого нового потока. Дополнительно можно указать диапазон портов, которые будут использоваться только для нужд SNAT.

TOS

TOS используется для установки бит в поле TOS IP-заголовка. Поле TOS содержит восемь бит, которые используются для маршрутизации пакетов. Это одно из нескольких полей, используемых iproute2. Данное поле может обрабатываться различными маршрутизаторами с целью выбора маршрута движения пакета. Как уже указывалось выше, это поле, в отличие от MARK, сохраняет свое значение при движении по сети, а поэтому может использоваться для маршрутизации пакета. Данное действие допускается выполнять только в пределах таблицы mangle.

Ключ --set-tos для действия TOS определяет числовое значение в десятичном или шестнадцатеричном виде.

Пример

```
iptables -t mangle -A PREROUTING -p TCP --dport 22 -j TOS --set-tos 0x10
```

Поскольку поле TOS является 8-битным, то можно указать число в диапазоне от 0 до 255 (0x00–0xFF). Большинство значений этого поля никак не используется. Лучше использовать мнемонические обозначения: Minimize-Delay (16 или 0x10), Maximize-Throughput (8 или 0x08), Maximize-Reliability (4 или 0x04), Minimize-Cost (2 или 0x02) или Normal-Service (0 или 0x00). По умолчанию большинство пакетов имеют признак Normal-Service или 0. Список мнемоник можно получить, выполнив команду:

```
iptables -j TOS -h
```

TTL

TTL используется для изменения содержимого поля TTL в IP-заголовке. Один из вариантов применения этого действия — устанавливать значение поля TTL во всех исходящих пакетах в одно и то же значение.

Действие TTL можно указывать только в таблице mangle.

Ключи для действия TTL приведены в таблице 49.

Таблица 49

Ключ	Описание

--ttl-set	Устанавливает поле TTL в заданное значение. Оптимальным считается значение около 64.
	
	Пример
	
	`iptables -t mangle -A PREROUTING -o eth0 -j TTL \`
	`  --ttl-set 64`

--ttl-dec	Уменьшает значение поля TTL на заданное число. Например, пусть входящий пакет имеет значение TTL, равное 53, выполняется команда `--ttl-dec 3`. Тогда пакет покинет компьютер с полем TTL, равным 49. Сетевой код автоматически уменьшит значение TTL на 1, поэтому фактически получается: `53 - 3 - 1 = 49`. Пример `iptables -t mangle -A PREROUTING -o eth0 -j TTL --ttl-dec 1`
--ttl-inc	Увеличивает значение поля TTL на заданное число. Пусть поступает пакет с TTL, равным 53, тогда после выполнения команды `--ttl-inc 4` на выходе с компьютера пакет будет иметь TTL, равный 56, не стоит забывать об автоматическом уменьшении поля TTL сетевым кодом ядра, т. е. фактически получается выражение: `53 + 4 - 1 = 56`. Пример `iptables -t mangle -A PREROUTING -o eth0 -j TTL --ttl-inc 1`

ULOG

ULOG предоставляет возможность журналирования пакетов в пользовательское пространство. Оно заменяет традиционное действие LOG, базирующееся на системном журнале. При использовании этого действия пакет через сокеты netlink передается специальной службе, которая может выполнять очень детальное журналирование в различных форматах (например, обычный текстовый файл) и к тому же поддерживает возможность добавления надстроек (плагинов) для формирования различных выходных форматов и обработки сетевых протоколов.

Ключи для действия ULOG приведены в таблице 50.

Таблица 50

Ключ	Описание
--ulog-nlgroup	Сообщает ULOG, в какую группу netlink должен быть передан пакет. Всего существует 32 группы (от 1 до 32). Если необходимо передать пакет в пятую группу, то можно указать: --ulog-nlgroup 5 По умолчанию используется первая группа. Пример iptables -A INPUT -p TCP --dport 22 -j ULOG \ --ulog-nlgroup 2
--ulog-prefix	Имеет тот же смысл, что и аналогичный параметр в действии LOG. Длина строки префикса не должна превышать 32 символа. Пример iptables -A INPUT -p TCP --dport 22 -j ULOG \ --ulog-prefix "SSH connection attempt: "
--ulog-cprange	Определяет какую долю пакета в байтах надо передавать демону ULOG. Если указать число 100, как показано в примере, то демону будет передано только 100 Б из пакета, это означает, что демону будет передан заголовок пакета и некоторая часть области данных пакета. Если указать 0, то будет передан весь пакет, независимо от его размера. Значение по умолчанию равно 0. Пример iptables -A INPUT -p TCP --dport 22 -j ULOG \ --ulog-cprange 100
--ulog-qthreshold	Устанавливает величину буфера в области ядра. Например, если задать величину буфера, равной 10, как в примере, то ядро будет накапливать журналируемые пакеты во внутреннем буфере и передавать в пользовательское пространство группами по 10 пакетов. Пример iptables -A INPUT -p TCP --dport 22 -j ULOG \ --ulog-qthreshold 10

12.8. Поддержка фильтрации на основе классификационных меток

12.8.1. Модули `iptables` для работы с классификационными метками

Поддержка фильтрации на основе классификационных меток реализована с помощью дополнительного модуля тестирования `astralabel`, обеспечивающего тестирование значений мандатных атрибутов с помощью параметров `--maclev` и `--maccat`.

Для работы с классификационными метками необходимо установить пакеты `iptables` (по умолчанию пакеты не устанавливаются):

- `iptables-astralabel` — модули для использования с ядром generic;
- `iptables-astralabel-common` — общие модули.

Для установки пакетов выполнить:

```
sudo apt install iptables-astralabel-common iptables-astralabel
```

Модуль `astralabel` поддерживает стандартный синтаксис командной строки, используемый в `iptables`, а также дополнительные параметры для контроля мандатных атрибутов сетевых пакетов, приведенные в таблице 51.

Таблица 51

Параметр	Описание
`-m astralabel`	Указание на использование модуля `astralabel` для обработки сетевого трафика
`--maclev <уровень> [:<уровень>]`	Применение правила к пакетам, имеющим указанный иерархический уровень конфиденциальности. Для задания диапазона уровней конфиденциальности уровни указываются через символ «:» включительно. Пример Не принимать пакеты с иерархическими уровнями конфиденциальности от 1 до 3: `iptables -A INPUT -m astralabel --maclev 1:3 -j \ DROP` Параметр `maclev` может применяться одновременно с параметром `maccat` в одном правиле. Результирующий фильтр будет представлять собой объединение фильтров, заданных данными параметрами

Окончание таблицы 51

Параметр	Описание

!	Значение фильтра, задаваемого параметром `--maclev`, может быть инверсировано с помощью модификатора «!».
	
	Пример
	
	Не пропускать исходящие пакеты, имеющие ненулевой иерархический уровень конфиденциальности:
	
	`iptables -A OUTPUT -m astralabel ! --maclev 0 \`
	`-j DROP`
	
	Данное правило будет пропускать исходящие пакеты, имеющие нулевой иерархический уровень конфиденциальности и ненулевые неиерархические категории доступа

`--maccat <бит_категории>`	Применение правила к пакетам, имеющим указанные неиерархические категории конфиденциальности. Задание диапазонов и инверсирование не поддерживаются. В одном правиле может быть указано несколько параметров `maccat`, в таком случае правило будет применяться только к пакетам с установленными одновременно всеми указанными категориями (битами).
	Примечание. Нумерация битов категорий начинается с единицы.
	
	Пример
	
	Не пропускать исходящие пакеты с установленными одновременно битами категорий 1 и 2:
	
	`iptables -A OUTPUT -m astralabel --maccat 1 \`
	`--maccat 2 -j DROP`
	
	Параметр `maccat` может применяться одновременно с параметром `maclev` в одном правиле. Результирующий фильтр будет представлять собой объединение фильтров, заданных данными параметрами.
	
	Пример
	
	Не принимать пакеты с установленными битами категорий 1 и 2 и уровнем конфиденциальности 3:
	
	`iptables -A INPUT -m astralabel --maclev 3 \`
	`--maccat 1 --maccat 2 -j DROP`

ВНИМАНИЕ! Если параметры для фильтрации пакетов на основе классификационной метки не указаны, то правило применяется ко всем пакетам, имеющим ненулевую классификационную метку. Например, правило

```
iptables -A OUTPUT -m astralabel -j DROP
```

запретит все исходящие пакеты, имеющие ненулевую классификационную метку (т.е. имеющие ненулевой уровень конфиденциальности и/или ненулевые категории доступа).

Описание модулей `iptables` для фильтрации пакетов на основе классификационной метки приведено на справочной странице `man iptables-astralabel`.

12.8.2. Использование ufw для работы с классификационными метками

В межсетевом экране ufw включена поддержка работы с классификационными метками по умолчанию. Для управления сетевыми соединениями с учетом классификационных меток в межсетевой экран ufw добавлены параметры `mac`, `maclev` и `maccat`, по действию аналогичные параметрам `iptables`.

Для того, чтобы данные параметры работали, в системе должны быть установлены в соответствии с 12.8.1 пакеты `iptables-astralabel-common` и `iptables-astralabel`.

Примеры:

1. Запрет отправки на 80 IP-порт пакетов с ненулевой классификационной меткой:

```
ufw deny out 80/tcp mac
```

Аналогично параметрам `iptables`:

```
-A OUTPUT -p tcp --dport 80 -m astralabel -j DROP
```

2. Запрет отправки на 80 IP-порт (протокол HTTP) сетевых пакетов, имеющих уровень конфиденциальности 2:

```
ufw deny out 80/tcp maclev 2
```

Аналогично параметрам `iptables`:

```
-A OUTPUT -p tcp --dport 80 -m astralabel --maclev 2 -j DROP
```

3. Запрет отправки на 80 IP-порт (протокол HTTP) сетевых пакетов, имеющих категорию конфиденциальности 3:

```
ufw deny out 80/tcp maccat 3
```

Аналогично параметрам `iptables`:

```
-A OUTPUT -p tcp --dport 80 -m astralabel --maccat 3 -j DROP
```

Возможность инверсии правил и возможность одновременного указания нескольких категорий в одном правиле текущей реализацией ufw не поддерживаются.

Описание ufw для фильтрации пакетов на основе классификационной метки приведено в `man ufw`.

12.8.3. Использование Open vSwitch для работы с классификационными метками

Программный коммутатор Open vSwitch (OVS) из состава ОС поддерживает классификационные метки (уровни и категории конфиденциальности) и может использоваться для фильтрации сетевого потока в условиях мандатного управления доступом.

Фильтрация IP-пакетов, которые содержат классификационные метки, сначала выполняется на портах, затем на основе правил OpenFlow коммутатора OVS. Параметры фильтрации задаются путем назначения классификационных меток портам и добавления классификационных меток в правила OpenFlow.

Для назначения порту или указания в правиле OpenFlow уровня конфиденциальности используется параметр `astra_mac_level`:

```
astra_mac_level=<уровень_конфиденциальности>
```

где `<уровень_конфиденциальности>` — десятичное число, соответствующее уровню конфиденциальности.

Для назначения порту или указания в правиле OpenFlow категорий конфиденциальности используется параметр `astra_mac_categories`:

```
astra_mac_categories=<категории_конфиденциальности>
```

где `<категории_конфиденциальности>` — шестнадцатеричное число, битовая маска которого определяет набор категорий конфиденциальности.

Если параметры уровня и/или категорий конфиденциальности отсутствуют, то по умолчанию для них принимается значение 0 — нулевой уровень конфиденциальности и/или без категорий конфиденциальности.

При фильтрации IP-пакет пропускается через порт, если IP-пакету назначена классификационная метка не выше, чем классификационная метка порта. IP-пакет, пройденный через порт, пропускается правилом OpenFlow, если IP-пакету назначена классификационная метка не выше, чем в правиле OpenFlow. Например, если в правиле OpenFlow задана блокировка со значениями параметров `astra_mac_level=2` и `astra_mac_categories=0x03`, то при применении правила не будут пропущены IP-пакеты, имеющие нулевой, первый или второй уровень конфиденциальности и имеющие категории конфиденциальности, битовая маска которых содержится в битовой маске `0x03`.

Примеры:

1. Добавить порт `tap0` и назначить ему второй уровень конфиденциальности и категории конфиденциальности, соответствующие битовой маске `0x03`:

```
sudo ovs-vsctl add-port br0 tap0 astra_mac_level=2 \
     astra_mac_categories=0x03
```

2. Добавить порт `tap1` и назначить ему категории конфиденциальности, соответствующие битовой маске `0x08`, по умолчанию будет назначен нулевой уровень конфиденциальности:

```
sudo ovs-vsctl add-port br0 tap1 astra_mac_categories=0x08
```

3. Добавить порт `tap3` с нулевым уровнем конфиденциальности и без категорий конфиденциальности:

```
sudo ovs-vsctl add-port br0 tap3
```

4. Для порта `tap0` изменить текущий уровень конфиденциальности на третий уровень конфиденциальности (будут изменены только указанные параметры):

```
sudo ovs-vsctl upd-port br0 tap0 astra_mac_level=3
```

5. Для порта `tap3` изменить текущие уровень и категории конфиденциальности на первый уровень конфиденциальности и категории конфиденциальности, соответствующие битовой маске `0x7` (будут изменены только указанные параметры):

```
sudo ovs-vsctl upd-port br0 tap3 astra_mac_level=1 \
     astra_mac_categories=0x7
```

6. Добавить правило блокировки IP-пакетов, которые в своем уровне IP имеют `ip_dst=87.250.250.242` и которым назначен третий уровень конфиденциальности и категории конфиденциальности, соответствующие битовой маске `0x200`:

```
sudo ovs-ofctl add-flow br0 ip,ip_dst=87.250.250.242,astra_mac_level=3,\
     astra_mac_categories=0x200,actions=drop
```

7. Добавить правило блокировки IP-пакетов с третьим уровнем конфиденциальности и любым набором категорий конфиденциальности:

```
sudo ovs-ofctl add-flow br0 ip,astra_mac_level=3,actions=drop
```

Информация о назначенных портам коммутатора OVS уровнях и категориях конфиденциальности отображается в выводе команды:

```
sudo ovs-vsctl show
```

Пример

```
sudo ovs-vsctl show
```

Вывод команды:

```
aea83ec6-49bd-4b44-aa6b-82967bcd4a5b
Bridge br0
datapath_type: netdev
Port tap1
MAC categories: 0x0000000000000008
Interface tap1
Port tap0
MAC Level: 3
MAC categories: 0x0000000000000003
Interface tap0

Port tap3
MAC Level: 1
MAC categories: 0x0000000000000007
Interface tap3
Port br0
Interface br0
type: internal
ovs_version: "3.0.1"
```

Для получения информации о правилах, в которых заданы уровни и категории конфиденциальности, выполнить команду:

```
sudo ovs-ofctl -F astra dump-flows bridge
```

13. МАРКИРОВКА ДОКУМЕНТОВ

13.1. Общие сведения

Защищенный комплекс программ печати и маркировки документов обеспечивает маркировку выводимых на печать документов. Маркировка документов осуществляется при включенном в ОС мандатном управлении доступом.

Дополнительно к дискреционному управлению доступом на основе политик сервера печати CUPS используется мандатное управление доступом. Мандатные атрибуты порождаемых объектов в CUPS наследуют мандатный контекст, получаемый из сетевого соединения.

Все субъекты доступа (пользователи) сервера печати делятся на две группы: администраторов и обычных пользователей.

Администраторам разрешено выполнять ряд действий по модификации сервера печати (добавление/удаление принтеров, модификация их параметров и т.д.).

ВНИМАНИЕ! Данные операции должны проводиться только под нулевым мандатным контекстом.

Пользователям разрешается выводить документы на печать, выполнять модификацию и просмотр заданий согласно их мандатному контексту доступа. В таблице 52 приведено разделение операций по группам доступа и типам операций доступа.

Таблица 52

	Чтение	Запись

Административные операции	`CUPS-Get-Devices`
	`CUPS-Get-PPDs`
	`CUPS-Get-PPD`
	`CUPS-Get-Policies`
	`MAC-Get-Info`
		`Pause-Printer`
		`Resume-Printer`
		`Set-Printer-Attributes`
		`Enable-Printer`
		`Disable-Printer`
		`Hold-New-Jobs`
		`Release-New-Jobs`
		`CUPS-Add-Modify-Printer`
		`CUPS-Delete-Printer`
		`CUPS-Add-Modify-Class`
		`CUPS-Delete-Class`
		`CUPS-Set-Default`
		`CUPS-Create-Local-Printer`

Окончание таблицы 52

	Чтение	Запись
Неадминистративные операции	CUPS-Get-Printers Get-Printer-Supported-Values Get-Printer-Attributes CUPS-Get-Default Get-Subscription-Attributes Get-Subscriptions Get-Notifications Get-Jobs Validate-Job Get-Job-Attributes CUPS-Get-Document MAC-Get-Journal	Purge-Jobs Print-Job Cancel-Jobs Create-Job-Subscription Create-Printer-Subscriptions Renew-Subscription Cancel-Subscription Create-Job Send-Document Cancel-Job Hold-Job Release-Job Restart-Job Set-Job-Attributes Cancel-My-Jobs Close-Job Cups-Authenticate-Job CUPS-Move-Jobs MAC-Set-Job-Attributes MAC-Mark-Document MAC-Lock-Job MAC-Unock-Job

13.2. Настройка печати документа с ненулевой классификационной меткой

Для обеспечения возможности обрабатывать задания печати, формируемые в ненулевом мандатном контексте, а также для печати данных заданий на принтере необходимо серверу CUPS и принтеру установить политику `parsec`, выполнив, соответственно, от имени учетной записи администратора команды:

```
sudo cupsctl DefaultPolicy=parsec
sudo lpattr -p <имя_принтера> -s printer-op-policy=parsec
```

Описание инструмента командной строки `cupsctl` приведено на справочной странице `man cupsctl`.

Для принтера установить политику `parsec` также можно с использованием графической утилиты `fly-admin-printer` во вкладке «МАС». Дополнительная информация об утилите `fly-admin-printer` приведена в электронной справке.

Принтеры и классы являются сущностями-контейнерами, в которые помещаются задания, и обладают атрибутами `mac-printer-mac-max` и `mac-printer-mac-min`.

Атрибут `mac-printer-mac-max` определяет максимальный мандатный контекст заданий, которые могут находиться в сущности-контейнере.

Атрибут mac-printer-mac-min определяет минимальный мандатный контекст заданий, которые могут находиться в сущности-контейнере.

Для разрешения печати на принтере документов пользователей, работающих в ненулевом мандатном контексте, необходимо для принтера установить допустимый диапазон мандатного контекста, выполнив следующие команды:

```
sudo lpattr -p <имя_принтера> -s mac-printer-mac-min=LLabel
sudo lpattr -p <имя_принтера> -s mac-printer-mac-max=MLabel
```

где LLabel — минимальный мандатный контекст заданий, которые разрешено печатать;
MLabel — максимальный мандатный контекст заданий, которые разрешено печатать.

Допустимый диапазон мандатного контекста также возможно установить с помощью графической утилиты fly-admin-printer, задав во вкладке «МАС» допустимый диапазон уровней конфиденциальности, категорий конфиденциальности и категории целостности.

Если ЕПП не используется, то настройка принтера выполняется от имени администратора через механизм sudo или от имени пользователя, входящего в локальную группу администраторов печати (группа lpadmin, см. документ РУСБ.10015-01 95 01-1). Если ЕПП используется, то выполняется от имени пользователя, входящего в локальную группу администраторов печати.

Параметры сервера печати CUPS определены в конфигурационном файле /etc/cups/cupsd.conf. Редактирование конфигурационного файла сервера печати CUPS выполняется от имени пользователя, входящего в локальную группу администраторов печати с использованием:

- инструмента командной строки cupsctl, запускаемого через механизм sudo;
- графической утилиты fly-admin-printer;
- web-интерфейса сервера печати CUPS (http://127.0.0.1:631).

Для настройки маркировки используются параметры конфигурационного файла /etc/cups/cupsd.conf, приведенные в таблице 53.

Таблица 53

Параметр	Значение по умолчанию	Описание
MacAudit	on	Включить регистрацию событий с помощью библиотеки libastraeventse
MacDuplicate	off	Включить дублирование заданий. Копии заданий сохраняются по умолчанию в каталоге /var/spool/cups/parsec в PDF-формате

Окончание таблицы 53

Параметр	Значение по умолчанию	Описание
MacDuplex	off	Режим печати маркера на обратной стороне (при наличии поддержки со стороны принтера двусторонней печати). Возможные значения: - off — отключено. Печать документа в обычном режиме, при котором создаются два задания: задание печати страниц документа с проставленными маркерами и задание печати маркера на обороте последней страницы. При этом последнюю страницу потребуется перевернуть вручную; - marker — автоматическая печать маркера на обороте всех страниц документа; - fonarik — автоматическая печать маркера на обороте последней страницы. Будут созданы два задания на печать: задание печати страниц документа с проставленными маркерами без последней страницы и задание из двух страниц (последняя страница документа и маркер на обороте последней страницы) для автоматической двусторонней печати
MacEnable	on	Включить поддержку меток безопасности в CUPS
MacEnableFonarik	on	Создавать задание для печати регистрационной информации документа («фонарика» на обороте последнего листа документа)
MacJournal	off	Включить журнал маркировки. По умолчанию журнал записывается в базу данных SQLITE /var/spool/cups/parsec/marking-journal.sqlite
MacConvertToPDF	on	Преобразовывать унаследованные задания в PDF-формат (для устранения проблем с печатью шрифтов на некоторых PostScript-принтерах)
MacHighIntAdmin	off	Требовать высокую метку целостности для операций администрирования (редактирование принтеров, перенос задания на другой принтер и т.д.). Работает только для локальных сокетов
MacSelectFont	off	Запрашивать у пользователя шрифт для маркировки
MacFullyQualifiedNames	off	Использовать полное доменное имя пользователя (вида <пользователь>@<домен>) при определении его членства в группах, а также при проверке пользователя на соответствие политикам CUPS. Требуется для работы в домене FreeIPA

13.3. Настройка маркировки

Настройка маркировки осуществляется редактированием следующих файлов:

1) /etc/cups/marker.template — шаблон маркера, описывающий элементы маркировки на первой странице, последующих страницах и на обороте последней страницы. Файл шаблона по умолчанию /usr/share/cups/marker.template устанавливается вместе с пакетом и может использоваться при необходимости вернуть комплекс программ печати и маркировки документов в исходное состояние. Описание параметров шаблона маркера приведено в 13.3.1;

2) /etc/cups/cups-marker-vars.conf — конфигурационный файл описания переменных маркировки. Значения этих переменных будут запрошены у пользователя перед маркировкой. Описание файла приведено в 13.3.2. Файл /usr/share/cups/cups-marker-vars.conf.default содержит описание переменных маркировки по умолчанию, устанавливается вместе с пакетом и может использоваться при необходимости вернуть комплекс программ печати и маркировки документов в исходное состояние;

3) /usr/share/cups/psmarker/marker.defs — файл описания положения элементов маркера на страницах;

4) /usr/share/cups/fonarik/fonarik.defs — файл описания положения элементов маркера на обороте последней страницы;

5) /usr/share/cups/fonts/*.t42 — шрифты для маркировки;

6) /usr/share/cups/charsets/utf-8.*, /usr/share/cups/charsets/utf-8 — наборы символов для различных шрифтов маркировки.

