Утвержден

РУСБ.10015-37-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.

Бюллетень № 2023-0426SE17

РУСБ.10015-37 97 01-1

Листов 290

2023

АННОТАЦИЯ

Настоящий документ является первой частью руководства по комплексу средств защиты (КСЗ) операционной системы специального назначения «Astra Linux Special Edition» РУСБ.10015-37 (далее по тексту — ОС).

Документ предназначен для администраторов безопасности.

Руководство по КСЗ состоит из двух частей:

- РУСБ.10015-37 97 01-1 «Операционная система специального назначения «Astra Linux Special Edition». Руководство по КСЗ. Часть 1»;
- РУСБ.10015-37 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 ограничения действий пользователя Киоск-2, а также порядок запуска и применения ОС.

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

Во второй части руководства приведено описание тестов КСЗ.

СОДЕРЖАНИЕ

1. Общие сведения 12
1.1. Состав КСЗ 12
1.2. Контролируемые функции 12
1.3. Средства организации ЕПП 14
2. Идентификация и аутентификация 16
3. Дискреционное управление доступом 19
3.1. Общие сведения 19
3.2. Linux-привилегии 24
3.3. Средства управления дискреционными ПРД 24
3.3.1. chown 25
3.3.2. chgrp 26
3.3.3. chmod 26
3.3.4. umask 28
3.3.5. getfacl 29
3.3.6. setfacl 30
3.3.6.1. Элементы ACL 32
3.3.6.2. Автоматически созданные права доступа 32
3.4. Дискреционное управление доступом в СУБД PostgreSQL 33
3.5. Средства управления дискреционными ПРД к объектам БД СУБД PostgreSQL . . 36
4. Мандатное управление доступом и мандатный контроль целостности 38
4.1. Общие сведения 38
4.2. Мандатное управление доступом 38
4.2.1. Уровень конфиденциальности 38
4.2.2. Категория конфиденциальности 39
4.2.3. Дополнительные атрибуты сущностей для мандатного управления доступом . . 40
4.3. Мандатный контроль целостности 40
4.3.1. Метка целостности 41
4.3.2. Дополнительные атрибуты сущностей для мандатного контроля целостности . 42
4.4. Мандатный контекст безопасности 43
4.5. Расширенный режим мандатного контроля целостности 44

4.6. Применение правил мандатного управления доступом и мандатного контроля целостности . . . 45
4.7. PARSEC-привилегии . . . 47
4.8. Включение и выключение мандатного управления доступом . . . 49
4.8.1. Включение мандатного управления доступом . . . 49
4.8.2. Выключение мандатного управления доступом . . . 49
4.9. Включение и выключение мандатного контроля целостности . . . 50
4.10. Мандатный контроль целостности на файловой системе . . . 51
4.11. Администрирование ОС при включенном МКЦ . . . 52
4.12. Запуск служб systemd с уровнем целостности и конфиденциальности . . . 53
4.13. Сетевое взаимодействие. Механизм privsock . . . 54
4.14. Шина межпроцессного взаимодействия D-Bus . . . 55
4.14.1. Виды сообщений . . . 56
4.14.2. Процесс взаимодействия с шиной . . . 56
4.14.3. Процесс соединения с системной шиной . . . 57
4.14.4. Объекты и субъекты системы . . . 58
4.14.5. Алгоритм проверки меток для различных сообщений и режимов работы . . . 59
4.14.6. Привилегии процесса dbus-daemon . . . 61
4.14.7. Расширенное управление политиками . . . 61
4.14.7.1. Конфигурационный файл . . . 61
4.14.7.2. Формирование клиентских политик из политик шины . . . 66
4.15. Средства управления мандатными ПРД . . . 67
4.15.1. pdpl-file . . . 68
4.15.2. pdp-id . . . 70
4.15.3. pdp-init-fs . . . 70
4.15.4. pdp-ls . . . 71
4.15.5. pdpl-ps . . . 71
4.15.6. pdpl-user . . . 72
4.15.7. pdp-exec . . . 73
4.15.8. sumac . . . 74
4.15.9. sumic . . . 75
4.15.10. userlev . . . 75
4.15.11. usercat . . . 76

4.15.12. Устаревшие утилиты управления мандатными ПРД 76
4.15.12.1. chmac 77
4.15.12.2. macid 78
4.15.12.3. lsm 78
4.15.12.4. psmac 79
4.15.12.5. usermac 79
4.15.12.6. getfmac 80
4.15.12.7. setfmac 81
4.16. Средства управления привилегиями пользователей и процессов 82
4.16.1. usercaps 82
4.16.2. execaps 83
4.16.3. pscaps 84
4.17. Мандатное управление доступом в СУБД PostgreSQL 85
4.17.1. Порядок применения мандатных правил управления доступом 86
4.17.2. Средства управления мандатными ПРД к объектам БД 92
4.17.3. Целостность мандатных атрибутов кластера баз данных 95
4.17.4. Ссылочная целостность мандатных атрибутов 95
4.17.5. Особенности создания правил, системы фильтрации и триггеров 97
4.17.6. Особенности использования представлений и материализованных представлений 97
4.17.7. Функции сравнения для типа `maclabel` 98
4.17.8. Система привилегий СУБД 98
4.18. Мандатное управление доступом в комплексах программ гипертекстовой обработки данных и электронной почты 99
4.19. Настройка загрузчика GRUB 2 100
5. Защита среды виртуализации 102
5.1. Дискреционное управление доступом в среде виртуализации 103
5.2. Мандатное управление доступом к виртуальной машине 104
5.3. Ролевое управление доступом в среде виртуализации 105
5.3.1. Настройка ролей 105
5.3.2. Настройка ролевого управления доступом с использованием драйвера доступа polkit 108
5.3.3. Настройка ролевого управления доступом с использованием драйвера доступа parsec 109

5.4. Режим «только чтение»: запрет модификации образа виртуальной машины . . . 109
5.5. Идентификация и аутентификация пользователей в среде виртуализации 110
5.6. Доверенная загрузка виртуальных машин 111
5.6.1. Применение режима контроля целостности файлов при их открытии на основе ЭЦП 112
5.6.1.1. Контроль файлов конфигурации виртуального оборудования виртуальных машин113
5.6.1.2. Контроль файлов виртуальной базовой системы ввода-вывода (первичного загрузчика виртуальной машины) 114
5.6.2. Применение механизма контроля целостности с использованием алгоритма работы с контрольными суммами («отпечатка конфигурации») 116
5.6.3. Применение механизма контроля целостности файлов гостевой операционной системы 119
5.7. Контроль целостности в среде виртуализации 122
5.7.1. Применение динамического контроля целостности в режиме ЗПС 123
5.7.1.1. Режим контроля неизменности и подлинности загружаемых исполняемых файлов формата ELF 123
5.7.1.2. Режим контроля целостности файлов при их открытии на основе ЭЦП 124
5.7.2. Применение регламентного контроля целостности AFICK 124
5.8. Регистрация событий безопасности в среде виртуализации 124
5.8.1. Настройка регистрации событий безопасности, связанных с функционированием средства виртуализации 124
5.8.2. Механизм централизованного сбора журналов с удаленных хостов виртуализации125
5.9. Резервное копирование в среде виртуализации 126
5.9.1. Резервное копирование образов виртуальных машин 127
5.9.2. Резервное копирование конфигурации виртуального оборудования виртуальных машин 127
5.9.3. Резервное копирование параметров настройки средства виртуализации 128
5.9.4. Создание снимков текущего состояния машины 128
5.9.5. Резервное копирование и полная очистка журналов 128
5.10. Управление потоками информации в среде виртуализации 129
5.10.1. Управление сетевыми фильтрами nwfilter 129
5.10.2. Реализация собственных правил 133
5.11. Защита памяти в среде виртуализации 136

5.12. Применение механизма контроля целостности областей памяти по запросу из гостевой операционной системы . . . 139
5.13. Ограничение программной среды в среде виртуализации . . . 143
5.14. Централизованное управление . . . 143
5.14.1. Пример организации распределенного хранилища . . . 144
5.14.2. Пример миграции виртуальных машин . . . 151
6. Регистрация событий безопасности . . . 153
6.1. Правила регистрации событий . . . 153
6.2. Регистрация событий на основе меток безопасности . . . 155
6.3. Журнал аудита . . . 155
6.4. Средства управления аудитом . . . 155
6.4.1. Графические утилиты . . . 155
6.4.2. getfaud . . . 156
6.4.3. setfaud . . . 157
6.4.4. useraud . . . 158
6.4.5. psaud . . . 159
6.4.6. ausearch . . . 160
6.4.7. Дополнительные параметры регистрации событий . . . 163
6.5. Подсистема регистрации событий . . . 163
6.5.1. Регистрация событий и уведомление о событиях . . . 163
6.5.2. Журнал событий . . . 165
6.5.3. Самодиагностика подсистемы регистрации событий . . . 166
6.6. Регистрация событий в СУБД PostgreSQL . . . 167
6.6.1. Режимы регистрации событий . . . 167
6.6.2. Настройка маски регистрации событий . . . 167
6.6.3. Назначение списков регистрации событий в режиме `internal` . . . 169
6.6.4. Назначение списков регистрации событий в режиме `external` . . . 170
6.6.5. Назначение списков регистрации событий в режимах `external`, `internal` и `internal,external` . . . 171
6.6.6. Назначение списков регистрации событий в режиме `none` . . . 172
6.7. Средства централизованного аудита и протоколирования . . . 172
7. Изоляция процессов . . . 173
7.1. Изоляция процессов ОС . . . 173

7.2. Создание и защита изолированных программных сред (контейнеров) 173
7.2.1. Изоляция контейнеров и пространств 174
7.2.2. Выявление уязвимостей в образах контейнеров 175
7.2.3. Обеспечение корректности конфигурации контейнеров 176
7.2.4. Контроль целостности контейнеров и их образов 177
7.2.5. Регистрация событий безопасности, связанных с контейнерами 179
7.2.6. Идентификация и аутентификация пользователей 180
7.2.7. Работа с Docker в непривилегированном режиме с ненулевыми метками безопасности 181
7.2.7.1. Принцип функционирования 181
7.2.7.2. Управление запуском контейнера с ненулевой меткой безопасности 183
7.2.7.3. Копирование образа в репозиторий пользователя 183
7.2.7.4. Выполнение команд и запуск контейнеров в непривилегированном режиме от имени пользователя 184
8. Защита памяти 185
8.1. Очистка памяти 185
8.2. Средства ограничения прав доступа к страницам памяти 187
9. Контроль целостности 189
9.1. Средство подсчета контрольных сумм файлов и оптических дисков 189
9.2. Средство подсчета контрольных сумм файлов в deb-пакетах 190
9.3. Средство контроля соответствия дистрибутиву 190
9.4. Средства регламентного контроля целостности 191
10. Надежное функционирование 196
10.1. Восстановление ОС после сбоев и отказов 196
10.1.1. Комплекс программ Bacula 198
10.1.2. Утилита архивирования `tar` 198
10.2. Восстановление СУБД PostgreSQL после сбоев и отказов 199
10.2.1. Создание и восстановление резервных копий баз данных с мандатными атрибутами 200
10.2.2. pg_dump 201
10.2.3. pg_dumpall 202
10.2.4. pg_restore 203
10.3. Восстановление в режиме «Мобильный» 204

11. Фильтрация сетевого потока 206
11.1. Включение фильтрации сетевого потока 206
11.2. Фильтр сетевых пакетов `iptables` 206
11.3. Формирование правил 207
11.4. Порядок прохождения таблиц и цепочек 208
11.5. Механизм трассировки соединений 211
11.6. Критерии выделения пакетов 216
11.7. Действия и переходы 217
11.8. Поддержка фильтрации на основе классификационных меток 224
11.8.1. Модули `iptables` для работы с классификационными метками 224
11.8.2. Использование ufw для работы с классификационными метками 226
11.8.3. Использование Open vSwitch для работы с классификационными метками . . 227
12. Маркировка документов 230
12.1. Общие сведения 230
12.2. Настройка печати документа с ненулевой классификационной меткой 231
12.3. Настройка маркера печати 233
12.3.1. Файл описания переменных маркировки 234
12.3.2. Шаблон маркера 237
12.3.3. Файлы описания маркера 240
12.3.4. Изменение шрифта маркировки 241
13. Контроль подключения съемных машинных носителей информации 244
14. Сопоставление пользователя с устройством 245
15. Генерация КСЗ 246
16. Ограничение программной среды и функции безопасности 248
16.1. Замкнутая программная среда 248
16.1.1. Режимы функционирования 248
16.1.2. Настройка модуля digsig_verif 249
16.1.3. Подписывание файлов 253
16.2. Режим Киоск-2 256
16.2.1. Профили пакета `parsec-kiosk2` 256
16.2.2. Синтаксис профилей `parsec-kiosk2` 256
16.2.3. Синтаксис, совместимый с `parsec-kiosk` 258
16.2.4. Работа с Киоск-2 через консоль 258

16.2.4.1. Включение и выключение Киоск-2 . . . 258
16.2.4.2. Протоколирование процессов . . . 259
16.2.4.3. Создание профиля . . . 260
16.2.5. Графическая утилита управления профилями . . . 260
16.2.6. Киоск Fly . . . 260
16.3. Изоляция приложений . . . 261
16.4. Графический киоск в режиме «Мобильный» . . . 262
16.5. Функции безопасности системы . . . 262
16.5.1. Общие параметры вызова переключателей . . . 263
16.5.2. Монитор безопасности . . . 263
16.5.3. Установка квот на использование системных ресурсов . . . 265
16.5.4. Режим запрета установки бита исполнения . . . 265
16.5.5. Блокировка консоли для пользователей . . . 265
16.5.6. Блокировка консоли в режиме «Мобильный» . . . 266
16.5.7. Блокировка интерпретаторов . . . 266
16.5.8. Блокировка макросов . . . 267
16.5.9. Блокировка трассировки ptrace . . . 268
16.5.10. Блокировка клавиши <SysRq> . . . 268
16.5.11. Управление автоматическим входом . . . 269
16.5.12. Блокировка запуска программ пользователями . . . 269
16.5.13. Управление загрузкой ядра hardened . . . 270
16.5.14. Запуск контейнеров Docker на пониженном уровне МКЦ . . . 270
16.5.15. Управление сетевыми службами . . . 270
16.5.16. Управление загрузкой модуля ядра lkrg . . . 270
16.5.17. Блокировка неиспользуемых модулей ядра . . . 271
16.5.18. Блокировка автоматического конфигурирования сетевых подключений . . . 271
16.5.19. Отключение отображения меню загрузчика . . . 271
16.5.20. Ограничение на форматирование съемных носителей . . . 271
16.5.21. Запрет монтирования съемных носителей . . . 272
16.5.22. Включение на файловой системе режима работы «только чтение» . . . 272
16.5.23. Блокировка выключения компьютера пользователями . . . 272
16.5.24. Управление вводом пароля для sudo . . . 273
16.5.25. Блокировка использования утилиты sumac . . . 273

16.5.26. Управление межсетевым экраном ufw 273
16.5.27. Переключение уровней защищенности 273
16.5.28. Управление мандатным контролем целостности 275
16.5.29. Управление замкнутой программной средой 275
16.5.30. Управление безопасным удалением файлов 276
16.5.31. Управление очисткой разделов подкачки 276
16.5.32. Включение и выключение мандатного управления доступом 276
16.5.33. Управление AstraMode и MacEnable 277
16.6. Модуль безопасности yama 277
16.7. Модуль безопасности lockdown 278
17. Условия эксплуатации ОС 281
17.1. Обеспечение безопасности среды функционирования 281
17.2. Указания по эксплуатации ОС 282
17.3. Условия применения ПО 284
17.4. Условия исключения скрытых каналов 286
Перечень сокращений 288

1. ОБЩИЕ СВЕДЕНИЯ

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

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

1.1. Состав КСЗ

В состав КСЗ входят следующие основные подсистемы:

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

1.2. Контролируемые функции

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

- идентификацию и аутентификацию;
- управление доступом;
- регистрацию событий безопасности;
- ограничение программной среды;
- изоляцию процессов;
- защиту памяти;
- контроль целостности;

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

- надежное функционирование;
- фильтрацию сетевого потока;
- маркирование документов;
- защиту среды виртуализации;
- защиту средств контейнеризации;
- контроль подключения съемных машинных носителей информации (защиту ввода-вывода на отчуждаемый физический носитель информации).

Реализация перечисленных выше функций основана на следующих основных положениях:

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

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

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

- сущность-объект (объект) — пассивный компонент ОС (например, файл, сетевое соединение (файл-сокет), устройство (файл-устройство), файл-образ виртуальной машины или контейнера, как средства изоляции программной среды), доступ к которому регламентируется политиками управления доступом, к частям которого по отдельности управление доступом не осуществляется;
- сущность-контейнер (контейнер) — пассивный составной компонент ОС (например, каталог, том), доступ к которому регламентируется политиками управления доступом, состоящий из сущностей-объектов или сущностей-контейнеров, к которым по отдельности возможно осуществление управления доступом;

2) с каждым пользователем системы связан уникальный численный идентификатор — идентификатор пользователя (UID), который 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;

3) каждый пользователь входит в одну или более групп. Группа — это список пользователей системы, имеющий собственный идентификатор (GID). Поскольку группа объединяет несколько пользователей системы, в терминах политики безопасности она соответствует понятию «множественный субъект». GID является ярлыком множественного субъекта, которых у номинального субъекта может быть более одного. Таким образом, одному UID соответствует список GID;

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

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

1.3. 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 служб на базе механизмов NSS и PAM. Подобный подход

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

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

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

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

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

Более подробное описание ЕПП приведено в РУСБ.10015-37 95 01-1 «Операционная система специального назначения «Astra Linux Special Edition». Руководство администратора. Часть 1».

2. ИДЕНТИФИКАЦИЯ И АУТЕНТИФИКАЦИЯ

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

В общем случае идентификация и аутентификация охватывают:

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

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

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

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

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

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

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

Сценарии аутентификации (т. е. работа этих функций) описываются в конфигурационном файле `/etc/pam.conf` и в 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 (файл `/etc/passwd`) и локальной БД пользовательских паролей (файл `/etc/shadow`). Подробную информацию о структуре этих файлов можно получить с помощью команд `man 5 passwd` и `man 5 shadow` соответственно. В ОС реализована

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

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

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

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

Для управления пользователями, группами и 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 инструмент `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 уровней и категорий конфиденциальности: минимальный и максимальный уровни конфиденциальности, минимальный и максимальный наборы категорий конфиденциальности (см. раздел 4). Отсутствие установленных допустимых диапазонов мандатных атрибутов приводит к запрещению доступа при обращении к сетевым службам защищенных комплексов программ гипертекстовой обработки данных, электронной почты, СУБД и печати.

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

неуспешных попыток аутентификации пользователя может быть просмотрено следующей командой:

```
faillog -u <имя_пользователя>
```

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

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

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

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

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

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 прав доступа (ПРД) именованных субъектов (пользователей) к именованным сущностям (файлам, каталогам). Реализация механизма дискреционных ПРД обеспечивает создание для каждой пары (субъект-сущность) явного и недвусмысленного перечисления разрешенных типов доступа.

Дискреционное управление доступом применяется к каждой сущности и субъекту и заключается в том, что на защищаемые именованные сущности при их создании автоматически устанавливаются базовые ПРД в виде идентификаторов номинальных субъектов (UID и GID), которые вправе распоряжаться доступом к данной сущности, и прав доступа данных субъектов к созданной сущности.

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 сущности (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 сущности, являющейся файловым объектом, 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 сущностей 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 запускать файлы даже 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 на каталог можно только от имени суперпользователя с использованием механизма `sudo`. `Sticky`-бит каталога остается до тех пор, пока владелец каталога или суперпользователь не удалит каталог или не изменит права доступа. Владелец каталога 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 каждой сущности задавать права на доступ субъектов к ней.

ACL состоит из набора записей. Права доступа к сущности 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, каждой сущности 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 сущности, 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 сущности выполняется в два этапа. На первом этапе выбирается запись ACL, соответствующая сущности. Записи ACL для сущности 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 сущности. Если субъект доступа не является пользователем-владельцем сущности, то проверка продолжается. Если субъект доступа является пользователем-владельцем сущности, 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-владельцем сущности. Если ни одна из групп субъекта доступа не является группой-владельцем сущности, то проверка продолжается. Если одна из групп субъекта доступа явля-

ется группой-владельцем сущности, 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 субъекта 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 сущностей-контейнеров ACL, используемого по умолчанию. Названный ACL наследуется сущностями, создаваемыми в сущности-контейнере.

Сущностями 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 сущностей, являющихся файловыми объектами.

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 сущности, как указанные в битовой маске для трех классов (пользователя-владельца, группы-владельца и всех остальных), так и указанные в записях ACL, могут быть изменены субъектом, являющимся пользователем-владельцем сущности.

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

При использовании для сущности 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 сущности 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 принцип дискреционного управления доступом, предусматривает санкционированное изменение дискреционных ПРД, включая санкционированное изменение списка субъектов и списка защищаемых сущностей.

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, описание которых приведено в 4.7, обеспечивают работу с механизмом мандатного управления 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 4.16.

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

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.

Синтаксис команды:

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

Описание основных параметров команды приведено в таблице 1.

Таблица 1

Параметр	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. Данный параметр доступен, только если имеется системный вызов lchown
--from=ТЕКУЩИЙ_ВЛАДЕЛЕЦ:ТЕКУЩАЯ_ГРУППА	Изменить владельца и/или группу каждого файла, только если текущий владелец и/или группа совпадает с ТЕКУЩИЙ_ВЛАДЕЛЕЦ:ТЕКУЩАЯ_ГРУППА. 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=СФАЙЛ	Вместо заданных значений ВЛАДЕЛЕЦ:ГРУППА использовать владельца и группу файла СФАЙЛ
-R, --recursive	Рекурсивно изменять владельца каталогов и всего их содержимого
--help	Вывести справку и выйти

Окончание таблицы 1

Параметр	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

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

Синтаксис команды:

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

Описание параметров команды приведено в таблице 2.

Таблица 2

Параметр	Описание
`-c`, `--changes`	То же, что и `--verbose`, но выводить сообщение только тогда, когда действительно была изменена группа файла
`--dereference`	Изменить владельца файла, на который указывает символьная ссылка, вместо самой символьной ссылки
`-h`, `--no-dereference`	Изменить владельца символьной ссылки, а не владельца файла, на который указывает эта ссылка (доступна только на системах, имеющих системный вызов `lchown`)
`-f`, `--silent`, `--quiet`	Не выводить сообщения об ошибках
`--reference=RFILE`	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

Команда `chmod` применяется для изменения прав доступа.

Синтаксис команды:

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

Описание параметров команды `chmod` приведено в таблице 3.

Таблица 3

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

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.

Числовой режим состоит из не более четырех восьмеричных цифр (от 0 до 7), 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.

Формат символьного вида параметра `MODE`:

```
[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).

Пример

Команда chmod g-s file снимает бит set-group-ID (sgid), команда chmod ug+s file устанавливает биты suid и sgid, в то время как команда chmod o+s file ничего не делает

Описание параметров символьного вида MODE приведено в таблице 4.

Таблица 4

Параметр	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

Команда chmod не изменяет права на символьные ссылки, т. к. этого не делает системный вызов chmod (права символьных ссылок не используются). Однако для каждой символьной ссылки, заданной в командной строке, chmod изменяет права доступа связанного с ней файла. При рекурсивной обработке каталогов chmod игнорирует встречающиеся символьные ссылки.

3.3.4. umask

Команда umask — это пользовательская маска (user mask), которая используется для присвоения определенных прав доступа по умолчанию при создании файла или каталога.

Синтаксис команды:

```
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 задан параметр -S, выдается текущее значение маски. Параметр -S вызывает выдачу маски в символьном формате; по умолчанию выдается восьмеричное число. Если указан параметр -p, а маска не задана, 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 необходимый режим доступа к создаваемым файлам и каталогам.

3.3.5. getfacl

Для каждого файла getfacl выводит имя файла, владельца, группу-владельца и назначенные ACL. Если каталог имеет ACL по умолчанию (default), то getfacl выводит также ACL по умолчанию. ACL по умолчанию определяют списки доступа, которые будут назначаться создаваемым в данном каталоге объектам. Файлы не могут иметь ACL по умолчанию.

Синтаксис команды:

```
getfacl [-dRLP] <файл> ...
```

Формат вывода:

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

Строки 4, 6 и 9 относятся к 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, предоставляемые 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).

Описание параметров инструмента `getfacl` приведено в таблице 5.

Таблица 5

Параметр	Описание
`-a, --access`	Вывести только ACL файла
`-d, --default`	Вывести только ACL по умолчанию
`-c, --omit-header`	Не показывать заголовок (имя файла)
`-e, --all-effective`	Показать все эффективные права
`-E, --no-effective`	Не показывать эффективные права
`-s, --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
`-t, --tabular`	Использовать табулированный формат вывода
`-n, --numeric`	Показывать числовые значения пользователя/группы
`-p, --absolute-names`	Не удалять ведущие «/» из пути файла
`-h, --help`	Вывести справку и выйти
`-v, --version`	Вывести информацию о версии и выйти

3.3.6. setfacl

Команда `setfacl` 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.).

Синтаксис команды:

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

Описание параметров команды приведено в таблице 6.

Таблица 6

Параметр	Описание
-m, --modify=acl	Изменить текущий ACL для файла
-M, --modify-file=file	Прочитать записи ACL для модификации из файла
-x, --remove=acl	Удалить записи из ACL файла
-X, --remove-file=file	Прочитать записи ACL для удаления из файла
-b, --remove-all	Удалить все расширенные записи ACL
-k, --remove-default	Удалить ACL по умолчанию
--set=acl	Установить ACL для файла, заменив текущий ACL
--set-file=file	Прочитать записи ACL для установления из файла
--mask	Пересчитать маску эффективных прав
-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. Этот параметр не может использоваться совместно с другими параметрами, за исключением параметра --test
--test	Режим тестирования (ACL не изменяются)
-h, --help	Вывести справку и выйти
-v, --version	Вывести информацию о версии и выйти

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

При чтении ACL из файла при помощи параметров --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) права доступа отдельного пользователя:

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

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

2) права доступа отдельной группы:

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

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

3) маска эффективных прав:

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

4) права доступа всех остальных:

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

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 (от 0 до 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.

Сущности (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 также 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 PostgreSQL объектами дискреционного управления доступом могут быть столбцы таблицы, поскольку они однозначно идентифицируются по составному имени таблицы и столбца, т.к. имя столбца внутри таблицы является уникальным.

В то же время отдельная строка таблицы не является однозначно идентифицируемым объектом, и в общем случае дискреционные и любые другие правила разграничения доступа к ней применены быть не могут. Поскольку каждая строка идентифицируется только набором содержимого своих полей, то разработчику потребуется выбрать ту или иную процедуру идентификации строк в БД, например, создание первичного ключа или создание физического уникального идентификатора строки в БД.

Дополнительно для ограничения набора данных, выдаваемых пользователю, можно применять входящую в PostgreSQL систему фильтрации строк (POLICY) под названием ROW

LEVEL SECURITY — фильтровать строки, выдаваемые из таблицы указанному пользователю (пользователям) на основании вычисления заданного логического выражения.

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 также 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 сущностям не может быть предоставлено другим пользователям и доступно только администратору БД (при соответствующих настройках сервера может быть предоставлено и владельцу сущности).

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

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

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

Разграничение доступа к перечисленным операциям на уровне СУБД также 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. В общем случае, владельцем является создатель объекта или администратор БД. Право на создание (CREATE) применяется к объектам БД, являющимся 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, в случае отсутствия 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.

3.5. 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. МАНДАТНОЕ УПРАВЛЕНИЕ ДОСТУПОМ И МАНДАТНЫЙ КОНТРОЛЬ ЦЕЛОСТНОСТИ

4.1. Общие сведения

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

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

4.2. Мандатное управление доступом

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

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

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

4.2.1. Уровень конфиденциальности

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

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

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

Субъект с определенным уровнем конфиденциальности может получить доступ на чтение к сущности, если его уровень конфиденциальности не ниже уровня конфиденциальности сущности.

Субъект с определенным уровнем конфиденциальности может получить доступ на запись к сущности, если его уровень конфиденциальности совпадает с уровнем конфиденциальности сущности.

Таким образом, пользователь, если ему не присвоены специальные привилегии в соответствии с 4.7 и/или если файлам не заданы дополнительные атрибуты сущностей для мандатного управления доступом в соответствии с 4.2.3, может читать файлы, уровень конфиденциальности которых не превосходит его уровня доступа, а передавать данные только на одном уровне конфиденциальности.

4.2.2. Категория конфиденциальности

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

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

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

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

Субъект с определенным набором категорий конфиденциальности может получить доступ на чтение к сущности, если в его наборе категорий конфиденциальности присутствуют категории конфиденциальности, заданные данной сущности.

Субъект с определенным набором категорий конфиденциальности может получить доступ на запись к сущности, если его набор категорий конфиденциальности соответствует набору категорий конфиденциальности, заданному данной сущности.

Таким образом, пользователь, если ему не присвоены PARSEC-привилегии в соответствии с 4.7 и/или если файлам не заданы дополнительные атрибуты сущностей для мандатного управления доступом в соответствии с 4.2.3, может читать файлы, набор категорий конфиденциальности которых входит в набор доступных пользователю категорий, а изменять файлы — только если эти наборы совпадают.

4.2.3. Дополнительные атрибуты сущностей для мандатного управления доступом

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

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

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

Дополнительные атрибуты сущностей для мандатного управления доступом могут быть установлены (сняты) только субъектом с наличием привилегии `PARSEC_CAP_CHMAC` (см. таблицу 8) или субъектом с привилегией `root` и с максимальным уровнем целостности (такой субъект обладает привилегией `PARSEC_CAP_CHMAC` по умолчанию).

4.3. Мандатный контроль целостности

При реализации политики мандатного контроля целостности субъектам и сущностям задаются метки целостности (неиерархический уровень (категория) целостности и

иерархический (линейный) уровень целостности), описание метки целостности приведено в 4.3.1.

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

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

4.3.1. Метка целостности

Субъектам и сущностям задаются метки целостности — совокупность (декартово произведение) неиерархических уровней (категорий) целостности и иерархических (линейных) уровней целостности.

Метка целостности сущности отражает степень уверенности в целостности содержащейся в ней информации. Метка целостности субъекта соответствует его полномочиям по доступу к сущности в зависимости от ее метки целостности, а также отражает степень уверенности в корректности его функциональности.

Процесс при его непосредственном запуске наследует метку целостности процесса-родителя.

Субъект с определенной меткой целостности может получить доступ на запись к сущности, если его метка целостности не ниже метки целостности сущности.

В стандартной реализации иерархический (линейный) уровень целостности в ОС зарезервирован и не поддерживается его использование.

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

В ОС по умолчанию выделены нулевой, четыре ненулевых и несравнимых между собой (далее — изолированных) неиерархических уровня целостности, а также максимальный неиерархический уровень целостности, который не меньше всех остальных в системе.

При установке ОС по умолчанию предлагается максимальным неиерархический уровень целостности `max_ilev`, равный 63 (битовая маска 00111111), а минимальный всегда 0.

Дополнительно для обозначения максимального уровня целостности в установленной ОС зарезервировано специальное наименование уровня целостности `Высокий` (`High`), для обозначения нулевого уровня целостности зарезервировано специальное наименование `Низкий` (`Low`).

Непривилегированным пользователям по умолчанию присваивается нулевой уровень целостности, администратору присваивается максимальный уровень целостности 63,

за системными службами, перечень и описание которых приведены в таблице 7, зарезервированы четыре изолированных уровня целостности.

Таблица 7

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

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

После установки ОС максимальный уровень целостности в системе может быть повышен. Максимальными уровнями целостности в системе могут быть числа, у которых битовая маска включает битовые маски всех остальных используемых уровней целостности в системе, например, 63 (0x3F, битовая маска 00111111), 127 (0x7F, битовая маска 01111111), 191 (0xBF, битовая маска 10111111) и т.д.

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

4.3.2. Дополнительные атрибуты сущностей для мандатного контроля целостности

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

- `silev` — присваивается файлам. Процессу, запускаемому из файла с атрибутом `silev`, присваивается метка целостности, равная наибольшему значению, которое одновременно меньше или равно значениям метки целостности файла и максимальной метки целостности в системе (значение параметра командной строки ядра `parsec.max_ilev`). Например, если метка целостности файла меньше или равна максимальной метке целостности в системе, то процессу назначается метка целостности, равная метке целостности файла. Атрибут `silev` необходим для корректного запуска процесса смены пароля из файла `/usr/bin/passwd`, имеющего высокую метку целостности, пользователем с низкой меткой целостности.

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

- `irelax` — присваивается каталогам. Определяет, что в каталог может осуществлять запись процесс с любой меткой целостности. Создаваемые в данном каталоге сущности наследуют метку целостности от создающего их процесса, если его метка целостности меньше или равна метке целостности данного каталога. Если метка целостности процесса выше или несравнима с меткой целостности данного каталога, то метка целостности создаваемой в нем сущности устанавливается как наибольшее значение, которое одновременно меньше или равно значениям метки целостности данного каталога и метки целостности процесса. Атрибут доступен только при включенном расширенном режиме МКЦ (см. 4.5).

Дополнительные атрибуты сущностей для МКЦ могут быть установлены (сняты) только субъектом с привилегией `root` и с максимальным уровнем целостности.

4.4. Мандатный контекст безопасности

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

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

Метка безопасности состоит из классификационной метки и метки целостности:

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

Субъекты или сущности, которым явно не задан мандатный контекст безопасности, считаются имеющими минимальный (нулевой) мандатный контекст безопасности, т.е. метка безопасности имеет минимальные допустимые значения (например, равные нулю) и привилегии или дополнительные атрибуты отсутствуют.

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

Правила принятия решения о предоставлении доступа на основе метки безопасности описаны в 4.6.

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

ВНИМАНИЕ! Невозможен вход в сессию с выбранными одновременно ненулевой меткой конфиденциальности и ненулевым уровнем целостности.

ВНИМАНИЕ! При включенном в системе расширенном режиме МКЦ при входе в сессию уровень целостности назначается автоматически из максимально доступного данному пользователю (см. 4.5).

4.5. Расширенный режим мандатного контроля целостности

В режиме МКЦ процесс при его непосредственном запуске наследует уровень целостности процесса-родителя. При этом в расширенном режиме МКЦ (strict mode) непосредственный запуск процесса запрещен в том случае, если исполняемый файл, из которого запускается процесс, имеет уровень целостности меньше или несравнимый с уровнем целостности процесса-родителя. В данном случае процесс возможно запустить только с использованием инструмента `sumic`, описание которого приведено в 4.15.9.

Создаваемому файлу (каталогу) назначается уровень целостности, равный уровню целостности того каталога, в котором он создается. При этом запрещено создавать файл (каталог) с уровнем целостности выше или несравнимым с уровнем целостности процесса, создающего данный файл (каталог).

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

Включение расширенного режима МКЦ осуществляется путем выполнения от имени администратора команды:

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

В результате будут выполнены необходимые настройки метки целостности сущностей и для параметра командной строки ядра `parsec.strict_mode` установлено значение `1`. Для активации расширенного режима МКЦ необходимо перезагрузить ОС

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

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

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

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

При включении расширенного режима МКЦ домашним каталогам пользователей, для которых задан ненулевой уровень целостности, присваивается уровень целостности, соответствующий максимально доступному уровню целостности для данного пользователя. В дальнейшем при назначении пользователю ненулевого уровня целостности также следует его домашнему каталогу назначить уровень целостности, соответствующий максимально доступному уровню целостности для данного пользователя. Если уровень целостности его домашнему каталогу не будет назначен, то после перезагрузки ОС он будет назначен автоматически.

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

1) вход в сессию возможен только на максимально доступном пользователю неиерархическом уровне целостности (выбирается автоматически). Т.е. администратор с высоким уровнем целостности, если при входе в сессию была выбрана нулевая классификационная метка, не сможет выполнить вход на нулевом уровне целостности. И наоборот, если администратором с высоким уровнем целостности при входе в сессию была выбрана ненулевая классификационная метка, то вход будет выполнен на нулевом уровне целостности;
2) не применяется привилегия `PARSEC_CAP_IGNMACINT` (см. таблицу 8);
3) не применяется параметр командной строки ядра `parsec.ccnr_relax` (см. таблицу 31);
4) возможно применение привилегии `PARSEC_CAP_CCNR_RELAX` (см. таблицу 8);
5) возможно применение атрибута `irelax` (см. 4.3.2).

4.6. Применение правил мандатного управления доступом и мандатного контроля целостности

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

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

Использование уровня целостности обеспечивает целостность информации путем запрета модификации сущностей с высоким уровнем целостности недоверенными субъектами с более низким уровнем целостности.

Операции чтения и исполнения разрешены, если уровень конфиденциальности субъекта не ниже уровня конфиденциальности сущности, биты набора категорий конфи-

денциальности субъекта включают биты набора категорий конфиденциальности сущности. Разрешение не зависит от значений метки целостности.

Операция записи разрешена, если уровни и категории конфиденциальности субъекта и сущности совпадают, а уровень целостности субъекта больше или равен уровню целостности сущности.

В отношении метки безопасности действуют следующие правила наследования:

- если в сессии субъект создает другого субъекта (процесс создает процесс), то созданный субъект полностью наследует метку безопасности (уровень конфиденциальности, категории конфиденциальности, уровень целостности);
- если субъект создает сущность (процесс создает файл), то созданная сущность наследует только классификационную метку (уровень конфиденциальности и категории конфиденциальности), и, независимо от уровня целостности субъекта, всегда получает нулевой уровень целостности.

Классификационные метки сущностей не могут превышать значения классификационной метки контейнера, их содержащего.

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

ВНИМАНИЕ! Изменять классификационную метку сущности (т.е. изменять уровень конфиденциальности и/или категории конфиденциальности) может только субъект, имеющий уровень целостности не ниже уровня целостности этой сущности и обладающий привилегией `PARSEC_CAP_CHMAC` (см. таблицу 8), или субъект с привилегией `root` и с максимальным уровнем целостности.

ВНИМАНИЕ! Понижать уровень целостности сущности может только субъект с наличием привилегии `PARSEC_CAP_CHMAC` (см. таблицу 8) и имеющий уровень целостности не ниже уровня целостности этой сущности. Произвольно изменять уровень целостности сущности может только субъект с привилегией `root` и с максимальным уровнем целостности.

ВНИМАНИЕ! Для изменения классификационной метки (т.е. уровня конфиденциальности и/или категории конфиденциальности) процесса используется привилегия `PARSEC_CAP_SETMAC` (см. таблицу 8).

ВНИМАНИЕ! Для процесса возможно только понизить уровень целостности. Для выполнения этого безопасным способом, т. е. с закрытием наследуемых от родительского процесса файловых дескрипторов, рекомендуется использовать утилиту `sumic` (см. 4.15.9) и библиотечную функцию `pdp_set_pid_safe`.

Для создания в контейнере, имеющем атрибут ccnr, вложенной сущности с уровнем и категориями меньшими, чем у контейнера, необходимо обладать специальными привилегиями PARSEC_CAP_IGNMACCAT и PARSEC_CAP_IGNMACLVL, описание которых приведено в 4.7.

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

Мандатный контекст безопасности на корне файловой системы определяет максимальный мандатный контекст безопасности сущностей. Мандатный контекст безопасности, устанавливаемый по умолчанию на корень файловой системы, а также мандатный контекст отдельных сущностей, определен в сценарии /usr/sbin/pdp-init-fs.

4.7. PARSEC-привилегии

PARSEC-привилегии, как и Linux-привилегии, приведенные в 3.2, наследуются процессами от процессов-родителей. Процессы, запущенные от имени суперпользователя, имеющего максимальный (Высокий) уровень целостности по умолчанию, независимо от явного назначения им привилегий, имеют возможность осуществлять большинство привилегированных действий.

PARSEC-привилегии и их описание приведены в таблице 8.

Таблица 8

Привилегия Битовая маска	Описание
PARSEC_CAP_AUDIT 0x00002	Позволяет управлять политикой аудита
PARSEC_CAP_BYPASS_XATTR 0x40000	Отключает проверку подписи файлов в xattr
PARSEC_CAP_CAP 0x00400	Позволяет процессу назначать себе любой непротиворечивый набор привилегий и читать привилегии, присвоенные процессам
PARSEC_CAP_CCNR_RELAX 0x100000	Позволяет осуществлять в каталоге с установленным атрибутом ccnr действия (создание, удаление и др.) над вложенными файлами и каталогами с уровнями конфиденциальности, не выше уровня конфиденциальности данного каталога. Привилегия применяется при включенном расширенном режиме МКЦ (см. 4.5)

Продолжение таблицы 8

Привилегия Битовая маска	Описание
PARSEC_CAP_CHMAC 0x00008	Позволяет субъекту изменять метку безопасности сущности, если уровень целостности сущности меньше или равен уровню целостности субъекта. Привилегия позволяет изменять атрибуты сущностей для мандатного управления доступом (см. 4.2.3). Привилегия позволяет только понижать уровень целостности сущности. Привилегия не позволяет изменять атрибуты сущностей для МКЦ (см. 4.3.2)
PARSEC_CAP_FILE_CAP 0x00001	Не используется. Для изменения меток безопасности файловых объектов использовать привилегию PARSEC_CAP_CHMAC. Для изменения меток безопасности процессов использовать PARSEC_CAP_SETMAC. Для изменения привилегий процессов использовать PARSEC_CAP_CAP
PARSEC_CAP_IGNMACCAT 0x00020	Позволяет игнорировать мандатную политику по категориям доступа
PARSEC_CAP_IGNMACINT 0x02000	Для данного процесса отключает проверку правил доступа на основе уровней целостности. Привилегия не применяется при включенном расширенном режиме МКЦ (см. 4.5)
PARSEC_CAP_IGNMACLVL 0x00010	Позволяет игнорировать мандатную политику по уровням конфиденциальности
PARSEC_CAP_INHERIT_INTEGRITY 0x20000	При создании файловых объектов на них автоматически устанавливается максимально возможная целостность, однако не выше, чем целостность процесса-создателя и целостность контейнера, в котором создается файловый объект. Используется для установщика пакетов (dpkg)
PARSEC_CAP_IPC_OWNER 0x10000	Отменяет мандатные ограничения при работе с сущностями IPC, такими как shared memory, message queue и т.д. (PARSEC-привилегия, аналогичная Linux-привилегии CAP_IPC_OWNER)
PARSEC_CAP_MAC_SOCK 0x00800	Позволяет изменять метки безопасности точек соединения (UNIX-сокетов). Для сетевых сокетов использовать привилегию PARSEC_CAP_PRIV_SOCK
PARSEC_CAP_PRIV_SOCK 0x00100	Позволяет создавать новые сетевые сокеты процесса в привилегированном режиме (атрибут ehole). Привилегированный сокет позволяет осуществлять сетевое взаимодействие, игнорируя мандатную политику. Для точек соединения (UNIX-сокетов) использовать привилегию PARSEC_CAP_MAC_SOCK
PARSEC_CAP_PROCFS 0x80000	Включает игнорирование уровней конфиденциальности и уровней целостности при работе с /proc
PARSEC_CAP_READSEARCH 0x00200	Позволяет игнорировать мандатную политику при чтении и поиске файловых объектов (но не при записи)

Окончание таблицы 8

Привилегия Битовая маска	Описание
PARSEC_CAP_SETMAC 0x00004	Позволяет изменять классификационную метку текущего процесса. Привилегия позволяет только добавлять категории конфиденциальности и повышать уровень конфиденциальности
PARSEC_CAP_SIG 0x00040	Позволяет посылать сигналы процессам, игнорируя мандатные права
PARSEC_CAP_SUMAC 0x04000	Позволяет запускать процессы с другой классификационной меткой (с другим иерархическим уровнем конфиденциальности и другими неиерархическими категориями доступа)
PARSEC_CAP_UNSAFE_SETXATTR 0x01000	Позволяет устанавливать мандатные атрибуты объектов файловой системы без учета мандатных атрибутов родительского контейнера. Привилегия используется для восстановления объектов файловой системы из резервных копий и только после установки значения «1» для параметра /parsecfs/unsecure_setxattr
PARSEC_CAP_UPDATE_ATIME 0x00080	Позволяет изменять время доступа к файловым объектам

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

4.8. Включение и выключение мандатного управления доступом

4.8.1. Включение мандатного управления доступом

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

Включение мандатного управления доступом после установки ОС выполняется с помощью инструмента `astra-mac-control`, описанного в 16.5.32, или графической утилиты `fly-admin-smc` (см. электронную справку).

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

4.8.2. Выключение мандатного управления доступом

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

вести к несогласованному состоянию объектов файловой системы с разными уровнями конфиденциальности.

Выключение мандатного управления доступом выполняется с помощью инструмента `astra-mac-control`, описанного в 16.5.32, или графической утилиты `fly-admin-smc`.

