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 172

2018

Литера $О_1$

АННОТАЦИЯ

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

Руководство по КСЗ состоит из двух 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».

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 киоска и порядок запуска и применения ОС.

Во второй части руководства приведено описание тестов КСЗ.

СОДЕРЖАНИЕ

1. ОБЩИЕ СВЕДЕНИЯ

ОС предназначена для построения автоматизированных систем в защищенном исполнении, обрабатывающих информацию, содержащую сведения, составляющие государственную тайну с грифом не выше «совершенно секретно».

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. 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 подключаемым устройствам;
- средства разграничения доступа к виртуальным машинам.

1.2. Контролируемые функции

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

- идентификацию и аутентификацию;
- дискреционное управление доступом;
- мандатное управление доступом;
- регистрацию событий безопасности;
- ограничение программной среды;

- изоляцию процессов;
- защиту памяти;
- контроль целостности;
- обеспечение надежного функционирования;
- фильтрацию сетевого потока;
- маркировку документов;
- защиту среды виртуализации;
- контроль подключения съемных машинных носителей информации.

Реализация перечисленных выше функций основана на следующих основных положениях:

1) с каждым пользователем системы связан уникальный численный идентификатор — идентификатор пользователя (UID), который является ключом к соответствующей записи 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;

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

3) роль действительного (работающего с объектами) 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 объекту доступ предоставляется по результатам процедуры авторизации, т. е. обработки запроса на основе мандатных и дискреционных ПРД;

4) механизм ПРД реализован в ядре ОС, что обеспечивает его правильное функционирование при использовании любых компонентов, предоставляемых ОС. Реализа-

ция мандатного управление 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 относительно дискреционных и мандатных ПРД. Запрещается в случае, если доступ запрещен относительно любого из механизмов.

1.3. 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 кросс-платформенных 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 сервисов 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 мандатные уровни и категории;
- минимальные и максимальные мандатные уровни, доступные пользователям при входе в систему;
- минимальные и максимальные наборы мандатных категорий, 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. Период обновления локального кэша задается параметром `CACHE_REFRESH_PERIOD` в конфигурационном файле `/etc/ald/ald.conf`.

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

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 ряде 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 `/etc/passwd`. Информация, которая хранится в `/etc/shadow` и используется для пользователей в локальной БД. В ОС реализована возможность хранения аутентификационной информации пользователей, полученной с использованием хеш-функций по ГОСТ Р 34.11-94 и по ГОСТ Р 34.11-2012.

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 сервисы, 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 сервисам.

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 настройками их атрибутов используется графическая утилита `fly-admin-smc`. Описание графической утилиты см. в электронной справке.

Для управления БД ALD в режиме командной строки используется утилита `ald-admin`, подробное описание которой приведено в `man ald-admin`.

Примечание. При создании локальных пользователей или пользователей ЕПП необходимо обязательно устанавливать для них диапазоны допустимых мандатных уровней и категорий: минимальный и максимальный мандатные уровни, минимальный и максимальный наборы мандатных категорий (раздел 4). Отсутствие установленных допустимых диапазонов мандатных атрибутов приводит к запрещению доступа при обращении к сетевым сервисам 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 сервис 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 user_name
```

После успешного завершения попытки аутентификации пользователя счетчик неуспешных попыток аутентификации пользователя сбрасывается в ноль. Максимальное число неуспешных попыток аутентификации пользователя определяется утилитой `faillog` и значениями параметров `deny` и `per_user`:

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

Использование параметра `per_user` означает, что при установке утилитой `faillog` максимального значения неуспешных попыток аутентификации для пользователя, не равного `0`, применяется указанное значение. Иначе применяется значение, определяемое параметром `deny`. При отсутствии установленного параметра `per_user` используется значение параметра `deny`.

Для сброса счетчика неуспешных попыток аутентификации необходимо выполнить следующую команду:

```
faillog -r -u user_name
```

Более подробное описание см. в руководстве `man` на `faillog` и `pam_tally`.

3. ДИСКРЕЦИОННОЕ УПРАВЛЕНИЕ ДОСТУПОМ

3.1. Общие сведения

В ОС реализован механизм дискреционных ПРД именованных субъектов (пользователей) к именованным объектам. Реализация механизма дискреционных ПРД обеспечивает наличие для каждой пары (субъект-объект) явное и недвусмысленное перечисление разрешенных типов доступа.

Дискреционный контроль доступа применяется к каждому объекту и каждому субъекту и заключается в том, что на защищаемые именованные объекты устанавливаются (автоматически при их создании) базовые ПРД в виде идентификаторов номинальных субъектов (UID и GID), которые вправе распоряжаться доступом к данному объекту и прав доступа к объекту. Определяются три вида доступа: чтение (read, r), запись (write, w) и исполнение (execution, x). Права доступа включают список (битовую маску) из девяти пунктов: по три вида доступа для трех классов — пользователя-владельца, группы-владельца и всех остальных. Каждый пункт в этом списке может быть либо разрешен, либо запрещен (равен 1 или 0).

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

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

Существуют также специальные биты, такие как:

- `SUID` (Set User ID) — бит смены идентификатора пользователя;
- `SGID` (Set Group ID) — бит смены идентификатора группы;
- `Sticky` — определяет владельца объектов в каталоге.

Когда пользователь или процесс запускает исполняемый файл с одним из установленных битов `SUID` или `SGID`, то файлу временно назначаются права его (файла) владельца или группы (в зависимости от того, какой бит задан). Таким образом, пользователь может даже запускать файлы от имени суперпользователя.

Каталог с установленным `Sticky`-битом означает, что удалить файл из этого каталога может только владелец файла или суперпользователь. Другие пользователи лишаются права удалять файлы. Установить `Sticky`-бит в каталоге можно только от имени суперпользователя с использованием механизма `sudo`. `Sticky`-бит каталога остается до тех пор, пока владелец каталога или суперпользователь не удалит каталог явно или не изменит права доступа. Владелец может удалить `Sticky`-бит, но не может его установить.

Дополнительно в ОС механизмом дискреционных ПРД поддерживаются списки контроля доступа ACL (Access Control List), реализованные на основе расширенных атрибутов файловых систем. С использованием ACL можно дополнительно для каждого объекта задавать права на доступ субъектов к нему.

ACL состоит из набора записей. Права доступа к объекту для пользователя-владельца, группы-владельца и всех остальных имеют соответствующее представление в ACL в виде отдельных записей. ACL соответствующий базовым ПРД называется минимальным ACL. Таким образом, к каждому объекту доступа всегда сопоставляется минимальный ACL, включающий три записи: для пользователя-владельца, группы-владельца и всех остальных. Права доступа для дополнительных субъектов определяются в дополнительных записях ACL.

ACL включающий более трех записей называется расширенным ACL. Он дополнительно содержит запись для маски доступа и набор записей для именованных пользователей и именованных групп.

В общем случае ACL включает записи следующих типов:

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

Запись маски доступа используется для ограничения распространения прав доступа именованных пользователей и групп.

В механизме дискреционных ПРД реализовано отображение прав доступа к объекту, указанных в битовой маске для трех классов (пользователя-владельца, группы-владельца и всех остальных) в соответствующие записи ACL.

При использовании минимального ACL:

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

При использовании расширенного ACL:

- права доступа из битовой маски для класса пользователь-владелец (например, `rwx`) отображаются в идентичные права доступа записи ACL типа пользователь-владелец (например, `user::rwx`);
- права доступа из битовой маски для класса группа-владельца (например, `rw-`) отображаются в идентичные права доступа записи ACL типа маска доступа (например, `mask::rw-`);
- права доступа из битовой маски для класса все остальные (например, `r--`) отображаются в идентичные права доступа записи ACL типа все остальные (например, `other::r--`).

Реализованная в механизме дискреционных ПРД проверка прав доступа субъекта к объекту выполняется в два этапа. На первом этапе выбирается запись ACL, соответствующая субъекту доступа. Записи ACL для объекта доступа просматриваются в следующем порядке:

1) запись для пользователя-владельца;
2) записи именованных пользователей;
3) запись группы-владельца;
4) записи именованных групп;
5) запись для всех остальных.

Решение о доступе принимается только на основе одной выбранной записи ACL. На втором этапе проверяется, что запись ACL содержит необходимые права доступа.

Алгоритм проверки прав доступа имеет следующий вид:

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

2) проверяется является ли субъект доступа одним из именованных пользователей, указанных в записях ACL. Если субъект доступа не является именованным пользователем, указанным в записях ACL, то проверка продолжается. Если субъект доступа является именованным пользователем, 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;

3) проверяется, является ли одна из групп субъекта доступа группой-владельца объекта. Если ни одна из групп субъекта доступа не является группой-владельца

объекта, то проверка продолжается. Если одна из групп субъект доступа является группой-владельца объекта, то проверяется разрешен ли в соответствии с выбранной записью ACL запрошенный субъектом вид доступа. По результатам проверки доступ либо разрешается либо запрещается;

4) проверяется, является ли одна из групп субъекта доступа одной из именованных групп, указанных в записях ACL. Если ни одна из групп субъекта доступа не является именованной группой, указанной в записях ACL, то проверка продолжается. Если одна из групп субъект доступа является именованной группой, 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;

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 доступ запрещается.

Реализованный в ОС механизм дискреционных ПРД предусматривает наличие у объектов-контейнеров ACL, используемого по умолчанию. Названный ACL наследуется объектами, создаваемыми в объекте-контейнере.

Объектами доступа являются:

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

Механизм, реализующий дискреционное управление доступом, обеспечивает возможность санкционированного изменения списка пользователей и списка защищаемых файловых объектов.

Право изменения ПРД предоставлено выделенному субъекту - суперпользователю root (пользователю с UID, имеющим значение 0). Администратор может изменять права с использованием механизма `sudo`. Кроме того, права доступа к объекту, как указанные в битовой маске для трех классов (пользователя-владельца, группы-владельца и всех остальных), так указанные в записях ACL могут быть изменены субъектом являющимся пользователем-владельцем объекта.

Реализация в ОС механизма дискреционных ПРД обеспечивает непротиворечивость правил изменения дискреционных ПРД.

При использовании для объекта минимального ACL прямое и обратное отображение

ПРД обеспечивается следующим образом:

1) при изменении прав доступа в битовой маске для пользователя-владельца идентичным образом изменяются права доступа для пользователя-владельца в записи ACL;

2) при изменении прав доступа для пользователя-владельца в записи ACL идентичным образом изменяются права доступа в битовой маске для пользователя-владельца;

3) при изменении прав доступа в битовой маске для группы-владельца идентичным образом изменяются права доступа для группы-владельца в записи ACL;

4) при изменении прав доступа для группы-владельца в записи ACL идентичным образом изменяются права доступа в битовой маске для группы-владельца;

5) при изменении прав доступа в битовой маске для всех остальных идентичным образом изменяются права доступа для всех остальных в записи ACL;

6) при изменении прав доступа для всех остальных в записи ACL идентичным образом изменяются права доступа для всех остальных в битовой маске.

При использовании для объекта расширенного ACL прямое и обратное отображение ПРД обеспечивается следующим образом:

1) при изменении прав доступа в битовой маске для пользователя-владельца идентичным образом изменяются права доступа для пользователя-владельца в записи ACL;

2) при изменении прав доступа для пользователя-владельца в записи ACL идентичным образом изменяются права доступа в битовой маске для пользователя-владельца;

3) при изменении прав доступа в битовой маске для группы-владельца идентичным образом изменяется маска доступа в записи ACL;

4) при изменении маски доступа в записи ACL идентичным образом изменяются права доступа в битовой маске для группы-владельца;

5) при изменении прав доступа в битовой маске для всех остальных идентичным образом изменяются права доступа для всех остальных в записи ACL;

6) при изменении прав доступа для всех остальных в записи ACL идентичным образом изменяются права доступа для всех остальных в битовой маске.

Таким образом, реализованный в ОС механизм, регулирующий дискреционный принцип контроля доступа, предусматривает санкционированное изменение дискреционных ПРД, включая санкционированное изменение списка субъектов и списка защищаемых объектов.

3.2. Linux-привилегии

Linux-привилегии предназначены для передачи 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 суперпользователя, независимо от наличия у них привилегий, имеют возможность осуществлять все перечисленные привилегированные действия.

Система привилегий ОС расширена привилегиями, относящимися к системе PARSEC. PARSEC-привилегии, описанные в 4.5, обеспечивают работу с механизмом мандатного управление доступом.

Все привилегии пользователя наследуются запущенными от имени его учетной записи процессами. 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 невозможности запуска приложений, использующих динамическую загрузку библиотек из нестандартных каталогов (например, Firefox, Thunderbird, LibreOffice, fly-scan).

Для настройки КСЗ могут использоваться как Linux-, так и PARSEC-привилегии. Порядок управления привилегиями описан в 4.9.

3.3. Средства управления дискреционными ПРД

Для управления дискреционными ПРД используется графическая утилита fly-fm («Менеджер файлов»). Более подробное описание утилиты см. в электронной справке.

Для управления Linux-привилегиями пользователей системы используется графическая утилита fly-admin-smc («Управление политикой безопасности»). Более подробное описание утилиты см. в электронной справке.

Далее рассмотрены средства управления дискреционными ПРД в режиме командной строки.

3.3.1. chown

Синтаксис:

```
chown [OPTION]... OWNER[:[GROUP]] FILE...
chown [OPTION]... :GROUP FILE...
chown [OPTION]... --reference=RFILE FILE...
```

Команда `chown` изменяет владельца и/или группу, владеющую каждым из указанных файлов, согласно заданным аргументам, которые интерпретируются в последовательном порядке. Если задано только имя пользователя (или его числовой идентификатор), то данный пользователь становится владельцем каждого из указанных файлов, а группа этих файлов не изменяется. Если за именем пользователя через двоеточие следует имя группы (или числовой идентификатор группы) без пробелов между ними, то изменяется также и группа файлов. Если двоеточие или точка следует за именем пользователя, но группа не задана, то данный пользователь становится владельцем указанных файлов, а группа указанных файлов изменяется на основную группу пользователя. Если опущено имя пользователя, а двоеточие или точка вместе с группой заданы, то будет изменена только группа указанных файлов; в этом случае `chown` выполняет ту же функцию, что и `chgrp` (3.3.2). Команда `chown` изменяет владельца и/или группу каждого `FILE` на `OWNER` и/или `GROUP`.

Опции приведены в таблице 1.

Таблица 1

Опция	Описание
`-c`, `--changes`	То же, что и `--verbose`. Подробно описывать только файлы, чей владелец действительно изменяется
`--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. Данная опция доступна, 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 быть опущены, 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`	Вместо заданных значений `OWNER:GROUP` использовать владельца и группу файла, которые имеют `RFILE`
`-R`, `--recursive`	Рекурсивно изменять владельца каталогов и всего их содержимого
`--help`	Вывести справку и выйти

Окончание таблицы 1

Опция	Описание
`--version`	Вывести информацию о версии и выйти

Владелец не изменяется, если он не существует. Группа также не изменяется, если отсутствует, но изменяется на группу по умолчанию, если не задан пользователь.

3.3.2. chgrp

Синтаксис:

```
chgrp [OPTION]... GROUP FILE...
chgrp [OPTION]... --reference=RFILE FILE...
```

Команда `chgrp` изменяет группу, владеющую каждым из указанных файлов `FILE`, на группу `GROUP`, которая может быть задана именем группы или числовым идентификатором группы.

Опции приведены в таблице 2.

Таблица 2

Опция	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`	Изменить группу файла `FILE` на ту, что владеет файлом `RFILE`
`-R`, `--recursive`	Рекурсивно изменять владельца каталогов и их содержимого
`-v`, `--verbose`	Выводить диагностическое сообщение об изменении владельца для каждого файла
`--help`	Вывести справку и выйти
`--version`	Вывести информацию о версии и выйти

3.3.3. chmod

Синтаксис:

```
chmod [OPTION]... MODE[,MODE]... FILE...
chmod [OPTION]... OCTAL-MODE... FILE...
chmod [OPTION]... --reference=RFILE FILE...
```

Команда `chmod` изменяет права доступа указанного файла `FILE` в соответствии с правами доступа, указанными в параметре `MODE`, который может быть представлен как

в символьном виде, так и в виде восьмеричного числа, представляющего битовую маску новых прав доступа.

Формат символьного режима:

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

Каждый аргумент — это список символьных команд изменения прав доступа, разделенных запятыми. Каждая такая команда начинается с нуля или более букв `ugoa`, комбинация которых указывает, чьи права доступа к файлу будут изменены: пользователя, владеющего файлом (`u`); пользователей в данной группе (`g`); остальных пользователей, не входящих в данную группу (`o`), или же всех пользователей (`a`). Буква `a` эквивалентна `ugo`. Если не задана ни одна буква, то автоматически будет использоваться буква `a`, но биты, установленные в `umask`, не будут затронуты.

Оператор «+» добавляет выбранные права доступа к уже имеющимся у каждого файла; «-» удаляет эти права; а «=» присваивает только эти права каждому указанному файлу.

Буквы `rwxXstugo` выбирают новые права доступа для пользователя, заданного одной из букв `ugoa`: чтение (`r`); запись (`w`); исполнение (или доступ к каталогу) (`x`); выполнение, если файл является каталогом или уже имеет право на выполнение для какого-нибудь пользователя (`X`); setuid- или setgid-биты (`s`); sticky-бит (`t`); установка для остальных таких же прав доступа, которые имеет пользователь, владеющий этим файлом (`u`); установка для остальных таких же прав доступа, которые имеет группа файла (`g`); установка для остальных таких же прав доступа, которые имеют остальные пользователи (не входящие в группу файла) (`o`). (Так, `chmod g-s file` снимает бит `set-group-ID` (`sgid`), `chmod ug+s file` устанавливает биты `suid` и `sgid`, в то время как `chmod o+s file` ничего не делает.)

Числовой режим состоит из не более четырех восьмеричных цифр (от нуля до семи), которые складываются из битовых масок 4, 2 и 1. Любые пропущенные разряды дополняются лидирующими нулями. Первая цифра выбирает установку идентификатора пользователя (`setuid`) (4) или идентификатора группы (`setgid`) (2) или sticky-бита (1). Вторая цифра выбирает права доступа для пользователя, владеющего данным файлом: чтение (4), запись (2) и исполнение (1); третья цифра выбирает права доступа для пользователей, входящих в данную группу, с тем же смыслом, что и у второй цифры; и четвертый разряд выбирает права доступа для остальных пользователей (не входящих в данную группу), опять с тем же смыслом.

Команда `chmod` никогда не изменяет права на символьные ссылки, т. к. этого не делает системный вызов `chmod`. Это не является проблемой: права символьных ссылок никогда не используются. Однако для каждой символьной ссылки, заданной в командной строке, `chmod` игнорирует символьные ссылки, встречающиеся во время рекурсивной обработки

каталогов.

Команда `chmod` изменяет права доступа каждого файла `FILE` на `MODE`.

Опции приведены в таблице 3.

Таблица 3

Опция	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

Каждый `MODE` представляет собой комбинацию из одного или более символов `ugoa` в начале и один из символов «+», «-», «=», затем одна или несколько букв из `rwxXstugo`.

Символьная форма приведена в таблице 4.

Таблица 4

Опция	Описание
`u`	Пользователь (владелец файла) — от user (пользователь)
`g`	Группа — от group (группа)
`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
`r`	Разрешение на чтение — от read (читать)
`w`	Разрешение на изменение — от write (писать)
`x`	Разрешение на исполнение — от execute (выполнять)
`l`	Блокировка файла для других пользователей при доступе

3.3.4. umask

Синтаксис:

```
umask [-p] [-S] [маска]
```

Пользовательская маска создания файла устанавливается равной аргументу 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 задана опция -S, выдается текущее значение маски. Опция -S вызывает выдачу маски в символьном формате; по умолчанию выдается восьмеричное число. Если указана опция -p, а маска не задана, результат выдается в виде, который можно использовать во входной команде. Статус выхода — 0, если маска была успешно изменена или не указана, и 1 — в противном случае.

Команда umask распознается и выполняется оболочкой shell.

Команду umask целесообразно включить в пользовательский pro-файл. Тогда она будет автоматически вызываться при входе в систему и установит нужный режим доступа к создаваемым файлам и каталогам.

3.3.5. getfacl

Синтаксис:

```
getfacl [-dRLP] файл ...
```

Для каждого файла getfacl выводит имя файла, 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.

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 традиционным битам прав доступа к файлу, соответственно, для владельца, группы-владельца и всех остальных. Эти три элемента являются базовыми. Строки 5 и 7 являются элементами для отдельных пользователя и группы. Строка 8 — маска эффективных прав. Этот элемент ограничивает эффективные права, предостав-

ляемые всем группам и отдельным пользователям. Маска не влияет на права для владельца файла и всех других. Строки 10–14 показывают ACL-по умолчанию, ассоциированный с данным каталогом.

Команда `getfacl` выводит ACL файлов и каталогов по умолчанию.

Для большого количества файлов `getfacl` выводит ACL, разделенные пустыми строками. Результаты команды `getfacl` могут использоваться как входные данные для команды `setfacl` (3.3.6).

Опции приведены в таблице 5.

Таблица 5

Опция	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
`--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

Синтаксис:

```
setfacl [-bkndRLP] { -m|-M|-x|-X ... } файл ...
```

Эта команда изменяет ACL к файлам или каталогам. В командной строке за последовательностью команд идет последовательность файлов (за которой, в свою очередь, также может идти последовательность команд и т. д.).

Опции приведены в таблице 6.

Таблица 6

Опция	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=datei	Прочитать записи ACL для установления из файла
--mask	Пересчитать маску эффективных прав
-n, --no-mask	Не пересчитывать маску эффективных прав, обычно setfacl пересчитывает маску (кроме случая явного задания маски) для того, чтобы включить ее в максимальный набор прав доступа элементов, на которые воздействует маска (для всех групп и отдельных пользователей)
-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 ей подобной. Все права доступа дерева каталогов восстанавливаются, используя этот механизм. Если вводимые данные содержат элементы для владельца или группы-владельца и команда setfacl выполняется пользователем с именем root, то владелец и группа-владелец всех файлов также восстанавливаются. Эта опция не может использоваться совместно с другими опциями, за исключением опции --test
--test	Режим тестирования (ACL не изменяются)
--help	Вывести справку и выйти
--version	Вывести информацию о версии и выйти

При использовании опций --set, -m и -x должны быть перечислены записи ACL в командной строке. Элементы ACL разделяются одинарными кавычками.

При чтении ACL из файла при помощи опций --set-file, -M и -X команда setfacl принимает множество элементов в формате вывода getfacl. В строке обычно содержится не больше одного элемента ACL.

3.3.6.1. Элементы ACL

Команда `setfacl` использует следующие форматы элементов ACL:

1) `[d[efault]:] [u[ser]:]uid [:[+|^]perms]`

Права доступа отдельного пользователя. Если не задан `uid`, то права доступа владельца файла.

2) `[d[efault]:] g[roup]:gid [:[+|^]perms]`

Права доступа отдельной группы. Если не задан `gid`, то права доступа группы-владельца.

3) `[d[efault]:] m[ask]:[+|^] perms`

Маска эффективных прав.

4) `[d[efault]:] o[ther]:[+|^] perms`

Права доступа всех остальных.

Элемент ACL является абсолютным, если он содержит поле `perms` и является относительным, если он включает один из модификаторов: «+» или «^». Абсолютные элементы могут использоваться в операциях установки или модификации ACL. Относительные элементы могут использоваться только в операции модификации ACL. Права доступа для отдельных пользователей, группы, не содержащие никаких полей после значений `uid`, `gid` (поле `perms` при этом отсутствует), используются только для удаления элементов.

Значения `uid` и `gid` задаются именем или числом. Поле `perms` может быть представлено комбинацией символов `r`, `w`, `x`, `-` или цифр (0–7).

3.3.6.2. Автоматически созданные права доступа

Изначально файлы и каталоги содержат только три базовых элемента ACL: для владельца, группы-владельца и всех остальных пользователей. Существует ряд правил, которые следует выполнять:

1) не могут быть удалены сразу три базовых элемента. Должен присутствовать хотя бы один;

2) если ACL содержит права доступа для отдельного пользователя или группы, то ACL также должен содержать маску эффективных прав;

3) если ACL содержит какие-либо элементы ACL-по умолчанию, то в последнем должны также присутствовать три базовых элемента (т. е. права доступа по умолчанию для владельца, группы-владельца и всех остальных);

4) если ACL-по умолчанию содержит права доступа для отдельных пользователей или групп, то в ACL также должна присутствовать маска эффективных прав.

Для того чтобы помочь пользователю выполнять эти правила, `setfacl` создает права доступа, используя уже существующие, согласно следующим условиям:

1) если права доступа для отдельного пользователя или группы добавлены в ACL, а

маски прав не существует, то создается маска с правами доступа группы-владельца;

2) если создан элемент ACL-по умолчанию, а трех базовых элементов не было, тогда делается их копия и они добавляются в ACL-по умолчанию;

3) если ACL-по умолчанию содержит какие-либо права доступа для конкретных пользователя 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.

3.4. Дискреционное управление доступом в СУБД PostgreSQL

В качестве защищенной СУБД в составе ОС используется PostgreSQL, доработанная в соответствии с требованием интеграции с ОС в части мандатного управления доступом к информации.

СУБД PostgreSQL является объектно-реляционной. На низком уровне данные хранятся в отношениях (таблицах, видах), и доступ к данным разграничивается в понятиях реляционной СУБД.

Данные в реляционной БД хранятся в отношениях (таблицах), состоящих из строк и столбцов. При этом единицей хранения и доступа к данным является строка, состоящая из полей, идентифицируемых именами столбцов. Кроме таблиц, существуют другие объекты БД (виды, процедуры и т. п.), которые предоставляют доступ к данным, хранящимся в таблицах.

С каждым типом объектов БД ассоциируется определенный набор типов доступа (возможных операций). Для каждого объекта явно задается список разрешенных для каждого из поименованных субъектов БД (пользователей, групп или ролей) типов доступа (т. е. ACL). И в дальнейшем при разборе запроса к БД осуществляется проверка возможности предоставления доступа субъекта к объекту типа, соответствующего запросу.

В общем случае отдельная строка таблицы не является однозначно идентифицируемым объектом (каждая строка идентифицируется только набором содержимого своих полей, но без специальных действий, например создания первичного ключа или физического уникального идентификатора строки в БД, такая идентификация не является уникальной), и дискреционные правила разграничения доступа к ней применены быть не могут. В PostgreSQL объектами дискреционного управления доступом могут являться и столбцы объектов, поскольку могут быть однозначно идентифицированы по составному имени объекта и столбца, т. к. имя столбца внутри объекта является уникальным.

В рамках дискреционных ПРД определены следующие операции над таблицами и хранящимися в них данными:

- `SELECT` — чтение данных из таблицы;
- `INSERT` — вставка новых данных в таблицу;

- DELETE — удаление некоторых/всех данных в таблице;
- UPDATE — изменение данных в таблице;
- REFERENCES — использование данных таблицы для внешних ключей;
- TRIGGER — создание и назначение для таблицы триггеров;
- TRUNCATE — очистка таблицы (удаление всех данных).

Для более гибкой работы с данными в СУБД введены следующие объекты, к каждому из которых так же существует набор операций:

1) вид — способ организации предварительно подготовленных запросов. Набор операций совпадает с набором операций для таблиц, за исключением создания триггеров и внешних ключей:
   - SELECT — чтение данных из вида;
   - INSERT — вставка новых данных в вид;
   - DELETE — удаление некоторых/всех данных в виде;
   - UPDATE — изменение данных в виде;

2) последовательность — способ получения уникальных значений (счетчик). Определены следующие операции:
   - SELECT — чтение значения счетчика;
   - UPDATE — установка значения счетчика;
   - USAGE — выполнение функций манипулирования счетчиком;

3) БД — способ организации области данных, содержащих все остальные объекты СУБД. Определены следующие операции:
   - CREATE — создание БД;
   - CONNECT — установка соединения с БД;
   - TEMPORARY/TEMP — создание временных таблиц в БД;

4) функция — программный код манипулирования данными на сервере. Определена операция EXECUTE — выполнение функции;

5) язык — язык написания функций на сервере. Определена операция USAGE — использование языка для написания функций;

6) схема — способ организации объектов в пределах отдельной БД. Определены следующие операции:
   - CREATE — создание объектов в указанной схеме;
   - USAGE — использование объектов указанной схемы;

7) табличное пространство — способ организации БД в ФС ОС. Определена операция CREATE — создание объектов в указанном табличном пространстве.

8) бинарный объект — способ хранения больших двоичных объектов (файлов, документов, фотографий, и т.п.) в БД. Определены следующие операции:

- SELECT — чтение бинарного объекта;
- UPDATE — изменение бинарного объекта;

9) В БД могут присутствовать дополнительные объекты, для использования которых определена операция USAGE.

Для контроля выполнения всех перечисленных операций дискреционных ПРД существуют соответствующие права доступа. Право на предоставление прав доступа к объектам не может быть предоставлено другим пользователям и доступно только администратору БД (при соответствующих настройках сервера может быть предоставлено и владельцу объекта).

Кроме рассмотренных (делегируемых) прав доступа, существует ряд прав, которые всегда принадлежат владельцам объектов и администраторам СУБД. Эти права не могут быть делегированы или отменены средствами СУБД. К таким правам относятся: удаление и модификация объекта и назначение пользователям делегируемых прав доступа к объектам.

Сразу же после создания объекта только его владелец и администраторы СУБД могут использовать его каким-либо образом. Для того чтобы с этим объектом могли работать другие пользователи, владелец объекта или администратор СУБД должен явно предоставить им соответствующие дискреционные права доступа.

Модификация метаданных возникает каждый раз при изменении структуры БД, что включает в себя создание, модификацию и удаление объектов БД.

Разграничение доступа к перечисленным операциям на уровне СУБД так же реализуется применением дискреционных ПРД. Для этого используется право владения объектом, право на создание объектов. Право владения объектом предоставляет владельцу объекта возможность модифицировать и удалять объект. Как правило, владельцем является создатель объекта или суперпользователь (администратор БД). Право на создание (CREATE) существует к объектам БД, являющихся контейнерами для других объектов, а именно: непосредственно сама БД, схема, табличное пространство.

При выполнении любого запроса пользователя (субъекта БД) к защищаемому ресурсу (объекту БД) выполняется дискреционное управление доступом на основе установленных пользователю прав. Для каждой выполняемой операции производится проверка наличия права у пользователя на выполнение данной конкретной операции.

Дискреционные ПРД применяются после разбора запроса пользователя и построения плана его выполнения.

Дискреционные ПРД к столбцам объекта применяются только при отсутствии явного разрешения на доступ к самой таблице. Таким образом, права доступа к объекту являются доминирующими. При этом при отсутствии явно заданных прав на объект нельзя сказать определенно о предоставлении доступа до тех пор, пока не будут проверены права на столбцы объекта.

В СУБД PostgreSQL параметр конфигурации ac_enable_trusted_owner позволяет администратору запретить владельцам объектов передавать права на доступ к ним другим пользователям СУБД. В случае установки значения этой переменной конфигурации в FALSE распределение прав доступа к объектам БД разрешено только администраторам СУБД.

Параметр конфигурации ac_allow_grant_options позволяет администратору запретить передачу уже имеющихся прав доступа на объект другим ролям. Если ac_allow_grant_options установлен в FALSE, то запрещается использовать команду GRANT с привилегией WITH GRANT OPTION. Если у роли есть привилегия GRANT OPTIONS и ac_allow_grant_options = false, то передача прав доступа другим ролям также запрещается. Изъятие (REVOKE) привилегии GRANT OPTIONS разрешается всегда.

Параметр конфигурации ac_allow_admin_options позволяет администратору запретить передачу прав членства роли другим ролям. Если ac_allow_admin_options установлен в FALSE, то запрещается использовать GRANT с привилегией WITH ADMIN OPTION. Если у роли есть привилегия ADMIN OPTIONS и ac_allow_admin_options = false, то передача прав членства другим ролям также запрещается. Изъятие (REVOKE) привилегии ADMIN OPTIONS разрешается всегда.

Параметр конфигурации ac_enable_truncate позволяет администратору запретить владельцам объектов и любым пользователям, обладающим соответствующим правом TRUNCATE, выполнять удаление всех записей из таблиц. В случае установки значения этой переменной конфигурации в FALSE выполнение команды TRUNCATE запрещено всем пользователям.

3.4.1. Средства управления дискреционными ПРД к объектам БД СУБД PostgreSQL

Для управления дискреционными ПРД к объектам БД СУБД PostgreSQL используется графическая утилита pgadmin3.

Для делегирования дискреционных прав доступа к объектам используется команда SQL GRANT, а для отмены — команда REVOKE. Например, если в системе существует пользователь ivanov, то ему может быть предоставлено право на изменение данных в таблице Счета с помощью следующей команды:

```
GRANT UPDATE ON "Счета" TO ivanov
```

Для предоставления прав доступа к объекту сразу всем пользователям системы существует специальное «имя пользователя» PUBLIC, а для предоставления всех прав — специальное «право» ALL. Например, чтобы дать всем пользователям полный доступ к таблице Счета, следует использовать следующую команду:

```
GRANT ALL ON "Счета" TO PUBLIC
```

При необходимости право доступа может быть предоставлено пользователю (но не группе) с возможностью делегирования данного права другим ролям. Для этого используется ключевая фраза `WITH GRANT OPTION`:

```
GRANT UPDATE ON "Счета" TO ivanov WITH GRANT OPTION
```

Владелец объекта может отменить собственные делегируемые права, например, переведя объект в режим «только для чтения» для себя, так же как и для всех остальных пользователей.

4. МАНДАТНЫЕ УПРАВЛЕНИЕ ДОСТУПОМ И КОНТРОЛЬ ЦЕЛОСТНОСТИ

4.1. Общие сведения

Мандатное управление доступом включается автоматически при установке ОС. Порядок включения мандатного контроля целостности (МКЦ) описан в 4.3.

Механизмы мандатного управления доступом и мандатного контроля целостности реализованы в ядре ОС. При этом принятие решения о запрете или разрешении доступа субъекта к объекту принимается на основе типа операции (чтение/запись/исполнение), мандатного контекста безопасности субъекта и мандатной метки объекта.

Механизмы мандатного управления доступом и мандатного контроля целостности затрагивают следующие подсистемы:

- механизмы IPC;
- стек TCP/IP (IPv4);
- ФС Ext2/Ext3/Ext4;
- сетевые ФС CIFS;
- ФС proc, tmpfs.

Мандатная метка определяется уровнем целостности (категориями целостности) и классификационной меткой (определяется уровнем и категориями). Правила принятия решения о предоставлении доступа на основе мандатной метки описаны в 4.2.

ОС позволяет использовать 256 значений для иерархических уровней конфиденциальности (целые числа от 0 до 255) и 64 различных неиерархических категорий (представленных в виде разрядов битовой маски).

Примечание. В информационных системах с мандатным управлением доступом как правило применяются метки конфиденциальности, в которых используется только 4 уровня конфиденциальности от 0 до 3 и 64-битовая маска с различными сочетаниями категорий.

В ОС используется решетка уровней целостности (аналог решетки неиерархических категорий) в диапазоне значений от 0 до 255. Этот диапазон содержит набор несравнимых между собой уровней целостности (например, 1=0b00000001, 2=0b00000010, 4=0b00000100, 8=0b00001000,16=0b00010000,32=0b00100000,64=0b01000000,128=0b10000000), которые могут быть задействованы для системных сервисов (например, уровень целостности 8 зарезервирован для графического сервера Xorg). При установке ОС по умолчанию предлагается максимальный уровень целостности 63 (в двоичной системе 0b00111111), минимальный уровень всегда 0. Максимальными уровнями целостности в системе могут быть числа, у которых битовая маска включает битовые маски всех остальных используемых уровней целостности в системе (например, 63=0b00111111, 127=0b01111111, 191=0b10111111 и 255=0b11111111).

ВНИМАНИЕ! Устанавливать для пользователя одновременно высокий уровень конфиденциальности (классификационную метку) и высокий уровень целостности не рекомендуется.

4.2. Правила применения

Пусть контекст безопасности субъекта содержит уровень конфиденциальности `L0`, уровень целостности `iL0` и категории `C0`, а мандатная метка объекта содержит уровень конфиденциальности `L1`, уровень целостности `iL1` и категории `C1`. Операции сравнения уровней и категорий и доступа субъектов к объектам определяются следующим образом:

1) уровень `L0` меньше уровня `L1` (`L0<L1`), если численное значение `L0` меньше численного значения `L1`;

2) уровень `L0` равен уровню `L1` (`L0==L1`), если численные значения `L0` и `L1` совпадают;

3) уровень целостности `iL0` меньше уровня `iL1` (`iL0<iL1`), если все биты набора `iL0` являются подмножеством набора бит `iL1`;

4) уровень целостности `iL0` равен уровню целостности `iL1` (`iL0==iL1`), если значения `iL0` и `iL1` совпадают;

5) категории `C0` меньше категорий `C1` (`C0<C1`), если все биты набора `C0` являются подмножеством набора бит `C1`;

6) категории `C0` равны категориям `C1` (`C0==C1`), если значения `C0` и `C1` совпадают;

7) операция записи разрешена, если `L0==L1`, `iL0>=iL1` и `C0==C1`;

8) операция чтения разрешена, если `L0>=L1`, `C0>=C1`, $\forall$`iL0`, $\forall$`iL1`;

9) операция исполнения разрешена, если `L0>=L1` и `C0>=C1`, $\forall$`iL0`, $\forall$`iL1`.

В отношении атрибутов доступа действуют следующие правила наследования:

- если в сессии порождаются другие процессы, то они наследуют метку целостности и конфиденциальности;
- процесс на любом уровне целостности создает объект только с нулевым уровнем целостности. Повысить уровень целостности может только субъект с высоким уровнем целостности и с наличием привилегии `PARSEC_CAP_CHMAC`;
- процесс создает объекты только полностью наследуя свою метку конфиденциальности. Изменить ее может только привилегированный процесс с привилегией `PARSEC_CAP_CHMAC`.

В ОС предусмотрено существование структурированных объектов доступа — контейнеров (например, каталогов ФС), т.е. объектов доступа, которые могут содержать другие объекты доступа (каталоги или файлы). Метка контейнера определяет максимальную метку вложенных объектов доступа. Контейнеры и объекты имеют дополнительные атрибуты

управления доступом — тип метки.

Тип метки может использоваться для изменения правила доступа к контейнерам и объектам по следующим правилам:

- тип метки `ccnr` применяется к контейнерам и определяет, что контейнер может содержать объекты с различными классификационными метками, но не большими, чем его собственная классификационная метка. Мандатные правила для чтения контейнера игнорируются, но при чтении контейнера субъекту видны лишь объекты с классификационной меткой не большей, чем его собственная классификационная метка;
- тип метки `ccnri` применяется к контейнерам и определяет, что контейнер может содержать объекты с различными уровнями целостности, но не большими, чем его собственный уровень целостности;
- тип метки `ehole` применяется к простым объектам (файлам) с минимальной мандатной меткой и приводит к игнорированию мандатных правил разграничения доступа к ним. Предназначена 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`);
- тип метки `whole` применяется к простым объектам (файлам) с мандатной меткой и разрешает запись в них субъектам с более низкого уровня конфиденциальности.

Тип метки может быть установлен только привилегированным процессом. К самим процессам атрибут тип метки неприменим. Перечисленные типы меток для контейнеров могут использоваться совместно. Таким образом, контейнер может иметь тип `ccnr,ccnri`, что эквивалентно объединяющему типу `ccnra`. Простой объект (файл) может иметь тип либо `ehole`, либо `whole`.

Для создания в контейнере, имеющем тип `ccnr`, вложенного объекта с уровнем и категориями меньшими, чем у контейнера, необходимо обладать специальными привилегиями `PARSEC_CAP_IGNMACCAT` и `PARSEC_CAP_IGNMACLVL` (см. 4.5). Для создания в контейнере, имеющем тип `ccnri`, вложенного объекта с уровнем целостности меньшим, чем у контейнера, необходимо обладать специальной привилегией `PARSEC_CAP_IGNMACINT` (см. 4.5).

ВНИМАНИЕ! Если в загрузчике ОС указать значение параметра ядра `parsec.ccnr_relax=1`, непривилегированный пользователь сможет производить запись файлов с разным уровнем конфиденциальности в контейнер (каталог) с установленным флагом `ccnr`.

ВНИМАНИЕ! Мандатные атрибуты на корне файловой системы определяют максимальный мандатный контекст безопасности объектов. Мандатные атрибуты, устанавливае-

мые по умолчанию на корень файловой системы и ряд вложенных объектов, определены в сценарии `pdp-init-fs`, который расположен в каталоге `/usr/sbin`.

4.3. Мандатный контроль целостности

4.3.1. Включение мандатного контроля целостности на ОС

В процессе установки ОС режим МКЦ для ОС включается по умолчанию. Для системного параметра ядра в загрузчике ОС Grub устанавливается значение `max_ilev=63` — максимальный уровень целостности по умолчанию. Все процессы, начиная от `init` до утилиты графического входа в систему `fly-dm`, будут запускаться на данном уровне целостности.

ВНИМАНИЕ! Графический сервер Xorg по умолчанию работает от пользователя на выделенном уровне целостности 8.

Если режим МКЦ был выключен (см.4.3.3), то его повторное включение выполняется с помощью команды:

```
astra-mic-control enable
```

или используя графическую утилиту `fly-admin-smc` (см. электронную справку).

После включения МКЦ необходимо перезагрузить ОС.

4.3.2. Включение мандатного контроля целостности на файловой системе

Для правильного функционирования МКЦ при работе ОС необходимо установить атрибуты МКЦ на корневой файловой системе (поддерживаются файловые системы EXT2, EXT3, EXT4, XFS). Сразу после установки ОС атрибуты МКЦ на корневой файловой системе нулевые и должны быть включены администратором после завершения всех настроек ОС и установки ПО. Перед включением режима МКЦ на файловой системе необходимо проверить, что режим МКЦ включен в ОС. Для этого выполнить команду:

```
cat /proc/cmdline | grep "parsec.max_ilev"
```

В выводе результата выполнения команды ненулевое значение параметра `parsec.max_ilev` означает, что режим включен.

Включение МКЦ на файловой системе выполняется на ОС путем выполнения команды:

```
set-fs-ilev
```

или используя графическую утилиту `fly-admin-smc` (см. электронную справку).

После включения МКЦ на файловой системе появляется возможность задавать уровни целостности для отдельных процессов и файлов в соответствии с 4.4.

4.3.3. Выключение МКЦ

Выключение МКЦ на файловой системе выполняется путем выполнения команды:

```
astra-mic-control disable
```

или используя графическую утилиту `fly-admin-smc` (см. электронную справку).

Перед выключением режима МКЦ рекомендуется снять атрибуты целостности с объектов файловой системы командой:

```
unset-fs-ilev
```

или используя графическую утилиту `fly-admin-smc` (см. электронную справку).

После выключения МКЦ перезагрузить ОС.

ВНИМАНИЕ! Выключение МКЦ крайне не рекомендуется, т.к. многие механизмы защиты связаны с включенным режимом МКЦ, а именно: блокировка интерпретаторов, `nochmodx`, блокировка доступа с конфиденциальной информации и т.д.

4.3.4. Администрирование ОС при включенном режиме МКЦ

Непривилегированный пользователь может выполнять вход в систему только на низком уровне целостности (соответствует минимальному уровню целостности). Привилегированный пользователь, при наличии соответствующего права, может входить в систему на высоком уровне целостности (соответствует максимальному уровню целостности ОС) и только для выполнения задач по конфигурированию ОС.

Любая настройка ОС при включенном режиме МКЦ выполняется при входе в систему на высоком уровне целостности. Обычный режим работы осуществляется на низком уровне целостности.

Администратор, созданный при установке ОС, может выполнять вход в систему с высоким уровнем целостности (по умолчанию 63) или с низким уровнем целостности.

При графическом входе в систему для такого администратора по умолчанию выбран высокий уровень целостности. Графический рабочий стол на высоком уровне целостности имеет красный фон.

При консольном входе в систему администратор должен вручную выставлять уровень контроля целостности (для высокого уровня — 63, для низкого — 0 или пропустить данный шаг).

4.4. Запуск служб systemd с уровнем целостности и конфиденциальности

Для запуска службы systemd под уровнем конфиденциальности необходимо в конфигурационном файле (юните) соответствующей службы (`<имя_юнита>.service`) в разделе `[Service]` добавить следующий параметр:

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

Формат метки `PDPLabel` аналогичен принятому в системе PARSEC за исключением поля типа метки — метка службы не может иметь флагов (наличие флагов `ccnr/ccnri` и `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 имён могут оказаться задействованы сетевые ресурсы, например, LDAP-каталоги, что может привести к ошибкам конфигурации, которые сложно диагностировать.

Для назначения службе PARSEC-привилегий, приведенных в 4.5, в соответствующем юните (<имя_юнита>.service) в разделе [Service] добавить параметр:

```
[Service]
CapabilityParsec=PARSEC_CAP_PRIV_SOCK,...
```

где через запятую 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> службы используется команда:

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

4.5. PARSEC-привилегии

PARSEC-привилегии также, как и Linux-привилегии (см. 3.2), наследуются процессами от своих «родителей». Процессы, запущенные от имени суперпользователя, независимо от наличия у них привилегий, имеют возможность осуществлять все перечисленные привилегированные действия.

PARSEC-привилегии приведены в таблице 7.

Таблица 7

Привилегия	Атрибут	Битовая маска	Описание
PARSEC_CAP_SIG	pcapsig	0x40:0x0	Посылать сигналы процессам, игнорируя дискреционные и мандатные права
PARSEC_CAP_SETMAC	pcapsetmac	0x4:0x0	Изменить мандатную метку и установить другие привилегии
PARSEC_CAP_CHMAC	pcapchmac	0x8:0x0	Менять мандатные метки файлов
PARSEC_CAP_AUDIT	pcapaudit	0x2:0x0	Управление политикой аудита

Окончание таблицы 7

Привилегия	Атрибут	Битовая маска	Описание
PARSEC_CAP_READSEARCH	pcapreadsearch	0x200:0x0	Игнорировать мандатную политику при чтении и поиске файлов (но не при записи)
PARSEC_CAP_PRIV_SOCK	pcapprivsock	0x100:0x0	Создавать привилегированный сокет и менять его мандатную метку. Привилегированный сокет позволяет осуществлять сетевое взаимодействие, игнорируя мандатную политику
PARSEC_CAP_UPDATE_ATIME	pcapupdateatime	0x80:0x0	Изменять время доступа к файлу
PARSEC_CAP_IGNMACLVL	pcapignmaclvl	0x10:0x0	Игнорировать мандатную политику по уровням
PARSEC_CAP_IGNMACCAT	pcapignmaccat	0x20:0x0	Игнорировать мандатную политику по категориям
PARSEC_CAP_FILE_CAP	pcapfilecap	0x1:0x0	Устанавливать привилегии на файлы
PARSEC_CAP_CAP	pcapcap	0x400:0x0	Устанавливать любой непротиворечивый набор привилегий для другого процесса
PARSEC_CAP_MAC_SOCK	pcapmacsock	0x800:0x0	Смена мандатной точки соединения
PARSEC_CAP_UNSAFE_SETXATTR	pcapunsafesetxat	0x1000:0x0	Устанавливать мандатные атрибуты объектов ФС без учета мандатных атрибутов родительского объекта-контейнера. Привилегия используется для восстановления объектов ФС из резервных копий (см. 10.2) и только после установки значения 1 для параметра /parsecfs/unsecure_setxattr
PARSEC_CAP_IGNMACINT	pcapignmacint	0x2000:0x0	Игнорировать мандатную политику по уровням целостности
PARSEC_CAP_SUMAC	pcapsumac	0x4000:0x0	Запускать процессы под другим уровнем конфиденциальности

Для настройки КСЗ могут использоваться как PARSEC-, так и Linux-привилегии. Порядок управления привилегиями описан в 4.9.

4.6. Сетевое взаимодействие

В качестве основной сетевой ФС используется CIFS, которая является расширением SMB и поддерживает атрибуты ФС UNIX и имеет ограниченную поддержку расширенных атрибутов. Данная ФС широко распространена и работает в гетерогенных сетях (поддерживается многими ОС), а также поддерживает аутентификацию средствами PAM и Kerberos.

Взаимодействие при помощи сетевого протокола IPv4 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 контроля доступа в сетевые пакеты протокола IPv4 внедряются классификационные метки. Порядок присвоения классификационных меток и их формат соответствует национальному стандарту ГОСТ Р «Защита информации. Управление потоками информации в информационной системе. Формат классификационных меток». Прием сетевых пакетов подчиняется мандатным ПРД. Следует отметить, что метка сокета может иметь тип, позволяющий создавать сетевые сервисы, 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 метки на объекте доступа эквивалентно нулевой мандатной метке. Таким образом, ядро ОС, в которой все объекты и субъекты доступа имеют уровень секретности «несекретно», функционирует аналогично стандартному ядру ОС Linux.

Для ряда сетевых сервисов (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 сервиса. Для предоставления названной возможности в подсистеме безопасности PARSEC реализован механизм запуска сетевых сервисов с использованием привилегированного сокета для ожидания входящих соединений (механизм `privsock`), описанный в 4.6.1.

4.6.1. Механизм privsock

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

Для его использования при функционировании сетевого сервиса необходимо от-

редактировать файл /etc/parsec/privsock.conf, добавив в него строку, содержащую полный путь к исполняемому файлу сервиса. 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 сервиса, 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 в man privsock.

4.7. 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 мандатной меткой пользователя системы и все взаимодействующие через нее процессы также имеют данную мандатную метку. Взаимодействие процессов с иной мандатной меткой 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 демона D-Bus и через нее взаимодействуют процессы различных

уровней.

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

4.7.1. 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 сервиса, то сообщение направляется только определенному соединению (т. е. первичному владельцу данного сервиса).

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.

4.7.2. Процесс взаимодействия с шиной

Для идентификации соединения на шине используются параметры, приведенные в таблице 8.

Таблица 8

Параметр	Описание
unique connection name (UCN)	Уникальное имя соединения — идентификатор соединения
org.share.server	Общеизвестное имя (сервис) 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. Каждый сервис 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 8

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.

4.7.3. Процесс соединения с системной шиной

После аутентификации клиент отправляет сообщение типа `method_call Hello` серверу D-Bus. Далее проверяется возможность доставки данного сообщения с использование `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 сервис `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 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 сервиса. Под сервисом 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), то 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 (сервису) 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. Относится к данному сервису — если другое соединение потребует заменить владельца, то этот флаг разрешит выполнить смену с учетом меток и целостности;
- заменить владельца. Если разрешена замена, то он будет заменен у ранее созданного сервиса;
- 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 сервиса формируется сигнал `NameOwnerChanged`. После освобождения имени формируется сигнал `NameLost`.

4.7.4. Объекты и субъекты системы

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

Проверка `pdp` доступа осуществляется при следующих событиях:

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

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

4.7.5. Алгоритм проверки меток для различных сообщений и режимов работы

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

- сообщения между сервисами и соединениями;
- сообщения, адресованные демону D-Bus. Здесь представлены сервисные сообщения, которые позволяют получать информацию о других сервисах и соединениях на шине. Т. е. в параметрах сообщения может быть указан заданный сервис, информацию о котором необходимо получить.

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

Для сообщений, адресованных демону 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 заданному сервису.

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

Для получения владения сервисом 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, отличной от метки сервиса. Если параметр pdplinux_allow_different_owners установлен, то возможно получение владения сервисом.

Для непосредственной проверки возможности замены владельца сервиса модифицирована функция bus_service_get_allow_replacement(), которая выполняет проверку целостности и не позволяет проводить соответствующую замену в случае, если целостность сервиса выше 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 источника сообщения и соответствующего сервиса, совпадающего с его текущим владельцем. Если метки 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 сервис, метод которого запускается, имеет привилегии 0x30, то запуск разрешается. Также существует возможность запуска метода, если соединение получатель имеет метку 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().

4.7.6. 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
```

4.7.7. Расширенное управление политиками

4.7.7.1. 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 сервисами 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 сервиса) на шине. В результате для определения политики достаточно будет создать <allow>-правила для разрешений типа own, чтобы позволить процессам создавать собственные сервисы, и добавить параметр send_destination, чтобы разрешить трафик от некоторых или всех пользователей к данным сервисам.

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 9. Параметры правил <deny> определяют необходимость запрета указанного в параметрах сообщения.

Таблица 9

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 данному сервису. То есть, если соединение владеет сервисами A, B, C, и отправка к A запрещена, то отправка к B или C также будет запрещена
send_type	Возможные значения: method_call, method_return, signal и error. Задает тип исходящего сообщения
send_path	Путь вида /path/name отправителя сообщения

Продолжение таблицы 9

Параметр	Описание
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 данным сервисом
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 9

Параметр	Описание
eavesdrop	Возможные значения: true, false. Прослушивание (Eavesdropping) возникает, когда приложение получает сообщение, адресованное сервису, которым 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 сервисы, например, Telepathy и ReserveDevice, определяют значения сервисов 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

Не имеет смысла запрещать пользователя или группу внутри секции <policy> — запрет пользователя/группы может быть только в политике context="default" или context="mandatory".

Отдельное правило <deny> может содержать комбинации параметров. В данном случае запрет выполняется только тогда, когда все параметры соответствуют сообщению. Например, правило <deny send_interface="foo.bar" send_destination="foo.blah"/> запрещает сообщения с интерфейсом foo.bar и указанным именем сервиса foo.blah.

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 не является обязательным. В частности, нельзя указывать <deny send_interface="org.foo.Bar"/> — данное правило приведет к тому, что сообщения без интерфейса будут заблокированы для всех сервисов. Всегда необходимо использовать правило в следующей форме:

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

Примеры:

1. <deny send_destination="org.freedesktop.Service"
send_interface="org.freedesktop.System" send_member="Reboot"/>
2. <deny send_destination="org.freedesktop.System"/>
3. <deny receive_sender="org.freedesktop.System"/>
4. <deny user="john"/>
5. <deny group="enemies"/>

4.7.7.2. 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*`.

4.8. Средства управления мандатными ПРД

Для управления мандатными ПРД используются следующие графические утилиты:

- `fly-fm` («Менеджер файлов») — управление мандатными атрибутами файлов;
- `fly-admin-smc` («Управление политикой безопасности») — управление протоколированием, привилегиями и мандатными атрибутами пользователей, работа с пользователями и группами.

Более подробное описание утилит см. в электронной справке.

Для управления мандатными ПРД в режиме командной строки используются следующие утилиты:

- `pdpl-file` — управление мандатными атрибутами файлов (см. 4.8.1);
- `pdp-id` — отображение мандатных атрибутов сессии пользователя ОС (см. 4.8.2);
- `pdp-init-fs` — скрипт инициализации мандатных атрибутов ФС (см. 4.8.3);
- `pdp-ls` — вывод аналогично стандартной команде `ls` информации о файлах с отображением мандатных атрибутов (см. 4.8.4);
- `pdpl-ps` — управление мандатными атрибутами процессов (см. 4.8.5);
- `pdpl-user` — управление допустимыми мандатными уровнями и категориями пользователей ОС (см. 4.8.6);
- `sumac` — запуск процесса с заданными мандатными уровнем и категорией в отдельной графической сессии (см. 4.8.7);
- `userlev` — изменение БД мандатных уровней (см. 4.8.8);
- `usercat` — изменение БД мандатных категорий (см. 4.8.9).

Для совместимости с предыдущими версиями ОС сохранены следующие утилиты командной строки для управления мандатными ПРД:

- `chmac` — управление мандатными атрибутами файлов (см. 4.8.10.1);
- `lsm` — вывод аналогично стандартной команде `ls` информации о файлах с отображением мандатных атрибутов (см. 4.8.10.3);
- `macid` — отображение мандатных атрибутов сессии пользователя ОС (см. 4.8.10.2);
- `psmac` — управление мандатными атрибутами процессов (см. 4.8.10.4);
- `usermac` — управление допустимыми мандатными уровнями и категориями пользователей ОС (см. 4.8.10.5);
- `getfmac` — получение мандатных меток файловых объектов (см. 4.8.10.6);
- `setfmac` — изменение мандатных меток файловых объектов (см. 4.8.10.7).

4.8.1. pdpl-file

Синтаксис:

```
pdpl-file [опции]...[уровень][:уровень целостности[:категория[:флаги]]] [файл]
```

Команда `pdpl-file` изменяет мандатные атрибуты файлов ОС, которые включают мандатную метку и специальные мандатные атрибуты файла.

Опции приведены в таблице 10.

Таблица 10

Опция	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
`-R`, `--recursive`	Применить рекурсивно
`-h`, `--help`	Вывести справку и выйти
`--version`	Вывести информацию о версии и выйти

Уровень и категория могут быть заданы именем или шестнадцатеричным значением.

Пример

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

Данная команда рекурсивно для всех файлов каталога `/tmp` изменит уровень на `Секретно` и категорию на `Категория_А` (уровень и категория должны быть определены в системе).

Флаги `ccnr`, `ccnri`, `ehole` и `whole` (см. 4.2) могут быть заданы значением или именами через запятую. Для сочетания `ccnr` и `ccnri` флаги могут быть заданы псевдонимом `CCNRA`.

Пример

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

4.8.2. pdp-id

Синтаксис:

```
pdp-id [опции]
```

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

Опции приведены в таблице 11.

Таблица 11

Опция	Описание
`-l`,`--level`	Вывести только мандатный уровень конфиденциальности

Окончание таблицы 11

Опция	Описание
`-i,--ilevel`	Вывести только мандатный уровень целостности
`-c,--categories`	Вывести только мандатную категории
`-r,--categories`	Вывести только роли
`-n,--name`	Для опций `-lc` выводить имена вместо числовых значений
`-h,--help`	Вывести справку и выйти
`--version`	Вывести информацию о версии и выйти

При отсутствии опций выводит строку текущих мандатных свойств.