Редактирование конфигурационных файлов и установка шрифтов возможны только локально на сервере печати через механизм sudo или с помощью утилиты fly-admin-marker совместно с механизмом KAuth.

Редактирование файлов /usr/share/cups/fonarik/fonarik.defs и /usr/share/cups/psmarker/marker.defs возможно только от имени администратора через механизм sudo.

Описание графической утилиты fly-admin-marker см. в электронной справке.

13.3.1. Шаблон маркера

Шаблон определяет внешний вид маркеров на страницах, а также содержит параметры, используемые программами psmarker и fonarik в процессе маркировки.

ВНИМАНИЕ! Для изменения маркера необходимо редактировать файл /etc/cups/marker.template. Файл /usr/share/cups/marker.template создается при установке пакета и может быть перезаписан в процессе обновления системы.

В шаблоне в начале файла задаются параметры маркировки:

1) `fonarik_boundary` — задается число копий, после печати которого будет сменен маркер на обороте последней страницы;

2) `replace_underscore` — преобразование символа нижнего подчеркивания «_» в пробел в названии уровней и категорий (0 — отключить, 1 — включить);

3) `charset` — набор символов маркировки, позволяет изменить шрифт маркировки.

Далее построчно задаются маркеры в виде:

`<страница>:<начертание>:<размер>:<семейство>:<расположение>:<строка_маркировки>`

где:

1) `<страница>` — страница, на которой будет располагаться данная строка маркера. Возможные значения:

   а) `fonarik` — оборот последней страницы, если число копий меньше или равно значению `fonarik_boundary`;

   б) `fonarik_gt_5` — оборот последней страницы, если число копий больше значения `fonarik_boundary`;

   в) `first` — первая страница документа;

   г) `any` — страницы документа, кроме первой и последней;

   д) `last` — последняя страница документа;

2) `<начертание>` — начертание шрифта маркировки. Возможные значения:

   а) `normal` — нормальное начертание;

   б) `bold` — жирное начертание;

   в) `italic` — наклонное начертание;

   г) `bolditalic` — жирное наклонное начертание;

3) `<размер>` — размер шрифта маркировки;

4) `<семейство>` — шрифт, не реализовано для строки маркировки, используется шрифт, заданный параметром `charset`;

5) `<расположение>` — расположение данной строки на странице. Возможные значения:

   а) `top-left` — вверху слева;

   б) `top-center` — вверху по центру;

   в) `top-right` — вверху справа;

   г) `bottom-left` — внизу слева;

   д) `bottom-center` — внизу по центру;

   е) `bottom-right` — внизу справа;

6) <строка_маркировки> — строка маркировки, которая будет добавлена на страницу. Может содержать как произвольный текст, так и переменные, вместо которых будут подставлены значения в процессе маркировки. Переменные указываются в фигурных скобках {}. Переменные бывают следующих типов:

а) встроенные переменные — записываются заглавными буквами. Данные переменные заданы в CUPS и программах маркировки psmarker и fonarik. Их значение подставляется автоматически и не может быть изменено пользователем в процессе маркировки. Список встроенных переменных приведен в таблице 54.

Таблица 54

Переменная	Описание
CURRENT_COPY	Текущая копия документа, отправленная на печать
CURRENT_PAGE	Текущая страница документа
NUM_PAGES	Число страниц документа
LABEL_NAME	Метка конфиденциальности задания в формате Уровень:Категория
LABEL_NAME_FULL	Метка безопасности задания в формате Уровень:Категория:Целостность
LEV_NAME	Уровень конфиденциальности задания. Оставлена для совместимости. Рекомендуется использовать mac-job-mac-level-name
ILEV_NAME	Категория целостности задания. Оставлена для совместимости. Рекомендуется использовать mac-job-mac-ilevel-name
CAT_NAME	Категория конфиденциальности задания. Оставлена для совместимости. Рекомендуется использовать mac-job-mac-category-name
DATE	Текущая дата в формате ДД.ММ.ГГГГ
TIME	Текущее время в формате ЧЧ:ММ:СС

б) атрибуты задания — записываются строчными буквами и должны соответствовать атрибутам переменных маркировки (в т. ч. пользовательских) или стандартным атрибутам (см. 13.3.2);

в) сценарии — задаются в виде {SCRIPT:/path/myscript.sh}. При маркировке переменная будет заменена выводом сценария. Сценарий должен выводить одну строку через стандартный поток (stdout) и быть доступным для выполнения от имени учетной записи root;

г) атрибуты пользователя FreeIPA — задаются в виде {FREEIPA:sn}, где sn — имя атрибута FreeIPA пользователя, который создал задание. Посмотреть полный список доступных атрибутов можно командой ipa user-show printuser --raw --all.

Настроить шаблон маркировки также возможно в графической утилите fly-admin-marker (см. электронную справку).

Пример

Шаблон /etc/cups/marker.template

```
fonarik_boundary=5
replace_underscore=0
charset=utf-8

fonarik:normal:12:Arial:top-left:{mac-inv-num}
fonarik:normal:12:Arial:top-left:Экземпляров {copies}
fonarik:normal:12:Arial:top-left:Количество страниц {NUM_PAGES}
fonarik:normal:12:Arial:top-left:{mac-workplace-id}
fonarik:normal:12:Arial:top-left:{mac-distribution}
fonarik:normal:12:Arial:top-left:Исполнитель {mac-job-originating-
     user-full-name}
fonarik:normal:12:Arial:top-left:Тел. {mac-owner-phone}
fonarik:normal:12:Arial:top-left:Отпечатал {mac-job-user-full-name}
fonarik:normal:12:Arial:top-left:{DATE}

fonarik_gt_5:normal:12:Arial:top-left:Исполнитель
     {mac-job-originating-user-full-name}
fonarik_gt_5:normal:12:Arial:top-left:Тел. {mac-owner-phone}

first:normal:12:Arial:top-right:{LABEL_NAME}
first:normal:12:Arial:top-right:Экз. {CURRENT_COPY}
first:normal:12:Arial:bottom-right:{mac-inv-num}
any:normal:12:Arial:bottom-right:{mac-inv-num}
last:normal:12:Arial:bottom-right:{mac-inv-num}
first:normal:12:Arial:top-center:{CURRENT_PAGE} из {NUM_PAGES}
any:normal:12:Arial:top-center:{CURRENT_PAGE} из {NUM_PAGES}
last:normal:12:Arial:top-center:{CURRENT_PAGE} из {NUM_PAGES}
```

13.3.2. Переменные маркировки

При маркировке документа используются переменные маркировки. Переменные маркировки могут быть основными и пользовательскими и должны быть описаны в конфигурационном файле /etc/cups/cups-marker-vars.conf. Значение переменных маркировки запрашивается у пользователя перед маркировкой.

ВНИМАНИЕ! Для описания переменных маркировки необходимо редактировать файл /etc/cups/cups-marker-vars.conf. Файл /usr/share/cups/cups-marker-vars.conf.default создается при установке пакета и может быть перезаписан в процессе обновления системы.

Переменные маркировки описываются в следующем виде:

```
атрибут:имя_переменной:значение_по_умолчанию:2 #необязательная переменная
атрибут:имя_переменной:значение_по_умолчанию:1 #обычная переменная
атрибут:имя_переменной:значение_по_умолчанию:0 #скрытая переменная
```

Каждой переменной маркировки в конфигурационном файле `/etc/cups/cups-marker-vars.conf` должен соответствовать атрибут.

Основные переменные заданы по умолчанию в программе маркировки и их настраивать не требуется. Также для основных переменных не требуется указывать их имена в конфигурационном файле. Атрибуты основных переменных приведены в таблице 55.

Таблица 55

Атрибут	Описание
`mac-distribution`	Список рассылки
`mac-inv-num`	Инвентарный номер
`mac-owner-phone`	Телефон исполнителя
`mac-workplace-id`	Идентификатор рабочего места
`mac-job-mac-level-reason`	Основание для степени секретности (пункт перечня)
`mac-job-user-full-name`	Полное имя пользователя, выполнившего маркировку (значение пункта «отпечатал»)
`mac-job-originating-user-full-name`	Полное имя пользователя, создавшего задание (значение пункта «исполнитель»)
`mac-marker-font`	Шрифт для маркировки

Пользовательские переменные могут быть созданы пользователем и для них в конфигурационном файле необходимо указывать имя.

Примечание. В названиях переменных и значениях по умолчанию в файле `/etc/cups/cups-marker-vars.conf` необходимо использовать экранирование специального символа «:» (двоеточие), используемого для разделения полей, путем замены данного символа на «\:». Кроме того, экранирование используется при подстановке значений встроенных переменных, что учитывается клиентскими программами.

Имя атрибута для пользовательской переменной должно иметь префикс `mac-`. Если атрибут пользовательской переменной задан без данного префикса, то значение переменной маркировки не будет запрашиваться у пользователя.

Для каждой переменной маркировки в файле `/etc/cups/cups-marker-vars.conf` указывается ее тип и значение по умолчанию. Переменная маркировки может иметь следующий тип:

- «1» — обычная переменная. Обязательное для ввода значение. Запрашивается у пользователя. Если значение не введено пользователем, то заполняется значением по умолчанию;
- «2» — необязательная (опциональная) переменная. Запрашивается у пользователя, но при этом не обязательное к вводу значение. Если значение не введено пользователем, то соответствующему атрибуту присваивается значение «не указано»;
- «0» — скрытая переменная. Не запрашивается у пользователя и соответствующему атрибуту присваивается значение по умолчанию.

Значение по умолчанию может быть пустым или произвольным текстом. Также в качестве значения по умолчанию может использоваться одна из служебных переменных, перечень и описание которых приведены в таблице 56, или один из стандартных атрибутов, перечень и описание которых приведены в таблице 57. Значения служебных переменных и стандартных атрибутов присваиваются до маркировки.

Таблица 56

Переменная	Описание
`{LABEL_NAME}`	Метка конфиденциальности задания в формате `уровень:категория`
`{LABEL_NAME_FULL}`	Метка безопасности задания в полном текстовом виде
`{LEVEL_NAME}`	Уровень конфиденциальности задания. Переменная оставлена для совместимости. Рекомендуется использовать `mac-job-mac-level-name`
`{ILEVEL_NAME}`	Категория целостности задания. Переменная оставлена для совместимости. Рекомендуется использовать `mac-job-mac-ilevel-name`
`{CATEGORY_NAME}`	Категория конфиденциальности задания. Переменная оставлена для совместимости. Рекомендуется использовать `mac-job-mac-category-name`
`{GECOS_OWNER_NAME}`	Полное имя пользователя, создавшего задание (подставляется из GECOS). При отсутствии данных в GECOS используется имя учетной записи пользователя (логин)
`{GECOS_REQUESTING_USER_NAME}`	Полное имя пользователя, выполняющего маркировку (подставляется из GECOS). При отсутствии данных в GECOS используется имя учетной записи пользователя (логин)
`{GECOS_OWNER_PHONE}`	Телефон пользователя, создавшего задание (подставляется из GECOS). При отсутствии данных в GECOS заменяется пустой строкой

Таблица 57

Атрибут	Описание
mac-job-marking-required	Определяет, требуется ли маркировка задания
job-name	Имя задания
copies	Число копий
mac-job-derive	Число унаследованных заданий
mac-job-mac-label	Метка безопасности задания
mac-job-marked	Определяет, выполнена ли маркировка
mac-job-origin	Номер исходного задания, к которому относится унаследованное задание
mac-job-user-name	Имя пользователя, выполнившего маркировку
mac-job-mac-level-name	Уровень конфиденциальности задания. Формируются на основе атрибутов сессии пользователя, создавшего задание на печать
mac-job-mac-ilevel-name	Категория целостности задания. Формируются на основе атрибутов сессии пользователя, создавшего задание на печать
mac-job-mac-category-name	Категория конфиденциальности задания. Формируются на основе атрибутов сессии пользователя, создавшего задание на печать
mac-marking-session-id	Идентификатор сессии
job-originating-user-name	Имя пользователя, создавшего задание на печать

Пример

Файл cups-marker-vars.conf

```
#основные переменные
mac-inv-num:::1
mac-owner-phone::{GECOS_OWNER_PHONE}:1
mac-workplace-id:::1
mac-distribution:::2
mac-job-mac-level-reason:::0
mac-job-originating-user-full-name::{GECOS_OWNER_NAME}:0
mac-job-user-full-name::{GECOS_REQUESTING_USER_NAME}:0
#пользовательские
mac-myvar:Название переменной:значение по умолчанию:0
```

Настроить переменные маркировки также возможно в графической утилите fly-admin-marker (см. электронную справку).

13.3.3. Файлы описания маркера

Для изменения положения маркера, проставляемого на первой и последующих страницах, необходимо в файле `/usr/share/cups/psmarker/marker.defs` изменить значения параметров (единица измерения каждого параметра — типографский пункт):

- `MarkerTopShift` — отступ от верхней границы страницы;
- `MarkerBottomShift` — отступ от нижней границы страницы;
- `MarkerLeftShift` — отступ от левой границы страницы;
- `MarkerRightShift` — отступ от правой границы страницы;
- `MarkerStringInterval` — интервал между строками.

Для изменения положения маркера, проставляемого на обороте последней страницы, необходимо в файле `/usr/share/cups/fonarik/fonarik.defs` изменить значения параметров для соответствующего формата и ориентации страницы (единица измерения каждого параметра — типографский пункт):

- `FonarikTopShift` — отступ от верхней границы страницы;
- `FonarikBottomShift` — отступ от нижней границы страницы;
- `FonarikLeftShift` — отступ от левой границы страницы;
- `FonarikRightShift` — отступ от правой границы страницы;
- `FonarikStringInterval` — интервал между строками;
- `A0, ..., A5` — печатный формат страницы A0, ..., A5;
- `L, P` — ориентация страницы: P — вертикальная, L — горизонтальная.

Пример

Содержание файла `/usr/share/cups/fonarik/fonarik.defs`

```
...
A4:P:FonarikTopShift=520.0
A4:P:FonarikBottomShift=20.0
A4:P:FonarikLeftShift=36.0
A4:P:FonarikRightShift=36.0
A4:P:FonarikStringInterval=4.0
...
```

Любые изменения содержания и формата маркера страниц может производить только администратор.

Настроить маркеры также возможно в графической утилите `fly-admin-marker` (см. электронную справку).

13.3.4. Изменение шрифта маркировки

Шрифт маркировки задается файлом сопоставления шрифтов в шаблоне маркировки. Для каждого шрифта имеется свой файл сопоставления. Изменения шрифта маркировки производится путем указания в шаблоне маркировки для параметра `charset` соответствующего требуемому шрифту файла сопоставления, например:

```
charset=utf-8.PTAstraSans
```

ВНИМАНИЕ! Редактировать необходимо шаблон маркировки `/etc/cups/marker.template`. Если данный шаблон отсутствует, то скопировать установленный вместе с пакетом шаблона маркировки `/usr/share/cups/marker.template` в `/etc/cups/marker.template`.

Доступные файлы сопоставления находятся в `/usr/share/cups/charsets`. При выборе несуществующего файла сопоставления используется по умолчанию `utf-8`, что соответствует шрифту PT Astra Serif.

Возможно создать собственный шрифт маркировки на основе имеющегося шрифта в формате TrueType (`*.ttf`). Исходный шрифт должен содержать кириллицу и иметь четыре варианта начертания (нормальное, жирное, наклонное и жирное наклонное), т.е. состоять из четырех файлов с расширением `*.ttf`.

Все четыре файла необходимо преобразовать в формат Type 42 (расширение `*.t42`). Для этого необходимо:

1) установить пакет `fontforge`;

2) в каталоге с ttf-шрифтами создать сценарий `converter.pe` (если отсутствует) со следующим содержанием:

```
#!/usr/bin/fontforge

i=1
while ( i<$argc )
Open($argv[i])
RenameGlyphs("Adobe Glyph List")
Generate($argv[i]:r + ".t42")
i = i+1
endloop
```

3) сделать сценарий исполняемым, например, командой `chmod +x`;

4) запустить сценарий `converter.pe` на исполнение:

```
/home/user/converter.pe *.ttf
```

Полученные четыре файла с расширением `*.t42` необходимо скопировать в `/usr/share/cups/fonts`.

Далее создать файл сопоставления для нового шрифта в каталоге `/usr/share/cups/charsets`. Рекомендуется имя файла сопоставления задавать в виде `utf-8.<имя_шрифта>`, например, `utf-8.PTAstraSans`.

Пример

Файл сопоставления для шрифта PT Astra Sans

```
0000 00FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
0100 01FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
0200 02FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
0300 03FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
0400 04FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
1E00 1EFF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
2000 20FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
2100 21FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
2300 23FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
2400 24FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
2500 25FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
2600 26FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
```

В каждой строке после `single` находится список из четырех файлов, разделенных пробелом, соответствующих каждому варианту написания в следующем порядке: обычный, жирный, наклонный и жирный наклонный.

Установку и изменение шрифта также можно осуществить в графической утилите `fly-admin-marker` (см. электронную справку).

Примечание. Возможны проблемы с некоторыми PostScript-принтерами из-за особенностей поддержки шрифтов в формате `Type 42`, поэтому по умол-

чанию задание на печать преобразуется в формат `*.pdf` сразу после маркировки. Для отключения данного действия необходимо выполнить команду:
`sudo cupsctl MacConvertToPDF=Off`
или в `cupsd.conf` для параметра `MacConvertToPDF` установить значение `Off`, затем перезапустить сервер печати CUPS.

14. КОНТРОЛЬ ПОДКЛЮЧЕНИЯ СЪЕМНЫХ МАШИННЫХ НОСИТЕЛЕЙ ИНФОРМАЦИИ

Съемные машинные носители информации могут рассматриваться относительно ОС с двух точек зрения:

- как блочные или символьные устройства ввода-вывода;
- как блочное устройство, которое может быть смонтировано.

В первом случае устройство представляет собой специальный файловый объект, доступ к которому контролируется мандатными и дискреционными ПРД обычным образом и, следовательно, ввод-вывод остается в рамках контроля этих правил.

Во втором случае съемный машинный носитель информации содержит в себе образ ФС, которая и хранит данные. Данный носитель может быть смонтирован в заданный каталог, и при этом ФС носителя становится частью (представленной в виде поддерева) корневой ФС. Доступ к объектам данной ФС подчиняется мандатным и дискреционным ПРД обычным образом и, следовательно, ввод-вывод на съемный машинный носитель информации остается в рамках контроля этих правил.

ВНИМАНИЕ! В качестве файловых систем на съемных машинных носителях информации должны использоваться только файловые системы ext2/ext3/ext4/XFS, поддерживающие расширенные (в т.ч. мандатные) атрибуты файловых объектов и обеспечивающие гарантированное уничтожение (стирание) информации.

Для ОС возможность санкционированного монтирования определенными пользователями только разрешенных устройств, в т. ч. с указанными ФС, определяется администратором системы. В ОС реализованы средства разграничения доступа к подключаемым устройствам на основе генерации правил для менеджера устройств `udev`.

Учет носителей и управление их принадлежностью, протоколированием и мандатными атрибутами (в том числе и при работе в ЕПП) осуществляется с помощью модуля «Устройства и правила» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_devices_and_rules
```

В режиме «Мобильный» реализовано управление блокировкой подключаемых устройств с помощью утилиты USBGuard.

Рекомендации по использованию указанных средств приведены в документе РУСБ.10015-01 95 01-1.

15. СОПОСТАВЛЕНИЕ ПОЛЬЗОВАТЕЛЯ С УСТРОЙСТВОМ

ОС обеспечивает ввод-вывод информации на запрошенное пользователем устройство как для произвольно используемых им устройств, так и для идентифицированных (при совпадении маркировки)[1)] .

ОС включает в себя механизм, обеспечивающий надежное сопоставление мандатного контекста пользователя с уровнем и категориями конфиденциальности, установленными для устройства. Кроме того, механизм сопоставления пользователя с устройством, реализованный в ОС, обеспечивает при проверке совпадения маркировок носителя и пользователя применение дискреционных ПРД.

[1)] В режиме «Мобильный» реализовано управление блокировкой подключаемых устройств.

16. ОГРАНИЧЕНИЕ ПРОГРАММНОЙ СРЕДЫ И ФУНКЦИИ БЕЗОПАСНОСТИ

16.1. Замкнутая программная среда

16.1.1. Режимы функционирования

Инструменты из состава ОС предоставляют возможность создавать замкнутую программную среду (ЗПС). Использование ЗПС обеспечивает динамический контроль неизменности (целостности) и подлинности файлов и предназначено для выявления фактов несанкционированного изменения исполняемых файлов и других файлов (в т. ч. относящихся к КСЗ), предотвращения загрузки измененных исполняемых файлов, а также открытия других измененных файлов.

Контроль целостности реализован в невыгружаемом модуле ядра ОС `digsig_verif` и производится на основе проверки:

- цифровой подписи, внедренной в файл;
- цифровой подписи, содержащейся в расширенных атрибутах файловой системы файла;
- отсоединенной цифровой подписи (содержащейся в отдельном файле).

Цифровая подпись реализована в соответствии с ГОСТ Р 34.10-2012 (256 бит) и ГОСТ Р 34.10-2001 (256 бит), использование ГОСТ Р 34.10-2001 в ОС сохранено для совместимости.

При настроенной ЗПС не блокируется использование инструмента `qemu-user-static`, с помощью которого возможно запустить неподписанные исполняемые файлы, предназначенные для другой аппаратной архитектуры. Поэтому при включении ЗПС необходимо дополнительно включать блокировку интерпретаторов, которая блокирует в том числе использование `qemu-*-static` (см. 16.6.7). При этом блокировка интерпретаторов не влияет на использование ядерного механизма `binfmt`, который позволяет запускать исполняемые файлы другой аппаратной платформы через `qemu-*-static` (по умолчанию `binfmt` отключен).

Включение ЗПС может быть выполнено в процессе установки ОС путем выбора соответствующего пункта программы установки ОС.

Включение и выключение ЗПС после установки ОС выполняется с помощью инструмента `astra-digsig-control`, описанного в 16.6.28.

16.1.2. Типы подписей и наборы ключей

В ЗПС используются типы подписей, описанные в таблице 58.

Таблица 58

Тип	Описание
Встраиваемая подпись	Может быть встроена в следующие типы файлов: - бинарные исполняемые файлы и библиотеки формата ELF (в ELF-секции файла); - PE-файлы, в том числе формата DLL, для контролируемого запуска Windows-программ в среде Wine (в конце файла)
Подпись в расширенных атрибутах ФС	Ноль или более произвольных символов, исключая символ «/»
Отсоединенная подпись	Предназначена для файлов любых типов, в том числе для модулей ядра. Содержится в отдельных файлах с именами вида `@путь@имя_файла.расширение.sign` (например, `@tmp@example@settings.conf.sign`) в каталоге `/etc/digsig/external_sig/`. Она не нарушает структуру исходного файла, что может играть важную роль в системах, использующий периодический контроль целостности по эталонным значениям контрольных сумм. Так как файл подписи содержит в своем имени путь к подписанному файлу, при перемещении последнего он должен быть подписан заново

При проверке подписей модуль использует два набора ключей:

1) основной набор — три встроенных ключа изготовителя, используется для проверки любых подписей (в т.ч. для проверки подписи ключей из дополнительного набора);

2) дополнительный набор — состоит из открытых ключей в каталогах `/etc/digsig/keys/` и `/etc/digsig/xattr_keys/`, изначально ключи отсутствуют.

Ключи из каталога `/etc/digsig/keys/` применяются для проверки всех типов подписей. В каталог `/etc/digsig/keys/` добавляются открытые ключи в формате GnuPG, созданные с помощью инструмента командной строки `gpg` или переданные изготовителем (см. 16.1.5). Ключи из данного каталога должны быть подписаны ключом изготовителя. При этом ключи могут быть организованы в иерархическую структуру: файлы ключей в каталоге `/etc/digsig/keys/` должны быть подписаны ключом изготовителя, а файлы ключей в дочерних каталогах `/etc/digsig/keys/<имя_каталога>/` могут быть подписаны ключом изготовителя или ключом из `/etc/digsig/keys/`.

Пример