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

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

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

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

4.9. Включение и выключение мандатного контроля целостности

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

Включение и выключение МКЦ после установки ОС выполняется с помощью инструмента `astra-mic-control`, описанного в 16.5.28, или графической утилиты `fly-admin-smc` (см. электронную справку). При включении/выключении МКЦ автоматически включается/выключается МКЦ на файловой системе (см. 4.10).

При включении МКЦ для параметра командной строки ядра `parsec.max_ilev` в загрузчике ОС устанавливается значение максимального уровня целостности в системе. По умолчанию значение максимального уровня целостности в системе 63 (если при включении МКЦ не было указано другое значение, см. 16.5.28).

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

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

4.10. Мандатный контроль целостности на файловой системе

При включении МКЦ согласно 4.9 объектам файловой системы автоматически присваиваются атрибуты МКЦ.

Присвоение атрибутов МКЦ объектам файловой системы осуществляется в соответствии с конфигурационным файлом `/etc/parsec/fs-ilev.conf`. В данном конфигурационном файле перечислены объекты файловой системы и их уровень целостности в формате:

`<уровень_целостности> <путь>`

где `<уровень_целостности>` — уровень целостности для объекта файловой системы, указанного в `<путь>`;

`<путь>` — объект/объекты файловой системы или путь к ним.

Значения, указываемые в конфигурационном файле в качестве уровня целостности `<level>`, приведены в таблице 9.

Таблица 9

Значение	Описание
`<число>`	Определенный уровень целостности, заданный числовым значением. Может быть десятичным, восьмеричным, шестнадцатеричным или двоичным числом
`high`	Текущий `max_ilev` — максимальный уровень целостности в ОС, заданный в параметре командной строки ядра `parsec.max_ilev`
`max`	Текущий `max_ilev` — максимальный уровень целостности в ОС, заданный в параметре командной строки ядра `parsec.max_ilev`
`low`	То же что и нулевой уровень целостности
`min`	То же что и нулевой уровень целостности
`exc`	Игнорировать файл при проверке целостности. В качестве символа подстановки в конце пути можно использовать символ «*»

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

Пример

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

```
exc     /etc/xdg/autostart/vboxclient.desktop
exc     /etc/X11/Xsession.d/98vboxadd-xclient
exc     /etc/ld.so.*
exc     /etc/resolv.conf
exc     /root/.config/*
exc     /root/.gnupg/gpg-agent.conf
max     /etc
```

```
max     /lib
max     /lib64
max     /lib32
max     /bin
max     /sbin
max     /boot
max     /root
max     /opt
max     /srv
max     /usr
```

Для управления МКЦ на файловой системе используется инструмент командной строки `set-fs-ilev`.

Пример

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

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

Подробное описание инструмента `set-fs-ilev` приведено в `man set-fs-ilev`.

Также для управления МКЦ на файловой системе может использоваться графическая утилита `fly-admin-smc` (см. электронную справку).

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

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

4.11. Администрирование ОС при включенном МКЦ

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.

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).

ВНИМАНИЕ! Вход в систему привилегированным пользователем (администратором) необходим только для выполнения настроек системы и только с высоким уровнем целостности. Для обычного (штатного) режима работы рекомендуется осуществлять вход в систему от имени непривилегированного пользователя на низком уровне целостности.

ВНИМАНИЕ! При включенном в системе расширенном режиме МКЦ при входе в сессию уровень целостности назначается автоматически из максимально доступного данному пользователю.

4.12. 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 (метку):

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

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

Уровень целостности назначается только в пределах `max_ilev`.

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

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

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

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

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

где через пробел могут быть перечислены PARSEC-привилегии.

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

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

Для просмотра метки службы выполнить команду:

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

где <PID> — идентификатор службы.

Для просмотра установленных на службе PARSEC-привилегий выполнить команду:

```
pscaps <PID>
```

Для определения PID используется команда:

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

4.13. Сетевое взаимодействие. Механизм privsock

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

Взаимодействие при помощи сетевого протокола IPv4 (IPv6) осуществляется через программный интерфейс объектов доступа, являющихся элементами межпроцессного и сетевого взаимодействия (например, сетевых сокетов), которые обеспечивают обмен данными между процессами в рамках одной или нескольких ОС, объединенных в локальную вычислительную сеть.

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

При необходимости для обеспечения целостности заголовка IP-пакетов, содержащего классификационную метку, допускается применение программного средства OpenVPN. Описание использования OpenVPN приведено в документе РУСБ.10015-37 95 01-1.

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

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

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

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

Пример

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

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

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

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

Пример

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

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

Подробное описание механизма `privsock` приведено в `man privsock`.

4.14. Шина межпроцессного взаимодействия D-Bus

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

Сессионная шина создается для каждой пользовательской сессии при ее запуске. Она работает с меткой безопасности пользователя системы и все взаимодействующие через нее процессы также имеют данную метку безопасности. Взаимодействие процессов с

иной меткой безопасности не происходит. Дополнительные сессионные шины могут быть созданы с использованием утилиты `dbus-launch`, которая позволяет запустить процесс одновременно с созданием новой сессионной шины. При этом имеется возможность указать, используя переменную `DBUS_SESSION_BUS_ADDRESS`, какую из сессионных шин данный процесс будет воспринимать как сессионную шину.

Пример

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

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

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

4.14.1. Виды сообщений

D-Bus поддерживает следующие виды сообщений:

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

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

4.14.2. Процесс взаимодействия с шиной

Для идентификации соединения на шине используются параметры, приведенные в таблице 10.

Таблица 10

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

Для вызова метода необходимо:

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

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

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

4.14.3. Процесс соединения с системной шиной

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

позволяют субъекту (dbus) игнорировать метку безопасности соединения, пытающегося выполнить его метод.

После этого формируется ответное сообщение типа `method_return`, которое возвращает ответный статус об успешном создании соединения с шиной. Одновременно происходит установка владельца службы. Под службой понимается имя на шине, которое соответствует некоторому соединению. При этом владельцами уникальных имен (вида `:1.6`) могут быть только сами соединения. Если используется пользовательское имя, например, `org.share.linux`, то его владельцем может быть соединение с некоторым уникальным именем. Другие процессы (соответственно с другими уникальными именами соединений) также могут запросить доступ владения данным именем. В зависимости от настроек может произойти как замена владельца, так и установка его в очередь на владение. При этом, когда текущий владелец освободит данное имя (отключится от шины), владельцем станет следующее по списку соединение (если такие есть). Важно, что именно владелец, т. е. соединение и соответствующий ему процесс, будет обрабатывать адресованное данному имени (службе) сообщение. В дополнение к ответу источнику сообщения `method_call Hello` происходит формирование сигнала `signal`. D-Bus формирует сигнал `NameAquired` (т. е. источником является D-Bus), который сообщает, что имя (в данном случае UCN) получено.

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

- разрешить замену владельца. Относится к данной службе — 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 созданной службы;
- не ставить в очередь, если владельца нельзя сейчас заменить. Иначе данное соединение встанет в конец очереди, и когда будут освобождены все владельцы — оно станет владельцем.

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

4.14.4. Объекты и субъекты системы

В рамках D-Bus объектами и субъектами являются сущности, сопоставленные с именами соединений и служб (т. е. уникальными именами и общеизвестными именами). Соответственно, каждое соединение имеет свою метку безопасности. Кроме этого, при создании службы (получения имени через `dbus_request_name`), а также создании уникального

имени, происходит назначение им метки безопасности. Сведения о метках безопасности содержатся в хеш-таблице, где каждому имени соответствует метка безопасности. При освобождении имени, когда последний владелец уходит из очереди или клиент отсоединяется от шины, выполняется удаление соответствующей записи таблицы (при использовании LOCKED-настройки метка безопасности остается).

Проверка pdp доступа осуществляется при следующих событиях:

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

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

4.14.5. Алгоритм проверки меток для различных сообщений и режимов работы

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

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

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

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

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

Для получения владения службой в методах acquire_service, add_owner, swap_owner функция bus_pdplinux_service_allows_connection_to_service вызывается с указанием псевдодоступа ETOS_ADDOWNER (для acquire_service используется ETOS_ADDOWNER_ACQUIRE_SERVICE), которая в зависимости от настроек параметра конфигурации <pdplinux_allow_different_owners> (BUS_CONTEXT_PDPLINUX_GET_pdplinux_allow_different_owners) проверяет наличие потенциальных владельцев в очереди с меткой безопасности, отличной от метки безопасности службы. Если параметр pdplinux_allow_different_owners установлен, то возможно получение владения службой.

Для непосредственной проверки возможности замены владельца службы модифицирована функция bus_service_get_allow_replacement(), которая выполняет проверку целостности и не позволяет проводить соответствующую замену в случае, если целостность службы 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 безопасности источника сообщения и соответствующей службы, совпадающей с ее текущим владельцем. Если метки безопасности равны, то доступ разрешается. Иначе вызывается функция bus_pdplinux_allows_execution(), которая разрешает низкоуровневым объектам запуск функций объекта более высокого уровня. Если запуск запрещен, то проверяются привилегии. Например, если служба, метод которой запускается, имеет привилегии 0x30, то запуск разрешается. Также существует возможность запуска метода, если соединение-получатель имеет дополнительный атрибут ehole в функции bus_pdplinux_allows_execution_helper().

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

4.14.6. Привилегии процесса dbus-daemon

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

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

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

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

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

```
CapabilitiesParsec=PARSEC_CAP_PRIV_SOCK PARSEC_CAP_IGNMACCAT PARSEC_CAP_IGNMACLVL
```

4.14.7. Расширенное управление политиками

4.14.7.1. Конфигурационный файл

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

Пример

Конфигурационный файл шины accessibility

```
<!DOCTYPE busconfig PUBLIC "-//freedesktop//DTD D-Bus Bus Configuration 1.0//EN"
    "http://www.freedesktop.org/standards/dbus/1.0/busconfig.dtd">
```

```
<busconfig>
  <type>accessibility</type>
<servicedir>/usr/share/dbus-1/accessibility-services</servicedir>
  <auth>EXTERNAL</auth>
  <listen>unix:tmpdir=/tmp</listen>

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

```
    <deny send_interface="org.a11y.atspi.Document" send_member=
       "GetAttributeValue"/>
  </policy>
```

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

Системная шина имеет политику по умолчанию, которая запрещает отправку сообщений типа method_call и получение владения службами на шине (own). Остальные действия, в частности, ответы на сообщения (reply), получение ошибок (error) и сигналов (signal) по умолчанию в политике разрешены.

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

Секции <policy> назначается один из четырех параметров:

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

Политики применяются к соединениям в следующем порядке:

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

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

Возможные параметры правил политик приведены в таблице 11. Параметры правил <deny> определяют необходимость запрета указанного в параметрах сообщения.

Таблица 11

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

Окончание таблицы 11

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

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 службы 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 служб. 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"/>

4.14.7.2. Формирование клиентских политик из политик шины

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

Пример

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

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

Инициализация обработчика в функции XML_SetElementHandler:

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

При анализе вызовов формирование политик происходит следующим образом:

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

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

4.15. Средства управления мандатными ПРД

Для управления локальными мандатными ПРД используются следующие графические утилиты:

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

Более подробное описание утилит см. в электронной справке.

Для управления локальными мандатными ПРД в режиме командной строки используются следующие инструменты:

- pdpl-file — управление мандатными атрибутами файлов, описание приведено в 4.15.1;
- pdp-id— отображение мандатных атрибутов сессии пользователя ОС, описание приведено в 4.15.2;
- pdp-init-fs — сценарий инициализации мандатных атрибутов ФС, описание приведено в 4.15.3;
- pdp-ls— вывод аналогично стандартной команде ls информации о файлах с отображением мандатных атрибутов, описание приведено в 4.15.4;
- pdpl-ps— управление мандатными атрибутами процессов, описание приведено в 4.15.5;
- pdpl-user— управление допустимыми уровнями конфиденциальности и категориями пользователей ОС, описание приведено в 4.15.6;
- pdp-exec — запуск процессов в заданном окружении, описание приведено в 4.15.7;
- sumac— запуск процесса с заданными мандатными уровнем и категорией в отдельной графической сессии, описание приведено в 4.15.8;
- sumic — запуск процесса на пониженном (заданном) уровне целостности, описание приведено в 4.15.9;
- userlev — изменение БД уровней конфиденциальности, описание приведено в 4.15.10;
- usercat— изменение БД категорий конфиденциальности, описание приведено в 4.15.11.

Для обеспечения совместимости в ОС сохранены следующие утилиты командной строки для управления мандатными ПРД:

- `chmac` — управление мандатными атрибутами файлов, описание приведено в 4.15.12.1;
- `lsm` — вывод, аналогично стандартной команде `ls`, информации о файлах с отображением мандатных атрибутов, описание приведено в 4.15.12.3;
- `macid` — отображение мандатных атрибутов сессии пользователя ОС, описание приведено в 4.15.12.2;
- `psmac` — управление мандатными атрибутами процессов, описание приведено в 4.15.12.4;
- `usermac` — управление допустимыми уровнями конфиденциальности и категориями пользователей ОС, описание приведено в 4.15.12.5;
- `getfmac` — получение меток безопасности файловых объектов, описание приведено в 4.15.12.6;
- `setfmac` — изменение меток безопасности файловых объектов, описание приведено в 4.15.12.7.

Для управления мандатными ПРД в ЕПП используются следующие графические инструменты:

- web-интерфейс FreeIPA — если развернута служба FreeIPA;
- `fly-admin-smc` («Управление политикой безопасности») — если развернута служба ALD.

Для управления мандатными ПРД в ЕПП в режиме командной строки используются следующие инструменты:

- `ipa user-mod` — если развернута служба FreeIPA (подробная информация по использованию инструмента командной строки доступны в справочной странице `ipa help user-mod`);
- `ald-admin` — если развернута служба ALD (подробная информация по использованию инструмента командной строки доступны в справочной странице `man ald-admin`).

4.15.1. pdpl-file

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

Синтаксис инструмента:

```
pdpl-file [<параметр>[...]]
    [<уровень_конфиденциальности>][:<уровень_целостности>
```

```
[:<категория_конфиденциальности>[:<дополнительный_атрибут>]]] [<сущность>]
```

Уровень и категория конфиденциальности могут быть заданы именем или шестнадцатеричным значением.

Пример

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

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

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

Пример

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

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

Пример

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

Описание параметров инструмента `pdpl-file` приведено в таблице 12.

Таблица 12

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
`-u`, `--unite`	Объединить текущую метку безопасности сущности с указанной в качестве аргумента
`-s`, `--subtract`	Вычесть из текущей метки безопасности сущности метку безопасности, указанную в качестве аргумента. При этом для итоговой метки безопасности значения задаются по следующим правилам: - уровень конфиденциальности — минимальное значение из текущей метки безопасности и указанной в качестве аргумента; - уровень целостности — минимальное значение из текущей метки безопасности и указанной в качестве аргумента (должно быть указано в десятичном виде); - категории конфиденциальности — из текущей метки безопасности вычитаются категории, указанные в качестве аргумента; - дополнительные атрибуты — из текущей метки безопасности вычитаются дополнительные атрибуты, указанные в качестве аргумента
`-R`, `--recursive`	Применить рекурсивно

Окончание таблицы 12

Опция	Описание
`-r, --reverse`	Сначала файлы в каталоге, потом каталог
`-h, --help`	Вывести справку и выйти
`--version`	Вывести информацию о версии и выйти

4.15.2. pdp-id

Синтаксис:

```
pdp-id [параметры]
```

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

Параметры приведены в таблице 13.

Таблица 13

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

При отсутствии параметров 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 конфиденциальности `Секретно`, минимальный уровень целостности и категорию `Категория_А`.

4.15.3. pdp-init-fs

Синтаксис:

```
pdp-init-fs
```

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

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

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

4.15.4. pdp-ls

Синтаксис:

```
pdp-ls [параметры] [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 метка безопасности, то к ACL добавляется символ «m»;
- если на файле установлены списки регистрации событий, то к ACL добавляется символ «a»;
- доступен параметр `-M`, который может быть использован для просмотра меток безопасности на файловых объектах.

4.15.5. 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 14.

Таблица 14

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

4.15.6. 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 уровни конфиденциальности и категории пользователей ОС.

Опции приведены в таблице 15.

Таблица 15

Опция	Описание
`-d, --delete`	Удалить строку пользователя из файла
`-z, --zero`	Обнулить значения уровней и категорий
`-l, --levels`	Установить допустимые 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 `pdpl-user` при успешном выполнении всегда выводит значения установленных допустимых меток безопасности.

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

```
pdpl-user пользователь
```

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.

4.15.7. pdp-exec

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

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

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

Синтаксис инструмента:

```
pdp-exec [параметр[параметр...]] [--] [<команда>] [<параметры_запуска_команды>]
```

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

Описание параметров инструмента pdp-exec приведено в таблице 16.

Таблица 16

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

Привилегии задаются в виде битовой маски (как правило, в шестнадцатеричном виде). Соответствие отдельных битов полномочиям приведено в man parsec_capset, а также в таблице 8.

Метка безопасности задается в виде:

```
[<уровень_конфиденциальности>][:<уровень_целостности>
    [:<категория_конфиденциальности>]]
```

Примеры:

1. Перезапустить службу dbus с PARSEC-привилегией PARSEC_CAP_PRIV_SOCK:

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

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

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

Более подробное описание pdp-exec приведено в man pdp-exec.

4.15.8. sumac

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

Синтаксис инструмента:

```
sumac [<параметр>] [<команда>]
```

ВНИМАНИЕ! Для запуска процесса на уровне, отличном от уровня текущей сессии, не должна стоять блокировка использования sumac в графической утилите fly-admin-smc (см. электронную справку) и пользователю должна быть назначена привилегия PARSEC_CAP_SUMAC (см. 4.7).

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

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

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

Окно графического приложения, запущенного в отдельной сессии на другом уровне, будет иметь цветную рамку, цвет которой соответствует уровню конфиденциальности приложения. Описание цветовой индикации приведено в документе РУСБ.10015-37 93 01 «Операционная система специального назначения «Astra Linux Special Edition». Руководство пользователя».

Пример

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

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

Параметры инструмента sumac приведены в таблице 17.

Таблица 17

Параметр	Описание
-l , --level=	Запустить процесс с указанным уровнем конфиденциальности
-c , --category=	Запустить процесс с указанной категорией конфиденциальности
-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

4.15.9. sumic

Инструмент sumic позволяет запускать процессы на уровне целостности ниже, чем уровень целостности процесса-родителя. При этом запускаемый с использованием sumic процесс будет иметь уровень целостности не выше уровня целостности исполняемого файла, из которого он запущен. При запуске процесса с помощью sumic запрещается наследование открытых в процессе-родителе ресурсов, имеющих уровень целостности, превышающий уровень целостности создаваемого процесса. Запуск графической утилиты с использованием sumic выполняется в изолированном X-сервере.

Синтаксис инструмента:

```
sumic [параметр] [--] [<команда>] [<параметры_запуска_команды>]
```

Описание параметров инструмента sumic приведено в таблице 18.

Таблица 18

Параметр	Описание
-i <уровень_целостности>	Уровень целостности, на котором будет запущен процесс указанной команды. В случае отсутствия параметра процесс будет запущен на нулевом уровне целостности
-v, --version	Вывести информацию о версии и выйти
-h, --help	Вывести справку и выйти

4.15.10. 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 уровней конфиденциальности. Для просмотра всех уровней команду следует запускать без параметров. Вносить изменения в БД уровней может только администратор.

Опции приведены в таблице 19.

Таблица 19

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

4.15.11. 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 20.

Таблица 20

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

4.15.12. Устаревшие утилиты управления мандатными ПРД

Для обеспечения совместимости в 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, отличном от определенного в ОС формата.

4.15.12.1. chmac

Синтаксис:

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

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

Опции приведены в таблице 21.

Таблица 21

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_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).

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 уровней и категорий конфиденциальности при выполнении операций чтения, записи и исполнения.

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

- `equ` — игнорирование уровней и категорий конфиденциальности при выполнении операций чтения, записи и исполнения;

- equ_w — игнорирование уровней и категорий конфиденциальности при выполнении операции записи;

- low — игнорирование уровней и категорий конфиденциальности при выполнении операций чтения и исполнения.

ВНИМАНИЕ! В настоящее время используется другой набор специальных атрибутов. Данная утилита 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
```

4.15.12.2. macid

Синтаксис:

```
macid [параметры]
```

Команда macid выводит мандатные атрибуты сессии пользователя ОС.

Опции приведены в таблице 22.

Таблица 22

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

При отсутствии параметров 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.

4.15.12.3. lsm

Синтаксис:

```
lsm [параметры] [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 метка безопасности, то к ACL добавляется символ «m»;
- если на файле установлены списки регистрации событий, то к ACL добавляется символ «a»;
- доступен параметр `-M`, который может быть использована для просмотра меток безопасности на файловых объектах.

4.15.12.4. psmac

Синтаксис:

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

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

Если аргумент-метка безопасности присутствует, то команда устанавливает заданную метку безопасности на процесс. Метка безопасности 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 23.

Таблица 23

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

4.15.12.5. usermac

Синтаксис:

```
usermac [-dzhv [-m минимальный уровень:максимальный уровень]
```

```
[-c минимальная категория:максимальная категория]] пользователь
```

Команда `usermac` изменяет допустимые уровни и категории конфиденциальности пользователей ОС.

Опции приведены в таблице 24.

Таблица 24

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

Чтобы просмотреть текущие допустимые метки безопасности, 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.

4.15.12.6. 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 меток безопасности файловых объектов. Информация о метках безопасности посылается на стандартный вывод и может являться входными данными для команды `setfmac` (см. 4.15.12.7).

Опции приведены в таблице 25.

Таблица 25

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 безопасности 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

4.15.12.7. setfmac

Синтаксис:

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

Команда `setfmac` устанавливает метки безопасности на файлы. Метки безопасности 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` (4.15.12.6).

Метка безопасности 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 метки безопасности на файлы.

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

Опции приведены в таблице 26.

Таблица 26

Опция	Описание
-s, --set	Установить метку безопасности из командной строки
-m, --modify	Изменить метку безопасности из командной строки
-S, --set-file	Установить метки безопасности из файла
-B, --restore	Восстановить метки безопасности 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

4.16. 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, описание которых приведено, соответственно, в 4.16.1, 4.16.2 и 4.16.3.

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

4.16.1. usercaps

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

Синтаксис инструмента usercaps:

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

В качестве значений <linux-привилегии> и <PARSEC-привилегии> указывается слово или слова, разделенные запятой «,» или двоеточием «:». Каждое слово может быть строкой в верхнем или нижнем регистре или числом, перед которым стоит знак плюс «+» или знак минус «-». Верхний регистр для строк и знак «+» для чисел устанавливают соответствующую привилегию пользователю, нижний регистр и знак «-» снимают соответствующую привилегию с пользователя.

Список возможных Linux-привилегий с названиями в строковом и числовом виде можно вывести, выполнив команду:

```
usercaps -L
```

Список возможных PARSEC-привилегий с названиями в строковом и числовом виде можно вывести, выполнив команду:

```
usercaps -M
```

Для просмотра всех привилегий пользователя выполнить команду:

```
usercaps <пользователь>
```

Для просмотра Linux-привилегий пользователя выполнить команду:

```
usercaps -L <пользователь>
```

Для просмотра PARSEC-привилегий пользователя выполнить команду:

```
usercaps -M <пользователь>
```

Описание параметров `usercaps` приведено в таблице 27.

Таблица 27

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

Описание `usercaps` также приведено в `man usercaps`.

4.16.2. execaps

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

Синтаксис инструмента `execaps`:

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

Привилегии задаются в виде битовой маски (как правило, в шестнадцатеричном виде). Соответствие отдельных битов полномочиям приведено в `man parsec_capset`, а также в таблице 8.

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

Пример

Выполнить команду pscaps с привилегией 0x00008:

```
execaps -c 0x00008 pscaps 0
```

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

```
00000008 00000008 00000008
```

Описание основных параметров execaps приведено в таблице 28.

Таблица 28

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

Более подробное описание execaps приведено в man execaps.

4.16.3. pscaps

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

Синтаксис инструмента pscaps:

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

где <PID> — идентификатор процесса.

Если в качестве параметра указан только идентификатор процесса <PID>, то pscaps показывает набор PARSEC-привилегий (указанных в виде битовых масок привилегий) процесса. При указании в качестве <PID> значение 0 будет показан набор привилегий процесса pscaps.

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

Описание параметров pscaps приведено в таблице 29.

Таблица 29

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

Описание `pscaps` также приведено в `man pscaps`.

4.17. Мандатное управление доступом в СУБД PostgreSQL

В качестве защищенной СУБД в составе ОС используется СУБД PostgreSQL, 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.

ВНИМАНИЕ! ДП-модель в PostgreSQL работает только при настроенном в ОС механизме мандатного управления доступом.

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 безопасности ОС.

СУБД PostgreSQL не имеет собственного механизма назначения, хранения и модификации меток пользователей и использует для этого механизмы ОС.

Согласно ДП-модели в части реализации мандатного управления доступом дополнительно к классификационной метке вводится понятие сущностей-контейнеров (сущностей, которые могут содержать другие сущности). Для задания способа доступа к сущностям внутри контейнеров используется мандатный признак CCR (Container Clearance Required). В случае когда он установлен, доступ к контейнеру и его содержимому определяется его классификационной меткой, 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 классификационную метку сущности: метка сущности не может превышать метку контейнера, в котором она содер-

жится. Таким образом, для назначения классификационных меток данным сначала должны быть последовательно заданы максимальные классификационные метки соответствующих контейнеров: кластера, базы данных, табличного пространства, схемы и таблицы.

В реляционной модели в качестве структуры, обладающей меткой безопасности, естественно выбрать кортеж, поскольку именно на этом уровне детализации осуществляются операции чтения/записи информации в СУБД. При этом местом хранения метки безопасности может быть выбран только сам кортеж — только так метка безопасности будет неразрывно связана с данными, содержащимися в кортеже. Кроме этого, метка безопасности также может быть определена для объектов БД, к которым применимы виды доступа на чтение/запись данных, а именно: таблицы и виды. В этом случае метки безопасности объектов располагаются в записи системной таблицы, непосредственно описывающей защищаемый объект.

В качестве множества сущностей (сущностей-объектов и сущностей-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 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, либо привилегированным пользователем в специальном режиме. Таким образом, мандатное управление доступом применяется ко всем объектам БД. Метки безопасности системных объектов располагаются в записях таблиц системного каталога, непосредственно описывающих защищаемый объект.

4.17.1. Порядок применения мандатных правил управления доступом

Так как мандатное управление доступом может быть определено 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 безопасности, то он получает по умолчанию нулевую метку безопасности, соответствующую минимальному уровню доступа. Максимальная метка безопасности определяется по

заданной при регистрации пользователя в ОС. Поскольку сервер БД также может иметь метку безопасности, то если метка безопасности пользователя превышает метку безопасности сервера, то пользователю будут разрешены только операции чтения. Текущая метка безопасности пользователя определяется по установленному соединению и может быть установлена только в пределах назначенного ему 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 безопасности пользователя. Обновляемые записи сохраняют свою метку безопасности 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 30).

Таблица 30

Параметр	Описание
`ac_ignore_server_maclabel`	Определяет, будет ли сервер СУБД дополнительно использовать свою метку безопасности (метку безопасности пользователя `postgres`) при определении прав пользователя на занесение, удаление и модификацию данных или нет. Если этот параметр установлен в `FALSE`, то метка безопасности сервера используется для блокирования занесения в БД информации с меткой безопасности, превышающей метку безопасности сервера. Если этот параметр установлен в `TRUE`, то метка безопасности сервера не учитывается

Продолжение таблицы 30

Параметр	Описание
ac_ignore_socket_maclabel	Определяет, будет ли сервер СУБД использовать метку безопасности входящего соединения. Если этот параметр установлен в FALSE, то метка безопасности входящего соединения будет учитываться при определении максимальной доступной метки безопасности сессии, и после подключения будет доступна только информация с меткой безопасности, не превышающей метку безопасности входящего соединения. При установке этого параметра в TRUE метки безопасности сеанса будут определяться максимальной меткой безопасности пользователя, полученной из ОС
ac_ignore_maclabel	Обеспечивает возможность работы с базой при отключенном мандатном управлении доступом. Если данный параметр установлен в значение TRUE, то мандатные атрибуты пользователя не запрашиваются, метка безопасности сессии устанавливается нулевой вне зависимости от значения ac_ignore_socket_maclabel. Метка безопасности сервера принимается нулевой вне зависимости от значения параметра ac_ignore_server_maclabel
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 параметром 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 30

Параметр	Описание
ac_enable_sequence_ccr	При установке этого параметра в TRUE разрешается использование 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 восстановлении 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 в секундах информации 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 параметром 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 безопасности сокета соединения. Если этот параметр не указывается в конфигурационном файле pg_hba.conf или установлен в 0, то игнорирование метки безопасности сокета для этого подключения не происходит.

В ОС каждый пользователь может иметь множество меток безопасности, которое задается минимальной и максимальной метками безопасности диапазона. Чтобы поддержать эту модель в СУБД PostgreSQL каждой сессии пользователя назначаются три метки безопасности: 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 безопасности пользователя и принимает их как максимальную и минимальную метки безопасности сессии. При этом, если запись о метках безопасности для пользователя не найде-

на, то максимальная и минимальная метки безопасности 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 безопасности входящего соединения, и, если она попадает в диапазон меток безопасности, считанных из ОС, то максимальная метка безопасности сессии устанавливается равной метке безопасности входящего соединения;

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

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

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

Если пользователь имеет параметр `ac_capable_setmac`, то он может изменять свою текущую метку безопасности 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 безопасности 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 осуществляется 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 хранящиеся в них данные (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 безопасности записей объекта существуют ограничения на выполнение этих операций.

Операции удаления невозможны при наличии разных меток безопасности 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 безопасности субъекта и объекта. В случае, когда строки имеют разные метки безопасности, данное условие выполниться не может.

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

4.17.2. Средства управления мандатными ПРД к объектам БД

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

При создании объекта БД ему задается метка безопасности, равная текущей метке безопасности создавшего его пользователя, мандатный признак CCR при этом выставляется в значение ON.

Если пользователь имеет параметр ac_capable_chmac, то он может менять метку безопасности принадлежащих ему объектов в пределах своего диапазона меток безопасности с помощью следующей команды:

```
MAC LABEL ON <тип_объекта_БД> <имя_объекта_БД> IS <новая_метка_безопасности>;
```

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

Аналогичным образом для контейнеров может быть изменен мандатный признак CCR:

```
MAC CCR ON <тип_объекта_БД> <имя_объекта_БД> IS { ON | OFF };
```

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

```
MAC LABEL ON CLUSTER IS <новая_метка_безопасности>;
```

что является синонимом к команде

```
MAC LABEL ON TABLESPACE pg_global IS <новая_метка_безопасности>;
```

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

Значения мандатного признака CCR 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 метками безопасности. Для того чтобы создать таблицы с записями, защищенными метками безопасности, 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 метки безопасности создавших их пользователей. Пользователи, имеющие установленный мандатный атрибут `ac_capable_chmac`, могут явно задать значение метки безопасности вставляемой записи. Задаваемая метка безопасности должна быть в пределах диапазона меток безопасности пользователя, либо пользователь должен иметь атрибуты игнорирования мандатного управления `ac_capable_ignmaclvl` и `ac_capable_ignmaccat` с помощью варианта команды `INSERT`:

```
INSERT INTO <имя_отношения> (maclabel,...<список_столбцов>)
     VALUES (<значение_метки_безопасности>,...<значения_столбцов>)
```

Для защиты записей уже созданных таблиц без меток безопасности 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 безопасности таблицы.

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

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

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

```
CHMAC <имя_отношения> SET maclabel=<новое_значение_метки_безопасности> WHERE ...
```

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

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

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

```
SELECT maclabel FROM <имя_отношения>
```

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

Для того чтобы сохранить записи вместе с их метками безопасности в архиве и в дальнейшем загрузить их обратно, предусмотрен специальный флаг MACS команды COPY. Вывести доступные пользователю данные вместе с метками безопасности может любой пользователь, загрузить же обратно — только пользователь с установленным мандатным атрибутом ac_capabel_chmac. При этом метки безопасности загружаемых записей должны находиться в пределах диапазона меток безопасности пользователя, либо пользователь должен иметь атрибуты игнорирования мандатного управления 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 безопасности, равные текущей метке безопасности 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.

4.17.3. Целостность мандатных атрибутов кластера баз данных

В СУБД PostgreSQL ДП-модель накладывает ограничение на классификационную метку объекта: классификационная метка объекта не может превышать классификационную метку контейнера, в котором он содержится (4.17).

Для вывода информации о соблюдении ДП-модели между 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 метки безопасности 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`.

4.17.4. 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 — триггеры, 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 зависимости. 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 на 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.

4.17.5. Особенности создания правил, системы фильтрации и триггеров

В СУБД PostgreSQL правила (`RULE`), системы фильтрации (`POLICY`) 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 недоступна для пользователя в связи с мандатными ПРД. 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}`.

Системы фильтрации (`POLICY`), такие как `ROW LEVEL SECURITY`, могут дополнять встроенные механизмы разграничения доступа путем добавления логики, реализующей правила выдачи строк пользователю. Фильтрация может основываться как на данных строки, так и на внешних факторах (например, текущем времени).

4.17.6. Особенности использования представлений и материализованных представлений

Представления (`VIEW`) и материализованные представления (`MATERIALIZED VIEW`) не могут иметь метки безопасности 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, метка безопасности которого не превосходит метки безопасности представления, это представление не будет «видно» пользователю в силу мандатных ПРД. Для того, чтобы пользователь имел доступ к такому представлению, необходимо сбросить мандатный признак CCR представления.

4.17.7. 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 меток безопасности соответствующими операторами =, <>, <, <=, >, >=.

Для проверки на несравнимость меток безопасности 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.

4.17.8. Система привилегий СУБД

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 метка безопасности сессии пользователя СУБД. Эта метка безопасности определяет доступные пользователю объекты БД и является меткой безопасности по умолчанию для создаваемых пользователем объектов. При соединении пользователя с СУБД значение этого атрибута устанавливается равным метке безопасности 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 мандатное управление по уровням;
- ac_capable_ignmaccat — позволяет пользователю игнорировать мандатное управление по категориям;
- ac_capable_mac_readsearch — позволяет пользователю игнорировать мандатное управление по уровням и категориям при чтении данных;
- ac_capable_setmac — позволяет пользователю изменять текущую метку безопасности 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_безопасности>
SELECT set_config('ac_session_maclabel',<новая_метка_безопасности>,false);
```

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

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

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

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

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

В комплексе программ гипертекстовой обработки данных пользователь получает доступ к ресурсам, являющимся объектами ФС. Комплекс программ электронной почты использует технологию `maildir`, обеспечивающую хранение почтовых сообщений в виде отдельных объектов ФС. Создаваемые файлы почтовых сообщений маркируются метками безопасности, унаследованными от процесса-создателя. Таким образом, в обоих комплексах

программ ресурсы, к которым осуществляется доступ от имени серверных процессов, обрабатывающих запросы пользователей, являются объектами ФС. Следовательно, доступ к защищаемым ресурсам при приеме и обработке запросов пользователей в процессе функционирования серверов комплексов программ гипертекстовой обработки данных и электронной почты подчиняется мандатным ПРД.

4.19. Настройка загрузчика GRUB 2

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

В загрузчике GRUB 2 возможно задать параметры командной строки ядра PARSEC, приведенные в таблице 31.

Таблица 31

Параметр	Описание
`parsec.max_ilev`	Задает максимальный уровень целостности (воспринимаемый модулями) в ОС на момент запуска. Допустимые значения от 0 до 255. При установке ОС задается по умолчанию значение 63. **ВНИМАНИЕ!** Если в системе присутствуют файлы с метками безопасности, имеющими уровень целостности выше задаваемого в данный момент, это может вызвать отказ доступа к данным файлам (сокетам, каталогам и т. д.)
`parsec.ccnr_relax`	При заданном значении 1 позволяет непривилегированному пользователю выполнять запись файлов с разным уровнем конфиденциальности в контейнер (каталог) с установленным дополнительным мандатным атрибутом `ccnr`. Значение по умолчанию `0`. Параметр не применяется при включенном расширенном режиме МКЦ (см. 4.5)
`parsec.reset_ilev_on_chroot`	Сброс (обнуление) уровня целостности метки безопасности у дочерних процессов при использовании родительским процессом системных вызовов `chroot` и `pivot_root`. Текущие значение уровня целостности не влияет на применение параметра. Допустимые значения `0/1`, `y/n`. Значение по умолчанию `1` (обнуляет уровень)
`parsec.noload_files`	Запрет загрузки модулей ядра, встроенного программного обеспечения, образов `kexec`, ключей и сертификатов X.509 привилегированным пользователем (`root`, `uid=0`) с низким уровнем целостности. Допустимые значения `0/1`, `y/n`. Значение по умолчанию `1` (загрузка запрещена)

Окончание таблицы 31

Параметр	
`parsec.ccnr_reject`	Запрет установки дополнительного мандатного атрибута `ccnr` привилегированным пользователем (`root`, `uid=0`). Допустимые значения `0/1`, `y/n`. Значение по умолчанию `0` (установка разрешена)
`parsec.enable_exec_on_fuse`	Разрешение запуска сценариев и исполняемых файлов с файловых систем, смонтированных с помощью файловой системы FUSE. Допустимые значения `0/1`. Значение по умолчанию `0` (запуск запрещен)

5. ЗАЩИТА СРЕДЫ ВИРТУАЛИЗАЦИИ

ОС разработана с учетом применения встроенных средств защиты в виртуальной инфраструктуре и включает в свой состав ядро с поддержкой технологии виртуализации[1),2)] Kernel-based Virtual Machine (KVM, гипервизор II-типа). Ядро ОС поддерживает технологию KVM (Kernel-based Virtual Machine), обеспечивающую создание и функционирование виртуальной инфраструктуры. Технология KVM включает в себя специальный модуль ядра KVM и средство создания виртуального программно-аппаратного окружения QEMU.

KVM использует технологию аппаратной виртуализации, поддерживаемую процессорами от Intel и AMD и известную под названиями Intel-VT и AMD-V. Используя загруженный в память модуль ядра, KVM с помощью драйвера пользовательского режима (который представляет собой модифицированный драйвер от QEMU) эмулирует слой аппаратного обеспечения, в среде которого могут создаваться и запускаться виртуальные машины.

В архитектуре KVM виртуальная машина выполняется как обычный процесс ОС, на который распространяется действие мер по идентификации и аутентификации субъектов и объектов доступа в полном объеме возможностей функций безопасности ОС.

Управление средой виртуализации обеспечивается утилитой `virsh` с использованием программного интерфейса `libvirt`.

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

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

Основными средствами, необходимыми для создания среды виртуализации, являются:

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

[1)] Для процессоров, поддерживающих технологию виртуализации.

[2)] Недоступно в режиме «Мобильный».

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

5.1. Дискреционное управление доступом в среде виртуализации

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

Дискреционное управление доступом при работе с сервером виртуализации `libvirt` осуществляется совместным использованием модуля дискреционного управления доступом `dac` и специально разработанного модуля мандатного управления доступом `parsec`, взаимодействующего с подсистемой безопасности PARSEC ОС. Модуль поддержки дискреционного управления доступом `dac` использует дискреционные атрибуты, состоящие из уникальных идентификаторов. Основанием для принятия решения о предоставлении доступа является сравнение дискреционных атрибутов ВМ и дискреционных атрибутов пользователя с учетом выполняемой операции, а также с учетом роли пользователя.

Дискреционные атрибуты ВМ могут быть динамическими и статическими:

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

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

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

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

Непривилегированные действия доступны пользователям группы `libvirt`.

5.2. Мандатное управление доступом к виртуальной машине

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

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

Метка безопасности ВМ может быть динамической или статической:

- динамическая генерируется в момент запуска ВМ на основе метки безопасности запускающего пользователя;
- статическая задается администратором в конфигурации ВМ и определяет контекст безопасности ее запуска.

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

Мандатное управление доступом осуществляется по следующим правилам:

- создаваемая ВМ получает метку безопасности создающего ее пользователя;
- образ ВМ не обладает меткой безопасности (если для него не задана статическая метка безопасности);
- запущенная ВМ наследует метку безопасности запускающего пользователя (это отражается в виде наличия динамических меток безопасности как у процесса ВМ в ОС, так и у файлов-образов и устройств, принадлежащих ВМ);
- доступ к функционирующей ВМ предоставляется только при равенстве меток безопасности пользователя и ВМ;
- доступ к получению информации о ВМ и ее конфигурации предоставляется в соответствии с правилами мандатного управления доступом с учетом меток безопасности пользователя и ВМ.

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 виртуальная машина не должна запускаться в мандатном контексте. Вместо этого необходимо выполнять

удаленный вход с требуемым уровнем конфиденциальности доступа средствами ОС.

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

5.3. Ролевое управление доступом в среде виртуализации

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

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

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

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

5.3.1. Настройка ролей

Реализация ролевого метода управления доступом подразумевает разграничение доступа по ролям:

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

Назначение ролей и полномочий осуществляется администратором системы с высоким уровнем целостности.

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

Роль администратора средства виртуализации позволяет создавать и управлять учетными записями пользователей средств виртуализации, назначать права доступа пользователям средства виртуализации к виртуальным машинам, создавать и удалять виртуальное оборудование, изменять конфигурации виртуального оборудования, управлять доступом виртуальных машин к физическому и виртуальному оборудованию, управлять квотами

доступа виртуальных машин к физическому и виртуальному оборудованию, управлять перемещением виртуальных машин, удалять виртуальные машины, запускать и останавливать виртуальные машины, создавать снимки состояния виртуальных машин (включающих файлы конфигурации виртуальной машины, образа виртуальной машины и образа памяти виртуальной машины). Пользователю с данной ролью необходимо назначить возможность работы под высоким уровнем целостности (63), а также:

1) при использовании драйвера `polkit`:

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

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

- открыть файл:

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

- добавить строку:

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

2) при использовании драйвера `parsec`:

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

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

- открыть файл:

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

- добавить строку:

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

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

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

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

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

1) перейти в свойства ВМ;
2) выбрать раздел «Безопасность»;
3) во вкладке «Подробности» в таблице «Дискреционный контроль доступа» для пользователя установить разрешение «Использование» в соответствии с рис. 1.

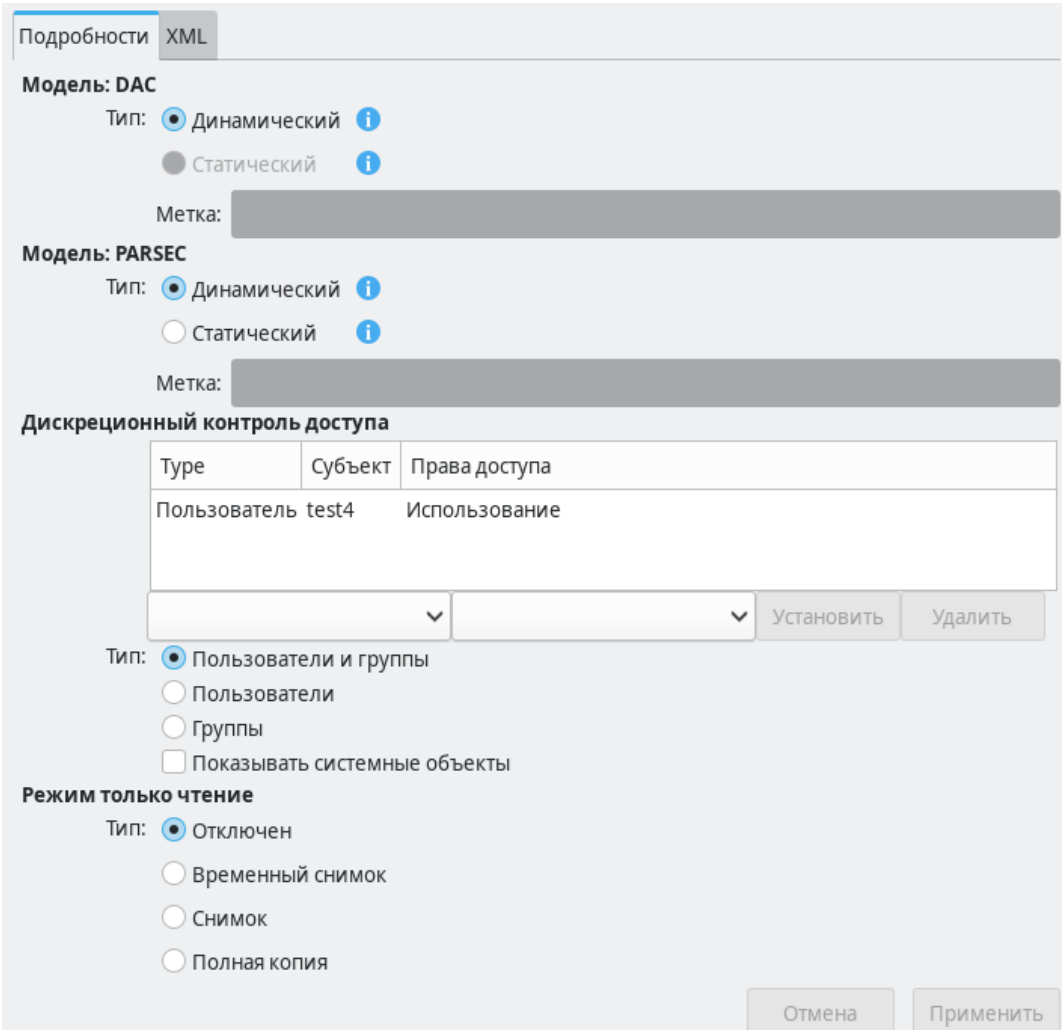

Рис. 1

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

5.3.2. Настройка ролевого управления доступом с использованием драйвера доступа polkit

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

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

Затем перезагрузить службу libvirtd, выполнив команды:

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

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

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

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

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

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

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

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

5.3.3. Настройка ролевого управления доступом с использованием драйвера доступа parsec

Для включения ролевого управления доступом на основе драйвера доступа parsec:

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

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

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

```
enable = 1
```

После выполненных действий необходимо перезагрузить сервис `libvirtd`:

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

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

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

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

Пример

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

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

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

5.4. Режим «только чтение»: запрет модификации образа виртуальной машины

Поддержка функционирования виртуальной машины в режиме запрета модификации ее образа осуществляется специальными способами запуска виртуальной машины, при которых основной образ защищается от записи. В зависимости от выбранного режима

осуществляется создание физической копии или применяются различные варианты создания снимков образа с последующим их удалением после завершения работы виртуальной машины.

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

- временный снимок — способ запуска виртуальной машины, при котором основной образ защищается от записи, а все результаты работы пользователя фиксируются во временном снимке, существующем только в процессе функционирования виртуальной машины;
- 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 образа, удаляемая после завершения функционирования виртуальной машины.

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

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

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

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

5.5. Идентификация и аутентификация пользователей в среде виртуализации

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

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

Вход в систему осуществляется по идентификатору и паролю. При входе в систему осуществляется идентификация и проверка подлинности субъектов доступа процедурой аутентификации. По умолчанию в системе отключен режим автоматического входа по

сохраненным учетным данным или без пароля. Настройка входа в систему осуществляется с использованием утилиты `fly-admin-dm`.

Установка пароля пользователя осуществляется администратором с помощью инструментов управления политикой безопасности (утилита `fly-admin-smc`).

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

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

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

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

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

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

Описание перечисленных выше процедур приведено в разделе 2.

Для аутентификации в условиях ЕПП и при удаленном доступе к серверу виртуализации используются методы SSH-, SASL- или SSL/TLS-аутентификации в соответствии с РУСБ.10015-37 95 01-1.

5.6. Доверенная загрузка виртуальных машин

Доверенная загрузка виртуальных машин обеспечивается с использованием средств динамического контроля целостности (создание замкнутой программной среды) в режиме контроля целостности файлов при их открытии на основе ЭЦП в расширенных атрибутах файловой системы. При выявлении нарушения целостности конфигурации виртуального оборудования или целостности файлов виртуальной базовой системы ввода-вывода сред-

ствами ОС осуществляется блокировка запуска виртуальных машин. Описание применения замкнутой программной среды (ЗПС) приведено в 16.1.

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

5.6.1. Применение режима контроля целостности файлов при их открытии на основе ЭЦП

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

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` (см. электронную справку) или путем редактирования конфигурационного файла `/etc/digsig/digsig_initramfs.conf`.

Для включения проверки подписей в режиме запрета открытия поставленных на контроль файлов с неверной ЭЦП или без ЭЦП в расширенных атрибутах необходимо установить для параметра `DIGSIG_XATTR_MODE` значение `1`:

```
DIGSIG_XATTR_MODE=1
```

Для возможности осуществления подписи файлов необходимо создать ключевую пару с помощью утилиты `gpg` (более подробное описание настройки модуля `digsig_verif` и подписания файлов приведено в 16.1):

1) сгенерировать ключ:

```
gpg --full-generate-key
```

В выводе команды сохранить `uid` ключа (вида `TestTest <test@astralinux.ru>`);

2) выполнить экспорт ключа в каталог `/etc/digsig/xattr_keys/` с указанием его `uid` ключа:

```
gpg --export "TestTest <test@astralinux.ru>" >
     /etc/digsig/xattr_keys/secondary_gost_key.gpg"
```

ВНИМАНИЕ! После внесения изменений в конфигурационный файл `/etc/digsig/digsig_initramfs.conf` и для загрузки модулем `digsig_verif` ключей после их размещения 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
```

5.6.1.1. Контроль файлов конфигурации виртуального оборудования виртуальных машин

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

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

Пример

Для внедрения ЭЦП в файл конфигурации виртуального оборудования виртуальной машины alse необходимо выполнить команду:

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

Проверить наличие подписи можно командой:

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

или командой:

```
bsign -v /etc/libvirt/qemu/alse.xml
```

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

Пример

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

```
echo '/etc/libvirt/qemu/alse.xml' >> /etc/digsig/xattr_control
```

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

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

Внести имя контролируемого файла в /etc/digsig/xattr_control возможно и с использованием графической утилиты управления политикой безопасности fly-admin-smc (см. электронную справку), добавив в поле «Шаблоны имен файлов для» в соответствии с рис. 2.