Пример

```
pdp-id
Уровень конф: Секретно Уровень целостности: 0 Категории: Категория_А
```

В этом примере текущая сессия пользователя имеет уровень секретности `Секретно`, минимальный уровень целостности и категорию `Категория_А`.

4.8.3. pdp-init-fs

Синтаксис:

```
pdp-init-fs
```

Скрипт 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), начина с корня ФС.

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

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

4.8.4. pdp-ls

Синтаксис:

```
pdp-ls [опции] [имя файла]
```

Команда `pdp-ls` выводит аналогично стандартной команде `ls` информацию о файлах (по умолчанию — о текущем каталоге).

Использование данной команды в целом не отличается от использования `pdp-ls`, за исключением следующих особенностей:

- если на файле установлены ACL, то к ним добавляется символ «+»;
- если на файле установлена ненулевая мандатная метка, то к ACL добавляется символ «m»;

- если на файле установлены списки протоколирования, то к ACL добавляется символ «a»;

- доступна опция `-M`, которая может быть использована для просмотра мандатных меток на файловых объектах.

4.8.5. pdpl-ps

Синтаксис:

```
pdpl-ps [-nzhv] <идентификатор процесса>
```

Команда `pdpl-ps` позволяет считать мандатный контекст безопасности с процесса, заданного параметром (идентификатор процесса).

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

Опции приведены в таблице 12.

Таблица 12

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

4.8.6. pdpl-user

Синтаксис:

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

Команда `pdpl-user` отображает и устанавливает допустимые мандатные уровни и категории пользователей ОС.

Опции приведены в таблице 13.

Таблица 13

Опция	Описание
`-d, --delete`	Удалить строку пользователя из файла
`-z, --zero`	Обнулить значения уровней и категорий
`-l, --levels`	Установить допустимые уровни конфиденциальности

Окончание таблицы 13

Опция	Описание
-i, --ilevel	Установить максимальный уровень целостности
-m, --maclabels	то же, что и -l (используется для совместимости)
-c, --category	Установить допустимые категории
-h, --help	Вывести справку и выйти
-v, --version	Вывести информацию о версии и выйти

Если в качестве параметра ключей -m или -c указано одно значение или одно значение с предшествующим двоеточием, это значение интерпретируется как максимальное значение, если одно значение с последующим двоеточием — как минимальное.

Команда pdpl-user при успешном выполнении всегда выводит значения установленных допустимых мандатных меток.

Чтобы просмотреть текущие допустимые метки, выполнить команду без ключей:

```
pdpl-user пользователь
```

Пример

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

Данная команда для пользователя user1 установит:

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

Уровни Уровень_0, Уровень_3, категория Категория_2 должны быть определены в системе. Значения уровней и категорий могут быть заданы в числовой форме.

4.8.7. sumac

Синтаксис:

```
sumac [-h, --help] [-v, --version] [-l, --level=] [-c, --category=]
  [-i, --stdin=] [-o, --stdout=] [-e, --stderr=] [-x, --xauth] [command]
```

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

ВНИМАНИЕ! Для запуска программ на уровне, отличном от уровня текущей сессии, не должна стоять блокировка использования команды sumac в графической утилите fly-admin-smc (см. электронную справку) и пользователю должна быть назначена приви-

легия PARSEC_CAP_SUMAC (см. 4.5).

Если указанный мандатный уровень выше текущего, т. е. происходит увеличение уровня, то переменные окружения наследуются от текущего процесса. Если происходит уменьшение мандатного уровня, то текущие переменные окружения сбрасываются, чтобы избежать утечки информации. Аналогично при порождении нового процесса закрываются все файловые дескрипторы, мандатная метка которых не совпадает с указанной в командной строке. В том числе закрываются stdin, stdout, stderr. Перенаправить стандартный ввод и вывод для нового процесса можно с помощью опций -i, -o, -e для stdin, stdout и stderr, соответственно.

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

Пример

Запуск графического приложения xterm с мандатным уровнем 2 и категорией 0xffff

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

Опции приведены в таблице 14.

Таблица 14

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

4.8.8. userlev

Синтаксис:

```
userlev [-d, --delete] [-a, --add<значение>] [-r, --rename<имя>]
  [-m, --modify<значение>] [-h, --help] [--version] <уровень>
```

Команда userlev служит для просмотра и изменения в БД мандатных уровней. Для просмотра всех уровней команду следует запускать без параметров. Вносить изменения в БД уровней может только администратор.

Опции приведены в таблице 15.

Таблица 15

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

4.8.9. usercat

Синтаксис:

```
usercat [-d, --delete] [-a, --add<значение>] [-r, --rename<имя>]
  [-m, --modify<значение>] [-h, --help] [--version] [категория]
```

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

Опции приведены в таблице 16.

Таблица 16

Опция	Описание
-d, --delete	Удалить категорию из БД
-a, --add<значение>	Добавить новую категорию в БД
-r, --rename<новое имя>	Переименовать существующую категорию
-m, --modify<новое значение>	Изменить значение категории
-h, --help	Вывести справку и выйти
--version	Вывести информацию о версии и выйти

4.8.10. Устаревшие утилиты управления мандатными ПРД

Для совместимости с предыдущими версиями ОС сохранен ряд утилит командной строки для управления мандатными ПРД.

ВНИМАНИЕ! Настоятельно не рекомендуется применять устаревшие утилиты. Данные утилиты не позволяют работать с уровнями целостности и отображают мандатные атрибуты в формате предыдущих версий ОС.

4.8.10.1. chmac

Синтаксис:

```
chmac [опции] [уровень][:категория[:специальные атрибуты]] [имя файла]
```

Команда chmac изменяет мандатные атрибуты файлов ОС, которые включают мандатную метку и специальные мандатные атрибуты файла.

Опции приведены в таблице 17.

Таблица 17

Опция	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
-R, --recursive	Применить рекурсивно
-h, --help	Вывести справку и выйти
--version	Вывести информацию о версии и выйти

Уровень и категория могут быть заданы именем или шестнадцатеричным значением.

Пример

```
chmac -Rv Секретно:Категория_A /tmp
```

Данная команда рекурсивно для всех файлов каталога /tmp изменит уровень на Секретно и категорию на Категория_A (уровень и категория должны быть определены в системе).

Специальные атрибуты могут быть заданы значением или строкой символов rwxrwx, в которой любой из символов может быть заменен на «-» для снятия соответствующего атрибута.

Пример

```
chmac 0:0:rwxrwx /tmp
```

Данная команда для каталога /tmp установит игнорирование мандатных уровней и категорий при выполнении операций чтения, записи и исполнения.

При задании специальных атрибутов вместо rwx могут быть использованы следующие сокращения:

- equ — игнорирование мандатных уровней и категорий при выполнении операций чтения, записи и исполнения;
- equ_w — игнорирование мандатных уровней и категорий при выполнении операции записи;
- low — игнорирование мандатных уровней и категорий при выполнении операций чтения и исполнения.

ВНИМАНИЕ! В настоящее время используется другой набор специальных атрибутов. Данная утилиты не позволяет работать с ними. Существует соответствие между старым атрибутом `equ` (`rwxrwx`) и новым `ehole`. Таким образом, раздельное управление специальными атрибутами вида `rwxrwx` не предусмотрено.

Пример

```
chmac 0:0:equ /tmp
```

4.8.10.2. macid

Синтаксис:

```
macid [опции]
```

Команда `macid` выводит мандатные атрибуты сессии пользователя ОС.

Опции приведены в таблице 18.

Таблица 18

Опция	Описание
`-l,--level`	Вывести только мандатный уровень
`-c,--categories`	Вывести только мандатную категорию
`-n,--name`	Для опций `-lc` выводить имена вместо числовых значений
`-h,--help`	Вывести справку и выйти
`--version`	Вывести информацию о версии и выйти

При отсутствии опций выводит строку текущих мандатных свойств.

Пример

```
macid
Уровень=2(Секретно) Категория=1(Категория_А) Привилегии=0
```

В этом примере текущая сессия пользователя имеет уровень секретности `Секретно` и категорию `Категория_А`.

4.8.10.3. lsm

Синтаксис:

```
lsm [опции] [имя файла]
```

Команда `lsm` выводит аналогично стандартной команде `ls` информацию о файлах (по умолчанию — о текущем каталоге).

Использование данной команды в целом не отличается от использования `ls`, за исключением следующих особенностей:

- если на файле установлены ACL, то к ним добавляется символ «+»;
- если на файле установлена ненулевая мандатная метка, то к ACL добавляется символ «m»;

- если на файле установлены списки протоколирования, то к ACL добавляется символ «a»;

- доступна опция -M, которая может быть использована для просмотра мандатных меток на файловых объектах.

4.8.10.4. psmac

Синтаксис:

```
psmac [-d, --delete] [-n, --numeric] [-h, --help] [--version]
  <идентификатор процесса> [мандатная метка...]
```

Команда psmac позволяет изменять или считать мандатный контекст безопасности выбранного процесса. Если в качестве аргумента не указана метка, то команда считывает мандатный контекст с процесса, заданного параметром (идентификатор процесса).

Если аргумент-метка присутствует, то команда устанавливает заданную мандатную метку на процесс. Мандатная метка задается в виде:

<Уровень>[:<Категории>]

где <Уровень> и <Категории> могут быть заданы как в численном, так и в символьном виде. Сложные <Категории> могут быть заданы в виде списка своих составляющих.

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

Опции приведены в таблице 19.

Таблица 19

Опция	Описание
-d, --delete	Обнулить мандатный контекст безопасности процесса
-n, --numeric	Вывести информацию о контексте в численном виде
-h, --help	Вывести справку и выйти
--version	Вывести информацию о версии и выйти

4.8.10.5. usermac

Синтаксис:

```
usermac [-dzhv [-m минимальный уровень:максимальный уровень]
  [-c минимальная категория:максимальная категория]] пользователь
```

Команда usermac изменяет допустимые мандатные уровни и категории пользователей ОС.

Опции приведены в таблице 20.

Таблица 20

Опция	Описание
-d, --delete	Удалить строку пользователя из файла
-z, --zero	Обнулить значения уровней и категорий
-l, --levels	Установить допустимые мандатные уровни конфиденциальности
-i, --ilevel	Установить максимальный уровень целостности
-m, --maclabels	то же, что и -l (используется для совместимости)
-c, --category	Установить допустимые категории
-h, --help	Вывести справку и выйти
-v, --version	Вывести информацию о версии и выйти

Если в качестве параметра ключей -m или -c указано одно значение или одно значение с предшествующим двоеточием, это значение интерпретируется как максимальное значение, если одно значение с последующим двоеточием — как минимальное.

Команда usermac при успешном выполнении всегда выводит значения установленных допустимых мандатных меток.

Чтобы просмотреть текущие допустимые метки, выполнить команду без ключей:

```
usermac пользователь
```

Пример

```
usermac -m несекретно:секретно -c категория_A:категория_B user1
```

Данная команда для пользователя user1 установит:

- минимальный уровень — несекретно;
- максимальный уровень — секретно;
- минимальную категорию — категория_A;
- максимальную категорию — категория_B.

Уровни несекретно, секретно и категории категория_A, категория_B должны быть определены в системе. Значения уровней и категорий могут быть заданы в числовой форме.

4.8.10.6. getfmac

Синтаксис:

```
getfmac [-R, --recursive] [-L, --logical] [-P, --physycal] [-n, --numeric]
  [-p, --absolute-names] [-c, --omit-header] [-s, --skip-empty]
  [-h, --help] [-v, --version] файлы и/или каталоги
```

Команда getfmac служит для получения мандатных меток файловых объектов. Информация о метках посылается на стандартный вывод и может являться входными данными для команды setfmac (см. 4.8.10.7).

Опции приведены в таблице 21.

Таблица 21

Опция	Описание
-R, --recursive	Для подкаталогов рекурсивно
-L, --logical	Следовать по символическим ссылкам
-P, --phisical	Не следовать по символическим ссылкам
-n, --numeric	Выводить информацию о компонентах метки в цифровой форме
-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

4.8.10.7. setfmac

Синтаксис:

```
setfmac [-s, --set] [-m, --modify] [-S, --set-file] [-B, --restore]
  [-R, --recursive] [-L, --logical] [-P, --physical] [-h, --help]
  [-v, --version] [мандатная метка] Файлы и/или каталоги
```

Команда setfmac устанавливает мандатные метки на файлы. Метки задаются или в командной строке (параметры -s, -m), или в файле (параметры -S, -B). При этом, файлы могут быть сформированы с помощью перенаправления вывода команды getfmac (4.8.10.6).

Мандатная метка задается в виде:

```
<Уровень>[:<Категории>[:<Тип>]]
```

где <Уровень> и <Категории> могут быть заданы как в численном, так и в символьном виде. Сложные <Категории> могут быть заданы в виде списка своих составляющих, например: Танки, Самолеты.

Только администратор может устанавливать мандатные метки на файлы.

ВНИМАНИЕ! С помощью данной команды невозможно установить мандатную метку на файловый объект-сокет.

Опции приведены в таблице 22.

Таблица 22

Опция	Описание
-s, --set	Установить мандатную метку из командной строки
-m, --modify	Изменить мандатную метку из командной строки

Окончание таблицы 22

Опция	Описание
`-S, --set-file`	Установить метки из файла
`-B, --restore`	Восстановить метки из файла
`-R, --recursive`	Для подкаталогов 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

4.9. Средства управления привилегиями пользователей и процессов

Для управления привилегиями пользователей и процессов используется графическая утилита `fly-admin-smc` («Управление политикой безопасности»). Более подробное описание утилиты см. в электронной справке.

Также для управления привилегиями пользователей и процессов используются инструменты командной строки `usercaps`, `execaps` и `pscaps`, описанные, соответственно, в 4.9.1, 4.9.2 и 4.9.3.

ВНИМАНИЕ! Управление привилегиями пользователей выполняется только суперпользователем с максимальным уровнем целостности, установленным в ОС.

4.9.1. usercaps

Синтаксис:

```
usercaps [-d, --delete] [-z, --zero] [-f, --full] [-l, --linux] [-m, --parsec]
  [-l, --Linux] [-M, --PARSEC] [-n, --numeric] [-h, --help] [-v, --version]
  пользователь
```

Команда `usercaps` позволяет просматривать и устанавливать привилегии пользователей, должна использоваться только администратором через механизм `sudo`.

```
usercaps [-[l строка модифицирования linux-привилегий]
  [-[m строка модифицирования PARSEC-привилегий][dzLMnfhv]] пользователь
```

Строка модифицирования привилегий состоит из слов, разделенных «,» или «:», каждое слово может быть строкой в верхнем или нижнем регистре или числом, перед которым стоит знак «+» или «-». Список возможных привилегий с сокращенными именами и числами можно увидеть, введя для Linux-привилегий:

```
usercaps -L
```

для PARSEC-привилегий:

```
usercaps -M
```

Верхний регистр для слов и знак «+» для чисел устанавливает флаг для соответствующей привилегии, нижний регистр и знак «-» снимает его.

`usercaps <пользователь>` выводит все привилегии пользователя; `usercaps -L <пользователь>` или `usercaps -M <пользователь>` выводят только Linux- или только PARSEC-привилегии.

Опции приведены в таблице 23.

Таблица 23

Опция	Описание
`-d, --delete`	Удалить строку пользователя из файла привилегий
`-z, --zero`	Сбросить все привилегии пользователя
`-f, --full`	Присвоить все возможные привилегии пользователю
`-l, --linux`	Изменить Linux-привилегии пользователя
`-m, --parsec`	Изменить PARSEC-привилегии пользователя
`-L, --Linux`	Вывести список возможный Linux-привилегий
`-M, --PARSEC`	Вывести список возможных PARSEC-привилегий
`-n, --numeric`	Вывести привилегии в шестнадцатеричном формате
`-h, --help`	Вывести справку и выйти
`-v, --version`	Вывести информацию о версии и выйти

4.9.2. execaps

Синтаксис:

```
execaps [-v, --version] [-h, --help] [-c,--capability (привилегии)] [--]
  команда
```

Команда `execaps` может быть использована администратором для запуска процесса с одновременной установкой выбранных PARSEC-привилегий.

```
execaps -c <вектор привилегий> -- <программа и ее аргументы>
```

Привилегии задаются в виде числа — битовой маски (как правило, в шестнадцатеричном виде). Соответствие отдельных бит полномочиям приведено в `man parsec_capset(2)`.

Пример

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

Будет выполнен перезапуск сервиса `dbus` с установленной привилегией `PARSEC_CAP_PRIV_SOCK`.

Опции приведены в таблице 24.

Таблица 24

Опция	Описание
`-c, --capability`	Установить привилегии

Окончание таблицы 24

Опция	Описание
`-f, --force`	Вызвать программу, даже если не удалось установить привилегии
`-h, --help`	Вывести справку и выйти
`-v, --version`	Вывести информацию о версии и выйти

4.9.3. pscaps

Синтаксис:

```
pscaps [--version] [-h, --help] [действующие полномочия
  [разрешенные полномочия [наследуемые полномочия]]]
```

Команда `pscaps` может быть использована для просмотра и изменения (в численном виде) PARSEC-полномочий процесса.

```
pscaps [effective_caps [permitted_caps [inheritable_caps]]]
```

Если в качестве аргумента указан только идентификатор процесса, то команда показывает набор полномочий заданного процесса, в противном случае пытается установить заданные в командной строке в виде шестнадцатеричных чисел полномочия.

Опции приведены в таблице 25.

Таблица 25

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

4.10. Мандатное управление доступом в СУБД PostgreSQL

В качестве защищенной СУБД в составе ОС используется СУБД PostgreSQL версии 9.6, доработанная в соответствии с требованием интеграции с ОС в части мандатного управления доступом к информации и содержащая реализацию ДП-модели управления доступом и информационными потоками. Данная ДП-модель описывает все аспекты дискреционного, мандатного и ролевого управления доступом с учетом безопасности информационных потоков.

В основе механизма мандатного управления доступом лежит управление доступом к защищаемым ресурсам БД на основе иерархических и неиерархических меток доступа. Это позволяет реализовать многоуровневую защиту с обеспечением разграничения доступа пользователей к защищаемым ресурсам БД и управление потоками информации. В качестве иерархических и неиерархических меток доступа при использовании СУБД в ОС используются метки конфиденциальности или метки безопасности ОС.

СУБД PostgreSQL не имеет собственного механизма назначения, хранения и моди-

фикации меток пользователей и использует для этого механизмы ОС.

Согласно ДП-модели в части реализации мандатного управления доступом дополнительно к мандатной метке конфиденциальности вводится понятие объектов-контейнеров (объектов, которые могут содержать другие объекты). Для задания способа доступа к объектам внутри контейнеров используется мандатный признак CCR (Container Clearance Required). В случае когда он установлен, доступ к контейнеру и его содержимому определяется его мандатной меткой конфиденциальности, в противном случае доступ к содержимому разрешен без учета уровня конфиденциальности контейнера.

В качестве главного контейнера выбрано табличное пространство `pg_global`, которое создается одно на кластер базы данных. Таким образом, кластер является совокупностью ролей, баз данных и табличных пространств.

ВНИМАНИЕ! ДП-модель накладывает ограничение на мандатную метку конфиденциальности объекта: метка объекта не может превышать метку контейнера, в котором он содержится. Таким образом для назначения меток данным, сначала должны быть последовательно заданы максимальные метки соответствующих контейнеров: кластера, базы данных, табличного пространства, схемы и таблицы.

В реляционной модели в качестве структуры, обладающей меткой, естественно выбрать кортеж, поскольку именно на этом уровне детализации осуществляются операции чтения/записи информации в СУБД. При этом, местом хранения метки может быть выбран только сам кортеж, только так метка будет неразрывно связана с данными, содержащимися в кортеже.

В качестве множества сущностей (объектов и контейнеров) с заданной на нем иерархической структурой рассматриваются носящие подобный характер объекты реляционных баз данных, применяемые в СУБД PostgreSQL. При этом, поскольку записи базы данных содержат в своем составе мандатный уровень конфиденциальности — они рассматриваются в модели в качестве объектов, а содержащие их таблицы, соответственно, — в качестве контейнеров.

Системный каталог (метаданные) рассматривается как самостоятельная БД, реализованная с помощью средств СУБД. При этом все операции с этой БД осуществляются либо с помощью специальных конструкций языка запросов SQL или привилегированным пользователем в специальном режиме. Таким образом мандатное управление доступом применяется ко всем объектам БД. Метки системных объектов располагаются в записях таблиц системного каталога, непосредственно описывающих защищаемый объект.

Так как мандатный контроль доступа может быть определен только для видов доступа на чтение и на запись информации, все множество операций с данными в защищаемых объектах приводится к ним следующим образом:

- INSERT — доступ на запись;
- UPDATE, DELETE — последовательное выполнение доступа на чтение и запись информации;
- SELECT — доступ на чтение.

При обращении пользователя к БД определяются его допустимый диапазон меток и набор специальных мандатных атрибутов. Если пользователю не присвоена метка, то он получает по умолчанию нулевую метку, соответствующую минимальному уровню доступа. Максимальная метка определяется по заданной при регистрации пользователя в ОС. Поскольку сервер БД так же может иметь метку, при превышении метки пользователя метки сервера ему будут разрешены только операции чтения. Текущая метка пользователя определяется по установленному соединению и может быть установлена в пределах его допустимого диапазона мандатных атрибутов при наличии соответствующей привилегии.

Применение мандатных ПРД осуществляется на уровне доступа к объектам БД и на уровне доступа непосредственно к данным (на уровне записей).

Проверка мандатных прав доступа к объектам осуществляется одновременно с проверкой дискреционных прав доступа к ним, после разбора и построения плана запроса, непосредственно перед его выполнением, когда определены все необходимые для проверки данные и проверяемые объекты. Таким образом, доступ предоставляется только при одновременном санкционировании дискреционными ПРД.

Проверка мандатных прав доступа к записям таблиц осуществляется в процессе выполнения запроса при последовательном или индексном сканировании данных.

Все записи, помещаемые в таблицы, для которых установлена защита на уровне записей, наследуют текущую метку пользователя. Обновляемые записи сохраняют свою метку при изменении. Доступ к существующим записям и возможность их обновления и удаления определяются установленными мандатными правилами.

Для администратора БД предусмотрены системные привилегии игнорирования мандатного управления доступом, только таким образом можно производить регламентные работы с БД (например, восстановление резервной копии), т. к. это требует установки меток данных, сохраненных ранее.

Для настройки работы сервера с мандатным управлением доступом существует ряд конфигурационных параметров, указываемых в конфигурационном файле postgresql.conf конкретного кластера данных (таблица 26).

Таблица 26

Параметр	Описание
ac_ignore_socket_maclabel	Определяет, будет ли сервер СУБД использовать метку входящего соединения. Если этот параметр установлен в FALSE, то метка входящего соединения будет учитываться при определении максимальной доступной метки сессии и после подключения будет доступна только информация с меткой не выше метки входящего соединения. При установке этого параметра в TRUE метки сеанса будут определяться максимальной меткой пользователя, полученной из ОС.
ac_ignore_server_maclabel	Определяет, будет ли сервер СУБД дополнительно использовать свою метку (метку пользователя postgres) при определении прав пользователя на занесение, удаление и модификацию данных или нет. Если этот параметр установлен в FALSE, то метка сервера используется для блокирования занесения в БД информации с меткой, превышающей метку сервера. Если этот параметр установлен в TRUE, то метка сервера не учитывается.
ac_enable_trusted_owner	Определяет, могут ли владельцы объектов назначать права на доступ к ним другим пользователям. Если этот параметр установлен в значение FALSE, то право назначать права на доступ к любым объектам БД имеют только суперпользователи. Это предотвращает неконтролируемое распространение прав на доступ к информации. Если этот параметр установлен в TRUE, то, кроме суперпользователей, каждый владелец объекта может назначать права на доступ пользователей к «своему» объекту.
ac_enable_grant_options	Определяет, могут ли роли передавать права на доступ с опцией WITH GRANT OPTIONS другим ролям. Если ac_allow_grant_options установлен в FALSE, то запрещается использовать команду GRANT с привилегией WITH GRANT OPTION. Если у роли есть привилегия GRANT OPTIONS и ac_allow_grant_options = false, то передача прав доступа другим ролям также запрещается. Изъятие (REVOKE) привилегии GRANT OPTIONS разрешается всегда.
ac_enable_admin_options	Определяет, могут ли роли передавать право членства роли другим ролям. Если ac_allow_admin_options установлен в FALSE, то запрещается использовать GRANT с привилегией WITH ADMIN OPTION. Если у роли есть привилегия ADMIN OPTIONS и ac_allow_admin_options = false, то передача прав членства другим ролям также запрещается. Изъятие (REVOKE) привилегии ADMIN OPTION разрешается всегда.
ac_enable_truncate	Блокирует (FALSE) или разблокирует (TRUE) возможность выполнения команды TRUNCATE.

Окончание таблицы 26

Параметр	Описание
ac_enable_sequence_ccr	При установке этого параметра в TRUE разрешается ипользование признака CCR для последовательностей аналогично таблицам и видам, в противном случае признак CCR для последовательностей считается всегда установленным.
ac_enable_dblink_mac	Если параметр конфигурации установлен в TRUE, разрешается использование dblink и внешних таблиц FOREIGN TABLE в условиях мандатного управления доступом.
ac_auto_adjust_macs	Если параметр конфигурации установлен в TRUE, разрешается автоматическая установка меток контейнеров. Применяется при восстановления резервных копий, созданных в предыдущих версиях СУБД.
ac_enable_copy_to_file	Блокирует (FALSE) или разблокирует (TRUE) возможность выполнения команды COPY с выводом результатов в файл, доступный серверу СУБД.
ac_caps_ttl	Время жизни информации в секундах о привилегиях пользователя подсистемы безопасности PARSEC (определяет время жизни кэшированной информации о PARSEC-привилегиях пользователя; уменьшение значения приводит к увеличению числа обращений сервера СУБД к подсистеме безопасности PARSEC и как следствие — к снижению производительности сервера СУБД).
ac_debug_print	Если установлен в TRUE, добавляет в журнал сервера отладочную информацию о работе механизмов защиты.

Для более гибкой настройки сервера СУБД расширен синтаксис конфигурационного файла pg_hba.conf опцией ignore_socket_maclabel. Данный параметр может быть указан после метода аутентификации.

Параметр ignore_socket_maclabel=1 позволяет пользователям подключаться к базам данных с игнорированием метки сокета соединения. Если этот параметр не указывается в конфигурационном файле pg_hba.conf или установлен в 0, то игнорирование метки сокета для этого подключения не происходит.

В ОС каждый пользователь может иметь множество меток, которое задается минимальной и максимальной метками диапазона. Чтобы поддержать эту модель в СУБД PostgreSQL каждой сессии пользователя назначаются три метки: максимальная, минимальная и текущая. Их начальная инициализация осуществляется по следующему алгоритму:

1) после прохождения пользователем стандартной процедуры аутентификации сервер считывает из ОС значения максимальной и минимальной меток пользователя и принимает их как максимальную и минимальную метки сессии. При этом, если запись о метках для пользователя не найдена, то максимальная и минимальная метки принимаются равными нулю. Следовательно, пользователи, зарегистрированные

только в сервере СУБД PostgreSQL и не имеющие учетной записи в ОС сервера, всегда имеют минимальный уровень доступа к информации;

2) если параметр конфигурации `ac_ignore_socket_maclabel` установлен в `FALSE`, считывается метка входящего соединения, и, если она попадает в диапазон меток, считанных из ОС, то максимальная метка сессии устанавливается равной метке входящего соединения. При этом, если минимальная метка такого пользователя в ОС сервера выше нулевой, то он вообще не будет допущен к работе с БД;

3) если параметр конфигурации `ac_ignore_server_maclabel` установлен в `FALSE`, то считывается метка серверного процесса и, если она не совместима с максимальной меткой сессии, то процесс аутентификации прерывается;

4) текущей меткой сессии становится максимальная метка сформированного таким образом диапазона.

Если на любом из этих этапов возникает ситуация с несовместимостью меток или выходом за пределы диапазона, то процесс аутентификации клиента прерывается и доступ к БД блокируется.

Если пользователь имеет мандатный атрибут `ac_capable_setmac`, то он может изменять свою текущую мандатную метку в диапазоне от минимальной до максимальной.

СУБД PostgreSQL предоставляет пользователям возможность создавать функции (и, следовательно, триггеры), указывая при этом, будут ли они выполняться с уровнем доступа пользователя, прямо или косвенно вызвавшего функцию (`SECURITY INVOKER`), или с уровнем доступа пользователя, создавшего эту функцию (`SECURITY DEFINER`). При этом в понятие «уровня доступа» входят как дискреционный уровень доступа, так и мандатный, который в данном случае определяется текущими мандатными атрибутами пользователя СУБД, вызвавшего или создавшего функцию, соответственно. При этом метки текущей сессии пользователя, вызвавшего функцию, не изменяются.