```
/etc/digsig/keys/key1.gpg - открытый ключ 1, подписанный
    на первичном ключе изготовителя
/etc/digsig/keys/key2.gpg - открытый ключ 2, подписанный
    на первичном ключе изготовителя
```

```
/etc/digsig/keys/key1/key1-1.gpg - открытый ключ, подписанный
    на ключе 1
/etc/digsig/keys/key1/key1-2.gpg - открытый ключ, подписанный
    на ключе 1
/etc/digsig/keys/key2/key2-1.gpg - открытый ключ, подписанный
    на ключе 2
/etc/digsig/keys/key2/key2-2.gpg - открытый ключ, подписанный
    на ключе 2
```

Ключи из каталога `/etc/digsig/xattr_keys/` применяются для проверки подписей в расширенных атрибутах ФС и отсоединенных подписей. В каталог `/etc/digsig/xattr_keys/` добавляются открытые ключи в формате GnuPG (см. 16.1.5). Подписи на ключах в данном каталоге не проверяются.

16.1.3. Отзыв сертификата ключа

В случае потери доверия к ключу цифровой подписи возможно исключить его сертификат из проверки подписей файлов. Список отзыва сертификатов расположен в файле `/etc/digsig/revoked_keys`.

ВНИМАНИЕ! При работе с данным списком следует использовать дозапись в файл, чтобы случайно не перезаписать его и не уничтожить предыдущие значения отозванных сертификатов.

Для отзыва сертификата ключа используется значение его SHA-1 отпечатка, который можно получить с помощью команды:

```
sha1sum <файл сертификата.gpg> | awk '{print $1}'
```

Для отзыва одного сертификата следует использовать команду:

```
echo "<SHA-1_отпечаток_сертификата_отзыва>" >> /etc/digsig/revoked_keys
```

Для отзыва сертификатов по заготовленному списку следует использовать команду:

```
cat <файл_списка_отзыва> >> /etc/digsig/revoked_keys
```

ВНИМАНИЕ! После выполнения операции отзыва повторная загрузка сертификата в модуль `digsig_verif` будет невозможна. Проверка подписей файлов, подписанных ключом, сертификат которого был отозван, будет завершена с ошибкой.

Для предварительной проверки того, какие файлы были подписаны отзываемым ключом, можно произвести поиск файлов по заданному сертификату ключа и сохранить их список в файл:

```
sudo bsign-integrator --sign-analyse <идентификатор_сертификата_ключа> \
    -E > SIGNED
```

Чтобы выполнить поиск файлов по списку сертификатов, следует использовать команду:

```
sudo bsign-integrator --sign-analyse -L <файл_списка_идентификаторов_ключей> \
    > SIGNED
```

Описание инструмента командной строки `bsign-integrator` приведено в 16.1.6.

Для применения настроек следует выполнить команду:

```
sudo update-initramfs -u -k all
```

и перезагрузить систему.

16.1.4. Модуль digsig_verif

16.1.4.1. Архитектура и настройка модуля digsig_verif

Модуль `digsig_verif` имеет следующие интерфейсы внутренней архитектуры:

1) `/sys/digsig/elf_mode` — режим работы при проверке цифровой подписи исполняемых файлов и разделяемых библиотек (см. 16.1.4.2);

2) `/sys/digsig/elf_verification_mode` — режим проверки цифровой подписи исполняемых файлов и разделяемых библиотек (см. 16.1.4.2);

3) `/sys/digsig/xattr_mode` — режим работы при проверке цифровой подписи в расширенных атрибутах ФС для неисполняемых файлов (см. 16.1.4.3);

4) `/sys/digsig/keys` — файл загрузки дополнительных ключей для проверки всех типов цифровых подписей;

5) `/sys/digsig/ignore_gost2001` — режим игнорирования проверки цифровой подписи по ГОСТ Р 34.10-2001;

6) `/sys/digsig/xattr_control` — файл загрузки шаблонов имен файлов, для которых проверяется цифровая подпись в расширенных атрибутах ФС;

7) `/sys/digsig/ignore_xattr_keys` – режим игнорирования ключей из `/etc/digsig/xattr_keys/` для проверки подписей в расширенных атрибутах ФС и отсоединенных подписей;

8) /sys/digsig/xattr_keys — файл загрузки дополнительных ключей, используемых только при проверке цифровой подписи в расширенных атрибутах ФС и отсоединенной подписи.

Файлы интерфейсов позволяют изменять настройки модуля digsig_verif в пределах текущей сессии пользователя, например с целью отладки. Запись значений в файлы выбора режимов переопределяет текущие режимы модуля digsig_verif. При записи значений в файлы загрузки эти значения будут добавлены к уже перечисленным в соответствующих файлах в каталоге /etc/digsig/.

Настройка модуля digsig_verif осуществляется путем редактирования конфигурационного файла /etc/digsig/digsig_initramfs.conf, в котором для соответствующих режимов работы задается их состояние (1 — включен, 0 — отключен).

Пример

Для отключения проверки цифровой подписи по ГОСТ Р 34.10-2001 необходимо в конфигурационном файле /etc/digsig/digsig_initramfs.conf для соответствующего параметра задать значение 1:

```
DIGSIG_IGNORE_GOST2001=1
```

После внесения изменений в конфигурационный файл /etc/digsig/digsig_initramfs.conf, а также для загрузки модулем digsig_verif ключей после их размещения в каталогах /etc/digsig/keys/ и /etc/digsig/xattr_keys/ необходимо выполнить команду:

```
sudo update-initramfs -u -k all
```

Также настройка режимов работы может быть выполнена в графической утилите astra-systemsettings («Параметры системы», модуль «Замкнутая программная среда» в разделе «Ограничения программной среды», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_digsig
```

16.1.4.2. Проверка подписи в исполняемых файлах и разделяемых библиотеках

Модуль digsig_verif позволяет проверять исполняемые файлы и разделяемые библиотеки, подписанные встраиваемой подписью, подписью в расширенных атрибутах ФС, а также отсоединенной подписью. В модуле digsig_verif реализован механизм, позволяющий при проверке подписей файлов использовать ключи из разных наборов и проверять несколько подписей. Для проверки подписи используются открытые ключи из основного набора и из

дополнительного набора (см. 16.1.2). Проверка подписи производится при запуске файла на исполнение.

Настройка режима проверки подписи исполняемых файлов осуществляется путем редактирования конфигурационного файла `/etc/digsig/digsig_initramfs.conf` или посредством модуля «Замкнутая программная среда» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку).

Для управления режимом проверки цифровой подписи в исполняемых файлах и разделяемых библиотеках используется параметр `DIGSIG_ELF_MODE` со следующими значениями:

1) `DIGSIG_ELF_MODE=0` — отключить проверку цифровой подписи в исполняемых файлах и разделяемых библиотеках;

2) `DIGSIG_ELF_MODE=1` — включить проверку цифровой подписи в исполняемых файлах и разделяемых библиотеках. При отсутствующей или неверной подписи запуск файла или библиотеки запрещается;

3) `DIGSIG_ELF_MODE=2` — включить отладочный режим проверки цифровой подписи в исполняемых файлах и разделяемых библиотеках. При отсутствующей или неверной подписи запуск файла или библиотеки разрешается с выдачей предупреждения.

Проверку текущего режима работы можно выполнить с помощью команды:

```
cat /sys/digsig/elf_mode
```

Если для параметра `DIGSIG_ELF_MODE` выставлено значение `1` или `2` (включенная проверка или отладочный режим проверки цифровой подписи в исполняемых файлах и разделяемых библиотеках), то возможно установить следующие варианты проверки цифровой подписи с помощью значения параметра `DIGSIG_ELF_VERIFICATION_MODE`:

1) `DIGSIG_ELF_VERIFICATION_MODE=0` — проверять только встраиваемую подпись (используется по умолчанию);

2) `DIGSIG_ELF_VERIFICATION_MODE=1` — проверять только подпись в расширенных атрибутах ФС;

3) `DIGSIG_ELF_VERIFICATION_MODE=2` — проверять первую найденную подпись, если она неверная, то запуск запрещается. Поиск подписи осуществляется в следующем порядке:

   а) встраиваемая подпись;

   б) подпись в расширенных атрибутах ФС;

   в) отсоединенная подпись;

4) DIGSIG_ELF_VERIFICATION_MODE=3 — проверять все типы подписей, используется первая найденная верная подпись;

5) DIGSIG_ELF_VERIFICATION_MODE=4 — проверять встраиваемую подпись и подпись в расширенных атрибутах ФС. Запуск разрешается только при успешной проверке обоих видов подписи.

ВНИМАНИЕ! Установка значения 1 или 4 параметра DIGSIG_ELF_VERIFICATION_MODE приведет к отказу загрузки ОС, если все исполняемые файлы и разделяемые библиотеки не были предварительно подписаны в расширенных атрибутах файловой системы (см. 16.1.6).

После внесения изменений в конфигурационный файл /etc/digsig/digsig_initramfs.conf необходимо от имени учетной записи администратора через механизм sudo выполнить команду:

```
sudo update-initramfs -u -k all
```

16.1.4.3. Проверка подписи в неисполняемых файлах

Модуль digsig_verif позволяет проверять цифровую подпись неисполняемых файлов любых типов в расширенных атрибутах файловой системы. Для проверки подписи используются ключи из основного набора и открытые ключи из дополнительного набора (см. 16.1.2). Проверка подписи производится при открытии файла.

Для проверки подписи в расширенных атрибутах ФС проверяемые файлы должны быть указаны в файле /etc/digsig/xattr_control. Каждая строка в файле задает шаблон имени проверяемого файла в виде маски полного пути.

Пример

Проверка цифровой подписи всех файлов в каталоге /bin, имя которых начинается на lo:

```
/bin/lo
```

Для проверки отдельных файлов (без маски) необходимо указать дополнительный символ «/» в начале пути.

Пример

```
//home/start.sh
```

Настройка режима проверки файлов на основе цифровой подписи в расширенных атрибутах ФС осуществляется путем редактирования конфигурационного фай-

ла /etc/digsig/digsig_initramfs.conf или с помощью модуля «Замкнутая программная среда» в разделе «Ограничения программной среды» графической утилиты astra-systemsettings («Параметры системы», см. электронную справку).

Для управления режимом проверки цифровой подписи в расширенных атрибутах ФС для неисполняемых файлов используются следующие параметры:

1) DIGSIG_XATTR_MODE=0 — отключить проверку цифровой подписи в расширенных атрибутах ФС для неисполняемых файлов;

2) DIGSIG_XATTR_MODE=1 — включить проверку цифровой подписи в расширенных атрибутах ФС для неисполняемых файлов. При отсутствующей или неверной подписи открытие файла запрещается;

3) DIGSIG_XATTR_MODE=2 — включить отладочный режим проверки цифровой подписи в расширенных атрибутах ФС для неисполняемых файлов. При отсутствующей или неверной подписи открытие файла разрешается с выдачей предупреждения.

После внесения изменений в конфигурационный файл /etc/digsig/digsig_initramfs.conf необходимо от имени учетной записи администратора через механизм sudo выполнить команду:

```
sudo update-initramfs -u -k all
```

Проверку текущего режима работы можно выполнить с помощью команды:

```
cat /sys/digsig/xattr_mode
```

16.1.5. Добавление дополнительных ключей

В дополнительный набор может быть добавлен открытый ключ, подписанный ключом изготовителя или ключом из иерархии подписанных изготовителем ключей, или произвольный открытый ключ без требований к его подписи (см. 16.1.2). Возможно как скопировать переданный открытый ключ в соответствующий каталог, так и создать самостоятельно.

Для создания ключей используется GNU Privacy Guard (GnuPG). В доработанном GnuPG доступно использование алгоритма по ГОСТ Р 34.11-2012 для вычисления хеш-функции. Результат вычисления хеш-функции записывается вместе с подписью. Для получения списка доступных алгоритмов выполнить команду:

```
gpg --version
```

Результат выполнения команды:

```
gpg (GnuPG) 1.4.18
```

```
Copyright (C) 2014 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Home: ~/.gnupg
Поддерживаются следующие алгоритмы:
С открытым ключом: RSA, RSA-E, RSA-S, ELG-E, DSA,
           GOST_R 34.10-2001, GOST_R 34.10-2012
Симметричные: 3DES, CAST5, BLOWFISH, AES, AES192,
           AES256, TWOFISH, CAMELLIA128,
           CAMELLIA192, CAMELLIA256
хеш-функции: MD5, SHA1, RIPEMD160, SHA256, SHA384,
           SHA512, SHA224, GOST_R34.11-2012,
           GOST_R34.11-94
Алгоритмы сжатия: Без сжатия, ZIP, ZLIB, BZIP2
```

Для создания ключа необходимо создать ключевую пару (закрытый и открытый ключи) по ГОСТ Р 34.10-2012, выполнив команду:

```
sudo gpg --full-generate-key
```

В процессе выполнения команды необходимо отвечать на запросы системы. На запрос о выборе типа ключа следует указать номер пункта, соответствующий ГОСТ Р 34.10-2012.

Вывод команды:

```
gpg (GnuPG) 2.2.40; Copyright (C) 2022 g10 Code GmbH
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Выберите тип ключа:
(1) RSA и RSA (по умолчанию)
(2) DSA и Elgamal
(3) DSA (только для подписи)
(4) RSA (только для подписи)
(14) Имеющийся на карте ключ
(14) GOST R34.10-2001 (sign only) OBSOLETE
(15) GOST R34.10-2012 (sign only)
Ваш выбор? 15
длина ключей GOST_R34.10-2012 может быть от 1024 до 4096.
Какой размер ключа Вам необходим? (3072)
Запрошенный размер ключа - 3072 бит
Выберите срок действия ключа.
0 = не ограничен
<n> = срок действия ключа - n дней
<n>w = срок действия ключа - n недель
<n>m = срок действия ключа - n месяцев
<n>y = срок действия ключа - n лет
```

```
Срок действия ключа? (0)
Срок действия ключа не ограничен
Все верно? (y/N) y
```

GnuPG должен составить идентификатор пользователя для идентификации ключа.

```
Ваше полное имя: test user
Адрес электронной почты: test@test.ru
Примечание:
Вы выбрали следующий идентификатор пользователя:
"test user <test@test.ru>"

Сменить (N)Имя, (C)Примечание, (E)Адрес; (O)Принять/(Q)Выход? o
Необходимо получить много случайных чисел. Желательно, чтобы Вы
в процессе генерации выполняли какие-то другие действия (печать
на клавиатуре, движения мыши, обращения к дискам); это даст генератору
случайных чисел больше возможностей получить достаточное количество энтропии.

gpg: /root/.gnupg/trustdb.gpg: создана таблица доверия
gpg: создан каталог '/root/.gnupg/openpgp-revocs.d'
gpg: сертификат отзыва записан в '/root/.gnupg/openpgp-revocs.d/07888\
   A89440B749338619DF1FBBB20B915E8F5EA.rev'.
открытый и секретный ключи созданы и подписаны.

pub gP256 2024-06-14 [SC]
07888A89440B749338619DF1FBBB20B915E8F5EA
uid test user <test@test.ru>
```

В созданной ключевой паре открытый ключ имеет идентификатор `07888A89440B749338619DF1FBBB20B915E8F5EA`, с которым он добавляется в хранилище gpg-ключей `/root/.gnupg/pubring.kbx` и по которому он определяется в командах экспорта и копирования.

Если созданный ключ предназначен для подписания в расширенных атрибутах ФС или отсоединенной подписью и нет требований подтверждать подпись открытым ключом изготовителя, то необходимо:

1) экспортировать открытый ключ созданной пары в файл, выполнив команду:

```
sudo gpg --export <идентификатор_ключа> > <имя_файла_ключа>.gpg
```

Пример

```
sudo gpg --export 07888A89440B749338619DF1FBBB20B915E8F5EA > \
    /tmp/secondary_gost_key.gpg
```

2) скопировать полученный файл открытого ключа в каталог /etc/digsig/xattr_keys/.

Если созданный ключ предназначен для подписания любой подписью (в т. ч. встраиваемой) и присутствует ключ, открытый ключ которого подписан изготовителем или ключом из иерархии подписанных изготовителем ключей, то необходимо:

1) подписать открытый ключ созданной пары, выполнив команду:

```
gpg --sign-key <идентификатор_ключа>
```

Пример

```
gpg --sign-key 07888A89440B749338619DF1FBBB20B915E8F5EA
```

2) экспортировать подписанный открытый ключ в файл, выполнив команду:

```
sudo gpg --export <идентификатор_ключа> > <имя_файла_ключа>.gpg
```

Пример

```
gpg --export 07888A89440B749338619DF1FBBB20B915E8F5EA > \
     /tmp/secondary_gost_key_signed.gpg
```

3) скопировать полученный файл подписанного открытого ключа в каталог /etc/digsig/keys.

Для загрузки ключей выполнить команду:

```
sudo update-initramfs -u -k all
```

Для просмотра всех ключей, созданных с помощью gpg или импортированных в gpg, следует использовать команду:

```
sudo gpg --list-keys
```

16.1.6. Подписывание файлов

Открытые ключи из основного набора доступны только для проверки цифровой подписи. Для подписывания файлов необходимо использовать отдельные закрытые ключи, например созданные с помощью gpg (см. 16.1.5) или переданные изготовителем. При этом для проверки этих подписей их открытые ключи должны быть добавлены в дополнительный набор (см. 16.1.2).

Подписывание файлов осуществляется с помощью инструмента командной строки `bsign-integrator` или с помощью модуля «Замкнутая программная среда» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Инструмент `bsign-integrator` позволяет осуществлять все операции с цифровыми подписями и поддерживает все типы цифровых подписей (см. 16.1.2).

Синтаксис команды:

```
sudo bsign-integrator <действие> [параметр] [файл]
```

Описание действий и параметров инструмента `bsign-integrator` приведено в таблицах 59 и 60.

Таблица 59 – Действия инструмента bsign-integrator

Действие	Описание
`-b, --bsign`	Вывести значение инструмента работы с подписями, заданного по умолчанию
`-d, --default <инструмент>`	Указать инструмент работы с подписями, который будет использоваться по умолчанию
`-w, --show-info`	Вывести информацию о подписях для указанного файла
`-s, --sign`	Подписать файл или файлы. С действием должен быть указан единичный файл для подписания или файл со списком файлов (с параметром `-L`). Если в параметрах не указаны типы подписей, нужный тип определяется автоматически
`-a, --sign-analyse <отпечаток>`	Вывести список файлов, подпись которых соответствует указанному SHA1 отпечатку сертификата ключа проверки цифровой подписи. Если в параметрах не указаны типы подписей, то выполняется поиск по файлам со всеми видами цифровой подписи
`-T, --tools-list`	Вывести список поддерживаемых инструментов работы с подписями

Таблица 60 – Параметры инструмента bsign-integrator

Параметр	Описание
`-C, --current <путь>`	Указать путь к инструменту работы с подписями, который будет использован в текущей команде
`-L, --input_list`	Использовать файл со списком файлов в качестве входных значений
`-E, --elf-only`	Подпись в `elf`-секции файла (только для файлов типа ELF)
`-A, --attached`	Подпись в конце файла
`-X, --xattr-only`	Подпись в расширенных атрибутах ФС
`-D, --detached`	Подпись в отдельном файле (отсоединенная подпись)
`-k=<идентификатор>, --key=<идентификатор>`	Указать идентификатор ключа для использования в текущей команде. Используется вместе с параметром `--passphrase`

Окончание таблицы 60

Параметр	Описание
-p=<пароль>, --passphrase=<пароль>	Указать пароль ключа, использованного в параметре --key
-j, --jobs=<количество>	Количество процессов, которое разрешено запускать одновременно. Если не задано, то соответствует количеству процессоров в системе

Более подробное описание инструмента командной строки bsign-integrator приведено на справочной странице man bsign-integrator.

Примеры:

1. Подписать один файл созданным ключом с автоматическим определением требуемого типа подписи:

```
sudo bsign_integrator -s -p=JHbd<jcd \
    -k=07888A89440B749338619DF1FBBB20B915E8F5EA filetosign
```

2. Подписать файлы из списка:

```
sudo bsign_integrator -s -p=JHbd<jcd \
    -k=07888A89440B749338619DF1FBBB20B915E8F5EA -L filelist
```

3. Подписать исполняемый файл в расширенных атрибутах ФС и отсоединенной подписью:

```
sudo bsign_integrator -s -p=JHbd<jcd \
    -k=07888A89440B749338619DF1FBBB20B915E8F5EA -X -D filetosign
```

4. Подписать созданным ключом все файлы типа ELF в расширенных атрибутах ФС:

```
sudo find / -type f -exec file {} \; | grep " ELF " | cut -d: -f1 > ELFS
sudo bsign-integrator -s -k=07888A89440B749338619DF1FBBB20B915E8F5EA \
    -p=JHbd<jcd  -X -L ELFS
```

Примечание. Данный процесс является очень продолжительным и может занять несколько часов.

16.2. Режим Киоск-2

Режим Киоск-2 — это инструмент подсистемы безопасности PARSEC для ограничения возможностей, предоставляемых непривилегированным пользователям. Работу режима Киоск-2 обеспечивает пакет parsec-kiosk2.

16.2.1. Профили пакета parsec-kiosk2

Профили пользователей parsec-kiosk2 располагаются в каталоге /etc/parsec/kiosk2 и служат для применения ограничений действий пользователей. Профиль пользователя — это текстовый файл, имя которого совпадает с именем пользователя. Один профиль пользователя служит для установки ограничений только для одного пользователя.

Если при включенном режиме Киоск-2 в /etc/parsec/kiosk2 отсутствует профиль пользователя, то права данного пользователя в ОС не ограничиваются.

ВНИМАНИЕ! Если для пользователя создан пустой профиль, то данному пользователю запрещены любые действия.