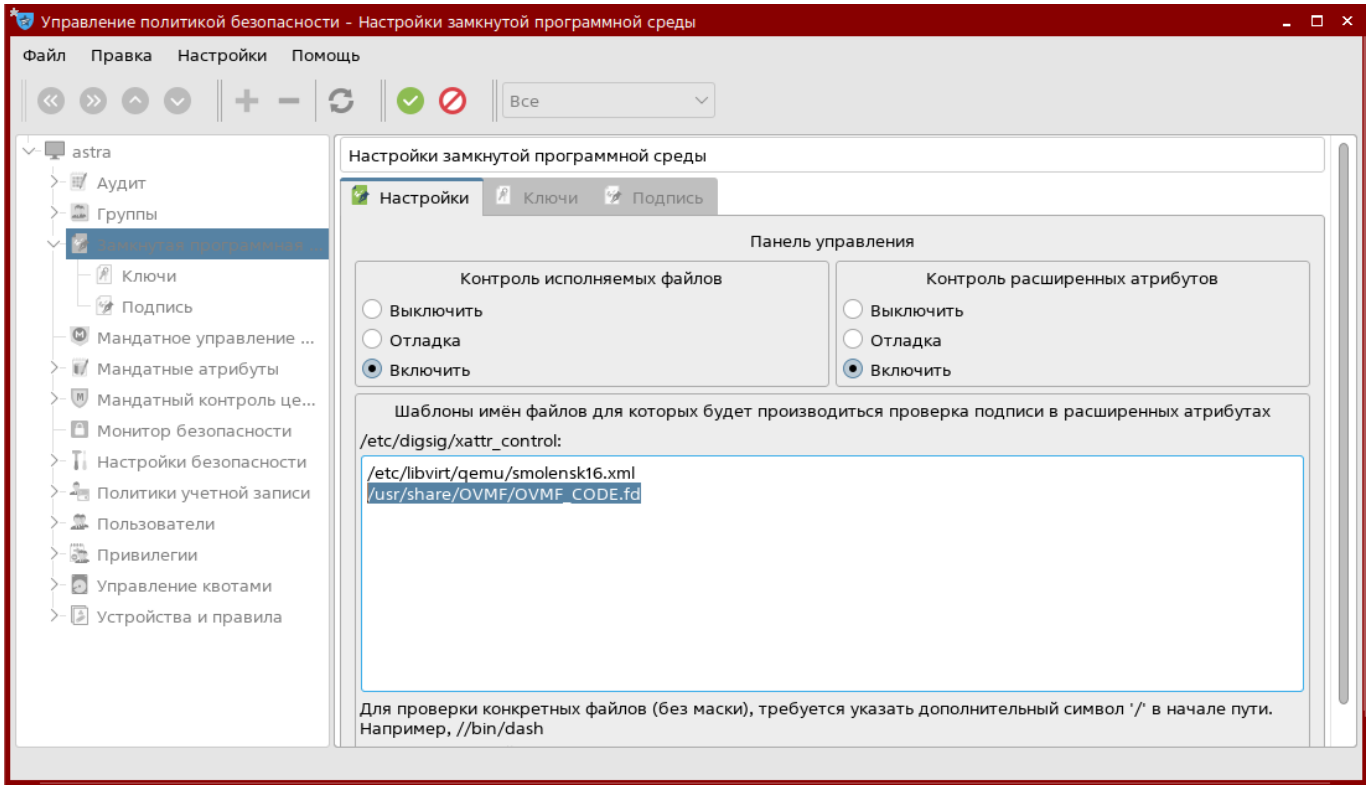

Рис. 2

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

В ОС входит пакет `ovmf` (должен быть установлен пакет `astra-kvm-secure`), который предоставляет поддержку загрузчика UEFI и позволяет использовать в виртуальной машине загрузчик UEFI вместо традиционного BIOS. Традиционный BIOS встроен в QEMU, исполняемый файл `qemu-system-x86_64` которого по умолчанию подписан для контроля запуска в ЗПС (при этом должен быть включен режим запрета запуска исполняемых файлов и разделяемых библиотек с неверной ЭЦП, а также без ЭЦП — значение параметра `DIGSIG_ELF_MODE=1`, см. 16.1).

Для использования в виртуальной машине UEFI вместо традиционного BIOS необходимо в процессе создания виртуальной машины с использованием `virt-manager` установить флаг «Проверить конфигурацию перед установкой» и нажать **[Готово]** в соответствии с рис. 3.

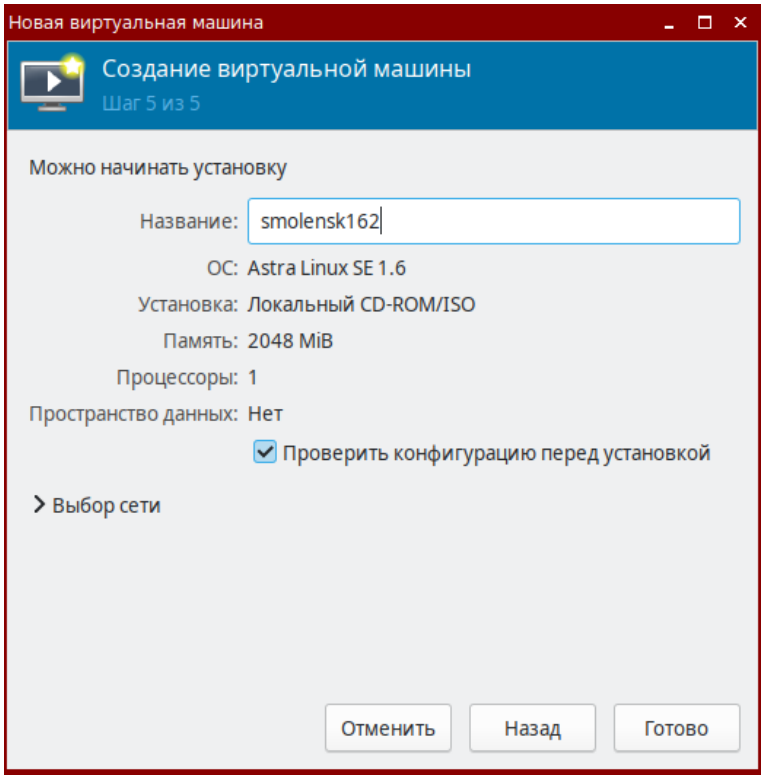

Рис. 3

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

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

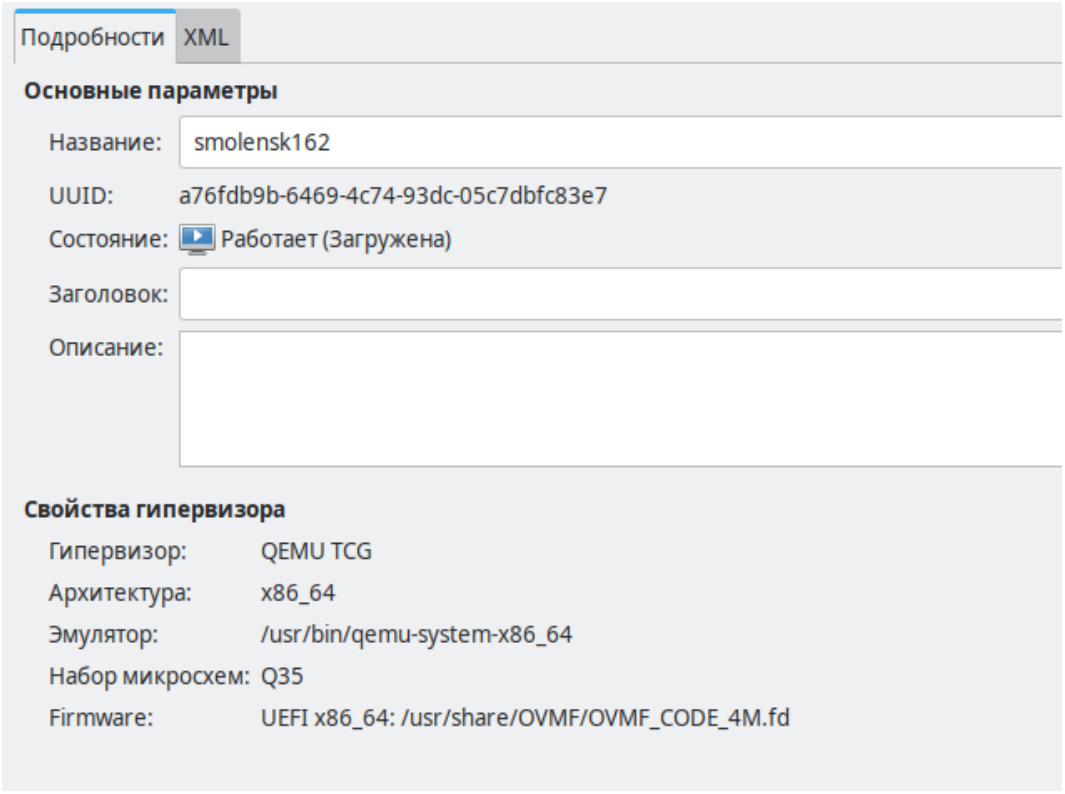

Рис. 4

Расположенные в каталоге `/usr/share/OVMF` файлы с расширением `.fd` должны быть установлены администратором на контроль целостности.

Для внедрения ЭЦП в расширенные атрибуты необходимо подписать на созданном вторичном ключе контролируемые файлы с использованием утилиты `bsign`.

Пример

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

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

Проверить наличие подписи командой:

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

Далее выполнить настройку записей имен файлов, на которые распространяется проверка ЭЦП в расширенных атрибутах ФС. Для этого можно с использованием графической утилиты управления политикой безопасности `fly-admin-smc` (см. электронную справку) добавить в поле «Шаблоны имен файлов для....» путь к контролируемому файлу загрузчика в соответствии с рис. 2.

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

Применение механизма контроля целостности обеспечивает динамическое создание хеша с использованием алгоритма подсчета контрольных сумм по стандарту

ГОСТ Р 34.11-2012 для каждого легитимно загружаемого файла конфигурации и сохранение их в каталоге с высокой целостностью `/var/lib/libvirt/hash`.

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

Механизм контролирует следующий перечень конфигураций виртуальной инфраструктуры:

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

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

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

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

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

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

```
integrity_control = 1
```

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

Для применения настроек следует перезагрузить службу `libvirtd`:

```
sudo systemctl restart libvirtd
```

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

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

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

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

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

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

```
virsh edit <VM>
```

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

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

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

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

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

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

5.6.3. Применение механизма контроля целостности файлов гостевой операционной системы

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

Подсчет контрольных сумм может осуществляться как в в процессе загрузки, так и периодически в процессе работы гостевой операционной системы. Эталонные значения контрольных сумм хранятся в конфигурационных файлах виртуальных машин, установленных на контроль, что позволяет применять механизм в условиях миграции ВМ. При выявлении нарушения целостности файлов гостевой операционной системы обеспечивается блокировка ее запуска и регистрация фактов нарушения целостности объектов контроля в журнал подсистемы регистрации событий (см. 6.5) и в журнал `/var/log/syslog`.

Настройка механизма контроля целостности файлов гостевой операционной системы осуществляется в конфигурационном файле `/etc/libvirt/libvirtd.conf`. Настройку рекомендуется выполнять с использованием `virsh` путем выполнения команды:

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

Для включения механизма контроля целостности файлов гостевой операционной системы в процессе ее функционирования необходимо в файле `/etc/libvirt/libvirtd.conf` для параметра `file_integrity_check_period_s` задать значение периода проверки (в секундах):

```
file_integrity_check_period_s = 30
```

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

Для принудительного выключения функционирующей ВМ в случае нарушения целостности установленных на контроль файлов необходимо в файле `/etc/libvirt/libvirtd.conf` для параметра `file_integrity_check_shutdown_domain` задать значение `1`:

```
file_integrity_check_shutdown_domain = 1
```

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

Для применения настроек следует перезагрузить службу libvirtd:

```
sudo systemctl restart libvirtd
```

Для включения механизма контроля целостности файлов гостевой операционной системы в процессе ее загрузки необходимо в файле /etc/libvirt/libvirtd.conf для параметра file_integrity_check_shutdown_domain задать значение 1:

```
file_integrity_on_startup_VM = 1
```

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

Для применения настроек следует перезагрузить службу libvirtd.

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

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

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

- полный путь к контролируемому файлу гостевой операционной системы;
- точку монтирования;
- наименование диска хранения контролируемого файла гостевой операционной системы.

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

Пример

Вывод команды lsblk:

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

Файловая система расположена на диске vda1, соответствующая точка монтирования указана в столбце MOUNTPOINT — «/».

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

Для установки на контроль целостности объектов контроля используется команда:

```
virsh file-integrity <domain> --path-add <путь_к_файлу> --mount-point
<точка_монтирования> --disk-name <имя_диска>
```

где <domain> — наименование ВМ;

<путь_к_файлу> — полный путь к файлу, подлежащему контролю;

<точка_монтирования> — точка монтирования раздела, в котором хранится файл;

имя_диска — наименование диска, на котором хранится файл в гостевой ОС.

Пример

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

Установку на контроль целостности файлов гостевой операционной системы рекомендуется выполнять при выключенной ВМ.

После установки на контроль файлов они будут указаны в виде записей блока <fileintegrity> конфигурационного файла ВМ.

Пример

```
<fileintegrity check="-1">
<item mount_point="/" disk_name="vda1" alg="gost512" path="/bin/afick">
</fileintegrity>
```

Параметр check может принимать следующие значения, определяющие состояние проверок целостности файлов гостевой операционной системы:

1) значение -1 — первичное состояние проверок механизмом, когда в гостевой операционной системе еще не определено наличие гостевого агента qemu-guest-agent. После загрузки ВМ и успешно выполненной проверки наличия гостевого агента значение -1 автоматически меняется на 1;

2) значение 1 — контроль целостности файлов гостевой операционной системы для данной ВМ выполняется;

3) значение 0 — контроль целостности файлов гостевой операционной системы для данной ВМ выключен.

После загрузки гостевой операционной системы и успешно выполненных проверок поиска агента qemu-guest-agent и указанных файлов в записях, ассоциированных с контролируемыми объектами, формируются значения эталонных контрольных сумм.

Пример

```
<fileintegrity check="1">
<item mount_point="/" disk_name="vda1" alg="gost512" path="/bin/afick"
hash="67f6...">
</fileintegrity>
```

Примечание. Для <item> возможно автоматическое добавление параметров disk_num (порядковый номер диска в конфигурации ВМ), part_num (порядковый номер раздела на диске) и alg (длина ключа алгоритма подсчета контрольных сумм). Значения для данных параметров формируются автоматически и необходимы для корректной работы механизма.

В случае отсутствия в гостевой операционной системе ВМ установленного агента qemu-guest-agent контроль целостности файлов будет осуществляться только в процессе загрузки ВМ.

Перерасчет контрольных сумм в случае нарушения контроля целостности объектов контроля выполняется с использованием команды:

```
virsh file-integrity <domain> --update-hash <путь_к_файлу>
```

Перерасчет контрольных сумм рекомендуется выполнять при выключенной ВМ.

Выключение контроля целостности файлов гостевой операционной системы выполняется командой:

```
virsh file-integrity <domain> --stop
```

Более подробная информация об использовании команды virsh file-integrity приведена на странице помощи:

```
virsh file-integrity --help
```

5.7. Контроль целостности в среде виртуализации

Для осуществления контроля целостности в среде виртуализации используются следующие механизмы:

1) механизм контроля запуска исполняемых файлов формата ELF и контроля расширенных атрибутов в ЗПС в соответствии с 5.7.1;

2) механизм регламентного контроля целостности Another File Integrity Checker (AFICK) в соответствии с 5.7.2;

3) механизм контроля целостности «отпечаток конфигурации» в соответствии с 5.6.2;

4) механизм контроля целостности исполняемых файлов гостевой операционной системы в соответствии с 5.6.3;

5) механизм контроля целостности областей памяти ВМ (по запросу из гостевой ОС) (п.5.12)

Контроль целостности осуществляется для следующих типов объектов: конфигурации виртуального оборудования виртуальных машин, конфигураций объектов виртуальной инфраструктуры, исполняемых файлов и параметров настройки средств виртуализации, файлов виртуальной базовой системы ввода-вывода (первичного загрузчика виртуальной машины), исполняемых файлов гостевой операционной системы виртуальной машины, областей памяти виртуальной машины.

Регистрация событий безопасности, связанных с целостностью средств виртуализации и виртуальных машин, осуществляется подсистемой регистрации событий в соответствии с 6.5.

5.7.1. Применение динамического контроля целостности в режиме ЗПС

Инструменты ЗПС предоставляют возможность внедрения ЭЦП в исполняемые файлы формата ELF и в расширенные атрибуты файловой системы, обеспечивая таким образом контроль целостности как в процессе загрузки средства виртуализации, так и динамически в процессе функционирования.

5.7.1.1. Режим контроля неизменности и подлинности загружаемых исполняемых файлов формата ELF

Для обеспечения контроля целостности исполняемых файлов средства виртуализации должен быть включен режим запрета запуска исполняемых файлов и разделяемых библиотек с неверной ЭЦП или без ЭЦП в соответствии с 16.1.

Настройка режима осуществляется с помощью графической утилиты `fly-admin-smc` (см. электронную справку) или путем редактирования конфигурационного файла `/etc/digsig/digsig_initramfs.conf`.

Для включения проверки подписей в режиме запрета загрузки на исполнение файлов и разделяемых библиотек с неверной ЭЦП, а также без ЭЦП, необходимо установить для параметра `DIGSIG_ELF_MODE` значение `1`:

```
DIGSIG_ELF_MODE=1
```

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

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

Исполняемые файлы средства виртуализации подписаны по умолчанию для контроля запуска в ЗПС.

ВНИМАНИЕ! Для контроля целостности исполняемых файлов гостевой операционной системы рекомендуется в качестве гостевой операционной системы использовать ОС с включенным режимом запрета загрузки на исполнение файлов и разделяемых библиотек с неверной ЭЦП, а также без ЭЦП.

5.7.1.2. Режим контроля целостности файлов при их открытии на основе ЭЦП

Для обеспечения динамического контроля целостности применяется режим запрета открытия файлов, поставленных на контроль, с неверной ЭЦП или без ЭЦП в расширенных атрибутах в соответствии с 5.6.1.

Контролю подлежат следующие файлы:

1) конфигурационные файлы libvirt с расширением `.conf`, расположенные в каталоге `/etc/libvirt/`;

2) конфигурационные файлы виртуальных машин в формате XML, расположенные в каталоге `/etc/libvirt/qemu/`;

3) файлы виртуальной базовой системы ввода-вывода с расширением `.fd`, расположенные в каталоге `/usr/share/OVMF/`.

5.7.2. Применение регламентного контроля целостности AFICK

Организация регламентного контроля целостности объектов контроля обеспечивается программным средством AFICK путем вычисления контрольных сумм файлов и соответствующих им атрибутов расширенной подсистемы безопасности PARSEC (мандатных атрибутов и атрибутов расширенной подсистемы протоколирования) с последующим сравнением вычисленных значений с эталонными в соответствии с 9.4.

Возможность проведения периодического контроля осуществляется с использованием системного планировщика заданий `cron`. Системный планировщик заданий `cron` также позволяет выполнять проверку целостности объектов контроля в процессе загрузки ОС.

Для контроля целостности необходимо в конфигурационном файле `/etc/afick.conf` задать список файлов и каталогов, которые будет контролировать AFICK.

Контролю подлежат следующие файлы:

1) конфигурационные файлы libvirt с расширением `.conf`, расположенные в каталоге `/etc/libvirt/`;

2) конфигурационные файлы виртуальных машин в формате XML, расположенные в каталоге `/etc/libvirt/qemu/`;

3) файлы виртуальной базовой системы ввода-вывода с расширением `.fd`, расположенные в каталоге `/usr/share/OVMF/`.

5.8. Регистрация событий безопасности в среде виртуализации

5.8.1. Настройка регистрации событий безопасности, связанных с функционированием средства виртуализации

В среде виртуализации обеспечивается регистрация событий безопасности, связанных с функционированием средств виртуализации, и оповещение администратора безопас-

ности средства виртуализации о событиях безопасности. Состав регистрируемой информации соответствует ГОСТ Р 59548-2022.

Регистрация событий безопасности, настройка реагирования системы на события и информирование администратора осуществляется подсистемой регистрации событий из состава ОС. Описание подсистемы регистрации событий и журнала событий приведено в 6.5.

Также регистрацию событий осуществляет служба `libvirtd`. Запись событий осуществляется в журнал подсистемы регистрации событий и в системный журнал `/var/log/syslog`.

Для каждой записи в журнале регистрируются номер (уникальный идентификатор), дата, время и тип события безопасности.

Просмотр системного журнала осуществляется администратором с использованием инструментов командной строки (`ausearch`, `aureport`, `aulast` и `auvirt`) и графической утилиты `ksystemlog`.

Инструмент `auvirt` используется для поиска в журнале аудита записей, созданных libvirt, чтобы вывести список сеансов ВМ. Также он выполняет поиск событий, связанных с гостевыми системами (остановка хостовой системы, отказ в доступе), и событий, связанных с QEMU-процессами.

5.8.2. Механизм централизованного сбора журналов с удаленных хостов виртуализации

Для решения задач централизованного сбора журналов с удаленных хостов виртуализации возможно применение встроенных механизмов подсистемы регистрации событий (см. 6.5).

Служба `syslog-ng` из состава подсистемы регистрации событий предоставляет возможность централизованного сбора событий безопасности, связанных с функционированием средства виртуализации, из журналов событий безопасности.

При установке пакета `astra-kvm-secure` создаются конфигурационные файлы для удаленного сбора данных о событиях безопасности.

Для настройки отправки событий с удаленного хоста необходимо в конфигурационном файле `/etc/syslog-ng/conf.d/mod-astra-libvirtd-remote.conf` в соответствующей строке установить значение IP-адреса сервера сбора журналов (вместо указанного по умолчанию `"127.0.0.1"`) и раскомментировать параметр `destination(libvirtd_remote_dst)`:

```
destination libvirtd_remote_dst {
network(
"127.0.0.1" #ip for remote host
```

```
port(2222) # for example
transport(tcp) # or udp
);
};

log {
source(s_src);
filter(libvirtd_syslog_filter);
destination(libvirtd_remote_dst);
};
```

Для осуществления возможности приема пакетов с удаленных хостов необходимо в конфигурационном файле сервера сбора журналов `/etc/syslog-ng/conf.d/mod-astra-libvirtd-remote-server.conf` раскомментировать параметр `destination(astra_json_dst)`:

```
source libvirtd_remote_src {
network(
port(2222) # for example
transport(tcp) # or udp
);
};

log {
source(libvirtd_remote_src);
filter(libvirtd_syslog_filter);
parser(libvirtd_syslog_parser);
parser(astra_libvirtd_parser);
destination(astra_json_dst);
};
```

5.9. Резервное копирование в среде виртуализации

Резервное копирование образов виртуальных машин и конфигурации виртуального оборудования виртуальных машин, а также параметров настройки средств виртуализации и сведений о событиях безопасности реализуется с использованием встроенных в средства виртуализации ОС механизмов резервного копирования (инструментов командной строки `virsh backup-begin`, `virsh dumpxml` и `virsh snapshot-create`), а также встроенных в ОС средств резервного копирования.

ПО резервного копирования и восстановления из состава ОС включает инструменты командной строки и распределенные системы управления хранилищами данных:

1) комплекс программ Bacula;
2) инструмент копирования `rsync`;
3) инструменты архивирования и копирования `tar`, `cpio`, `gzip`, `cp`.

Описание указанных инструментов приведено в РУСБ.10015-37 95 01-1.

5.9.1. Резервное копирование образов виртуальных машин

Полное резервное копирование текущего состояния образа виртуальной машины выполняется с использованием инструмента `virsh backup-begin`, при этом ВМ должна быть включена:

```
virsh backup-begin <domain-id|domain-name|domain-uuid>
```

После выполнения резервного копирования копии образов сохраняются в каталог `/var/lib/libvirt/qemu/images/`.

Пример

Для выполнения резервного копирования конфигурации ВМ `alse` необходимо от имени администратора средства виртуализации запустить ВМ `alse`:

```
virsh -c qemu:///system list --all
virsh -c qemu:///system start alse
```

Осуществить резервное копирование образа ВМ `alse`:

```
virsh -c qemu:///system backup-begin alse
```

Для проверки статуса задания, запущенного в отношении ВМ `alse`, можно выполнить команды с параметром `domjobinfo`:

```
virsh -c qemu:///system domjobinfo alse
virsh -c qemu:///system domjobinfo alse --completed
```

Проверить наличие созданной резервной копии командой:

```
sudo ls -lash /var/lib/libvirt/images/
```

5.9.2. Резервное копирование конфигурации виртуального оборудования виртуальных машин

Выполнение резервного копирования конфигурации виртуального оборудования созданных виртуальных машин (существующего XML-файла ВМ) осуществляется с помощью инструмента `virsh dumpxml`:

```
virsh dumpxml <domain-id|domain-name|domain-uuid> <имя_копии>
```

Пример

Для выполнения резервного копирования конфигурации ВМ `alse` необходимо от имени администратора средства виртуализации запустить ВМ `alse` и выполнить команду резервного копирования конфигурации:

```
virsh -c qemu:///system list --all
```

```
virsh -c qemu:///system start alse
virsh -c qemu:///system dumpxml alse > alse_backup.xml
```

5.9.3. Резервное копирование параметров настройки средства виртуализации

Выполнение резервного копирования параметров настройки средства виртуализации осуществляется с использованием инструментов архивирования и копирования из состава ОС: `tar`, `cpio`, `gzip`, `cp`.

Пример

Для выполнения резервного копирования конфигурационных файлов в каталоге `/etc/libvirt/` выполнить команды:

```
cd /etc/libvirt
tar --xattrs --acls -czv conf_backup.tar.gz /etc/libvirt/libvirtd.conf
/etc/libvirt/libvirt.conf /etc/libvirt/libvirt-admin.conf
```

5.9.4. Создание снимков текущего состояния машины

Управление снимками ВМ из командной строки выполняется с использованием инструмента `virsh snapshot-create`:

1) создание снимка (ОЗУ и диск) из файла XML:

```
virsh snapshot-create <domain> [--xmlfile <строка>] [--disk-only]
[-- live]...
```

2) создание снимка (ОЗУ и диск) путем задания набора параметров:

```
virsh snapshot-create-as <domain> [--name <строка>] [--disk-only]
[-- live]...
```

Созданные снимки ВМ сохраняются в каталог `/var/lib/libvirt/qemu/snapshot/`.

Примеры:

1. Для просмотра существующих снимков домена `alse` выполнить команду:

```
virsh snapshot-list --domain alse
```

2. Для восстановления ВМ из снимка выполнить команду:

```
virsh snapshot-revert --domain alse --snapshotname snapshot_alse
```

Для управления снимками ВМ в утилите `virt-manager` необходимо:

1) в главном окне утилиты выбрать ВМ;

2) нажать **[Открыть]**;

3) в открывшемся окне нажать **[Управление снимками]**.

5.9.5. Резервное копирование и полная очистка журналов

Резервное копирование сведений о событиях безопасности реализуется с использованием встроенных в ОС средств ротации и архивирования журналов `logrotate`. На-

стройка ротации возможна с использованием утилиты `fly-admin-events` («Настройка регистрации системных событий»), описание утилиты приведено в электронной справке.

Запустить ротацию журнала возможно вручную, без ожидания установленного в `cron` правила, для этого выполнить команду:

```
sudo logrotate /etc/logrotate.d/syslog-ng-mod-astra
```

5.10. Управление потоками информации в среде виртуализации

Управление потоками информации между виртуальными машинами и информационными (автоматизированными) системами на канальном и сетевом уровнях реализуется штатными средствами ОС, реализующими технологию виртуальных локальных сетей (VLAN) стандарта IEEE 802.1q.

Для управления потоками информации в виртуальной инфраструктуре применяются следующие механизмы, обеспечивающие сетевую фильтрацию:

1) драйвер виртуальных сетей libvirt — предоставляет изолированное мостовое устройство (без подключения физических сетевых карт), к которому подключены гостевые TAP-устройства. Для ВМ разрешается обмен сетевым трафиком между собой и с хостом. Для виртуальной сети возможны три конфигурации:

- «Изолированный» (isolated) — весь внешний трафик полностью блокируется;
- NAT — исходящий трафик во внешнюю сеть разрешен, но замаскирован;
- «Маршрутизация» (forward) — исходящий трафик в локальную сеть разрешен;

2) драйвер сетевых фильтров libvirt — обеспечивает полностью настраиваемую сетевую фильтрацию трафика на гостевых сетевых картах с использованием сетевых фильтров `nwfilter`. Наборы правил для управления трафиком определяются на уровне хоста. Затем наборы правил связываются с определенными гостевыми сетевыми картами;

3) изоляция сетей с помощью VLAN — программный многоуровневый коммутатор Open vSwitch (OVS) для виртуальных сетей обеспечивает изоляцию сети с помощью VLAN путем тегирования портов, а также фильтрацию сетевого трафика. Описание настройки и работы OVS приведено в РУСБ.10015-37 95 01-1.

5.10.1. Управление сетевыми фильтрами nwfilter

Контроль взаимодействия виртуальных машин между собой реализуется с использованием сетевых фильтров `nwfilter`. Сетевые фильтры предоставляют возможность настраивать и применять правила фильтрации сетевого трафика на виртуальных машинах, а также управлять параметрами входящего и исходящего сетевого трафика. Поскольку правила фильтрации нельзя обойти внутри виртуальной машины, это делает их обязательными при использовании виртуальной машины.

Управление фильтрами выполняется как автономными объектами верхнего уровня и осуществляется администратором средства виртуализации с использованием интерфейсов управления средствами виртуализации libvirt. В сетевых фильтрах могут быть настроены правила фильтрации сетевого трафика ВМ отдельно для каждого сетевого интерфейса через конфигурационный XML-файл виртуальных машин. Правила применяются на хосте при запуске виртуальной машины и могут быть изменены во время работы виртуальной машины (путем изменения XML-описания сетевого фильтра).

Несколько виртуальных машин могут использовать один и тот же сетевой фильтр. При изменении такого фильтра обновляются правила фильтрации сетевого трафика всех запущенных виртуальных машин, в которых применяется данный сетевой фильтр.

Также возможно настраивать правила фильтрации сетевого трафика на отдельных сетевых интерфейсах, настроенных для определенных типов сетевых конфигураций виртуальных машин (network, ethernet в режиме моста, bridge).

Сетевые фильтры задаются в формате XML и располагаются в каталоге `/etc/libvirt/nwfilter/`.

Пример

```
filter name='no-spamming' chain='XXXX'>
<uuid>d217f2d7-5a04-0e01-8b98-ec2743436b74</uuid>
<rule ...>
....
</rule>
<filterref filter='XXXX'/>
</filter>
```

Каждый фильтр имеет имя и UUID, которые служат уникальными идентификаторами. Фильтр может содержать `<rule>`, которые используются для фактического определения сетевых элементов управления. Элемент `<rule>` имеет параметры `action` (действие), `direction` (направление) и необязательный `priority` (приоритет).

Пример

```
<rule action='drop' direction='out' priority='500'>
```

В правилах используются элементы сопоставления с конкретным протоколом. Поддерживаемые протоколы: `mac`, `arp`, `rarp`, `ip`, `ipv6`, `tcp/ip`, `icmp/ip`, `igmp/ip`, `udp/ip`, `udplite/ip`, `esp/ip`, `ah/ip`, `sctp/ip`, `tcp/ipv6`, `icmp/ipv6`, `igmp/ipv6`, `udp/ipv6`, `udplite/ipv6`, `esp/ipv6`, `ah/ipv6`, `sctp/ipv6`. Каждый протокол определяет, что допустимо указывать внутри элемента `<rule>`. Запись имеет вид:

```
<protocol match='yes|no' attribute1='value1' attribute2='value2'/>
```

Пример

Запись для протокола TCP с портами 0-1023:

```
<tcp match='yes' srcportstart='0' srcportend='1023'/>
```

Набор правил по умолчанию можно посмотреть, выполнив команду:

```
virsh nwfilter-list
```

Перечень фильтров, автоматически устанавливаемых вместе с libvirt, приведен в таблице 32.

Таблица 32

Фильтр	Описание
`no-arp-spoofing`	Запретить виртуальной машине подделку ARP-трафика. Фильтр разрешает только сообщения запросов и ответов ARP и обеспечивает, чтобы эти пакеты содержали MAC-адреса и IP-адреса виртуальной машины
`allow-arp`	Разрешить трафик ARP в обоих направлениях
`allow-ipv4`	Разрешить трафик IPv4 в обоих направлениях
`allow-ipv6`	Разрешить трафик IPv6 в обоих направлениях
`allow-incoming-ipv4`	Разрешить входящий трафик IPv4
`allow-incoming-ipv6`	Разрешить входящий трафик IPv6
`allow-dhcp`	Разрешить виртуальной машине запрашивать IP-адрес через DHCP (с любого DHCP-сервера)
`allow-dhcpv6`	Аналогично `allow-dhcp`, но для DHCPv6
`allow-dhcp-server`	Разрешить виртуальной машине запрашивать IP-адрес с указанного DHCP-сервера. Десятичный IP-адрес DHCP-сервера с точками должен быть указан в ссылке на этот фильтр. Имя переменной должно быть `DHCPSERVER`
`allow-dhcpv6-server`	Аналогично `allow-dhcp-server`, но для DHCPv6
`no-ip-spoofing`	Запретить виртуальной машине отправлять пакеты IPv4 с исходным IP-адресом, отличным от адреса в пакете
`no-ipv6-spoofing`	Аналогичен `no-ip-spoofing`, но для IPv6
`no-ip-multicast`	Запретить виртуальной машине отправлять многоадресные IP-пакеты
`no-ipv6-multicast`	Аналогичен `no-ip-multicast`, но для IPv6
`clean-traffic`	Запретить виртуальной машине подделку MAC, IP и ARP. Данный фильтр ссылается на остальные фильтры

Большинство фильтров по умолчанию являются составными (используют другие фильтры по умолчанию). При этом фильтр `clean-traffic` объединяет все фильтры в один фильтр, который затем можно связать с сетевой картой ВМ. Такое применение позволяет предотвратить самые распространенные атаки: подделка MAC, IP и ARP.

Посмотреть конфигурацию фильтра можно, выполнив команду:

```
virsh nwfilter-dumpxml <filter_name>|<UUID>
```

Для обновления фильтров рекомендуется использовать инструмент `virsh nwfilter-define`, что гарантирует корректное применение правил `iptables/ebtables` для ВМ.

Для управления сетевыми фильтрами используется набор инструментов `virsh`:

- `virsh nwfilter-define` — позволяет определить или обновить сетевой фильтр из XML-файла;
- `virsh nwfilter-undefine` — позволяет отменить определение сетевого фильтра;
- `virsh nwfilter-dumpxml` — позволяет вывести информацию о сетевом фильтре в формате XML;
- `virsh nwfilter-list` — выводит список сетевых фильтров;
- `virsh nwfilter-edit` — позволяет изменить конфигурацию XML для сетевого фильтра.

Для применения правила фильтрации требуется в XML-файл ВМ добавить ссылку на фильтр.

Пример

Связать фильтр `clean-traffic` с ВМ. Для этого необходимо отредактировать XML-файл ВМ, включив в `<interface>` ссылку на фильтр `<filterref>`, а также указав IP-адрес, который разрешено использовать ВМ:

```
<interface type='bridge'>
<mac address='52:54:00:56:44:32'/>
<source bridge='br1'/>
<ip address='10.33.8.131'/>
<target dev='vnet0'/>
<model type='virtio'/>
<filterref filter='clean-traffic'/>
</interface>
```

Если IP-адрес не указан, драйвер сетевого фильтра активирует `learning mode` (режим обучения). Для этого используется модуль `libpcap`, который отслеживает сетевой трафик, отправляемый ВМ, и пытается определить первый используемый им IP-адрес. В результате будет заблокирован трафик на этот адрес.

Драйвер сетевого фильтра использует комбинацию `ebtables`, `iptables` и `ip6tables` в зависимости от того, на какие протоколы ссылается фильтр. Правила фильтрации `clean-traffic` требуют использования только `ebtables`. При указании протоколов `tcp`, `udp` и др. (например, если добавить новый фильтр, блокирующий порт 25, для предотвращения спама по электронной почте) будет также использоваться `iptables`.

Пример

Фильтр сетевого трафика, предотвращающий подмену IP-адреса со стороны ВМ:

```
<filter name='no-ip-spoofing' chain='ipv4'>
<uuid>fce8ae33-e69e-83bf-262e-30786c1f8072</uuid>
<rule action='drop' direction='out' priority='500'>
<ip match='no' srcipaddr='$IP'/>
</rule>
</filter>
```

Данный фильтр реализует правило отбрасывания трафика, если IP-адрес (указанный через значение переменной IP) в исходящем IP-пакете не соответствует ожидаемому.

5.10.2. Реализация собственных правил

На хосте вся фильтрация трафика, осуществляемая подсистемой сетевых фильтров libvirt, сначала проходит через поддержку фильтрации, реализованную `ebtables`, и только затем через фильтры `iptables` или `ip6tables`. Если в дереве фильтров есть правила с протоколами `mac`, `stp`, `vlan arp`, `rarp`, `ipv4` или `ipv6`, то правила `ebtables` будут созданы автоматически.

Роль атрибута цепочки в XML-файле сетевого фильтра заключается в том, что внутри создается новая, определяемая пользователем, таблица `ebtables`, которая затем, например, если указана цепочка ARP, получает весь трафик ARP, исходящий от виртуальной машины или направляющийся к ней. Далее в корневой цепочке интерфейса генерируется правило, которое направляет весь IPv4-трафик в заданную пользователем цепочку. Следовательно, все правила трафика ARP затем должны быть помещены в фильтры, определяющие эту цепочку. Этот тип ветвления в пользовательские таблицы поддерживается только при фильтрации на уровне `ebtables`.

Можно создавать несколько цепочек для одного и того же протокола. Для этого имя цепочки должно иметь префикс одного из ранее перечисленных протоколов. Чтобы создать дополнительную цепочку для обработки ARP-трафика, можно указать цепочку с именем `arp-test`.

Например, можно фильтровать UDP-трафик по исходным и конечным портам, используя фильтр IP-протокола и указывая атрибуты для протокола, исходного и конечного IP-адресов и портов UDP-пакетов, которые должны быть приняты. Это позволяет фильтровать UDP-трафик с помощью `ebtables`. Однако после того, как пакет IP или IPv6 (например, UDP) прошел уровень `ebtables` и если в дереве фильтров есть хотя бы одно правило, которое создает экземпляры правил `iptables` или `ip6tables`, необходимо наличие правила, разрешающего прохождение пакета UDP. Этого можно добиться с помощью правила, содержащего соответствующий узел фильтрации трафика `udp` или `udp-ipv6`.

Пример

Для виртуальной машины `test` с интерфейсом `eth0` создать фильтр `test-eth0`, отвечающий следующим требованиям:

1) предотвращает подделку интерфейсов виртуальной машины MAC, IP и ARP4;
2) открывает только TCP-порты 22 и 80 интерфейса виртуальной машины;
3) позволяет виртуальной машине отправлять ping-трафик с интерфейса, но не позволяет пинговать виртуальную машину на интерфейсе;
4) позволяет виртуальной машине выполнять поиск DNS (UDP к порту 53).