При этом следует учитывать, что:

1) при определении функции как `SECURITY DEFINER` она будет всегда вызываться с переустановкой мандатных атрибутов на атрибуты создавшего ее пользователя;

2) при определении функции как `SECURITY INVOKER` она всегда будет выполняться без изменения текущего значения мандатных атрибутов;

3) при вызове функции в качестве триггера выполняются следующие правила в дополнение к указанным:

- перед вызовом в качестве триггера встроенной в СУБД функции к текущим мандатным атрибутам всегда добавляются флаги `ac_capable_ignmaclvl` и `ac_capable_ignmaccat`, чтобы обеспечить полноценную проверку ссылочной

целостности БД;

- перед вызовом в качестве триггера не встроенной функции в качестве текущих мандатных атрибутов всегда устанавливаются мандатные атрибуты пользователя, запустившего данную сессию (соединение) (т. е. пользователя с именем `SESSION_USER`). Это необходимо, чтобы предотвратить получение функцией-триггером пользователя с низким уровнем доступа высоких привилегий в случае каскадного вызова триггеров.

После возврата управления из функции значения текущих мандатных атрибутов всегда восстанавливаются в исходные (до вызова функции) значения.

Функции, написанные на языках низкого уровня, после их подключения имеют полный доступ ко всем внутренним структурам сервера СУБД PostgreSQL и могут произвольно их модифицировать. Кроме этого, поскольку они выполняются в рамках процесса сервера, они имеют соответствующие права доступа к объектам ОС в среде функционирования сервера. Именно поэтому права пользователя `postgres`, под которым запускается сервер, необходимо свести к необходимому минимуму, минуя какой-либо контроль с его стороны (включая текущие мандатные атрибуты).

Поскольку в процессе работы с данными в СУБД возможно изменение организации их хранения путем изменения схемы объектов БД (метаданных), к подобным операциям так же применяются ПРД.

Модификация метаданных возникает каждый раз при изменении структуры БД, что включает в себя создание, модификацию и удаление объектов БД.

Так как некоторые действия над объектами БД могут влиять на хранящихся в них данных (как правило, модификация или удаление объекта или его части), при использовании мандатного управления доступом к данным объекта необходимо разграничивать и доступ к изменению метаданных в части, относящейся к этому объекту.

Аналогично операциям с данными: действия с объектами БД должны быть приведены к видам доступа на чтение и на запись информации для возможности применения к ним мандатных ПРД. Все множество операций с метаданными может быть приведено следующим образом:

- `CREATE`, `ADD` — доступ на запись;

- `ALTER`, `DROP` — последовательное выполнение доступа на чтение и запись информации;

- использование или обращение к объекту в других SQL-командах — доступ на чтение.

Проверка мандатных прав доступа к метаданным осуществляется одновременно с проверкой дискреционных прав доступа к ним после разбора и построения плана запроса

непосредственно перед его выполнением, когда определены все необходимые для проверки данные и проверяемые объекты. Таким образом, доступ предоставляется только при одновременном санкционировании дискреционными ПРД.

Некоторые операции над объектами, такие как DROP всего объекта или его столбца и TRUNCATE влекут за собой удаление данных. В случае защиты метками записей объекта существуют ограничения на выполнение этих операций.

Операции удаления невозможны при наличии разных меток на записях, т. к. операция применяется ко множеству строк. Это связано с тем, что операция удаления интерпретируется как последовательное предоставление доступа на чтение и на запись, что возможно только при равенстве меток субъекта и объекта. В случае, когда строки имеют разные метки, данное условие выполниться не может.

Операция удаления доступна только для администратора и пользователей, обладающих привилегиями игнорирования мандатного управления доступом.

4.10.1. Средства управления мандатными ПРД к объектам БД

Для управления мандатными ПРД к объектам БД СУБД PostgreSQL используется графическая утилита pgadmin3.

При создании мандатная метка объекта БД устанавливается равной текущей мандатной метке создавшего его пользователя, мандатный признак CCR при этом выставляется в значение ON.

Если пользователь имеет мандатный атрибут ac_capable_chmac, то он может менять мандатную метку принадлежащих ему объектов в пределах своего диапазона мандатных меток с помощью следующей команды:

```
MAC LABEL ON тип_объекта имя_объекта IS новое_значение_мандатной_метки;
```

В качестве типа объекта указывается тип объекта БД, например DATABASE, TABLE, FUNCTION.

Аналогичным образом для объектов-контейнеров может быть изменен мандатный признак CCR:

```
MAC CCR ON тип_объекта имя_объекта IS { ON | OFF };
```

Дополнительно введен новый тип объекта — кластер CLUSTER. Для установки метки на кластер используется следующий запрос:

```
MAC LABEL ON CLUSTER IS новое_значение_мандатной_метки;
```

что является синонимом к команде

```
MAC LABEL ON TABLESPACE pg_global IS новое_значение_мандатной_метки;
```

Для изменения мандатного признака CCR кластера используется следующая команда:

```
MAC CCR ON CLUSTER IS { ON | OFF };
```

что является синонимом к команде

```
MAC CCR ON TABLESPACE pg_global IS { ON | OFF };
```

Значения мандатных меток объектов содержатся в полях maclabel таблиц системного каталога, откуда могут быть выбраны соответствующим запросом.

Значения мандатного признака CCR объектов-контейнеров содержатся в следующих полях таблиц системного каталога:

- для баз данных - datmacccr системной таблицы pg_database;
- для схем - nspmacccr системной таблицы pg_namespace;
- для табличных пространств - spcmacccr системной таблицы pg_tablespace;
- для отношений - relmacccr системной таблицы pg_class.

Для просмотра мандатного признака CCR кластера может быть использована следующая команда:

```
SELECT cluster_macccr;
```

По умолчанию записи создаваемых таблиц не защищены мандатными метками. Для того чтобы создать таблицы с защищенными метками записями, следует использовать следующий вариант команды CREATE TABLE:

```
CREATE TABLE имя_таблицы (
  ... -- список_столбцов
) WITH ( MACS = true, ... );
```

При этом все вставляемые записи по умолчанию наследуют текущие мандатные метки создавших их пользователей. Пользователи, имеющие установленный мандатный атрибут ac_capable_chmac, могут явно задать значение мандатной метки вставляемой записи. Задаваемая метка должна быть в пределах диапазона меток пользователя, либо пользователь должен иметь атрибуты игнорирования мандатного контроля ac_capable_ignmaclvl и ac_capable_ignmaccat с помощью варианта команды INSERT:

```
INSERT INTO имя_отношения (maclabel, ... список_столбцов)
  VALUES (значение_мандатной_метки, ... значения_столбцов)
```

Для защиты записей уже созданных таблиц без мандатных меток следует использовать следующий вариант команды ALTER TABLE:

```
ALTER TABLE имя_таблицы SET WITH MACS;
```

После исполнения этой команды все записи таблицы автоматически получат текущую метку таблицы.

Для того, чтобы убрать защиту записей мандатными метками, следует использовать следующий вариант команды ALTER TABLE:

```
ALTER TABLE имя_таблицы SET WITHOUT MACS;
```

Для изменения мандатных меток существующих записей пользователи с атрибутом ac_capable_chmac могут использовать команду CHMAC:

```
CHMAC имя_отношения SET maclabel = новое_значение_мандатной_метки WHERE ...
```

Совокупная метка записей таблицы (максимальная по мандатному уровню и наиболее полная по мандатным категориям) может быть получена с помощью агрегирующей функции supmaclabel:

```
SELECT supmaclabel(maclabel) FROM имя_отношения;
```

Просмотреть значения мандатных меток доступных записей можно с помощью команды SELECT:

```
SELECT maclabel FROM имя_отношения
```

В командах INSERT и CHMAC значения мандатных меток не обязательно должны быть заданы в явном виде. Для задания метки допускается использование любого скалярного выражения, возвращающего результат, приводимый к типу мандатной метки.

Для того чтобы сохранить записи вместе с их метками в архиве и в дальнейшем загрузить их обратно, предусмотрен специальный флаг MACS команды COPY. Вывести доступные пользователю данные вместе с метками может любой пользователь, загрузить же обратно — только пользователь с установленным мандатным атрибутом ac_capabel_chmac. При этом метки загружаемых записей должны находиться в пределах диапазона меток пользователя, либо пользователь должен иметь атрибуты игнорирования мандатного контроля ac_capable_ignmaclvl и ac_capable_ignmaccat. Например, выгрузка и обратная загрузка данных из/в таблицы test может выглядеть так:

```
COPY имя_отношения TO stdout WITH MACS
COPY имя_отношения FROM stdin WITH MACS
```

Использовать команду COPY без указания меток может любой пользователь. Загруженные таким образом данные будут иметь метки, равные текущей метке сессии пользователя.

Параметр конфигурации сервера ac_enable_copy_to_file разрешает выполнять команду COPY с выводом результатов в файл, доступный серверу СУБД. Для этого он должен быть установлен в TRUE.

4.10.2. Целостность мандатных атрибутов кластера баз данных

В СУБД PostgreSQL ДП-модель накладывает ограничение на мандатную метку конфиденциальности объекта: метка объекта не может превышать метку контейнера, в котором он содержится (4.10).

Для вывода информации о соблюдении ДП-модели между объектами-контейнерами и находящимися в них объектами реализована SQL-функция check_mac_integrity, которая выводит информацию в следующем виде:

- objid — Идентификатор объекта;
- classid — Идентификатор класса объекта;
- cobjid — Идентификатор контейнера, содержащего объект;
- cclassid — Идентификатор класса контейнера, содержащего объект;

- status — Результат проверки. Может принимать следующие значения: OK(модель соблюдается для объекта и контейнера) и FAIL(модель не соблюдается для объекта и контейнера).

Примечание. Информация о соблюдении модели выводится только для отношений (таблиц, представлений, последовательностей), схем, баз данных и табличных пространств.

Для исправления некорректно установленной мандатной метки отношения, например, при восстановлении резервной копии кластера ранних версий, используется SQL функция fix_mac_integrity. Данная функция может быть исполнена только пользователем с правами администратора, а также при установленном параметре ac_auto_adjust_macs = true в конфигурационном файле postgresql.conf.

4.10.3. Ссылочная целостность мандатных атрибутов

Средства обеспечения целостности в реляционных БД представляют собой механизмы автоматической поддержки системы правил, определяющих допустимость и корректность обрабатываемых данных, и могут быть разбиты на несколько видов:

- ограничение целостности полей данных — ограничения, накладываемые на используемые в отношении домены (задание разрешенных диапазонов значений, запрет наличия неопределенных значений NULL, задание значений по умолчанию) или ограничения непосредственно таблицы (уникальность каждой записи, ограничение уникальности по выбранным столбцам, вычисляемые значения столбцов и условия допустимых сочетаний значений в столбцах);
- декларативная ссылочная целостность — описание зависимостей между разными отношениями в БД (наличия в одном отношении вторичного ключа, ссылающегося на первичный ключ в другом). Подобные зависимости могут быть выявлены при анализе предметной области между ее сущностями или при проектировании БД в процессе нормализации (в этом случае одна сущность предметной области может состоять из набора отношений). Ссылочная целостность реализуется путем описания ограничений на значения вторичных ключей в одном отношении и правил их обработки в случае изменения первичных ключей в другом;
- динамическая ссылочная целостность — триггера, назначаемые для выполнения при реализации заданного вида доступа к конкретной таблице. Триггер представляет собой исполняемый код, который может динамически проверить заданные условия корректности выполняемой операции, и при необходимости внести изменения в другие таблицы.

В приведенном определении отсутствуют мандатные атрибуты, так как в общем случае между классификационными метками записей разных таблиц может не быть зависи-

мости. С другой стороны, существует частный случай ссылочной целостности, образованный между таблицами, являющимися частями одной сущности предметной области, что может возникнуть в процессе нормализации.

Для обеспечения целостности мандатных атрибутов список событий для ограничений целостности наряду с существующими событиями ON SELECT, ON INSERT, ON UPDATE, ON DELETE расширен событием изменения мандатных атрибутов ON CHMAC. Таким образом, возможно создание ограничение ссылочной целостности следующим образом:

```
ALTER TABLE имя_таблицы1 ADD CONSTRAINT имя_ограничения
  FOREIGN KEY (список_полей1) REFERENCES имя_таблицы2 (список_полей2)
    ON CHMAC {NO ACTION | RESTRICT | CASCADE }
```

Механизм действия такого ограничения целостности аналогичен механизму действия подобного ограничения для события ON UPDATE для данных: при изменении мандатных атрибутов записи в одной таблице можно указать каскадное изменение мандатных атрибутов связанных записей второй таблицы, либо запрет возможность такого изменения.

Аналогично существует возможность создания триггеров для указанного события изменения мандатных атрибутов (CHMAC), например:

```
CREATE TRIGGER имя_триггера
  BEFORE CHMAC ON имя_таблицы
  FOR EACH ROW
  EXECURE PROCEDURE имя_процедуры()
```

Таким же образом могут быть заданы и правила для видов:

```
CREATE RULE имя_правила
  AS ON CHMAC TO имя_вида
  DO ALSO ...
CREATE RULE имя_правила
  AS ON CHMAC TO имя_вида
  DO INSTEAD ...
```

В этом случае при изменении мандатных атрибутов записи будут вызываться соответствующие функции и правила, что дает возможность запрограммировать произвольную логику обеспечения целостности мандатных атрибутов внутри БД.

4.10.4. Особенности создания правил и триггеров

В СУБД PostgreSQL правила (RULE) и триггеры (TRIGGER) наследуют метку таблицы, для которой они созданы. Эти объекты могут быть созданы пользователем-владельцем таблицы (метка которого будет совпадать с меткой таблицы), либо пользователями с привилегиями игнорирования мандатного доступа. В противном случае генерируется ошибка доступа.

Если триггер использует триггерную функцию, метка которой отлична от {0,0}, то

при исполнении триггера для таблицы со сброшенным мандатным признаком CCR возможны нарушения в работе триггеров. Это обусловлено тем, что запись триггерной функции может быть не доступна для пользователя в связи с МПРД. В таких случаях будет выведено сообщение:

```
cache lookup failed: внутренняя ошибка или отсутствуют необходимые мандатные атрибуты
```

Во избежании этого настоятельно рекомендуется устанавливать мандатную метку триггерной функции в значение `{0,0}`.

4.10.4.1. Особенности использования представлений и материализованных представлений

Представления (`VIEW`) и материализованные представления (`MATERIALIZED VIEW`) не могут иметь метки на строках, поскольку выполняют агрегацию данных из других источников данных (таблиц и представлений).

Примечание. Представления создаются с установленным флагом CCR, поэтому при попытке доступа к нему от имени пользователя, мандатная метка которого не превосходит метки представления, это представление не будет «видно» пользователю в силу МПРД. Для того, чтобы пользователь имел доступ к такому представлению, необходимо сбросить мандатный признак CCR представления.

4.10.5. Функции сравнения для типа `maclabel`

В СУБД PostgreSQL изменено поведение следующих функций для типа `maclabel`: `eq`, `ne`, `lt`, `le`, `gt`, `ge`. В указанных функциях вместо индексного сравнения используется сравнение мандатных меток соответствующими операторами =, <>, <, <=, >, >=.

Для проверки на несравнимость мандатных меток используется функция `nc`.

Операторы индексного сравнения переименованы в `maclabel_idx_eq`, `maclabel_idx_ne`, `maclabel_idx_lt`, `maclabel_idx_le`, `maclabel_idx_gt`, `maclabel_idx_ge`.

4.10.6. Система привилегий СУБД и управление ими

Система привилегий СУБД PostgreSQL предназначена для передачи отдельным пользователям прав выполнения определенных административных действий. Обычный пользователь системы не имеет дополнительных привилегий.

Привилегии являются подклассом атрибутов пользователя СУБД PostgreSQL.

Привилегии ОС, используемые в СУБД PostgreSQL, кроме атрибута `ac_session_maclabel`, не могут быть изменены с помощью средств СУБД ни пользователями, ни администраторами СУБД:

- `ac_session_maclabel` — текущая мандатная метка сессии пользователя СУБД.

Эта метка определяет доступные пользователю объекты БД и является меткой по

умолчанию для создаваемых пользователем объектов. При соединении пользователя с СУБД значение этого атрибута устанавливается равным метки соединения или ac_user_max_maclabel;

- ac_user_max_maclabel — максимально возможное значение для ac_session_maclabel;
- ac_user_min_maclabel — минимально возможное значение для ac_session_maclabel;
- ac_capable_ignmaclvl — позволяет пользователю игнорировать мандатный контроль по уровням;
- ac_capable_ignmaccat — позволяет пользователю игнорировать мандатный контроль по категориям;
- ac_capable_mac_readsearch — позволяет пользователю игнорировать мандатный контроль по уровням и категориям при чтении данных;
- ac_capable_setmac — позволяет пользователю изменять текущую метку своей сессии в пределах, заданных ее минимальным и максимальным значением;
- ac_capable_chmac — позволяет пользователю изменять метки объектов БД.

В случае, если пользователь СУБД не зарегистрирован в ОС на стороне сервера СУБД, все его мандатные атрибуты имеют нулевое значение.Администраторам СУБД дополнительно к их атрибутам из ОС всегда добавляются атрибуты ac_capable_ignmaclvl, ac_capable_ignmaccat и ac_capable_chmac.

Для управления привилегиями СУБД PostgreSQL может быть использована графическая утилита pgadmin3.

Просмотреть текущие значения привилегий (атрибутов пользователя) можно с помощью команды:

```
SHOW attr_name
```

Установить новое значение атрибута ac_session_maclabel можно с помощью команд:

```
SET ac_session_maclabel = новое_значение_метки
SELECT set_config ('ac_session_maclabel', новая_метка, false);
```

В первой форме в качестве нового значения метки можно использовать только явно заданные значения метки, во второй— значение метки может быть любым выражением, возвращающим скалярное значение, приводимое к типу мандатной метки.

4.11. 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 интер-

фейса 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, являющихся объектами ФС. Комплекс программ электронной почты использует технологию `maildir`, обеспечивающую хранение почтовых сообщений в виде отдельных объектов ФС. Создаваемые файлы почтовых сообщений маркируются мандатными метками, 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.

4.12. Настройка загрузчика Grub

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

В загрузчике Grub возможно задать параметры ядра PARSEC, приведенные в таблице 27.

Таблица 27

`max_ilev` (Maximal Integrity Level)	Задает максимальный уровень целостности в системе на момент старта (воспринимаемый модулями). Допустимые значения от 0 до 255. При установке ОС задается по умолчанию значение 63. **ВНИМАНИЕ!** Если в системе присутствуют файлы с метками, имеющими уровень целостности выше задаваемого в данный момент, это может вызвать отказ доступа к данным файлам (сокетам, каталогам и т.д.)

Окончание таблицы 27

parsec.ccnr_relax	При заданном значении 1 позволяет непривилегированному пользователю выполнять запись файлов с разным уровнем конфиденциальности в контейнер (каталог) с установленным флагом ccnr (значение по умолчанию 0)
reset_ilev_on_chroot (Reset Label on chroot)	Сброс (обнуление) уровня целостности метки у дочерних процессов при использовании родительским процессом системных вызовов chroot(2) и pivot_root(2). Текущие значение целостности никак не влияет на поведение. Допустимые значения 0/1, y/n и т.д. Значение по умолчанию true (обнуляет)
noload_files (Reject load at low integrity level)	Запрет загрузки модулей ядра, firmware, образов kexec, ключей и сертификатов X.509 привилегированным пользователям (root, uid=0) с низким уровнем целостности. Допустимые значения 0/1, y/n и т.д. Значение по умолчанию true (запрещено)
ccnr_reject (Disallow root to set CCNR* flags)	Запрет установки флага ccnr привилегированным пользователям (root, uid=0). Допустимые значения 0/1, y/n и т.д. Значение по умолчанию 0 (разрешено)

5. ЗАЩИТА СРЕДЫ ВИРТУАЛИЗАЦИИ

Работа в среде виртуализации осуществляется при обязательном прохождении процедуры аутентификации на сервере виртуализации libvirt и в условиях применения дискреционных и мандатных правил разграничения доступа.

Настройка сервера виртуализации libvirt для работы в ЕПП ОС СН, а также идентификация и аутентификация при доступе к серверу виртуализации и к рабочему столу виртуальных машин выполняется в соответствии с РУСБ.10015-01 95 01-1.

5.1. Дискреционное управление доступом к виртуальной машине

При взаимодействии с сервером виртуализации libvirt осуществляется дискреционное управление доступом к управлению виртуальными машинами. Разграничение выполняется драйвером доступа `parsec`, специально разработанным с использованием прикладного программного интерфейса драйверов доступа libvirt. Основанием для принятия решения о предоставлении доступа является сравнение дискреционных атрибутов виртуальной машины и дискреционных атрибутов пользователя с учетом выполняемой операции.

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

Операции по изменению состава или конфигурации виртуальных машин требуют вхождения пользователя в специальную локальную административную группу. Имя административной группы задается в конфигурационном файле `/etc/libvirt/libvirtd.conf` параметром `admin_group = "libvirt-admin"`.

Для каждой виртуальной машины задается список контроля доступа, в котором указываются субъекты доступа (пользователи и группы), обладающие доступом к виртуальной машине. Различаются три типа доступа к виртуальной машине:

- «Просмотр свойств» — видимость виртуальной машины в списке и просмотр ее свойств;
- «Использование» — просмотр свойств, запуск и работа с виртуальной машиной;
- «Администрирование» — полный доступ к виртуальной машине, включая запуск, изменение ее свойств и управление правами доступа к ней.

Пользователи видят в списке виртуальных машин только те виртуальные машины, к которым им явно предоставлен доступ.

После создания виртуальной машины список контроля доступа пуст и доступ к виртуальной машине разрешен только членам административной группы. При наличии права

использования пользователям позволяется запускать виртуальную машину и работать с ней. При этом завершить работу или произвести иные разрешенные действия с виртуальной машиной может только пользователь, который ее запустил.

Члены административной группы могут завершать работу виртуальных машин других пользователей.

В момент запуска виртуальной машины модуль поддержки дискреционного управления доступом `dac` назначает владельцем виртуальной машины запускающего ее пользователя, а группой — группу `libvirt-qemu`. Это отражается в виде наличия динамической метки безопасности модели `dac` и идентификатора владельца как процесса виртуальной машины в ОС СН, так и необходимых устройств и файл-образов, принадлежащих виртуальной машине. При последующих операциях данная информация используется для проверки дискреционного доступа к виртуальной машине.

5.2. Мандатное управление доступом к виртуальной машине

При взаимодействии с сервером виртуализации libvirt осуществляется мандатное управление доступом к управлению виртуальными машинами. Разграничение выполняется драйвером доступа `parsec`, специально разработанным с использование прикладного программного интерфейса драйверов доступа libvirt. Основанием для принятия решения о предоставлении доступа является сравнение метки безопасности виртуальной машины и мандатного уровня доступа пользователя с учетом выполняемой операции.

Все операции с виртуальной машиной 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 виртуальных машин — записи.

Мандатное управление доступом осуществляется следующим образом:

- остановленная виртуальная машина не обладает мандатными атрибутами (если для нее не задана статическая метка безопасности модели PARSEC);
- запущенная виртуальная машина наследует мандатную метку запускающего пользователя;
- доступ к функционирующей виртуальной машине предоставляется только при равенстве мандатного уровня доступа пользователя мандатной метке виртуальной машины;
- доступ к получению информации о виртуальной машине и ее конфигурации предоставляется в соответствии с мандатным уровнем пользователя.

В момент запуска виртуальной машины модуль поддержки мандатного управления доступом `parsec` устанавливает виртуальной машине мандатный уровень, определенный

по соединению запускающего пользователя. Это отражается в виде наличия динамической метки безопасности модели PARSEC и мандатной метки как процесса виртуальной машины в ОС СН, так и необходимых устройств и файл-образов, принадлежащих виртуальной машине. При последующих операциях данная информация используется для проверки мандатного доступа к виртуальной машине.

Существует возможность задания статической метки безопасности модели PARSEC. В этом случае виртуальная машина может быть запущена только под заданным мандатным уровнем доступа.

ВНИМАНИЕ! Существуют ограничения по конфигурированию виртуальной машины: в качестве сетевого адаптера не может быть выбрано устройство `virtio`.

Примечания:

1. В случае использования в качестве гостевой системы ОС СН виртуальная машина не должна запускаться в мандатном контексте. Вместо этого необходимо выполнять удаленный вход с требуемым мандатным уровнем доступа средствами ОС СН.
2. Настоятельно рекомендуется использовать режим «Только чтение» при запуске виртуальных машин в ненулевом мандатном контексте.

5.3. Режим запрета модификации файлов-образов виртуальной машины

В некоторых случаях требуется обеспечение неизменности файлов-образов виртуальной машины в процессе ее функционирования, что позволяет выполнять повторный запуск заранее подготовленной виртуальной машины из фиксированного состояния. В этом случае все результаты работы пользователя после завершения функционирования виртуальной машины удаляются.

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

- временный снимок — способ функционирования средства эмуляции аппаратного обеспечения QEMU, при котором основной файл-образ защищается от записи, а все результаты работы пользователя фиксируются во временном снимке, существующем только в процессе функционирования виртуальной машины;
- снимок — во время запуска виртуальной машины в заданном каталоге создается снимок, удаляемый после завершения функционирования виртуальной машины;
- полная копия — во время запуска виртуальной машины в заданном каталоге создается полная копия файла-образа, удаляемая после завершения функционирования виртуальной машины.

Создание полной копии может замедлять процесс запуска виртуальной машины по причине копирования файла-образа большого размера. Но данный вариант позволяет ис-

пользовать 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"
```

6. РЕГИСТРАЦИЯ СОБЫТИЙ БЕЗОПАСНОСТИ

В ОС реализована расширенная подсистема протоколирования, осуществляющая регистрацию событий в двоичные файлы с использованием сервиса `parlogd`. В 6.1 приведено описание настройки параметров протоколирования для объектов ФС (файловый аудит) и для пользователей (аудит процессов). Применение настроенных параметров аудита процессов осуществляется PAM-модулем `pam_parsec_aud`. По умолчанию регистрация настроенных для пользователя событий аудита процессов включена в PAM-сценарии: `fly-dm`, `fly-dm-np`, `login`, `su`, `sumac`, `sumac.xauth`. Для протоколирования событий аудита процессов пользователя, проходящего аутентификацию через другие PAM-сценарии, необходимо включить в соответствующие сценарии строку следующего вида:

```
session required pam_parsec_aud.so
```

Также в состав ОС входит программное средство Zabbix, обеспечивающее мониторинг, централизованный аудит и протоколирование событий сетевых ресурсов.

В библиотеках подсистемы безопасности PARSEC реализован программный интерфейс для протоколирования событий с использованием расширенной подсистемы протоколирования, применяемый для регистрации событий в СУБД PostgreSQL и комплексе программ электронной почты.

6.1. Средства управления протоколированием

Для работы с подсистемой протоколированием имеется ряд графических утилит, которые могут быть использованы для настройки параметров регистрации событий и просмотра протоколов:

- `fly-admin-smc` («Управление политикой безопасности») — управление протоколированием, привилегиями и мандатными атрибутами пользователей, работа с пользователями и группами;
- `fly-admin-viewaudit` («Журнал безопасности») — выборочный просмотр протоколов аудита.

Более подробное описание утилит см. в электронной справке.

Далее рассмотрены средства управления протоколированием в режиме командной строки.

6.1.1. getfaud

Синтаксис:

```
getfaud [-d, --default] [-R, --recursive] [-L, --logical] [-P, --physycal]
  [-n, --numeric] [-l, --long] [-p, --absolute-names] [-c, --omit-header]
  [-s, --skip-empty] [-h, --help] [-v, --version] файлы и/или каталоги
```

Команда `getfaud` служит для получения списков правил протоколирования над файловыми объектами. Следующие события доступны для протоколирования:

- `o`, `open` — открытие файла;
- `c`, `create` — создание файла;
- `x`, `exec` — исполнение файла;
- `u`, `delete` — удаление файла (в каталоге);
- `r`, `acl` — смена ACL;
- `n`, `chown` — смена владельца;
- `m`, `mac` — изменение метки;
- `y`, `modify` — изменение файла;
- `a`, `audit` — изменение списка регистрируемых событий файла;
- `d`, `chmod` — изменение прав доступа к файлу.

Информация о списках посылается на стандартный вывод и может являться входными данными для команды `setfaud` (6.1.2).

Опции приведены в таблице 28.

Таблица 28

Опция	Описание
`-d`, `--default`	Работать со списком правил протоколирования по умолчанию
`-R`, `--recursive`	Для поддиректорий рекурсивно
`-L`, `--logical`	Следовать по символическим ссылкам
`-P`, `--phisical`	Не следовать по символическим ссылкам
`-n`, `--numeric`	Выводить информацию о флагах регистрации событий в цифровой форме
`-l`, `--long`	Выводить флаги регистрации событий в длинной форме
`-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

6.1.2. setfaud

Синтаксис:

```
setfaud [-s, --set] [-b, --remove] [-m, --modify] [-d, --default]
  [-S, --set-file] [-X, --remove-all] [-M, --modify-file] [-B, --restore]
  [-R, --recursive] [-L, --logical] [-P, --physical] [-h, --help]
  [-v, --version] [правила протоколирования] файлы...
```

Команда `setfaud` устанавливает списки правил протоколирования на файлы. Правила протоколирования задаются в виде:

```
[u:<пользователь>:<флаги протоколирования>]
  [,g:<группа>:<флаги протоколирования>][,o:<флаги протоколирования>], ... ,
```

где `<пользователь>` и `<группа>` — символические или численные идентификаторы пользователя и группы; `u:` означает правило для пользователя, `g:` — для группы, `o:` — для остальных.