Системные профили parsec-kiosk2 располагаются в каталоге /etc/parsec/kiosk2-profiles и также служат для применения ограничений действий пользователей. Содержимое файла системного профиля представляет собой список файлов и профилей приложений с абсолютными или относительными путями, а также с информацией о правах доступа и владельцах этих файлов. Системный профиль можно добавить в профили нескольких пользователей, тогда для всех этих пользователей будет доступно использование приложения в соответствии с ограничениями в правах доступа и владельцах, указанных в системном профиле данного приложения.

16.2.2. Синтаксис профилей parsec-kiosk2

Синтаксис разрешения на доступ к файлу:

```
+file rwc u/o: <имя_файла>
```

или на доступ к ссылке:

```
+link rwc uo: <имя_файла>
```

где rwc — это доступ к файлу с правами на чтение (r), запись (w) и создание (c), которые указываются при наличии соответствующего разрешения;
u — доступ владельцу файла;
o — доступ остальным пользователям;
<имя_файла> — файл, права доступа к которому назначаются. При этом имя файла для ссылки должно соответствовать существующей символьной ссылке. Целевой файл для ссылки, который не обязан существовать в момент загрузки профиля, добавляется в список доступных.

Примечание. В синтаксисе профилей режима Киоск-2 можно использовать кириллицу.

В синтаксисе Киоск-2 также могут быть использованы метасимволы, например, в имени файла (см. таблицу 61).

ВНИМАНИЕ! Метасимволы в имени файла для ссылки не интерпретируются.

Таблица 61

Метасимвол	Описание
`**`	Ноль или более произвольных символов, включая символ «/»
`*`	Ноль или более произвольных символов, исключая символ «/»
`?`	Один произвольный символ, исключая символ «/»
`[набор_символов]`	Один из символов, принадлежащий набору. В наборе любые символы теряют свое специальное значение. Символ «]» может входить в набор только на первой позиции
`\d`	Одна десятичная цифра
`\D`	Одна или более десятичная цифра
`\h`	Одна шестнадцатеричная цифра
`\H`	Одна или более шестнадцатеричная цифра
`\xNN`	Байт с заданным значением
`\z`	Пустая строка или подстрока `(deleted)`, которая добавляется к еще не созданным или уже удаленным файлам. Данная конструкция используется при управлении созданием файла. Пример `+file wc u: /home/*/.config/example/example.conf\z` В этом случае разрешается доступ как к файлу `example.conf (deleted)`, так и к `example.conf` Если удалить `\z`, то доступ к `example.conf (deleted)` будет запрещен. Открытие существующего файла на запись будет успешным, а создание нового — нет
`\<любой_другой_символ>`	Соответствующий символ без специальной интерпретации, например, `\\,\[,\*)`

Для включения одного профиля в другой профиль используется `@include`:

```
@include other-profile-name
```

16.2.3. Синтаксис, совместимый с `parsec-kiosk`

В профилях режима Киоск-2 корректно распознаются строки, написанные по синтаксису режима киоска (устаревший инструмент для ограничений действий пользователя). При этом при использовании синтаксиса режима киоска в режиме Киоск-2:

1) в имени файла спецсимволы не интерпретируются;
2) любое из прав на чтение (r) или исполнение (x) преобразуется в право на чтение (r), а право на запись (w) преобразуется в права на запись (w) и создание (c);
3) правила одинаково применяются для доступа как владельцев, так и других пользователей;
4) строка обязательно должна начинаться с символа «"» или «/»;
5) если файл представляет символьную ссылку, то режим Киоск-2 обрабатывает целевой файл ссылки;
6) имя профиля должно начинаться только с латинской буквы, а также в нем должен отсутствовать символ «/».

Если профиль, созданный в режиме киоска, отредактировать в `fly-admin-kiosk`, то он сохранится с использованием синтаксиса режима Киоск-2.

16.2.4. Работа с Киоск-2 через консоль

16.2.4.1. Включение и выключение Киоск-2

При первом запуске ОС режим Киоск-2 выключен. Чтобы включить режим контроля доступа Киоск-2 и применять ограничения на работу с файлами, требуется выполнить команду:

```
echo 1 | sudo tee /sys/module/parsec/parameters/uc_enforce
```

Для включения протоколирования попыток нарушений установленных фильтров доступа выполнить команду:

```
echo 1 | sudo tee /sys/module/parsec/parameters/uc_complain
```

ВНИМАНИЕ! Результаты применения данных команд будут действительны до перезагрузки ОС.

Для включения режима контроля доступа Киоск-2 и протоколирования с сохранением данного состояния после перезагрузки требуется, соответственно, выполнить команды:

```
echo 1 | sudo tee /etc/parsec/kiosk2_enforce
echo 1 | sudo tee /etc/parsec/kiosk2_complain
```

и перезагрузить ОС.

Результаты применения этих команд будут действительны после перезагрузки ОС.

Для выключения режима контроля доступа Киоск-2 и протоколирования в соответствующих командах требуется заменить `1` на `0`.

Для того чтобы избирательно включать режим Киоск-2 и исключать выбранных пользователей, требуется:

1) создать каталог `disabled` в `/etc/parsec/kiosk2`:

```
cd /etc/parsec/kiosk2
sudo mkdir disabled
```

2) перенести в каталог `disabled` профили тех пользователей, для которых не должны применяться ограничения:

```
sudo mv <имя_пользователя1> <имя_пользователя2> disabled
```

3) включить режим Киоск-2:

```
echo 1 | sudo tee /sys/module/parsec/parameters/uc_enforce
```

После выполнения данных действий ограничения режима Киоск-2 будут применены ко всем пользователям, кроме `<имя_пользователя1>`, `<имя_пользователя2>` и тех пользователей, для которых не были созданы профили. После перезагрузки ОС режим Киоск-2 вернется в состояние, указанное в `/etc/parsec/kiosk2_enforce`. При повторном включении режима Киоск-2 исключения для `<имя_пользователя1>`, `<имя_пользователя2>` и тех пользователей, для которых не были созданы профили, сохранятся и ограничения доступа к ним не будут применены.

16.2.4.2. Протоколирование процессов

Для того чтобы ограничения на работу приложений выполнялись корректно и без ошибок, требуется учитывать все зависимости файлов данного приложения. Поэтому при создании системного профиля приложения рекомендуется выполнить протоколирование процессов при работе с соответствующим приложением. Результат протоколирования процессов показывает все необходимые файлы приложения и пути к ним. Для выполнения протоколирования процессов требуется:

1) выключить режим контроля доступа Киоск-2:

```
echo 0 | sudo tee /sys/module/parsec/parameters/uc_enforce
```

2) включить протоколирование процессов:

```
echo 1 | sudo tee /sys/module/parsec/parameters/uc_complain
```

3) создать профиль пользователя:

```
true | sudo tee /etc/parsec/kiosk2/<имя_пользователя>
```

4) запустить сессию пользователя <имя_пользователя>:

```
su - <имя_пользователя>
```

5) запустить протоколируемое приложение;

6) выйти из сессии пользователя <имя_пользователя>:

```
exit
```

7) выключить протоколирование процессов:

```
echo 0 | sudo tee /sys/module/parsec/parameters/uc_complain
```

8) получить сообщения о запрещенных операциях:

```
sudo dmesg | grep PARSEC-UC:
```

16.2.4.3. Создание профиля

Для создания системного профиля протоколируемого приложения требуется в каталоге `/etc/parsec/kiosk2-profiles` создать файл, задав в качестве его имени название приложения, и вписать в него все пути с учетом синтаксиса из результатов выполнения команды `dmesg | grep PARSEC-UC:` в соответствии с пунктом 8) перечисления на странице 300.

Для создания профиля пользователя требуется в каталоге `/etc/parsec/kiosk2` создать файл с именем, совпадающим с именем данного пользователя.

16.2.5. Графическая утилита управления профилями

Для управления режимом Киоск-2 и профилями, расположенными в каталогах `/etc/parsec/kiosk2` и `/etc/parsec/kiosk2-profiles`, используется графическая утилита `fly-admin-kiosk`. Описание утилиты приведено в электронной справке.

ВНИМАНИЕ! Для работы с профилями режима киоска с использованием утилиты `fly-admin-kiosk` необходимо адаптировать их синтаксис и поместить профили в каталоги `/etc/parsec/kiosk2` и `/etc/parsec/kiosk2-profiles` пакета `parsec-kiosk2` соответственно (см. 16.2.3).

Для запуска утилиты выполнить в консоли команду:

```
sudo fly-admin-kiosk
```

или перейти «меню «Пуск» — Панель управления — Безопасность — Системный киоск».

16.3. Графический киоск

Для ограничения прав запуска программ локальными пользователями возможно использовать режим графического киоска. Настройка режима графического киоска выполняется с использованием модуля «Графический киоск» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_graphics_kiosk
```

ВНИМАНИЕ! Настройка режима графического киоска с помощью модуля «Графический киоск» графической утилиты `astra-systemsettings` осуществляется администратором на максимальном уровне мандатного контроля целостности, установленном в ОС.

При входе в пользовательскую сессию в режиме графического киоска отображается рабочий стол с разрешенными для запуска программами. Пользователю предоставляется доступ к каталогам, ярлыкам и т.д., необходимым для работы в разрешенных программах.

Графический киоск также может работать в режиме одной программы — разрешенная программа автоматически запускается при входе в сессию, а при выходе из программы сессия завершается.

16.4. Графический киоск в режиме «Мобильный»

Графический киоск в режиме «Мобильный» ограничивает доступ пользователя к функциям системы и возможность запуска графических приложений. Настройка графического киоска выполняется администратором в конфигурационном файле `/etc/mobile-kiosk/mobile-kiosk.conf` (если указанные каталог и конфигурационный файл отсутствуют, их необходимо создать). В конфигурационном файле необходимо указать имя учетной записи пользователя, для которого настраивается графический киоск, и в качестве значения параметра `Applications` перечислить desktop-файлы приложений, которые можно будет запускать указанному пользователю:

```
[<имя_пользователя>]
Applications=<приложение1>,<приложение2>,...
```

Пример

Настройка киоска для пользователя `user`, разрешающая запуск только приложений «Галерея» и «Музыка»:

```
[user]
```

```
Applications=fly-gallery,fly-music
```

Киоск для указанного пользователя будет включаться автоматически в начале каждой последующей сессии данного пользователя.

В графическом киоске пользователю будут доступны только указанные приложения, а также следующие функции системы:

1) просмотр и удаление уведомлений;
2) строка поиска;
3) изменение громкости звука и яркости экрана;
4) завершение работы.

Остальные приложения и функции системы (в том числе добавление, перемещение и удаление значков, изменение обоев, добавление виджетов) пользователю будут недоступны.

Если в конфигурационном файле `/etc/mobile-kiosk/mobile-kiosk.conf` в качестве значения параметра `Applications` указано одно приложение и дополнительно указан параметр `SingleMode=true`, киоск будет работать в одиночном режиме — указанное приложение будет запускаться автоматически при входе пользователя в сессию и пользователь сможет работать только в данном приложении. При завершении работы приложения будет автоматически завершена пользовательская сессия. Из функций системы пользователю будет доступно только завершение работы.

Пример

Настройка киоска для пользователя `user`, разрешающая работать в одиночном режиме в приложении «Калькулятор»:

```
[user]
Applications=org.kde.kalk
SingleMode=true
```

16.5. Изоляция приложений

Для обеспечения изолированного выполнения графических и консольных приложений в состав ОС входит инструмент Firejail . Целью применения Firejail является минимизация риска компрометации основной ОС при запуске недоверенных или потенциально уязвимых программ.

Для обеспечения изоляции приложений в Firejail используются:

- механизм пространств имен `namespaces`;

- фильтрация системных вызовов `seccomp-bpf`.

После запуска инструмент Firejail и все его дочерние процессы используют отдельные представления ресурсов ядра, таких как сетевой стек, таблица процессов и точки монтирования.

Firejail не требует подготовки системного образа: состав окружения формируется при запуске на основе содержимого текущей ФС, и удаляется после завершения работы приложения.

При использовании Firejail предоставляются средства задания правил доступа к ФС, позволяющие:

- определять файлы и каталоги, к которым разрешен или запрещен доступ;
- предоставлять доступ к файлам или каталогам только для чтения;
- подключать для данных временные ФС (`tmpfs`);
- совмещать каталоги через `bind-mount` и `overlayfs`.

При установке ОС установка инструмента Firejail выполняется автоматически.

В целях повышения безопасности при установке Firejail снимается бит `suid`. Для того чтобы с помощью Firejail можно было запускать приложения, необходимо включить этот бит, выполнив от имени администратора команду:

```
chmod u+s /usr/bin/firejail
```

В состав ОС входят профили изоляции системных вызовов для большинства популярных приложений, в том числе для Firefox, Chromium, VLC.

Для выполнения программы в режиме изоляции требуется указать имя приложения в качестве параметра для инструмента Firejail, например:

```
firejail firefox
```

Примечание. Для запуска браузера Firefox может потребоваться внести изменения в конфигурационный файл `/etc/firejail/firefox.profile`, заменив строку `seccomp` на строку:

```
seccomp.drop @clock,@cpu-emulation,@debug,@module,@obsolete,@raw-io,
@reboot,@resources,@swap,acct,add_key,bpf,fanotify_init,io_cancel,
io_destroy,io_getevents,io_setup,io_submit,ioprio_set,kcmp,keyctl,
mount,name_to_handle_at,nfsservctl,ni_syscall,open_by_handle_at,
personality,pivot_root,process_vm_readv,ptrace,remap_file_pages,
request_key,setdomainname,sethostname,syslog,umount,umount2,userfaultfd,
vhangup,vmsplice
```

16.6. Функции безопасности системы

Пакет astra-safepolicy содержит инструменты управления функциями безопасности системы. Описание инструментов приведено в 16.6.3-16.6.32.

Все инструменты из состава пакета astra-safepolicy поддерживают стандартный набор параметров вызова, приведенный в 16.6.1. Некоторые инструменты дополнительно к стандартному набору параметров вызова поддерживают дополнительные параметры. Описание дополнительных параметров приведено в описании соответствующего инструмента.

Инструменты пакета astra-safepolicy выступают в роли команд-переключателей, осуществляющих включение и выключение соответствующих функций безопасности. Для просмотра состояния некоторых команд-переключателей можно использовать инструмент astra-security-monitor из состава пакета astra-safepolicy, описание инструмента приведено в 16.6.2.

16.6.1. Общие параметры вызова переключателей

Все команды-переключатели, входящие в состав пакета astra-safepolicy, поддерживают стандартный набор параметров вызова. Синтаксис использования команды-переключателя:

```
<имя_команды-переключателя> <параметр>
```

Перечень параметров вызова приведен в таблице 62.

Таблица 62

Параметр	Описание
enable	Включить действие, выполняемое функцией безопасности
disable	Выключить действие, выполняемое функцией безопасности
status	Отобразить, если возможно, фактическое состояние переключателя на текущий момент времени. Результат выполнения команды: - АКТИВНО — системные ограничения применяются (функция выполняется); - НЕАКТИВНО — системные ограничения не применяются (функция отключена). Например, для некоторых команд-переключателей вызовы enable/disable меняют положение переключателя (см. параметр is-enabled), но для изменения состояния требуется перезагрузка. Для этих команд-переключателей до момента перезагрузки вывод is-enabled и вывод status будут различаться.
is-enabled	Отобразить положение переключателя. Результат выполнения команды: - ВКЛЮЧЕНО — системные ограничения включены; - ВЫКЛЮЧЕНО — системные ограничения выключены

Для некоторых инструментов при первом их включении создается соответствующий юнит службы `systemd`. В дальнейшем состояние данной функции безопасности после перезагрузки системы устанавливается в соответствии с положением переключателя путем запуска соответствующего юнита, если запуск юнита разрешен. Список юнитов, созданных инструментами, можно просмотреть в файле `astra-safepolicy.target` (в строках, начинающихся с `Wants=`).

16.6.2. Монитор безопасности

Инструмент командной строки `astra-security-monitor` отображает информацию о состоянии некоторых функций безопасности, а также выводит информацию о состоянии функции безопасности по ее идентификатору.

Примечание. Отображение состояния функции безопасности также зависит от наличия установленных в системе соответствующих пакетов или служб.

Для функций безопасности, информацию о которых выводит инструмент командной строки, возможны следующие состояния:

- `ВКЛЮЧЕНО` — функция безопасности активна;
- `ВЫКЛЮЧЕНО` — функция безопасности неактивна;
- `ВКЛЮЧАЕТСЯ` — функция безопасности находится в процессе включения или будет включена после перезагрузки, но в настоящий момент неактивна;
- `ВЫКЛЮЧАЕТСЯ` — функция безопасности находится в процессе выключения или будет выключена после перезагрузки, но в настоящий момент активна;
- `ЧАСТИЧНО` — функция безопасности включена, но не все параметры, контролируемые этой функцией, соответствуют заданным по умолчанию.

При отображении состояний отдельных функций безопасности учитывается следующее:

1) при выводе информации о состоянии МКЦ на файловой системе проверяется соответствие меток целостности объектов ФС меткам целостности, указанным в конфигурационном файле `/etc/parsec/fs-ilev.conf` (см. 4.10). При несоответствии меток целостности выводится сообщение о количестве файловых объектов, метка целостности которых не соответствует заданной в конфигурационном файле:
   а) «ниже» — метка целостности файлового объекта ниже заданной;
   б) «выше» — метка целостности файлового объекта выше заданной;
   в) «норма» — метка целостности файлового объекта соответствует заданной.

   Список файловых объектов, метка целостности которых не соответствует заданной в файле `/etc/parsec/fs-ilev.conf`, можно вывести командой:

```
sudo set-fs-ilev status -v
```

2) для функции проверки подписи в расширенных атрибутах `xattr` — включена ли проверка подписи не только в исполняемых файлах, но и в других файлах, которые подписаны в `xattr` соответствующей утилитой.

Дополнительно монитор безопасности выводит следующую информацию:

1) запрет входа `root` по ssh (если установлены средства удаленного подключения ssh) — запрет удаленного входа в систему пользователю `root`. По умолчанию `root` не может войти через ssh, функция запрета имеет состояние `ВКЛЮЧЕНО`;

2) безопасный вход в домен (если компьютер введен в домен) — значение `ВКЛЮЧЕНО`, если в файле `/etc/parsec/parsec.conf` параметр `login_local` имеет значение `admin` (вход для локального пользователя разрешен, если пользователь входит в группу `astra-admin`) или значение `no` (вход для локальных пользователей запрещен);

3) системный киоск — значение `ВКЛЮЧЕНО`, если системный киоск включен и настроен хотя бы для одного пользователя;

4) графический киоск — значение `ВКЛЮЧЕНО`, если графический киоск настроен хотя бы для одного пользователя.

Подробное описание инструмента приведено на справочной странице `man astra-security-monitor`.

Просмотреть информацию о состоянии функций безопасности также можно с помощью модуля «Монитор безопасности» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_security_monitor
```

16.6.3. Установка квот на использование системных ресурсов

Инструмент командной строки `astra-ulimits-control` включает или выключает ограничения на использование некоторых ресурсов системы для предотвращения нарушений доступности системы в результате исчерпания ресурсов. Параметры системных ограничений задаются в файле `/etc/security/limits.conf`.

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

Изменение режима ограничений не повлияет на уже вошедших в систему пользователей и будет применено только при следующем входе.

Описание инструмента приведено на справочной странице `man astra-ulimits-control`.

Управление системными ограничениями также может осуществляться с помощью модуля «Квоты» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_quotas
```

Дополнительные ограничения на получение доступа и выполнение кода на уровне ядра ОС для привилегированного пользователя обеспечиваются модулем lockdown в соответствии с 16.8.

16.6.4. Запрет установки бита исполнения

В ОС реализована возможность запрета установки бита исполнения, обеспечивающая предотвращение несанкционированного запуска исполняемых файлов и сценариев для командной оболочки.

Запрет распространяется на все учетные записи, кроме `root`.

Если в системе включен запрет установки бита исполнения, то установка пакетов программ, создающих файлы с битом исполнения, будет завершаться с ошибкой.

Для включения запрета используется инструмент командной строки `astra-nochmodx-lock`. Параметры вызова, используемые инструментом, приведены в таблице 62.

Также запрет установки бита исполнения включается автоматически при включении блокировки интерпретаторов с помощью инструмента `astra-interpreters-lock` (см. 16.6.7).

Изменение режима запрета вступает в действие немедленно.

Описание инструмента приведено на справочной странице `man astra-nochmodx-lock`.

Управление режимом запрета установки бита исполнения также может осуществляться с помощью модуля «Системные параметры» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_system_parameters
```

16.6.5. Блокировка консоли для пользователей

Инструмент командной строки `astra-console-lock` осуществляет блокировку доступа к консоли и терминалам для пользователей, не входящих в группу `astra-console`. Параметры вызова, используемые данным инструментом, приведены в таблице 62.

Если при включении блокировки группа `astra-console` отсутствует в ОС, то она будет создана автоматически. При этом в нее будут включены пользователи, состоящие в группе `astra-admin` на момент включения этой функции.

Изменение блокировки вступает в действие немедленно.

Описание инструмента приведено на справочной странице `man astra-console-lock`.

Управление блокировкой консоли для пользователей также может осуществляться с помощью модуля «Системные параметры» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_system_parameters
```

16.6.6. Блокировка консоли в режиме «Мобильный»

В режиме «Мобильный» для блокировки доступа к консоли необходимо заблокировать доступ к ней для пользователей, не входящих в группу `astra-console`, с помощью инструмента `astra-console-lock` (см. 16.6.5).

Дополнительно необходимо заблокировать использование командной строки из меню поиска, вызываемого комбинацией клавиш **<ALT+F2>**. Для этого требуется в файл `/etc/xdg/kdeglobals` добавить следующие строки:

```
[KDE Action Restrictions][$i]
run_command=false
```

Блокировка комбинации клавиш **<ALT+F2>** осуществляется для всех пользователей, в т.ч. для администратора.

Для разблокировки комбинации клавиш **<ALT+F2>** требуется в файле `/etc/xdg/kdeglobals` для параметра `run_command` указать значение `true`:

```
[KDE Action Restrictions][$i]
run_command=true
```

или удалить из файла строки:

```
[KDE Action Restrictions][$i]
run_command=false
```

16.6.7. Блокировка интерпретаторов

В ОС реализована функция блокировки следующих интерпретаторов:

- bash,
- csh,
- dash,
- expect,
- irb,
- ksh,
- lua,
- nodejs,
- perl,
- php,
- ptksh,
- python,
- qemu-*-static,
- ruby,
- tcl,
- tk,
- zsh.

Для блокировки интерпретаторов (кроме интерпретатора bash) используется инструмент командной строки `astra-interpreters-lock`.

Для блокировки интерпретатора bash используется инструмент `astra-bash-lock`.

ВНИМАНИЕ! Блокировка интерпретатора bash может ограничить функционал некоторых программ и служб, использующих bash, что приведет к нарушению штатной работы ОС.

ВНИМАНИЕ! После блокировки интерпретатора bash консольный вход пользователей с оболочкой bash будет невозможен.

Блокировка интерпретаторов не распространяется на учетные записи из группы `astra-admin`.

Параметры вызова, используемые данными инструментами, приведены в таблице 62.

Блокировка интерпретаторов предотвращает несанкционированное использование интерпретатора для выполнения кода напрямую из командной строки или из неименованного канала (pipe). При этом сценарии, написанные для этих интерпретаторов, выполняются в штатном режиме (у файла сценария при этом должны быть выставлены права на выполнение).

При включении блокировки интерпретаторов автоматически включается запрет установки бита исполнения (см. 16.6.4). Это обусловлено тем, что пользователь может создать сценарий, назначить ему бит исполнения и запустить, обойдя таким образом блокировку интерпретаторов.

Изменение блокировки вступает в действие немедленно.

Описание инструментов приведено на справочных страницах `man astra-interpreters-lock` и `man astra-bash-lock`.

Управление блокировкой интерпретаторов также может осуществляться с помощью модуля «Системные параметры» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_system_parameters
```

16.6.8. Блокировка макросов

Блокировка исполнения макросов в документах LibreOffice осуществляется с помощью инструмента командной строки `astra-macros-lock`.