Требование по предотвращению подделки выполняется существующим фильтром `clean-traffic`, поэтому возможно ссылаться на фильтр libvirt из создаваемого пользовательского фильтра.

Чтобы включить трафик для TCP-портов 22 и 80, необходимо добавить два правила для включения этого типа трафика. Чтобы разрешить виртуальной машине отправлять ping-трафик, необходимо добавить правило для ICMP-трафика. Возможно разрешить инициировать общий ICMP-трафик с виртуальной машины, а не только эхо-запросы ICMP и ответные сообщения. Чтобы в дальнейшем запретить остальной исходящий и входящий трафик для виртуальной машины, потребуется добавить правило, которое отбрасывает весь другой трафик.

Для создания фильтра необходимо:

1) создать файл фильтра в каталоге `/etc/libvirt/nwfilter/`:

```
sudo touch /etc/libvirt/nwfilter/test-eth0.xml
```

2) добавить в созданный файл XML-код:

```
<filter name='test-eth0'>
<!-- reference the clean traffic filter to prevent
MAC, IP and ARP spoofing. By not providing
and IP address parameter, libvirt will detect the
IP address the VM is using. -->
<filterref filter='clean-traffic'/>
<!-- enable TCP ports 22 (ssh) and 80 (http) to be reachable -->
<rule action='accept' direction='in'>
<tcp dstportstart='22'/>
</rule>
<rule action='accept' direction='in'>
<tcp dstportstart='80'/>
</rule>
<!-- enable general ICMP traffic to be initiated by the VM;
this includes ping traffic -->
```

```
<rule action='accept' direction='out'>
<icmp/>
</rule>
<!-- enable outgoing DNS lookups using UDP -->
<rule action='accept' direction='out'>
<udp dstportstart='53'/>
</rule>
<!-- drop all other traffic -->
<rule action='drop' direction='inout'>
<all/>
</rule>
</filter>
```

Правила в добавляемом XML-коде не содержат IP-адрес виртуальной машины в качестве исходного или конечного адреса, однако фильтрация трафика будет работать корректно, т.к. внутренняя интерпретация правил выполняется для каждого интерфейса, и правила интерпретируются на основе информации о том, какой интерфейс (TAP) отправил или получит пакет, а не на том, какой IP-адрес источника или получателя может быть;

3) загрузить (применить) правило:

```
sudo virsh -c qemu:///system nwfilter-define test-eth0.xml
```

4) в конфигурации виртуальной машины в секции `network` в качестве значения параметр `filterref filter` указать фильтр:

```
<interface type='bridge'>
<source bridge='mybridge'/>
<filterref filter='test-eth0'/>
</interface>
```

5) включить ВМ и проконтролировать, что правило применилось, для этого выполнить команду:

```
virsh nwfilter-binding-list
```

Результат выполнения команды:

```
Устройство порта   Фильтр
-------------------------------
vnet195            test-eth0
```

Чтобы более строго контролировать трафик ICMP и гарантировать, что с виртуальной машины могут отправляться только эхо-запросы ICMP, а виртуальная машина может получать только эхо-ответы ICMP, приведенное выше правило ICMP можно заменить следующими двумя правилами:

```
<!-- enable outgoing ICMP echo requests-->
<rule action='accept' direction='out'>
<icmp type='8'/>
</rule>
<!-- enable incoming ICMP echo replies-->
<rule action='accept' direction='in'>
<icmp type='0'/>
</rule>
```

5.11. Защита памяти в среде виртуализации

Для очистки остаточной информации в памяти средства вычислительной техники используются механизмы, которые обеспечивают очищение неиспользуемых блоков ФС непосредственно при их освобождении (распределении) и очищение активных разделов страничного обмена. Описание механизмов приведено в 8.1.

Управление механизмом очистки неиспользуемых блоков ФС непосредственно при их освобождении осуществляется с использованием инструмента `astra-sec-control`. Управление механизмом очистки активных разделов страничного обмена осуществляется с использованием инструмента `astra-swapwiper-control`.

Защита задач ядра и процессов пользователей при доступе к страницам оперативной памяти обеспечивается архитектурой и параметрами ядра ОС (см. 8.2).

Для предупреждения несанкционированных изменений модулей ядра в составе ОС применяется модуль lkrg (см. 16.5.16), который обеспечивает мониторинг угроз и блокирование несанкционированных изменений в ядре ОС. Таким образом, целостность всей области памяти ВМ при использовании lkrg обеспечивается по умолчанию. Для обработки критически важных данных рекомендуется на хосте использовать ОС с включенным модулем lkrg и ЗПС (см. 16.1), а в гостевых операционных системах применять ОС на усиленном или максимальном уровне защищенности с ядром hardened и включенным ЗПС.

Для изоляции и управления виртуальными гостевыми машинами используется технология KVM, которая включает специальный модуль ядра KVM и средство создания виртуального аппаратного окружения QEMU для изоляции и управления виртуальными гостевыми машинами. KVM, используя загруженный в память модуль ядра, с помощью драйвера пользовательского режима (который представляет собой модифицированный драйвер от QEMU) эмулирует слой аппаратного обеспечения, в среде которого могут создаваться и запускаться виртуальные машины.

В архитектуре KVM виртуальная машина выполняется как обычный процесс ОС, на который распространяется действие мер по изоляции процессов. Каждая гостевая опера-

ционная система в рамках своей виртуальной машины на уровне хостовой операционной системы является обычным процессом, выполняющимся с копией процесса `qemu`. Для каждого виртуального процессора создается поток. Эти процессы и потоки не отличаются от аналогичных в хостовой операционной системе. В части изоляции и разграничения доступа процессы виртуальных машин будут рассматриваться как обычные процессы операционной системы и для них будут применяться стандартные правила разграничения доступа и контроля ресурсов системы. Такие правила гарантируют, что программное обеспечение, запущенное в виртуальной машине, не сможет снизить производительность или нарушить работу других виртуальных машин или гипервизора.

Решение задачи изоляции адресных пространств процессов основано на архитектуре ядра ОС (см. 7.1). Изоляция областей памяти виртуальных машин обеспечивается путем применения механизма управления памятью аппаратной платформы IOMMU (input/output memory management unit), который отвечает за трансляцию виртуальных адресов, видимых аппаратным устройством, в физические адреса и позволяет задавать ограничения операций ввода-вывода для изоляции при использовании виртуализации.

Блок управления памятью для операций ввода-вывода (IOMMU) — это тип блока управления памятью (MMU), который подключает шину расширения с поддержкой прямого доступа к памяти (DMA) к основной памяти. Он расширяет системную архитектуру, добавляя поддержку виртуализации адресов памяти, используемых периферийными устройствами. Кроме того, он обеспечивает изоляцию и защиту памяти, позволяя системному программному обеспечению контролировать, к каким областям физической памяти может обращаться устройство ввода-вывода. Это также помогает фильтровать и переназначать прерывания от периферийных устройств. Устройство может получить доступ только к областям памяти, которые сопоставлены для него. Следовательно, неисправные и/или вредоносные устройства не могут повредить память. Изоляция памяти позволяет безопасно назначать устройства виртуальной машине без ущерба для хоста и других гостевых ОС.

Для применения механизма управления памятью аппаратной платформы необходимо включить аппаратную поддержку виртуализации в настройках UEFI для процессоров Intel/AMD и включить вложенную виртуализацию для ВМ. Для этого следует:

1) убедиться, что в настройках BIOS аппаратной платформы включена поддержка виртуализации, процессор AMD-Vi/Intel VT-d поддерживается и включен в настройках BIOS;

2) создать файл `/etc/modprobe.d/kvm.conf` с содержимым в зависимости от архитектуры:

```
kvm_intel nested=1
```

или

```
kvm_amd nested=1
```

3) выполнить перезагрузку хостовой машины. После загрузки хостовой машины проверить активацию nested виртуализации можно командой:

```
sudo virt-host-validate
```

4) для включения IOMMU на хосте гипервизора необходимо добавить в конфигурационный файл GRUB /etc/default/grub параметр intel_iommu=on или amd_iommu=on (в зависимости от архитектуры):

```
GRUB_CMDLINE_LINUX_DEFAULT="parsec.mac=0 quiet net.ifnames=0
intel_iommu=on"
```

Затем обновить конфигурацию загрузчика:

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

5) перезагрузить систему и выполнить проверку наличия модуля IOMMU в системе, выполнив команду:

```
sudo virt-host-validate
```

Результат выполнения команды при успешной проверке:

```
"QEMU: Checking if IOMMU is enabled by kernel"
```

6) в конфигурацию ВМ в секцию <features> добавить строку <ioapic driver="qemu"/>:

```
<features>
<acpi/>
<apic/>
<vmport state="off"/>
<ioapic driver="qemu"/>
</features>
```

7) в конфигурацию ВМ в секцию <devices> добавить следующие строки (указав для параметра iommu model в качестве значения intel или amd в зависимости от архитектуры):

```
<iommu model="intel">
<driver intremap="on" caching_mode="on"/>
</iommu>
```

8) загрузить ВМ и в гостевой операционной системе в конфигурационном файле GRUB /etc/default/grub также указать загрузку IOMMU:

```
GRUB_CMDLINE_LINUX_DEFAULT="parsec.mac=0 quiet net.ifnames=0
intel_iommu=on"
```

Затем обновить конфигурацию загрузчика:

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

9) перезагрузить гостевую операционную систему;

10) загрузить ВМ и в гостевой операционной системе выполнить команду:

```
sudo dmesg | grep –e DMAR –e IOMMU
```

В выводе проверить наличие информации об активации IOMMU:

```
[    0.008446] ACPI: DMAR 0x000000007FFE2961 0000E8 (v01 BOCHS  BXPC
00000001 BXPC 00000001)
[    0.008457] ACPI: Reserving DMAR table memory at [mem
0x7ffe2961-0x7ffe2a48]
[    0.021407] DMAR: IOMMU enabled
```

11) в гостевой операционной системе для проверки наличия модуля IOMMU выполнить команду:

```
sudo virt-host-validate
```

Результат выполнения команды при успешной проверке:

```
"QEMU: Checking if IOMMU is enabled by kernel"
```

5.12. Применение механизма контроля целостности областей памяти по запросу из гостевой операционной системы

Механизм контроля целостности областей памяти ВМ базируется на создании канала приема-передачи данных между ВМ и средством виртуализации с использованием механизма передачи запросов из гостевой операционной системы посредством служебного сокета типа `vsock`.

Контроль осуществляется средством виртуализации путем вычисления контрольных сумм заданных областей памяти в соответствии с ГОСТ Р 34.11-2012 с последующим сравнением вычисленных значений с эталонными. Подсчет контрольных сумм осуществляется периодически в процессе работы гостевой операционной системы. При миграции ВМ данные об эталонных значениях установленных на контроль областей памяти будут также передаваться вместе с другой служебной информацией о ВМ для возможного последующего выполнения контроля целостности на другом хосте.

Для приема запросов на осуществление контроля целостности областей памяти гостевой операционной системы используется служба `vsockd` на хосте гипервизора, которая открывает слушающий сокет VSOCK. Виртуальная машина передает в сокет сообщение с параметрами `start_addr` (физический адрес начала блока памяти гостевой операционной системы) и `end_addr` (физический адрес конца блока памяти гостевой операционной системы). Служба `vsockd` на хосте принимает сообщение, получает адрес CID VSOCK и отправляет запрос в средство виртуализации (libvirt) для последующего вычисления контрольной суммы по алгоритму ГОСТ Р 34.11-2012.

Эталонные значения контрольных сумм областей памяти хранятся в памяти средства виртуализации до выключения ВМ. При выявлении нарушения целостности областей памяти

гостевой операционной системы возможна блокировка работы ВМ и регистрация фактов нарушения целостности объектов контроля в журнал подсистемы регистрации событий (см. 6.5) и в журнал `/var/log/syslog`.

Для использования механизма в системе должен быть установлен пакет `astra-kvm-secure`.

Настройка механизма контроля целостности файлов гостевой операционной системы осуществляется в конфигурационном файле `/etc/libvirt/libvirtd.conf`. Настройку рекомендуется выполнять с использованием `virsh` путем выполнения команды:

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

Для включения механизма контроля целостности областей памяти гостевой операционной системы необходимо в файле `/etc/libvirt/libvirtd.conf` для параметра `memory_integrity_check_period_s` задать значение периода проверки (в секундах):

```
memory_integrity_check_period_s = 30
```

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

Для принудительного выключения функционирующей ВМ в случае нарушения целостности установленных на контроль областей памяти необходимо в файле `/etc/libvirt/libvirtd.conf` для параметра `memory_integrity_check_shutdown_domain` задать значение `1`:

```
memory_integrity_check_shutdown_domain = 1
```

При этом будет обеспечиваться регистрация фактов нарушения целостности областей памяти с последующим принудительным выключением ВМ, если значение контрольной суммы хотя бы одной установленной на контроль области памяти гостевой операционной системы не совпадет с эталонным значением, хранящимся в конфигурационном файле ВМ.

Для применения настроек следует перезагрузить службу `libvirtd`:

```
sudo systemctl restart libvirtd
```

К ВМ необходимо подключить сокет VirtIO VSOCK для связи гостевой операционной системы с хостом. Для этого необходимо в `virt-manager`:

1) открыть свойства ВМ и нажать **[Добавить оборудование]**;
2) в окне «Добавление виртуального оборудования» выбрать «VirtIO VSOCK»;
3) во вкладке «Подробности» для параметра «CID гостевой системы» установить флаг «Авто»;
4) нажать **[Готово]**.

В результате в конфигурационном файле ВМ будет сформирована запись вида:

```
<vsock model="virtio">
```

```
<cid auto="yes" address="3"/>
<address type="pci" domain="0x0000" bus="0x07" slot="0x00" function="0x0"/>
</vsock>
```

Каждому сокету автоматически присваивается метка CID гостевой операционной системы, которая обозначает сквозной идентификатор каждой ВМ.

После добавления сокета необходимо выполнить запуск гостевой операционной системы.

Использование механизма контроля целостности возможно при условии использования ОС в качестве гостевой операционной системы и установки в ней гостевого агента `qemu-guest-agent` из состава ОС.

Установку на контроль областей памяти обеспечивает инструмент `/sbin/memcontrol_VM`, принимающий в качестве аргументов данные об идентификаторе контролируемого процесса (pid) и диапазон виртуальных адресов блока памяти в шестнадцатеричном формате.

Синтаксис команды:

1) в случае использования виртуальных адресов процесса с pid:

```
memcontrol_VM --virt <pid> <address_start> <address_end>
```

2) в случае использования физических адресов памяти гостевой операционной системы для постановки на контроль:

```
memcontrol_VM --phys <address_start> <address_end>
```

Для определения диапазона виртуальных адресов блока памяти ВМ в гостевой операционной системе используется инструмент `ps`, выводящий идентификатор процесса по его названию, и файл `/proc/<pid>/maps`, содержащий адреса областей памяти, которые используются процессом в данный момент, с указанием прав доступа к ним.

Пример

Для установки на контроль областей памяти, занимаемых процессом `syslog-ng`, необходимо:

1) определить идентификатор процесса командой:

```
ps aux | grep syslog-ng
```

Вывод команды:

```
root  394  0.5  1.8  624928  38180  ?      Ssl  16:20  0:27
/usr/sbin/syslog-ng -F --no-caps
u    1438  0.0  0.0    6228    872  pts/2  S+   17:44  0:00 grep
syslog-ng
```

В выводе команды pid процесса равен `394`;

2) выполнить поиск адресов областей памяти, которые используются процессом с pid, равным `394`, в данный момент:

```
sudo cat /proc/394/maps
```

Вывод команды:

```
00400000-00402000 r--p 00000000 08:01 1320884 /usr/sbin/syslog-ng
00402000-00403000 r-xp 00002000 08:01 1320884 /usr/sbin/syslog-ng
00403000-00404000 r--p 00003000 08:01 1320884 /usr/sbin/syslog-ng
00404000-00405000 r--p 00003000 08:01 1320884 /usr/sbin/syslog-ng
00405000-00406000 rw-p 00004000 08:01 1320884 /usr/sbin/syslog-ng
00c12000-01165000 rw-p 00000000 00:00 0 [heap]
731f6c000000-731f6c070000 rw-p 00000000 00:00 0
731f6c070000-731f70000000 ---p 00000000 00:00 0
731f737ff000-731f73800000 ---p 00000000 00:00 0
731f73800000-731f74000000 rw-p 00000000 00:00 0
731f74000000-731f74023000 rw-p 00000000 00:00 0
731f74023000-731f78000000 ---p 00000000 00:00 0
731f78000000-731f78021000 rw-p 00000000 00:00 0
731f78021000-731f7c000000 ---p 00000000 00:00 0
731f7c000000-731f7c021000 rw-p 00000000 00:00 0
731f7c021000-731f80000000 ---p 00000000 00:00 0
731f80000000-731f80021000 rw-p 00000000 00:00 0
731f80021000-731f84000000 ---p 00000000 00:00 0
731f84000000-731f84032000 rw-p 00000000 00:00 0
731f84032000-731f88000000 ---p 00000000 00:00 0
731f88000000-731f88021000 rw-p 00000000 00:00 0
731f88021000-731f8c000000 ---p 00000000 00:00 0
731f8c000000-731f8c025000 rw-p 00000000 00:00 0
731f8c025000-731f90000000 ---p 00000000 00:00 0
```

Сегмент с правами доступа `r-xp` указывает на хранение исполняемого кода `syslog-ng`. Соответствующая ему область памяти имеет диапазон адресов `00402000-00403000` (`0x402000-0x403000` в шестнадцатеричном формате);

3) из гостевой операционной системы выполнить запрос постановки на контроль целостности выбранной области памяти для процесса с pid, равным `394`:

```
sudo memcontrol_VM --virt 394 0x402000 0x403000
```

В выводе команды будут указаны физические адреса начала и конца блока поставленной на контроль памяти гостевой операционной системы и рассчитанное для этой области значение эталонной контрольной суммы.

В результате контрольная сумма области памяти будет записана в памяти средства виртуализации с целью дальнейшего контроля целостности.

Для управления режимом контроля целостности областей памяти для отдельно взятой виртуальной машины используется команда:

```
virsh memory-integrity <domain> [--clear] [--start] [--stop]
```

5.13. Ограничение программной среды в среде виртуализации

Контроль за запуском компонентов программного обеспечения, обеспечивающий выявление и блокировку запуска компонентов программного обеспечения, не включенных в перечень (список) компонентов, разрешенных для запуска, осуществляется штатными средствами ОС:

1) механизмом динамического контроля целостности (режим ЗПС) исполняемых файлов и разделяемых библиотек формата ELF при запуске программы на выполнение (см. 16.1);

2) применением режима Киоск-2, который служит для ограничения прав пользователей на запуск программ в ОС (см. 16.2). Степень этих ограничений задается маской киоска, которая накладывается на права доступа к исполняемым файлам при любой попытке пользователя получить доступ.

Выявление и блокировка запуска компонентов программного обеспечения, целостность которого нарушена, осуществляется механизмом динамического контроля целостности исполняемых файлов и разделяемых библиотек формата ELF путем внедрения цифровой подписи в файлы, входящие в состав ПО. При включенном режиме контроля цифровой подписи выполняться будут только подписанные исполняемые файлы.

5.14. Централизованное управление

В среде виртуализации реализовано создание, модификация, хранение, получение и удаление (в т. ч. централизованное) образов виртуальных машин в информационной (автоматизированной) системе.

Для централизованного хранения образов ВМ используется хранилище данных — пул (pool). Хранилище определяет физическое расположение файлов-образов и может быть различных типов. Хранилище должно быть доступно для подключения на серверах виртуализации.

Для распределенного хранилища следует использовать кластерную файловую систему ocfs2 или блочное устройство ceph/rbd. Образы виртуальных машин помещаются в данное хранилище и становятся доступными для всех узлов виртуализации.

ОС поддерживает возможность миграции виртуальных машин между серверами виртуализации. Миграция позволяет перенести работу виртуальной машины с одного физического хоста (сервера виртуализации) на другой без остановки ее работы. При этом образ диска виртуальной машины доступен на обоих серверах виртуализации. Для этого используется распределенное хранилище данных: на обоих серверах виртуализации хранилище должно монтироваться в одно и то же место и образ виртуальной машины должен быть расположен в данном хранилище.

Управление перемещением виртуальных машин реализуется с использованием интерфейсов управления средствам виртуализации libvirt в соответствии с установленными правилами сетевого взаимодействия.

Для решения задач централизованного протоколирования и анализа журналов аудита используется Zabbix.

Для решения задач централизованного сбора журналов с удаленных хостов виртуализации возможно применение встроенных механизмов сбора информации о событиях `syslog-ng`. Вместе с установкой пакета `astra-kvm-secure` обеспечивается создание конфигурационных файлов для удаленного сбора данных о событиях безопасности, связанных с функционированием средства виртуализации, из журналов событий безопасности.

5.14.1. Пример организации распределенного хранилища

Пример организации распределенного хранилища на гипервизоре `host` с установленной ОС. На хосте будут развернуты два виртуальных сервера виртуализации (виртуальные машины `alse-virtcert1` и `alse-virtcert2`) с установленной на них ОС. К виртуальным машинам подключен пул хранения данных (пул `ocfs2-vol`).

Для организации распределенного хранилища необходимо:

1) на гипервизоре `host` запустить `virt-manager`, выбрать соединение QEMU/KVM, нажатием правой кнопкой мыши раскрыть контекстное меню и выбрать «Детали»;

2) в открывшемся окне перейти во вкладку «Пространство данных», нажать на кнопку **[+]** (**[Добавить пул]**);

3) в открывшемся окне в поле «Название» ввести наименование пула «pool» и нажать **[Готово]**;

4) после создания хранилища можно добавлять образы или тома в это хранилище. Для этого во вкладке «Пространство данных» выбрать «pool», в области просмотра справа во вкладке «Подробности» в строке «Список томов» нажать на **[+]** (**[Создать том]**) в соответствии с рис. 5;

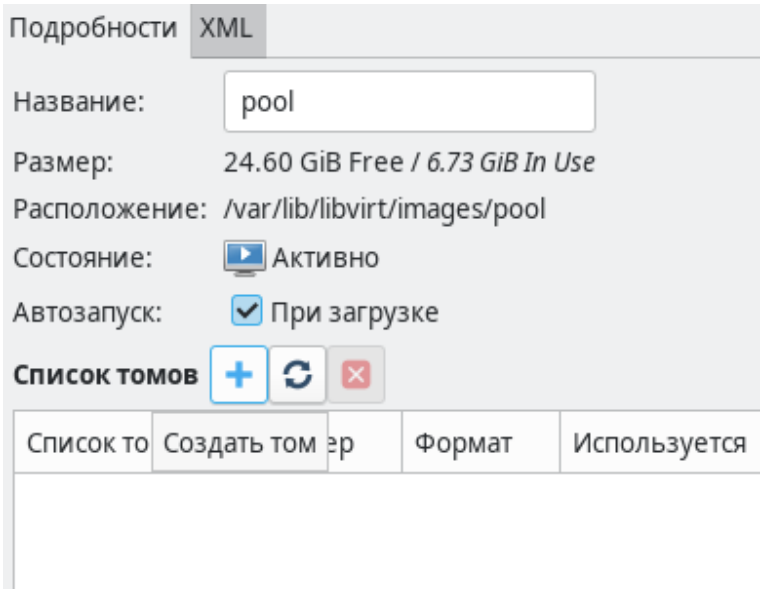

Рис. 5

5) в открывшемся окне в поле «Название» ввести наименование тома «ocfs2-vol», в выпадающем списке «Формат» выбрать «raw» и нажать **[Готово]**;

6) с использованием `virt-manager` создать на гипервизоре две ВМ `alse-virtcert1` и `alse-virtcert2`, которые будут настроены в качестве серверов виртуализации. На каждой ВМ выполнить установку ОС с актуальным обновлением безопасности, при этом:

а) для ВМ `alse-virtcert1` в процессе установки ОС задать имя хоста `astra`. При выборе ПО для установки отметить средства виртуализации. После установки задать ВМ IP-адрес `10.0.2.15`;

б) для ВМ `alse-virtcert2` в процессе установки ОС задать имя хоста `astra2`. При выборе ПО для установки отметить средства виртуализации. После установки задать ВМ IP-адрес `10.0.2.16`;

7) ВМ должны быть доступны друг другу по сети и пинговаться по имени хоста (в `/etc/hosts` должны быть указаны IP-адреса и имена ВМ);

8) для каждой ВМ (`alse-virtcert1` и `alse-virtcert2`) выполнить подключение созданного пула:

а) на гипервизоре `host` открыть свойства ВМ, нажав правой кнопкой мыши на ВМ и в контекстном меню выбрав «Открыть», далее нажать кнопку **[Показать виртуальное оборудование]** и затем нажать **[Добавить оборудование]** в соответствии с рис. 6;

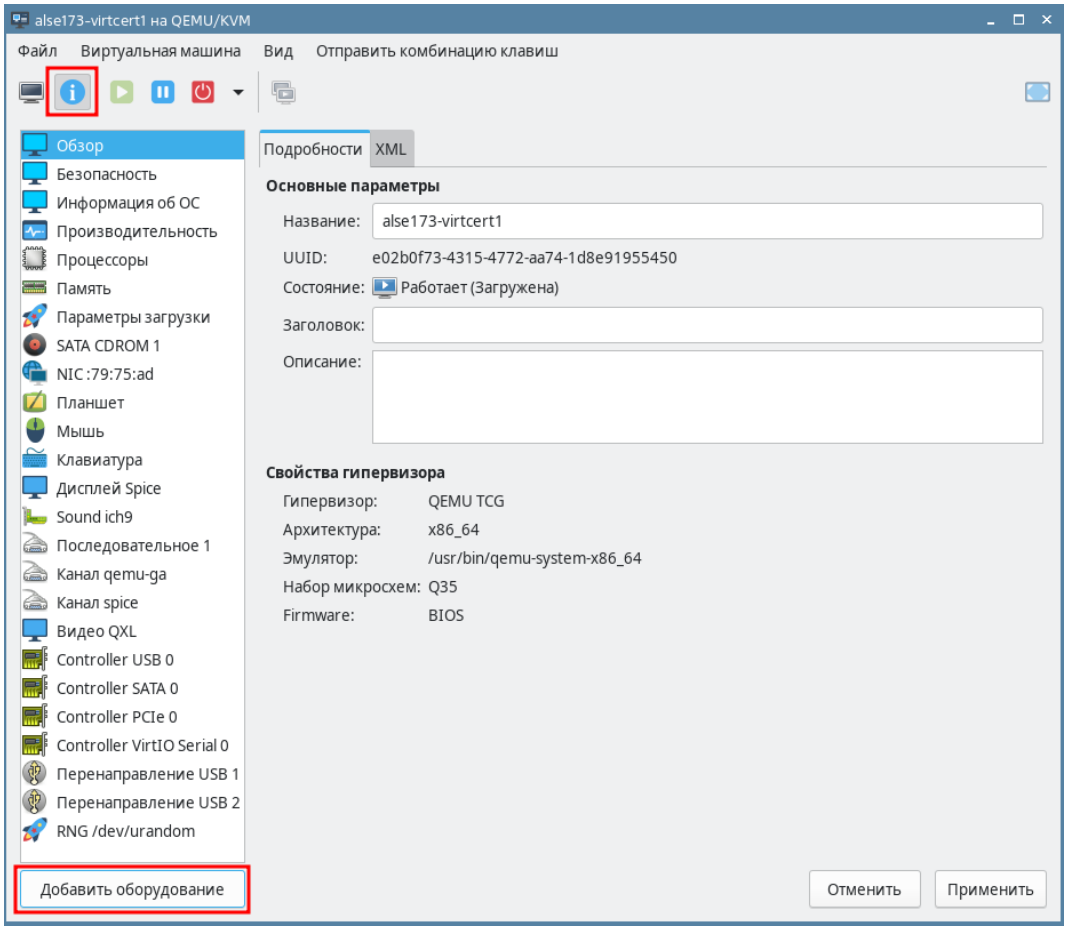

Рис. 6

б) выбрать раздел «Хранилище» и в области просмотра справа во вкладке «Подробности» установить флаг «Выбрать или создать дополнительное пространство данных», затем нажать **[Настроить]**;

в) в открывшемся окне выбрать «pool», во вкладке «Подробности» в списке томов выбрать «ocfs2-vol.img» и затем нажать **[Выбор тома]**;

г) во вкладке «Подробности» в выпадающем списке «Тип устройства» выбрать «Дисковое устройство»;

д) во вкладке «Подробности» в выпадающем списке «Тип шины» выбрать «VirtIO»;

е) во вкладке «Подробности» в поле «Дополнительные параметры» установить флаг «Общий»;

9) на каждой ВМ (`alse-virtcert1` и `alse-virtcert2`) сконфигурировать `ocfs2`, для этого в ОС каждой ВМ выполнить следующие действия:

а) выполнить команду:

```
lsblk
```

Результат выполнения команды:

```
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sr0 11:0 1 1024M 0 rom
```

```
vda 252:0 0 20G 0 disk
|--vda1 252:1 0 512M 0 part /boot/efi
|--vda2 252:2 0 18,6G 0 part /
|--vda3 252:3 0 976M 0 part [SWAP]
vdb 252:16 0 10G 0 disk
```

Убедиться, что в выводе присутствует подключенный диск `vdb`;

б) если пакет `ocfs2-tools` не был установлен, то установить его командой:

```
sudo apt install ocfs2-tools
```

в) сконфигурировать `ocfs2`:

- создать конфигурационный файл `/etc/ocfs2/cluster.conf` со следующим содержимым:

```
node:
ip_port = 7777
ip_address = 10.0.2.15
number = 0
name = astra
cluster = ocfs2cluster

node:
ip_port = 7777
ip_address = 10.0.2.16
number = 1
name = astra2
cluster = ocfs2cluster

cluster:
node_count = 2
name = ocfs2cluster
```

- выполнить команду для:

```
sudo dpkg-reconfigure ocfs2-tools
```

На запрос системы запускать кластер при загрузке нажать **[Да]**. Затем в следующем окне подтвердить имя кластера (`ocfs2cluster`), нажав **[OK]**;

- перезапустить службу `o2cb`;

```
sudo systemctl restart o2cb
```

г) только на одной из ВМ выполнить форматирование ФС `/dev/vdb`:

```
sudo mkfs.ocfs2 /dev/vdb
```

Результат выполнения команды:

```
mkfs.ocfs2 1.8.5
```

```
Cluster stack: classic o2cb
Label:
Features: sparse extended-slotmap backup-super unwritten inline-data
strict-journal-super xattr indexed-dirs refcount discontig-bg
append-dio
Block size: 4096 (12 bits)
Cluster size: 4096 (12 bits)
Volume size: 10737418240 (2621440 clusters) (2621440 blocks)
Cluster groups: 82 (tail covers 8704 clusters, rest cover 32256
clusters)
Extent allocator size: 4194304 (1 groups)
Journal size: 67108864
Node slots: 8
Creating bitmaps: done
Initializing superblock: done
Writing system files: done
Writing superblock: done
Writing backup superblock: 2 block(s)
Formatting Journals: done
Growing extent allocator: done
Formatting slot map: done
Formatting quota files: done
Writing lost+found: done
mkfs.ocfs2 successful
```

д) узнать UUID `/dev/vdb`, выполнив команду:

```
sudo blkid
```

Результат выполнения команды:

```
/dev/vda1: UUID="61F0-BB35" TYPE="vfat"
PARTUUID="f14d9f7b-013d-43f2-b7a1-e7e515b6cacb"
/dev/vda2: UUID="8fcc63aa-ba8b-4ae5-bbd0-4714a279656d" TYPE="ext4"
PARTUUID="fad1023c-3ad2-465a-aaf1-878af2bca8b0"
/dev/vda3: UUID="73e7faad-1a0d-4706-b434-c6436918b17c" TYPE="swap"
PARTUUID="36af9ea7-2e1d-4534-b9a8-0dd87427ab67"
/dev/vdb: UUID="279b26f3-e332-4d7b-b37d-8549b5c35bba" TYPE="ocfs2"
```

е) создать каталог `/mnt/ocfs2/` командой:

```
sudo mkdir /mnt/ocfs2
```

ж) добавить в `/etc/fstab` запись с UUID `/dev/vdb`:

```
UUID=279b26f3-e332-4d7b-b37d-8549b5c35bba /mnt/ocfs2 ocfs2 defaults
```

0 0

з) выполнить перезагрузку каждой ВМ;

10) добавить централизованное хранилище образов на каждом сервере виртуализации, для этого:

а) войти в ОС виртуальной машины `alse-virtcert1`;

б) запустить `virt-manager`, выбрать соединение QEMU/KVM, нажатием правой кнопкой мыши раскрыть контекстное меню и выбрать «Детали»;

в) в открывшемся окне перейти во вкладку «Пространство данных», нажать на кнопку **[+]** (**[Добавить пул]**);

г) в открывшемся окне в поле «Название» ввести наименование пула «ocfs2», в поле «Путь к цели» указать «/mnt/ocfs2» и нажать **[Готово]**;

д) проверить наличие подключенного хранилища — пул должен отображаться во вкладке «Пространство данных» в соответствии с рис. 7;

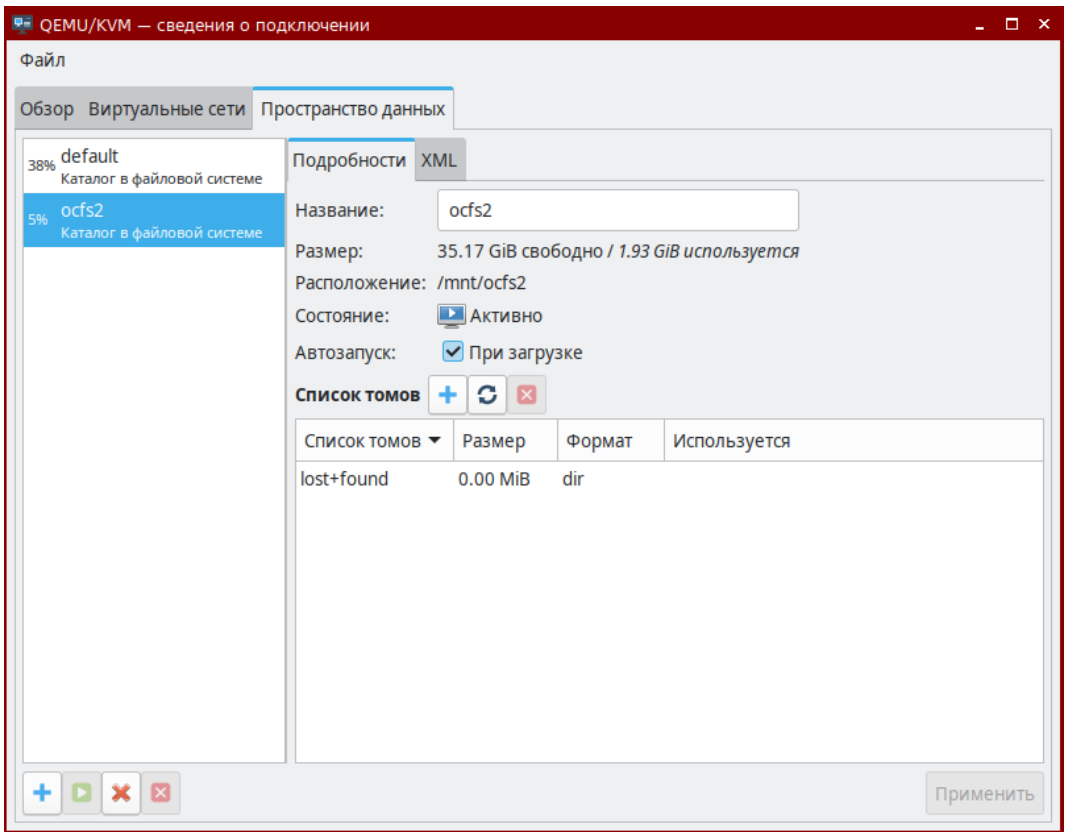

Рис. 7

е) войти в ОС ВМ `alse-virtcert2`;

ж) запустить `virt-manager`, выбрать соединение QEMU/KVM, нажатием правой кнопкой мыши раскрыть контекстное меню и выбрать «Детали»;

з) в открывшемся окне перейти во вкладку «Пространство данных», нажать на кнопку **[+]** (**[Добавить пул]**);

и) в открывшемся окне в поле «Название» ввести наименование пула «ocfs2», в поле «Путь к цели» указать «/mnt/ocfs2» и нажать **[Готово]**;

к) проверить наличие подключенного хранилища — пул должен отображаться во вкладке «Пространство данных» в соответствии с рис. 7;

11) в ОС виртуальной машины `alse-virtcert2` выполнить копирование образа ВМ `alse-vanilla-1.7.2-qemu-max-mg8.0.0.qcow2` в хранилище:

```
cp /var/lib/libvirt/images/alse-vanilla-1.7.2-qemu-max-mg8.0.0.qcow2
    /mnt/ocfs2
```

12) образ должен отобразиться во вкладке «Пространство данных» в списке томов (предварительно требуется нажать кнопку обновления списка томов) в соответствии с рис. 8;

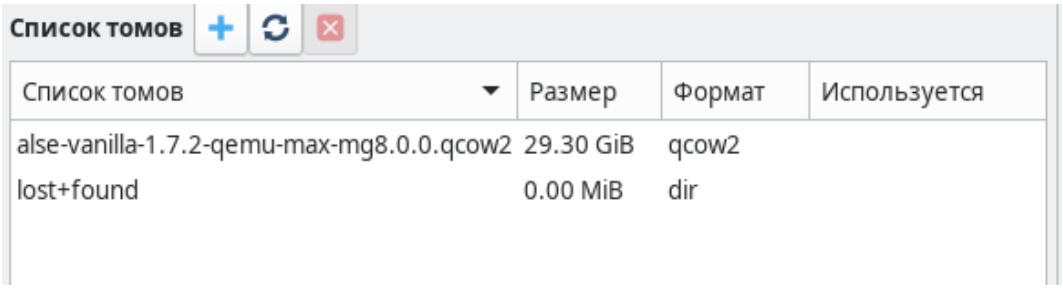

Рис. 8

13) в ОС виртуальной машины `alse-virtcert1` проверить наличие образа во вкладке «Пространство данных» в списке томов (предварительно требуется нажать кнопку обновления списка томов) в соответствии с рис. 8;

14) в ОС виртуальной машины `alse-virtcert1` выполнить создание новой ВМ с помощью `virt-manager`, указав в качестве источника ВМ образ в хранилище `/mnt/ocfs2` в соответствии с рис. 9;

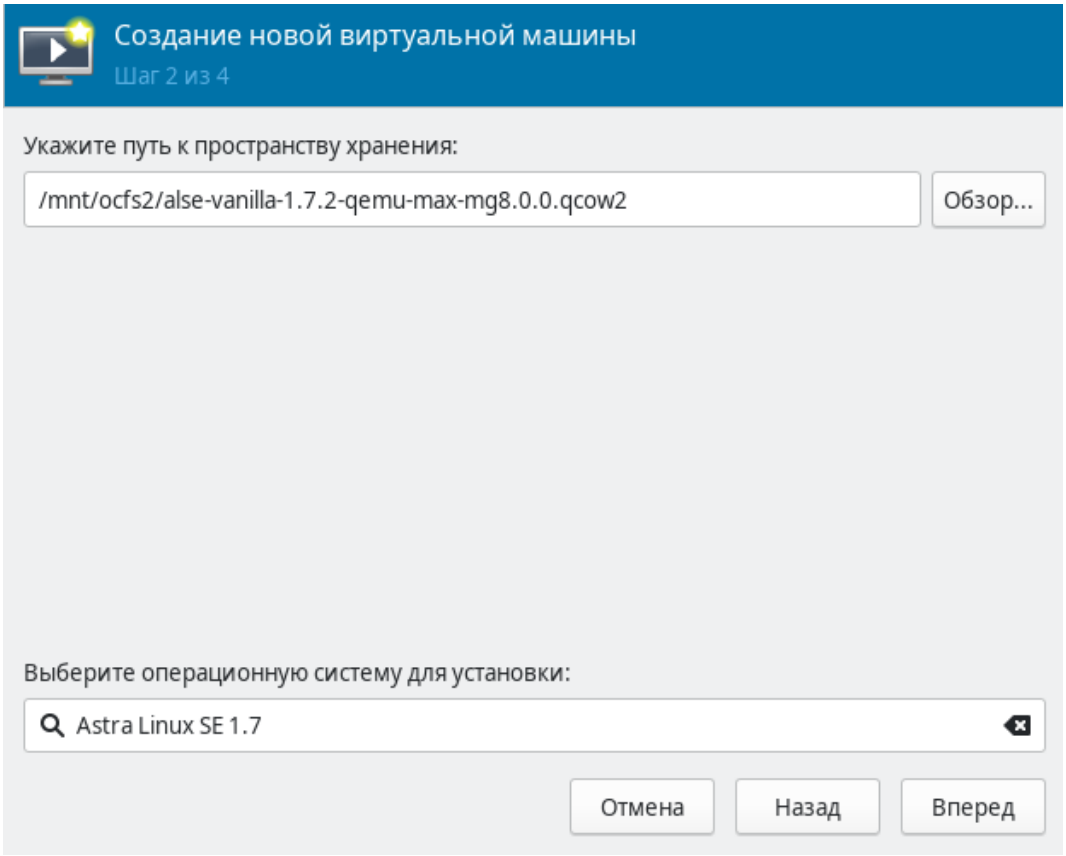

Рис. 9

5.14.2. Пример миграции виртуальных машин

Для осуществления миграции ВМ между серверами необходимо предварительно выполнить подключение второго сервера виртуализации:

1) войти в ОС виртуальной машины `alse-virtcert1` на хосте `astra`;

2) запустить `virt-manager`, выбрать пункт меню «Файл — Добавить соединение» и добавить подключение по ssh к хосту `astra2` в соответствии с рис. 10;

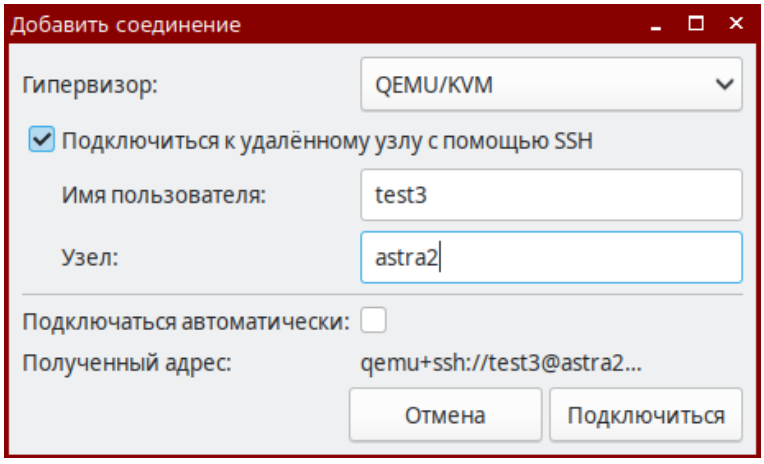

Рис. 10

3) проверить наличие подключения к хосту `astra2` в списке соединений в соответствии с рис. 11.

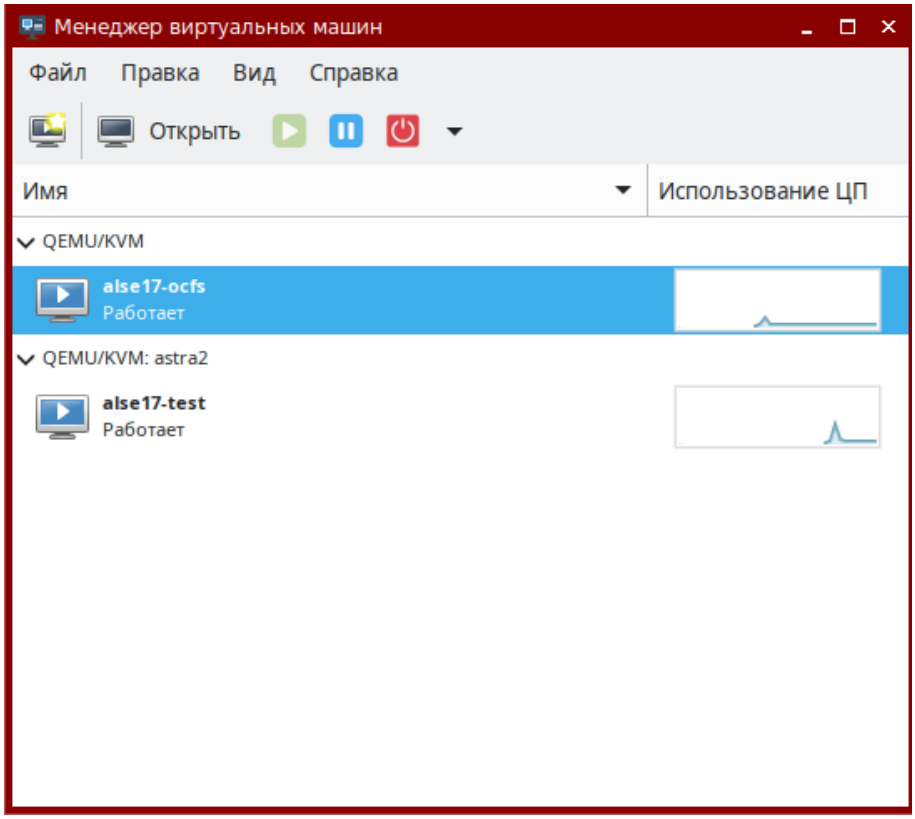

Рис. 11

Для выполнения миграции ВМ необходимо в главном окне `virt-manager` нажать правой кнопкой мыши на нужную ВМ (ВМ при этом должна быть запущена) и в открывшемся меню выбрать «Миграция». В появившемся окне выбрать конечный хост и нажать **[Миграция]** в соответствии с рис. 12.