```
<флаги протоколирования> := <флаги успешных операций>[[:<флаги неуспешных
  операций>], ...]
```

При этом флаги операций могут иметь вид:

```
<+|-><имя протоколируемого события>,...
```

(например, `+exec`, `-open`) или:

```
[+|-]<число>
```

или:

```
<сокращенное имя протоколируемого события#1><сокращенное имя протоколируемого
  события#2>...
```

(например, `ou` — `+open`, `+delete`).

Чтобы посмотреть список протоколируемых событий, набрать:

```
setfaud -h
```

Правила задаются или в командной строке (параметры `-s`, `-m`), или в файлах (параметры `-S`, `-M`, `-B`). При этом, файлы могут быть сформированы с помощью перенаправления вывода команды `getfaud` (см. 6.1.1).

Только администратор может изменять списки правил протоколирования у файлов.

Опции приведены в таблице 29.

Таблица 29

Опция	Описание
`-s, --set`	Установить список протоколирования из командной строки
`-b, --remove`	Удалить все элементы списка протоколируемых событий
`-m, --modify`	Изменить или добавить элементы списка из командной строки
`-d, --default`	Работать со списком протоколирования по умолчанию
`-S, --set-file`	Установить список протоколируемых событий из файла
`-X, --remove-all`	Удалить все списки протоколируемых событий
`-M, --modify-file`	Изменить или добавить элементы списка протоколируемых событий из файла
`-B, --restore`	Восстановить атрибуты из файла
`-R, --recursive`	Для поддиректорий рекурсивно
`-L, --logical`	Следовать по символическим ссылкам

Окончание таблицы 29

Опция	Описание
-P, --physical	Не следовать по символическим ссылкам
-h, --help	Вывести справку и выйти
-v, --version	Вывести информацию о версии и выйти

6.1.3. useraud

Синтаксис:

```
useraud [-d, --delete] [-n, --numeric] [-l, --long] [-g, --group]
  [-o, --other] [-m, --modify] [-h,--help] [-v, --version]
  [пользователь/группа]
```

Команда useraud позволяет просматривать и изменять правила протоколирования для пользователей.

```
useraud [-dnghvolm] [имя пользователя(группы)] [флаги протоколирования]
```

где <флаги протоколирования>: = <флаги успешных операций>

```
[[:<флаги неуспешных операций>], ...]
```

При этом флаги операций могут иметь вид:

```
<+|-><имя протоколируемого события>,...
```

(например, +exec, -open) или:

```
[+|-]<число>
```

или:

```
<сокращенное имя протоколируемого события#1><сокращенное имя протоколируемого
  события#2>...
```

(например, ou — +open,+delete).

Список событий можно получить из помощи команды (параметр -h, --help).

Опции приведены в таблице 30.

Таблица 30

Опция	Описание
-d, --delete	Сбросить правила протоколирования
-n, --numeric	Вывести флаги в шестнадцатеричном формате
-l, --long	Длинный формат вывода флагов
-g, --group	Для группы (по умолчанию — для пользователя)
-o, --other	Для остальных (любой пользователь)
-m, --modify	Изменить существующее правило
-h, --help	Вывести справку и выйти
-v, --version	Вывести информацию о версии и выйти

6.1.4. parselog

Синтаксис:

```
parselog [-v, --version] [-h, --help] [-c, --count] [-f, --follow]
  [-l, --syslog] [-s, --silent] [-b, --binary] [-a, --facility] [-e, --events]
  [-t, --time] [-u, --status] [-x, --uids] [-y, --gids] [-z, --euids]
  [-0..F, --arg0 ... --argF] [файл-журнал]
```

Команда `parselog` может быть использована для анализа двоичных файлов аудита, записанных с помощью `parlogd`.

```
parselog [параметры] [двоичный файл аудита]
```

Если в качестве аргумента не указан файл с данными журнала, то данные ожидаются из стандартного ввода, таким образом, совместно с опцией `-b`, позволяя организовывать конвейеры.

Опции приведены в таблице 31.

Таблица 31

Опция	Описание
`-c, --count`	Вывести статистику
`-f, --follow`	Ожидать появления новых записей в файле
`-l, --syslog`	Записывать выходные данные в систему `syslog` (как служба `LOG_LOCAL0`)
`-s, --silent`	Не выводить ничего на консоль
`-b, --binary`	Вывод в двоичном формате (для конвейеризации)
`-h, --help`	Вывести справку и выйти
`-v, --version`	Вывести информацию о версии и выйти

Параметры-фильтры приведены в таблице 32.

Таблица 32

Параметр-фильтр	Описание
`-a, --facility`	Список служб. Доступные службы: `user`, `proc`, `file`, `custom` или десятичное число от 0 до 15
`-e, --events`	Список событий. Для просмотра списка имен событий (зависит от плагинов) использовать `-e help`
`-t, --time`	Временной диапазон в формате: `<от даты>[-до даты]` где формат даты — `%y[%m[%d[%H[%M[%S]]]]]`
`-u, --status`	Статус. Доступные статусы: `success`, `failed`
`-x, --uids`	Список пользователей (в символьном или десятичном формате)
`-y, --gids`	Список групп (в символьном или десятичном формате)

Окончание таблицы 32

Параметр-фильтр	Описание
`-z, --euids`	Список эффективных идентификаторов пользователей в символьном или десятичном формате
`-0..F, --arg0 ... --argF`	Поиск аргумента (от 1 до 15) с помощью регулярных выражений. `arg0` всегда соответствует программе-контексту события. Предполагается, что возвращаемое значение — это последний аргумент события

6.1.5. kernlog, userlog

Команды `kernlog` и `userlog` предназначены для анализа двоичных файлов журнала регистрации событий ядра и событий, приходящих от пользователя, соответственно. Обе команды используют `parselog` (см. 6.1.4) и являются надстройками над ней. Команда `parselog` принимает имя обрабатываемого двоичного файла в качестве параметра. Для команд-надстроек имя анализируемого файла предопределено. Для `kernlog` анализируемым файлом является `/var/log/parsec/kernel.mlog`. В этом файле регистрируются события ядра (сервис `parlogd`). Для `userlog` анализируемым файлом является `/var/log/parsec/user.mlog`. В этом файле регистрируются события, приходящие от пользовательских процессов (сервис `parlogd`). В остальном команды `kernlog` и `userlog` аналогичны `parselog` и принимают те же аргументы командной строки (см. 6.1.4).

6.1.6. psaud

Синтаксис:

```
psaud [-d, --delete] [-n, --numeric] [-l, --long] [-h, --help] [--version]
  [правила протоколирования]
```

Команда `psaud` позволяет изменить или считать правила протоколирования выбранного процесса. Если правила протоколирования не указаны в качестве аргумента, то команда выполняет их считывание с процесса, заданного параметром (идентификатор процесса).

Если аргумент правила протоколирования присутствует, то команда устанавливает правила на процесс. Правила задаются в виде:

```
<флаги протоколирования> := <флаги успешных
  операций>[[:<флаги неуспешных операций>], ...]
```

При этом флаги операций могут иметь вид:

```
<+|-><имя протоколируемого события>,...
```

(например, `+exec, -open`) или:

```
[+|-]<число>
```

или:

```
<сокращенное имя протоколируемого события#1><сокращенное имя протоколируемого
```

```
события#2>...
```

(например, `ou — +open,+delete`).

Список событий можно получить из помощи команды (параметр `-h, --help`).

Только администратор может устанавливать и считывать правила протоколирования процессов. Правила протоколирования наследуются порожденными процессами.

Опции приведены в таблице 33.

Таблица 33

Опция	Описание
`-d, --delete`	Снять все правила протоколирования с процесса
`-n, --numeric`	Выводить информацию о правилах протоколирования в численном виде
`-l, --long`	Выводить информацию о правилах протоколирования в длинной форме
`-h, --help`	Вывести справку и выйти
`--version`	Вывести информацию о версии и выйти

6.1.7. Дополнительные параметры системы протоколирования событий

Для тестирования ОС на новых аппаратных конфигурациях можно отключить протоколирование набора системных вызовов одним из двух следующих способов:

1) отключение протоколирования системных вызовов, не используемых для мандатного управления доступом. Для этого необходимо выполнить команду:

```
echo 1 > /parsecfs/disable-non-mac-audit
```

Если вывод команды:

```
cat /parsecfs/disable-non-mac-audit
```

равен `1`, то протоколирование системных вызовов, не используемых для мандатного управления доступом, отключено.

2) отключение протоколирования всех системных вызовов. Для этого необходимо выполнить команду:

```
echo 1 > /parsecfs/disable-all-audit
```

Если вывод команды:

```
cat /parsecfs/disable-all-audit
```

равен `1`, то протоколирование всех системных вызовов отключено.

Примечание. Если необходимо, чтобы отключение происходило при загрузке ОС, то указанные выше команды необходимо поместить в файл `/etc/rc.local`.

6.2. Средства централизованного аудита и протоколирования

Программное решение Zabbix предоставляет 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.

6.3. Регистрация событий в СУБД PostgreSQL

В СУБД PostgreSQL для настройки режима работы подсистемы регистрации событий используются конфигурационный параметр `ac_audit_mode` файла `postgresql.conf` (6.3.2). Этот параметр может быть изменен только перезапуском сервера. Параметр может принимать следующие значения:

- `internal` — для настройки регистрации событий используются соответствующие команды SQL, а настройки хранятся в таблице `pg_db_role_settings`;
- `external` — для настройки используется внешний файл `pg_audit.conf` (6.3.1);
- `external,internal` — смешанный режим. Настройки берутся сначала из внешнего файла `pg_audit.conf`, после чего дополняются настройками из таблицы `pg_db_role_settings`;
- `internal,external` — смешанный режим. Настройки берутся сначала из таблицы `pg_db_role_settings`, после чего дополняются настройками из внешнего файла `pg_audit.conf`;
- `none` — протоколирование в данном режиме отключено.

Примечание. В СУБД PostgreSQL версии 9.6 подсистема регистрации событий по умолчанию работает в режиме `internal`.

6.3.1. Настройка файла `pg_audit.conf` для регистрация событий

Настройка подсистемы сообщений аудита в СУБД PostgreSQL обеспечивается конфигурационным файлом `pg_audit.conf` конкретного кластера данных, который имеет следующий формат:

- аудит действий администратора СУБД:

```
success events mask = F00E7 failure events mask = 0 user = postgres
```

- для пользователя `sny` выполнять регистрацию только неуспешных действий:

```
success events mask = 0 failure events mask = FFFFF user = sny
```

- для всех остальных пользователей выполнять регистрацию всех неуспешных действий и всех успешных действий, кроме доступа к данным:

```
success events mask = F0707 failure events mask = FFFFF
```

В этом конфигурационном файле задаются списки успешных (success events mask) и неуспешных (failure events mask) типов запросов на доступ, которые будут регистрироваться в журнале СУБД и подсистеме аудита ОС для отдельных пользователей и по умолчанию. Списки типов запросов на доступ задаются в виде шестнадцатеричных чисел, в которых каждому типу запроса соответствует установленный (для регистрируемых запросов) или сброшенный (для не регистрируемых запросов) бит (таблица 34).

Таблица 34

Тип запроса	Описание	Бит	Шестнадцатеричное значение
SUBJECT	Добавление/изменение/удаление пользователей и групп	0	1
CONFIGURATION	Изменение конфигурации, влияющей на доступ к данным (запрос на изменение значения переменной ac_session_maclabel)	1	2
RIGHTS	Изменение прав доступа к объектам БД	2	4
CHECK_RIGHTS	Модификация прав доступа к объектам БД	3	8
SELECT	Выборка информации из БД	4	10
INSERT	Добавление информации в БД	5	20
UPDATE	Изменение информации в БД	6	40
DELETE	Удаление информации из БД	7	80
TRUNCATE	Очистка данных	8	100
REFERENCES	Задание столбца таблицы в качестве внешнего ключа	10	400
TRIGGER	Добавление триггера к таблице	11	800
EXECUTE	Запуск хранимой процедуры или триггера	12	1000
USAGE	Использование объекта БД	13	2000
CREATE	Создание объектов в БД	16	10000
CREATE_TEMP	Создание временных объектов в БД	17	20000
DROP	Удаление объектов БД	18	40000
ALTER	Изменение объекта БД	19	80000
CONNECT	Соединение пользователя с БД	30	40000000
DISCONNECT	Разъединение пользователя с БД	31	80000000

Информация о соединении пользователей с БД (CONNECT) и разъединении с ней (DISCONNECT) регистрируется всегда, при условии, что список событий не установлен в 0.

6.3.2. Настройка маски регистрации событий

В СУБД PostgreSQL версии 9.6 маска регистрации событий устанавливается в процессе авторизации пользователя согласно выбранному режиму работы подсистемы регистрации событий и находится в атрибуте сессии ac_session_audit.

При этом реализован следующий порядок применения настроек регистрации событий:

1) настройки для конкретной роли и конкретной базы данных;
2) настройки для конкретной роли;
3) настройки для конкретной базы данных;

4) для всех остальных.

Маска регистрации событий имеет вид {УСПЕХ:ОТКАЗ}, где УСПЕХ — список успешных событий, ОТКАЗ — список неуспешных событий. Она может быть задана с помощью буквенных кодов или с помощью шестнадцатеричного числа. Вывод маски производится в тестовом виде.

В таблице 35 приведено соответствие между событиями, буквенным и шестнадцатеричным значением маски регистрации событий.

Таблица 35

Событие	Символ	Шестнадцатеричное значение	Описание
SUBJECT	S	1	Добавление/изменение/удаление пользователей и групп
CONFIGURATION	s	2	Изменение конфигурации, влияющей на доступ к данным (запрос на изменение значения переменной ac_session_maclabel)
RIGHTS	R	4	Запрос на модификацию прав доступа к объектам БД
CHECK_RIGHTS	V	8	Проверка прав доступа
SELECT	r	10	Выборка информации из БД
INSERT	a	20	Добавление информации в БД
UPDATE	w	40	Изменение информации в БД
DELETE	d	80	Удаление информации из БД
TRUNCATE	D	100	Очистка данных
REFERENCES	x	400	Задание столбца таблицы в качестве внешнего ключа
TRIGGER	t	800	Добавление триггера к таблице
EXECUTE	X	1000	Запуск хранимой процедуры или триггера
USAGE	U	2000	Использование объекта БД
CREATE	C	10000	Создание объектов в БД
CREATE TEMP	T	20000	Создание временного объекта
DROP	E	40000	Удаление объектов БД
ALTER	M	80000	Изменение объекта БД
CONNECT	c	40000000	Запрос на начало сессии
DISCONNECT	e	80000000	Запрос на окончание сессии
Зарезервированный символ	*	C00F3DFF	Полная маска протоколирования
Зарезервированный символ	O	0	Протоколирование отключено

Атрибут сессии ac_session_audit может быть изменен только администратором с помощью команды SET:

```
SET ac_session_audit TO 'новое_значение';
```

и просмотрен с помощью команды:

```
SHOW ac_session_audit;
```

Для просмотра маски сессии используется следующая команда:

```
SELECT session_audit;
```

Для просмотра текущей маски используется следующая команда:

```
SELECT current_audit;
```

Для конвертации маски протоколирования из текстового (буквенного) в шестнадцатеричное значение и из шестнадцатеричного в текстовое (буквенное) используются SQL-функции text_to_auditmask(TEXT) и auditmask_to_text(TEXT) соответственно. Например:

Пример

```
SELECT text_to_auditmask('{ace:ce}');
    text_to_auditmask
-------------------------
 {0xC0000020:0xC0000000}
(1 строка)

SELECT auditmask_to_text('{0xC0000020:0xC0000000}');
 auditmask_to_text
-------------------
 {ace:ce}
(1 строка)
```

6.3.2.1. Назначение списков регистрации событий в режиме internal

Для назначения маски событий в режиме internal используется команда ALTER ROLE:

```
ALTER ROLE { ALL | имя_роли } [ IN DATABASE имя_базы_данных ] SET
ac_session_audit TO новое_значение;
```

Для удаления списка регистраций событий используется следующая команда:

```
ALTER ROLE { ALL | имя_роли } [ IN DATABASE имя_базы_данных ] RESET
ac_session_audit;
```

При модификации маски происходит автоматическое обновление атрибута ac_session_audit.

Примечание. Для выполнения приведенных команд требуются права администратора.

Примечание. При инициализации кластера баз данных автоматически добавляются следующие правила:

```
ALTER ROLE postgres SET ac_session_audit TO '{SsRawdCTEMce:ce}';
ALTER ROLE ALL SET ac_session_audit TO '{SsRDxCTEMce:SsRVrawdDxtXUCTEMce}';
```

6.3.2.2. Назначение списков регистрации событий в режиме external

Для назначения маски событий в режиме `external` используется конфигурационный файл `pg_audit.conf` следующего вида:

```
success events mask = value failure events mask = value user =
имя_пользователя database = имя_базы_данных
success events mask = value failure events mask = value user =
имя_пользователя
success events mask = value failure events mask = value
```

Примечание. Любые изменения этого файла будут применены только при перезапуске сервера.

6.3.2.3. Назначение списков регистрации событий в режимах external, internal и internal,external

Загрузка маски регистрации событий в режиме `external,internal` двухэтапная: сначала выполняется загрузка маски регистрации событий из файла, после чего дополняется настройками из `pg_db_role_settings`, если в `pg_db_role_settings` есть более точные настройки. Для изменения маски регистрации событий сессии могут быть использованы команды из 6.3.2.1.

Аналогично и для режима `internal,external`.

6.3.2.4. Назначение списков регистрации событий в режиме none

В режиме `none` регистрация событий отключена, однако, администратор может изменять маску регистрации событий с помощью команд из 6.3.2.1.

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. ЗАЩИТА ПАМЯТИ

8.1. 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).

Дополнительные возможности по очистке остаточной информации предоставляет ядро с усиленной самозащитой Hardened. Набор изменений и опций ядра Hardened обеспечивает:

- очистку остаточной информации в ядерном стеке (STACKLEAK);
- очистку остаточной информации в ядерной куче (PAGE_POISONING).

В ОС реализован механизм, который очищает неиспользуемые блоки ФС непосредственно при их освобождении. Работа данного механизма снижает скорость выполнения операций удаления и усечения размера файла. Данные любых удаляемых/урезаемых файлов в пределах заданной ФС предварительно очищаются предопределенной или псевдослучайной маскирующей последовательностью. Механизм является настраиваемым и позволяет обеспечить работу ФС ОС (Ext2/Ext3/Ext4) в одном из следующих режимов:

1) очистка осуществляется посредством 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 посредством 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, равному `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 посредством 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, равному 6, в список параметров монтирования добавляется параметр secdelrnd=6.

Пример

```
/dev/sda1 /home ext4 acl,defaults,secdelrnd=6 0 2
```

Установка параметра монтирования для очистки блоков памяти при их освобождении может быть выполнена с использованием графической утилиты fly-admin-smc, запущенной администратором. Более подробное описание утилиты см. в электронной справке.

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
```

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"
```

Настройка очистки разделов страничного обмена при выключении системы может быть выполнена с использованием графической утилиты `fly-admin-smc`, запущенной администратором. Более подробное описание утилиты см. в электронной справке.

8.2. Средства ограничения прав доступа к страницам памяти

Средства ограничения прав доступа к страницам памяти реализованы на основе стандартных возможностей ядра, а также набора изменений и особых параметров ядра ОС (hardened). 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;
- рандомизацию адресного пространства процесса.

Выполнение произвольного кода обеспечивается на основе 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 (рандомизация расположения виртуального адресного пространства) и KASLR (рандомизация адресного пространства ядра). При наличии аппаратной поддержки NX-бита и PCID на новых 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()`.

9. КОНТРОЛЬ ЦЕЛОСТНОСТИ

Для обеспечения контроля целостности (в т. ч. контроля целостности КСЗ) в ОС реализованы:

- средство подсчета контрольных сумм файлов и оптических дисков (9.1);
- средство подсчета контрольных сумм файлов в deb-пакетах (9.2;
- средство контроля соответствия дистрибутиву (9.3);
- средства регламентного контроля целостности (9.4);
- средства создания замкнутой программной среды (16.1).

Для решения задач контроля целостности предназначена библиотека `libgost`, в которой для вычисления контрольных сумм реализованы функции хэширования в соответствии с ГОСТ Р 34.11-94, ГОСТ Р 34.11-2012 с длиной хэш-кода 256 бит и ГОСТ Р 34.11-2012 с длиной хэш-кода 512 бит. Названная библиотека используется в средствах подсчета контрольных сумм файлов и оптических дисков, контроля соответствия дистрибутиву и регламентного контроля целостности, модулях аутентификации.

В ОС реализован механизм, обеспечивающий проверку неизменности и подлинности загружаемых исполняемых файлов формата ELF. Проверка производится на основе контрольных сумм, вычисляемых в соответствии с ГОСТ Р 34.11-94 и ГОСТ Р 34.11-2012 с длиной хэш-кода 256 бит, и ЭЦП, реализованной в соответствии с ГОСТ Р 34.10-2001 и ГОСТ Р 34.10-2012, которые внедрены в исполняемые файлы формата ELF в процессе сборки ОС. Данный механизм предназначен для выявления фактов несанкционированного изменения исполняемых файлов формата ELF (в т. ч. относящихся к КСЗ) и предотвращения их загрузки.

В ОС реализован механизм, обеспечивающий проверку неизменности и подлинности файлов. Проверка производится на основе контрольных сумм, вычисляемых в соответствии с ГОСТ Р 34.11-94 и ГОСТ Р 34.11-2012 с длиной хэш-кода 256 бит, и ЭЦП, реализованной в соответствии с ГОСТ Р 34.10-2001 и ГОСТ Р 34.10-2012, которые внедряются в расширенные атрибуты файловой системы. Данный механизм предназначен для выявления фактов несанкционированного изменения исполняемых файлов и предотвращения их открытия.

9.1. Средство подсчета контрольных сумм файлов и оптических дисков

Для подсчета контрольных сумм файлов и оптических дисков в состав ОС включена утилита командной строки `gostsum`. Для вывода информации о синтаксисе утилиты `gostsum` необходимо выполнить команду:

```
gostsum -h
```

Синтаксис:

```
gostsum [КЛЮЧ] ... [ФАЙЛ]
```

Опции приведены в таблице 36.

Таблица 36

Опция	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 хэш-кода 256 бит (по умолчанию)
`--gost-2012-512`	Устанавливает, что будет использован алгоритм ГОСТ Р 34.11-2012 с длиной хэш-кода 512 бит
`-b`	Устанавливает размер блоков, которыми будет считываться файл
`-o`	Задает имя файла для вывода контрольной суммы (по умолчанию — стандартный поток вывода)
`-d`	Задает имя файла устройства чтения оптических дисков (файла с образом оптического диска) для подсчета контрольной суммы
`-t`	Тестирование алгоритмов подсчета контрольных сумм
`-p`	Тестирование алгоритмов подсчета контрольных сумм в многопоточной среде
`-h [--help]`	показать эту справку и выйти

Далее приведен пример подсчета контрольной суммы оптического диска:

```
gostsum -d /dev/cdrom
```

9.2. Средство подсчета контрольных сумм файлов в deb-пакетах

Для подсчета контрольных сумм файлов в deb-пакетах в состав ОС включена утилита командной строки `gostsum_from_deb`. Для вывода информации о синтаксисе утилиты `gostsum_from_deb` необходимо выполнить команду:

```
gostsum_from_deb -h
```

Синтаксис:

```
gostsum_from_deb [gostsum аргументы] [-d директория] [-p deb-пакет]
```

Опции приведены в таблице 37.

Таблица 37

Опция	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 хэш-кода 256 бит (по умолчанию)
`--gost-2012-512`	Устанавливает, что будет использован алгоритм ГОСТ Р 34.11-2012 с длиной хэш-кода 512 бит
`gostsum arguments`	аргументы утилиты `gostsum`
`-d директория`	Задает имя каталога, содержащего deb-пакеты, для файлов в которых вычисляются контрольные суммы

Окончание таблицы 37

Опция	Описание
`-p deb-пакет`	Задает имя deb-пакета, для файлов которого вычисляются контрольные суммы

9.3. Средство контроля соответствия дистрибутиву

Средство контроля соответствия дистрибутиву предоставляет возможность для контроля соответствия объектов ФС ОС дистрибутиву. Для обеспечения контроля целостности объектов ФС ОС (в т. ч. СЗИ) в состав дистрибутива входит файл `gostsums.txt` со списком контрольных сумм по ГОСТ Р 34.11-2012 с длиной хэш-кода 256 бит для всех файлов, входящих в пакеты программ дистрибутива. Используя графическую утилиту `fly-admin-int-check`, можно провести вычисление контрольных сумм файлов системы и проверку соответствия полученных контрольных сумм файлов системы эталонным контрольным суммам. Более подробное описание утилиты см. в электронной справке.

9.4. Средства регламентного контроля целостности

Организация регламентного контроля целостности ОС, прикладного ПО и СЗИ обеспечивается набором программных средств на основе «Another File Integrity Checker». 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 34.11-94.

Эталонные значения контрольных сумм и атрибутов фалов хранятся в БД. База контрольных сумм и атрибутов может быть создана при помощи команды:

```
afick -i
```

Для вычисления контрольных сумм могут использоваться алгоритмы: MD5-Digest, SHA1 и ГОСТ Р 34.11-2012 с длиной хэш-кода 256 бит.

9.4.1. Настройка

Для настройки достаточно параметров, которые указаны в конфигурационном файле по умолчанию (`etc/afick.conf`). Кроме различных путей, например, к файлам БД:

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

где содержится указание о том, какие файлы/каталоги подвергаются контролю целостности и с какими правилами.

Правило `PARSEC` выглядит следующим образом:

```
PARSEC = p+d+i+n+u+g+s+b+md5+m+e+t
```

где p+d+i+n+u+g+s+b+md5+m означает слежение за всеми стандартными атрибутами файла и использование хэш-функции MD5-Digest для слежения за целостностью содержимого файлов. +e+t означает контроль расширенных атрибутов: мандатной метки и флагов аудита, соответственно. Контроль ACL осуществляется при установке флага +g.

Правило GOST выглядит следующим образом:

```
GOST = p+d+i+n+u+g+s+b+gost+m+e+t
```

где p+d+i+n+u+g+s+b+gost+m означает слежение за всеми стандартными атрибутами файла и использование хэш-функции ГОСТ Р 34.11-2012 с длиной хэш-кода 256 бит для слежения за целостностью содержимого файлов. +e+t означает контроль расширенных атрибутов: мандатной метки и флагов аудита, соответственно. Контроль ACL осуществляется при установке флага +g.

Правило для каталогов:

```
DIR = p+i+n+u+g
```

Правило означает слежение за правами доступа, метаданными, количеством ссылок и другими стандартными атрибутами (подробнее см. /etc/afick.conf).

В файле конфигурации задаются пути к файлам и каталогам, контролируемых afick, например:

```
/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
```

Кроме того, на выбор администратора представлен ряд дополнительных путей с правилами. Соответствующие строки помечены знаком комментария # и могут быть активированы снятием этого знака.

При запуске afick с параметром -i:

```
afick -i
```

будет создан файл /var/lib/afick/afick. Это и есть БД формата ndbm. Если посмотреть ее содержимое, то можно обнаружить набор строк, каждая из которых — имя файла и далее

через пробел его атрибуты и сигнатуры.

БД защищается системой разграничения доступа.

При запуске AFICK автоматически установит 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 есть простой язык макросов, который используется при определении переменных для заданий системного планировщика заданий cron.

Необходимо обеспечить с использованием аппаратно-программных модулей доверенной загрузки и утилиты afick контроль целостности следующих объектов:

1) файлы образов ядра ОС:

/boot/vmlinuz-*

2) файлы образов временной файловой системы, используемой ядром ОС при начальной загрузке (добавляются в список контроля целостности после создания после выполнения всех необходимых операций по настройке, требующих обновления образов временной файловой системы):

/boot/initrd.img-*

3) конфигурационный файл меню загрузчика grub:

/boot/grub/menu.lst

4) кофигурационный файл, определяющий используемый по умолчанию графический дисплейный менеджер:

/etc/X11/default-display-manager

5) конфигурационный файл настройки файловых системы, доступных для монтирования через NFS:

/etc/exports

6) конфигурационный файл, содержащий информацию о различных устройствах хранения и файловых системах:

/etc/fstab

7) файл, содержащий перечень локальных групп ОС (добавляется в список контроля целостности после создания всех необходимых групп):

/etc/group

8) скрипты для запуска сервисов в каталоге

/etc/init.d/

9) конфигурационный файл, определяющий параметры работы первого процесса пользовательского режима init:

/etc/inittab

10) конфигурационные файлы, определяющие 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) файл, содержащий перечень локальных пользователей ОС (добавляется в список контроля целостности после создания всех необходимых пользователей):