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

При включении блокировки макросов уровень безопасности запуска макросов LibreOffice («Сервис — Параметры — LibreOffice — Безопасность — Безопасность макросов») устанавливается в значение «Очень высокий», при котором запуск макросов возможен только из библиотек, расположенных в доверенном каталоге `/opt/libreoffice` (каталог создается вручную). До выключения блокировки изменение уровня защищенности и доверенного источника недоступно, в т. ч. администратору.

Изменение блокировки вступает в действие немедленно. Если изменение блокировки осуществлялось при запущенной программе пакета `libreoffice`, то оно вступает в действие при следующем запуске программы.

Описание инструмента приведено на справочной странице `man astra-macros-lock`.

Управление блокировкой исполнения макросов также может осуществляться с помощью модуля «Системные параметры» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_system_parameters
```

16.6.9. Блокировка трассировки ptrace

Блокировка трассировки `ptrace` осуществляется с помощью инструмента командной строки `astra-ptrace-lock`, который устанавливает максимальные ограничения на использование механизма `ptrace` путем задания параметру `kernel.yama.ptrace_scope` значения `3`.

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

Значение устанавливается сразу при включении данной функции и сохраняется после перезагрузки.

Функция не может быть выключена без перезагрузки.

Управление блокировкой трассировки `ptrace` также может осуществляться с помощью модуля «Системные параметры» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_system_parameters
```

Дополнительные возможности по ограничению применения трассировки процессов ОС обеспечиваются модулем безопасности yama в соответствии с 16.7.

16.6.10. Блокировка клавиши <SysRq>

Блокировка клавиши **<SysRq>** осуществляется с помощью инструмента командной строки `astra-sysrq-lock` путем изменения значения параметра `kernel.sysrq`. Значение сохраняется в файле `/etc/sysctl.d/999-astra.conf`. При этом блокируются функции системы, доступные при нажатии клавиши **<SysRq**>.

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

По умолчанию данная функция безопасности включена, т.е. клавиша **<SysRq>** не работает.

ВНИМАНИЕ! Если в командной строке ядра указан параметр `sysrq_always_enabled`, то при включении функции блокировки клавиши **<SysRq>** вызов `status` будет выводить `НЕАКТИВНО`, а вызов `is-enabled` будет выводить `ВКЛЮЧЕНО`.

Для управления блокировкой клавиши **<SysRq>** может непосредственно использоваться инструмент `sysctl`:

1) для включения блокировки клавиши **<SysRq**> команда:

```
sysctl -w kernel.sysrq=0
```

2) для выключения блокировки клавиши **<SysRq**> команда:

```
sysctl -w kernel.sysrq=1
```

3) для проверки состояния используется команда:

```
cat /proc/sys/kernel/sysrq
```

результат выполнения команды:
- `0` — блокировка клавиш SysRq включена;
- `1` — блокировка клавиш SysRq выключена.

Управление блокировкой клавиши **<SysRq>** также может осуществляться с помощью модуля «Системные параметры» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_system_parameters
```

Примечание. Для управления блокировкой клавиши **<SysRq>** рекомендуется использовать инструмент `astra-sysrq-lock` или графическую утилиту `astra-systemsettings`.

16.6.11. Управление автоматическим входом

Управление автоматическим входом в графическую среду осуществляется с помощью инструмента командной строки `astra-autologin-control`.

Параметры вызова, используемые данным инструментом, приведены в таблице 62. При использовании вызова `enable` требуется дополнительный параметр `<имя_пользователя>`, от имени которого будет выполняться автоматический вход в систему после загрузки.

Описание инструмента приведено на справочной странице `man astra-autologin-control`.

16.6.12. Блокировка запуска программ пользователями

Управление блокировкой запуска пользователями программ `df`, `chattr`, `arp`, `ip` осуществляется с помощью инструмента командной строки `astra-commands-lock`. Данные программы необходимо блокировать при обработке в одной системе информации разных уровней конфиденциальности, т. к. с их помощью можно организовать скрытый канал передачи информации между уровнями.

Программы блокируются путем выставления на них прав доступа `750 (rwx r-x ---)`.

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

Изменение блокировки вступает в действие немедленно.

Описание инструмента приведено на справочной странице `man astra-commands-lock`.

Управление блокировкой запуска программ пользователями также может осуществляться с помощью модуля «Системные параметры» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_system_parameters
```

16.6.13. Запуск контейнеров Docker на пониженном уровне МКЦ

Инструмент командной строки `astra-docker-isolation` применяется для перевода службы Docker с высокой категории целостности на категорию целостности `2`, для чего в каталоге `/etc/systemd` размещается override-файл для службы Docker.

Данный инструмент доступен к использованию, если в ОС установлена программа Docker.

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

Изменения вступают в силу после перезапуска службы Docker.

Описание инструмента приведено на справочной странице `man astra-docker-isolation`.

16.6.14. Управление сетевыми службами

Инструмент командной строки `astra-ilev1-control` при его включении переводит сетевые службы `apache2`, `dovecot` и `exim4` с высокой категории целостности на категорию целостности `1`, для чего в каталоге `/etc/systemd` размещаются override-файлы для соответствующих служб.

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

Если указанные службы не были установлены при включении этой функции, то функция будет автоматически применена к данным службам при их установке.

Изменения вступают в силу после перезапуска служб.

Описание инструмента приведено на справочной странице `man astra-ilev1-control`.

16.6.15. Управление загрузкой модуля ядра lkrg

Инструмент командной строки `astra-lkrg-control` управляет загрузкой модуля ядра lkrg и настраивает его автозагрузку. Модуль ядра lkrg обеспечивает обнаружение некоторых уязвимостей и защиту пространства памяти ядра от модификации.

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

Если установлен пакет `lkrg`, то при включении данной функции сразу загружается модуль ядра lkrg и настраивается его автозагрузка при загрузке системы (на этапе загрузки `initrd`).

При выключении функции модуль lkrg сразу выгружается из ядра и удаляется из автозагрузки.

Описание инструмента приведено на справочной странице `man astra-lkrg-control`.

16.6.16. Блокировка автоматического конфигурирования сетевых подключений

Инструмент командной строки `astra-noautonet-control` блокирует автоматическое конфигурирование сетевых подключений путем блокировки работы служб `NetworkManager` (`network-manager`) и `connman`, а также выключает отображение элемента управления сетевыми подключениями в области уведомлений панели задач.

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

Изменение блокировки вступает в действие немедленно.

Описание инструмента приведено на справочной странице `man astra-noautonet-control`.

16.6.17. Отключение отображения меню загрузчика

Инструмент `astra-nobootmenu-control` отключает отображение меню загрузчика GRUB 2 при загрузке системы. Параметры вызова, используемые данным инструментом, приведены в таблице 62.

Для применения изменений требуется перезагрузка.

Описание инструмента приведено на справочной странице `man astra-nobootmenu-control`.

16.6.18. Ограничение на форматирование съемных носителей

Инструмент командной строки `astra-format-lock` устанавливает или отменяет необходимость запроса пароля администратора при форматировании съемных носителей информации.

Параметры вызова, используемые данным инструментом, приведены в таблице 62. По умолчанию пароль администратора запрашивается.

Описание инструмента приведено на справочной странице `man astra-format-lock`.

16.6.19. Запрет монтирования съемных носителей

Инструмент командной строки `astra-mount-lock` запрещает монтирование съемных носителей информации пользователям, входящим в группу `floppy`, при этом запрет не распространяется на пользователей из группы `astra-admin`.

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

Изменение вступает в действие немедленно.

Описание инструмента приведено на справочной странице `man astra-mount-lock`.

Управление запретом также может осуществляться с помощью модуля «Системные параметры» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_system_parameters
```

16.6.20. Включение на файловой системе режима работы «только чтение»

Инструмент командной строки `astra-overlay` включает overlay (режим работы «только чтение») на корневой ФС.

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

После включения функции безопасности фактическое содержимое корневой ФС монтируется в overlay одновременно с ФС, хранящейся в памяти. Изменения файлов сохраняются только в памяти, а ФС на носителе остается без изменений. После перезагрузки все изменения теряются, и система каждый раз загружается в исходном состоянии.

Данная функция может применяться в тех случаях, когда носитель, на котором расположена корневая ФС, аппаратно защищен от записи либо необходимо программно защитить его от изменений.

Функционал overlay не касается файловых систем, хранящихся на отдельных разделах, отличных от корневого. Например, если каталог `/home` хранится на отдельном разделе или носителе, вносимые в него изменения будут сохраняться после перезагрузки.

Для применения изменений требуется перезагрузка.

Описание инструмента приведено на справочной странице `man astra-overlay`.

Управление режимом также может осуществляться с помощью модуля «Системные параметры» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_system_parameters
```

16.6.21. Запрет выключения компьютера пользователями

Инструмент командной строки `astra-shutdown-lock` используется для запрета выключения компьютера пользователями. Параметры вызова, используемые данным инструментом, приведены в таблице 62.

При включении данной функции добавляется политика polkit, которая запрещает выключение компьютера без ввода пароля администратора. Также при включении инструмента блокируется возможность перезагрузки компьютера путем нажатия комбинации клавиш **<Ctrl+Alt+Delete>**.

Изменение вступает в действие после перезагрузки.

Описание инструмента приведено на справочной странице `man astra-shutdown-lock`.

Управление запретом выключения компьютера пользователями также может осуществляться с помощью модуля «Системные параметры» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку).

Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_system_parameters
```

16.6.22. Управление вводом пароля для sudo

Инструмент командной строки `astra-sudo-control` включает и выключает требование ввода пароля при использовании механизма `sudo`. Параметры вызова, используемые данным инструментом, приведены в таблице 62.

Изменение вступает в действие немедленно.

Описание инструмента приведено на справочной странице `man astra-sudo-control`.

Управление вводом пароля для `sudo` также может осуществляться с помощью модуля «Системные параметры» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_system_parameters
```

16.6.23. Блокировка использования утилиты sumac

Инструмент командной строки `astra-sumac-lock` блокирует работу утилит `sumac` и `fly-sumac`. Параметры вызова, используемые данным инструментом, приведены в таблице 62.

Если данная блокировка включена, то все пользователи, даже у которых есть привилегия `PARSEC_CAP_SUMAC`, не смогут использовать команду `sumac`. Для этого устанавливаются права доступа `000` на исполняемый файл `sumac` и библиотеку `libsumacrunner.so`.

Изменение блокировки вступает в действие немедленно.

Описание инструмента приведено на справочной странице `man astra-sumac-lock`.

Управление блокировкой также может осуществляться с помощью модуля «Системные параметры» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_system_parameters
```

16.6.24. Управление межсетевым экраном ufw

Инструмент командной строки `astra-ufw-control` включает и выключает межсетевой экран ufw. Параметры вызова, используемые данным инструментом, приведены в таблице 62.

Если при включении межсетевого экрана ufw включен другой межсетевой экран (firewalld), то ufw не будет включен.

Изменение вступает в действие немедленно.

Описание инструмента приведено на справочной странице `man astra-ufw-control`.

Управление межсетевым экраном ufw также может осуществляться с помощью модуля «Системные параметры» в разделе «Ограничения программной среды» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_system_parameters
```

16.6.25. Блокировка доступа по протоколу SSH для root

Блокировка доступа по протоколу SSH для учетной записи `root` осуществляется с помощью инструмента командной строки `astra-rootloginssh-control`, который изменяет значение параметра `PermitRootLogin` в конфигурационном файле `/etc/ssh/sshd_config`.

Для управления блокировкой в системе должен быть установлен пакет `ssh`.

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

При использовании вызова `enable` параметру `PermitRootLogin` присваивается значение `no` — запрещен доступ по протоколу SSH для учетной записи `root`.

При использовании вызова `disable` параметру `PermitRootLogin` присваивается значение `prohibit-passwd` (значение по умолчанию) — доступ по протоколу SSH для учетной записи `root` разрешен только с использованием ключей.

Инструмент разрешает для `root` доступ по протоколу SSH только с использованием ключей.

Изменение вступает в действие немедленно.

Описание инструмента приведено на справочной странице `man astra-rootloginssh-control`.

Управление блокировкой также может осуществляться с помощью модуля «Системные параметры» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_system_parameters
```

16.6.26. Переключение уровней защищенности

Инструмент командной строки `astra-modeswitch` позволяет переключать уровни защищенности ОС и отображает текущий уровень. Уровень защищенности определяет перечень доступных для использования функций безопасности. В ОС реализованы следующие уровни защищенности:

- базовый;
- усиленный;
- максимальный.

Переключение уровня защищенности делает доступными/недоступными для включения функции безопасности соответствующего уровня защищенности.

Функции безопасности базового уровня защищенности доступны на всех уровнях защищенности.

Функции безопасности, доступные на усиленном и максимальном уровнях защищенности, приведены в таблице 63.

Таблица 63

Функция безопасности	Уровень защищенности		
	базовый	усиленный	максимальный
Замкнутая программная среда	Недоступна	Доступна	Доступна
Очистка памяти	Недоступна	Доступна, в т.ч. для ЗСУБД	Доступна, в т.ч. для ЗСУБД
Мандатный контроль целостности	Недоступна	Доступна	Доступна
Мандатное управление доступом	Недоступна	Недоступна	Доступна, в т.ч. для ЗСУБД

ВНИМАНИЕ! Функции безопасности, недоступные на текущем уровне защищенности, не могут быть включены.

ВНИМАНИЕ! После переключения уровня защищенности:

- если функция безопасности недоступна на данном уровне защищенности, то она автоматически отключается;
- если функция безопасности доступна на данном уровне защищенности, то ее состояние не меняется, но она может быть включена с использованием соответствующих инструментов.

Для включения функций безопасности используются следующие инструменты:

1) «Замкнутая программная среда» — управление осуществляется с помощью инструмента `astra-digsig-control`, описанного в 16.6.28;

2) «Очистка освобождаемой внешней памяти» — включает в себя две функции безопасности: безопасное удаление файлов и очистка разделов подкачки. Управление осуществляется с помощью инструментов `astra-secdel-control` и `astra-swapwiper-control`, описанных в 16.6.29 и 16.6.30 соответственно;

3) «Мандатный контроль целостности» — управление осуществляется с помощью инструмента `astra-mic-control`, описанного в 16.6.27;

4) «Мандатное управление доступом» — управление осуществляется с помощью инструмента `astra-mac-control`, описанного в 16.6.31.

Описание функций безопасности, доступных для каждого из уровней защищенности, приведено также в документе РУСБ.10015-01 31 01.

С инструментом командной строки `astra-modeswitch` возможно использовать только параметры, приведенные в таблице 64.

Таблица 64

Параметр	Описание
`list`	Вывести список доступных уровней защищенности. В выводе команды: 0 `base(orel)` — базовый уровень защищенности; 1 `advanced(voronezh)` — усиленный уровень защищенности; 2 `maximum(smolensk)` — максимальный уровень защищенности.
`get`	Вывести текущий уровень защищенности в числовом представлении
`getname`	Вывести текущий уровень защищенности в текстовом представлении
`set m`	Установить уровень защищенности, заданный параметром `m`. В качестве параметра `m` используется числовое или текстовое представление уровня защищенности

Смена уровня защищенности осуществляется после перезагрузки ОС.

16.6.27. Управление мандатным контролем целостности

Инструмент командной строки `astra-mic-control` включает и выключает МКЦ, описанный в 4.9, а также изменяет значение максимальной категории целостности системы. Управление МКЦ осуществляется путем изменения значения параметра командной строки ядра `parsec.max_ilev`.

Параметры вызова, используемые данным инструментом, приведены в таблице 62. При использовании вызова `enable` возможно указать необязательный параметр `-i <уровень>`,

задающий значение максимальной категории целостности (после установки ОС максимальная категория целостности равна 63).

После выключения МКЦ и перезагрузки метка целостности на объектах файловой системы сбрасывается на нулевые значения.

После включения МКЦ и перезагрузки на объектах файловой системы восстанавливаются принятые по умолчанию значения целостности (высокая категория целостности на каталогах `/dev`, `/proc`, `/run`, `/sys`).

Для применения изменений требуется перезагрузка.

Описание инструмента приведено на справочной странице `man astra-mic-control`.

16.6.28. Управление замкнутой программной средой

Инструмент командной строки `astra-digsig-control` позволяет включать и выключать ЗПС в исполняемых файлах из командной строки. Описание ЗПС приведено в 16.1.

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

Изменение состояния ЗПС выполняется после перезагрузки.

Описание инструмента приведено на справочной странице `man astra-digsig-control`.

16.6.29. Управление безопасным удалением файлов

Инструмент командной строки `astra-secdel-control` включает и выключает механизм безопасного удаления файлов в соответствии с 9.1 на разделах с файловыми системами ext2, ext3, ext4, XFS, указанных в `/etc/fstab`.

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

При включении механизма безопасного удаления файлов по умолчанию используется режим с параметром `secdel` в соответствии с 9.1.

Изменение режима работы вступает в действие после следующего монтирования ФС (в т.ч. после перезагрузки ОС).

Описание инструмента приведено на справочной странице `man astra-secdel-control`.

Управление режимом также может осуществляться с помощью модуля «Политика очистки памяти» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_policy_clean_memory
```

16.6.30. Управление очисткой разделов подкачки

Инструмент командной строки `astra-swapwiper-control` контролирует очистку разделов подкачки при завершении работы ОС в соответствии с 9.1.

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

При использовании команды-переключателя вносятся изменения в конфигурационный файл `/etc/parsec/swap_wiper.conf`.

Изменение вступает в действие немедленно.

Описание инструмента приведено на справочной странице `man astra-swapwiper-control`.

Управление блокировкой также может осуществляться с помощью модуля «Политика очистки памяти» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_policy_clean_memory
```

16.6.31. Включение и выключение мандатного управления доступом

Инструмент командной строки `astra-mac-control` включает и выключает мандатное управление доступом в соответствии с 4.8, изменяя значение параметра командной строки ядра `parsec.mac`.

Для применения изменений требуется перезагрузка ОС.

При выключении мандатного управления доступом инструмент отключает возможность работы программ на ненулевом уровне конфиденциальности.

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

ВНИМАНИЕ! Отключение возможности работы программ на ненулевом уровне конфиденциальности ограничивает доступ к файловым объектам, имеющим ненулевую классификационную метку. Доступ к таким объектам может быть получен администратором (если в ОС включен МКЦ, то администратором с высокой категорией целостности). Доступ к таким объектам также может быть получен при наличии неконтролируемого физического доступа к компьютеру и возможности использовать на нем нештатные средства. Необходимо принять дополнительные меры по защите информации ограниченного доступа.

Описание инструмента приведено на справочной странице `man astra-mac-control`.

16.6.32. Управление AstraMode и MacEnable

Инструмент командной строки `astra-mode-apps` включает и выключает режим AstraMode сервера Apache2, а также управляет состоянием параметра `MacEnable` сервера печати CUPS.

Описание режима AstraMode приведено в РУСБ.10015-01 95 01-1, описание параметра `MacEnable` приведено в таблице 53.

Параметры вызова, используемые инструментом `astra-mode-apps`, приведены в таблице 62.

При использовании команды-переключателя вносятся изменения в конфигурационные файлы `/etc/apache2/apache2.conf` и `/etc/cups/cupsd.conf`, изменяя значения параметров `AstraMode` и `MacEnable` соответственно.

Для применения изменений требуется перезапуск служб.

Описание инструмента приведено на справочной странице `man astra-mode-apps`.

Включение и выключение режима AstraMode и управление параметром `MacEnable` также можно осуществлять с помощью модуля «Мандатное управление доступом» в разделе «Управление доступом» графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку). Для вызова модуля можно использовать команду:

```
astra-systemsettings astra_kcm_mac
```

16.6.33. Выключение и включение аудита файлов и процессов

Подсистема аудита может оказывать существенное влияние на производительность ОС. При необходимости возможно отключить PARSEC-аудит файлов и процессов для оптимизации нагрузки на ОС, а также для повышения производительности в целом.

Инструмент командной строки `astra-audit-control` позволяет выключать и включать постоянные правила PARSEC-аудита, располагающиеся в файле `/etc/audit/rules.d/10-parsec.rules` (описание приведено в 6.1).

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

При выключении PARSEC-аудита файл `/etc/audit/rules.d/10-parsec.rules` удаляется из каталога `/etc/audit/rules.d/`, в результате чего данные правила аудита перестают выполняться. Значение в файлах `/parsecfs/disable-all-audit`, `/parsecfs/disable-denied-audit` и `/parsecfs/disable-non-mac-audit` изменяются на `1` (описание параметров приведено в 6.4.7).

При включении PARSEC-аудита файл с правилами `10-parsec.rules`, шаблон которого хранится в `/usr/lib/parsec/audit/rules.d/`, копируется в каталог `/etc/audit/rules.d/` и правила начинают выполняться. Значения в файлах `/parsecfs/disable-all-audit`, `/parsecfs/disable-denied-audit` и `/parsecfs/disable-non-mac-audit` изменяются на `0`.

Изменения вступают в действие немедленно.

Параметр вызова `is-enabled` выводит информацию о наличии файла `/etc/audit/rules.d/10-parsec.rules`, для которого возможны следующие состояния:

1) `ВКЛЮЧЕНО` — файл с заданными правилами аудита присутствует в каталоге;
2) `ВЫКЛЮЧЕНО` — файл с заданными правилами аудита отсутствует в каталоге.

16.6.34. Выключение и включение сетевого аудита

Сетевой аудит, как и подсистема аудита в целом, могут оказывать существенное влияние на производительность ОС. При необходимости возможно отключить сетевой PARSEC-аудит, что значительно уменьшит объем журналов регистрации сетевых событий.

Инструмент командной строки `astra-audit-network-control` позволяет выключать и включать постоянные правила сетевого PARSEC-аудита, располагающиеся в файле `/etc/audit/rules.d/10-parsec-nw.rules` (описание приведено в 6.1).

Параметры вызова, используемые данным инструментом, приведены в таблице 62.

При выключении сетевого PARSEC-аудита файл `/etc/audit/rules.d/10-parsec-nw.rules` удаляется из каталога `/etc/audit/rules.d/`, в результате чего правила аудита перестают выполняться.

При включении сетевого PARSEC-аудита файл с правилами `10-parsec-nw.rules`, шаблон которого хранится в `/usr/lib/parsec/audit/rules.d/`, копируется в каталог `/etc/audit/rules.d/` и правила начинают выполняться.

Изменения вступают в действие немедленно.

Параметр вызова `is-enabled` выводит информацию о наличии файла `/etc/audit/rules.d/10-parsec-nw.rules`, для которого возможны следующие состояния:

1) `ВКЛЮЧЕНО` — файл с заданными правилами аудита присутствует в каталоге;
2) `ВЫКЛЮЧЕНО` — файл с заданными правилами аудита отсутствует в каталоге.

16.7. Модуль безопасности Yama

Модуль безопасности Yama ограничивает возможности использования механизма `ptrace` для отладки (трассировки) процессов ОС.

Для активации отладки процессов используется системный вызов `ptrace`:

- для назначения родительского процесса отладчиком текущего процесса:

```
ptrace(PTRACE_TRACEME, 0, NULL, NULL)
```

- для назначения текущего процесса отладчиком процесса с идентификатором `pid`:

```
ptrace(PTRACE_ATTACH, pid, NULL, NULL)
```

Модуль Yama поддерживает четыре режима работы:

1) `0` — отладка всех процессов, запущенных от имени текущего пользователя;

2) `1` — отладка только дочерних процессов, за исключением случаев явного указания текущим процессом возможного процесса-отладчика через вызов `prctl` с параметром `PR_SET_PTRACER`, задав `pid` процесса-отладчика (когда любые процессы, связанные с конкретной службой, должны отлаживаться специально выделенным для этого процессом) или параметр `PR_SET_PTRACER_ANY` (для разрешения любому процессу отлаживать текущий процесс), например:

```
prctl(PR_SET_PTRACER, PR_SET_PTRACER_ANY, 0, 0, 0)
```

В случае использования `PTRACE_TRACEME` дополнительных ограничений не накладывается, для возможности выполнения `PTRACE_ATTACH` необходимо наличие у процесса-отладчика привилегии `CAP_SYS_PTRACE`;

3) `2` — возможна отладка любых процессов с использованием `PTRACE_TRACEME` или `PTRACE_ATTACH`, но только при наличии у процесса-отладчика привилегии `CAP_SYS_PTRACE`;

4) `3` — отладка процессов с использованием `ptrace` запрещена.

Узнать режим, в котором работает модуль Yama в текущий момент, можно с помощью команды:

```
sysctl kernel.yama.ptrace_scope
```

либо прочитав файл `/proc/sys/kernel/yama/ptrace_scope`:

```
cat /proc/sys/kernel/yama/ptrace_scope
```

Для изменения режима работы модуля Yama можно использовать инструмент `sysctl`, указав для параметра `kernel.yama.ptrace_scope` значение требуемого режима работы (от `0` до `3`):

```
sysctl kernel.yama.ptrace_scope=2
```

либо откорректировав соответствующим образом файл `/proc/sys/kernel/yama/ptrace_scope`:

```
echo 2 > /proc/sys/kernel/yama/ptrace_scope
```

ВНИМАНИЕ! Заданный режим работы модуля Yama сохраняется только до перезагрузки ОС (выключения компьютера). При запуске ОС модуль Yama всегда будет загружаться в режиме работы `1`.

Для изменения режима работы модуля Yama пользователь должен обладать привилегией `CAP_SYS_PTRACE`.

После перевода модуля Yama в режим работы `3` (полный запрет трассировки процессов), дальнейшее изменение режимов работы может быть выполнено только после перезагрузки ОС.

16.8. Модуль безопасности lockdown

Модуль безопасности lockdown устанавливает для привилегированного пользователя запрет на процедуры получения доступа и выполнения кода на уровне ядра ОС, которые в штатном режиме он может осуществить, например, с помощью подмены ядра ОС (используя системный вызов `kexec`) или через `/dev/kmem`, читая/записывая данные в память.

Модуль lockdown поддерживает два режима работы:

1) режим обеспечения целостности ядра ОС (режим `integrity`), включающий следующие ограничения:

а) `LOCKDOWN_KEXEC` — запрет выполнения системного вызова `kexec()`, обеспечивающего подмену текущего ядра ОС;

б) `LOCKDOWN_HIBERNATION` — запрет гибернации, предотвращающий возможность подмены ядра ОС при выходе из нее;

в) `LOCKDOWN_MODULE_SIGNATURE` — включение проверки подписей модулей ядра ОС;

г) `LOCKDOWN_DEV_MEM` — запрет чтения и записи в `/dev/mem`, `/dev/kmem`, `/dev/port`;

д) LOCKDOWN_PCI_ACCESS — блокировка оборудования, которое потенциально может генерировать прямую адресацию памяти (DMA);

е) LOCKDOWN_IOPORT — блокировка ioctl-вызывов (ioctl_console) KDADDIO, KDDELIO, KDENABIO и KDDISABIO для терминалов и виртуальных консолей;

ж) LOCKDOWN_EFI_TEST — запрет чтения /dev/efi_test;

з) LOCKDOWN_ACPI_TABLES — ограничение доступа к интерфейсам ACPI;

2) режим обеспечения конфиденциальности некоторых компонентов ядра ОС (режим confidentiality), включающий все ограничения режима integrity, а также:

а) LOCKDOWN_KCORE — запрет чтения /proc/kcore;

б) LOCKDOWN_KPROBES — ограничение доступа к отладочному режиму kprobes;

в) LOCKDOWN_BPF_READ — ограничение доступа к механизму фильтрации пакетов BPF;

г) LOCKDOWN_TRACEFS — ограничение доступа к ФС tracefs.

Состояние модуля lockdown задается в файле /sys/kernel/security/lockdown, который по умолчанию имеет следующий вид:

```
[none] integrity confidentiality
```

В данном файле приведены возможные состояния модуля lockdown, а его текущее состояние отображается в квадратных скобках ([none] — модуль отключен).

После загрузки ОС модуль lockdown по умолчанию отключен.

Активировать модуль lockdown в нужном режиме работы можно в процессе функционирования ОС путем записи соответствующего режима в файл /sys/kernel/security/lockdown командой:

```
sudo echo <режим_модуля> > /sys/kernel/security/lockdown
```

где <режим_модуля> — integrity (для активации режима integrity) или confidentiality (для активации режима confidentiality), при этом:

- если модуль был отключен, то возможно активировать любой режим;
- если модуль работал в режиме integrity, то возможно только активировать режим confidentiality;
- если модуль работал в режиме confidentiality, то изменить режим или отключить модуль невозможно.

Изменения в режиме работы модуля lockdown сохраняются до перезагрузки ОС.

17. ГЕНЕРАЦИЯ КСЗ

Генерация КСЗ подразумевает настройку КСЗ из состава ОС в соответствии с требованиями безопасности, предъявляемыми к рабочему месту.

17.1. Настройка КСЗ

Генерация КСЗ осуществляется в одном из следующих режимов:

- режим ЕПП;
- локальный режим.

Для использования режима ЕПП необходимо наличие в сети установленного и настроенного сервера FreeIPA. В ОС на рабочих местах пользователей должны быть установлены клиентские пакеты FreeIPA и выполнены действия по настройке (см. документ РУСБ.10015-01 95 01-1).

Для генерации КСЗ необходимо выполнить следующие действия:

1) проверить, что в ОС присутствует необходимый набор уровней и категорий конфиденциальности. При необходимости создать или отредактировать;

2) создать служебные учетные записи, наличие которых необходимо для функционирования защищенных комплексов программ СУБД, гипертекстовой обработки данных, электронной почты и программ маркировки и учета печатных документов из состава ОС;

3) установить для служебных учетных записей необходимые привилегии;

4) установить необходимые параметры аудита (регистрации событий) в ОС.

Примечание. В информационных системах с мандатным управлением доступом как правило применяются классификационные метки, в которых используется только четыре уровня конфиденциальности (от 0 до 3) и 64-битовая маска с различными сочетаниями категорий конфиденциальности.

Генерация КСЗ выполняется:

1) в локальном режиме — при помощи графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку) или при помощи соответствующих инструментов, описание которых приведено в настоящем документе, от имени учетной записи администратора через механизм `sudo`;

2) в режиме ЕПП с развернутым доменом FreeIPA — при помощи веб-интерфейса FreeIPA или инструмента командной строки `ipa user-mod` (подробная информация

по использованию инструмента командной строки доступна на справочной странице `ipa help user-mod`).

ВНИМАНИЕ! Для изменения максимальной метки безопасности в системе и объектов ФС (см. 4.4) необходимо внести изменения в сценарий `pdp-init-fs` (см. 4.15.3), переопределив значения переменных:

- `sysmaxlev` — максимальный уровень конфиденциальности;
- `sysmaxilev` — максимальная категория целостности;
- `sysmaxcat` — максимальный набор категорий конфиденциальности.

После завершения генерации КСЗ осуществляется создание учетных записей пользователей, установка для них разрешенных уровней и категорий конфиденциальности и др.

ВНИМАНИЕ! Устанавливать для пользователей одновременно высокий уровень конфиденциальности (классификационную метку) и высокую метку целостности не рекомендуется.

Создание и управление учетными записями пользователей, а также другие настройки могут выполняться:

1) в локальном режиме — при помощи графической утилиты `astra-systemsettings` или при помощи соответствующих инструментов, описание которых приведено в настоящем документе, от имени учетной записи администратора через механизм `sudo`;

2) в режиме ЕПП с развернутым доменом FreeIPA — при помощи web-интерфейса FreeIPA или инструмента командной строки `ipa user-mod` (подробная информация по использованию инструмента командной строки доступна на справочной странице `ipa help user-mod`).

Дополнительно в соответствии с требованиями о защите информации следует настроить необходимые подсистемы и функции безопасности из состава ОС, обеспечивающие безопасную эксплуатацию ОС (см. 17.2).

17.2. Безопасная настройка ОС

Безопасная настройка ОС осуществляется в соответствии с методическими рекомендациями по безопасной настройке операционной системы специального назначения «Astra Linux Special Edition», приведенными на официальном ресурсе разработчика по ссылке `wiki.astralinux.ru` (раздел «Рекомендации по безопасной установке и эксплуатации Astra Linux»).

Для безопасной настройки ОС также могут использоваться входящие в состав ОС профили системы (см. 17.3).

Безопасная настройка ОС обеспечивает реализацию мер по защите информации от угроз, приведенных в 17.5.

17.3. Профили настройки КСЗ

Для настройки КСЗ в соответствии с требованиями о защите информации, предъявляемыми к определенному классу информационных систем, в состав ОС входят профили, выполняющие настройку КСЗ соответствующим образом. Выбрать и применить соответствующий профиль системы можно в разделе «Профили системы» графической утилиты `astra-systemsettings`.

При необходимости возможно создать новый профиль системы. Новый профиль создается путем экспорта текущих настроек КСЗ. Для создания профиля необходимо:

1) в каталоге `/etc/astra-system-profiles/profiles` создать файл вида `<имя_профиля>.desktop`;

2) в созданный файл добавить запись следующего вида (указанная информация будет отображаться в разделе «Профили системы» графической утилиты `astra-systemsettings`):

```
[Desktop Entry]
Name=<Имя профиля на английском языке>
Name[ru]=<Имя профиля на русском языке>

Comment=<Описание профиля на английском языке>
Comment[ru]=<Описание профиля на русском языке>
```

3) создать каталог для размещения конфигурационных файлов профиля вида `/etc/astra-system-profiles/profiles_data/<имя_профиля>`.

ВНИМАНИЕ! Имя каталога должно полностью совпадать с именем ранее созданного файла;

4) экспортировать текущие настройки КСЗ в созданный профиль с использованием инструмента командной строки `astra-settings-export`:

```
astra-settings-export /etc/astra-system-profiles/profiles_data/
<имя_профиля>
```

17.4. Импорт и экспорт настроек КСЗ

Текущие настройки КСЗ могут быть экспортированы для их применения на другом рабочем месте или для создания резервной копии.

Экспорт настроек КСЗ в каталог или архив осуществляется с использованием инструмента командной строки `astra-settings-export` из пакета `astra-settings-exchange` командой:

```
astra-settings-export <путь_к_каталогу_или_архиву> security_settings
```

ВНИМАНИЕ! При экспорте в каталог необходимо указывать существующий каталог.

В указанный каталог или архив будет добавлен каталог `security_settings`, содержащий конфигурационные файлы с параметрами настройки КСЗ и их значениями. Например, конфигурационный файл `security_settings/ase_mac_settings.ini`, содержащий настройки мандатного управления доступом, может иметь следующий вид:

```
[General]
Apache=@Invalid()
Categories="{\"Категория_1\":0,\"Категория_2\":1}"
Cups=true
Levels="{\"Уровень_0\":0,\"Уровень_1\":1,\"Уровень_2\":2,\"Уровень_3\":3}"
Mac-kernel=true
```

Полное описание инструмента командной строки `astra-settings-export` приведено на странице помощи:

```
astra-settings-export -h
```

Восстановить настройки КСЗ из резервной копии или применить экспортированные с другого рабочего места возможно с использованием инструмента командной строки `astra-settings-import` из пакета `astra-settings-exchange`. Восстановление выполняется из каталога или архива, в который были экспортированы настройки ранее, для этого выполнить команду:

```
astra-settings-import <путь_к_каталогу_или_архиву> security_settings
```

ВНИМАНИЕ! Указанный каталог или архив должен содержать каталог `security_settings` с конфигурационными файлами.

Полное описание инструмента командной строки `astra-settings-import` приведено на странице помощи:

```
astra-settings-import -h
```

В состав пакета `astra-settings-exchange` также входят следующие инструменты командной строки:

1) `astra-settings-merge` — применяется только для экспорта настроек в архив, объединяет текущие настройки КСЗ с уже экспортированными в указанный архив. При этом существующие в архиве настройки КСЗ не будут перезаписаны, а будут только добавлены отсутствующие. Полное описание приведено на странице помощи:

```
astra-settings-merge -h
```

2) `astra-settings-remove` — удаляет все настройки из указанного каталога или архива. Полное описание приведено на странице помощи:

```
astra-settings-remove -h
```

Инструменты командной строки из пакета `astra-settings-exchange` позволяют импортировать, экспортировать или удалить из указанного каталога или архива не все настройки, а только отдельные конфигурационные файлы или параметры (строки в конфигурационном файле). Для этого в командах необходимо указать соответствующие параметры. Описание параметров, применяемых с инструментами, приведено на страницах помощи соответствующих инструментов.

17.5. Возможности по защите от угроз безопасности информации средствами защиты ОС

В состав ОС входят средства, обеспечивающие защиту информации от различных угроз безопасности. Средства защиты, применяемые для исключения угроз безопасности информации, приведены в 17.5.1–17.5.13.

17.5.1. Угрозы, связанные с внедрением вредоносного кода и повышением привилегий

17.5.1.1. ЗПС обеспечивает запрет автоматического или ручного запуска стороннего, не обладающего корректной цифровой подписью программного обеспечения (исполняемых модулей или библиотек) в соответствии с 16.1.

17.5.1.2. МКЦ обеспечивает защиту системных компонентов (процессов, файлов) ОС, параметров функционирования СЗИ от деструктивного воздействия со стороны недоверенных пользователей. Процессы, а также конфигурационные файлы СЗИ и системного программного обеспечения обладают высокой меткой целостности, что исключает возможность оказания несанкционированного воздействия на них даже в случае получения нарушителем доступа к системе с правами `root` с низкой меткой целостности (см. 4.3).

17.5.1.3. Средства контейнеризации обеспечивают запуск программного обеспечения, включая веб-браузеры, в контейнерах от имени непривилегированных пользователей (`rootless`-режим), что обеспечивает защиту системных и пользовательских данных и процессов в

случае реализации угроз безопасности информации внутри изолированного контейнера. При этом средствами контейнеризации обеспечивается блокирование запуска контейнеров при нарушении целостности образов и/или конфигурации, а также при обнаружении уязвимостей в них. В случае эксплуатации контейнеров в привилегированном режиме и при включенном МКЦ может быть использован механизм запуска контейнеров с пониженной категорией целостности для защиты от реализации нарушителем угроз, связанных с преодолением изоляции контейнера, и минимизации их последствий (см. раздел 8).

17.5.1.4. Киоск-2 обеспечивает возможность реализации ограничений, связанных с разрешениями запуска программного обеспечения и получением доступа к объектам ФС, для отдельных пользователей путем настройки профилей пользователей (системных профилей), что обеспечивает невозможность несанкционированного запуска произвольного кода и доступа к системным компонентам ОС (см. 16.2).

17.5.1.5. Блокировка интерпретатора `bash`, а также иных интерпретаторов (`python`, `perl`, `expect`, `ruby`, `dash`, `php`, `irb`, `csh`, `lua`, `ksh`, `tcl`, `tk`, `zsh`, `nodejs` и др.) совместно с механизмом запрета установки бита исполнения (см. 16.6.4) обеспечивает запрет несанкционированного использования интерпретатора(ов) недоверенными пользователями для выполнения кода напрямую из командной строки, неименованного канала (`pipe`) и/или запуска сторонних сценариев (см. 16.6.7).

17.5.1.6. Блокировка использования макросов в стандартных приложениях, включая пакеты офисного программного обеспечения предотвращает несанкционированный запуск программного кода при их использовании (см. 16.6.8).

17.5.1.7. Средства регламентного контроля целостности обеспечивают периодическую (по запросу или расписанию) проверку целостности программного обеспечения на предмет несанкционированного изменения (в том числе за счет внедрения вредоносного программного кода) в соответствии с 10.4.

17.5.1.8. Регистрация событий безопасности обеспечивает гибкую настройку регистрируемых событий, их автоматизированную обработку и визуализацию. Подсистема позволяет контролировать события безопасности информации, связанные как со штатными подсистемами управления доступом операционных систем семейства Linux, так и с дополнительными механизмами защиты ОС, что обеспечивает возможность своевременного реагирования на возникающие инциденты информационной безопасности, в том числе связанные с попытками обхода СЗИ (см. раздел 6).

17.5.1.9. Модуль безопасности yama ограничивает использование механизмов отладки (трассировки) процессов ОС, что исключает возможность инжектирования за счет этого механизма недоверенного кода в адресное пространство функционирующих процессов (см. 16.7).

17.5.1.10. Модуль безопасности lockdown обеспечивает запрет получения несанкционированного доступа и выполнения кода на уровне ядра ОС, в том числе со стороны привилегированного пользователя (см. 16.8).

17.5.1.11. Ядро ОС обеспечивает запрет записи в область памяти, помеченную как исполняемая; запрет прямого создания исполняемых областей памяти; запрет создания исполняемого стека; а также рандомизацию адресного пространства ядра и процессов (см. 9.2).

17.5.1.12. Модуль ядра lkrg обеспечивает обнаружение некоторых уязвимостей и защиту пространства памяти ядра от модификации (см. 16.6.15).

17.5.2. Угрозы виртуальной инфраструктуры

17.5.2.1. МКЦ обеспечивает защиту системных компонентов (процессов, файлов) ОС, параметров функционирования СЗИ от деструктивного воздействия со стороны недоверенных пользователей. Процессы, а также конфигурационные файлы СЗИ и системного программного обеспечения обладают высокой меткой целостности, что исключает возможность оказания несанкционированного воздействия на них даже в случае получения нарушителем доступа к системе с правами `root` с низкой меткой целостности (см. 4.3).

17.5.2.2. Мандатное управление доступом позволяет категорировать защищаемую информацию (данные) по уровням и категориям конфиденциальности, что обеспечивает дополнительную защиту информации от несанкционированного доступа, в том числе в среде виртуализации (запуск ВМ с заданным уровнем конфиденциальности) в соответствии с 4.2.

17.5.2.3. ЗПС предоставляет возможность осуществления динамического контроля целостности файлов конфигурации, а также исполняемых файлов и загружаемых библиотек при запуске и в процессе инициализации и функционирования среды виртуализации (см. 16.1).

17.5.2.4. Механизм доверенной загрузки ВМ обеспечивает контроль целостности средствами ОС файлов образов, конфигураций и базовой системы ввода-вывода ВМ (см. 5.6).

17.5.2.5. Механизм контроля целостности областей памяти ВМ обеспечивает постановку на контроль средствами среды виртуализации и хостовой ОС областей памяти виртуальных машин по запросу гостевой ОС (см. 5.12).

17.5.2.6. Режим функционирования ВМ «только чтение» обеспечивает защиту от несанкционированного раскрытия информации, а также модификации системных компонентов образов ВМ в процессе их функционирования (см. 5.4).

17.5.2.7. Ролевое управления доступом в среде виртуализации обеспечивает дополнительные возможности разграничения полномочий по управлению средой виртуализации и ВМ, в частности действий, связанных с инициализацией ВМ (создание, запуск, изменение конфигурации и т. п.) в соответствии с 5.3.

17.5.2.8. Механизм изоляции адресных пространств ВМ поддерживается средствами ядра ОС с применением механизмов управления памятью аппаратной платформы, предоставляемых модулем ядра KVM, что обеспечивает защиту от несанкционированного доступа в адресные пространства процессов других ВМ. В соответствии с описанием раздела 7.

17.5.2.9. Модуль безопасности yama ограничивает использование механизмов отладки (трассировки) процессов ОС, что исключает возможность инжектирования за счет этого механизма недоверенного кода в адресное пространство функционирующих процессов (см. 16.7).

17.5.2.10. Модуль безопасности lockdown обеспечивает запрет получения несанкционированного доступа и выполнения кода на уровне ядра ОС, в том числе со стороны привилегированного пользователя (см. 16.8).

17.5.2.11. Механизм очистки памяти предотвращает несанкционированный доступ к остаточной информации, в том числе содержащей чувствительные данные гостевых операционных систем (см. 9.1).

17.5.3. Угрозы несанкционированного изменения конфигурации системы и средств защиты информации

17.5.3.1. МКЦ обеспечивает защиту от несанкционированного изменения конфигурации программного обеспечения и параметров функционирования средств защиты информации за счет назначения соответствующим объектам (файлам) высокой метки целостности. Соответствующие уровни целостности присваиваются по умолчанию при включении МКЦ, назначаются в дальнейшем динамически в соответствии со строгими правилами подсистемы управления доступом. Изменение уровней целостности, при необходимости, возможно только от имени привилегированного пользователя, обладающего высокой меткой целостности, что исключает несанкционированное воздействие со стороны непривилегированных пользователей, в том числе суперпользователя `root` с низкой меткой целостности (см. 4.3).

17.5.3.2. ЗПС предоставляет возможность осуществления динамического контроля целостности файлов конфигурации, а также исполняемых файлов и загружаемых библиотек при запуске и в процессе функционирования системного и прикладного программного обеспечения (см. 16.1).

17.5.3.3. Средства регламентного контроля целостности обеспечивают периодическую (по запросу или расписанию) проверку целостности системного и программного обеспечения, а также конфигурационных файлов компонентов ОС на предмет несанкционированного изменения (см. 10.4).

17.5.3.4. Киоск-2 обеспечивает возможность реализации ограничений, связанных с разрешениями запуска программного обеспечения и получением доступа к объектам ФС, для отдельных пользователей путем настройки профилей пользователей (системных профи-

лей), что обеспечивает невозможность несанкционированного запуска произвольного кода и доступа к системным компонентам ОС (см. 16.2).

17.5.3.5. Мониторинг событий безопасности информации обеспечивает гибкую настройку регистрируемых событий, их автоматизированную обработку и визуализацию. Подсистема позволяет контролировать события безопасности информации, связанные как со штатными подсистемами управления доступом операционных систем семейства Linux, так и с дополнительными механизмами защиты ОС, что обеспечивает возможность своевременного реагирования на возникающие инциденты информационной безопасности, в том числе связанные с попытками обхода СЗИ (см. раздел 6).

17.5.3.6. Модуль безопасности lockdown в том числе обеспечивает запрет задания потенциально опасных параметров загрузки ядра ОС, а также выполнения ряда небезопасных системных вызовов и прямого доступа на запись (через специальные файлы устройств) к пространству ядра ОС со стороны привилегированного пользователя (см. 16.8).

17.5.3.7. Отключение отображения меню загрузчика обеспечивает невозможность влияния на процесс инициализации системы со стороны недоверенного пользователя (см. 16.6.17).

17.5.4. Угрозы несанкционированного доступа к информации

17.5.4.1. МКЦ обеспечивает защиту от несанкционированного изменения конфигурации программного обеспечения и параметров функционирования средств защиты информации за счет назначения соответствующим объектам (файлам) высокой метки целостности, что исключает несанкционированное воздействия со стороны непривилегированных пользователей, а также суперпользователя `root` с низкой меткой целостности (см. 4.3).

17.5.4.2. Мандатное управление доступом позволяет категорировать защищаемую информацию (данные) по уровням и категориям конфиденциальности, обеспечивая строгие правила доступа к ней (как на чтение, так и на запись), в том числе суперпользователю `root` с низкой меткой целостности (см. 4.2).

17.5.4.3. ЗПС предоставляет возможность осуществления контроля целостности программного обеспечения и средств защиты информации как в процессе загрузки, так и динамически в процессе исполнения (см. 16.1).

17.5.4.4. Киоск-2 обеспечивает возможность реализации ограничений (в дополнение к дискреционному управлению доступом), связанных с разрешениями запуска программного обеспечения и получением доступа к объектам ФС, для отдельных пользователей путем настройки профилей пользователей (системных профилей), что обеспечивает невозможность несанкционированного запуска произвольного кода и доступа к защищаемой информации (см. 16.2).

17.5.4.5. Блокировка интерпретатора `bash`, а также иных интерпретаторов (`python`, `perl`, `expect`, `ruby`, `dash`, `php`, `irb`, `csh`, `lua`, `ksh`, `tcl`, `tk`, `zsh`, `nodejs` и др.) совместно с

механизмом запрета установки бита исполнения (см. 16.6.4) обеспечивает запрет несанкционированного использования интерпретатора(ов) недоверенными пользователями для выполнения кода напрямую из командной строки, неименованного канала (`pipe`) и/или запуска сторонних сценариев (см. 16.6.7).

17.5.4.6. Блокировка использования макросов в стандартных приложениях, включая пакеты офисного программного обеспечения предотвращает несанкционированный запуск программного кода при их использовании (см. 16.6.8).

17.5.4.7. Средства контейнеризации обеспечивают запуск программного обеспечения, включая веб-браузеры, в контейнерах от имени непривилегированных пользователей (`rootless`-режим), что обеспечивает защиту системных и пользовательских данных и процессов в случае реализации угроз безопасности информации внутри изолированного контейнера. При этом средствами контейнеризации обеспечивается блокирование запуска контейнеров при нарушении целостности образов и/или конфигурации, а также при обнаружении уязвимостей в них. В случае эксплуатации контейнеров в привилегированном режиме и при включенном МКЦ может быть использован механизм запуска контейнеров с пониженной категорией целостности для защиты от реализации нарушителем угроз, связанных с преодолением изоляции контейнера, и минимизации их последствий (см. раздел 8).

17.5.4.8. Средства виртуализации обеспечивают запуск виртуальных машин с применением аппаратной виртуализации, позволяющей осуществлять изоляцию адресных пространств виртуальных машин и задание ограничений на потребляемые ими ресурсы (см. документ РУСБ.10015-01 95 01-1).

17.5.5. Угрозы несанкционированного использования ресурсов системы

17.5.5.1. МКЦ обеспечивает защиту от несанкционированного изменения конфигурации программного обеспечения и параметров функционирования средств защиты информации за счет назначения соответствующим объектам (файлам) высокой метки целостности, что исключает несанкционированное воздействия со стороны непривилегированных пользователей, а также суперпользователя `root` с низкой меткой целостности (см. 4.3).

17.5.5.2. Мандатное управление доступом обеспечивает защиту от выполнения функций в контексте процессов с низким уровнем конфиденциальности при попытке доступа к ним процессов с более высоким уровнем конфиденциальности через шину D-Bus (см. 4.2).

17.5.5.3. ЗПС предоставляет возможность осуществления контроля целостности программного обеспечения и средств защиты информации как в процессе загрузки, так и динамически в процессе исполнения (см. 16.1).

17.5.5.4. Киоск-2 обеспечивает возможность реализации ограничений (в дополнение к дискреционному управлению доступом), связанных с разрешениями запуска программного обеспечения и получением доступа к объектам ФС, для отдельных пользователей путем на-

стройки профилей пользователей (системных профилей), что обеспечивает невозможность несанкционированного запуска произвольного кода и доступа к защищаемой информации (см. 16.2).

17.5.5.5. Блокировка интерпретатора `bash`, а также иных интерпретаторов (`python`, `perl`, `expect`, `ruby`, `dash`, `php`, `irb`, `csh`, `lua`, `ksh`, `tcl`, `tk`, `zsh`, `nodejs` и др.) совместно с механизмом запрета установки бита исполнения (см. 16.6.4) обеспечивает запрет несанкционированного использования интерпретатора(ов) недоверенными пользователями для выполнения кода напрямую из командной строки, неименованного канала (`pipe`) и/или запуска сторонних сценариев (см. 16.6.7).

17.5.5.6. Блокировка использования макросов в стандартных приложениях, включая пакеты офисного программного обеспечения предотвращает несанкционированный запуск программного кода при их использовании (см. 16.6.8).

17.5.5.7. Средства контейнеризации обеспечивают запуск программного обеспечения, включая веб-браузеры, в контейнерах от имени непривилегированных пользователей (`rootless`-режим), что обеспечивает защиту системных и пользовательских данных и процессов в случае реализации угроз безопасности информации внутри изолированного контейнера. При этом средствами контейнеризации обеспечивается блокирование запуска контейнеров при нарушении целостности образов и/или конфигурации, а также при обнаружении уязвимостей в них. В случае эксплуатации контейнеров в привилегированном режиме и при включенном МКЦ может быть использован механизм запуска контейнеров с пониженной категорией целостности для защиты от реализации нарушителем угроз, связанных с преодолением изоляции контейнера, и минимизации их последствий (см. раздел 8).

17.5.5.8. Средства виртуализации обеспечивают запуск виртуальных машин с применением аппаратной виртуализации, позволяющей осуществлять изоляцию адресных пространств виртуальных машин и задание ограничений на потребляемые ими ресурсы (см. документ РУСБ.10015-01 95 01-1).

17.5.5.9. Механизм установки квот на использование системных ресурсов реализует ограничения на использование некоторых ресурсов системы для предотвращения нарушений доступности системы в результате исчерпания ресурсов (см. 16.6.3).

17.5.6. Угрозы процедур идентификации/аутентификации

17.5.6.1. МКЦ обеспечивает защиту от несанкционированного изменения конфигурации программного обеспечения и параметров функционирования средств защиты информации (в том числе обеспечивающих идентификацию и аутентификацию пользователей) за счет назначения соответствующим объектам (файлам) высокой метки целостности. Соответствующие метки целостности присваиваются по умолчанию при включении МКЦ, назначаются в дальнейшем динамически в соответствии с правилами подсистемы управления доступом. Изменение меток целостности, при необходимости, возможно только от имени привилегиро-

ванного пользователя, обладающего высокой меткой целостности, что исключает несанкционированное воздействие со стороны непривилегированных пользователей, в том числе суперпользователя `root` с низкой меткой целостности (см. 4.3).

17.5.6.2. Мандатное управление доступом позволяет категорировать защищаемую информацию (данные) по уровням и категориям конфиденциальности, что обеспечивает дополнительные возможности разграничения доступа, в том числе в случае использования уязвимых криптографических алгоритмов — нарушителю для получения доступа к защищаемой информации необходимо обладать соответствующим мандатным уровнем доступа (см. 4.2).

17.5.6.3. ЗПС предоставляет возможность осуществления динамического контроля целостности файлов конфигурации, а также исполняемых файлов и загружаемых библиотек при запуске и в процессе функционирования системного и прикладного программного обеспечения (в том числе участвующего в обеспечении идентификации и аутентификации пользователей) в соответствии с 16.1.

17.5.6.4. Контроль подключения съемных машинных носителей информации (см. раздел 14) и сопоставление пользователя с устройством (см. раздел 15) позволяют регистрировать съемные носители и устройства, минимизация возможности заменить или подменить устройство (в случае использования двухфакторной аутентификации).

17.5.6.5. Средства регламентного контроля целостности обеспечивают целостность конфигурационных файлов, определяющих порядок работы PAM-модулей, файла, содержащего перечень всех локальных пользователей ОС, а также других файлов, участвующих в процедуре аутентификации (см. 10.4).

17.5.6.6. Мониторинг событий безопасности информации обеспечивает гибкую настройку регистрируемых событий, их автоматизированную обработку и визуализацию. Подсистема позволяет контролировать события безопасности информации, связанные как со штатными подсистемами управления доступом операционных систем семейства Linux, так и с дополнительными механизмами защиты ОС, что обеспечивает возможность своевременного реагирования на возникающие инциденты информационной безопасности, в том числе связанные с попытками обхода СЗИ (см. раздел 6).

17.5.7. Угрозы сетевой инфраструктуры и веб-технологий

17.5.7.1. МКЦ обеспечивает защиту от несанкционированного изменения конфигурации программного обеспечения и параметров функционирования средств защиты информации, а также глобальных переменных за счет назначения им высокой метки целостности, что исключает несанкционированное воздействия даже со стороны суперпользователя `root` (см. 4.3).

17.5.7.2. Мандатное управление доступом позволяет категорировать защищаемую информацию (данные) по уровням и категориям конфиденциальности, обеспечивая правила доступа к ней (как на чтение, так и на запись) в том числе на сетевом уровне (см. 4.2).

17.5.7.3. ЗПС предоставляет возможность осуществления контроля целостности программного обеспечения и средств защиты информации как в процессе загрузки, так и динамически в процессе исполнения (см. 16.1).

17.5.7.4. Блокировка интерпретатора `bash`, а также иных интерпретаторов (`python`, `perl`, `expect`, `ruby`, `dash`, `php`, `irb`, `csh`, `lua`, `ksh`, `tcl`, `tk`, `zsh`, `nodejs` и др.) совместно с механизмом запрета установки бита исполнения (см. 16.6.4) обеспечивает запрет несанкционированного использования интерпретатора(ов) недоверенными пользователями для выполнения кода напрямую из командной строки, неименованного канала (`pipe`) и/или запуска сторонних сценариев (см. 16.6.7).

17.5.7.5. Блокировка использования макросов в стандартных приложениях, включая пакеты офисного программного обеспечения предотвращает несанкционированный запуск программного кода при их использовании (см. 16.6.8).

17.5.7.6. Средства контейнеризации обеспечивают запуск программного обеспечения, включая веб-браузеры, в контейнерах от имени непривилегированных пользователей (`rootless`-режим), что обеспечивает защиту системных и пользовательских данных и процессов в случае реализации угроз безопасности информации внутри изолированного контейнера. При этом средствами контейнеризации обеспечивается блокирование запуска контейнеров при нарушении целостности образов и/или конфигурации, а также при обнаружении уязвимостей в них. В случае эксплуатации контейнеров в привилегированном режиме и при включенном МКЦ может быть использован механизм запуска контейнеров с пониженной категорией целостности для защиты от реализации нарушителем угроз, связанных с преодолением изоляции контейнера, и минимизации их последствий (см. раздел 8).

17.5.7.7. Блокировка автоматического конфигурирования сетевых подключений блокирует автоматическое конфигурирование сетевых подключений путем блокировки работы служб `network-manager` и `connman` (см. 16.6.16).

17.5.8. Угрозы раскрытия информации

17.5.8.1. ЗПС предоставляет возможность осуществления контроля целостности программного обеспечения и средств защиты информации как в процессе загрузки, так и динамически в процессе исполнения (см. 16.1).

17.5.8.2. МКЦ обеспечивает защиту системных компонентов (процессов, файлов) ОС, параметров функционирования СЗИ от деструктивного воздействия со стороны недоверенных пользователей. Процессы, а также конфигурационные файлы СЗИ и системного программного обеспечения обладают высокой меткой целостности, что исключает возможность оказания

несанкционированного воздействия на них даже в случае получения нарушителем доступа к системе с правами `root` с низкой меткой целостности (см. 4.3).

17.5.8.3. Киоск-2 обеспечивает возможность реализации ограничений, связанных с разрешениями запуска программного обеспечения и получением доступа к объектам ФС, для отдельных пользователей путем настройки профилей пользователей (системных профилей), что обеспечивает невозможность несанкционированного запуска произвольного кода и доступа к системным компонентам ОС (см. 16.2).

17.5.8.4. Механизм очистки освобождаемой внешней памяти (включая файлы подкачки) обеспечивает затирание (в том числе путем многократной перезаписи данных псевдослучайной последовательностью байтов) освобождаемых блоков ФС непосредственно при их освобождении и/или по запросу пользователя (см. 9.1).

17.5.8.5. Механизм очистки памяти (в ядерном стеке (STACKLEAK), в ядерной куче (PAGE_POISONING)) обеспечивается ядром ОС и предотвращает несанкционированный доступ к остаточной информации, в том числе содержащей чувствительные данные (см. 9.1).

17.5.8.6. Модуль безопасности lockdown обеспечивает запрет получения несанкционированного доступа к памяти и выполнения кода на уровне ядра ОС, в том числе со стороны привилегированного пользователя (суперпользователя `root`) в соответствии с 16.8.

17.5.8.7. Модуль безопасности yama ограничивает использование механизмов отладки (трассировки) процессов ОС, что исключает возможность получения за счет привилегии отладки несанкционированного доступа к адресному пространству других процессов (см. 16.7).

17.5.9. Угрозы целостности данных

17.5.9.1. ЗПС предоставляет возможность осуществления контроля целостности программного обеспечения и средств защиты информации как в процессе загрузки, так и динамически в процессе исполнения (см. 16.1).

17.5.9.2. МКЦ обеспечивает защиту от несанкционированного изменения конфигурации программного обеспечения и параметров функционирования средств защиты информации (в том числе обеспечивающих идентификацию и аутентификацию пользователей) за счет назначения соответствующим объектам (файлам) высокой метки целостности. Соответствующие уровни целостности присваиваются по умолчанию при включении МКЦ, назначаются в дальнейшем динамически в соответствии со строгими правилами подсистемы управления доступом. Изменение уровней целостности, при необходимости, возможно только от имени привилегированного пользователя, обладающего высокой меткой целостности, что исключает несанкционированное воздействие со стороны непривилегированных пользователей, в том числе суперпользователя `root` с низкой меткой целостности (см. 4.3).

17.5.9.3. Мандатное управление доступом позволяет категорировать защищаемую информацию (данные) по уровням и категориям конфиденциальности, обеспечивая строгие правила

доступа к ней (как на чтение, так и на запись), в том числе суперпользователю `root` с низкой меткой целостности (см. 4.2).

17.5.9.4. Киоск-2 обеспечивает возможность реализации ограничений, связанных с разрешениями запуска программного обеспечения и получением доступа к объектам ФС, для отдельных пользователей путем настройки профилей пользователей (системных профилей), что обеспечивает невозможность несанкционированного запуска произвольного кода и доступа к системным компонентам ОС (см. 16.2).

17.5.9.5. Мониторинг событий безопасности информации обеспечивает гибкую настройку регистрируемых событий, их автоматизированную обработку и визуализацию. Подсистема позволяет контролировать события безопасности информации, связанные как со штатными подсистемами управления доступом операционных систем семейства Linux, так и с дополнительными механизмами защиты ОС, что обеспечивает возможность своевременного реагирования на возникающие инциденты информационной безопасности, в том числе связанные с попытками обхода СЗИ (см. раздел 6).

17.5.9.6. Контроль целостности программного обеспечения и средств защиты информации обеспечивается применением комплекса средств регламентного и периодического контроля целостности, включая:

- средство подсчета контрольных сумм файлов и оптических дисков (10.1);
- средство подсчета контрольных сумм файлов в deb-пакетах (10.2);
- средство контроля соответствия дистрибутиву (10.3);
- средства регламентного контроля целостности (10.4);
- средства создания и проверки ЭП (10.5);
- средства создания замкнутой программной среды (16.1).

17.5.9.7. Модуль безопасности lockdown обеспечивает запрет получения несанкционированного доступа (чтение/запись) к памяти и выполнения кода на уровне ядра ОС, в том числе со стороны привилегированного пользователя (суперпользователя `root`) в соответствии с 16.8.

17.5.9.8. Модуль безопасности yama ограничивает использование механизмов отладки (трассировки) процессов ОС, что исключает возможность получения за счет привилегии отладки несанкционированного доступа к адресному пространству других процессов (см. 16.7).

17.5.9.9. Модуль ядра lkrg обеспечивает обнаружение некоторых уязвимостей и защиту пространства памяти ядра от модификации (см. 16.6.15).

17.5.10. Угрозы некорректного использования функционала системы и ПО

17.5.10.1. ЗПС предоставляет возможность осуществления контроля целостности программного обеспечения и средств защиты информации как в процессе загрузки, так и динамически в процессе исполнения (см. 16.1).

17.5.10.2. МКЦ обеспечивает защиту системных компонентов (процессов, файлов) ОС, параметров функционирования СЗИ от деструктивного воздействия со стороны недоверенных пользователей. Процессы, а также конфигурационные файлы СЗИ и системного программного обеспечения обладают высокой меткой целостности, что исключает возможность оказания несанкционированного воздействия на них даже в случае получения нарушителем доступа к системе с правами `root` с низкой меткой целостности (см. 4.3).

17.5.10.3. Блокировка интерпретатора `bash`, а также иных интерпретаторов (`python`, `perl`, `expect`, `ruby`, `dash`, `php`, `irb`, `csh`, `lua`, `ksh`, `tcl`, `tk`, `zsh`, `nodejs` и др.) совместно с механизмом запрета установки бита исполнения (см. 16.6.4) обеспечивает запрет несанкционированного использования интерпретатора(ов) недоверенными пользователями для выполнения кода напрямую из командной строки, неименованного канала (`pipe`) и/или запуска сторонних сценариев (см. 16.6.7).

17.5.10.4. Блокировка использования макросов в стандартных приложениях, включая пакеты офисного программного обеспечения предотвращает несанкционированный запуск программного кода при их использовании (см. 16.6.8).

17.5.10.5. Средства контейнеризации обеспечивают запуск программного обеспечения, включая веб-браузеры, в контейнерах от имени непривилегированных пользователей (`rootless`-режим), что обеспечивает защиту системных и пользовательских данных и процессов в случае реализации угроз безопасности информации внутри изолированного контейнера. При этом средствами контейнеризации обеспечивается блокирование запуска контейнеров при нарушении целостности образов и/или конфигурации, а также при обнаружении уязвимостей в них. В случае эксплуатации контейнеров в привилегированном режиме и при включенном МКЦ может быть использован механизм запуска контейнеров с пониженной категорией целостности для защиты от реализации нарушителем угроз, связанных с преодолением изоляции контейнера, и минимизации их последствий (см. раздел 8).

17.5.11. Угрозы несанкционированного восстановления информации

17.5.11.1. Механизм очистки освобождаемой внешней памяти (включая файлы подкачки) обеспечивает затирание (в том числе путем многократной перезаписи данных псевдослучайной последовательностью байтов) освобождаемых блоков ФС непосредственно при их освобождении и/или по запросу пользователя (см. 9.1).

17.5.11.2. Механизм очистки памяти (в ядерном стеке (STACKLEAK), в ядерной куче (PAGE_POISONING)) обеспечивается ядром ОС и предотвращает несанкционированный доступ к остаточной информации, в том числе содержащей чувствительные данные (см. 9.1).

17.5.11.3. Средствами ядра ОС обеспечивается рандомизация адресного пространства процессов (см. 9.2).

17.5.12. Угрозы несанкционированного доступа к низкоуровневым данным

17.5.12.1. ЗПС предоставляет возможность осуществления контроля целостности программного обеспечения и средств защиты информации как в процессе загрузки, так и динамически в процессе исполнения (см. 16.1).

17.5.12.2. МКЦ обеспечивает защиту системных компонентов (процессов, файлов) ОС, параметров функционирования СЗИ от деструктивного воздействия со стороны недоверенных пользователей. Процессы, а также конфигурационные файлы СЗИ и системного программного обеспечения обладают высокой меткой целостности, что исключает возможность оказания несанкционированного воздействия на них даже в случае получения нарушителем доступа к системе с правами `root` с низкой меткой целостности (см. 4.3).

17.5.12.3. Киоск-2 обеспечивает возможность реализации ограничений, связанных с разрешениями запуска программного обеспечения и получением доступа к объектам ФС, для отдельных пользователей путем настройки профилей пользователей (системных профилей), что обеспечивает невозможность несанкционированного запуска произвольного кода и доступа к системным компонентам ОС (см. 16.2).

17.5.12.4. Блокировка интерпретатора `bash`, а также иных интерпретаторов (`python`, `perl`, `expect`, `ruby`, `dash`, `php`, `irb`, `csh`, `lua`, `ksh`, `tcl`, `tk`, `zsh`, `nodejs` и др.) совместно с механизмом запрета установки бита исполнения (см. 16.6.4) — обеспечивает запрет несанкционированного использования интерпретатора(ов) недоверенными пользователями для выполнения кода напрямую из командной строки, неименованного канала (`pipe`) и/или запуска сторонних сценариев (см. 16.6.7).

17.5.12.5. Механизм очистки памяти (в ядерном стеке (STACKLEAK), в ядерной куче (PAGE_POISONING)) обеспечивается ядром ОС и предотвращает несанкционированный доступ к остаточной информации, в том числе содержащей чувствительные данные (см. 9.1).

17.5.12.6. Средствами ядра ОС обеспечивается рандомизация адресного пространства процессов (см. 9.2).

17.5.12.7. Модуль безопасности lockdown обеспечивает запрет получения несанкционированного доступа (чтение/запись) к памяти и выполнения кода на уровне ядра ОС, прямого доступа к портам ввода-вывода `ioport` и т. п., в том числе со стороны привилегированного пользователя (суперпользователя `root`) в соответствии с 16.8.

17.5.12.8. Модуль безопасности yama ограничивает использование механизмов отладки (трассировки) процессов ОС, что исключает возможность получения за счет привилегии отладки несанкционированного доступа к адресному пространству других процессов (см. 16.7).

17.5.12.9. Модуль ядра lkrg обеспечивает обнаружение некоторых уязвимостей и защиту пространства памяти ядра от модификации (см. 16.6.15).

17.5.13. Угрозы «отказ в обслуживании»

17.5.13.1. МКЦ обеспечивает защиту системных компонентов (процессов, файлов) ОС, параметров функционирования СЗИ от деструктивного воздействия со стороны недоверенных пользователей. Процессы, а также конфигурационные файлы СЗИ и системного программного обеспечения обладают высокой меткой целостности, что исключает возможность оказания несанкционированного воздействия на них даже в случае получения нарушителем доступа к системе с правами `root` с низкой меткой целостности (см. 4.3).

17.5.13.2. Мандатное управление доступом позволяет категорировать защищаемую информацию (данные) по уровням и категориям конфиденциальности, что обеспечивает дополнительные возможности разграничения доступа (см. 4.2).

17.5.13.3. Киоск-2 обеспечивает возможность реализации ограничений, связанных с разрешениями запуска программного обеспечения и получением доступа к объектам ФС, для отдельных пользователей путем настройки профилей пользователей (системных профилей), что обеспечивает невозможность несанкционированного запуска произвольного кода и доступа к системным компонентам ОС (см. 16.2).

17.5.13.4. ЗПС предоставляет возможность осуществления динамического контроля целостности файлов конфигурации, а также исполняемых файлов и загружаемых библиотек при запуске и в процессе функционирования системного и прикладного программного обеспечения (см. 16.1).

17.5.13.5. Модуль безопасности lockdown обеспечивает запрет получения несанкционированного доступа (чтение/запись) к памяти и выполнения кода на уровне ядра ОС, прямого доступа к портам ввода-вывода `ioport` и т. п., в том числе со стороны привилегированного пользователя (суперпользователя `root`) в соответствии с 16.8.

17.5.13.6. Модуль безопасности yama ограничивает использование механизмов отладки (трассировки) процессов ОС, что исключает возможность получения за счет привилегии отладки несанкционированного доступа к адресному пространству других процессов (см. 16.7).

17.5.13.7. Механизм установки квот на использование системных ресурсов реализует ограничения на использование некоторых ресурсов системы для предотвращения нарушений доступности системы в результате исчерпания ресурсов (см. 16.6.3).

17.5.13.8. Мониторинг событий безопасности информации обеспечивает гибкую настройку регистрируемых событий, их автоматизированную обработку и визуализацию. Подсистема позволязет контролировать события безопасности информации, связанные как со штатными подсистемами управления доступом операционных систем семейства Linux, так и с дополнительными механизмами защиты ОС, что обеспечивает возможность своевременного реагирования на возникающие инциденты информационной безопасности, в том числе связанные с попытками обхода СЗИ (см. раздел 6).

17.5.13.9. Модуль ядра lkrg обеспечивает обнаружение некоторых уязвимостей и защиту пространства памяти ядра от модификации (см. 16.6.15).

18. УСЛОВИЯ ЭКСПЛУАТАЦИИ ОС

В целях защиты информации от несанкционированного доступа, включая обеспечение конфиденциальности, целостности, доступности информации и устойчивого функционирования информационных систем, необходимо обеспечить условия эксплуатации ОС согласно 18.1–18.4.

18.1. Обеспечение безопасности среды функционирования

В целях обеспечения безопасности среды функционирования ОС на объектах информатизации должны быть выполнены следующие требования:

1) СВТ, предназначенное для установки и функционирования ОС, должно соответствовать требованиям, указанным в документе РУСБ.10015-01 31 01;

2) для создания защищенной среды виртуализации изделие должно применяться совместно с аппаратным обеспечением, поддерживающим соответствующие технологии VT/SVM, в том числе при необходимости предоставления доступа виртуальным машинам к физическим устройствам;

3) установка, управление и конфигурирование ОС должны проводиться в соответствии с эксплуатационной документацией;

4) должна быть обеспечена защита от осуществления действий, направленных на нарушение физической целостности СВТ, на котором функционирует ОС;

5) должна быть обеспечена доверенная загрузка ОС. Доверенную загрузку ОС рекомендуется выполнять с помощью сертифицированных средств доверенной загрузки или аппаратно-программных модулей доверенной загрузки. При технической невозможности или нецелесообразности использования таких средств должны быть приняты организационно-технические меры, предотвращающие возможность доступа пользователя к ресурсам СВТ в обход механизмов защиты ОС (должна быть исключена возможность загрузки альтернативной операционной системы на СВТ, а также модификация модулей загружаемой ОС);

6) при использовании сетевого взаимодействия необходимо обеспечить доверенный канал передачи информации между СВТ, на которых установлена ОС (например, контроль несанкционированного подключения к ЛВС в пределах контролируемой зоны, защищенная передача сетевого трафика за пределами контролируемой зоны);

7) должны быть определены лица, отвечающие за эксплуатацию и безопасность ОС, которым будут предоставлены административные полномочия в ОС. Данные лица не должны считаться нарушителями в модели угроз безопасности информации;

8) лица, ответственные за эксплуатацию ОС, должны обеспечить, чтобы аутентификационная информация для каждой учетной записи пользователя ОС содержалась в тайне и была недоступна лицам, не уполномоченным использовать данную учетную запись.

18.2. Указания по эксплуатации ОС

18.2.1. Перед началом эксплуатации ОС администратор безопасности должен обеспечить следующие условия:

1) если разрешен интерактивный вход суперпользователя `root`, то для предотвращения подбора пароля должна быть заблокирована возможность удаленного входа суперпользователя `root` в ОС. Для этого необходимо включить PAM-модуль `pam_securetty` в файл сценария `/etc/pam.d/common-auth`, добавив в начале секции `Primary block` строку:

```
auth required pam_securetty.so
```

2) должна быть настроена политика паролей. Выполнить настройку возможно либо с помощью графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку), запущенной от имени администратора через механизм `sudo`, либо путем редактирования файлов сценариев:

а) в файле `/etc/pam.d/common-password` в строке `password requisite pam cracklib.so` установить значение `minlen=<минимальная_длина_пароля>` и добавить параметры `dcredit=-1`, `ucredit=-1` и `lcredit=-1` (данные параметры устанавливают требования к сложности пароля);

б) в файле `/etc/login.defs` для переменной `PASS_MAX_DAYS` установить требуемое значение максимального срока действия пароля в днях;

в) в файле `/etc/pam.d/common-password` возможно при необходимости установить запрет повторного использования заданного числа последних паролей, для этого в строке `...pam_unix.so` добавить параметр `remember=<число_паролей>`;

3) должны быть заданы настройки блокировки при неуспешных попытках входа пользователя. Задать настройки возможно либо с помощью графической утилиты `astra-systemsettings` («Параметры системы», см. электронную справку), запущенной от имени администратора через механизм `sudo`, либо путем корректировки файла `/etc/pam.d/common-auth` (параметры `deny`, `lock_time` и `unlock_time`);

4) установку ОС следует производить на системный раздел с файловой системой ext4/XFS. При использовании других файловых систем необходимо учитывать, что функции разграничения доступа, мандатного контроля целостности и гарантированного уничтожения (стирания) информации в полном объеме реализованы только для файловых систем ext2/ext3/ext4/XFS, поддерживающих расширенные (в т.ч. мандатные) атрибуты файловых объектов;

5) при необходимости установки в систему внешних модулей уровня ядра ОС, такие модули должны быть получены из доверенного источника, и перед их установкой с ОС должна осуществляться проверка их целостности;

6) использование файловой системы NFS для доступа к сетевым дискам возможно при выполнении следующих условий:

а) сетевые диски не предназначены для хранения файловых объектов, требующих обеспечения мандатного управления доступом к ним;

б) версия используемого протокола NFS не ниже 3;

7) должно быть задано значение времени неактивности для блокировки экрана, для этого отредактировать (или создать, если отсутствует) файл `usr/share/fly-wm/theme.master/themerc`, указав в нем строки:

```
[Variables]
ScreenSaverDelay=<время_неактивности_в_секундах>
LockerOnSleep=true
LockerOnDPMS=true
LockerOnLid=true
LockerOnSwitch=true
```

18.2.2. При использовании механизма гарантированного удаления данных дополнительно при использовании разделов подкачки в ОС необходимо активировать в файле `/etc/parsec/swap_wiper.conf` их очистку согласно 9.1.

18.2.3. При использовании мандатного контроля целостности дополнительно при использовании инструмента `qemu-ga` (из состава пакета `qemu-guest-agent`) значения параметров `-p` (в том числе при использовании параметра `-m unix-listen`) и `-t` должны указывать на сокеты и файлы, метки целостности которых совпадают с меткой целостности запускаемого процесса.

18.2.4. При использовании мандатного управления доступом должны дополнительно быть выполнены следующие условия:

1) с использованием средств управления запуском служб должна быть отключена служба `gpm` для поддержки мыши в консольном режиме;

2) в случае необходимости использования мандатного управления доступом на сетевых дисках, их монтирование в файловую систему ОС должно осуществляться только с использованием файловой системы CIFS, поддерживающей расширенные (в т.ч. мандатные) атрибуты пользователей;

3) в конфигурационном файле защищенного сервера печати из состава изделия `/etc/cups/cupsd.conf` не допускается установка значения `None` параметра `DefaultAuthType` (отключение аутентификации) и внесение изменений в параметры политики подсистемы безопасности PARSEC, не соответствующих эксплуатационной документации;

4) при использовании защищенного сервера СУБД из состава ОС в режиме мандатного управления доступом необходимо в конфигурационном файле кластера `postgresql.conf` для параметра `enable_bitmapscan` установить значение `off` и для параметра `ac_ignore_socket_maclabel` установить значение `false`;

5) при использовании защищенного сервера СУБД из состава ОС в режиме мандатного управления доступом не допускается отключать аутентификацию субъектов доступа установкой в конфигурационном файле кластера `pg_hba.conf` режима `trust` (без аутентификации);

6) при использовании защищенного комплекса программ электронной почты из состава ОС в режиме мандатного управления доступом конфигурационные параметры агента передачи электронной почты Exim и агента доставки электронной почты Dovecot не должны допускать отправку и прием сообщений электронной почты без аутентификации;

7) для штатной работы ОС не допускается отключение администратором системы расширения `XPARSEC` у X-сервера (использование отладочной опции `XPARSEC=Disable` в конфигурационных файлах X-сервера, см. документ РУСБ.10015-01 31 01).

18.2.5. В целях обеспечения удаленного доступа пользователей с использованием сетей связи общего пользования к средствам виртуализации из состава ОС должны применяться средства криптографической защиты информации, прошедшие процедуру оценки соответствия согласно законодательству Российской Федерации.

18.2.6. Запрещается использование образа контейнера при выявлении в нем уязвимостей.

18.3. Условия применения ПО

18.3.1. ПО, функционирующее в среде ОС, не должно изменять алгоритм принятия решения о предоставлении доступа субъектов (пользователей) к объектам ОС и содержать скрытых или явных возможностей, позволяющих:

1) подменять файлы образа ядра `vmlinuz-*` и образов временной файловой системы начальной загрузки `/boot/initrd.img-*` (в случае отсутствия явной необходимости обновления отдельных компонентов, входящих в состав образа), находящиеся в каталоге `/boot`, файлы модулей ядра, находящиеся в каталоге `/lib/modules`, и прочие файлы, входящие в состав изделия или стороннего ПО;

2) статически или динамически изменять таблицу системных вызовов и поля структуры типа `security_operations` и иных структур типа `*security*`;

3) переопределять основной процесс ОС в конфигурационном файле загрузчика изделия путем установки параметра `init=<полный_путь_к_исполняемому_файлу>`;

4) изменять параметры аутентификации в конфигурационных файлах PAM-сценариев, находящихся в каталоге `/etc/pam.d`, результатом чего может являться снижение установленного уровня доверия к результатам идентификации и аутентификации (по ГОСТ Р 58833-2020);

5) устанавливать подгружаемые модули аутентификации (PAM-модули), определяющие мандатные атрибуты сессии пользователя с использованием функций API библиотеки `libparsec-mac` подсистемы безопасности PARSEC, имена которых начинаются с префиксов `mac_set_`, `parsec_` или `pdp_`;

6) устанавливать PAM-модули, определяющие привилегии PARSEC сессии пользователя с использованием функций API библиотеки `libparsec-cap` подсистемы безопасности PARSEC, имена которых начинаются с префиксов `mcap` или `parsec_`;

7) устанавливать интерпретаторы команд, заменяющие интерпретаторы, входящие в состав изделия (bash, dash, PHP, Perl, Python, TCL, Ruby);

8) получать доступ к памяти других процессов ПО с использованием привилегии `CAP_SYS_PTRACE` и системного вызова `ptrace`;

9) изменять системное время (если наличие возможностей по изменению времени не предусмотрено функциональным назначением ПО);

10) динамически изменять сегмент кода ядра ОС и использовать неэкспортируемые символы ядра ОС;

11) осуществлять доступ к файлу `ctl` файловой системы `parsecfs` посредством системного вызова `ioctl`, минуя системные функции API-библиотек `libparsec-*`.

18.3.2. Для ПО, функционирующего в среде ОС, должны соблюдаться следующие условия:

1) необходимость и корректность использования в ПО привилегий администратора, прикладного программного интерфейса подсистемы безопасности PARSEC, мандатных привилегий, а также функционирования ПО в пространстве ядра или только с высокой меткой целостности должны быть обоснованы в документации на данное ПО;

2) для ПО, использующего модуль `wsgi_mod` для Apache из состава изделия, в документации разработчика ПО должен быть определен перечень сценариев работы модуля `wsgi_mod`. При этом должны быть исключены сценарии работы `wsgi_mod` в режимах, предусматривающих работу вне контекста пользователя и изменяющих процесс аутентификации.

18.3.3. При использовании защищенного сервера печати из состава изделия установка и использование альтернативных серверов печати должны быть исключены.

18.3.4. При использовании механизма мандатного управления доступом дополнительно должны соблюдаться следующие условия:

1) ПО при обработке запросов пользователей не должно взаимодействовать с защищенной СУБД из состава изделия от имени пользователя с привилегиями `PARSEC_CAP_SETMAC` и `PARSEC_CAP_CHMAC`, позволяющими изменять мандатные атрибуты защищаемых объектов в СУБД;

2) ПО, содержащее сетевые службы (программы, обрабатывающие запросы пользователей с применением протоколов стека TCP/IP), которые используют привилегии

(PARSEC_CAP_SETMAC, PARSEC_CAP_CHMAC и PARSEC_CAP_PRIV_SOCK) и прикладной программный интерфейс подсистемы безопасности PARSEC из состава изделия, должно пройти сертификационные исследования, подтверждающие корректность реализации указанных служб;

3) программный компонент RabbitMQ не применять при одновременной обработке данных с различными уровнями конфиденциальности либо для обеспечения взаимодействия процессов с различными уровнями доступа;

4) ПО, обрабатывающее одновременно данные с различными уровнями конфиденциальности, не должно использовать программные пакеты Erlang.

18.3.5. Не допускается изменять (подменять) ПО ОС, реализующее функции безопасности информации, приведенные в настоящем документе.

При проверке подлинности и целостности ПО, реализующего сертифицированные функции безопасности информации изделия, путем подсчета контрольных сумм с применением средств контроля целостности (fly-admin-int-check, astra-int-check, см. 10.3) необходимо применять шаблон проверки со списком файлов, реализующих функции безопасности.

18.3.6. В случае применения среды выполнения не из состава ОС (например, Java Runtime Environment) для запуска СПО, написанного на интерпретируемом или компилируемом в промежуточное представление языке программирования и предназначенного для обработки информации ограниченного доступа, необходимо обеспечить функционирование данного СПО в рамках (в контексте) одного изолированного компонента (процесса) среды выполнения. В противном случае, в целях исключения угроз безопасности информации и изоляции процессов рекомендуется запускать такое СПО в изолированной программной среде (контейнере) от имени непривилегированного пользователя и/или применять сторонние среды исполнения, имеющие действующий сертификат соответствия требованиям безопасности в данном классе систем.

18.4. Условия исключения скрытых каналов

Вопросы исключения скрытых каналов должны рассматриваться при построении информационных систем, обрабатывающих информацию, содержащую государственную тайну, и могут рассматриваться при построении информационных систем, обрабатывающих информацию ограниченного доступа, не содержащую государственную тайну (конфиденциальную информацию).

Для исключения скрытых каналов (информационных потоков) при защите от угрозы целостности информации используется мандатный контроль целостности (см. 4.9), который в условиях установленных в компьютерной системе меток целостности (уровней доверия) обеспечивает невозможность записи отправителем с низким уровнем доверия информации в объекты с более высоким уровнем доверия.

Для исключения возможности использования отправителем и получателем общего буфера обмена в ОС обеспечивается изоляция процессов (см. раздел 7) в совокупности с очисткой областей оперативной памяти при освобождении или выделении (см. 9.1), а также обеспечение запуска процессов в замкнутой относительно остальных процессов среде по памяти, в частности, использование Киоск-2 (см. 16.2) и Xephyr в графической подсистеме, обеспечивающих изоляцию буфера обмена.

Для исключения возможности запуска программ, не участвующих в процессе обработки информации, должен быть активирован механизм замкнутой программной среды и настроен для работы в штатном режиме.

Таким образом, условиями исключения скрытых каналов является реализуемая ОС политика дискреционного управления доступом (см. раздел 3), мандатного управления доступом и мандатного контроля целостности (см. раздел 4), которая является неотъемлемой частью модели политики безопасности ОС. С учетом указанных условий эксплуатации скрытые каналы по памяти не могут быть реализованы, а по времени крайне затруднены, т.е. имеют низкие пропускную способность и вероятность возникновения условий для реализации.

ПЕРЕЧЕНЬ ТЕРМИНОВ

Закрытый ключ — сохраняемый в тайне ключ из ключевой пары, принадлежащий владельцу и не подлежащий распространению.

Ключ — параметр в виде последовательности псевдослучайных чисел (не предназначен для защиты информации в контексте использования для целей, установленных в документации изделия; к ключам не предъявляются требования по источнику псевдослучайных чисел, криптографической стойкости, времени действия и т. п.).

Ключевая пара — упорядоченная пара математически однозначно связанных ключей, определяющих взаимосвязанные защитные преобразования.

Открытый ключ — ключ из ключевой пары, который может быть сделан общедоступным.

Сертификат открытого ключа — артефакт, содержащий открытый ключ, информацию о владельце ключа и подтверждающий принадлежность открытого ключа владельцу, защищенный с применением закрытого ключа.

Хеш — строка бит, являющаяся выходным результатом функции хеширования.

Центр аутентификации — программный компонент, реализующий возможность подтверждения подлинности ключей с помощью сертификатов.

Цифровая подпись — результат преобразования хеша для его защиты от несанкционированного доступа с использованием закрытого ключа (не предназначена для криптографической защиты информации).

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

БД — база данных
ЕПП — единое пространство пользователей
КСЗ — комплекс средств защиты
МКЦ — мандатный контроль целостности
НСД — несанкционированный доступ
ОС — операционная система специального назначения «Astra Linux Special Edition»
ПО — программное обеспечение
ПРД — правила разграничения доступа
СВТ — средство вычислительной техники
СЗИ — средства защиты информации
СЗФС — сетевая защищенная файловая система
СПО — специальное программное обеспечение
СУБД — система управления базами данных
ФС — файловая система
ЭП — электронная подпись (в соответствии с № 63-ФЗ от 06.04.2011)

ACL — Access Control List (список контроля доступа)
BIND — Berkley Internet Name Domain (служба доменных имен в сети Интернет)
CCR — Container Clearance Required (атрибут способа доступа к содержимому контейнера в рамках мандатного управления доступом)
CIFS — Common Internet File System (общий протокол доступа к файлам Интернет)
DAC — Discretionary Access Control (дискреционное управление доступом)
DNS — Domain Name System (система доменных имен)
FTP — File Transfer Protocol (протокол передачи файлов)
GID — Group Identifier (идентификатор группы)
IP — Internet Protocol (межсетевой протокол)
IPC — InterProcess Communication (межпроцессное взаимодействие)
KVM — Kernel-based Virtual Machine (программное решение, обеспечивающее виртуализацию в среде Linux на платформе, которая поддерживает аппаратную виртуализацию на базе Intel VT (Virtualization Technology) либо AMD SVM (Secure Virtual Machine))
LDAP — Lightweight Directory Access Protocol (легковесный протокол доступа к сервисам каталогов)
MAC — Mandatory Access Control (мандатное управление доступом)
NFS — Network File System (сетевая файловая система)
NIS — Network Information Service (сетевая информационная служба)
NSS — Name Service Switch (служба для организации унифицированного доступа к информации о пользователях, группах, сетевых именах, службах и т.п. на основе конфигурации различных источников, таких как: локальные файлы, различные средства сетевой идентификации (DNS, NIS), а также LDAP)

PAM — Pluggable Authentication Modules (подключаемые модули аутентификации)
PID — Process Identifier (идентификатор процесса)
QEMU — Quick Emulator (средства эмуляции аппаратного обеспечения)
RSA — Rivest Shamir Adelman (алгоритм по схеме открытого ключа)
SMB — Server Message Block (блок сообщений сервера)
SQL — Structured Query Language (язык структурированных запросов)
TCP — Transmission Control Protocol (протокол передачи данных)
UDP — User Datagram Protocol (протокол управления передачей данных)
UID — User Identifier (идентификатор пользователя)

Лист регистрации изменений

Изм.	Номера листов (страниц)				Всего листов (страниц) в документе	Номер документа	Входящий номер сопроводительного документа и дата	Подпись	Дата
	измененных	замененных	новых	аннулированных					
1		Все				РУСБ.14.03-11 от 20.10.11			
2		Все				РУСБ.14.04-13 от 21.03.13			
3		Все				РУСБ.14.05-14 от 19.11.14			
4		Все				РУСБ.14.02-16 от 17.02.16			
5		Все				РУСБ.14.03-17 от 05.04.17			
6		Все				РУСБ.14.02-18 от 11.04.18			
7		Все				РУСБ.14.03-18 от 19.04.18			
8		Все				РУСБ.08-20 от 18.02.20			
9		Все				РУСБ.20-20 от 23.07.20			
10		Все				РУСБ.06-21 от 01.04.21			
11		Все				РУСБ.12-21 от 30.07.21			
12		Все				РУСБ.10-22 от 03.11.22			
13		Все				РУСБ.03-23 от 26.04.23			
14		Все				РУСБ.14-23 от 26.12.23			
15		Все				РУСБ.02-24 от 25.03.24			
16		Все				РУСБ.04-24 от 01.07.24			