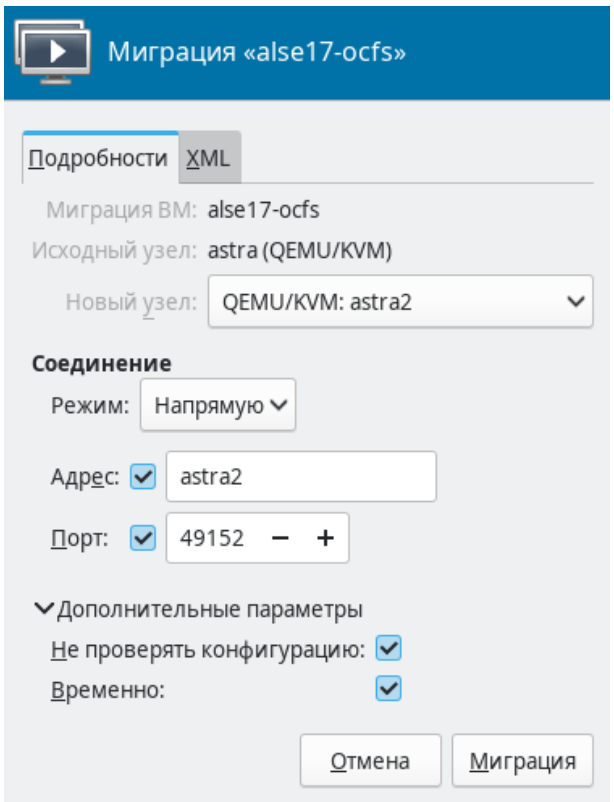

Рис. 12

6. РЕГИСТРАЦИЯ СОБЫТИЙ БЕЗОПАСНОСТИ

В ОС регистрация событий безопасности выполняется с учетом требований ГОСТ Р 59548-2022 «Защита информации. Регистрация событий безопасности. Требования к регистрируемой информации».

Регистрация событий безопасности реализуется использованием службы `auditd` и подсистемы регистрации событий из состава ОС, описание которой приведено в 6.5.

Служба `auditd` выполняет регистрацию событий объектов файловой системы (аудит файлов) и пользователей (аудит процессов) согласно заданным правилам. Работа с правилами аудита описана в 6.1 и 6.2. Регистрация событий осуществляется в журнал аудита, описание которого приведено в 6.3. Описание настройки параметров аудита приведено в 6.4.

Применение настроенных параметров аудита процессов осуществляется PAM-модулем `pam_parsec_aud`. По умолчанию регистрация событий аудита процессов включена в PAM-сценарии: `fly-dm`, `fly-dm-np`, `login`, `su`, `sshd`, `sumac.xauth`. Для регистрации событий аудита процессов пользователя, проходящего аутентификацию через другие PAM-сценарии, необходимо включить в соответствующие сценарии строку следующего вида:

```
session required pam_parsec_aud.so
```

В библиотеках подсистемы безопасности PARSEC реализован программный интерфейс для регистрации событий с использованием службы регистрации событий безопасности ОС, применяемый для регистрации событий в СУБД PostgreSQL (описание приведено в 6.6) и комплексе программ электронной почты.

6.1. Правила регистрации событий

Регистрация событий осуществляется в соответствии с правилами аудита, правила делятся на два типа:

1) временные — действуют до перезагрузки системы. Такие правила задаются посредством инструмента `auditctl`;

2) постоянные — действуют всегда, даже после перезагрузки системы. Такие правила задаются в файлах формата `*.rules`, располагающихся в каталоге `/etc/audit/rules.d/`.

Подробное описание временных правил аудита, а также синтаксис использования инструмента `auditctl` приведены в справочной странице `man auditctl`.

Примеры:

1. Записывать все системные вызовы от процесса с идентификатором (PID) `1005`:

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

2. Записывать все файлы, открытые пользователем с идентификатором `auid 510`:

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

При добавлении постоянных правил аудита в файлах используется синтаксис инструмента `auditctl` без указания имени инструмента.

Пример

Записывать все системные вызовы от процесса с идентификатором (PID) `1005`:

```
-a exit,always -S all -F pid=1005
```

Постоянные правила, необходимые для работы встроенного аудита, заданы в файле `/etc/audit/rules.d/10-parsec.rules`:

1) аудит процессов (данное правило необходимо для работы утилит `useraud` и `psaud`):

```
-a always,exit -F subj_type=psaud -S all -k parsec-p
```

где `-a always,exit` — записывать события в журнал, добавить правило в список `exit` (события, происходящие при выходе из системного вызова);

`-F subj_type=psaud` — обрабатывать правила, заданные утилитами `useraud` и `psaud`;

`-S all` — перехватывать события при любых системных вызовах;

`-k parsec-p` — присвоить ключ фильтрации `parsec-p` событиям по данному правилу;

2) аудит файлов (данное правило необходимо для работы команд `getfaud` и `setfaud`):

```
-a always,exit -F obj_type=faud -S all -k parsec-f
```

где `-a always,exit` — записывать события в журнал, добавить правило в список `exit` (события, происходящие при выходе из системного вызова);

`-F obj_type=faud` — обрабатывать правила, заданные командой `setfaud`;

`-S all` — перехватывать события при любых системных вызовах;

`-k parsec-f` — присвоить ключ фильтрации `parsec-f` событиям по данному правилу.

В дополнение к этим правилам можно задавать собственные правила аудита.

Постоянные правила рекомендуется добавлять в файл `/etc/audit/rules.d/audit.rules`. При необходимости возможно создать в каталоге `/etc/audit/rules.d/` новый файл вида `*.rules` с произвольным именем и задать в нем нужные правила. Файл `/etc/audit/rules.d/audit.rules` можно редактировать вручную или с помощью графической утилиты `system-config-audit` («Конфигурация аудита», описание утилиты приведено в электронной справке). Другие файлы правил можно редактировать только вручную.

6.2. Регистрация событий на основе меток безопасности

В ОС поддерживается регистрация событий на основе меток безопасности субъектов и объектов. Для этого используются параметры `subj_type` и `obj_type` и операции «=» (равно) и «!=» (не равно) с ними.

Метка имеет следующий вид (при этом любое из значений может быть пустым):

```
<Уровень>:<Уровень_целостности>:<Категории>:<Тип>
```

Пример

Регистрация событий доступа к объектам каталога субъектами нулевого уровня конфиденциальности:

```
-a always,exit -S all -F dir=/var/level1_dir -F perm=rwxa -F subj_type=0:::
```

6.3. Журнал аудита

Служба `auditd` регистрирует события безопасности в журнале аудита `/var/log/audit`.

Также в журнале аудита `/var/log/audit` регистрируются действия с журналом событий подсистемы регистрации событий (см. 6.5) — удаление, переименование, перемещение, ротация файла журнала событий.

Для просмотра журнала аудита используется графическая утилита `ksystemlog`, описание утилиты приведено в электронной справке.

Действия с журналом аудита `/var/log/audit` (удаление, переименование, перемещение файла журнала аудита) регистрируются подсистемой регистрации событий и указываются в журнале событий (см. 6.5.2).

6.4. Средства управления аудитом

6.4.1. Графические утилиты

Для управления аудитом могут использоваться следующие графические утилиты (описание утилит доступно в электронной справке):

- `fly-admin-smc` («Управление политикой безопасности») — управление аудитом, привилегиями и мандатными атрибутами пользователей, работа с пользователями и группами;
- `system-config-audit` («Конфигурация аудита») — включение и выключение регистрации событий, настройка службы `auditd`, настройка журнала аудита, а также добавление, удаление и редактирование правил аудита;
- `fly-admin-events` («Настройка регистрации системных событий») — утилита из состава подсистемы регистрации событий (см. 6.5), с помощью которой доступно выполнять регистрацию событий запуска и остановки службы `auditd`, регистрацию

событий добавления и удаления правил auditd, регистрацию действий с журналом аудита. Дополнительно утилита позволяет добавлять правила аудита.

6.4.2. getfaud

Команда getfaud служит для получения списков правил регистрации событий над файловыми объектами. Следующие события доступны для регистрации:

- 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, описание которой приведено в 6.4.3.

Синтаксис команды:

```
getfaud [<параметр> [<параметр>...]] <файлы_и/или_каталоги>
```

Описание параметров getfaud приведено в таблице 33.

Таблица 33

Параметр	Описание
-d, --default	Работать со списком правил регистрации по умолчанию
-R, --recursive	Для подкаталогов рекурсивно
-L, --logical	Следовать по символическим ссылкам
-P, --physical	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

Описание команды приведено в справочной странице man getfaud.

6.4.3. setfaud

Команда `setfaud` устанавливает на файлы списки правил регистрации событий. 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. 6.4.2).

Только администратор может изменять списки правил регистрации событий у файлов.

Синтаксис команды:

```
setfaud [<параметр> [<параметр>...]] [<правила_регистрации>] <файлы>
```

Правила регистрации задаются в виде:

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

где `<пользователь>` и `<группа>` — символические или численные идентификаторы пользователя и группы;

`u:` — правило для пользователя;

`g:` — правило для группы;

`o:` — правило для остальных пользователей (для которых правила не заданы явно).

Флаги регистрации задаются в виде:

```
<флаги_успешных_операций>[[:<флаги_неуспешных_операций>], ...]
```

При этом флаги операций могут иметь вид:

1) `<+|-><имя_регистрируемого_события>,...`

Пример

Регистрация успешного события выполнения файла:

```
+exec
```

Регистрация неуспешного события открытия файла:

```
-open
```

2) `[+|-]<число>`

3) `<сокращенное_имя_регистрируемого_события>,...`

Пример

Регистрация успешных событий открытия и удаления файла:

```
ou
```

Пример

Аудит всех операций (кроме изменения файла) — как успешных, так и не успешных — с файлом `filename>`, произведенных пользователями, для которых правила не заданы явно:

```
setfaud -m o:oxudnarmc:oxudnarmc filename
```

Описание параметров `setfaud` приведено в таблице 34.

Таблица 34

Параметр	Описание
`-s, --set`	Установить список регистрируемых событий из командной строки
`-b, --remove`	Удалить все элементы списка регистрируемых событий
`-m, --modify`	Изменить или добавить элементы списка из командной строки
`-d, --default`	Работать со списком регистрируемых событий по умолчанию
`-S, --set-file`	Установить список регистрируемых событий из файла
`-X, --remove-all`	Удалить все списки регистрируемых событий
`-M, --modify-file`	Изменить или добавить элементы списка регистрируемых событий из файла
`-B, --restore`	Восстановить атрибуты из файла
`-R, --recursive`	Для подкаталога 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

Список регистрируемых событий, а также описание команды, приведено в справочной странице `man setfaud`.

6.4.4. useraud

Команда `useraud` позволяет просматривать и изменять правила регистрации событий для пользователей.

Синтаксис команды:

```
useraud [<параметр> [<параметр>...]] [имя_пользователя(группы)]
     [флаги_регистрации]
```

Флаги регистрации задаются в виде:

```
<флаги_успешных_операций>[[:<флаги_неуспешных_операций>], ...]
```

При этом флаги операций могут иметь вид:

1) `<+|-><имя_регистрируемого_события>,...`

Пример

Регистрация успешного события выполнения файла:

```
+exec
```

Регистрация неуспешного события открытия файла:

```
-open
```

2) <сокращенное_имя_регистрируемого_события>,...

Пример

Регистрация успешных событий открытия и удаления файла:

```
ou
```

Описание параметров useraud приведено в таблице 35.

Таблица 35

Параметр	Описание
-d, --delete	Сбросить правила регистрации событий
-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

Описание команды и список регистрируемых событий приведены в справочной странице man useraud.

6.4.5. psaud

Правила регистрации событий наследуются порожденными процессами. Команда psaud позволяет изменить или считать правила регистрации событий выбранного процесса.

Только администратор может устанавливать и считывать правила регистрации событий аудита процессов.

Синтаксис команды:

```
psaud [<параметр> [<параметр>...]] [<флаги_регистрации>]
```

Если параметр <флаги_регистрации> присутствует, то команда устанавливает указанные флаги регистрации событий на процесс. Если флаги регистрации событий не указаны, то команда выполняет их считывание с процесса, заданного параметром (идентификатор процесса).

Флаги регистрации задаются в виде:

```
<флаги_успешных_операций>[[:<флаги_неуспешных_операций>], ...]
```

При этом флаги операций могут иметь вид:

1) <+|-><имя_регистрируемого_события>,...

Пример

Регистрация успешного события выполнения файла:

```
+exec
```

Регистрация неуспешного события открытия файла:

```
-open
```

2) <сокращенное_имя_регистрируемого_события>,...

Пример

Регистрация успешных событий открытия и удаления файла:

```
ou
```

Описание параметров psaud приведено в таблице 36.

Таблица 36

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

Список событий можно просмотреть на странице помощи команды psaud -h.

Описание команды приведено в справочной странице man psaud.

6.4.6. ausearch

Команда ausearch предназначена для просмотра файлов журнала регистрации событий ядра, а также событий пользователя.

Синтаксис команды:

```
ausearch [<параметр> [<параметр>...]]
```

Описание параметров ausearch приведено в таблице 37.

Таблица 37

Параметр	Описание
-a, --event	Поиск событий с заданным идентификатором события. Сообщения обычно начинаются с записи вида msg=audit(1116360555.329:2401771). Идентификатор события — число после «:». Все события аудита, связанные с одним системным вызовом, имеют одинаковый идентификатор
-c, --comm	Искать события с заданным именем исполняемого файла задачи
--debug	Вывести информацию о некорректных событиях, не попавших в стандартный вывод ошибок
-e, --exit	Искать события по заданному коду завершения системного вызова или коду последней ошибки

Продолжение таблицы 37

Параметр	Описание
-f, --file	Искать события с заданным именем файла
-ga, --gid-all	Искать события с заданным эффективным или обычным идентификатором группы
-ge, --gid-effective	Искать события с заданным эффективным идентификатором группы или именем группы
-gi, --gid	Искать события с заданным идентификатором группы или именем группы
-h, --help	Вывести справку
-hn, --host	Искать события с заданным именем узла. Имя узла может быть именем узла, полным доменным именем или сетевым адресом
-i, --interpret	Транслировать числовые значения в текстовые. Например, идентификатор пользователя будет транслирован в имя пользователя. Трансляция выполняется с использованием данных с той машины, где запущен ausearch
-if, --input	Использовать указанный файл или каталог вместо журнала аудита. Может быть полезно при анализе журнала аудита с другой машины или при анализе частично сохраненного журнала
--input-logs	Использовать расположение файла журнала, указанное в auditd.conf, в качестве входных данных для поиска. Это необходимо при использовании ausearch в задаче cron
--just-one	Искать до первого совпадения
-k, --key	Искать события с заданным ключевым словом
-l, --line-buffered	Очистить вывод каждой строки. Может быть полезно, когда стандартный вывод конвейерно перенаправляется и использование стратегии буферизации блоков по умолчанию нежелательно
-m, --message	Искать события с заданным типом. Возможно указать список значений, разделенных запятыми. Для получения всех сообщений системы аудита указать тип ALL. Если указать данный параметр без значения, будет выведен список допустимых типов. Тип сообщения может быть строкой или числом
-n, --node	Вывести события с указанной машины. Возможно указать несколько машин, тогда выводятся события для каждой из указанных машин
-o, --object	Искать события с заданным контекстом (объектом)
-p, --pid	Искать события с заданным идентификатором процесса
-pp, --ppid	Искать события с заданным идентификатором родительского процесса
-r, --raw	Необработанный вывод. Используется для извлечения записей для дальнейшего анализа
-sc, --syscall	Искать события с заданным системным вызовом. Можно указать его номер или имя. Если указано имя, оно будет проверено на машине, где запущен ausearch
-se, --context	Искать события с заданным контекстом (scontext/subject или tcontext/object)

Окончание таблицы 37

Параметр	Описание
-su, --subject	Искать события с заданным контекстом scontext (subject)
-sv, --success	Искать события с заданным флагом успешного выполнения. Допустимые значения: yes (успешно) и no (неуспешно)
-te, --end	Искать события, которые произошли в указанное время или раньше. Формат даты и времени зависит от региональных настроек. Если дата не указана, то подразумевается текущий день (today). Если не указано время, то подразумевается текущий момент (now). Используется 24-часовая нотация времени. Например, дата может быть задана как 10/24/2005, а время — 18:00:00
-ts, --start	Искать события, которые произошли в указанное время или позже. Формат даты и времени зависит от региональных настроек. Если дата не указана, то подразумевается текущий день (today). Если не указано время, то подразумевается полночь (midnight). Используется 24-часовая нотация времени. Например, дата может быть задана как 10/24/2005, а время — 18:00:00
-tm, --terminal	Искать события с заданным терминалом. Некоторые демоны (например, cron и atd) используют имя демона как имя терминала
-ua, --uid-all	Искать события, у которых идентификатор пользователя, эффективный идентификатор пользователя или loginuid (auid) совпадают с заданным идентификатором пользователя
-ue, --uid-effective	Искать события с заданным эффективным идентификатором пользователя
-ui, --uid	Искать события с заданным идентификатором пользователя
-ul, --loginuid	Искать события с заданным идентификатором пользователя. Все программы, которые его используют, должны использовать pam_loginuid
-uu, --uuid	Искать события с заданным uuid
-v, --verbose	Показать версию и выйти
-vm, --vm-name	Совпадение с полным словом. Поддерживается для имени файла, имени узла, терминала и контекста
-w, --word	Поиск по заданной строке с полным совпадением. Поддерживается для имени файла, имени узла, терминала и контекста
-x, --executable	Искать события с заданным именем исполняемой программы

Описание команды приведено в справочной странице man ausearch.

Примеры:

1. Отобразить события процессов и пользователей

```
ausearch -i -ts "11:35:35" -te "11:36:00" -k parsec-p
```

2. Регистрация событий с файлами

```
ausearch -i -ts "11:35:35" -te "11:36:00" -k parsec-f
```

3. Протоколирование событий, связанных с мандатным управлением доступом и аудитом

```
ausearch -i -ts "11:35:35" -te "11:36:00" -m avc
```

4. Протоколирование событий, приходящих от пользователя

```
ausearch -i -ts "11:35:35" -te "11:36:00" -m user_avc
```

6.4.7. Дополнительные параметры регистрации событий

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

1) отключение регистрации 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 регистрация системных вызовов, не используемых для мандатного управления доступом, отключена;

2) отключение регистрации 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 регистрация всех системных вызовов отключена;

3) отключение регистрации запретов доступа (на основе мандатного управления доступом):

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

Если вывод команды:

```
cat /parsecfs/disable-denied-audit
```

равен `1`, то регистрация запретов доступа отключена.

6.5. Подсистема регистрации событий

6.5.1. Регистрация событий и уведомление о событиях

В ОС реализована подсистема регистрации событий, которая собирает информацию о событиях, в том числе о событиях безопасности, из различных источников и предоставляет инструменты для просмотра собранных данных и реагирования на события.

Подсистема регистрации событий включает следующие основные компоненты:

1) менеджер и маршрутизатор событий `syslog-ng` — основная служба подсистемы регистрации событий, которая реализована в соответствии со стандартом Syslog. Служба `syslog-ng` принимает информацию о событиях из различных источников (события от `auditd`, собственные подключаемые модули, файлы, прикладное ПО и др.), выполняет фильтрацию и обработку полученных данных, регистрирует события

в журнал, а также, в зависимости от конфигурации, сохраняет в файл, отправляет по сети и т.д.;

2) модуль `syslog-ng-mod-astra` — модуль для `syslog-ng`, выполняющий дополнительную обработку и фильтрацию событий;

3) `astra-event-watcher` — демон уведомления пользователя о событиях, обработанных менеджером `syslog-ng`;

4) `astra-event-diagnostics` — инструмент диагностики подсистемы регистрации событий, описание инструмента приведено в 6.5.3.

Работа модуля `syslog-ng-mod-astra` настраивается в следующих конфигурационных файлах:

1) `/etc/astra-syslog.conf` — список регистрируемых событий;

2) `/usr/share/syslog-ng-mod-astra/event-settings/` — каталог с файлами настроек по умолчанию для каждого события.

Для настройки регистрируемых событий используются:

1) графическая утилита `fly-admin-events` («Настройка регистрации системных событий»), описание утилиты приведено в электронной справке;

2) инструмент командной строки `astra-admin-events`.

Для просмотра с помощью инструмента `astra-admin-events` списка регистрируемых событий необходимо выполнить команду без параметров:

```
astra-admin-events
```

В выводе списка событий значение «`1`» — событие регистрируется, значение «`0`» — событие не регистрируется. Порядок использования инструмента приведен на странице помощи:

```
astra-admin-events -h
```

Модуль `syslog-ng-mod-astra` информацию о событиях регистрирует в следующих файлах:

1) `/parsec/log/astra/events` — журнал событий в формате JSON, описание журнала приведено в 6.5.2;

2) `/var/log/astra/prevlogin-<имя_пользователя>` — журнал в формате JSON сводной статистики предыдущих входов в систему пользователя `<имя_пользователя>`. Включает данные о последней завершенной сессии данного пользователя, а также количество успешных и неуспешных входов данного пользователя со времени начала ведения статистики. Доступен для чтения только пользователю `<имя_пользователя>`.

Параметры уведомлений демона `astra-event-watcher` определены в файле `/usr/share/knotifications5/astra-event-watcher.notifyrc`. Параметры уве-

домлений для определенного пользователя могут быть переопределены в файле /home/<имя_пользователя>/.config/astra-event-watcher.notifyrc.

Информирование (оповещение) о событиях безопасности осуществляется с помощью графической утилиты fly-notifications («Центр уведомлений»), описание утилиты приведено в электронной справке.

6.5.2. Журнал событий

Служба syslog-ng выполняет регистрацию событий в журнал /parsec/log/astra/events. В журнале событий регистрируются попытки запуска неподписанных файлов, успешная и неуспешная авторизация, данные о пользовательских сессиях и другие события безопасности, регистрация которых настроена (см. 6.5.1).

Также в журнале событий регистрируются действия с журналом аудита /var/log/audit службы auditd (см. 6.3) — удаление, переименование, перемещение файла журнала аудита.

Файл журнала событий доступен для чтения только пользователям из группы astra-audit.

Файл журнала событий доступен для чтения и записи в конец файла только администратору. Разрешение на запись в конец файла задано установкой на файл атрибута a (chattr +a).

При включенном МКЦ файл журнала событий имеет высокий уровень целостности. Процесс, осуществляющий регистрацию событий в файле журнала, также имеет высокий уровень целостности. Таким образом, изменение файла журнала событий доступно только администратору на высоком уровне целостности.

Для просмотра журнала событий может использоваться:

1) графическая утилита fly-event-viewer («Журнал системных событий»), описание утилиты приведено в электронной справке;

2) инструмент командной строки astra-event-viewer, порядок использования инструмента приведен на странице помощи:

```
astra-event-viewer -h
```

Действия с журналом событий (удаление, переименование, перемещение, ротация файла журнала событий) регистрируются подсистемой регистрации событий и указываются первой записью в журнале событий:

- удаление журнала событий регистрируется событием «Журнал событий удален»;
- переименование или перемещение журнала событий регистрируется событием «Журнал событий переименован или перемещен»;
- ротация журнала событий регистрируется событием «Журнал событий ротирован»;

- действия с журналом событий недоверенными процессами (всеми процессами, кроме процессов `syslog-ng` и `logrotate`) регистрируются событием «Журнал событий изменен недоверенным процессом».

Также действия с журналом событий регистрируются службой `auditd` и указываются в журнале аудита `/var/log/audit` (см. 6.3).

6.5.3. Самодиагностика подсистемы регистрации событий

Инструмент `astra-event-diagnostics` обеспечивает самодиагностику подсистемы регистрации событий.

Самодиагностика выполняется:

1) автоматически при загрузке ОС после запуска служб `syslog-ng` и `auditd`;
2) периодически в процессе функционирования ОС, периодичность самодиагностики настраивается в юните `astra-event-diagnostics.timer`.

Также провести диагностику возможно выполнив команду:

```
sudo astra-event-diagnostics -v
```

При выполнении самодиагностики производятся следующие проверки:

1) запуск и работоспособность служб `syslog-ng` и `auditd`;
2) наличие необходимых модулей из состава `syslog-ng-mod-astra`;
3) корректность конфигурационных файлов:
   а) `/etc/audit/rules.d/astra-syslog.rules`;
   б) `/etc/astra-syslog.conf`;
   в) `/etc/syslog-ng/conf.d/mod-astra.conf`;
   г) некоторых файлов из `/usr/share/syslog-ng-mod-astra/event-settings`.

Возможно настроить автоматическое блокирование учетной записи пользователя, если в результате самодиагностики выявлены ошибки. Для этого необходимо запустить команду самодиагностики с параметром `-b`:

```
sudo astra-event-diagnostics -b
```

Описание параметров команды приведено на странице помощи:

```
sudo astra-event-diagnostics -h
```

Разблокировка учетной записи, заблокированной после выявления ошибок при самодиагностики, может быть выполнена в графической утилите `fly-admin-smc` («Политика безопасности»).

Регистрация событий о результатах самодиагностики осуществляется:

1) в журнал `/parsec/log/astra/astra-event-diagnostics`;
2) в журнал `/var/log/syslog`;
3) в журнал событий `/parsec/log/astra/events` (в случае, если подсистема регистрации событий частично работоспособна);

4) с использованием `astra-sosreport`.

Если в результате самодиагностики были выявлены ошибки в работе подсистемы регистрации событий, то события о результатах регистрируются с уровнем важности «Критический».

Уведомление о результатах самодиагностики обеспечивает демон `astra-event-watcher`.

Информирование (оповещение) о событиях осуществляется с помощью утилиты `fly-notifications` («Центр уведомлений»). Описание утилиты приведено в электронной справке.

6.6. Регистрация событий в СУБД PostgreSQL

6.6.1. Режимы регистрации событий

В СУБД PostgreSQL для настройки режима регистрации событий используется конфигурационный параметр `ac_audit_mode` файла `postgresql.conf`. 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 регистрации событий используется внешний файл `pg_audit.conf`;
- `external,internal` — смешанный режим. Настройки регистрации событий берутся сначала из внешнего файла `pg_audit.conf`, после чего дополняются настройками из таблицы `pg_db_role_settings`;
- `internal,external` — смешанный режим. Настройки регистрации событий берутся сначала из таблицы `pg_db_role_settings`, после чего дополняются настройками из внешнего файла `pg_audit.conf`;
- `none` — регистрация событий отключена.

6.6.2. Настройка маски регистрации событий

В СУБД PostgreSQL версии 11 маска регистрации событий устанавливается в процессе авторизации пользователя согласно выбранному режиму 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 текстовом виде.

В таблице 38 приведено соответствие между событиями, буквенным и шестнадцатеричным значением маски регистрации событий.

Таблица 38

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 регистрации событий
Зарезервированный символ	O	0	Регистрация событий отключена

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 регистрируемых событий 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.

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)
```

6.6.3. 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 регистрации 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}';
```

6.6.4. Назначение списков регистрации событий в режиме external

Для назначения маски событий в режиме external используется конфигурационный файл pg_audit.conf конкретного кластера данных, который имеет следующий формат:

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

Пример

Файл pg_audit.conf

- аудит действий администратора СУБД:

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

- для пользователя any выполнять регистрацию только неуспешных действий:

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

- 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 конфигурационном файле задаются списки успешных (success events mask) и неуспешных (failure events mask) типов запросов на доступ, которые будут регистрироваться в журнале СУБД и журнале 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. Типы запросов и их описание приведены в таблице 39.

Таблица 39

Тип запроса	Описание	Бит	Шестнадцатеричное значение
SUBJECT	Добавление/изменение/удаление пользователей и групп	0	1

Окончание таблицы 39

Тип запроса	Описание	Бит	Шестнадцатеричное значение
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.

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

6.6.5. 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 6.6.3.

Аналогично и для режима internal,external.

6.6.6. Назначение списков регистрации событий в режиме none

В режиме `none` регистрация событий отключена, однако, администратор может изменять маску регистрации событий с помощью команд из 6.6.3.

6.7. Средства централизованного аудита и протоколирования

В состав ОС входит программное средство Zabbix, обеспечивающее функционал для настройки, сбора данных и мониторинга состояния сети, а также жизнеспособности и целостности ресурсов сети. Описание установки и настройки Zabbix приведено в документе РУСБ.10015-37 95 01-1.

7. ИЗОЛЯЦИЯ ПРОЦЕССОВ

7.1. Изоляция процессов ОС

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.

7.2. Создание и защита изолированных программных сред (контейнеров)

ОС содержит программные средства (средства контейнеризации), реализующие создание и функционирование изолированных программных сред, с обеспечением выполнения следующих функций безопасности:

- изоляция контейнеров;
- выявление уязвимостей в образах контейнеров;
- проверка корректности конфигурации контейнеров;
- контроль целостности контейнеров и их образов;
- регистрация событий безопасности;
- идентификация и аутентификация пользователей.

Средства контейнеризации реализуют функциональные возможности по созданию образов контейнеров, формированию среды выполнения контейнеров и обеспечения выполнения их процессов, запуску контейнера и управление данным контейнером.

В состав ОС входит программное обеспечение Docker для автоматизации развертывания и управления приложениями в средах с поддержкой контейнеризации. Описание установки, настройки и работы с контейнерами Docker приведено в РУСБ.10015-37 95 01-1.

7.2.1. Изоляция контейнеров и пространств

В ОС реализованы механизмы изоляции уровня ядра Linux, описание которых приведено в 7.1. Механизмы изоляции реализуют функции изоляции контейнеров, включая изоляцию пространств идентификаторов процессов контейнеров и пространств имен.

Изоляция доступа контейнеров к ресурсам аппаратной платформы достигается путем применения механизма групп управления `control groups` (`cgroups`), используемого для ограничения в ресурсах группы процессов, выполняющихся в контейнере, а также для ограничения других групп, например групп процессов, принадлежащих какой-либо службе или пользовательскому сеансу.

Различают следующие контроллеры:

1) `cpu` — накладывает ограничения на долю процессорного времени (cpu share);
2) `cpuacct` — подсчитывает потребленные ресурсы процессора;
3) `cpuset` — привязывает процессы к конкретным процессорам;
4) `memory` — ограничивает потребление памяти;
5) `devices` — ограничивает доступ к файлам устройств;
6) `freezer` — позволяет приостанавливать и возобновлять работу групп процессов;
7) `net_cls` и `net_prio` — назначают сетевому трафику процессов класс и приоритет обработки для использования сетевыми фильтрами и планировщиками;
8) `blkio` — ограничивает количество запросов к блочным (дисковым) устройствам при помощи планировщика CFQ;
9) `pids` — накладывает ограничение на количество процессов в группе.

Изоляция доступа контейнеров к системным вызовам ядра хостовой операционной системы реализуется механизмом ядра Linux SecComp, позволяющим процессам определять системные вызовы, которые они будут использовать. Если злоумышленник получит возможность выполнить произвольный код, SecComp не даст ему использовать системные вызовы, которые не были заранее объявлены.

Дополнительно для изоляции среды функционирования контейнера в ОС предусмотрена возможность запуска контейнера Docker на пониженном уровне целостности 2.

По умолчанию функция запуска контейнеров на пониженном уровне МКЦ выключена. Управление функцией производится с помощью инструмента командной строки `astra-docker-isolation`, описание которого приведено в 16.5.14.

Также в ОС сохранен для совместимости инструмент командой строки `docker-isolation`. Для включения функции запуска контейнеров на пониженном уровне МКЦ с использованием этого инструмента следует выполнить:

```
sudo /usr/share/docker.io/contrib/parsec/docker-isolation on
sudo systemctl restart containerd
sudo systemctl restart docker
```

Для выключения функции следует выполнить:

```
sudo /usr/share/docker.io/contrib/parsec/docker-isolation off
sudo systemctl restart containerd
sudo systemctl restart docker
```

Работа с контейнерами Docker осуществляется в соответствии с описанием в РУСБ.10015-37 95 01-1.

7.2.2. Выявление уязвимостей в образах контейнеров

В состав ОС входит программное обеспечение OpenSCAP и пакет `oscap-docker`, обеспечивающие выявление уязвимостей в образах контейнеров. OpenSCAP использует библиотеку `libopenscap` для сканирования. Сканирование уязвимостей выполняется с использованием заранее подготовленного файла базы данных уязвимостей.

Проверка образов и контейнеров на наличие уязвимостей выполняется автоматически при следующих событиях:

- создание образа из Dockerfile или из контейнера;
- загрузка образа из архива или потока ввода;
- создание файловой системы образа из архива;
- скачивание образа из реестра;
- запуск или перезапуск контейнера.

Периодическая проверка образов контейнеров на наличие известных уязвимостей осуществляется автоматически один раз в неделю.

При обнаружении уязвимостей дальнейшее использование образа контейнера запрещено. Соответствующее ограничение по эксплуатации приведено в 17.2.

Для блокировки запуска контейнера, в образе которого обнаружена уязвимость, применяется глобальный параметр `astra-sec-level` в конфигурационном файле Docker. В качестве значения параметра задается число от 1 до 6, которое определяет класс защиты:

1) 1-5 классы защиты — при обнаружении уязвимости в контейнере его запуск блокируется;

2) 6 класс защиты — отладочный режим, при обнаружении уязвимости в контейнере выводится соответствующее предупреждение, при этом запуск контейнера не блокируется.

Задать класс защиты можно одним из способов:

1) добавить в конфигурационный файл `/etc/docker/daemon.json` строку:

```
{
"debug" : true,
"astra-sec-level" : <класс_защиты>
}
```

Пример

```
{
"debug" : true,
"astra-sec-level" : 4
}
```

2) запуск `dockerd` с параметром `astra-sel-level`, в качестве значения которого указан класс защиты:

```
--astra-sec-level <класс_защиты>
```

Пример

```
sudo dockerd --astra-sec-level 2
```

Контейнер, в котором была обнаружена уязвимость, может быть запущен в режиме отладки (6 класс защиты) для устранения уязвимости и последующего запуска без блокировки с более высоким классом защиты.

В случае если класс защиты не задан или задан не из диапазона 1-6, то при обнаружении уязвимости в контейнере автоматически задается 1 класс защиты с выводом соответствующего сообщения в журнал и запуск контейнера блокируется.

События безопасности, связанные с выявлением уязвимостей в контейнере, регистрируются в соответствии с описанием 7.2.5.

7.2.3. Обеспечение корректности конфигурации контейнеров

Обеспечение корректности конфигурации контейнеров, в том числе ограничение прав прикладного программного обеспечения, выполняемого внутри контейнера, на использование периферийных устройств, вычислительных ресурсов хостовой операционной системы, устройств хранения данных и съемных машинных носителей информации (блочных устройств); а также монтирование корневой файловой системы хостовой операционной системы в режиме «только для чтения» и ограничение доступа прикладного программного обеспечения, выполняемого внутри контейнера, к системным вызовам ядра хостовой операционной системы, осуществляется средствами Docker.

Для ограничения прав прикладного программного обеспечения, выполняемого внутри контейнера, на использование вычислительных ресурсов (оперативной памяти, операций

ввода-вывода за период времени) хостовой операционной системы необходимо установить ограничение на объем памяти путем последовательного выполнения следующих команд:

1) установить ограничение:

```
sudo docker run -it --memory=<объем>m ubuntu bash
```

где <объем>m — объем памяти в МБ;

2) ограничить операции ввода-вывода:

```
sudo docker run -it --memory=<объем>m --device-write-bps
/dev/sda:<объем>mb ubuntu bash
```

Для настройки монтирования корневой файловой системы хостовой операционной системы в режиме «только для чтения» необходимо запустить `docker` с параметром `--read only`:

```
sudo docker run -it --read-only ubuntu bash
```

Доступ к системным вызовам ядра хостовой операционной системы реализуется ограничением запуска (блокировкой вызовов):

```
{
"defaultAction": "SCMP_ACT_ERRNO",
"architectures": [
"SCMP_ARCH_X86_64",
"SCMP_ARCH_X86",
"SCMP_ARCH_X32"
],
"syscalls": [
]
}
```

7.2.4. Контроль целостности контейнеров и их образов

Контроль целостности объектов контроля (образов контейнеров и исполняемых файлов контейнеров, параметров настройки средства контейнеризации) реализуется сертифицированными средствами регламентного контроля целостности AFICK из состава ОС (см. 9.4).

Для контроля целостности образов контейнеров и исполняемых файлов контейнеров необходимо получить образ контейнера и поставить на контроль целостности каталог с образами и каталог с кэшем, добавив в конфигурационный файл `/etc/afick.conf` в секции `file section` следующие строки:

```
##############
# file section
##############
...
```

```
/var/lib/docker/image ETC
/var/lib/docker/overlay2 ETC
...
```

Для контроля целостности параметров настройки средства контейнеризации необходимо в конфигурационный файл `/etc/afick.conf` в секции `file section` добавить файл `/etc/docker/daemon.json` для контроля `afick`:

```
##############
# file section
##############
...
/etc/docker/ ETC
...
```

Системный планировщик заданий `cron` позволяет выполнять проверку целостности объектов контроля в процессе загрузки операционной системы.

Локальное реагирование на события в автозапуске для сессии пользователя обеспечивается демоном `astra-event-watcher` из состава подсистемы регистрации событий и выполняет заданное действие, например уведомление (см. 6.5). Передача данных выполняется с использованием D-Bus. Реализация действий осуществляется модулем KNotifications. Демон получает информацию только о событиях, для которых настроено реагирование. Демон, запущенный от имени определенного пользователя, реагирует только на те события, которые настроены для данного пользователя. Для этого демон отслеживает сигнал, передаваемый `syslog-ng` по D-Bus всем заинтересованным приложениям. В сигнале также передается список имен пользователей и имен групп, для которых требуется реакция на событие.

Регистрация событий безопасности и оповещение администратора безопасности средства виртуализации о событиях безопасности осуществляется подсистемой регистрации событий (см. 6.5).

Целостность образов контейнеров и параметров настройки средства контейнеризации также контролируется путем применения цифровой подписи — создания замкнутой программной среды в соответствии с 16.1. Для выполнения контроля необходимо от имени учетной записи администратора:

1) настроить замкнутую программную среду (см. 16.1);
2) получить идентификатор контейнера:

```
root@astra-test:/var/lib/docker# docker inspect nginx | grep Id
```

3) подписать файл конфигурации образа:

```
/var/lib/docker# bsign --sign --xattr
```

`./image/overlay2/imagedb/content/<id>`

где <id> — идентификатор контейнера;

4) добавить имя контролируемого файла в `/etc/digsig/xattr_control`:

`sudo echo "var/lib/docker/image/overlay2/imagedb/content/<id>`

где <id> — идентификатор контейнера;

5) выполнить команду:

`sudo update-initramfs -u -k all`

6) выполнить перезагрузку.

При нарушении целостности запуск образа контейнера блокируется.

7.2.5. Регистрация событий безопасности, связанных с контейнерами

Регистрация событий безопасности, связанных с образами и контейнерами, осуществляется подсистемой регистрации событий (см. 6.5), а также штатными средствами в системные журналы. Событиям безопасности, связанным с образами и контейнерами, присваиваются метки `dockerd_audit`.

При регистрации событий, связанных с образами и контейнерами, указывается UID и имя пользователя ОС, от имени которого выполнено данное действие. Если действие было выполнено удаленно, то вместо UID и имени пользователя указывается IP-адрес компьютера, с которого выполнено действие. Если инициатор действия в той же ОС, что и служба `dockerd`, но не удалось определить UID пользователя, то вместо UID и имени пользователя указывается символ «@». Также для события указывается результат — успешно или неуспешно. В случае неуспешного события указывается причина ошибки.

События аутентификации пользователя, в т.ч. неуспешные попытки аутентификации, регистрируются:

1) в журнале подсистемы регистрации событий (см. 6.5);

2) в журнале `/var/log/auth.log`.

Регистрация событий безопасности, связанных с выявлением уязвимостей в образе и в контейнере, осуществляется в журнал подсистемы регистрации событий (см. 6.5), а также в системные журналы `/var/log/daemon.log` и `/var/log/syslog.log`. Регистрируются следующие события:

1) запуск сканирования (start) с указанием пути и ID сканируемого контейнера/образа;

2) результат сканирования:

   а) если уязвимость не обнаружена, то сообщение об окончании сканирования (complete) с указанием пути и ID сканируемого контейнера/образа;

   б) если уязвимость обнаружена, то сообщение о неуспешном сканировании (fail) с указанием версии ПО, операционной системы, обнаруженной уязвимо-

сти, адреса отчета сканирования. Если запуск контейнера был заблокирован (класс защиты 1-5), то также регистрируется сообщение о блокировке запуска контейнера.

Регистрация действий с образами и контейнерами (события создания, модификации и удаления образов контейнеров; получения доступа к образам контейнеров; запуска и остановки контейнеров с указанием причины остановки; модификации запускаемых контейнеров) осуществляется в журнале подсистемы регистрации событий (см. 6.5) и в журнале `/var/log/syslog.log`.

Настройка событий, которые подлежат регистрации, и реагирование на них системы осуществляется с помощью утилиты `fly-admin-events` («Настройка регистрации системных событий»). Оповещение администратора безопасности средства контейнеризации о событиях безопасности осуществляется с использованием утилиты `fly-notifications` («Центр уведомлений»). Описание утилит приведено в электронной справке.

7.2.6. Идентификация и аутентификация пользователей

Первичная идентификация пользователей средства контейнеризации осуществляется администратором безопасности средства контейнеризации. Идентификация и аутентификация пользователей в средстве контейнеризации осуществляется с учетом требований разделов ГОСТ Р 58833-2020. При входе в систему осуществляться идентификация и проверка подлинности субъектов доступа процедурой аутентификации. Описание процедуры приведено в разделе 2.

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

Установка пароля пользователя осуществляется администратором с помощью графической утилиты `fly-admin-smc` («Политика безопасности»). Описание утилиты приведено в электронной справке.

Пароль пользователя должен содержать не менее 8 символов при алфавите пароля не менее 70 символов. Максимальное количество неуспешных попыток аутентификации (ввода неправильного пароля) до блокировки — 4.

Запрет запуска средством контейнеризации в хостовой операционной системе процессов, обладающих привилегиями администратора информационной (автоматизированной) системы и администратора безопасности информационной (автоматизированной) системы, осуществляется с помощью утилиты `unshare`, позволяющей реализовать:

1) запрет запуск процессов от имени суперпользователя `root`;
2) запрет просмотра процессов хостовой машины;

3) невозможность какого-либо влияния на процессы хостовой машины и других пространств;

4) невлияние монтирования и размонтирования ФС на состав древа каталогов хостовой машины.

7.2.7. Работа с Docker в непривилегированном режиме с ненулевыми метками безопасности

7.2.7.1. Принцип функционирования

Контейнеры Docker в непривилегированном (rootless) режиме (описание режима приведено в РУСБ.10015-37 95 01-1) могут быть запущены от имени любого непривилегированного пользователя с ненулевой меткой безопасности контейнера.

Метка безопасности контейнера всегда задается четырьмя десятичными неотрицательными числами, разделенными двоеточием:

`0:63:0:0`

где первое число — иерархический уровень конфиденциальности;

второе число — иерархический уровень целостности;

третье число — неиерархические категории конфиденциальности;

четвертое число — зарезервировано для задания атрибутов МРД для файловых объектов, и в работе с контейнерами не применяется (следует всегда использовать значение `0`).