```
/etc/passwd
```

12) символические ссылки на скрипты для запуска сервисов в каталогах:

```
/etc/rc*
```

13) конфигурационный файл, определяющий перечень терминалов, с которых суперпользователь root может регистрироваться в системе:

```
/etc/securetty
```

14) конфигурационный файл, определяющий перечень регистрируемых оболочек в ОС:

```
/etc/shells
```

15) конфигурационный файл, содержащий значения параметров ядра:

```
/etc/sysctl.conf
```

16) модули ядра, входящие 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-модули в каталоге:

```
/lib/security/pam
```

18) системные исполняемые файлы в каталоге:

```
/sbin/
```

19) исполняемые файлы в каталогах:

```
/bin/
/sbin/
/usr/bin/
/usr/sbin/
```

10. НАДЕЖНОЕ ФУНКЦИОНИРОВАНИЕ

10.1. 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.

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 в `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 с резервной копии.

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 с резервной копии.

После серьезного повреждения ФС, когда компьютер невозможно перезагрузить, существует возможность восстановления без переустановки ОС. Для этого следует установить DVD-диск с дистрибутивом ОС в устройство чтения DVD-дисков и начать процедуру переустановки. Дождаться появления на экране монитора: «Информация о документации» и одновременно нажать клавиши **<Alt+F2>**. Произойдет переход в режим командной строки под управлением ядра, загруженного с DVD-диска. Затем ввести команду:

```
fdisk  \-l
```

На экране монитора должна появиться 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.) Определить имя раздела, в который была установлена ОС, и ввести следующую последовательность команд:

```
cd  /mnt/
mkdir  hard
mount  /dev/имя_раздела  /mnt/hard
```

В результате указанный раздел жесткого диска будет смонтирован во вновь созданной ФС. Затем ввести команду:

```
chroot  /mnt/hard
```

После этого можно будет использовать командную оболочку ОС и выполнить необходимые действия по восстановлению (например, редактирование файла `fstab`), после чего ввести команды:

```
exit
umount  /mnt/hard
```

Перезагрузить компьютер.

10.2. Средства резервного копирования и восстановления ОС

Резервное копирование выполняется с целью получения копий данных, сохраняемых на случай их потери или разрушения. Подобные копии должны создаваться периодически, в соответствии с заранее установленным графиком. Схемы резервного копирования изменяются в зависимости от размеров и степени охвата резервным копированием ОС, а также от выдвигаемых требований по надежности сохранения жизнеспособности системы. Элементы системы резервного копирования должны включать необходимое оборудование, носители резервных копий и СПО. В качестве оборудования для резервного копирования в ОС может использоваться достаточно широкий набор аппаратных средств, начиная от USB-накопителя и заканчивая библиотекой ленточных устройств. Тип и количество носителей определяются используемым оборудованием, объемами обрабатываемых данных и выбранной схемой резервирования данных. ПО резервного копирования, включенное в состав ОС, является очень разнородным, начиная от простых команд типа `tar`, `cpio`, `gzip` и заканчивая распределенными системами управления хранилищами данных.

Резервное копирование информации используется для:

- восстановления файлов, случайно удаленных пользователями или утерянных из-за отказов устройств хранения;
- получения периодически создаваемых моментальных снимков (snapshots) состояния данных;
- получения данных для восстановления после аварий.

Система резервного копирования является составной частью плана восстановления системы.

Основная идея резервного копирования — создание копий критической части содержания резервируемой системы. Основными исключениями, как правило, не входящими в процедуру резервного копирования функционирующей ОС, являются каталоги, содержащие служебные данные, меняющиеся в процессе функционирования (`/dev`, `/media`, `/mnt`, `/parsecfs`, `/proc`, `/run`, `/sys`, `/tmp`), а также сетевые каталоги (смонтированная NFS,

Samba и прочие виды сетевых данных).

В состав ОС входит множество средств, обеспечивающих решение различных задача резервного копирования данных. Утилиты `tar`, `cpio`, `gzip` представляют собой традиционные инструменты создания резервных копий и архивирования ФС. При создании архива командами `tar` и `gzip` передается список файлов и каталогов, указываемых как параметры командной строки. Любой указанный каталог просматривается рекурсивно. При создании архива с помощью команды `cpio` ей предоставляется список объектов (имена файлов и каталогов, символические имена любых устройств, гнезда доменов UNIX, поименованные каналы и т. п.). Описание команд приведено в руководстве `man` для команд `tar`, `rsync`, `cpio` и `gzip`.

Утилита `rsync` предоставляет возможности для локального и удаленного копирования (резервного копирования) или синхронизации файлов и каталогов, с минимальными затратами трафика.

Кроме того в состав ОС входит комплекс программ Bacula, предназначенных для решения различных задач резервного копирования и восстановления данных.

Для выполнения операций резервного копирования и восстановления объектов ФС с сохранением и восстановлением мандатных атрибутов и атрибутов аудита в ОС можно использовать комплекс программ Bacula, утилиту `rsync` или утилиту `tar`.

ВНИМАНИЕ! Работа с мандатными атрибутами и атрибутами аудита при использовании различных утилит создания резервных копий требует опций сохранения расширенных атрибутов (как правило вида `–xattrs`).

ВНИМАНИЕ! Для восстановления мандатных атрибутов файлов из резервных копий необходимо от имени учетной записи администратора выполнить команду:

```
sudo echo 1 > /parsecfs/unsecure_setxattr
```

ВНИМАНИЕ! Для восстановления мандатных атрибутов файлов из резервных копий процесс должен иметь PRASEC-привилегию `0x1000`. Привилегия может быть получена с использованием утилиты `execaps`:

```
sudo execaps –c 0x1000 tar .......
```

После восстановления из резервных копий файлов с мандатными атрибутами необходимо от имени учетной записи администратора выполнить команду:

```
sudo echo 0 > /parsecfs/unsecure_setxattr
```

Рассмотрим пример создания и восстановления резервной копии с использованием утилиты `tar`.

ВНИМАНИЕ! Предполагается, что уже создан пользователь `user1`, для которого заданы мандатные атрибуты и пользователь уже выполнял вход в систему.

Создание администратором архива домашнего каталога пользователя может быть выполнено с помощью команды:

```
sudo tar --xattrs --acls -cvzf /opt/home.tgz /home/.pdp/user1
```

Опция `--xattrs` означает включение поддержки расширенных атрибутов. Опция `--acls` означает включение поддержки POSIX ACL. Опции `-cvzf` необходимы для создания архива (`create`), включения режима отображения обрабатываемых файлов (`verboze`), применения метода сжатия (`gzip`), указания файла (`file`) соответственно. Путь `/opt/home.tgz` задает место расположения созданного архива и его имя, путь `/home/.pdp/user1` определяет, что именно будет вложено в архив.

Восстановление выполняется с помощью команды:

```
sudo execaps -c 0x1000 -- tar --xattrs
    --xattrs-include=security.{PDPL,AUDIT,DEF_AUDIT}
    --acls -xvf /opt/home.tgz -C /opt/home2/
```

Опция `--xattrs-include=security.{PDPL,AUDIT,DEF_AUDIT}` определяет подключаемый шаблон восстановления расширенных атрибутов (мандатных атрибутов, атрибутов аудита и атрибутов аудита по умолчанию) для ключа `xattrs`. Опции `-xvf` необходимы для извлечения из архива (`extract`), включения режима отображения обрабатываемых файлов (`verboze`), указания файла (`file`) соответственно.

10.2.1. Комплекс программ Bacula

`Bacula` представляет собой набор программ, позволяющий системному администратору управлять процессами резервного копирования и восстановления данных, а также проверять резервные копии, в т. ч. в гетерогенных сетях.

`Bacula` — это сетевая клиент-серверная система резервного копирования. Программа обладает множеством возможностей, позволяющих легко находить и восстанавливать утраченные или поврежденные файлы. Из-за своей модульной архитектуры `Bacula` может масштабироваться от небольших автономных систем до больших сетей, состоящих из сотен компьютеров.

`Bacula` состоит из следующих составных частей:

- `Bacula Director service` — центральная программа, координирующая все выполняемые операции (функционирует в фоне);
- `Bacula Console services` — программа, позволяющая администратору взаимодействовать с центральной программой;
- `Bacula File services` — клиентская программа, устанавливаемая на каждом обслуживаемом компьютере;
- `Bacula Storage services` — программа, обычно функционирующая на компьютере, к которому присоединены внешние устройства для хранения резервных копий;
- `Catalog services` — программа, отвечающая за индексирование и организацию

базы резервных данных.

Программа Bacula обеспечивает поддержку сохранения расширенных атрибутов каталогов и файлов и, при необходимости, их последующее восстановление.

ВНИМАНИЕ! Для восстановления объектов ФС с установленными мандатными атрибутами необходимо запустить консоль управления Bacula с PARSEC-привилегией 0x1000, выполнив команду:

```
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.

Описание установки и настройки Bacula приведено в документе РУСБ.10015-01 95 01-1.

10.3. Восстановление СУБД PostgreSQL после сбоев и отказов

Во избежание потерь данных БД PostgreSQL должны регулярно архивироваться.

В случае возникновения ошибок в хранящихся данных, нарушению целостности или в случае программного и/или аппаратного сбоя сервера БД необходимо проведение процедуры восстановления БД. При этом, в зависимости от тяжести повреждений может осуществляться как сохранение существующего кластера БД, с последующим его восстановлением, так и восстановление из резервных копий, созданных в процессе регулярного проведения регламентных работ.

В PostgreSQL существуют три фундаментально отличающихся подхода к резервному копированию данных:

- SQL-дамп;
- резервное копирование на уровне ФС;
- непрерывное архивирование.

Более подробное описание этих методов и процедур копирования и восстановления приведено в документации на СУБД PostgreSQL в пакете postgresql-doc-x.x.

СУБД PostgreSQL содержит ряд стандартных средств резервного копирования и восстановления БД. К ним относятся утилиты pg_dump (10.3.2), pg_dumpall (10.3.3), pg_restore (10.3.4) и, в том числе, интерактивный терминал psql, с помощью которого могут быть восстановлены резервные копии, сохраненные в виде скрипта SQL.

10.3.1. Создание и восстановление резервных копий баз данных с мандатными атрибутами

Для создания и восстановления резервных копий баз данных с мандатными атрибутами необходимо, чтобы пользователь имел привилегии parsec_cap_setmac,

parsec_cap_chmac.

В случае создания резервной копии необходимо назначить максимальную метку на каталог, в который будет выгружена базы данных (для назначения метки на файл копии).

В случае восстановления помимо указанных привилегий требуется права администратора в базе данных (для создания объектов и назначения мандатных атрибутов).

Для восстановления копии базы данных с мандатными атрибутами в другой базе данных, необходимо:

1) назначить максимальную метку на каталог, в который будет проводиться выгрузка резервной копии;

2) создать резервную копию исходной базы данных от имени пользователя с привилегиями parsec_cap_setmac и parsec_cap_chmac.

3) на целевом кластере назначить мандатные атрибуты на кластер;

4) восстановить резервную копию базы данных с помощью клиента psql (если резервная копия сделана в текстовом виде) или с помощью pg_restore (если резервная копия сделана в бинарном виде) от имени пользователя администратора базы данных с привилегиями parsec_cap_setmac и parsec_cap_chmac.

Примечания:

1. Утилита pg_dump, поставляемая вместе с СУБД версии 9.6 выгружает команды по установке комментариев, меток безопасности и назначению мандатных атрибутов в следующем виде:

```
COMMENT ON DATABASE CURRENT_DATABASE IS 'комментарий';
SECURITY LABEL ON DATABASE CURRENT_DATABASE IS '...';
MAC LABEL ON DATABASE CURRENT_DATABASE IS '...';
MAC CCR ON DATABASE CURRENT_DATABASE IS '...';
```

Такая резервная копия может быть восстановлена с установкой комментариев, меток безопасности и мандатных атрибутов в желаемую базу данных.

2. Для восстановления резервных копий баз данных, сделанных в предыдущих версиях, в СУБД версии 9.6 необходимо установить параметр ac_auto_adjust_macs = true;

3. Для переноса кластера с предыдущих версий СУБД на 9.6 необходимо воспользоваться утилитами pg_upgradecluster или pg_upgrade (см. документ «Операционная система специального назначения «Astra Linux Special Edition» Руководство администратора. Часть 2»).

10.3.2. pg_dump

Для создания резервной копии БД в виде файла в текстовом или других форматах используется утилита pg_dump, которая создает согласованную копию, даже если БД ис-

пользуется, при этом доступ к ней других пользователей (как читающих, так и пишущих) не блокируется.

Резервная копия может создаваться в виде скрипта или форматах упакованного файла. Скрипт резервной копии представляет собой текст, содержащий последовательность SQL-команд, необходимых для воссоздания БД до состояния, в котором она была сохранена. Для восстановления из скрипта он подается на вход утилиты `psql`. Скрипт может быть использован для воссоздания БД даже на другом сервере или архитектуре и с небольшими изменениями на других СУБД.

Синтаксис:

```
pg_dump [OPTION]... [DBNAME]
```

Опции общего характера приведены в таблице 38.

Таблица 38

Опция	Описание
`-f, --file=FILENAME`	Имя выходного файла
`-F, --format=c|t|p`	Формат выходного файла (пользовательский, `tar`, текстовый)
`-v, --verbose`	Режим вывода всех сообщений
`-Z, --compress=0-9`	Уровень сжатия для форматов сжатия
`--lock-wait-timeout=TIMEOUT`	Завершение ошибкой после ожидания `TIMEOUT` для блокировки таблицы
`--help`	Вывести справку и выйти
`--version`	Вывести информацию о версии и выйти

Если не используется `-f/--file`, SQL-скрипт будет направлен в стандартный вывод.

Опции установки соединения приведены в таблице 39.

Таблица 39

Опция	Описание
`-h, --host=HOSTNAME`	Имя сервера БД или каталог сокетов
`-l, --database=DBNAME`	Указать альтернативную БД — шаблон
`-p, --port=PORT`	Номер порта сервера БД
`-U, --username=NAME`	Соединиться как указанный пользователь
`-w, --no-password`	Не запрашивать пароль
`-W, --password`	Принудительный запрос пароля (должен происходить автоматически)

Для получения полной информации о способах использования утилиты `pg_dump`

см. руководство man для утилит pg_dump и psql.

10.3.3. pg_dumpall

Утилита pg_dumpall используется для создания резервной копии всего кластера в виде скрипта.

Скрипт содержит SQL-команды и может быть подан в дальнейшем на вход утилиты psql для восстановления. Операция осуществляется последовательным вызовом утилиты pg_dump для каждой БД кластера. Кроме этого, pg_dumpall сохраняет глобальные объекты, единые для всех БД (pg_dump подобные объекты не сохраняет). Данные объекты включают в себя информацию о пользователях и группах и такие свойства, как: права доступа, применяемые для всех БД в целом.

Синтаксис:

```
pg_dumpall [OPTION]...
```

Опции общего характера приведены в таблице 40.

Таблица 40

Опция	Описание
-f, --file=FILENAME	Имя выходного файла
--lock-wait-timeout=TIMEOUT	Завершение ошибкой после ожидания TIMEOUT для блокировки таблицы
--help	Вывести справку и выйти
--version	Вывести информацию о версии и выйти

Опции установки соединения аналогичны команде pg_dump (см. таблицу 39).

Для получения полной информации о способах использования утилиты pg_dumpall см. руководство man для утилит pg_dumpall и psql.

10.3.4. pg_restore

Для восстановления архивов резервных копий БД, полученных с помощью утилиты pg_dump, используется утилита pg_restore. Она выполняет команды, необходимые для воссоздания БД до состояния на момент времени создания резервной копии. Архивные файлы так же позволяют выбирать с помощью утилиты pg_restore, что именно восстанавливать, и даже менять порядок восстанавливаемых элементов. Файлы архивов разработаны переносимыми между разными архитектурами.

Утилита pg_restore может функционировать в двух режимах. При указании БД архив восстанавливается непосредственно в нее. В другом случае, скрипт, содержащий необходимые для пересоздания БД SQL-команды, создается и выводится в файл или стандартный поток вывода. Результирующий скрипт эквивалентен формату текстового вывода утилиты pg_dump. Вследствие этого некоторые опции, управляющие выводом,

аналогичны опциям pg_dump (см. 10.3.2).

Синтаксис:

```
pg_restore [ОПЦИЯ]... [ФАЙЛ]
```

Опции общего характера приведены в таблице 41.

Таблица 41

Опция	Описание
-d, --dbname=ИМЯ	Подсоединиться к указанной БД
-f, --file=FILENAME	Имя входного файла
-F, --format=c|t	Формат файла резервной копии (должно быть автоматически)
-l, --list	Напечатать итоговое оглавление архива
-v, --verbose	Режим вывода всех сообщений
--help	Вывести справку и выйти
--version	Вывести информацию о версии и выйти

Для получения полной информации о способах использования утилиты pg_restore см. руководство man для утилиты pg_restore.

11. ФИЛЬТРАЦИЯ СЕТЕВОГО ПОТОКА

11.1. Включение фильтрации сетевого потока

При помощи фильтра сетевых пакетов можно осуществлять контроль сетевого трафика, проходящего через данный компьютер.

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.

11.2. Фильтр сетевых пакетов `iptables`

Фильтрацию пакетов выполняет фильтр пакетов на основе технологии `iptables`. Данный фильтр позволяет выполнять следующие задачи:

1) фильтрацию пакетов — это механизм, который, основываясь на некоторых правилах, разрешает или запрещает передачу информации, проходящей через него, с целью ограждения некоторой подсети от внешнего доступа, или, наоборот, для недопущения выхода наружу. Фильтр пакетов может определять правомерность передачи информации на основе только заголовков IP-пакетов, а 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 ЛВС, а также для организации доступа из ЛВС 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 заставить пользователей из ЛВС пользоваться proxy-сервером маршрутизатора без дополнительного конфигурирова-

ния их клиентских программ.

Настройка рассмотренных механизмов (фильтрация пакетов, трансляция сетевых адресов и прозрачное проксирование) выполняется командой `iptables`.

11.3. Формирование правил

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

Правила записываются следующим образом:

```
iptables [-t table] command [match] [target/jump]
```

Если в правило не включается спецификатор `[-t table]`, то по умолчанию предполагается использование таблицы `filter`, если же предполагается использование другой таблицы, то это требуется указать явно. Спецификатор таблицы также можно указывать в любом месте строки правила, однако более или менее стандартным считается указание таблицы в начале правила.

Далее, непосредственно за именем 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 и т. п.

Раздел `matches` задает критерии проверки, по которым определяется, подпадает ли пакет под действие этого правила или нет. Здесь можно указать самые разные критерии — и IP-адрес источника пакета или сети, и сетевой интерфейс, и т. д.

Параметр `target` указывает, какое действие должно быть выполнено при условии выполнения критериев в правиле. Здесь можно заставить ядро передать пакет в другую цепочку правил, «сбросить» пакет и забыть про него, выдать на источник сообщение об ошибке и т. п.

11.4. 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 и после передается либо локальному приложению, либо переправляется на другой компьютер. Порядок следования пакета приводится в таблице 42.

Таблица 42

Шаг	Таблица	Цепочка	Описание
1	=	=	Кабель
2	=	=	Сетевой интерфейс (например, `eth0`)

Окончание таблицы 42

Шаг	Таблица	Цепочка	Описание
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

Пакет проходит несколько этапов, прежде чем он будет передан далее. На каждом из них пакет может быть остановлен. Цепочку FORWARD проходят все пакеты, которые движутся через сетевой фильтр. В таблице 43 представлен порядок движения пакета, предназначенного локальному процессу/приложению.

Таблица 43

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
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

Важно помнить, что пакеты идут через цепочку INPUT, а не через FORWARD. В таблице 44 представлен порядок движения пакетов, созданных локальными процессами.

Таблица 44

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 этой таблице не следует производить любого рода фильтрацию, маскировку или преобразование адресов (DNAT, SNAT), в ней допускается выполнять действия, приведенные в таблице 45.

Таблица 45

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 имеет две цепочки:

- PREROUTING — используется для внесения изменений на входе в сетевой фильтр перед принятием решения о маршрутизации;
- OUTPUT — для внесения изменений в пакеты, поступающие от приложений внутри сетевой фильтр.

nat

Только первый пакет из потока проходит через цепочки этой таблицы. Трансляция адресов или маскировка применяются ко всем последующим пакетам в потоке автоматически. Для этой таблицы характерны действия, приведенные в таблице 46.

Таблица 46

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 имеет две цепочки:

- PREROUTING — используется для внесения изменений в пакеты на входе в сетевой фильтр;
- OUTPUT — используется для преобразования пакетов, созданных приложениями внутри сетевого фильтра, перед принятием решения о маршрутизации.

filter

В этой таблице 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 без каких-либо сложностей, как в других таблицах. Имеется три встроенных цепочки:

- FORWARD — используется для фильтрации пакетов, идущих транзитом через сетевой фильтр;
- INPUT — проходят пакеты, которые предназначены локальным приложениям (сетевому фильтру);
- OUTPUT — используется для фильтрации исходящих пакетов, сгенерированных приложениями на самом сетевом фильтре.

11.5. 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

Таблицу трассировщика можно найти в файле /proc/net/ip_conntrack. Здесь 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 этом примере содержится вся информация, которая известна трассировщику по конкретному соединению. Первое, что можно увидеть — это 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 видно ключевое слово, которое сообщает о том, что ответного трафика через это соединение еще не было.

Приводится дополнительная информация по ожидаемому пакету, это IP-адреса отправителя/получателя (те же самые, только поменявшиеся местами, т. к. ожидается ответный пакет), то же касается и портов.

После получения пакетом ответа трассировщик снимет флаг [unreplied] и заменит его флагом [assured]. Этот флаг сообщает, что соединение установлено уверенно, 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 47 подробно рассмотрены каждое из возможных состояний и приведены необходимые комментарии.

Таблица 47

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
`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 48 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 48

Команда	Использование
-A, --append	Добавляет новое правило в конец заданной цепочки. Пример iptables -A INPUT ...
-D, --delete	Удаляет правило из цепочки. Команда имеет два формата записи: первый — когда задается критерий сравнения с опцией -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
-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 пр. 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

Окончание таблицы 48

Команда	Использование
`-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. Примеры: 1. `iptables -X allowed` 2. `iptables -P INPUT DROP`
`-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 49.

Таблица 49

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

Окончание таблицы 49

Ключ	Описание
`-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 символьные множители `K` (x1000), `M` (x1,000,000) и `G` (x1,000,000,000). Для того чтобы заставить команду `--list` выводить полное число (без употребления множителей), требуется применять ключ `-x`, который описан ниже. При использовании с другими командами на вывод будет выдан 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

11.6. 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.

11.7. 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, вызвавшем 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 так же 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 будут подвергнуты преобразованию адреса назначения и переданы на требуемое устройство, компьютер или сеть.

Может выполняться только в цепочках PREROUTING и OUTPUT таблицы nat и во вложенных подцепочках.

Ключ для действия DNAT — --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 опции --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 50.

Таблица 50

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
--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 для действия MARK — --set- — 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 для действия MASQUERADE — --to-ports — 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 для действия REDIRECT — --to-ports — 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 для действия `REJECT` — `--reject-with` — 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. изменение исходящего IP-адреса в IP-заголовке пакета. `SNAT` допускается выполнять только в таблице `nat`, в цепоч-

ке 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 для действия SNAT — --to-source — 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 для действия TOS — --set-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 51.

Таблица 51

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 специальному демону, который 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 52.

Таблица 52

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 аналогичная опция 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. МАРКИРОВКА ДОКУМЕНТОВ

Защищенный комплекс программ печати и маркировки документов обеспечивает маркировку выводимых на печать документов.

Дополнительно к дискреционному контролю доступа на основе политик сервера CUPS используется мандатный контроль доступа. Мандатные атрибуты порождаемых объектов в CUPS наследуют мандатный контекст, получаемый с сетевого соединения.

Все субъекты доступа (пользователи) сервера печати делятся на 2 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 53 приведено разделение операций по группам доступа и типам операций доступа.

Таблица 53

	Чтение	Запись

Административные операции	`CUPS-Get-Devices`
	`CUPS-Get-PPDs`
	`CUPS-Get-PPD`
	`CUPS-Get-Policies`
		`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 53

	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	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

Принтеры и классы являются объектами-контейнерами, в которые помещаются задания. Они обладают атрибутами `mac-printer-maclabel` и `pmac-printer-lower-maclabel`.

Атрибут `mac-printer-maclabel` определяет максимальный мандатный контекст заданий, которые могут находиться в нем.

Атрибут `mac-printer-lower-maclabel` определяет минимальный мандатный контекст заданий, которые могут находиться в нем.

Все задания, приобретающие ненулевой мандатный контекст задерживаются в сервером до проведения специальным привилегированным пользователем маркировки документа. Файлы заданий (в каталоге `/var/spoo/cups`) маркируются согласно их мандатному контексту.

Конфигурационные параметры сервера CUPS определены в файле `/etc/cups/cupsd.conf`. Для управления конфигурационными параметрами сервера CUPS используется утилита командной строки `cupsctl`, запускаемая от имени администратора через механизм `sudo`. Описание данной утилиты приведено в `man cupsctl`.

Для разрешения серверу CUPS удаленно принимать задания и команды необходимо от имени учетной записи администратора выполнить следующие команды:

```
sudo cupsctl --remote-admin --share-printers --remote-any
sudo cupsctl ServerAlias=*
```

Для разрешения серверу CUPS обрабатывать задания печати, формируемые в

ненулевом мандатном контексте, необходимо от имени учетной записи администратора выполнить следующие команды:

```
sudo cupsctl MarkerUser=ipp
sudo cupsctl DefaultPolicy=parsec
```

Для разрешения печати на принтере документов пользователей, работающих в ненулевом мандатном контексте, необходимо выполнить от имени учетной записи администратора следующие команды:

```
sudo lpadmin -p <имя_принтера> -o printer-op-policy=parsec
sudo lpadmin -p <имя_принтера> -o mac-printer-lower-maclabel=LLabel
sudo lpadmin -p <имя_принтера> -o mac-printer-maclabel=MLabel
```

где `LLabel` — минимальный контекст заданий, который заданий, а `MLabel` — максимальный контекст заданий.

Если ЕПП не используется, то команды выполняются от имени администратора через механизм `sudo`. Если ЕПП используется, то команды выполняются от имени пользователя, входящего в специальную группу администраторов печати в БД службы каталогов LDAP (см. документ РУСБ.10015-01 95 01-1).

Маркировка осуществляется на основе модифицируемых файлов шаблонов:

- `/usr/share/cups/marker.template` — описание элементов маркера, проставляемых на первой, каждой, последней странице и на обороте последней страницы;
- `/usr/share/cups/psmarker/marker.defs` — описание положения элементов маркера на странице;
- `/usr/share/cups/fonarik/fonarik.defs` — описание положения элементов маркера на обороте последей страницы.

Для установки значений атрибутов, определенных в перечисленных выше файлах шаблонов, используется утилита командной строки `lpattr`. Для применения утилиты без ЕПП необходимо наличие локальной группы `lpmac` в ОС. Для применения утилиты в ЕПП необходимо наличие группы `lpmac` в БД службы каталогов LDAP. Запуск утилиты должен осуществляться от имени пользователя, входящего в группу `lpmac`.

Описание утилит `lpadmin` и `lpattr` приведено в `man lpadmin` и `man lpattr`, соответственно.

Для управления принтерами используется графическая утилита `fly-admin-printer`. Подробное описание утилиты см. в электронной справке.

13. 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.

Для ОС возможность санкционированного монтирования конкретным пользователем конкретных носителей с конкретными 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 атрибутам осуществляется с помощью утилиты `fly-admin-smc` («Управление политикой безопасности»), в том числе и при работе в ЕПП.

Рекомендации по использованию указанных средств приведены в документе РУСБ.10015-01 95 01-1.

14. 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).

ОС включает в себя механизм, обеспечивающий надежное сопоставление мандатного контекста пользователя с мандатным уровнем и категориями, установленными для устройства. Кроме того, механизм сопоставления пользователя с устройством, реализованный в ОС, обеспечивает при проверке совпадения маркировок носителя и пользователя применение дискреционных ПРД.

15. ГЕНЕРАЦИЯ КСЗ

Генерация КСЗ осуществляется в одном из двух следующих режимов:

- режим ЕПП;
- локальный режим.

Для использования режима ЕПП необходимо наличие в сети установленного и настроенного сервера ALD. На рабочих местах пользователей в ОС должен быть установлен пакет `ald-client` и выполнены соответствующие действия по настройке (см. документ РУСБ.10015-01 95 01-1).