Для контейнера в непривилегированном режиме с ненулевой меткой безопасности применяются следующие ограничения:

1) запуск процессов внутри контейнера возможен только с одинаковой для всех процессов меткой безопасности, равной метке безопасности контейнера (задается при запуске контейнера);

2) в метке безопасности контейнера ненулевой может быть либо только классификационная метка, либо только метка целостности (при этом отрицательная метка целостности также является ненулевой меткой);

3) для системных файлов внутри контейнера (исполняемых, конфигурационных и т.д.) всегда используется ненулевой уровень целостности (например, `0:2:0:0` или `0:63:0:0`), и, соответственно, всегда используется нулевая классификационная метка;

4) внутри контейнера, запущенного в непривилегированном режиме, API PARSEC не работает;

5) если внутри контейнера необходима работа с файлами с ненулевой классификационной меткой, то данные файлы должны группироваться в специально созданных каталогах:

`/home/.pdp/<имя_пользователя>/<уровень_конфиденциальности>:<иерархический_уровень_целостности>:<категории_конфиденциальности>:0`

При этом:

а) метки безопасности таких каталогов могут сочетать ненулевые классификационные метки и ненулевые метки целостности;

б) при работе в расширенном режиме МКЦ (strict mode) метки целостности создаваемых внутри этих каталогов файловых объектов наследуют значение метки целостности родительского каталога.

Для хостовой ОС контейнер, запущенный в непривилегированном режиме, является группой процессов:

1) имеющих метку безопасности субъекта, запустившего контейнер;

2) не имеющих прав администратора;

3) работающих с набором файловых объектов, расположенных в файловой системе контейнера и имеющих собственные метки безопасности.

При этом процессы внутри контейнера:

1) запущены от имени администратора;

2) имеют неограниченный доступ к объектам файловой системы контейнера (за исключением возможности изменять метки безопасности).

Файловые объекты в файловой системе контейнера (rootFS) создаются с меткой безопасности, в которой:

1) классификационная метка равна классификационной метке субъекта, запустившего контейнер;

2) метка целостности всегда нулевая.

При этом к процессам (субъектам) внутри контейнера, запущенного в непривилегированном режиме, будут применяться общие правила МРД и МКЦ хостовой ОС. Например, процессы контейнера в непривилегированном режиме, имеющего метку безопасности `1:0:0:0`, смогут изменять файлы с меткой безопасности `1:0:0:0`, читать содержимое файлов с меткой безопасности `1:0:0:0` или `0:0:0:0`).

Процессы (субъекты), работающие внутри контейнера в непривилегированном режиме, не могут изменять метки безопасности файловых объектов в файловой системе своего контейнера, т.к. для контейнеров в непривилегированном режиме нет возможности запуска с привилегиями (`docker run --privileged`) и, соответственно, нет доступа к программному интерфейсу Parsec (Parsec API) через ParsecFS.

Описание работы с контейнерами Docker в непривилегированном режиме с ненулевыми метками безопасности также приведено в `man rootless-helper-astra` `man rootlessenv`.

7.2.7.2. Управление запуском контейнера с ненулевой меткой безопасности

Для предоставления возможности работать с контейнерами в непривилегированном режиме от имени пользователя с ненулевой меткой безопасности следует запустить для этого пользователя непривилегированную службу Docker с указанием соответствующей метки безопасности.

Запуск службы для работы с ненулевой классификационной меткой выполняется командой:

```
sudo systemctl start rootless-docker@$(systemd-escape <имя_пользователя>@
<метка_безопасности>)
```

Запуск службы для работы с различными классификационными метками выполняется командой:

```
sudo systemctl start rootless-docker@$(systemd-escape <имя_пользователя>@
0:0:0:0@privsock)
```

при этом использование флага `privsock` запускает непривилегированную службу Docker с привилегией `PARSEC_CAP_PRIV_SOCK`, позволяющей выполнять команды для сетевых подключений (например, `docker pull`) игнорируя мандатные ограничения (см. 4.7).

Для настройки автоматического запуска службы после перезагрузки можно выполнить команду:

```
sudo systemctl enable rootless-docker@$(systemd-escape <имя_пользователя>@
<метка_безопасности>@privsock)
```

После запуска для пользователя непривилегированной службы Docker возможно выполнение команд Docker от имени данного пользователя (например, копирование образов в пользовательские репозитории образов или запуск контейнеров).

7.2.7.3. Копирование образа в репозиторий пользователя

Для запуска контейнера пользователем в сессии с ненулевой классификационной меткой требуется создать пользовательский репозиторий с соответствующей меткой, содержащий образы.

Для создания пользовательских копий образов Docker необходимо:

1) экспортировать образ:

```
rootlessenv docker save -o /tmp/<имя_архива>.tar <имя_образа>
```

Экспорт рекомендуется выполнять в каталог `/tmp/`, доступный для чтения всем пользователям;

2) разрешить чтение экспортированного образа:

а) всем пользователям:

```
chmod o+r /tmp/<имя_архива>.tar
```

б) только указанным пользователям:

```
setfacl -m u:<имя_пользователя>:r /tmp/<имя_архива>.tar
```

3) импортировать образ с помощью инструмента pdp-exec, указав в команде имя пользователя с нужными метками безопасности (при этом должны быть запущены непривилегированные службы Docker для указанного пользователя с соответствующими метками безопасности):

```
sudo pdp-exec -u <имя_пользователя> -l <метка_безопасности>
-- rootlessenv docker load -i /tmp/<имя_архива>.tar
```

7.2.7.4. Выполнение команд и запуск контейнеров в непривилегированном режиме от имени пользователя

Команды Docker, выполняемые пользователем (или администратором от имени пользователя) выполняются с меткой безопасности сессии пользователя (или указанной администратором):

1) для запуска контейнера пользователем из своей сессии (контейнер будет запущен из репозитория образов пользователя, имеющего метку безопасности, равную метке безопасности сессии пользователя, и унаследует данную метку безопасности):

```
rootlessenv docker run <имя_образа>
```

2) для запуск администратором оболочки командой строки контейнера, из которой можно выполнять команды Docker от имени указанного пользователя, выполнить команду:

```
sudo pdp-exec -u <имя_пользователя> -l <метка_безопасности>
-- rootlessenv
```

3) для запуска контейнера администратором от имени указанного пользователя выполнить команду:

```
sudo pdp-exec -u <имя_пользователя> -l <метка_безопасности>
-- rootlessenv docker run --rm -ti <имя_образа>
```

Пользователь (администратор) может со стороны хостовой ОС присваивать метки безопасности файловым объектам в файловой системе контейнера (например, `1:0:0:0` для конфиденциальных пользовательских файлов и `0:2:0:0` для системных файлов контейнера с высокой целостностью).

Местонахождение файловой системы контейнера на хостовой ОС можно узнать командой:

```
docker inspect <имя_контейнера> | egrep "(Lower|Upper)Dir"
```

При работе администратора с контейнером пользователя данная команда должна выполняться от имени пользователя с указанием нужной метки безопасности:

```
sudo pdp-exec -u <имя_пользователя> -l <метка_безопасности>
-- rootlessenv docker inspect <имя_контейнера> | egrep "(Lower|Upper)Dir"
```

8. 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).

В ОС реализован механизм очистки освобождаемой внешней памяти. Механизм может быть включен при установке ОС путем выбора соответствующего пункта. После установки ОС включение механизма осуществляется в соответствии с 16.5.30.

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).

Реализованный в ОС механизм очистки освобождаемой внешней памяти очищает неиспользуемые блоки ФС непосредственно при их освобождении, а также очищает разделы страничного обмена. Работа данного механизма снижает скорость выполнения операций удаления и усечения размера файла.

Данные любых удаляемых/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/XFS) в одном из следующих режимов:

1) очистка осуществляется путем перезаписи каждого байта в освобождаемой области посредством четырех сигнатур следующего вида: `11111111`, `01010101`, `10101010`, `00000000`. Использование режима включается параметром `secdel` в конфигурационном файле `/etc/fstab` для раздела ФС, на котором требуется очищать блоки памяти при их освобождении (например, `/dev/sda1`). В список параметров монтирования добавляется параметр `secdel`.

Пример

```
/dev/sda1 /home ext4 acl,defaults,secdel 0 2
```

2) очистка осуществляется путем перезаписи каждого байта в освобождаемой области посредством четырех сигнатур следующего вида: `11111111`, `01010101`, `10101010`, `00000000`. Количество перезаписей определяется администратором. Использование режима включается установкой значения параметра `secdel` в конфигурационном файле `/etc/fstab` для раздела ФС, на котором требуется очищать

блоки памяти при их освобождении (например, /dev/sda1). При установке числа перезаписей больше четырех сигнатуры используются повторно. Например, при установке числа перезаписей, равного 6, последовательность сигнатур, используемых для перезаписи, имеет следующий вид: 11111111, 01010101, 10101010, 00000000, 11111111, 01010101. В список параметров монтирования добавляется параметр secdel=6.

Пример

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

3) очистка осуществляется путем перезаписи каждого байта в освобождаемой области посредством четырех псевдослучайных сигнатур. Использование режима включается параметром secdelrnd в конфигурационном файле /etc/fstab для раздела ФС, на котором требуется очищать блоки памяти при их освобождении (например, /dev/sda1). В список параметров монтирования добавляется параметр secdelrnd.

Пример

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

4) очистка осуществляется посредством перезаписи каждого байта в освобождаемой области посредством псевдослучайных сигнатур. Количество перезаписей определяется администратором. Использование режима включается установкой значения параметра secdelrnd в конфигурационном файле /etc/fstab для раздела ФС, на котором требуется очищать блоки памяти при их освобождении (например, /dev/sda1). Например, при установке числа перезаписей, равного 6, в список параметров монтирования добавляется параметр secdelrnd=6.

Пример

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

Включение механизма очистки блоков памяти при их освобождении в первом режиме может быть выполнено с использованием инструмента astra-secdel-control, описанного в 16.5.30.

Установка параметра монтирования для очистки блоков памяти при их освобождении может быть выполнена с использованием графической утилиты fly-admin-smc, запущенной администратором.

Для включения очистки активных разделов страничного обмена необходимо установить в конфигурационном файле /etc/parsec/swap_wiper.conf для параметра ENABLED значение Y.

Пример

```
ENABLED=Y
```

Включение очистки активных разделов страничного обмена может быть выполнено с использованием инструмента `astra-swapwiper-control`, описанного в 16.5.31.

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.

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.

Предотвращение выполнения произвольного кода обеспечивается путем контроля доступа к страницам памяти по типам: чтение, запись, исполнение или их комбинации.

Применяются различные механизмы защиты памяти, которые основаны на эмуляции или аппаратной реализации NX-бита (бита исполнения на страницах памяти) и PCID (Processor-Context ID — идентификатор контекста выполнения), а также технологии ASLR (рандомизация расположения виртуального адресного пространства) и KASLR (рандомизация адресного пространства ядра). При наличии аппаратной поддержки NX-бита и PCID на процессорах x86-64 производительность вычислительной системы не ухудшается.

Ядро ОС имеет архитектуру, обеспечивающую невозможность его перемещения в физическом адресном пространстве, при этом технология KASLR задет случайное смещение начального адреса выполнения ядра при каждой его загрузке.

Технология ASLR обеспечивает случайный характер (рандомизацию) смещений сегментов кода и данных (в том числе стека и кучи) при использовании системного вызова отображения в память `mmap()`.

Гарантия того, что адреса с произвольным доступом не будут одновременно доступны на запись и выполнение, реализуется использованием возможностей системных вызовов `mmap()` и `mprotect()`.

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 используется библиотека `libgost`, в которой для вычисления контрольных сумм реализованы функции хеширования в соответствии с ГОСТ Р 34.11-94, ГОСТ Р 34.11-2012 с длиной хеш-кода 256 бит и ГОСТ Р 34.11-2012 с длиной хеш-кода 512 бит. По умолчанию используются функции хеширования в соответствии с ГОСТ Р 34.11-2012. Использование функции в соответствии с ГОСТ Р 34.11-94 сохранено для совместимости.

Библиотека `libgost` используется в средствах подсчета контрольных сумм файлов и оптических дисков, контроля соответствия дистрибутиву и регламентного контроля целостности, модулях аутентификации.

9.1. Средство подсчета контрольных сумм файлов и оптических дисков

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

```
gostsum -h
```

Синтаксис инструмента:

```
gostsum [КЛЮЧ] ... [ФАЙЛ]
```

Параметры инструмента `gostsum` приведены в таблице 40.

Таблица 40

Параметр	Описание
`--gost-94`	Устанавливает, что будет использован алгоритм ГОСТ Р 34.11-94
`--gost-2012`	Устанавливает, что будет использован алгоритм ГОСТ Р 34.11-2012 с длиной хеш-кода 256 бит (используется в качестве алгоритма по умолчанию, если алгоритм не задан параметром)
`--gost-2012-512`	Устанавливает, что будет использован алгоритм ГОСТ Р 34.11-2012 с длиной хеш-кода 512 бит
`-b`	Устанавливает размер блоков, которыми будет считываться файл
`-o`	Задает имя файла для вывода контрольной суммы (по умолчанию — стандартный поток вывода)

Окончание таблицы 40

Параметр	Описание
`-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

Пример

Подсчет 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 включен инструмент командной строки `gostsum_from_deb`. Для вывода информации о синтаксисе `gostsum_from_deb` необходимо выполнить команду:

```
gostsum_from_deb -h
```

Синтаксис инструмента:

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

Параметры инструмента `gostsum_from_deb` приведены в таблице 41.

Таблица 41

Параметр	Описание
`--gost-94`	Устанавливает, что будет использован алгоритм ГОСТ Р 34.11-94
`--gost-2012`	Устанавливает, что будет использован алгоритм ГОСТ Р 34.11-2012 с длиной хеш-кода 256 бит (по умолчанию)
`--gost-2012-512`	Устанавливает, что будет использован алгоритм ГОСТ Р 34.11-2012 с длиной хеш-кода 512 бит
`gostsum arguments`	Аргументы утилиты `gostsum`
`-d каталог`	Задает имя каталога, содержащего deb-пакеты, для файлов которых вычисляются контрольные суммы
`-p deb-пакет`	Задает имя deb-пакета, для файлов которого вычисляются контрольные суммы

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 хеш-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, представленного пакетом `afick`. В указанном наборе реализована возможность для проведения периодического (с использованием системного планировщика заданий `cron`) вычисления контрольных сумм файлов и соответствующих им атрибутов расширенной подсистемы безопасности PARSEC (мандатных атрибутов и атрибутов расширенной подсистемы протоколирования) с последующим сравнением вычисленных значений с эталонными. В указанном наборе программных средств реализовано использование библиотеки `libgost`, обеспечивающей подсчет контрольных сумм в соответствии с ГОСТ Р 34.11-2012 с длиной хеш-кода 256 бит.

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

```
afick -i
```

Созданный в результате выполнения команды файл `/var/lib/afick/afick` является БД формата `ndbm`. В файле содержится набор строк, в каждой строке указано имя файла и через пробел его атрибуты и сигнатуры. БД защищается системой разграничения доступа.

Для вычисления контрольных сумм могут использоваться алгоритмы MD5-Digest и ГОСТ Р 34.11-2012.

Настройка регламентного контроля целостности выполняется в конфигурационном файле `etc/afick.conf`, в котором указываются пути к файлам/каталогам, подвергаемым контролю целостности, и правила контроля целостности, применяемые к файлам и каталогам. Подробное описание файла приведено в `man afick.conf`.

По умолчанию в файле `etc/afick.conf` используются следующие правила контроля целостности:

1) правило `PARSEC`:

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

где `p+d+i+n+u+g+s+b+md5+m` — слежение за всеми стандартными атрибутами файла и использование хеш-функции MD5-Digest для слежения за целостностью содержимого файлов;

`+e+t` — контроль расширенных атрибутов: метки безопасности и флагов аудита, соответственно.

Контроль ACL осуществляется при установке флага `+g`;

2) правило `GOST`:

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

где `p+d+i+n+u+g+s+b+gost+m` — слежение за всеми стандартными атрибутами файла и использование хеш-функции ГОСТ Р 34.11-2012 с длиной хеш-кода 256 бит для слежения за целостностью содержимого файлов;

`+e+t` — контроль расширенных атрибутов: метки безопасности и флагов аудита, соответственно.

Контроль ACL осуществляется при установке флага `+g`;

В файле `/etc/afick.conf` на отдельной строке задается путь к файлу или каталогу и применяемое к нему правило контроля.

Пример

```
/boot            GOST
/bin             GOST
/etc/security    PARSEC
/etc/pam.d       PARSEC
/etc/fstab       PARSEC
/lib/modules     PARSEC
/lib64/security  PARSEC
/lib/security    PARSEC
/sbin            PARSEC
/usr/bin         PARSEC
/usr/lib         PARSEC
/usr/sbin        PARSEC
```

В конфигурационном файле также представлен ряд дополнительных путей с правилами контроля. Соответствующие строки помечены знаком комментария «#» и могут быть активированы удалением этого знака.

При запуске `afick` автоматически будет установлено ежедневное задание для `cron`. Файл с заданием находится в `/etc/cron.daily/afick_cron`.

Параметр `report_url:=stdout` задает местоположение файла-отчета.

В конфигурационном файле применен простой язык макросов, который используется при определении переменных для заданий системного планировщика заданий `cron`.

Рекомендуется утилитой `afick` и/или средствами аппаратно-программного модуля доверенной загрузки обеспечить контроль целостности следующих объектов:

1) файлов образов ядра ОС:

`/boot/vmlinuz-*`

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

`/boot/initrd.img-*`

3) конфигурационного файла меню загрузчика `GRUB 2`:

`/boot/grub/grub.cfg`

4) конфигурационного файла, определяющего используемый по умолчанию графический дисплейный менеджер:

`/etc/X11/default-display-manager`

5) конфигурационного файла настройки файловых систем, доступных для монтирования через NFS:

`/etc/exports`

6) конфигурационного файла, содержащего информацию о различных устройствах хранения и файловых системах:

`/etc/fstab`

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

`/etc/group`

8) сценариев для запуска служб в каталоге

`/etc/init.d/`

9) конфигурационного файла, определяющего параметры работы первого процесса пользовательского режима `init`:

`/etc/inittab`

10) конфигурационных файлов, определяющих порядок работы PAM-модулей:

`/etc/pam.conf`

`/etc/pam.d/chfn`

`/etc/pam.d/chsh`

`/etc/pam.d/common-account`

`/etc/pam.d/common-account.pam-old`

`/etc/pam.d/common-auth`

`/etc/pam.d/common-auth.pam-old`

```
/etc/pam.d/common-password
/etc/pam.d/common-password.pam-old
/etc/pam.d/common-session
/etc/pam.d/common-session.pam-old
/etc/pam.d/cron
/etc/pam.d/cups
/etc/pam.d/cvs
/etc/pam.d/dovecot
/etc/pam.d/fly-dm
/etc/pam.d/fly-dm-np
/etc/pam.d/login
/etc/pam.d/other
/etc/pam.d/passwd
/etc/pam.d/polkit
/etc/pam.d/samba
/etc/pam.d/sshd
/etc/pam.d/su
/etc/pam.d/sumac.xauth
```

11) файла, содержащего перечень локальных пользователей ОС (добавляется в список контроля целостности после создания всех необходимых пользователей):

```
/etc/passwd
```

12) символических ссылок на сценарии для запуска служб в каталогах:

```
/etc/rc*
```

13) конфигурационного файла, определяющего перечень терминалов, с которых суперпользователь `root` может регистрироваться в системе:

```
/etc/securetty
```

14) конфигурационного файла, определяющего перечень регистрируемых оболочек в ОС:

```
/etc/shells
```

15) конфигурационного файла, содержащего значения параметров командной строки ядра:

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

16) модулей ядра, входящих в подсистему безопасности PARSEC:

```
/lib/modules/*/misc/digsig_verif.ko
/lib/modules/*/misc/parsec.ko
/lib/modules/*/misc/parsec-cifs.ko
```

С помощью команд `lsmod` и `modinfo` можно определить перечень модулей ядра, подлежащих контролю целостности средствами АПМДЗ;

17) PAM-модулей в каталоге:

`/lib/security/pam`

18) исполняемых файлов в каталогах (в случае, если в ОС не используется замкнутая программная среда (см. 16.1)):

`/bin/`

`/sbin/`

`/usr/bin/`

`/usr/sbin/`

В режиме «Мобильный» в каталоге `/usr/sbin/` доступны сценарии регламентного контроля целостности:

1) `astra-mobile-create-int-db` — создание базы данных эталонного состояния системы. Если включена функция проверки целостности системы при запуске, то сценарий будет выполняться при каждом обновлении списка пакетов (установке/удалении пакетов);

2) `astra-mobile-int-check` — проверка целостности системы. Сценарий будет выполняться при запуске системы, если функция проверки целостности включена. В случае неудачной проверки делается соответствующую запись в файле `/tmp/astra-mobile-int-check.log`.

Описание настройки регламентного контроля целостности с помощью `afick` в графическом интерфейсе режима «Мобильный» приведено в электронной справке («Документация — Графический интерфейс — Режим «Мобильный»).

10. 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 из резервной копии. Инструменты создания резервных копий и восстановления из них описаны в документе РУСБ.10015-37 95 01-1, а также в 10.1.1 и 10.1.2.

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 необходимо:

1) установить DVD-диск с дистрибутивом ОС в устройство чтения DVD-дисков;
2) загрузить программу установки ОС с DVD-диска;
3) в окне приветствия программы установки выбрать язык установки (русский или английский);
4) в окне приветствия программы установки выбрать «Режим восстановления»;
5) в окне «[!] Лицензия» подтвердить согласие с лицензионным соглашением;
6) в окне «[!] Настройка клавиатуры» выбрать настройки переключения раскладки клавиатуры, после чего программой установки будет выполнена проверка оборудования и первичная загрузка программ;
7) в окне «[!] Настройка сети» задать имя компьютера (можно указать произвольное имя компьютера, настройки восстанавливаемой ОС не изменятся);

8) в окне «[!] Настройка времени» выбрать часовой пояс;

9) в окне «[! !] Войти в режим восстановления» последовательно выполнить следующие шаги:

а) выбрать пункт «Не использовать корневую файловую систему»;

б) выбрать следующую операцию режима восстановления: «Запуск оболочки в рабочей среде программы установки»;

в) нажать на кнопку **[<Продолжить>]**.

Будет выполнен переход в режим командной строки под управлением ядра, загруженного с DVD-диска;

10) определить имя раздела, в который была установлена ОС, для этого выполнить команду:

```
blkid
```

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).

Пример

Вывод выполнения команды `blkid`

```
/dev/sda1: UUID="bc485787-ef37-431c-8c8b-401055066c99" TYPE="ext4"
PARTUUID="9492e90e-01"
/dev/sda5: UUID="e8987cad-ee16-427a-a768-a9aa896b048c" TYPE="swap"
PARTUUID="9492e90e-05"
/dev/sr0: UUID="2021-06-11-12-41-04-00" LABEL="Astra Linux"
TYPE="iso9660" PTUUID="66c613b0" PTTYPE="dos"
```

В приведенном примере ОС была установлена в раздел `/dev/sda1`;

11) запустить автоматическую проверку и восстановление ФС, выполнив команду:

```
fsck.ext4 -p -f -c /dev/<имя раздела>
```

Пример

Вывод выполнения команды `fsck`

```
/dev/sda1:Updating bad block inode.
/dev/sda1:318177/2297456 files (0.2% non-contiguous), 4157309/9186816
blocks
```

12) после проверки нажать комбинацию клавиш **<Ctrl+D>** и извлечь DVD-диск с дистрибутивом ОС из устройства чтения DVD-дисков;

13) в окне «[! !] Войти в режим восстановления» выбрать пункт «Перезагрузка системы».

10.1.1. Комплекс программ Bacula

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

Описание комплекса программ Bacula, а также его установки и настройки приведено в документе РУСБ.10015-37 95 01-1.

ВНИМАНИЕ! Для восстановления объектов ФС с установленными мандатными атрибутами необходимо запустить консоль управления Bacula с PARSEC-привилегией `0x1000` (`PARSEC_CAP_UNSAFE_SETXATTR`). Привилегия может быть получена с использованием утилиты `execaps`:

```
sudo execaps -c 0x1000 -- bconsole
```

ВНИМАНИЕ! После восстановления объектов ФС с установленными мандатными атрибутами необходимо выполнить перемонтирование ФС, в которой восстанавливались объекты, или перезагрузить ОС.

10.1.2. Утилита архивирования `tar`

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

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

```
echo 1 | sudo tee /parsecfs/unsecure_setxattr
```

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

```
sudo execaps -c 0x1000 -- tar
```

ВНИМАНИЕ! Восстановление расширенных атрибутов файлов с использованием `unsecure_setxattr` возможно только в случае, если атрибуты восстанавливаются с помощью системного вызова `setxattr` путем установки атрибута `security.PDPL`. Использование `unsecure_setxattr` не влияет на возможность изменения мандатных атрибутов файлов системными вызовами `pdpl_set_path`, `pdpl_set_fd`.

После восстановления из резервных копий файлов с мандатными атрибутами необходимо выполнить команду:

```
echo 0 | sudo tee /parsecfs/unsecure_setxattr
```

Пример

Использование `tar`, при котором будут восстановлены расширенные атрибуты каталогов и файлов, вложенных в архив.

До использования утилиты должен быть создан пользователь user1, для которого заданы мандатные атрибуты, и пользователь должен выполнить вход в систему.

Создать от имени учетной записи администратора архив домашнего каталога пользователя с помощью команды:

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

где `-xattrs` — включает поддержку расширенных атрибутов;

`-acls` — включает поддержку POSIX ACL;

`-cvzf` — обеспечивают, соответственно, создание архива (`create`), включение режима отображения обрабатываемых файлов (`verboze`), применение метода сжатия (`gzip`), указание файла (`file`);

`/opt/home.tgz` — задает место расположения созданного архива и его имя;

`/home/.pdp/user1` — определяет каталоги или файлы для добавления в архив.

Перед восстановлением необходимо выполнить следующую команду, устанавливающую для параметра защиты файловой системы `/parsecfs/unsecure_setxattr` значение «1», разрешающее применять привилегию `PARSEC_CAP_UNSAFE_SETXATTR`:

```
echo 1 | sudo tee /parsecfs/unsecure_setxattr
```

Выполнить восстановление с помощью команды:

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

где `-xattrs-include=security.PDPL,AUDIT,DEF_AUDIT` — определяет подключаемый шаблон восстановления расширенных атрибутов (мандатных атрибутов, атрибутов аудита и атрибутов аудита по умолчанию) для ключа `xattrs`;

`-xvf` — обеспечивает, соответственно, извлечение из архива (`extract`), включение режима отображения обрабатываемых файлов (`verboze`), указание файла (`file`).

После восстановления необходимо выполнить команду, устанавливающую запрет на применение привилегии `PARSEC_CAP_UNSAFE_SETXATTR` (настройка ОС по умолчанию):

```
echo 0 | sudo tee /parsecfs/unsecure_setxattr
```

Дополнительная информация по команде `tar` приведена в РУСБ.10015-37 95 01-1.

10.2. Восстановление СУБД PostgreSQL после сбоев и отказов

Во избежание потерь данных БД PostgreSQL должны регулярно архивироваться.

В случае возникновения ошибок в хранящихся данных, нарушения 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 восста-

новлением, 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` (10.2.2), `pg_dumpall` (10.2.3), `pg_restore` (10.2.4) и, в том числе, интерактивный терминал `psql`, с помощью которого могут быть восстановлены резервные копии, сохраненные в виде сценария SQL.

10.2.1. 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 база данных (для назначения метки на файл копии).

В случае восстановления помимо указанных привилегий требуются 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 11, 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 11 необходимо установить параметр ac_auto_adjust_macs = true;

3. Для переноса кластера с предыдущих версий СУБД на версию 11 необходимо воспользоваться утилитами pg_upgradecluster или pg_upgrade (см. документ «Операционная система специального назначения «Astra Linux Special Edition» Руководство администратора. Часть 2»).

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

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

Синтаксис:

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

Описание основных параметров pg_dump приведено в таблице 42.

Таблица 42

Параметр	Описание
-f, --file=FILENAME	Имя выходного файла
-F, --format=c|t|p	Формат выходного файла (пользовательский, tar, текстовый)
-v, --verbose	Режим вывода всех сообщений

Окончание таблицы 42

Параметр	Описание
-Z, --compress=0-9	Уровень сжатия для форматов сжатия
--lock-wait-timeout=TIMEOUT	Завершение ошибкой после ожидания TIMEOUT для блокировки таблицы
--disable-macs	Исключить из выгрузки вывод команд MAC LABEL и MAC CCR
--help	Вывести справку и выйти
--version	Вывести информацию о версии и выйти

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

Параметры установки соединения приведены в таблице 43.

Таблица 43

Параметр	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.

10.2.3. pg_dumpall

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

Сценарий 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 44.

Таблица 44

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 43).

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

10.2.4. 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 также 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, сценарий, содержащий необходимые для пересоздания БД SQL-команды, создается и выводится в файл или стандартный поток вывода. Результирующий сценарий эквивалентен формату текстового вывода утилиты pg_dump. Вследствие этого некоторые параметры, управляющие выводом, аналогичны параметрам pg_dump (см. 10.2.2).

Синтаксис:

```
pg_restore [ОПЦИЯ]... [ФАЙЛ]
```

Опции общего характера приведены в таблице 45.

Таблица 45

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`.

10.3. Восстановление в режиме «Мобильный»

При функционировании ОС в режиме «Мобильный» в случае сбоя возможно восстановление ОС с сохранением данных в каталогах `/opt` и `/home` путем выполнения обновления с установочного USB-носителя или из образа в формате IMG (см. РУСБ.10015-37 95 01-1).

В случае ошибки или отказа ОС возможно сбросить ее настройки к значениям по умолчанию (на состояние после установки ОС), при этом все данные на устройстве буду удалены.

Для сброса настроек необходимо выполнить команду:

```
sudo astra-mobile-factory-reset
```

Также возможно выполнить сброс настроек в графическом интерфейсе:

1) открыть панель быстрого доступа и нажать на кнопку **[Настройки]**;
2) выбрать пункт «Обновление системы»;
3) нажать кнопку **[Сбросить к заводским настройкам]** и затем в предупреждении нажать **[Я уверен]**;
4) в окне аутентификации ввести пароль администратора и нажать **[Да]**;
5) в окне программы начальной настройки:

а) нажать **[Начать]**;

б) в окне «Пароль администратора» задать пароль для учетной записи `administrator` и перейти к следующей настройке;

в) в окне «Уровни защищенности» из выпадающего списка выбрать уровень защищенности (см. РУСБ.10015-37 95 01-1) и при необходимости включить функции безопасности:

- если выбран усиленный уровень защищенности, то для включения будет доступен мандатный контроль целостности;
- если выбран максимальный уровень защищенности, то для включения будут доступны мандатный контроль целостности и мандатное управление доступом.

Перейти к следующей настройке;

г) для применения настроенной конфигурации ОС нажать **[Сохранить]** и подтвердить действие в открывшемся диалоговом окне — после применения настроек ОС будет перезапущена. Если необходимо изменить выбранные настройки, следует нажать **[Назад]**. Для отмены настроек и закрытия программы началь-

ной настройки нажать **[Отменить]** — при этом устройство будет выключено, а при следующем включении устройства будет запущена программа начальной настройки.

11. ФИЛЬТРАЦИЯ СЕТЕВОГО ПОТОКА

11.1. Включение фильтрации сетевого потока

При помощи фильтра сетевых пакетов можно осуществлять контроль сетевого трафика, проходящего через 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`

Фильтр сетевых пакетов реализован на основе технологии `iptables` и позволяет выполнять следующие задачи:

1) фильтрацию пакетов — это механизм, который на основе заданных правил, разрешает или запрещает передачу информации, проходящей через него, с целью ограждения подсети от внешнего доступа, или, наоборот, для недопущения выхода наружу. Фильтр пакетов может определять правомерность передачи информации только на основе заголовков IP-пакетов, а также может анализировать и их содержимое, т. е. использовать данные протоколов более высокого уровня;

2) трансляцию сетевых адресов (т. н. «маскарадинг») — это подмена некоторых параметров в заголовках IP-пакетов. Используется для сокрытия реальных IP-адресов компьютеров защищаемой подсети, а также для организации доступа из подсети с компьютерами, не имеющими реальных IP-адресов, к глобальной сети;

3) прозрачное проксирование — это переадресация пакетов на другой порт компьютера. Обычно используется для того, чтобы пользователи из подсети использовали

proxy-сервер маршрутизатора без дополнительного конфигурирования их клиентских программ.

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

11.3. Формирование правил

Каждое правило — это строка, содержащая критерии, определяющие применимость данного правила к пакету, и действие, которое необходимо выполнить в случае выполнения критерия.

Правила имеют следующий синтаксис:

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

Если в правило не включается спецификатор таблицы `[-t table]`, то по умолчанию предполагается использование таблицы `filter`. Если необходимо использовать другую таблицу, то это требуется указать явно. Спецификатор таблицы можно указывать в любом месте строки правила, однако принято указывать в начале правила.

Непосредственно после имени таблицы должна стоять команда. Если спецификатора таблицы нет, то команда всегда должна стоять первой. Команда определяет действие `iptables`, например, вставить, добавить в конец цепочки или удалить правило.

В разделе `match` указываются критерии проверки, по которым определяется применимость данного правила к пакету. Возможно указать в качестве критерия IP-адрес источника пакета или сети, сетевой интерфейс и т. п.

Параметр `target` указывает, какое действие должно быть выполнено при условии выполнения критериев в правиле. Возможны такие действия как: передача пакета ядром в другую цепочку правил, «сбросить» пакет и забыть про него, выдать на источник сообщение об ошибке и т. п.

Созданные правила фильтрации сохраняются и применяются только до перезагрузки ОС, после перезагрузки ОС требуется повторное формирование правил.

Для обеспечения возможности восстановления правил фильтрации после перезагрузки ОС необходимо:

1) создать файл `/etc/iptables.rules` для сохранения правил фильтрации:

```
sudo touch /etc/iptables.rules
```

2) установить для созданного файла ограничения на чтение, так как из него можно получить информацию об открытых сетевых портах, которая может быть использована для атак на компьютер:

```
sudo chmod 640 /etc/iptables.rules
```

3) сохранить существующие правила фильтрации в файле `/etc/iptables.rules`:

```
sudo iptables-save | sudo tee /etc/iptables.rules > /dev/null
```

4) создать сценарий `/etc/network/if-pre-up.d/iptables` со следующим содержимым:

```
iptables-restore < /etc/iptables.rules
exit 0
```

5) создать сценарий `/etc/network/if-post-down.d/iptables` со следующим содержимым:

```
touch /etc/iptables.rules
chmod 640 /etc/iptables.rules
iptables-save > /etc/iptables.rules
exit 0
```

6) сделать созданные сценарии исполняемыми:

```
sudo chmod +x /etc/network/if-post-down.d/iptables
     /etc/network/if-pre-up.d/iptables
```

Примечание. Для сохранения правил фильтрации рекомендуется выполнять пункты 1)–3) перечисления на стр. 207 после каждого изменения правил. Сценарий, приведенный в пункте 5) перечисления, сохраняет текущую версию правил автоматически только при штатном выключении компьютера. При нештатном выключении компьютера или сбое изменения в правилах фильтрации могут быть утеряны.

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, прежде чем он будет передан далее. На каждом этапе пакет может быть остановлен. После прохождения таблиц пакет передается либо локальному приложению, либо перенаправляется на другой компьютер.

В таблице `filter` цепочку `FORWARD` проходят все пакеты, которые движутся через сетевой фильтр. Порядок следования пакета приведен в таблице 46.

Таблица 46

Шаг	Таблица	Цепочка	Описание
1	=	=	Кабель
2	=	=	Сетевой интерфейс (например, `eth0`)
3	`mangle`	`PREROUTING`	Используется для внесения изменений в заголовок пакета, например, для изменения битов `TOS` и пр.
4	`nat`	`PREROUTING`	Используется для трансляции сетевых адресов `DNAT`. `SNAT` выполняется позднее, в другой цепочке. Любого рода фильтрация в этой цепочке может производиться только в исключительных случаях

Окончание таблицы 46

Шаг	Таблица	Цепочка	Описание
5	=	=	Принятие решения о дальнейшей маршрутизации, т. е. в этой точке решается, куда пойдет пакет — локальному приложению или на другой узел сети
6	filter	FORWARD	Попадают только те пакеты, которые идут на другой компьютер. Вся фильтрация транзитного трафика должна выполняться здесь. Через эту цепочку проходит трафик в обоих направлениях, поэтому обязательно учитывать это обстоятельство при написании правил фильтрации
7	nat	POSTROUTING	Предназначена в первую очередь для SNAT. Не использовать для фильтрации без особой необходимости. Здесь же выполняется и маскарадинг
8	=	=	Выходной сетевой интерфейс (например, eth1)
9	=	=	Кабель

Пакеты, предназначенные локальному процессу/приложению, проходят через цепочку INPUT, а не через FORWARD. В таблице 47 представлен порядок движения пакета.

Таблица 47

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

В таблице 48 представлен порядок движения пакетов, созданных локальными процессами.

Таблица 48

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		

Окончание таблицы 48

Шаг	Таблица	Цепочка	Описание
4	=	=	Принятие решения о маршрутизации. Здесь решается — куда пойдет пакет дальше
5	nat	POSTROUTING	Здесь выполняется SNAT. Не следует в этой цепочке производить фильтрацию пакетов во избежание нежелательных побочных эффектов. Однако и здесь можно останавливать пакеты, применяя политику по умолчанию — DROP
6	=	=	Сетевой интерфейс (например, eth0)
7	=	=	Кабель

mangle

В таблице mangle не следует производить фильтрацию, маскировку или преобразование адресов (DNAT, SNAT), в ней допускается выполнять действия, приведенные в таблице 49.

Таблица 49

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 таблицы nat. Трансляция адресов или маскировка применяются ко всем последующим пакетам в потоке автоматически. Для этой таблицы характерны действия, приведенные в таблице 50.

Таблица 50

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

Окончание таблицы 50

Действие	Описание
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 таблице filter содержатся наборы правил для выполнения фильтрации пакетов. Пакеты могут пропускаться далее либо отвергаться в зависимости от их содержимого.

В таблице filter можно выполнить DROP, LOG, ACCEPT или REJECT. 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

Таблица трассировщика записана в файле /proc/net/ip_conntrack. В ней содержится список всех активных соединений.

Если модуль ip_conntrack загружен, то команда cat /proc/net/ip_conntrack должна вывести:

```
tcp 6 117 SYN_SENT src=192.168.1.6 dst=192.168.1.9 sport=32775 dport=22
     [UNREPLIED] src=192.168.1.9 dst=192.168.1.6 sport=22 dport=32775 use=2
```

В примере вывода команды содержится вся информация, которая известна трассировщику по конкретному соединению.

В начале указано название протокола, в данном случае — tcp. Далее следует некоторое число в обычном десятичном представлении. После него следует число, определяющее «время жизни» записи в таблице (т. е. количество секунд, через которое информация о соединении будет удалена из таблицы). В приведенном примере запись в таблице будет храниться еще 117 с, если через это соединение более не проследует ни одного пакета, в противном случае это значение будет установлено в значение по умолчанию для заданного состояния. Это число уменьшается на 1 каждую секунду. Далее следует фактическое состояние соединения. В примере это состояние имеет значение SYN_SENT. Внутреннее представление состояния несколько отличается от внешнего. Значение SYN_SENT говорит о том, что через данное соединение проследовал единственный пакет TCP SYN. Далее расположены адреса отправителя и получателя, порты отправителя и получателя. Здесь же указано ключевое слово, которое сообщает о том, что ответного трафика через это соединение еще не было. Далее приводится дополнительная информация по ожидаемому пакету — это IP-адреса и номера портов отправителя и получателя для ожидаемого ответного пакета (те же данные, только поменявшиеся местами).

После получения ответа трассировщик снимет флаг [UNREPLIED] и заменит его флагом [ASSURED]. Этот флаг сообщает, что соединение установлено, и эта запись не будет стерта по достижении максимально возможного количества трассируемых соединений.

Максимальное количество записей, которое может содержаться в таблице, зависит от значения по умолчанию, которое может быть установлено вызовом функции ipsysctl.

Для объема ОЗУ 256 МБ значение соответствует 16376 записям. Можно посмотреть и изменить это значение через:

```
/proc/sys/net/ipv4/ip_conntrack_max
```

Состояния

Сетевые пакеты могут иметь несколько различных состояний в пределах ядра в зависимости от типа протокола. Однако вне ядра имеется только четыре состояния, как было сказано выше. Параметры, описывающие состояние пакета, используются в критерии `--state`. Допустимыми являются: `NEW`, `ESTABLISHED`, `RELATED` и `INVALID`.

В таблице 51 подробно рассмотрено каждое из возможных состояний и приведены необходимые комментарии.

Таблица 51

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 52 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 52

Команда	Использование
`-A, --append`	Добавляет новое правило в конец заданной цепочки. Пример `iptables -A INPUT ...`

Продолжение таблицы 52

Команда	Использование
`-D, --delete`	Удаляет правило из цепочки. Команда имеет два формата записи: первый — когда задается критерий сравнения с параметром `-D`, второй — порядковый номер правила. Если задается критерий сравнения, то удаляется правило, которое имеет в себе этот критерий, если задается номер правила, то будет удалено правило с заданным номером. Счет правил в цепочках начинается с единицы. Пример `iptables -D INPUT --dport 80 -j DROP, iptables -D INPUT 1`
`-E, --rename-chain`	Выполняет переименование пользовательской цепочки. В примере цепочка `allowed` будет переименована в цепочку `disallowed`. Эти переименования не изменяют порядок работы. Пример `iptables -E allowed disallowed` Команда должна быть указана всегда
`-F, --flush`	Сброс (удаление) всех правил из заданной цепочки (таблицы). Если имя цепочки и таблицы не указывается, то удаляются все правила во всех цепочках. Пример `iptables -F INPUT`
`-I, --insert`	Вставляет новое правило в цепочку. Число, следующее за именем цепочки, указывает номер правила, перед которым следует вставить новое правило. В примере выше указывается, что данное правило должно быть первым в цепочке `INPUT`. Пример `iptables -I INPUT 1 --dport 80 -j ACCEPT`
`-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 52

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 53.

Таблица 53

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 53

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 символьных множителей `K` (x1000), `M` (x1,000,000) и `G` (x1,000,000,000). Чтобы команда `--list` выводила полное число (без употребления множителей), требуется применять ключ `-x`, который описан ниже. При использовании с другими командами на вывод будет подан подробный отчет о произведенной операции
`-x, --exact`	Используется вместе с командой `--list`. Для всех чисел в выходных данных выводятся их точные значения без округления и без применения множителей `K`, `M`, `G`. Важно то, что данный ключ используется только с командой `--list` и не применяется с другими командами

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, вызвавшим переход. Если к пакету во вложенной цепочке будет применено действие ACCEPT, то автоматически пакет будет считаться принятым и в вызывающей цепочке и уже не будет продолжать движение по вызывающим цепочкам. Однако пакет пойдет по другим цепочкам в других таблицах.

Действие — это предопределенная команда, описывающая действие, которое необходимо выполнить, если пакет совпал с заданным критерием. Например, можно применить действие DROP или ACCEPT к пакету. В результате выполнения одних действий пакет прекращает свое прохождение по цепочке, например DROP и ACCEPT; в результате других, после выполнения неких операций, продолжает проверку, например LOG; в результате работы третьих — даже видоизменяется, например DNAT и SNAT, TTL и TOS, но 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 параметра --protocol в критерии.

DROP

DROP сбрасывает пакет и iptables забывает о его существовании. Сброшенные пакеты прекращают свое движение полностью, т. е. они не передаются в другие таблицы, как это происходит в случае с действием ACCEPT. Следует помнить, что данное действие может иметь негативные последствия, поскольку может оставлять незакрытые сокеты как на стороне сервера, так и на стороне клиента, наилучшим способом защиты будет использование действия REJECT, особенно при защите от сканирования портов.

LOG

LOG служит для журналирования отдельных пакетов и событий. В журнал могут заноситься заголовки IP-пакетов и другая интересующая информация. Информация из

журнала может быть прочитана с помощью dmesg или syslogd, либо с помощью других команд.

Ключи действия LOG приведены в таблице 54.

Таблица 54

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. изменения исходящего IP-адреса в IP-заголовке пакета. SNAT допускается выполнять только в таблице nat, в цепочке POSTROUTING. Другими словами, только здесь допускается преобразование исходящих адресов. Если первый пакет в соединении подвергся преобразованию исходящего адреса, то все последующие пакеты из этого же соединения будут преобразованы автоматически и не пойдут через эту цепочку правил.

Ключ 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 55.

Таблица 55

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 добавле-

ния надстроек (плагинов) для формирования различных выходных форматов и обработки сетевых протоколов.

Ключи для действия ULOG приведены в таблице 56.

Таблица 56

Ключ	Описание
--ulog-nlgroup	Сообщает ULOG, в какую группу netlink должен быть передан пакет. Всего существует 32 группы (от 1 до 32). Если необходимо передать пакет в пятую группу, то можно указать: --ulog-nlgroup 5 По умолчанию используется первая группа. Пример iptables -A INPUT -p TCP --dport 22 -j ULOG --ulog-nlgroup 2
--ulog-prefix	Имеет тот же смысл, что и аналогичный параметр в действии LOG. Длина строки префикса не должна превышать 32 символа. Пример iptables -A INPUT -p TCP --dport 22 -j ULOG --ulog-prefix "SSH connection attempt: "
--ulog-cprange	Определяет какую долю пакета в байтах надо передавать демону ULOG. Если указать число 100, как показано в примере, то демону будет передано только 100 Б из пакета, это означает, что демону будет передан заголовок пакета и некоторая часть области данных пакета. Если указать 0, то будет передан весь пакет, независимо от его размера. Значение по умолчанию равно 0. Пример iptables -A INPUT -p TCP --dport 22 -j ULOG --ulog-cprange 100
--ulog-qthreshold	Устанавливает величину буфера в области ядра. Например, если задать величину буфера, равной 10, как в примере, то ядро будет накапливать журналируемые пакеты во внутреннем буфере и передавать в пользовательское пространство группами по 10 пакетов. Пример iptables -A INPUT -p TCP --dport 22 -j ULOG --ulog-qthreshold 10

11.8. Поддержка фильтрации на основе классификационных меток

11.8.1. Модули iptables для работы с классификационными метками

Поддержка фильтрации на основе классификационных меток реализована с помощью дополнительного модуля тестирования astralabel, обеспечивающего тестирование значений мандатных атрибутов с помощью параметров --maclev и --maccat.

Для работы с классификационными метками необходимо установить пакеты iptables (по умолчанию пакеты не устанавливаются):

- iptables-astralabel-generic — модули для использования с ядром generic;

- iptables-astralabel-hardened — для использования с ядром hardened;

- iptables-astralabel-common — общие модули.

Для установки пакетов выполнить:

- для ядра generic

```
sudo apt install iptables-astralabel-common iptables-astralabel-generic
```

- для ядра hardened

```
sudo apt install iptables-astralabel-common iptables-astralabel-hardened
```

При необходимости модули для generic и hardened могут быть установлены одновременно.

Модуль astralabel поддерживает стандартный синтаксис командной строки, используемый в iptables, а также дополнительные параметры для контроля мандатных атрибутов сетевых пакетов, приведенные в таблице 57.

Таблица 57

Параметр	Описание
-m astralabel	Указание на использование модуля astralabel для обработки сетевого трафика
--maclev <уровень> [:<уровень>]	Применение правила к пакетам, имеющим указанный иерархический уровень конфиденциальности. Для задания диапазона уровней конфиденциальности уровни указываются через символ «:» включительно. Пример Не принимать пакеты с иерархическими уровнями конфиденциальности от 1 до 3 `iptables -A INPUT -m astralabel --maclev 1:3 -j DROP` Параметр maclev может применяться одновременно с параметром maccat в одном правиле. Результирующий фильтр будет представлять собой объединение фильтров, заданных данными параметрами
!	Значение фильтра, задаваемого параметром --maclev, может быть инверсировано с помощью модификатора «!». Пример Не пропускать исходящие пакеты, имеющие ненулевой иерархический уровень конфиденциальности `iptables -A OUTPUT -m astralabel ! --maclev 0 -j DROP` Данное правило будет пропускать исходящие пакеты, имеющие нулевой иерархический уровень конфиденциальности и ненулевые неиерархические категории доступа

Окончание таблицы 57

Параметр	Описание

--maccat <бит_категории>	Применение правила к пакетам, имеющим указанные неиерархические категории конфиденциальности. Задание диапазонов и инверсирование не поддерживаются. В одном правиле может быть указано несколько параметров maccat, в таком случае правило будет применяться только к пакетам с установленными одновременно всеми указанными категориями (битами).
	Примечание. Нумерация битов категорий начинается с единицы.
	
	Пример
	Не пропускать исходящие пакеты с установленными одновременно битами категорий 1 и 2
	`iptables -A OUTPUT -m astralabel --maccat 1 --maccat 2 -j DROP`
	
	Параметр maccat может применяться одновременно с параметром maclev в одном правиле. Результирующий фильтр будет представлять собой объединение фильтров, заданных данными параметрами.
	
	Пример
	Не принимать пакеты с установленными битами категорий 1 и 2 и уровнем конфиденциальности 3
	`iptables -A INPUT -m astralabel -maclev 3 --maccat 1 --maccat 2 -j DROP`

ВНИМАНИЕ! Если параметры для фильтрации пакетов на основе классификационной метки не указаны, то правило применяется ко всем пакетам, имеющим ненулевую классификационную метку. Например, правило

```
iptables -A OUTPUT -m astralabel -j DROP
```

запретит все исходящие пакеты, имеющие ненулевую классификационную метку (т.е. имеющие ненулевой уровень конфиденциальности и/или ненулевые категории доступа).

Описание модулей iptables для фильтрации пакетов на основе классификационной метки приведено в руководстве man для iptables-astralabel.

11.8.2. Использование ufw для работы с классификационными метками

В межсетевом экране ufw включена поддержка работы с классификационными метками по умолчанию. Для управления сетевыми соединениями с учетом классификационных меток в межсетевой экран ufw добавлены параметры mac, maclev и maccat, по действию аналогичные параметрам iptables.

Для того, чтобы данные параметры работали, в системе должен быть установлен в соответствии с 11.8.1 пакет iptables-astralabel-common и один из пакетов iptables-astralabel-generic или iptables-astralabel-hardened, соответствующий используемому ядру.

Примеры:

1. Запрет отправки на 80 IP-порт пакетов с ненулевой классификационной меткой:

```
ufw deny out 80/tcp mac
```

Аналогично параметрам `iptables`:

```
-A OUTPUT -p tcp --dport 80 -m astralabel -j DROP
```

2. Запрет отправки на 80 IP-порт (протокол HTTP) сетевых пакетов, имеющих уровень конфиденциальности 2:

```
ufw deny out 80/tcp maclev 2
```

Аналогично параметрам `iptables`:

```
-A OUTPUT -p tcp --dport 80 -m astralabel --maclev 2 -j DROP
```

3. Запрет отправки на 80 IP-порт (протокол HTTP) сетевых пакетов, имеющих категорию конфиденциальности 3:

```
ufw deny out 80/tcp maccat 3
```

Аналогично параметрам `iptables`:

```
-A OUTPUT -p tcp --dport 80 -m astralabel --maccat 3 -j DROP
```

Возможность инверсии правил и возможность одновременного указания нескольких категорий в одном правиле текущей реализацией ufw не поддерживаются.

Описание ufw для фильтрации пакетов на основе классификационной метки приведено в руководстве `man` для `ufw`.

11.8.3. Использование Open vSwitch для работы с классификационными метками

Программный коммутатор Open vSwitch (OVS) из состава ОС поддерживает классификационные метки (уровни и категории конфиденциальности) и может использоваться для фильтрации сетевого потока в условиях мандатного управления доступом.

Фильтрация IP-пакетов, которые содержат классификационные метки, сначала выполняется на портах, затем на основе правил OpenFlow коммутатора OVS. Параметры фильтрации задаются путем назначения классификационных меток портам и добавления классификационных меток в правила OpenFlow.

Для назначения порту или указания в правиле OpenFlow уровня конфиденциальности используется параметр `astra_mac_level`:

```
astra_mac_level=<уровень_конфиденциальности>
```

где `<уровень_конфиденциальности>` — десятичное число, соответствующее уровню конфиденциальности.

Для назначения порту или указания в правиле OpenFlow категорий конфиденциальности используется параметр `astra_mac_categories`:

astra_mac_categories=<категории_конфиденциальности>

где <категории_конфиденциальности> — шестнадцатеричное число, битовая маска которого определяет набор категорий конфиденциальности.

Если параметры уровня и/или категорий конфиденциальности отсутствуют, то по умолчанию для них принимается значение 0 — нулевой уровень конфиденциальности и/или без категорий конфиденциальности.

При фильтрации IP-пакет пропускается через порт и правилом OpenFlow, если IP-пакету назначена классификационная метка не выше, чем классификационная метка порта и в правиле OpenFlow соответственно. Например, если в правиле OpenFlow задана блокировка со значениями параметров astra_mac_level=2 и astra_mac_categories=0x03, то при применении правила не будут пропущены IP-пакеты, имеющие нулевой, первый или второй уровень конфиденциальности и имеющие категории конфиденциальности, битовая маска которых содержится в битовой маске 0x03.

Примеры:

1. Добавить порт tap0 и назначить ему второй уровень конфиденциальности и категории конфиденциальности, соответствующие битовой маске 0x03:

```
sudo ovs-vsctl add-port br0 tap0 astra_mac_level=2
     astra_mac_categories=0x03
```

2. Добавить порт tap1 и назначить ему категории конфиденциальности, соответствующие битовой маске 0x08, по умолчанию будет назначен нулевой уровень конфиденциальности:

```
sudo ovs-vsctl add-port br0 tap1 astra_mac_categories=0x08
```

3. Добавить порт tap3 с нулевым уровнем конфиденциальности и без категорий конфиденциальности:

```
sudo ovs-vsctl add-port br0 tap3
```

4. Для порта tap0 изменить текущий уровень конфиденциальности на третий уровень конфиденциальности (будут изменены только указанные параметры):

```
sudo ovs-vsctl upd-port br0 tap0 astra_mac_level=3
```

5. Для порта tap3 изменить текущие уровень и категории конфиденциальности на первый уровень конфиденциальности и категории конфиденциальности, соответствующие битовой маске 0x7 (будут изменены только указанные параметры):

```
sudo ovs-vsctl upd-port br0 tap3 astra_mac_level=1
     astra_mac_categories=0x7
```

6. Добавить правило блокировки IP-пакетов, которые в своем уровне IP имеют ip_dst=87.250.250.242 и которым назначен третий уровень конфиденциальности и категории конфиденциальности, соответствующие битовой маске 0x200:

```
sudo ovs-ofctl add-flow br0 ip,ip_dst=87.250.250.242,astra_mac_level=3,
     astra_mac_categories=0x200,actions=drop
```

7. Добавить правило блокировки IP-пакетов с третьим уровнем конфиденциальности и любым набор категорий конфиденциальности:

```
sudo ovs-ofctl add-flow br0 ip,astra_mac_level=3,actions=drop
```

Информация о назначенных портам коммутатора OVS уровнях и категориях конфиденциальности отображается в выводе команды:

```
sudo ovs-vsctl show
```

Пример

```
sudo ovs-vsctl show
```

Вывод команды:

```
aea83ec6-49bd-4b44-aa6b-82967bcd4a5b
Bridge br0
datapath_type: netdev
Port tap1
MAC categories: 0x0000000000000008
Interface tap1
Port tap0
MAC Level: 3
MAC categories: 0x0000000000000003
Interface tap0

Port tap3
MAC Level: 1
MAC categories: 0x0000000000000007
Interface tap3
Port br0
Interface br0
type: internal
ovs_version: "3.0.1"
```

Для получения информации о правилах, в которых заданы уровни и категории конфиденциальности, выполнить команду:

```
sudo ovs-ofctl -F astra dump-flows bridge
```

12. МАРКИРОВКА ДОКУМЕНТОВ

12.1. Общие сведения

Защищенный комплекс программ печати и маркировки документов обеспечивает маркировку выводимых на печать документов. Маркировка документов осуществляется при включенном в ОС мандатном управлении доступом.

Дополнительно к дискреционному управлению доступом на основе политик сервера CUPS используется мандатное управление доступом. Мандатные атрибуты порождаемых объектов в CUPS наследуют мандатный контекст, получаемый с сетевого соединения.

Все субъекты доступа (пользователи) сервера печати делятся на две группы: администраторов и обычных пользователей.

Администраторам разрешено выполнять ряд действий по модификации сервера печати (добавление/удаление принтеров, модификация их параметров и т.д.).

ВНИМАНИЕ! Данные операции должны проводиться только под нулевым мандатным контекстом.

Пользователям разрешается выводить документы на печать, выполнять модификацию и просмотр заданий согласно их мандатному контексту доступа. В таблице 58 приведено разделение операций по группам доступа и типам операций доступа.

Таблица 58

	Чтение	Запись

Административные операции	`CUPS-Get-Devices`
	`CUPS-Get-PPDs`
	`CUPS-Get-PPD`
	`CUPS-Get-Policies`
	`MAC-Get-Info`
		`Pause-Printer`
		`Resume-Printer`
		`Set-Printer-Attributes`
		`Enable-Printer`
		`Disable-Printer`
		`Hold-New-Jobs`
		`Release-New-Jobs`
		`CUPS-Add-Modify-Printer`
		`CUPS-Delete-Printer`
		`CUPS-Add-Modify-Class`
		`CUPS-Delete-Class`
		`CUPS-Set-Default`
		`CUPS-Create-Local-Printer`

Окончание таблицы 58

	Чтение	Запись
Неадминистративные операции	CUPS-Get-Printers Get-Printer-Supported-Values Get-Printer-Attributes CUPS-Get-Default Get-Subscription-Attributes Get-Subscriptions Get-Notifications Get-Jobs Validate-Job Get-Job-Attributes CUPS-Get-Document MAC-Get-Journal	Purge-Jobs Print-Job Cancel-Jobs Create-Job-Subscription Create-Printer-Subscriptions Renew-Subscription Cancel-Subscription Create-Job Send-Document Cancel-Job Hold-Job Release-Job Restart-Job Set-Job-Attributes Cancel-My-Jobs Close-Job Cups-Authenticate-Job CUPS-Move-Jobs MAC-Set-Job-Attributes MAC-Mark-Document MAC-Lock-Job MAC-Unock-Job

12.2. Настройка печати документа с ненулевой классификационной меткой

Для обеспечения возможности обрабатывать задания печати, формируемые в ненулевом мандатном контексте, а также для печати данных заданий на принтере необходимо серверу CUPS и принтеру установить политику `parsec`, выполнив, соответственно, от имени учетной записи администратора команды:

```
sudo cupsctl DefaultPolicy=parsec
sudo lpattr -p <имя_принтера> -s printer-op-policy=parsec
```

Описание инструмента командной строки `cupsctl` приведено в `man cupsctl`.

Для принтера установить политику `parsec` также можно с использованием графической утилиты `fly-admin-printer` во вкладке «МАС». Дополнительная информация об утилите `fly-admin-printer` приведена в электронной справке.

Принтеры и классы являются сущностями-контейнерами, в которые помещаются задания, и обладают атрибутами `mac-printer-mac-max` и `mac-printer-mac-min`.

Атрибут `mac-printer-mac-max` определяет максимальный мандатный контекст заданий, которые могут находиться в сущности-контейнере.

Атрибут `mac-printer-mac-min` определяет минимальный мандатный контекст заданий, которые могут находиться в сущности-контейнере.

Для разрешения печати на принтере документов пользователей, работающих в ненулевом мандатном контексте, необходимо для принтера установить допустимый диапазон мандатного контекста, выполнив следующие команды:

```
sudo lpattr -p <имя_принтера> -s mac-printer-mac-min=LLabel
sudo lpattr -p <имя_принтера> -s mac-printer-mac-max=MLabel
```

где LLabel — минимальный мандатный контекст заданий, которые разрешено печатать;

MLabel — максимальный мандатный контекст заданий, которые разрешено печатать.

Либо с использованием графической утилиты `fly-admin-printer`, задав во вкладке «МАС» допустимый диапазон уровней конфиденциальности, категорий конфиденциальности и уровней целостности.

Если ЕПП не используется, то настройка принтера выполняется от имени администратора через механизм `sudo` или от имени пользователя, входящего в локальную группу администраторов печати (группа `lpadmin`, см. документ РУСБ.10015-37 95 01-1). Если ЕПП используется, то выполняется от имени пользователя, входящего в локальную группу администраторов печати.

Параметры сервера CUPS определены в конфигурационном файле `/etc/cups/cupsd.conf`. Редактирование конфигурационного файла сервера CUPS выполняется от имени пользователя, входящего в локальную группу администраторов печати с использованием:

- инструмента командной строки `cupsctl`, запускаемого через механизм `sudo`;
- графической утилиты `fly-admin-printer`;
- web-интерфейса сервера CUPS (`http://127.0.0.1:631`).

Для настройки маркировки используются параметры конфигурационного файла `/etc/cups/cupsd.conf`, приведенные в таблице 59.

Таблица 59

Параметр	Значение по умолчанию	Описание
`MacDuplicate`	`off`	Включает дублирование заданий. Копии заданий сохраняются по умолчанию в каталоге `/var/spool/cups/parsec` в `pdf`-формате

Окончание таблицы 59

Параметр	Значение по умолчанию	Описание
MacDuplex	off	Режим печати маркера на обратной стороне (при наличии поддержки со стороны принтера двусторонней печати). Возможные значения: - off — отключено. Печать документа в обычном режиме, при котором создаются два задания: задание печати страниц документа с проставленными маркерами и задание печати маркера на обороте последней страницы. При этом последнюю страницу потребуется перевернуть вручную; - marker — автоматическая печать маркера на обороте всех страниц документа; - fonarik — автоматическая печать маркера на обороте последней страницы. Будут созданы два задания на печать: задание печати страниц документа с проставленными маркерами без последней страницы и задание из двух страниц (последняя страница документа и маркер на обороте последней страницы) для автоматической двусторонней печати
MacEnable	on	Включает поддержку меток безопасности в CUPS
MacJournal	off	Включает журнал маркировки. По умолчанию журнал записывается в базу данных SQLITE /var/spool/cups/parsec/marking-journal.sqlite
MacConvertToPDF	on	Преобразовывать унаследованные задания в pdf-формат (для устранения проблем с печатью шрифтов на некоторых PostScript-принтерах)
MacHighIntAdmin	off	Требовать высокую целостность для операций администрирования (редактирование принтеров, перенос задания на другой принтер и т.д.). Работает только для локальных сокетов

12.3. Настройка маркера печати

Настройка элементов маркировки осуществляется редактированием файлов шаблонов:

- /etc/cups/marker.template — шаблон маркера, описывающий элементы маркировки на первой странице, последующих страницах и на обороте последней страницы. Файл шаблона по умолчанию /usr/share/cups/marker.template устанавливается вместе с пакетом и может использоваться при необходимости вернуть комплекс программ печати и маркировки документов в исходное состояние;
- /etc/cups/cups-marker-vars.conf — конфигурационный файл переменных маркировки, значения которых будут запрошены у пользователя перед маркировкой. Файл /usr/share/cups/cups-marker-vars.conf.default содержит описание по умолчанию, устанавливается вместе с пакетом и может использоваться при

необходимости вернуть комплекс программ печати и маркировки документов в исходное состояние;

- /usr/share/cups/psmarker/marker.defs — файл описания положения элементов маркера на странице;

- /usr/share/cups/fonarik/fonarik.defs — файл описания положения элементов маркера на обороте последней страницы;

- /usr/share/cups/fonts/*.t42 — шрифты для маркировки;

- /usr/share/cups/charsets/utf-8.*, /usr/share/cups/charsets/utf-8 — наборы символов для различных шрифтов маркировки.

Редактирование конфигурационных файлов и установка шрифтов возможны только локально на сервере печати через механизм sudo или с помощью утилиты fly-admin-marker совместно с механизмом KAuth.

Редактирование файлов fonarik.defs и marker.defs возможно только от имени администратора через механизм sudo.

Описание графической утилиты fly-admin-marker см. в электронной справке.

12.3.1. Файл описания переменных маркировки

При маркировке документа используются переменные маркировки. Переменные маркировки могут быть основными и пользовательскими и должны быть описаны в конфигурационном файле /etc/cups/cups-marker-vars.conf. Их значение запрашивается у пользователя перед маркировкой.

ВНИМАНИЕ! Для описания переменных маркировки необходимо редактировать файл /etc/cups/cups-marker-vars.conf. Файл /usr/share/cups/cups-marker-vars.conf.default создается при установке пакета и может быть перезаписан в процессе обновления системы.

Переменные маркировки описываются в следующем виде:

```
атрибут:имя_переменной:значение_по_умолчанию:2 #необязательная переменная
атрибут:имя_переменной:значение_по_умолчанию:1 #обычная переменная
атрибут:имя_переменной:значение_по_умолчанию:0 #скрытая переменная
```

Основные переменные заданы по умолчанию в программе маркировки и не требуют их настройки. Для основных переменных не требуется указывать их имена в конфигурационном файле.

Пользовательские переменные могут быть созданы пользователем и требуют задания имени в конфигурационном файле.

Каждой переменной маркировки в конфигурационном файле /etc/cups/cups-marker-vars.conf должен соответствовать атрибут задания.

Атрибуты заданий основных переменных приведены в таблице 60.

Таблица 60

Атрибут	Описание
mac-distribution	Список рассылки
mac-inv-num	Инвентарный номер
mac-owner-phone	Телефон исполнителя
mac-workplace-id	Идентификатор рабочего места
mac-job-mac-level-name	Степень секретности (название текстом)
mac-job-mac-level-reason	Основание для степени секретности (пункт перечня)
mac-job-user-full-name	Полное имя пользователя, выполнившего маркировку (значение пункта «отпечатал»)
mac-job-originating-user-full-name	Полное имя пользователя, создавшего задание (значение пункта «исполнитель»)

Имя атрибута задания пользовательской переменной должно иметь префикс mac-. Переменные с атрибутами без данного префикса будут игнорироваться при запросе у пользователя.

Для каждой переменной маркировки в файле /etc/cups/cups-marker-vars.conf указывается ее тип и значение по умолчанию.

Переменная маркировки может иметь следующий тип:

- «1» — обычная переменная. Обязательное для ввода значение. Запрашивается у пользователя, если значение не введено пользователем, то заполняется значением по умолчанию;
- «2» — необязательная (опциональная) переменная. Запрашивается у пользователя, но при этом не обязательное к вводу значение. Если значение не введено пользователем, то соответствующему атрибуту присваивается значение «не указано»;
- «0» — скрытая переменная. Не запрашивается у пользователя и соответствующему атрибуту задания присваивается значение по умолчанию.

Значение по умолчанию может быть пустым или строкой. Также в качестве значения по умолчанию может использоваться одна из служебных переменных, перечень которых приведен в таблице 61.

Таблица 61

Переменная	Описание
{LABEL_NAME}	Метка безопасности задания в формате уровень:категория
{LABEL_NAME_FULL}	Метка безопасности задания в полном текстовом виде

Окончание таблицы 61

Переменная	Описание
{LEVEL_NAME}	Уровень метки задания
{ILEVEL_NAME}	Уровень целостности задания
{CATEGORY_NAME}	Категория задания
{GECOS_OWNER_NAME}	Полное имя пользователя, создавшего задание (подставляется из GECOS). При отсутствии данных в GECOS используется имя учетной записи пользователя (логин)
{GECOS_REQUESTING_USER_NAME}	Полное имя пользователя, выполняющего маркировку (подставляется из GECOS). При отсутствии данных в GECOS используется имя учетной записи пользователя (логин)
{GECOS_OWNER_PHONE}	Телефон пользователя, создавшего задание (подставляется из GECOS). При отсутствии данных в GECOS заменяется пустой строкой
{атрибут_задания}	Значение атрибута задания (текстового или числового). Могут быть использованы только те атрибуты, которые присвоены до маркировки. Обычно это стандартные атрибуты. Например, {job-name}

К служебным переменным также относятся стандартные атрибуты задания, значения которым присваиваются до маркировки. Перечень стандартных атрибутов приведен в таблице 62.

Таблица 62

Атрибут	Описание
job-name	Имя задания
copies	Число копий
mac-job-derive	Число унаследованных заданий
mac-job-mac-label	Метка безопасности задания
mac-job-marked	Определяет, выполнена ли маркировка
mac-job-origin	Номер исходного задания, к которому относится унаследованное задание
mac-job-user-name	Имя пользователя, выполнившего маркировку
mac-marking-session-id	Идентификатор сессии
job-originating-user-name	Имя пользователя, создавшего задание

Примечание. В названиях переменных и значениях по умолчанию в файле /etc/cups/cups-marker-vars.conf необходимо использовать экранирование специального символа «:» (двоеточие), используемого для разделения полей, путем замены данного символа на «\:». Кроме того, экранирование используется при подстановке значений встроенных переменных, что учитывается клиентскими программами

Пример

Файл cups-marker-vars.conf

```
#основные переменные
mac-inv-num:::1
mac-owner-phone::{GECOS_OWNER_PHONE}:1
mac-workplace-id:::1
mac-distribution:::2
mac-job-mac-level-name::{LEVEL_NAME}:0
mac-job-mac-level-reason:::0
mac-job-originating-user-full-name::{GECOS_OWNER_NAME}:0
mac-job-user-full-name::{GECOS_REQUESTING_USER_NAME}:0
#пользовательские
mac-myvar:Название переменной:значение по умолчанию:0
```

12.3.2. Шаблон маркера

Шаблон определяет внешний вид маркеров на странице, а также содержит параметры, используемые программами psmarker и fonarik в процессе маркировки.

ВНИМАНИЕ! Для изменения маркера необходимо редактировать файл /etc/cups/marker.template. Файл /usr/share/cups/marker.template создается при установке пакета и может быть перезаписан в процессе обновления системы.

В шаблоне в начале файла задаются параметры маркировки:

- fonarik_boundary — граница смены маркера на обороте последней страницы (количество копий);
- replace_underscrore — преобразование символа нижнего подчеркивания «_» в пробел в названии уровней и категорий (0 — отключить, 1 — включить);
- charset — набор символом маркировки, позволяет изменить шрифт маркировки.

Далее построчно задаются маркеры в виде:

```
страница:начертание:размер:семейство:расположение:строка_маркировки
```

где:

1) страница — страница, на которой располагается данная строка. Возможные значения:

а) fonarik — страница маркера на обороте, если число копий меньше или равно значению fonarik_boundary;

б) fonarik_gt_5 — страница маркера на обороте, если число копий больше значения fonarik_boundary;

в) first — первая страница документа;

г) any — страницы документа, кроме первой и последней;

д) last — последняя страница документа;

2) начертание — начертание шрифта маркировки. Возможные значения:

а) normal — нормальное начертание;

б) bold — жирное начертание;

в) italic — наклонное начертание;

г) bolditalic — жирное наклонное начертание;

3) размер — размер шрифта маркировки;

4) семейство — шрифт, не реализовано для строки маркировки, используется шрифт, заданный параметром charset;

5) расположение — расположение данной строки на странице. Возможные значения:

а) top-left — вверху слева;

б) top-center — вверху по центру;

в) top-right — вверху справа;

г) bottom-left — внизу слева;

д) bottom-center — внизу по центру;

е) bottom-right — внизу справа;

6) строка_маркировки — строка маркировки, которая будет добавлена в документ. Может содержать как обычный текст, так и переменные, которые заменяются значениями в процессе маркировки. Переменные указываются в фигурных скобках {}. Переменные бывают нескольких типов:

а) встроенные переменные — записываются прописными буквами. Заданы в CUPS и программах маркировки psmarker и fonarik. Данные переменные подставляются автоматически и не могут быть изменены пользователем перед маркировкой. Полный список встроенных переменных приведен в таблице 63.

Таблица 63

Переменная	Описание
{CURRENT_COPY}	Текущая копия документа, отправленная на печать
{CURRENT_PAGE}	Текущая страница документа
{NUM_PAGES}	Число страниц документа
{LABEL_NAME}	Метка безопасности задания в формате Уровень:Категория
{LABEL_NAME_FULL}	Полная метка безопасности задания в формате Уровень:Категория:Целостность
{LEV_NAME}	Уровень задания
{ILEV_NAME}	Уровень целостности задания

Окончание таблицы 63

Переменная	Описание
{CAT_NAME}	Категория задания
{DATE}	Текущая дата в формате ДД.ММ.ГГГГ
{TIME}	Текущее время в формате ЧЧ:ММ:СС

б) атрибуты задания — записываются строчными буквами и должны соответствовать атрибутам задания. Перечень атрибутов задания приведен в 12.3.1. Значение атрибута может быть изменено пользователем перед маркировкой (при наличии описания атрибута в конфигурационном файле /etc/cups/cups-marker-vars.conf). Полный список атрибутов задания на печать можно посмотреть путем просмотра атрибутов данного задания в графической утилите fly-admin-printer;

в) сценарии — задаются в виде {SCRIPT:/path/myscript.sh}. При маркировке переменная будет заменена выводом сценария. Сценарий должен выводить одну строку через стандартный поток (stdout) и быть доступным для выполнения пользователем root;

г) атрибуты пользователя FreeIPA — задаются в виде {FREEIPA:sn}, где sn — имя атрибута FreeIPA пользователя, который создал задание. Посмотреть полный список доступных атрибутов можно командой ipa user-show printuser --raw --all.

Пример

Шаблон /etc/cups/marker.template

```
fonarik_boundary=5
replace_underscore=0
charset=utf-8

fonarik:normal:12:Arial:top-left:{mac-inv-num}
fonarik:normal:12:Arial:top-left:Экземпляров {copies}
fonarik:normal:12:Arial:top-left:Количество страниц {NUM_PAGES}
fonarik:normal:12:Arial:top-left:{mac-workplace-id}
fonarik:normal:12:Arial:top-left:{mac-distribution}
fonarik:normal:12:Arial:top-left:Исполнитель {mac-job-originating-user-full-name}
fonarik:normal:12:Arial:top-left:Тел. {mac-owner-phone}
fonarik:normal:12:Arial:top-left:Отпечатал {mac-job-user-full-name}
fonarik:normal:12:Arial:top-left:{DATE}
```

```
fonarik_gt_5:normal:12:Arial:top-left:Исполнитель
{mac-job-originating-user-full-name}
fonarik_gt_5:normal:12:Arial:top-left:Тел. {mac-owner-phone}

first:normal:12:Arial:top-right:{LABEL_NAME}
first:normal:12:Arial:top-right:Экз. {CURRENT_COPY}
first:normal:12:Arial:bottom-right:{mac-inv-num}
any:normal:12:Arial:bottom-right:{mac-inv-num}
last:normal:12:Arial:bottom-right:{mac-inv-num}
first:normal:12:Arial:top-center:{CURRENT_PAGE} из {NUM_PAGES}
any:normal:12:Arial:top-center:{CURRENT_PAGE} из {NUM_PAGES}
last:normal:12:Arial:top-center:{CURRENT_PAGE} из {NUM_PAGES}
```

12.3.3. Файлы описания маркера

Для изменения положения маркера, проставляемого на первой и последующих страницах, необходимо в файле `/usr/share/cups/psmarker/marker.defs` изменить значения параметров (единица измерения каждого параметра — типографский пункт):

- `MarkerTopShift` — отступ от верхней границы страницы;
- `MarkerBottomShift` — отступ от нижней границы страницы;
- `MarkerLeftShift` — отступ от левой границы страницы;
- `MarkerRightShift` — отступ от правой границы страницы;
- `MarkerStringInterval` — интервал между строками.

Для изменения положения маркера, проставляемого на обороте последней страницы, необходимо в файле `/usr/share/cups/fonarik/fonarik.defs` изменить значения параметров для соответствующего формата и ориентации страницы (единица измерения каждого параметра — типографский пункт):

- `FonarikTopShift` — отступ от верхней границы страницы;
- `FonarikBottomShift` — отступ от нижней границы страницы;
- `FonarikLeftShift` — отступ от левой границы страницы;
- `FonarikRightShift` — отступ от правой границы страницы;
- `FonarikStringInterval` — интервал между строками;
- `A0, ..., A5` — печатный формат страницы A0, ..., A5;
- `L, P` — ориентация страницы: P — вертикальная, L — горизонтальная.

Пример

Содержание файла `/usr/share/cups/fonarik/fonarik.defs`

```
...
A4:P:FonarikTopShift=520.0
```

```
A4:P:FonarikBottomShift=20.0
A4:P:FonarikLeftShift=36.0
A4:P:FonarikRightShift=36.0
A4:P:FonarikStringInterval=4.0
...
```

Любые изменения содержания и формата маркера страниц может производить только администратор.

12.3.4. Изменение шрифта маркировки

Шрифт маркировки задается файлом сопоставления шрифтов в шаблоне маркировки. Для каждого шрифта имеется свой файл сопоставления. Изменения шрифта маркировки производится путем указания в шаблоне маркировки для параметра `charset` соответствующего требуемому шрифту файла сопоставления, например:

```
charset=utf-8.PTAstraSans
```

ВНИМАНИЕ! Редактировать необходимо шаблон маркировки `/etc/cups/marker.template`. Если данный шаблон отсутствует, то скопировать установленный вместе с пакетом шаблона маркировки `/usr/share/cups/marker.template` в `/etc/cups/marker.template`.

Доступные файлы сопоставления находятся в `/usr/share/cups/charsets`. При выборе несуществующего файла сопоставления используется по умолчанию `utf-8`, что соответствует шрифту PT Astra Serif.

Возможно создать собственный шрифт маркировки на основе имеющегося шрифта в формате TrueType (`*.ttf`). Исходный шрифт должен содержать кириллицу и иметь четыре варианта начертания (нормальное, жирное, наклонное и жирное наклонное), т.е. состоять из четырех файлов с расширением `*.ttf`.

Все четыре файла необходимо преобразовать в формат Type 42 (расширение `*.t42`). Для этого необходимо:

1) установить пакет `fontforge`;

2) в каталоге с ttf-шрифтами создать сценарий `converter.pe` (если отсутствует) со следующим содержанием:

```
#!/usr/bin/fontforge

i=1
while ( i<$argc )
Open($argv[i])
RenameGlyphs("Adobe Glyph List")
Generate($argv[i]:r + ".t42")
```

```
i = i+1
endloop
```

3) сделать сценарий исполняемым, например, командой `chmod +x`;

4) запустить сценарий `converter.pe` на исполнение:

```
/home/user/converter.pe *.ttf
```

Полученные четыре файла с расширением `*.t42` необходимо скопировать в `/usr/share/cups/fonts`.

Далее создать файл сопоставления для нового шрифта в каталоге `/usr/share/cups/charsets`. Рекомендуется имя файла сопоставления задавать в виде `utf-8.<имя_шрифта>`, например, `utf-8.PTAstraSans`.

Пример

Файл сопоставления для шрифта PT Astra Sans

```
0000 00FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
0100 01FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
0200 02FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
0300 03FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
0400 04FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
1E00 1EFF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
2000 20FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
2100 21FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
2300 23FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
2400 24FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
2500 25FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
2600 26FF ltor single PTAstraSans-Regular.t42 PTAstraSans-Bold.t42
    PTAstraSans-Italic.t42 PTAstraSans-BoldItalic.t42
```

В каждой строке после `single` находится список из четырех файлов, разделенных пробелом, соответствующих каждому варианту написания в следующем порядке: обычный, жирный, наклонный и жирный наклонный.

Установку и изменения шрифта также можно осуществить с помощью графической утилиты `fly-admin-marker`.

Примечание. Возможны проблемы с некоторыми PostScript-принтерами из-за особенностей поддержки шрифтов в формате `Type 42`, поэтому по умолчанию задание на печать преобразуется в формат `*.pdf` сразу после маркировки. Для отключения данного действия необходимо выполнить команду:
`sudo cupsctl MacConvertToPDF=Off`
или в `cupsd.conf` для параметра `MacConvertToPDF` установить значение `Off`, затем перезапустить сервер CUPS.

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.

ВНИМАНИЕ! В качестве файловых систем на съемных машинных носителях информации должны использоваться только файловые системы ext2/ext3/ext4/XFS, поддерживающие расширенные (в т.ч. мандатные) атрибуты файловых объектов и обеспечивающие гарантированное уничтожение (стирание) информации.

Для ОС возможность санкционированного монтирования 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.

В режиме «Мобильный» реализовано управление блокировкой подключаемых устройств с помощью утилиты USBGuard.

Рекомендации по использованию указанных средств приведены в документе РУСБ.10015-37 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)[1)] .

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

[1)] В режиме «Мобильный» реализовано управление блокировкой подключаемых устройств.

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 службы FreeIPA или ALD. На рабочих местах пользователей в ОС должен быть установлен клиентский пакет соответствующей службы и выполнены действия по настройке (см. документ РУСБ.10015-37 95 01-1).

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

1) создать набор уровней конфиденциальности;
2) создать набор категорий конфиденциальности;
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 (регистрации событий) в ОС.

ВНИМАНИЕ! Для изменения максимального мандатного контекста объектов (см. 4.2) необходимо внести изменения в сценарий `pdp-init-fs`, размещенный в каталоге `/usr/sbin`, переопределив значения переменных:

- `sysmaxlev` – максимальный уровень конфиденциальности;
- `sysmaxilev` – максимальный уровень целостности;
- `sysmaxcat` – максимальный набор категорий конфиденциальности.

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

В режиме ЕПП указанные действия выполняются:

- если установлена служба ALD — при помощи графической утилиты `fly-admin-smc` или утилиты командной строки `ald-admin` (подробная информация по использованию инструмента командной строки доступны в справочной странице `man ald-admin`);
- если установлена служба FreeIPA — при помощи web-интерфейса FreeIPA или утилиты командной строки `ipa user-mod` (подробная информация по ис-

пользованию инструмента командной строки доступны в справочной странице `ipa help user-mod`).

В локальном режиме указанные действия выполняются при помощи графической утилиты `fly-admin-smc` или при помощи соответствующих утилит командной строки (см. 4.15) от имени учетной записи администратора (см. раздел 1) через механизм `sudo`. Более подробное описание графических утилит см. в электронной справке.

16. ОГРАНИЧЕНИЕ ПРОГРАММНОЙ СРЕДЫ И ФУНКЦИИ БЕЗОПАСНОСТИ

16.1. Замкнутая программная среда

16.1.1. Режимы функционирования

Инструменты из состава ОС предоставляют возможность создавать замкнутую программную среду (ЗПС). Использование ЗПС обеспечивает динамический контроль неизменности (целостности) и подлинности файлов и предназначено для выявления фактов несанкционированного изменения файлов формата ELF и других файлов (в т. ч. относящихся к КСЗ) и предотвращения загрузки измененных файлов формата ELF и открытия других измененных файлов.

Контроль целостности реализован в невыгружаемом модуле ядра ОС `digsig_verif` и производится на основе проверки ЭЦП[1)], внедренной в исполняемые файлы и разделяемые библиотеки формата ELF, входящие в состав ОС и устанавливаемого СПО, и в расширенные атрибуты файловой системы. ЭЦП реализована в соответствии с ГОСТ Р 34.10-2012 (256 бит) и ГОСТ Р 34.10-2001 (256 бит), использование ГОСТ Р 34.10-2001 в ОС сохранено для совместимости.

Контроль целостности исполняемых файлов и разделяемых библиотек формата ELF 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).

Контроль целостности файлов на основе ЭЦП в расширенных атрибутах файловой системы 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.

[1)] Электронная цифровая подпись — строка бит, полученная в результате процесса формирования подписи (применяется для подписи средствами ОС исполняемых файлов с использованием функции хеширования на базе асимметричного криптографического алгоритма.

Включение ЗПС может быть выполнено в процессе установки ОС путем выбора соответствующего пункта программы установки ОС.

Включение и выключение ЗПС после установки ОС выполняется с помощью инструмента `astra-digsig-control`, описанного в 16.5.29.

16.1.2. 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
```

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

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

В модуль `digsig_verify` встроены открытые ключи изготовителя, 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 изготовителя) — 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 (16.1.3), 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 изготовителя;
- файлы ключей в /etc/digsig/keys/subdirectory могут быть подписаны и встроенными ключами изготовителя, 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 изготовителя.

Пример

```
/etc/digsig/keys/key1.gpg - публичный ключ 1, подписанный
на первичном ключе изготовителя
/etc/digsig/keys/key2.gpg - публичный ключ 2, подписанный
на первичном ключе изготовителя
/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
```

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 системы, 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
```

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.