После определения режима работы КСЗ ОС для завершения процедуры генерации КСЗ необходимо выполнить следующие действия:

1) создать набор мандатных уровней;
2) создать набор мандатных категорий;
3) создать набор служебных пользователей, наличие которых необходимо для функционирования защищенных комплексов программ СУБД, гипертекстовой обработки данных, электронной почты и программ маркировки и учета печатных документов из состава ОС;
4) установить привилегии для служебных пользователей;
5) установить параметры аудита (протоколирования событий) в ОС.

ВНИМАНИЕ! Для изменения максимального мандатного контекста объектов (см. 4.1) необходимо внести изменения в сценарий `pdp-init-fs`, размещенный в каталоге `/usr/sbin`, переопределив значения переменных:

- `sysmaxlev` – максимальный мандатный уровнь;
- `sysmaxilev` – максимальный уровнь целостности;
- `sysmaxcat` – максимальный набор мандатных категорий.

После выполнения перечисленного набора действий осуществляется создание пользователей, установка для них разрешенных мандатных уровней и категорий, а в случае необходимости, параметров протоколирования.

В режиме ЕПП указанные действия выполняются при помощи графической утилиты `fly-admin-smc` или при помощи утилиты командной строки `ald-admin` (см. `man ald-admin`).

В локальном режиме указанные действия выполняются при помощи графической утилиты `fly-admin-smc` или при помощи соответствующих утилит командной строки (см. 4.8) от имени учетной записи администратора (см. раздел 1) через механизм `sudo`. Более подробное описание графических утилит см. в электронной справке.

16. ОГРАНИЧЕНИЕ ПРОГРАММНОЙ СРЕДЫ

16.1. Замкнутая программная среда

Инструменты замкнутой программной среды (ЗПС) предоставляют возможность внедрения цифровой подписи в исполняемые файлы формата ELF, входящие в состав устанавливаемого СПО (16.1.2), и в расширенные атрибуты файловой системы.

Механизм контроля целостности исполняемых файлов и разделяемых библиотек формата ELF при запуске программы на выполнение реализован в модуле ядра ОС `digsig_verif`, который является не выгружаемым модулем ядра Linux и может функционировать в одном из следующих режимов:

1) исполняемым файлам и разделяемым библиотекам с неверной ЭЦП, а также без ЭЦП загрузка на исполнение запрещается (штатный режим функционирования);
2) исполняемым файлам и разделяемым библиотекам с неверной ЭЦП, а также без ЭЦП загрузка на исполнение разрешается, при этом выдается сообщение об ошибке проверки ЭЦП (режим для проверки ЭЦП в СПО);
3) ЭЦП при загрузке исполняемых файлов и разделяемых библиотек не проверяется (отладочный режим для тестирования СПО).

Механизм контроля целостности файлов при их открытии на основе ЭЦП в расширенных атрибутах файловой системы также реализован в модуле ядра ОС `digsig_verif` и может функционировать в одном из следующих режимов:

1) запрещается открытие файлов, поставленных на контроль, с неверной ЭЦП или без ЭЦП;
2) открытие файлов, поставленных на контроль, с неверной ЭЦП или без ЭЦП разрешается, при этом выдается сообщение об ошибке проверки ЭЦП (режим для проверки ЭЦП в расширенных атрибутах файловой системы);
3) ЭЦП при открытии файлов не проверяется.

16.1.1. Настройка модуля digsig_verif

Настройка режима функционирования модуля `digsig_verif` осуществляется посредством графической утилиты `fly-admin-smc` («Панель управления — Безопасность — Политика безопасности — Замкнутая программная среда») или путем редактирования конфигурационного файла `/etc/digsig/digsig_initramfs.conf`. Описание использования утилиты приведено в электронной справке.

Редактирование конфигурационного файла `/etc/digsig/digsig_initramfs.conf` осуществляется следующим образом:

1) для использования отладочного режима для тестирования СПО необходимо установить для параметра `DIGSIG_ELF_MODE` значение `0`:

DIGSIG_ELF_MODE=0

2) для использования режима для проверки ЭЦП в СПО необходимо установить для параметра DIGSIG_ELF_MODE значение 2:

DIGSIG_ELF_MODE=2

3) для использования штатного режима функционирования необходимо установить для параметра DIGSIG_ELF_MODE значение 1:

DIGSIG_ELF_MODE=1

В модуль digsig_verify встроены открытые ключи АО «НПО РусБИТех», которые используются:

- для проверки подписи исполняемых файлов;
- для проверки подписи открываемых файлов в расширенных атрибутах;
- для проверки подписи на дополнительных ключах, загружаемых из /etc/digsig/keys.

В каждый момент времени модуль использует два набора ключей:

- основной набор (изначально три встроенных ключа АО «НПО РусБИТех») — для проверки любых подписей;
- дополнительный набор (изначально пустой) — только для проверки подписи открываемых файлов в расширенных атрибутах.

Если дополнительный набор не содержит ключа, который использовался для подписи файла в расширенных атрибутах, модуль ищет подходящий ключ в основном наборе.

Каталог /etc/digsig/keys содержит открытые ключи в формате gnupg --export, которыми расширяется основной набор ключей при загрузке системы. Каждый ключ, использованный для подписывания СПО (16.1.2), необходимо скопировать в каталог /etc/digsig/keys/, например, с использованием команды:

cp /<каталог>/<файл ключа> /etc/digsig/keys/

В каталоге /etc/digsig/keys/ может располагаться иерархическая структура ключей. Каждый ключ должен быть подписан уже загруженным ключом основного набора в момент его добавления в основной набор. Иерархия подкаталогов /etc/digsig/keys обрабатывается сверху вниз таким образом, что:

- файлы ключей в /etc/digsig/keys должны быть подписаны встроенными в модуль digsig_verify ключами АО «НПО РусБИТех»;
- файлы ключей в /etc/digsig/keys/subdirectory могут быть подписаны и встроенными ключами АО «НПО РусБИТех», и ключами из /etc/digsig/keys;
- в целом, каждый ключ из подкаталога внутри /etc/digsig/keys должен быть подписан либо ключом из вышележащего каталога, либо встроенным ключом АО «НПО РусБИТех».

Пример

```
/etc/digsig/keys/key1.gpg - публичный ключ 1, подписанный
на первичном ключе АО <<НПО РусБИТех>>
/etc/digsig/keys/key2.gpg - публичный ключ 2, подписанный
на первичном ключе АО <<НПО РусБИТех>>
/etc/digsig/keys/key1/key1-1.gpg - публичный ключ, подписанный на ключе 1
/etc/digsig/keys/key1/key1-2.gpg - публичный ключ, подписанный на ключе 1
/etc/digsig/keys/key2/key2-1.gpg - публичный ключ, подписанный на ключе 2
/etc/digsig/keys/key2/key2-2.gpg - публичный ключ, подписанный на ключе 2
```

Для проверки использования дополнительных ключей для контроля целостности исполняемых файлов и разделяемых библиотек формата ELF (до перезагрузки ОС) можно от имени учетной записи администратора через механизм `sudo` выполнить команды:

```
cat /etc/digsig/key_for_signing.gpg > /sys/digsig/keys
cat /etc/digsig/keys/key1.gpg >> /sys/digsig/keys
cat /etc/digsig/keys/key2.gpg >> /sys/digsig/keys
cat /etc/digsig/keys/key1/key1-1.gpg >> /sys/digsig/keys
cat /etc/digsig/keys/key1/key1-2.gpg >> /sys/digsig/keys
cat /etc/digsig/keys/key2/key2-1.gpg >> /sys/digsig/keys
cat /etc/digsig/keys/key2/key2-2.gpg >> /sys/digsig/keys
```

ВНИМАНИЕ! При проверке ЭЦП в расширенных атрибутах файловой системы установка для параметра `DIGSIG_XATTR_MODE` значения 1 запрещает открытие поставленных на контроль файлов с неверной ЭЦП или без ЭЦП.

Настройка режима функционирования механизма контроля целостности файлов при их открытии на основе ЭЦП в расширенных атрибутах файловой системы осуществляется с помощью графической утилиты `fly-admin-smc` («Панель управления — Безопасность — Политика безопасности — Замкнутая программная среда») или путем редактирования конфигурационного файла `/etc/digsig/digsig_initramfs.conf`.

Редактирование конфигурационного файла `/etc/digsig/digsig_initramfs.conf` осуществляется следующим образом:

1) для включения проверки подписей в режиме запрета открытия поставленных на контроль файлов с неверной ЭЦП или без ЭЦП в расширенных атрибутах файловой системы необходимо установить для параметра `DIGSIG_XATTR_MODE` значение 1:

```
DIGSIG_XATTR_MODE=1
```

2) для включения проверки подписей в режиме вывода предупреждений об открытии поставленных на контроль файлов с неверной ЭЦП или без ЭЦП в расширенных атрибутах файловой системы необходимо установить для параметра `DIGSIG_XATTR_MODE` значение 2:

```
DIGSIG_XATTR_MODE=2
```

3) для игнорирования дополнительных ключей, используемых только при проверке ЭЦП в расширенных атрибутах файловой, необходимо установить для параметра DIGSIG_IGNORE_XATTR_KEYS значение 1:

```
DIGSIG_IGNORE_XATTR_KEYS=1
```

4) для настройки шаблонов имен, используемых при проверке ЭЦП в расширенных атрибутах ФС, необходимо в файле /etc/digsig/xattr_control задать их список. Каждая строка задает свой шаблон в виде маски полного пути. Например, следующий шаблон определяет, что будет проверяться ЭЦП всех файлов в каталоге /bin, имя которых начинается на lo:

```
\bin\lo
```

Каталог /etc/digsig/xattr_keys содержит открытые ключи в формате gnupg --export, которыми расширяется дополнительный набор ключей модуля (изначально пустой набор, используемый только для проверки подписи в расширенных атрибутах файла). Подписи на ключах дополнительного набора не проверяются.

Каждый дополнительный ключ, использованный для подписывания файлов в расширенных атрибутах, необходимо скопировать в каталог /etc/digsig/xattr_keys/, например, с использованием команды:

```
cp /<каталог>/<файл ключа> /etc/digsig/xattr_keys/
```

В каталоге /etc/digsig/xattr_keys/ может располагаться иерархическая структура дополнительных ключей для контроля целостности файлов. В указанной структуре одни дополнительные ключи могут быть подписаны на других дополнительных ключах. При этом дополнительные ключи должны располагаться в подкаталогах таким образом, чтобы при их загрузке не нарушалась цепочка проверки подписей.

Пример

```
/etc/digsig/xattr_keys/key1.gpg - публичный ключ 1
/etc/digsig/xattr_keys/key2.gpg - публичный ключ 2
/etc/digsig/xattr_keys/key1/key1-1.gpg - публичный ключ, подписанный на ключе 1
/etc/digsig/xattr_keys/key1/key1-2.gpg - публичный ключ, подписанный на ключе 1
/etc/digsig/xattr_keys/key2/key2-1.gpg - публичный ключ, подписанный на ключе 2
/etc/digsig/xattr_keys/key2/key2-2.gpg - публичный ключ, подписанный на ключе 2
```

Для проверки использования дополнительных ключей для контроля целостности файлов (до перезагрузки ОС) можно от имени учетной записи администратора через механизм sudo выполнить команды:

```
cat /etc/digsig/xattr_keys/key1.gpg >> /sys/digsig/xattr_keys
cat /etc/digsig/xattr_keys/key2.gpg >> /sys/digsig/xattr_keys
cat /etc/digsig/xattr_keys/key1/key1-1.gpg >> /sys/digsig/xattr_keys
cat /etc/digsig/xattr_keys/key1/key1-2.gpg >> /sys/digsig/xattr_keys
```

```
cat /etc/digsig/xattr_keys/key2/key2-1.gpg >> /sys/digsig/xattr_keys
cat /etc/digsig/xattr_keys/key2/key2-2.gpg >> /sys/digsig/xattr_keys
```

Управление модулем digsig_verif осуществляется через интерфейс sysfs с использованием файлов:

- /sys/digsig/elf_mode — просмотр и переключение режима работы при проверке ЭЦП исполняемых файлов и разделяемых библиотек формата ELF;
- /sys/digsig/xattr_mode — просмотр и переключение режима работы при проверке ЭЦП в расширенных атрибутах файловой системы;
- /sys/digsig/keys — файл загрузки дополнительных ключей для проверки ЭЦП исполняемых файлов формата ELF и ЭЦП в расширенных атрибутах ФС;
- /sys/digsig/ignore_gost2001 — отключение проверки ЭЦП по ГОСТ Р 34.10-2001;
- /sys/digsig/ignore_xattr_keys — 1;
- /sys/digsig/xattr_control — список шаблонов имен, используемых при проверке ЭЦП в расширенных атрибутах ФС;
- /sys/digsig/xattr_keys — файл загрузки дополнительных ключей, используемых только при проверке ЭЦП в расширенных атрибутах ФС.

Проверка режимов работы выполняется командами:

```
cat /sys/digsig/elf_mode
cat /sys/digsig/xattr_mode
```

ВНИМАНИЕ! Для отключения проверки ЭЦП по ГОСТ Р 34.10-2001 необходимо в конфигурационном файле /etc/digsig/digsig_initramfs.conf установить следующее значение параметра:

```
DIGSIG_IGNORE_GOST2001=1
```

ВНИМАНИЕ! После внесения изменений в конфигурационный файл /etc/digsig/digsig_initramfs.conf и для загрузки модулем digsig_verif ключей после их размещения его в каталогах /etc/digsig/keys/ и /etc/digsig/xattr_keys/ необходимо от имени учетной записи администратора через механизм sudo выполнить команду:

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

16.1.2. Подписывание

В модуле ядра digsig_verif реализован механизм, позволяющий использовать несколько ключей при подписывании файлов формата ELF.

Порядок использования ключей для digsig_verif: дополнительные ключи записываются в /sys/digsig/keys или /sys/digsig/xattr_keys в иерархической последовательности цепочек подписей.

Все дополнительные ключи должны быть подписаны главным ключом или другим дополнительным ключом, подпись которого может быть проверена (за исключением первого ключа в каталоге `/sys/digsig/xattr_keys`).

Для создания дополнительных ключей используется GNU Privacy Guard. Модифицированный GnuPG выводит ГОСТ Р 34.11-94 в списке доступных алгоритмов. Для получения списка доступных алгоритмов необходимо выполнить команду:

```
# gpg --version
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
Хэш-функции: MD5, SHA1, RIPEMD160, SHA256, SHA384, SHA512,
          SHA224, GOST_R34.11-2012, GOST_R34.11-94
Алгоритмы сжатия: Без сжатия, ZIP, ZLIB,
          BZIP2
```

Далее приведен пример создания дополнительного ключа и его использования для подписывания СПО.

Пример

1) создается ключевая пара `GOST R 34.10-2001` и сохраняется в каталоге `~/.gnupg`. Для создания ключевой пары необходимо выполнить приведенную далее команду с последующим выбором в меню `gpg` алгоритма ГОСТ Р 34.10-2001:

```
keys@debian:~$ gpg --gen-key
gpg (GnuPG) 1.4.18; Copyright (C) 2014 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: создан каталог '/home/keys/.gnupg'
gpg: создан новый файл настоек '/home/keys/.gnupg/gpg.conf'
gpg: ВНИМАНИЕ: параметры в '/home/keys/.gnupg/gpg.conf' еще не активны при /
   этом запуске
gpg: создана таблица ключей '/home/keys/.gnupg/secring.gpg'
gpg: создана таблица ключей '/home/keys/.gnupg/pubring.gpg'
```

```
Выберите тип ключа:
(1) RSA и RSA (по умолчанию)
(2) DSA и Elgamal
(3) DSA (только для подписи)
(4) RSA (только для подписи)
(10) GOST R 34.10-2001 (только для подписи) устаревший
(12) GOST R 34.10-2012 (только для подписи)
Ваш выбор? 12
GOST keypair will have 256 bits.
Выборите срок действия ключа.
0 = без ограничения срока действия
<n> = срок действия - n дней
<n>w = срок действия - n недель
<n>m = срок действия - n месяцев
<n>y = срок действия - n лет
Срок действия ключа? (0) 0
Срок действия ключа не ограничен
Все верно? (y/N) y

Для идентификации Вашего ключа необходим ID пользователя. Программа создаст /
    его
из Вашего имени, комментария и адреса электронной почты в виде:
"Baba Yaga (pensioner) <yaga@deepforest.ru>"

Ваше настоящее имя: Test GOST R 34.10-2012 Secondary Key
Адрес электронной почты: test@gost.secondary.key
Комментарий:
Вы выбрали следующий ID пользователя:
"Test GOST R 34.10-2001 Secondary Key <test@gost.secondary.key>"

Сменить (N)Имя, (C)Комментарий, (E)адрес или (O)Принять/(Q)Выход? O
Для защиты закрытого ключа необходим пароль.
Введите пароль: ПАРОЛЬ

Повторите пароль: ПАРОЛЬ
```

2) экспорт ключа в файл осуществляется командой:

```
keys@debian:$ gpg --export "Test GOST R 34.10-2012 Secondary Key
<test@gost.secondary.key>" > /tmp/secondary_gost_key.gpg
```

3) ключ пользователя может быть заверен с использованием команд:

```
gpg --import /tmp/secondary_gost_key.gpg

gpg --sign-key "Test GOST R 34.10-2012 Secondary Key
<test@gost.secondary.key>"
```

```
gpg --export "Test GOST R 34.10-2001 Secondary Key
<test@gost.secondary.key>" > /tmp/secondary_gost_key_signed.gpg
```

4) пользователь подписывает на данном ключе некоторый файл формата ELF с использованием утилиты `bsign` (по умолчанию подпись внедряется в том числе и в расширенные атрибуты файла):

```
keys@debian:~$ bsign --sign test_elf
```

5) пользователь подписывает утилитой `bsign` на данном ключе произвольный файл с внедрением подписи только в расширенные атрибуты:

```
keys@debian:~$ bsign --sign --xattr test_elf
```

Дополнительная информация приведена в справке по утилите `bsign`;

6) для проверки правильности ЭЦП файла формата ELF используется утилита `bsign`:

```
keys@debian:~$ bsign -w test_elf
```

7) дополнительный ключ пользователя, подписанный на главном ключе, копируется в каталог `/etc/digsig/` (см. 16.1.1);

8) подписанный файл формата ELF может выполняться:

```
keys@debian:~$ ./test_elf
hello world!
keys@debian:~$
```

16.2. Режим киоска

16.2.1. Общие сведения

Режим киоска служит для ограничения прав пользователей в системе. Для использования функциональных возможностей данного режима необходимо установить пакет `parsec-cups`.

Степень этих ограничений прав пользователей задается маской киоска. Ее действие аналогично действию маски `umask` с тем отличием, что если `umask` накладывается при создании новых объектов ФС, то маска киоска накладывается на права доступа к файлу при любой попытке пользователя получить доступ. Маска киоска задается в конфигурационном файле `/etc/parsec/kiosk_mask` и по умолчанию равна `0000` (режим киоска выключен). Если маска равна `0003` (типичное значение при включенном режиме киоска), для пользователя блокируется доступ по записи и исполнению ко всем файлам, не принадлежащим ему, либо группе, в которую он входит. При маске, равной `0000`, поведение системы остается стандартным и на права доступа пользователя не накладывается никаких ограничений. Получить текущую маску киоска можно, прочитав содержимое файла `/parsecfs/mode_mask`. Маска киоска применяется только к обычным файлам. К каталогам, сокетам и т. д. маска не

применяется.

В режиме киоска (маска по умолчанию) пользователь не имеет возможности запустить ни одну системную программу, т. к. эти действия замаскированы.

16.2.2. mkiosk

Команда `mkiosk` позволяет задавать права доступа пользователя к конкретным файлам. Эти права доступа не подвержены действию маски и реализованы в виде ACL на специальные виртуальные файлы ФС `parsec`, являющиеся ссылками на реальные файлы. После перезагрузки все установленные ACL будут утеряны.

Права доступа к файлу задаются в виде:

```
<абсолютный_путь_к_файлу>
```

Так, например, права только на чтение для файла `/etc/hosts` можно задать в виде:

```
/etc/hosts r--
```

Права на чтение, запись и исполнение для файла `/usr/bin/example.sh` задаются в виде:

```
/usr/bin/example.sh rwx
```

Так как задавать все права доступа в командной строке было бы крайне неудобно, существует система профилей. Это файлы с готовыми наборами прав доступа для запуска каких-либо программ. Например, есть профиль для запуска `bash`, профиль для запуска `ls` и т. д. Эти профили хранятся в каталоге `/etc/parsec/kiosk-profiles`. Вместо прав доступа к конкретным файлам, в командной строке команды `mkiosk` можно указать готовый профиль. Она отличает задание файлов от задания профиля по наличию первого символа `/` в имени файла. Имена профилей задаются без указания полного пути к ним. В общем случае профиль может содержать в себе права доступа на более сложные действия, чем запуск одной программы. Существует также профиль с именем `default`, который используется автоматически при каждом запуске `mkiosk`. В нем содержатся права доступа, которые необходимы всегда. Это, например, право на использование динамического линковщика. Для создания профилей можно использовать команду `otrace` (16.3).

Для автоматизации процесса установки прав доступа для каждого пользователя существует конфигурационный файл, хранящийся в каталоге `/etc/parsec/kiosk` и содержащий все необходимые права доступа. По сути это такой же профиль, как и в случае профилей программ, но относящийся к конкретному пользователю. Этот файл может содержать как явное задание прав доступа к конкретным файлам, так и ссылки на профили программ. При входе пользователя в систему права доступа из конфигурационного файла будут установлены автоматически при помощи специального PAM-модуля.

Параметры команды приведены в таблице 54.

Таблица 54

Параметр	Описание
-h, --help	Вывести справку и выйти
-u , --user=	Установить права доступа для пользователя
-w, --without-profile	Не использовать профиль указанного пользователя для установки прав доступа. Будут использованы только права доступа из командной строки
-e, --mask	Указать маску киоска. Права доступа на файлы для пользователя будут устанавливаться только в том случае, когда необходимые биты маскируются указанной маской. По умолчанию используется текущая маска киоска из файла /parsecfs/mode_mask

Примеры:

1. Установить права доступа для пользователя ttt, взятые из его профиля /etc/parsec/kiosk/ttt

```
mkiosk -u ttt
```

2. Установить права на чтение файла /etc/passwd для пользователя ttt. При этом не учитывать профиль пользователя. Предполагается, что системная маска киоска равна 0003 и, соответственно, замаскированы права на запись и выполнение файлов

```
mkiosk -u ttt --mask=3 --without-profile "/etc/passwd r--"
```

16.3. otrace

Команда otrace предназначена для трассировки процессов относительно системных вызовов open() и execve().

Синтаксис:

```
otrace [-h, --help] [-s, --silent] [-o, --output=] [-k, --kiosk-dir=]
   [-p, --pid=] [-u, --user=] [-t, --trace] [-a, --audit-trace]
   [-m, --merge] [-f, --trace-failed] [-e, --mask=] [command]
```

Команда otrace используется в процессе конфигурирования режима киоска и служит для автоматизации создания профилей прав доступа.

В режиме киоска (стандартные настройки) пользователю запрещены запись и выполнение файлов, не принадлежащих ему, либо группе, в которую он входит. Чтобы пользователь имел возможность хотя бы войти в систему, необходимо явно указать права доступа ко всем файлам, прямо или косвенно участвующим при этой операции. Права доступа к файлам задаются с помощью установки ACL на специальные файлы-ссылки в ФС parsec. При этом права доступа на реальные файлы не изменяются. При перезагрузке системы все ACL на специальные файлы-ссылки будут утеряны.

Чтобы облегчить задачу установки пользователям прав доступа, используются профили прав доступа. Системные профили хранятся в каталоге `/etc/parsec/kiosk-profiles`. Далее приведен пример профиля, позволяющий запустить команду `ls`:

```
/bin/ls r-x
/etc/group r--
/etc/ld.so.cache r--
/etc/ld.so.preload r--
/etc/localtime r--
/etc/nsswitch.conf r--
/etc/passwd r--
/etc/selinux/config r--
/lib/libacl.so.1 r--
/lib/libattr.so.1 r--
/lib/libc.so.6 r--
/lib/libdl.so.2 r--
/lib/libnsl.so.1 r--
/lib/libnss_compat.so.2 r--
/lib/libnss_files.so.2 r--
/lib/libnss_nis.so.2 r--
/lib/libpthread.so.0 r--
/lib/librt.so.1 r--
/lib/libselinux.so.1 r--
/proc/mounts r--
/usr/lib/gconv/gconv-modules.cache r--
/usr/lib/gconv/KOI8-R.so r--
/usr/lib/locale/locale-archive r--
/usr/share/locale/locale.alias r--
/usr/share/locale/ru/LC_MESSAGES/coreutils.mo r--
/usr/share/locale/ru/LC_TIME/coreutils.mo r--
/usr/share/locale/ru_RU/LC_MESSAGES/coreutils.mo r--
/usr/share/locale/ru_RU/LC_TIME/coreutils.mo r--
/usr/share/locale/ru_RU.utf8/LC_MESSAGES/coreutils.mo r--
/usr/share/locale/ru_RU.UTF-8/LC_MESSAGES/coreutils.mo r--
/usr/share/locale/ru_RU.utf8/LC_TIME/coreutils.mo r--
/usr/share/locale/ru_RU.UTF-8/LC_TIME/coreutils.mo r--
/usr/share/locale/ru.utf8/LC_MESSAGES/coreutils.mo r--
/usr/share/locale/ru.UTF-8/LC_MESSAGES/coreutils.mo r--
```

```
/usr/share/locale/ru.utf8/LC_TIME/coreutils.mo r--
/usr/share/locale/ru.UTF-8/LC_TIME/coreutils.mo r--
```

Для применения профиля к конкретному пользователю используется команда `mkiosk` (см. 16.2.2).

Если профиль служит для запуска программы, как это было в примере, он должен содержать права доступа не только к исполняемому файлу, но и ко всем используемым библиотекам и всем файлам, которые открывает программа в процессе исполнения. Также необходимы права доступа на динамический линковщик, которые не попадут автоматически в профиль, созданный командой `otrace`. Минимальные права доступа, которые требуются всегда, содержатся в специальном профиле `/etc/parsec/kiosk-profile/default` и добавляются туда вручную.

Профили могут описывать права доступа для более сложных задач, чем запуск одной программы. Чтобы облегчить администрирование, можно использовать профили внутри других профилей. Если внутри профиля встречается строка с именем другого профиля, то содержимое указанного профиля полностью объединяется с содержимым текущего профиля. Объединение профилей осуществляется рекурсивно. Если строка в файле профиля начинается не с символа `/`, то она рассматривается как имя профиля. Команда `otrace` также занимается объединением профилей (см. опцию `--merge`).

Наконец, существуют профили, относящиеся к отдельным пользователям. Они хранятся в каталоге `/etc/parsec/kiosk` и заполняются вручную на основе готовых профилей либо стандартных строк, описывающих права доступа к конкретному файлу. При включенном режиме киоска профили пользователей автоматически применяются при входе пользователя в систему. Это реализовано при помощи специального PAM-модуля и команды `mkiosk` (см. 16.2.2).

Команда `otrace` может использовать два различных механизма для трассировки процессов. Первый — это программа `strace` (опция `--trace`). Второй — подсистема аудита PARSEC (опция `--audit-trace`). Программа `strace` имеет некоторые ограничения по использованию, поэтому применять механизм `--` следует только для трассировки довольно простых, не SUID-программ.

В режиме `--trace` цель трассировки может быть задана либо в командной строке команды `otrace` в качестве аргумента (тогда указанная программа будет запущена), либо может быть задан PID уже существующего процесса (опция `--pid`).

В режиме `--audit-trace` цель трассировки задается так же, как и в режиме `--trace`. Но кроме описанных, есть еще дополнительный способ задания цели трассировки — опция `--user`. При этом всем существующим процессам, принадлежащим указанному пользователю, будут выставлены соответствующие флаги аудита (см. 6.1.6). Таким обра-

зом, будут трассироваться все действия пользователя. Режим полезен, когда необходимо создать профиль, разрешающий пользователю выполнять целый набор сложных действий и трассировать каждую программу в отдельности затруднительно.

Если используется режим `--audit-trace`, то в системе не должно быть сторонних процессов, на которых установлены флаги аудита. Иначе в профиль может попасть информация, порожденная сторонними процессами. В режиме трассировки всех процессов указанного пользователя достаточно, чтобы в системе не было других процессов с установленными флагами аудита и принадлежащих этому пользователю.

Команда `otrace` по умолчанию записывает в профиль информацию только о тех действиях процесса (`open()`, `execve()`), которые прошли успешно. Однако для большей универсальности можно использовать опцию `--trace-failed`, которая позволит записать в профиль также информацию и о неудачных попытках. Это полезно, например, если процесс пытается открывать конфигурационные файлы. В момент трассировки файл может не существовать, но впоследствии он может появиться.

Параметры команды приведены в таблице 55.

Таблица 55

Параметр	Описание
`-h, --help`	Вывести справку и выйти
`-s, --silent`	Не выводить информационные сообщения
`-o , --output=`	Записать результаты трассировки в указанный файл. По умолчанию — `stdout`
`-k , --kiosk-dir=`	Указать путь к каталогу с профилями киоска. Используется в операции `--`. По умолчанию — `/etc/parsec/kiosk-profiles`
`-p , --pid=`	Трассировать процесс с указанным идентификатором, а также все порожденные им процессы
`-u , --user=`	Указать имя пользователя. Используется совместно с `--audit-trace` или `--merge`
`-t, --trace`	Использовать для трассировки процессов команду `strace`. Не может быть использована совместно с `--audit-trace`
`-a, --audit-trace`	Использовать для трассировки процессов подсистему аудита PARSEC. Не может быть использована совместно с `--trace`
`-m, --merge`	Объединять все права доступа, указанные каким-либо способом в единый поток с уникальными записями. Права доступа могут быть указаны в явном виде, в виде профилей, в виде профиля пользователя и профиля, используемого по умолчанию (`/etc/parsec/kiosk-profiles/default`)
`-f, --trace-failed`	Учитывать неудачные попытки вызова `open()` и `execve()`. Права доступа к этим файлам будут заданы согласно параметрам, с которыми процесс пытается получить доступ к ним

Окончание таблицы 55

Параметр	Описание
`-e , --mask=`	Указать маску киоска. Это позволяет обрабатывать данные согласно этой маске и учитывать только те файлы, права доступа к которым будут действительно замаскированы. По умолчанию маска равна 7

Примеры:

1. Запустить и трассировать процесс `ls /` с помощью команды `strace`. Записать результат в файл `/tmp/ls_trace`

```
otrace --trace -o /tmp/ls_trace ls /
```

2. Трассировать запущенные процессы пользователя `ttt` и все вновь порожденные ими процессы с помощью подсистемы аудита PARSEC. Отслеживать также информацию о неудачных попытках открытия файлов и запуска процессов. Результаты трассировки вывести в `stdout`

```
otrace --audit-trace -u ttt -f
```

3. Объединить содержимое профиля пользователя `ttt`, профиля с именем `ls` из каталога `/etc/parsec/kiosk-profiles` и добавить права на чтение и выполнение файла `/usr/bin/example`. Предполагать, что системная маска киоска равна 3 и, соответственно, учитывать только те файлы, для которых права доступа должны включать права на запись или выполнение

```
otrace --merge --mask=3 -u ttt ls "/usr/bin/example r-x"
```

16.3.1. fly-admin-kiosk

Кроме средств для работы в режиме командной строки, в распоряжении администратора имеется графическая утилита `fly-admin-kiosk`, которая может быть использована для настройки и управления режимом киоска.

Описание утилиты см. в электронной справке.

ВНИМАНИЕ! Перед использованием утилиты необходимо сделать резервную копию всех системных профилей, находящихся в каталоге `/etc/parsec/kiosk-profiles`, и системного профиля `fly-dm` в каталоге `/etc/parsec/kiosk`. Системные профили, устанавливаемые по умолчанию, находятся в пакете `parsec-kiosk`.

Переместить системный профиль `fly-dm` из каталога `/etc/parsec/kiosk` в каталог `/etc/parsec/kiosk-profiles`, выполнив команду:

```
mv /etc/parsec/kiosk/fly-dm /etc/parsec/kiosk-profiles/fly-dm
```

Создать пользовательский профиль `fly-dm` в каталоге `/etc/parsec/kiosk`, выполнив команду:

```
echo fly-dm > /etc/parsec/kiosk/fly-dm
```

Отредактировать содержимое файла `/etc/parsec/kiosk-profiles`, заключив в кавычки имена файлов:

```
"/lib64/ld-linux-x86-64.so.2" r-x
"/lib/x86_64-linux-gnu/ld-linux-x86-64.so.2" r-x
```

Далее можно использовать утилиту `fly-admin-kiosk`.

16.3.2. Настройка режима киоска для пользователя

Для разрешения входа пользователя в режиме текстовый консоли необходимо, используя механизм `sudo`, от имени пользователя `root` создать файл профиля для пользователя `имя_пользователя` в каталоге `/etc/parsec/kiosk` и добавить в файл `имя_пользователя` перечень профилей, необходимых для разрешения входа пользователя, выполнив команды:

```
echo default > /etc/parsec/kiosk/имя_пользователя
echo bash >> /etc/parsec/kiosk/имя_пользователя
```

Для добавления разрешенных для пользователя команд необходимо:

1) войти в систему от имени учетной записи пользователя на одной консоли (например, `tty1`);

2) войти в систему от имени учетной записи администратора на другой консоли (например, `tty2`), используя команду `sudo -s`, переключиться в сессию суперпользователя `root` и запустить протоколирование действий пользователя, выполнив команду:

```
otrace -a -o /etc/parsec/kiosk-profile/новый_файл_профиля -f --mask=3
      -u имя_пользователя
```

3) дождаться перезапуска сервиса `parlogd`;

4) перейти на консоль пользователя и выполнить все команды, которые должны быть разрешены в дальнейшем;

5) завершить сеанс пользователя;

6) в консоли пользователя `root` остановить трассировку, нажав клавишу **<Enter>**;

7) подключить полученный новый профиль к пользователю, выполнив команду:

```
echo новый_файл_профиля >> /etc/parsec/kiosk/имя_пользователя
```

8) в консоли пользователя `root` включить режим киоска и перезагрузить ОС, выполнив команды:

```
echo 0003 > /etc/parsec/kiosk_mask
reboot
```

Для разрешения входа пользователя в графическом режиме необходимо:

1) проверить наличие профиля `fly-dm` в файле `/etc/parsec/kiosk/fly-dm`;

2) подключить профиль `fly` к профилю пользователя `имя_пользователя`, выполнив от имени пользователя `root` команду:

```
echo fly >> /etc/parsec/kiosk/имя_пользователя
```

Для создания профиля `fly` необходимо:

1) войти в систему в режиме текстовой консоли от имени администратора, используя команду `sudo -s`, переключиться в сессию суперпользователя `root` и остановить работу `fly-dm`, выполнив команду:

```
service fly-dm stop
```

2) проверить наличие логической ссылки `/usr/bin/x-session-manager`, выполнив команду:

```
ls -l /usr/bin/x-session-manager
```

3) удалить указанную логическую ссылку и создать другую с суффиксом `.orig`, выполнив команды:

```
rm /usr/bin/x-session-manager
ln -s /etc/alternatives/x-session-manager
    /usr/bin/x-session-manager.orig
```

4) создать текстовый файл `/usr/bin/x-session-manager` со следующим содержимым:

```
#!/bin/bash
/usr/sbin/otrace -t -f -o /test/fly --mask=3
    /usr/bin/x-session-manager.orig $@ &
sleep 20
/usr/bin/fly-wmfunc FLYWM_EXIT
```

5) изменить права на созданный файл, выполнив команду:

```
chmod 777 /usr/bin/x-session-manager
```

6) установить пакет `strace`, выполнив команду:

```
apt-get install starce
```

7) создать каталог `/test` с правами `777`, выполнив команду:

```
mkdir -m 777 /test
```

8) запустить `fly-dm`, выполнив команду:

```
service fly-dm start
```

9) выполнить вход пользователя в графическом режиме и подождать 20 с до автоматического завершения сессии пользователя (можно открыть стартовую меню-панель Fly, терминал `fly-term`);

10) в сессии суперпользователя `root` проверить наличие профиля `fly` в каталоге `/test`, выполнив команду:

```
ls -l /test/fly
```

11) удалить файл `/usr/bin/x-session-manager`, выполнив команду:

```
rm /usr/bin/x-session-manager
```

12) создать логическую ссылку `/usr/bin/x-session-manager`, выполнив команду:

```
ln -s /etc/alternatives/x-session-manager /usr/bin/x-session-manager
```

13) открыть полученный профиль /test/fly в текстовом редакторе, удалить строки вида ... resumed ..., строки для /proc и строки, содержащие пути, начинающиеся с ./;

14) перенести профиль в каталог профилей, выполнив команду:

```
cp /test/fly /etc/parsec/kiosk-profiles/fly
```

15) подключить профиль fly к профилю пользователя, выполнив команду:

```
echo fly >> /etc/parsec/kiosk/имя_пользователя
```

16) включить режим киоска и перезагрузить ОС, выполнив команды:

```
echo 0003 > /etc/parsec/kiosk_mask
reboot
```

17) выполнить в графическом режиме вход в систему от имени пользователя;

18) если сессия не открывается, то проверить в консоли пользователя root содержимое файла /home/имя_пользователя/.xsession-errors, содержащего файлы, права на которые не выставлены в профиле fly, и добавить разрешение вручную.

Пример строки из файла /home/имя_пользователя/.xsession-errors:

```
/etc/X11/fly-dm/Xsession : line 55 /bin/df: отказано в доступе
```

Пример добавления строки в профиль fly:

```
echo '"/bin/df" r-x' >> /etc/parsec/kiosk-profiles/fly
```

19) после изменений применить профиль к пользователю, выполнив команду:

```
mkiosk -u user
```

20) при наличии в файле /home/имя_пользователя/.xsession-errors строки, содержащей:

```
unable to create error file
```

проверить наличие в профиле fly команд chmod, rm, cp и файлов XErrorDB, /usr/lib/parsec/bin/x-session-manager и, при необходимости, добавить в профиль полные пути к ним.

Для добавления пользователю разрешения на выполнение конкретных программ (например, firefox) необходимо:

1) выключить режим киоска, выполнив команды:

```
echo 0000 > /etc/parsec/kiosk_mask
reboot
```

2) выполнить в графическом режиме вход в систему от имени пользователя;

3) войти в систему в режиме текстовой консоли от имени администратора, используя команду sudo -s, переключиться в сессию root и запустить протоколирование действий пользователя, выполнив команду:

```
otrace -a -o /etc/parsec/kiosk-profile/firefox -f --mask=3
    -u имя_пользователя
```

4) дождаться перезапуска сервиса parlogd;

5) перейти в графический интерфейс пользователя и запустить/завершить `firefox`;

6) завершить сеанс пользователя;

7) в консоли пользователя `root` остановить трассировку, нажав клавишу **<Enter>**;

8) подключить полученный профиль `firefox` для пользователя, выполнив команду:

```
echo firefox >> /etc/parsec/kiosk/имя_пользователя
```

9) в консоли пользователя `root` включить режим киоска и перезагрузить ОС, выполнив команды:

```
echo 0003 > /etc/parsec/kiosk_mask
reboot
```

16.3.3. Киоск Fly

Для ограничения возможности запуска программ локальным пользователям администратор может использовать графическую утилиту `fly-admin-smc`. Для этого необходимо в настройках политики безопасности пользователя графической утилиты `fly-admin-smc` во вкладке «Графический киоск Fly» установить флаг «Режим графического киоска Fly». Флаг включает режим киоска при работе с приложениями из списка. Если в списке одно приложение, то режим киоска включается при работе с этим приложением. Если в списке несколько приложений, то запускается Рабочий стол с этими приложениями. Все доступные каталоги, ярлыки и т. д. устанавливаются в соответствии с предоставленным доступом.

Настройка режима киоска с помощью графической утилиты `fly-admin-smc` осуществляется администратором на максимальном уровне мандатного контроля целостности, установленного в ОС.

16.4. Установка системных ограничений

Параметры системных ограничений заданы в файле `/etc/security/limits.conf`. Управление системными ограничениями осуществляется с помощью графической утилиты `fly-admin-smc` или инструмента командной строки `astra-ulimits-control`.

Для включения системных ограничений используется команда:

```
astra-ulimits-control enable
```

Для выключения системных ограничений используется команда:

```
astra-ulimits-control disable
```

Для проверки состояния используется команда:

```
systemctl is-enabled astra-ulimits-control
```

Результат выполнения команды:

- `enabled` — системные ограничения включены;
- `disabled` — системные ограничения выключены;
- `Failed to get unit file state` — сервис не активирован.

16.5. Ограничение действий пользователя

16.5.1. Режим запрета установки бита исполнения

В ОС реализован режим запрета установки бита исполнения, обеспечивающий предотвращение несанкционированного создания пользователями или непреднамеренного создания администратором исполняемых сценариев для командной оболочки.

Включение режима запрета установки бита исполнения осуществляется от имени учетной записи администратора через механизм `sudo` с использованием следующих команд:

```
sudo echo 1 > /parsecfs/nochmodx
sudo echo 1 > /etc/parsec/nochmodx
```

или команды:

```
astra-nochmodx-lock enable
```

Отключение режима запрета установки бита исполнения осуществляется от имени учетной записи администратора через механизм `sudo` с использованием следующих команд:

```
sudo echo 0 > /parsecfs/nochmodx
sudo echo 0 > /etc/parsec/nochmodx
```

или команды:

```
astra-nochmodx-lock disable
```

ВНИМАНИЕ! В режиме запрета установки бита исполнения данная операция запрещена для всех пользователей ОС, включая администратора и суперпользователя `root`. Установка пакетов программ, создающих в ФС файлы с битом исполнения, будет завершаться с ошибкой.

Проверка состояния режима запрета установки бита исполнения выполняется командой:

```
cat /parsecfs/nochmodx
```

Результат выполнения команды:

- `1` — режим включен;
- `0` — режим выключен.

Для управления режимом запрета установки бита исполнения в графическом режиме используется утилита `fly-admin-smc` (см. электронную справку).

16.5.2. Блокировка консоли для пользователей

Блокировка консоли для пользователей осуществляется с помощью графической утилиты `fly-admin-smc` или инструмента командной строки `astra-console-lock`.

Для включения блокировки консоли для пользователей используется команда:

```
astra-console-lock enable
```

Для выключения блокировки консоли для пользователей используется команда:

```
astra-console-lock disable
```

Для проверки состояния используется команда:

```
systemctl is-enabled astra-console-lock
```

Результат выполнения команды:

- `enabled` — блокировка консоли для пользователей включена;
- `disabled` — блокировка консоли для пользователей выключена;
- `Failed to get unit file state` — сервис не активирован.

16.5.3. Блокировка интерпретаторов

В ОС реализована функция блокировки следующих интерпретаторов:

- Python;
- Perl;
- Expect;
- Ruby;
- Dash.

При включении блокировки интерпретаторов блокируется несанкционированное использование интерпретатора для выполнения кода напрямую из командной строки или из неименованного канала (pipe). При этом сценарии из каталога `/usr/bin/`, написанные для этих интерпретаторов, выполняются в штатном режиме.

Блокировка интерпретаторов осуществляется с помощью графической утилиты `fly-admin-smc` или инструмента командной строки `astra-interpreters-lock`.

Для включения блокировки интерпретаторов используется команда:

```
astra-interpreters-lock enable
```

Для выключения блокировки интерпретаторов используется команда:

```
astra-interpreters-lock disable
```

Для проверки состояния используется команда:

```
systemctl is-enabled astra-interpreters-lock
```

Результат выполнения команды:

- `enabled` — блокировка интерпретаторов включена;
- `disabled` — блокировка интерпретаторов выключена;
- `Failed to get unit file state` — сервис не активирован.

16.5.4. Блокировка макросов

Блокировка макросов осуществляется с помощью графической утилиты `fly-admin-smc` или инструмента командной строки `astra-macros-lock`.

Для включения блокировки макросов используется команда:

```
astra-macros-lock enable
```

Для выключения блокировки макросов используется команда:

```
astra-macros-lock disable
```

Для проверки состояния используется команда:

```
systemctl is-enabled astra-macros-lock
```

Результат выполнения команды:

- `enabled` — блокировка макросов включена;
- `disabled` — блокировка макросов выключена;
- `Failed to get unit file state` — сервис не активирован.

16.5.5. Блокировка трассировки ptrace

Блокировка трассировки ptrace осуществляется с помощью графической утилиты `fly-admin-smc` или инструмента командной строки `astra-ptrace-lock`.

Для включения блокировки трассировки ptrace используется команда:

```
astra-ptrace-lock enable
```

Для выключения блокировки трассировки ptrace используется команда:

```
astra-ptrace-lock disable
```

Для проверки состояния используется команда:

```
systemctl is-enabled astra-ptrace-lock
```

Результат выполнения команды:

- `enabled` — блокировка трассировки ptrace включена;
- `disabled` — блокировка трассировки ptrace выключена;
- `Failed to get unit file state` — сервис не активирован.

16.5.6. Блокировка клавиш SysRq

Блокировка клавиш SysRq осуществляется с помощью графической утилиты `fly-admin-smc` или инструмента командной строки.

Для включения блокировки клавиш SysRq используется команда:

```
sysctl -w kernel.sysrq=0
```

Для выключения блокировки клавиш SysRq используется команда:

```
sysctl -w kernel.sysrq=1
```

Для проверки состояния используется команда:

```
cat /proc/sys/kernel/sysrq
```

Результат выполнения команды:

- `0` — блокировка клавиш SysRq включена;
- `1` — блокировка клавиш SysRq выключена.

17. ЗАПУСК ОС

17.1. Запуск

При запуске ОС появляется окно, вид которого представлен на рис. 1.

Рис. 1 – Окно запуска ОС

В окне приведен перечень режимов загрузки ОС. Для загрузки в штатном режиме необходимо в перечне выбрать режим `generic`. Для загрузки в целях восстановления работоспособности ОС может использоваться режим `generic (single-user mode)`.

В процессе загрузки ОС осуществляется инициализация модуля ядра и запуск сервисов системы безопасности информации PARSEC.

По завершении загрузки ОС появляется окно, содержащее приглашение для ввода имени и пароля пользователя. В случае успешного прохождения идентификации и аутентификации пользователю будет предложено выбрать мандатный уровень и категории, которые будут использованы при создании сеанса пользователя в ОС. Выбор мандатной метки осуществляется в соответствии с текущими настройками максимального и минимального уровней и максимального и минимального наборов категорий, установленных для данного

пользователя.

Вид окна для выбора мандатной метки (уровня и категорий) представлен на рис. 2.

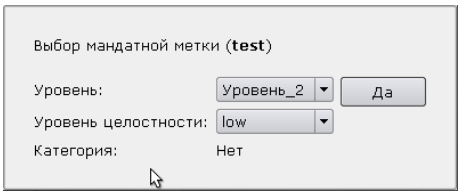

Рис. 2 – Окно выбора мандатной метки

17.2. Настройка параметров, необходимых для эксплуатации ОС

Перед началом эксплуатации ОС администратор безопасности должен обеспечить выполнение следующих условий:

1) механизм замкнутой программной среды должен быть настроен для работы в штатном режиме (см. 16.1);

2) с использованием средств управления дискреционными ПРД (см. 3.3) пользователям должен быть запрещен доступ к библиотеке `libpcprofile.so`;

3) с использованием средств управления мандатными ПРД (см. 4.8) всем отчуждаемым носителям, используемым на объекте эксплуатации, должны быть присвоены мандатные метки, соответствующие грифу обрабатываемой информации. Все отчуждаемые носители должны быть учтены режимно-секретным отделом организации, эксплуатирующей автоматизированную систему. Использование неучтенных отчуждаемых носителей должно быть запрещено;

4) с использованием средств управления дискреционными ПРД (см. 3.3) пользователям, не обладающим привилегиями администратора, должен быть запрещен запуск (использование) средств создания символических ссылок;

5) с использованием средств управления запуском сервисов должна быть отключена служба `gpm` для поддержки мыши в консольном режиме;

6) в случае разрешения интерактивного входа суперпользователя `root` для предотвращения подбора его пароля необходимо заблокировать возможность его удаленного входа в ОС посредством включения PAM-модуля `pam_securetty` в файл сценария `/etc/pam.d/common-auth`. Для этого необходимо в «Primary block» в указанном файле первой строкой добавить:

```
auth required pam_securetty.so
```

7) используя графическую утилиту `fly-admin-smc` («Управление политикой безопасности»), запущенную от имени администратора через механизм `sudo` (см. электронную справку), в категории «Политики учетной записи» задать следующие значения для настройки политики паролей:

а) «Сложность»:

- «Минимальная длина пароля» — 8;
- установить флаг «Минимальное количество строчных букв в новом пароле»;
- установить флаг «Минимальное количество заглавных букв в новом пароле»;
- установить флаг «Минимальное количество цифр в новом пароле»;

б) «Срок действия»:

- установить флаг «Минимальное количество дней между сменами пароля» и в поле задать значение «0 дней»;
- установить флаг «Максимальное количество дней между сменами пароля» и в поле задать значение «90 дней»;

либо задать настройки политики паролей путем корректировки файлов сценариев:

а) в файле `/etc/pam.d/common-password` в строке `password requisite pam cracklib.so` установить значение `minlen=8` и добавить параметры `dcredit=-1`, `ucredit=-1` и `lcredit=-1`;

б) в файле `/etc/login.defs` для переменной `PASS_MAX_DAYS` установить значение `90`.

Дополнительно необходимо запретить повторное использование последних четырех паролей, откорректировав файл `/etc/pam.d/common-password`. Для этого в указанном файле в строке «`...pam_unix.so`» добавить параметр `remember=4`;

8) используя графическую утилиту `fly-admin-smc` («Управление политикой безопасности»), запущенную от имени администратора через механизм `sudo` (см. электронную справку), в категории «Политики учетной записи» задать следующие значения для настройки блокировки:

- установить флаг «Неуспешных попыток» и в поле задать значение `6`;
- установить флаг «Период блокировки» и в поле задать значение «1800 секунд»;

либо настроить блокировку учетной записи, откорректировав файл `/etc/login.defs`. Для этого в указанном файле для переменной `LOGIN_RETRIES` установить значение `6` и для переменной `LOGIN_TIMEOUT` установить значение `1800`;

9) используя графическую утилиту `fly-admin-theme` («Программа «Темы рабочего стола»), запущенную от имени администратора через механизм `sudo` (см. электронную справку), в разделе «Блокировка» установить флаг «Блокировать экран через» и в поле задать значение «15 мин».

17.3. Условия применения ПО

ПО, образующее совместно с изделием доверенную программную среду, должно применяться при соблюдении следующих условий:

1) для ПО, функционирующего от имени суперпользователя `root`, должна быть обоснована необходимость и корректность использования привилегий суперпользователя;

2) ПО, использующее интерпретаторы PHP, Perl, Python, TCL из состава изделия, должно пройти сертификационные исследования совместно с ними. При проведении таких сертификационных исследований должен быть определен перечень сценариев ПО, исполняемых интерпретаторами, зафиксированы их контрольные суммы и определен регламент контроля их целостности;

3) 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;

4) ПО, содержащее сетевые сервисы (программы, обрабатывающие запросы пользователей с применением протоколов стека TCP/IP), которые используют привилегии (`PARSEC_CAP_SETMAC` и `PARSEC_CAP_PRIV_SOCK`) и прикладной программный интерфейс подсистемы безопасности PARSEC из состава изделия, должно пройти сертификационные исследования, подтверждающие корректность реализации указанных сервисов;

5) программный компонент RabbitMQ не применять при одновременной обработке данных с различными уровнями конфиденциальности либо для обеспечения взаимодействия процессов с различными уровнями доступа;

6) модуль `wsgi_mod` для Apache не применять при включенном режиме AstraMode (см. РУСБ.10015-01 95 01-1);

7) ПО, обрабатывающее одновременно данные с различными уровнями конфиденциальности, не должно использовать программные пакеты Erlang.

ПО, образующее совместно с изделием доверенную программную среду, не должно изменять алгоритм принятия решения о предоставлении доступа субъектов (пользователей) к объектам ОС и содержать скрытых или явных возможностей, позволяющих:

1) подменять образы ядра изделия `vmlinuz-*`, находящиеся в каталоге `/boot` корневой файловой системы;

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 конфигурацион-

ном файле загрузчика изделия путем установки параметра `init=<полный_путь_к_исполняемому_файлу>`;

4) изменять параметры аутентификации пользователей в конфигурационных файлах PAM-сценариев, находящихся в каталоге `/etc/pam.d`;

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 серверы печати, позволяющие осуществить вывод на печать документов в обход защищенного сервера печати из состава изделия;

8) устанавливать интерпретаторы команд, заменяющие интерпретаторы, входящие в состав изделия (bash, dash, PHP, Perl, Python, TCL);

9) получать доступ к памяти других процессов ПО с использованием привилегии `CAP_SYS_PTRACE` и системного вызова `ptrace`;

10) изменять системное время (если наличие возможностей по изменению времени не предусмотрено функциональным назначением ПО);

11) динамически изменять сегмент кода ядра ОС и использовать неэкспортируемые символы ядра ОС;

12) осуществлять доступ к файлу `ctl` файловой системы `parsecfs` посредством системного вызова `ioctl`, минуя системные функции API-библиотек `libparsec-*`.

Для управления авторизацией пользователей и обеспечения работы средств разграничения доступа в сервере гипертекстовой обработки данных Apache2 используется параметр `AstraMode` в конфигурационном файле `/etc/apache2/apache2.conf`. По умолчанию режим включен, а параметр `AstraMode` отсутствует, что соответствует значению `AstraMode on`. Для отключения режима авторизации в конфигурационном файле `/etc/apache2/apache2.conf` следует добавить/изменить параметр на `AstraMode off`.

ВНИМАНИЕ! При отключенной авторизации пользователей Apache2 осуществляет все запросы к своим ресурсам посредством только одной системной учетной записи (по умолчанию `www-data`), которая в случае настроенной политики мандатного контроля целостности имеет уровень целостности равный 1.

Для настройки безопасной конфигурации средств защиты ОС СН рекомендуется учитывать материал страницы Astra Linux Red-Book по адресу `https://wiki.astralinux.ru/display/doc/Astra+Linux+Red-Book`.

17.4. Проверка правильности запуска

В случае успешного входа пользователя в систему при наведении мыши на индикатор мандатных атрибутов в области уведомлений на панели задач (рис. 3) открывается всплывающее окно, которое отражает текущие мандатные атрибуты сеанса пользователя.

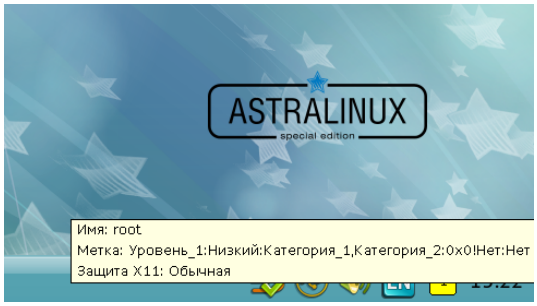

Рис. 3

При правильном запуске СЗИ данные атрибуты должны совпадать с введенными пользователем при входе в ОС.

Кроме того, для получения текущих мандатных атрибутов пользователя может быть использована консольная утилита `pdp-id`, описание которой приведено в `man pdp-id`.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

БД — база данных
ВФС — виртуальная файловая система
ЕПП — единое пространство пользователей
ЗПС — замкнутая программная среда
КСЗ — комплекс средств защиты
МКЦ — мандатный контроль целостности
НСД — несанкционированный доступ
ОС — операционная система
ПО — программное обеспечение
ПРД — правила разграничения доступа
СЗИ — средства защиты информации
СЗФС — сетевая защищенная файловая система
СПО — специальное программное обеспечение
СУБД — система управления базами данных
ФС — файловая система
ЭЦП — электронная цифровая подпись

ACL — Access Control List (список контроля доступа)
ALD — Astra Linux Directory (единое пространство пользователей)
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)
DHCP — Dynamic Host Configuration Protocol (протокол динамической конфигурации хоста)
DNS — Domain Name System (служба доменных имен)
FTP — File Transfer Protocol (протокол передачи файлов)
GID — Group Identifier (идентификатор группы)
IP — Internet Protocol (протокол Интернет)
IPC — InterProcess Communication (межпроцессное взаимодействие)
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 (подгружаемые аутентификационные модули)

PID — Process Identifier (идентификатор процесса)

RFC — Request for Comments (документ, содержащий технические спецификации и стандарты, применяемые в сети Интернет)

RSA — Rivest Shamir Adelman (алгоритм по схеме открытого ключа)

SQL — Structured Query Language (язык структурированных запросов)

TCP — Transmission Control Protocol (протокол передачи данных)

UDP — User Datagram Protocol (протокол пользовательских дейтаграмм)

UID — User Identifier (идентификатор пользователя)

Лист регистрации изменений

Изм.	Номера листов (страниц)				Всего листов (страниц) в документе	Номер документа	Входящий номер сопро-водительного документа и дата	Подпись	Дата
	измененных	замененных	новых	аннулированных					
1		Все			153	РУСБ.14.03-11			20.10.11
2		Все			153	РУСБ.14.04-13			21.03.13
3		Все			153	РУСБ.14.05-14			19.11.14
4		Все			153	РУСБ.14.02-16			17.02.16
5		Все			153	РУСБ.14.03-17			05.04.17
6		Все			153	РУСБ.14.02-18			11.04.18
7		Все			167	РУСБ.14.03-18			19.04.18