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 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
```

16.1.3. Подписывание файлов

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 (GnuPG). Модифицированный GnuPG выводит ГОСТ Р 34.11-94 в списке доступных алгоритмов. Для получения списка доступных алгоритмов необходимо выполнить команду:

```
gpg --version
```

Результат выполнения команды:

```
gpg (GnuPG) 1.4.18
Copyright (C) 2014 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Home: ~/.gnupg
Поддерживаются следующие алгоритмы:
С открытым ключом: RSA, RSA-E, RSA-S, ELG-E, DSA,
           GOST_R 34.10-2001, GOST_R 34.10-2012
Симметричные: 3DES, CAST5, BLOWFISH,
```

```
          AES, AES192, AES256, TWOFISH,
          CAMELLIA128, CAMELLIA192, CAMELLIA256
хеш-функции: MD5, SHA1, RIPEMD160, SHA256, SHA384, SHA512,
          SHA224, GOST_R34.11-2012, GOST_R34.11-94
Алгоритмы сжатия: Без сжатия, ZIP, ZLIB,
          BZIP2
```

Пример

Для создания дополнительного ключа и его использования для подписывания СПО необходимо:

1) создать ключевую пару `GOST_R 34.10-2001` и сохранить в каталоге `~/.gnupg`.

Для создания ключевой пары выполнить следующую команду:

```
gpg --full-generate-key
```

В процессе выполнения команды отвечать на запросы системы. На запрос системы выбрать тип ключа следует указать пункт, соответствующий ГОСТ Р 34.10-2001.

Вывод команды:

```
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:
"Name Surname (position) <surname@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) выполнить экспорт ключа пользователя в файл командой:

```
keys@debian:$ gpg --export "Test GOST R 34.10-2012 Secondary Key
<test@gost.secondary.key>" > /tmp/secondary_gost_key.gpg
```

3) подписать ключ пользователя, выполнив команды:

```
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) подписать файл формата ELF ключом пользователя с использованием утилиты `bsign` (по умолчанию подпись внедряется в том числе и в расширенные атрибуты файла):

```
keys@debian:~$ bsign --sign test_elf
```

5) подписать ключом пользователя с использованием утилиты `bsign` произвольный файл (с внедрением подписи только в расширенные атрибуты):

```
keys@debian:~$ bsign --sign --xattr test_elf
```

Дополнительная информация по использованию утилиты `bsign` приведена в `man bsign`;

6) для проверки правильности ЭЦП файла формата ELF выполнить команду:

```
keys@debian:~$ bsign -w test_elf
```

7) скопировать в каталог /etc/digsig/ дополнительный ключ пользователя, подписанный на главном ключе (см. 16.1.2);

8) проверить, что подписанный файл формата ELF может выполняться:

```
keys@debian:~$ ./test_elf
hello world!
keys@debian:~$
```

16.2. Режим Киоск-2

Режим Киоск-2 — это инструмент системы PARSEC для ограничения возможностей, предоставляемых непривилегированным пользователям. Работу режима Киоск-2 обеспечивает пакет parsec-kiosk2.

16.2.1. Профили пакета parsec-kiosk2

Профили пользователей parsec-kiosk2 располагаются в каталоге /etc/parsec/kiosk2 и служат для применения ограничений действий пользователей. Профиль пользователя — это текстовый файл, имя которого совпадает с именем пользователя. Один профиль пользователя служит для установки ограничений только для одного пользователя.

Если при включенном режиме Киоск-2 в /etc/parsec/kiosk2 отсутствует профиль пользователя, то права данного пользователя в ОС не ограничиваются.

ВНИМАНИЕ! Если для пользователя создан пустой профиль, то данному пользователю запрещены любые действия.

Системные профили parsec-kiosk2 располагаются в каталоге /etc/parsec/kiosk2-profiles и также служат для применения ограничений действий пользователей. Содержимое файла системного профиля представляет собой список файлов и профилей приложений с абсолютными или относительными путями, а также с информацией о правах доступа и владельцах этих файлов. Системный профиль можно добавить в профили нескольких пользователей, тогда для всех этих пользователей будет доступно использование приложения в соответствии с ограничениями в правах доступа и владельцах, указанных в системном профиле данного приложения.

16.2.2. Синтаксис профилей parsec-kiosk2

Синтаксис разрешения на доступ к файлу:

```
+file rwc u/o: <имя_файла>
```

или на доступ к ссылке:

```
+link rwc uo: <имя_файла>
```

где `rwc` — это доступ к файлу с правами на чтение (r), запись (w) и создание (c), которые указываются при наличии соответствующего разрешения;

`u` — доступ владельцу файла;

`o` — доступ остальным пользователям;

`<имя_файла>` — файл, права доступа к которому назначаются. При этом имя файла для ссылки должно соответствовать существующей символьной ссылке. Целевой файл для ссылки, который не обязан существовать в момент загрузки профиля, добавляется в список доступных.

Примечание. В синтаксисе профилей режима Киоск-2 можно использовать кириллицу.

В синтаксисе Киоск-2 также могут быть использованы метасимволы, например, в имени файла (см. таблицу 64).

ВНИМАНИЕ! Метасимволы в имени файла для ссылки не интерпретируются.

Таблица 64

Метасимвол	Описание
`**`	Ноль или более произвольных символов, включая символ «/»
`*`	Ноль или более произвольных символов, исключая символ «/»
`?`	Один произвольный символ, исключая символ «/»
`[набор_символов]`	Один из символов, принадлежащий набору. В наборе любые символы теряют свое специальное значение. Символ «]» может входить в набор только на первой позиции
`\d`	Одна десятичная цифра
`\D`	Одна или более десятичная цифра
`\h`	Одна шестнадцатеричная цифра
`\H`	Одна или более шестнадцатеричная цифра
`\xNN`	Байт с заданным значением
`\z`	Пустая строка или подстрока `(deleted)`, которая добавляется к еще не созданным или уже удаленным файлам. Данная конструкция используется при управлении созданием файла. Пример `+file wc u: /home/*/.config/example/example.conf\z` В этом случае разрешается доступ как к файлу `example.conf (deleted)`, так и к `example.conf` Если удалить `\z`, то доступ к `example.conf (deleted)` будет запрещен. Открытие существующего файла на запись будет успешным, а создание нового — нет
`\<любой_другой_символ>`	Соответствующий символ без специальной интерпретации, например, `\\,\[,\*)`

Для включения одного профиля в другой профиль используется `@include`:

```
@include other-profile-name
```

16.2.3. Синтаксис, совместимый с `parsec-kiosk`

В профилях режима Киоск-2 корректно распознаются строки, написанные по синтаксису режима киоска (устаревший инструмент для ограничений действий пользователя). При этом при использовании синтаксиса режима киоска в режиме Киоск-2:

1) в имени файла спецсимволы не интерпретируются;
2) любое из прав на чтение (r) или исполнение (x) преобразуется в право на чтение (r), а право на запись (w) преобразуется в права на запись (w) и создание (c);
3) правила одинаково применяются для доступа как владельцев, так и невладельцев;
4) строка обязательно должна начинаться с символа «"» или «/»;
5) если файл представляет символьную ссылку, то режим Киоск-2 обрабатывает целевой файл ссылки;
6) имя профиля должно начинаться только с латинской буквы, а также в нем должен отсутствовать символ «/».

Если профиль, созданный в режиме киоска, отредактировать в `fly-admin-kiosk`, то он сохранится с использованием синтаксиса режима Киоск-2.

16.2.4. Работа с Киоск-2 через консоль

16.2.4.1. Включение и выключение Киоск-2

При первом запуске ОС режим Киоск-2 выключен. Чтобы включить режим контроля доступа Киоск-2 и применять ограничения на работу с файлами, требуется выполнить команду:

```
echo 1 | sudo tee /sys/module/parsec/parameters/uc_enforce
```

Для включения протоколирования попыток нарушений установленных фильтров доступа выполнить команду:

```
echo 1 | sudo tee /sys/module/parsec/parameters/uc_complain
```

ВНИМАНИЕ! Результаты применения данных команд будут действительны до перезагрузки ОС.

Для включения режима контроля доступа Киоск-2 и протоколирования с сохранением данного состояния после перезагрузки требуется, соответственно, выполнить команды:

```
echo 1 | sudo tee /etc/parsec/kiosk2_enforce
echo 1 | sudo tee /etc/parsec/kiosk2_complain
```

и перезагрузить ОС.

Результаты применения этих команд будут действительны после перезагрузки ОС.

Для выключения режима контроля доступа Киоск-2 и протоколирования в соответствующих командах требуется заменить `1` на `0`.

Для того чтобы избирательно включать режим Киоск-2 и исключать выбранных пользователей, требуется:

1) создать каталог `disabled` в `/etc/parsec/kiosk2`:

```
cd /etc/parsec/kiosk2
sudo mkdir disabled
```

2) перенести в каталог `disabled` профили тех пользователей, для которых не должны применяться ограничения:

```
sudo mv <имя_пользователя1> <имя_пользователя2> disabled
```

3) включить режим Киоск-2:

```
echo 1 | sudo tee /sys/module/parsec/parameters/uc_enforce
```

После выполнения данных действий ограничения режима Киоск-2 будут применены ко всем пользователям, кроме <имя_пользователя1>, <имя_пользователя2> и тех пользователей, для которых не были созданы профили. После перезагрузки ОС режим Киоск-2 вернется в состояние, указанное в `/etc/parsec/kiosk2_enforce`. При повторном включении режима Киоск-2 исключения для <имя_пользователя1>, <имя_пользователя2> и тех пользователей, для которых не были созданы профили, сохранятся и ограничения доступа к ним не будут применены.

16.2.4.2. Протоколирование процессов

Для того чтобы ограничения на работу приложений выполнялись корректно и без ошибок, требуется учитывать все зависимости файлов данного приложения. Поэтому при создании системного профиля приложения рекомендуется выполнить протоколирование процессов при работе с соответствующим приложением. Результат протоколирования процессов показывает все необходимые файлы приложения и пути к ним. Для выполнения протоколирования процессов требуется:

1) выключить режим контроля доступа Киоск-2:

```
echo 0 | sudo tee /sys/module/parsec/parameters/uc_enforce
```

2) включить протоколирование процессов:

```
echo 1 | sudo tee /sys/module/parsec/parameters/uc_complain
```

3) создать профиль пользователя:

```
true | sudo tee /etc/parsec/kiosk2/<имя_пользователя>
```

4) запустить сессию пользователя <имя_пользователя>:

```
su - <имя_пользователя>
```

5) запустить протоколируемое приложение;

6) выйти из сессии пользователя <имя_пользователя>:

```
exit
```

7) выключить протоколирование процессов:

```
echo 0 | sudo tee /sys/module/parsec/parameters/uc_complain
```

8) получить сообщения о запрещенных операциях:

```
sudo dmesg | grep PARSEC-UC:
```

16.2.4.3. Создание профиля

Для создания системного профиля протоколируемого приложения требуется в каталоге `/etc/parsec/kiosk2-profiles` создать файл, задав в качестве его имени название приложения, и вписать в него все пути с учетом синтаксиса из результатов выполнения команды `dmesg | grep PARSEC-UC:` в соответствии с перечислением 8) на странице 260.

Для создания профиля пользователя требуется в каталоге `/etc/parsec/kiosk2` создать файл с именем, совпадающим с именем данного пользователя.

16.2.5. Графическая утилита управления профилями

Для управления режимом Киоск-2 и профилями, расположенными в каталогах `/etc/parsec/kiosk2` и `/etc/parsec/kiosk2-profiles`, используется графическая утилита `fly-admin-kiosk`. Описание утилиты приведено в электронной справке.

ВНИМАНИЕ! Для работы с профилями режима киоска с использованием утилиты `fly-admin-kiosk` необходимо адаптировать их синтаксис и поместить профили в каталоги `/etc/parsec/kiosk2` и `/etc/parsec/kiosk2-profiles` пакета `parsec-kiosk2` соответственно (см. 16.2.3).

Для запуска утилиты выполнить в консоли команду:

```
sudo fly-admin-kiosk
```

или перейти «меню «Пуск» — Панель управления — Безопасность — Системный киоск».

16.2.6. Киоск Fly

Для ограничения возможности запуска программ локальными пользователями администратор может использовать графическую утилиту `fly-admin-smc`. Для этого необходимо в главном окне графической утилиты `fly-admin-smc` раскрыть список «Пользователи», выбрать пользователя и во вкладке «Графический киоск Fly» установить флаг «Режим графического киоска». Флаг включает режим графического киоска при работе с приложениями из списка. Если в списке одно приложение, то режим графического 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 графического киоска с помощью графической утилиты `fly-admin-smc` осуществляется администратором на максимальном уровне мандатного контроля целостности, установленном в ОС.

16.3. Изоляция приложений

Для обеспечения изолированного выполнения графических и консольных приложений в состав ОС входит инструмент Firejail . Целью применения Firejail является минимизация риска компрометации основной ОС при запуске недоверенных или потенциально уязвимых программ.

Для обеспечения изоляции приложений в Firejail используются:

- механизм пространств имен `namespaces`;
- фильтрация системных вызовов `seccomp-bpf`.

После запуска инструмент Firejail и все его дочерние процессы используют отдельные представления ресурсов ядра, таких как сетевой стек, таблица процессов и точки монтирования.

Firejail не требует подготовки системного образа: состав окружения формируется при запуске на основе содержимого текущей ФС, и удаляется после завершения работы приложения.

При использовании Firejail предоставляются средства задания правил доступа к ФС, позволяющие:

- определять файлы и каталоги, к которым разрешен или запрещен доступ;
- предоставлять доступ к файлам или каталогам только для чтения;
- подключать для данных временные ФС (`tmpfs`);
- совмещать каталоги через `bind-mount` и `overlayfs`.

При установке ОС установка инструмента Firejail выполняется автоматически.

В целях повышения безопасности при установке Firejail снимается бит `suid`. Для того чтобы с помощью Firejail можно было запускать приложения, необходимо включить этот бит, выполнив от имени администратора команду:

```
chmod u+s /usr/bin/firejail
```

В состав ОС входят профили изоляции системных вызовов для большинства популярных приложений, в том числе для Firefox, Chromium, VLC.

Для выполнения программы в режиме изоляции требуется указать имя приложения в качестве параметра для инструмента Firejail, например:

```
firejail firefox
```

Примечание. Для запуска браузера Firefox может потребоваться внести изменения в конфигурационный файл `/etc/firejail/firefox.profile`, заменив строку `seccomp` на строку:

```
seccomp.drop @clock,@cpu-emulation,@debug,@module,@obsolete,@raw-io,
@reboot,@resources,@swap,acct,add_key,bpf,fanotify_init,io_cancel,
io_destroy,io_getevents,io_setup,io_submit,ioprio_set,kcmp,keyctl,
```

mount,name_to_handle_at,nfsservctl,ni_syscall,open_by_handle_at,
personality,pivot_root,process_vm_readv,ptrace,remap_file_pages,
request_key,setdomainname,sethostname,syslog,umount,umount2,userfaultfd,
vhangup,vmsplice

16.4. Графический киоск в режиме «Мобильный»

В режиме «Мобильный» для ограничения возможности запуска пользователями графических приложений администратору требуется выполнить настройку киоска путем редактирования (или создания, при его отсутствии) файла /etc/mobile-kiosk/mobile-kiosk.conf. В файле следует указать имя учетной записи пользователя, для которого устанавливаются ограничения на запуск графических приложений, а также в параметре Applications перечислить desktop-файлы приложений, которые можно будет запускать указанному пользователю:

```
[<имя_пользователя>]
Applications=<приложение1>,<приложение2>,...
```

Если в Applications указано одно приложение, оно запустится автоматически после входа в сессию.

Пример

Включение режима киоска для пользователя user, разрешающее запуск браузеров Firefox и Chromium

```
[user]
Applications=firefox,chromium
```

16.5. Функции безопасности системы

Пакет astra-safepolicy содержит инструменты управления функциями безопасности системы. Описание инструментов приведено в 16.5.3-16.5.33.

Все инструменты из состава пакета astra-safepolicy поддерживают стандартный набор параметров вызова, приведенный в 16.5.1. Некоторые инструменты дополнительно к стандартному набору параметров вызова поддерживают дополнительные параметры. Описание дополнительных параметров приведено в описании соответствующего инструмента.

Инструменты пакета astra-safepolicy выступают в роли команд-переключателей, осуществляющих включение и выключение соответствующих функций безопасности. Для просмотра состояния некоторых команд-переключателей можно использовать инструмент astra-security-monitor из состава пакета astra-safepolicy, описание инструмента приведено в 16.5.2.

16.5.1. Общие параметры вызова переключателей

Все команды-переключатели, входящие в состав пакета `astra-safepolicy`, поддерживают стандартный набор параметров вызова. Синтаксис использования команды-переключателя:

```
<имя_команды-переключателя> <параметр>
```

Перечень параметров вызова приведен в таблице 65.

Таблица 65

Параметр	Описание
`enable`	Включить действие, выполняемое функцией безопасности
`disable`	Выключить действие, выполняемое функцией безопасности
`status`	Отобразить, если возможно, фактическое состояние переключателя на текущий момент времени. Результат выполнения команды: - `АКТИВНО` — системные ограничения применяются (функция выполняется); - `НЕАКТИВНО` — системные ограничения не применяются (функция отключена). Например, для некоторых команд-переключателей вызовы `enable/disable` меняют положение переключателя (см. параметр `is-enabled`), но для изменения состояния требуется перезагрузка. Для этих команд-переключателей до момента перезагрузки вывод `is-enabled` и вывод `status` будут различаться.
`is-enabled`	Отобразить положение переключателя. Результат выполнения команды: - `ВКЛЮЧЕНО` — системные ограничения включены; - `ВЫКЛЮЧЕНО` — системные ограничения выключены

Для некоторых инструментов при первом их включении создается соответствующий юнит службы systemd. В дальнейшем состояние данной функции безопасности после перезагрузки системы устанавливается в соответствии с положением переключателя путем запуска соответствующего юнита, если запуск юнита разрешен. Список юнитов, созданных инструментами, можно просмотреть в файле `astra-safepolicy.target` (в строках, начинающихся с `Wants=`).

16.5.2. Монитор безопасности

Утилита `astra-security-monitor` отображает информацию о состоянии некоторых функций безопасности, а также выводит информацию о состоянии функции безопасности по ее идентификатору.

Примечание. Отображение состояния функции безопасности также зависит от наличия установленных в системе соответствующих пакетов или служб.

Для функций безопасности, информацию о которых выводит утилита, возможны следующие состояния:

- `ВКЛЮЧЕНО` — функция безопасности активна;
- `ВЫКЛЮЧЕНО` — функция безопасности неактивна;

- ВКЛЮЧАЕТСЯ — функция безопасности находится в процессе включения или будет включена после перезагрузки, но в настоящий момент неактивна;

- ВЫКЛЮЧАЕТСЯ — функция безопасности находится в процессе выключения или будет выключена после перезагрузки, но в настоящий момент активна;

- ЧАСТИЧНО — функция безопасности включена, но не все параметры, контролируемые этой функцией, соответствуют заданным по умолчанию.

При отображении состояний отдельных функций безопасности учитывается следующее:

1) при выводе информации о состоянии МКЦ на файловой системе проверяется соответствие уровней целостности объектов ФС уровням целостности, указанным в конфигурационном файле `/etc/parsec/fs-ilev.conf`. При несоответствии уровней целостности выводится сообщение о количестве файловых объектов, уровень целостности которых не соответствует заданному в конфигурационном файле:

а) «ниже» — уровень целостности файлового объекта ниже заданного;

б) «выше» — уровень целостности файлового объекта выше заданного;

в) «норма» — уровень целостности файлового объекта соответствует заданному.

Список файловых объектов, уровень целостности которых не соответствует заданному в файле `/etc/parsec/fs-ilev.conf`, можно вывести командой:

```
sudo set-fs-ilev status -v
```

2) для функции проверки подписи в расширенных атрибутах `xattr` — включена ли проверка подписи не только в исполняемых файлах, но и в других файлах, которые подписаны в `xattr` соответствующей утилитой.

Дополнительно монитор безопасности выводит следующую информацию:

1) запрет входа `root` по ssh (если установлены средства удаленного подключения ssh) — запрет удаленного входа в систему пользователю `root`. По умолчанию `root` не может войти через ssh, функция запрета имеет состояние ВКЛЮЧЕНО;

2) безопасный вход в домен (если компьютер введен в домен) — значение ВКЛЮЧЕНО, если в файле `/etc/parsec/parsec.conf` параметр `login_local` имеет значение `admin` (вход для локального пользователя разрешен, если пользователь входит в группу `astra-admin`) или значение `no` (вход для локальных пользователей запрещен);

3) системный киоск — значение ВКЛЮЧЕНО, если системный киоск включен и настроен хотя бы для одного пользователя;

4) графический киоск — значение ВКЛЮЧЕНО, если графический киоск настроен хотя бы для одного пользователя.

Подробное описание инструмента приведено в `man astra-security-monitor`.

При работе с astra-security-monitor также может использоваться графическая утилита fly-admin-smc («Политика безопасности», раздел «Монитор безопасности»).

16.5.3. Установка квот на использование системных ресурсов

Инструмент astra-ulimits-control включает или выключает ограничения на использование некоторых ресурсов системы для предотвращения нарушений доступности системы в результате исчерпания ресурсов. Параметры системных ограничений задаются в файле /etc/security/limits.conf.

Параметры вызова, используемые данным инструментом, приведены в таблице 65.

Изменение режима ограничений не повлияет на уже вошедших в систему пользователей и будет применено только при следующем входе.

Описание инструмента приведено в man astra-ulimits-control.

Управление системными ограничениями также может осуществляться с помощью графической утилиты fly-admin-smc.

Дополнительные ограничения на получение доступа и выполнение кода на уровне ядра ОС для привилегированного пользователя обеспечиваются модулем lockdown в соответствии с 16.7.

16.5.4. Режим запрета установки бита исполнения

В ОС реализован режим запрета установки бита исполнения, обеспечивающий предотвращение несанкционированного запуска исполняемых файлов и сценариев для командной оболочки.

ВНИМАНИЕ! В режиме запрета установки бита исполнения данная операция запрещена для пользователей ОС, включая пользователей из группы astra-admin. Запрет не распространяется на привилегированного пользователя root.

При включенной в системе данной функции безопасности установка пакетов программ, создающих в ФС файлы с битом исполнения, будет завершаться с ошибкой.

Параметры вызова, используемые данным инструментом, приведены в таблице 65.

Изменение режима запрета вступает в действие немедленно.

Описание инструмента приведено в man astra-nochmodx-lock.

Управление режимом запрета установки бита исполнения также может осуществляться с помощью графической утилиты fly-admin-smc.

16.5.5. Блокировка консоли для пользователей

Инструмент astra-console-lock осуществляет блокировку доступа к консоли и терминалам для пользователей, не входящих в группу astra-console. Параметры вызова, используемые данным инструментом, приведены в таблице 65.

Если при включении блокировки группа `astra-console` отсутствует в ОС, то она будет создана автоматически. При этом в нее будут включены пользователи, состоящие в группе `astra-admin` на момент включения этой функции.

Изменение блокировки вступает в действие немедленно.

Описание инструмента приведено в `man astra-console-lock`.

Управление блокировкой консоли для пользователей также может осуществляться с помощью графической утилиты `fly-admin-smc`.

16.5.6. Блокировка консоли в режиме «Мобильный»

В режиме «Мобильный» для блокировки доступа к консоли необходимо заблокировать доступ к консоли для пользователей, не входящих в группу `astra-console`, с помощью инструмента `astra-console-lock` (см. 16.5.5).

Дополнительно необходимо заблокировать использование командной строки из меню поиска, вызываемого комбинацией клавиш **<ALT+F2>**. Для этого требуется в файл `/etc/xdg/kdeglobals` добавить следующие строки:

```
[KDE Action Restrictions][$i]
run_command=false
```

Блокировка комбинации клавиш **<ALT+F2>** осуществляется для всех пользователей, в т.ч. для администратора.

Для разблокировки комбинации клавиш **<ALT+F2>** требуется в файле `/etc/xdg/kdeglobals` для параметра `run_command` указать значение `true`:

```
[KDE Action Restrictions][$i]
run_command=true
```

или удалить из файла строки:

```
[KDE Action Restrictions][$i]
run_command=false
```

16.5.7. Блокировка интерпретаторов

В ОС реализована функция блокировки следующих интерпретаторов:

- Python;
- Perl;
- Expect;
- Ruby;
- dash;
- irb;
- csh;
- lua;
- ksh;

- tcl;
- tk;
- zsh;
- bash;
- nodejs;
- php.

Для блокировки интерпретаторов (кроме интерпретатора bash) используется инструмент `astra-interpreters-lock`.

Для блокировки интерпретатора bash используется инструмент `astra-bash-lock`.

ВНИМАНИЕ! Блокировка интерпретатора bash может ограничить функционал некоторых программ и служб, не очевидным образом использующих bash, что приведет к нарушению штатной работы ОС.

ВНИМАНИЕ! После блокировки интерпретатора bash консольный вход пользователей с оболочкой bash будет невозможен.

Параметры вызова, используемые данными инструментами, приведены в таблице 65.

При включении блокировки интерпретаторов блокируется несанкционированное использование интерпретатора для выполнения кода напрямую из командной строки или из неименованного канала (pipe). При этом сценарии, написанные для этих интерпретаторов, выполняются в штатном режиме (у файла сценария при этом должны быть выставлены права на выполнение).

ВНИМАНИЕ! Блокировка запуска интерпретаторов должна использоваться совместно с инструментом `astra-nochmodx-lock`, т.к. в противном случае пользователь может сам создать сценарий, назначить ему бит исполнения и запустить, обойдя таким образом блокировку интерпретаторов.

Блокировка интерпретаторов не распространяется на пользователей из группы `astra-admin`.

Изменение блокировки вступает в действие немедленно.

Описание инструментов приведено в `man astra-interpreters-lock` и `man astra-bash-lock`.

Управление блокировкой интерпретаторов также может осуществляться с помощью графической утилиты `fly-admin-smc`.

16.5.8. Блокировка макросов

Блокировка исполнения макросов в документах LibreOffice осуществляется с помощью инструмента командной строки `astra-macros-lock`.

Параметры вызова, используемые данным инструментом, приведены в таблице 65.

При включении блокировки макросов уровень безопасности запуска макросов LibreOffice («Сервис — Параметры — LibreOffice — Безопасность — Безопасность макросов») устанавливается в значение «Очень высокий», при котором запуск макросов возможен только из библиотек, расположенных в доверенном каталоге `/opt/libreoffice` (каталог создается вручную). До выключения блокировки изменение уровня защищенности и доверенного источника недоступно, в .т.ч. администратору.

Изменение блокировки вступает в действие немедленно. Если изменение блокировки осуществлялось при запущенной программе пакета libreoffice, то оно вступает в действие при следующем запуске программы.

Описание инструмента приведено в `man astra-macros-lock`.

Управление блокировкой исполнения макросов также может осуществляться с помощью графической утилиты `fly-admin-smc`.

16.5.9. Блокировка трассировки ptrace

Блокировка трассировки `ptrace` осуществляется с помощью инструмента командной строки `astra-ptrace-lock`, который устанавливает максимальные ограничения на использование механизма `ptrace` путем задания параметру `kernel.yama.ptrace_scope` значения `3`.

Параметры вызова, используемые данным инструментом, приведены в таблице 65.

Значение устанавливается сразу при включении данной функции и сохраняется после перезагрузки.

Функция не может быть выключена без перезагрузки.

Управление блокировкой трассировки `ptrace` также может осуществляться с помощью графической утилиты `fly-admin-smc`.

Дополнительные возможности по ограничению применения трассировки процессов ОС обеспечиваются модулем безопасности yama в соответствии с 16.6.

16.5.10. Блокировка клавиши <SysRq>

Блокировка клавиши **<SysRq>** осуществляется с помощью инструмента командной строки `astra-sysrq-lock` путем изменения значения параметра `kernel.sysrq`. Значение сохраняется в файле `/etc/sysctl.d/999-astra.conf`. При этом блокируются функции системы, доступные при нажатии клавиши **<SysRq**>.

Параметры вызова, используемые данным инструментом, приведены в таблице 65.

По умолчанию данная функция безопасности включена, т.е. клавиша **<SysRq>** не работает.

ВНИМАНИЕ! Если в командной строке ядра указан параметр `sysrq_always_enabled`, то при включении функции блокировки клавиши **<SysRq>**

вызов status будет выводить НЕАКТИВНО, а вызов is-enabled будет выводить ВКЛЮЧЕНО.

Для управления блокировкой клавиши **<SysRq>** может непосредственно использоваться инструмент sysctl:

1) для включения блокировки клавиши **<SysRq>** команда:

```
sysctl -w kernel.sysrq=0
```

2) для выключения блокировки клавиши **<SysRq>** команда:

```
sysctl -w kernel.sysrq=1
```

3) для проверки состояния используется команда:

```
cat /proc/sys/kernel/sysrq
```

результат выполнения команды:

- 0 — блокировка клавиш SysRq включена;
- 1 — блокировка клавиш SysRq выключена.

Управление блокировкой клавиши **<SysRq>** также может осуществляться с помощью графической утилиты fly-admin-smc.

Примечание. Для управления блокировкой клавиши **<SysRq>** рекомендуется использовать инструмент astra-sysrq-lock или графическую утилиту fly-admin-smc.

16.5.11. Управление автоматическим входом

Управление автоматическим входом в графическую среду осуществляется с помощью инструмента astra-autologin-control.

Параметры вызова, используемые данным инструментом, приведены в таблице 65. При использовании вызова enable требуется дополнительный параметр <имя_пользователя>, от имени которого будет выполняться автоматический вход в систему после загрузки.

Описание инструмента приведено в man astra-autologin-control.

16.5.12. Блокировка запуска программ пользователями

Управление блокировкой запуска пользователями программ df, chattr, arp, ip осуществляется с помощью инструмента astra-commands-lock. Данные программы необходимо блокировать при обработке в одной системе информации разных уровней конфиденциальности, т.к. с их помощью можно организовать скрытый канал передачи информации между уровнями.

Программы блокируются путем выставления на них прав доступа 750 (rwx r-x ---).

Параметры вызова, используемые данным инструментом, приведены в таблице 65.

Изменение блокировки вступает в действие немедленно.

Описание инструмента приведено в man astra-commands-lock.

Управление блокировкой запуска программ пользователями также может осуществляться с помощью графической утилиты `fly-admin-smc`.

16.5.13. Управление загрузкой ядра hardened

Инструмент `astra-hardened-control` включает/отключает в загрузчике GRUB 2 загрузку ядра hardened по умолчанию, если оно присутствует в системе. Параметры вызова, используемые данным инструментом, приведены в таблице 65.

Для применения изменений требуется перезагрузка.

Описание инструмента приведено в `man astra-hardened-control`.

16.5.14. Запуск контейнеров Docker на пониженном уровне МКЦ

Инструмент `astra-docker-isolation` применяется для перевода службы Docker с высокого уровня целостности на второй уровень целостности, для чего в каталоге `/etc/systemd` размещается override-файл для службы Docker.

Данный инструмент доступен к использованию, если в ОС установлена программа Docker.

Параметры вызова, используемые данным инструментом, приведены в таблице 65.

Изменения вступают в силу после перезапуска службы Docker.

Описание инструмента приведено в `man astra-docker-isolation`.

16.5.15. Управление сетевыми службами

Инструмент `astra-ilev1-control` при его включении переводит сетевые службы `apache2`, `dovecot` и `exim4` с высокого уровня целостности на первый уровень целостности, для чего в каталоге `/etc/systemd` размещаются override-файлы для соответствующих служб.

Параметры вызова, используемые данным инструментом, приведены в таблице 65.

Если указанные службы не были установлены при включении этой функции, то функция будет автоматически применена к данным службам при их установке.

Для изменения уровня целостности работающей службы требуется его перезапуск.

Описание инструмента приведено в `man astra-ilev1-control`.

16.5.16. Управление загрузкой модуля ядра lkrg

Инструмент `astra-lkrg-control` управляет загрузкой модуля ядра lkrg и настраивает его автозагрузку. Модуль ядра lkrg обеспечивает обнаружение некоторых уязвимостей и защиту пространства памяти ядра от модификации.

Примечание. Модуль ядра lkrg работает с использованием механизма kprobes, предназначенного для отладки и трассирования ядра ОС. Этот механизм отключен в ядре hardened, в связи с этим функционирование модуля lkrg не поддерживается в ядре hardened.

Параметры вызова, используемые данным инструментом, приведены в таблице 65.

Если установлен пакет `lkrg`, то при включении данной функции сразу загружается модуль ядра lkrg и настраивается его автозагрузка при загрузке системы (на этапе загрузки `initrd`).

При выключении функции модуль lkrg сразу выгружается из ядра и удаляется из автозагрузки.

Описание инструмента приведено в `man astra-lkrg-control`.

16.5.17. Блокировка неиспользуемых модулей ядра

Инструмент `astra-modban-lock` блокирует загрузку неиспользуемых модулей ядра. Неиспользуемыми считаются те модули, которые не загружены в момент включения `astra-modban-lock`.

Параметры вызова, используемые данным инструментом, приведены в таблице 65.

Изменение блокировки вступает в действие немедленно.

Описание инструмента приведено в `man astra-modban-lock`.

16.5.18. Блокировка автоматического конфигурирования сетевых подключений

Инструмент `astra-noautonet-control` блокирует автоматическое конфигурирование сетевых подключений путем блокировки работы служб `NetworkManager` (`network-manager`) и `connman`, а также выключает отображение элемента управления сетевыми подключениями в области уведомлений панели задач.

Параметры вызова, используемые данным инструментом, приведены в таблице 65.

Изменение блокировки вступает в действие немедленно.

Описание инструмента приведено в `man astra-noautonet-control`.

16.5.19. Отключение отображения меню загрузчика

Инструмент `astra-nobootmenu-control` отключает отображение меню загрузчика GRUB 2 при загрузке системы. Параметры вызова, используемые данным инструментом, приведены в таблице 65.

Для применения изменений требуется перезагрузка.

Описание инструмента приведено в `man astra-nobootmenu-control`.

16.5.20. Ограничение на форматирование съемных носителей

Инструмент `astra-format-lock` устанавливает или отменяет необходимость запроса пароля администратора при форматировании съемных носителей информации.

Параметры вызова, используемые данным инструментом, приведены в таблице 65. По умолчанию пароль администратора запрашивается.

Описание инструмента приведено в `man astra-format-lock`.

16.5.21. Запрет монтирования съемных носителей

Инструмент `astra-mount-lock` запрещает монтирование съемных носителей информации пользователям, входящим в группу `floppy`, при этом запрет не распространяется на пользователей из группы `astra-admin`.

Параметры вызова, используемые данным инструментом, приведены в таблице 65.

Изменение запрета вступает в действие немедленно.

Описание инструмента приведено в `man astra-mount-lock`.

Управление запретом также может осуществляться с помощью графической утилиты `fly-admin-smc`.

16.5.22. Включение на файловой системе режима работы «только чтение»

Инструмент `astra-overlay` включает overlay (режим работы «только чтение») на корневой ФС.

Параметры вызова, используемые данным инструментом, приведены в таблице 65.

После включения функции безопасности фактическое содержимое корневой ФС монтируется в overlay одновременно с ФС, хранящейся в памяти. Изменения файлов сохраняются только в памяти, а ФС на носителе остается без изменений. После перезагрузки все изменения теряются, и система каждый раз загружается в исходном состоянии.

Данная функция может применяться в тех случаях, когда носитель, на котором расположена корневая ФС, аппаратно защищен от записи либо необходимо программно защитить его от изменений.

Функционал overlay не касается файловых систем, хранящихся на отдельных разделах, отличных от корневого. Например, если каталог `/home` хранится на отдельном разделе или носителе, вносимые в него изменения будут сохраняться после перезагрузки.

Для применения изменений требуется перезагрузка.

Описание инструмента приведено в `man astra-overlay`.

Управление режимом также может осуществляться с помощью графической утилиты `fly-admin-smc`.

16.5.23. Блокировка выключения компьютера пользователями

Инструмент `astra-shutdown-lock` блокирует выключение компьютера пользователями. Параметры вызова, используемые данным инструментом, приведены в таблице 65.

При включении данной функции добавляется политика polkit, которая запрещает выключение компьютера без ввода пароля администратора. Также при включении инструмента блокируется возможность перезагрузки компьютера путем нажатия комбинации клавиш **<Ctrl+Alt+Delete>**.

Изменение блокировки вступает в действие после перезагрузки.

Описание инструмента приведено в `man astra-shutdown-lock`.

Управление блокировкой выключения компьютера пользователями также может осуществляться с помощью графической утилиты `fly-admin-smc`.

16.5.24. Управление вводом пароля для sudo

Инструмент `astra-sudo-control` включает и выключает требование ввода пароля при использовании механизма `sudo`. Параметры вызова, используемые данным инструментом, приведены в таблице 65.

Изменение вступает в действие немедленно.

Описание инструмента приведено в `man astra-sudo-control`.

Управление вводом пароля для `sudo` также может осуществляться с помощью графической утилиты `fly-admin-smc`.

16.5.25. Блокировка использования утилиты sumac

Инструмент `astra-sumac-lock` блокирует работу утилит `sumac` и `fly-sumac`. Параметры вызова, используемые данным инструментом, приведены в таблице 65.

Если данная блокировка включена, то все пользователи, даже у которых есть привилегия `PARSEC_CAP_SUMAC`, не смогут использовать команду `sumac`. Для этого устанавливаются права доступа `000` на исполняемый файл `sumac` и библиотеку `libsumacrunner.so`.

Изменение блокировки вступает в действие немедленно.

Описание инструмента приведено в `man astra-sumac-lock`.

Управление блокировкой также может осуществляться с помощью графической утилиты `fly-admin-smc`.

16.5.26. Управление межсетевым экраном ufw

Инструмент `astra-ufw-control` включает и выключает межсетевой экран ufw. Параметры вызова, используемые данным инструментом, приведены в таблице 65.

Если при включении межсетевого экрана ufw включен другой межсетевой экран (firewalld), то ufw не будет включен.

Изменение вступает в действие немедленно.

Описание инструмента приведено в `man astra-ufw-control`.

Управление межсетевым экраном ufw также может осуществляться с помощью графической утилиты `fly-admin-smc`.

16.5.27. Переключение уровней защищенности

Инструмент `astra-modeswitch` позволяет переключать уровни защищенности ОС и отображает текущий уровень. Уровень защищенности определяет перечень доступных для использования функций безопасности. В ОС реализованы следующие уровни защищенности:

- базовый;
- усиленный;
- максимальный.

Переключение уровня защищенности делает доступными/недоступными для включения функции безопасности соответствующего уровня защищенности.

Функции безопасности базового уровня защищенности доступны на всех уровнях защищенности.

Функции безопасности, доступные на усиленном и максимальном уровнях защищенности, приведены в таблице 66.

Таблица 66

Функция безопасности	Уровень защищенности		
	базовый	усиленный	максимальный
Замкнутая программная среда	Недоступна	Доступна	Доступна
Очистка освобождаемой внешней памяти	Недоступна	Доступна	Доступна
Мандатный контроль целостности	Недоступна	Доступна	Доступна
Мандатное управление доступом	Недоступна	Недоступна	Доступна

ВНИМАНИЕ! Функции безопасности, недоступные на текущем уровне защищенности, не могут быть включены.

ВНИМАНИЕ! После переключения уровня защищенности:

- если функция безопасности недоступна на данном уровне защищенности, то она автоматически отключается;
- если функция безопасности доступна на данном уровне защищенности, то ее состояние не меняется, но она может быть включена с использованием соответствующих инструментов.

Для включения функций безопасности используются следующие инструменты:

1) «Замкнутая программная среда» — управление осуществляется с помощью инструмента `astra-digsig-control`, описанного в 16.5.29;

2) «Очистка освобождаемой внешней памяти» — включает в себя две функции безопасности: безопасное удаление файлов и очистка разделов подкачки. Управление осуществляется с помощью инструментов `astra-secdel-control` и `astra-swapwiper-control`, описанных в 16.5.30 и 16.5.31 соответственно;

3) «Мандатный контроль целостности» — управление осуществляется с помощью инструмента `astra-mic-control`, описанного в 16.5.28;

4) «Мандатное управление доступом» — управление осуществляется с помощью инструмента `astra-mac-control`, описанного в 16.5.32.

Описание функций безопасности, доступных для каждого из уровней защищенности, приведено также в РУСБ.10015-37 95 01-1.

С инструментом `astra-modeswitch` возможно использовать только параметры, приведенные в таблице 67.

Таблица 67

Параметр	Описание
`list`	Вывести список доступных уровней защищенности. В выводе команды: 0 `base(orel)` — базовый уровень защищенности; 1 `advanced(voronezh)` — усиленный уровень защищенности; 2 `maximum(smolensk)` — максимальный уровень защищенности.
`get`	Вывести текущий уровень защищенности в числовом представлении
`getname`	Вывести текущий уровень защищенности в текстовом представлении
`set m`	Установить уровень защищенности, заданный параметром `m`. В качестве параметра `m` используется числовое или текстовое представление уровня защищенности

Смена уровня защищенности осуществляется после перезагрузки ОС.

16.5.28. Управление мандатным контролем целостности

Инструмент `astra-mic-control` включает и выключает МКЦ, описанный в 4.9, а также изменяет значение максимального уровня целостности системы. Управление МКЦ осуществляется путем изменения значения параметра командной строки ядра `parsec.max_ilev`.

Параметры вызова, используемые данным инструментом, приведены в таблице 65. При использовании вызова `enable` возможно указать необязательный параметр `-i <уровень>`, задающий значение максимального уровня целостности (после установки ОС максимальный уровень целостности равен 63).

После выключения МКЦ и перезагрузки целостность на объектах файловой системы сбрасывается на нулевые значения.

После включения МКЦ и перезагрузки на объектах файловой системы восстанавливаются принятые по умолчанию значения целостности (высокий уровень целостности на каталогах `/dev`, `/proc`, `/run`, `/sys`).

Для применения изменений требуется перезагрузка.

Описание инструмента приведено в `man astra-mic-control`.

16.5.29. Управление замкнутой программной средой

Инструмент `astra-digsig-control` позволяет включать и выключать ЗПС в исполняемых файлах из командной строки. Описание ЗПС приведено в 16.1.

Параметры вызова, используемые данным инструментом, приведены в таблице 65.

Изменение состояния ЗПС выполняется после перезагрузки.

Описание инструмента приведено в `man astra-digsig-control`.

16.5.30. Управление безопасным удалением файлов

Инструмент `astra-secdel-control` включает и выключает механизм безопасного удаления файлов в соответствии с 8.1 на разделах с файловыми системами ext2, ext3, ext4, XFS, указанных в `/etc/fstab`.

Параметры вызова, используемые данным инструментом, приведены в таблице 65.

При включении механизма безопасного удаления файлов по умолчанию используется режим с параметром `secdel` в соответствии с 8.1.

Изменение режима работы вступает в действие после следующего монтирования ФС (в т.ч. после перезагрузки ОС).

Описание инструмента приведено в `man astra-secdel-control`.

Управление режимом также может осуществляться с помощью графической утилиты `fly-admin-smc`.

16.5.31. Управление очисткой разделов подкачки

Инструмент `astra-swapwiper-control` контролирует очистку разделов подкачки при завершении работы ОС в соответствии с 8.1.

Параметры вызова, используемые данным инструментом, приведены в таблице 65.

При использовании команды-переключателя вносятся изменения в конфигурационный файл `/etc/parsec/swap_wiper.conf`.

Изменение вступает в действие немедленно.

Описание инструмента приведено в `man astra-swapwiper-control`.

Управление блокировкой также может осуществляться с помощью графической утилиты `fly-admin-smc`.

16.5.32. Включение и выключение мандатного управления доступом

Инструмент `astra-mac-control` включает и выключает мандатное управление доступом в соответствии с 4.8, изменяя значение параметра командной строки ядра `parsec.mac`.

Для применения изменений требуется перезагрузка ОС.

При выключении мандатного управления доступом инструмент отключает возможность работы программ на ненулевом уровне конфиденциальности.

Параметры вызова, используемые данным инструментом, приведены в таблице 65.

ВНИМАНИЕ! Отключение возможности работы программ на ненулевом уровне конфиденциальности ограничивает доступ к файловым объектам, имеющим ненулевую классификационную метку. Доступ к таким объектам может быть получен администратором

(если в ОС включен МКЦ, то администратором на высоком уровне целостности). Доступ к таким объектам также может быть получен при наличии неконтролируемого физического доступа к компьютеру и возможности использовать на нем нештатные средства. Необходимо принять дополнительные меры по защите информации ограниченного доступа.

Описание инструмента приведено в `man astra-mac-control`.

16.5.33. Управление AstraMode и MacEnable

Инструмент `astra-mode-apps` включает и выключает режим AstraMode сервера Apache2, а также управляет состоянием параметра `MacEnable` сервера печати CUPS.

Описание режима AstraMode приведено в РУСБ.10015-37 95 01-1, описание параметра `MacEnable` приведено в таблице 59.

Параметры вызова, используемые инструментом `astra-mode-apps`, приведены в таблице 65.

При использовании команды-переключателя вносятся изменения в конфигурационные файлы `/etc/apache2/apache2.conf` и `/etc/cups/cupsd.conf`, изменяя значения параметров `AstraMode` и `MacEnable` соответственно.

Для применения изменений требуется перезапуск служб.

Описание инструмента приведено в `man astra-mode-apps`.

Включение и выключение режима AstraMode и управление параметром `MacEnable` также можно осуществлять с помощью графической утилиты `fly-admin-smc` (см. электронную справку).

16.6. Модуль безопасности yama

Модуль безопасности yama ограничивает возможности использования механизма `ptrace` для отладки (трассировки) процессов ОС.

Для активации отладки процессов используется системный вызов `ptrace`:

- для назначения родительского процесса отладчиком текущего процесса:

```
ptrace(PTRACE_TRACEME, 0, NULL, NULL)
```

- для назначения текущего процесса отладчиком процесса с идентификатором `pid`:

```
ptrace(PTRACE_ATTACH, pid, NULL, NULL)
```

Модуль yama поддерживает четыре режима работы:

1) `0` — отладка всех процессов, запущенных от имени текущего пользователя;

2) `1` — отладка только дочерних процессов, за исключением случаев явного указания текущим процессом возможного процесса-отладчика через вызов `prctl` с параметром `PR_SET_PTRACER`, задав `pid` процесса-отладчика (когда любые процессы, связанные с конкретной службой, должны отлаживаться специально выделенным

для этого процессом) или параметр PR_SET_PTRACER_ANY (для разрешения любому процессу отлаживать текущий процесс), например:

```
prctl(PR_SET_PTRACER, PR_SET_PTRACER_ANY, 0, 0, 0)
```

В случаи использования PTRACE_TRACEME дополнительных ограничений не накладывается, для возможности выполнения PTRACE_ATTACH необходимо наличие у процесса-отладчика привилегии CAP_SYS_PTRACE;

3) 2 — возможна отладка любых процессов с использованием PTRACE_TRACEME или PTRACE_ATTACH, но только при наличии у процесса-отладчика привилегии CAP_SYS_PTRACE;

4) 3 — отладка процессов с использованием ptrace запрещена.

Узнать режим, в котором работает модуль yama в текущий момент, можно с помощью команды:

```
sysctl kernel.yama.ptrace_scope
```

либо прочитав файл /proc/sys/kernel/yama/ptrace_scope:

```
cat /proc/sys/kernel/yama/ptrace_scope
```

Для изменения режима работы модуля yama можно использовать инструмент sysctl, указав для параметра kernel.yama.ptrace_scope значение требуемого режима работы (от 0 до 3):

```
sysctl kernel.yama.ptrace_scope=2
```

либо откорректировав соответствующим образом файл /proc/sys/kernel/yama/ptrace_scope:

```
echo 2 > /proc/sys/kernel/yama/ptrace_scope
```

ВНИМАНИЕ! Заданный режим работы модуля yama сохраняется только до перезагрузки ОС (выключения компьютера). При запуске ОС модуль yama всегда будет загружаться в режиме работы 1.

Для изменения режима работы модуля yama пользователь должен обладать привилегией CAP_SYS_PTRACE.

После перевода модуля yama в режим работы 3 (полный запрет трассировки процессов), дальнейшее изменение режимов работы может быть выполнено только после перезагрузки ОС.

16.7. Модуль безопасности lockdown

Модуль безопасности lockdown устанавливает для привилегированного пользователя запрет на процедуры получения доступа и выполнения кода на уровне ядра ОС, которые в штатном режиме он может осуществить, например, с помощью подмены ядра ОС (используя системный вызов kexec) или через /dev/kmem, читая/записывая данные в память.

Модуль lockdown поддерживает два режима работы:

1) режим обеспечения целостности ядра ОС (режим integrity), включающий следующие ограничения:

а) LOCKDOWN_KEXEC — запрет выполнения системного вызова kexec(), обеспечивающего подмену текущего ядра ОС;

б) LOCKDOWN_HIBERNATION — запрет гибернации, предотвращающий возможность подмены ядра ОС при выходе из нее;

в) LOCKDOWN_MODULE_SIGNATURE — включение проверки подписей модулей ядра ОС;

г) LOCKDOWN_DEV_MEM — запрет чтения и записи в /dev/mem, /dev/kmem, /dev/port;

д) LOCKDOWN_PCI_ACCESS — блокировка оборудования, которое потенциально может генерировать прямую адресацию памяти (DMA);

е) LOCKDOWN_IOPORT — блокировка ioctl-вызывов (ioctl_console) KDADDIO, KDDELIO, KDENABIO и KDDISABIO для терминалов и виртуальных консолей;

ж) LOCKDOWN_EFI_TEST — запрет чтения /dev/efi_test;

з) LOCKDOWN_ACPI_TABLES — ограничение доступа к интерфейсам ACPI;

2) режим обеспечение конфиденциальности некоторых компонентов ядра ОС (режим confidentiality), включающий все ограничения режима integrity, а также:

а) LOCKDOWN_KCORE — запрет чтения /proc/kcore;

б) LOCKDOWN_KPROBES — ограничение доступа к отладочному режиму kprobes;

в) LOCKDOWN_BPF_READ — ограничение доступа к механизму фильтрации пакетов BPF;

г) LOCKDOWN_TRACEFS — ограничение доступа к ФС tracefs.

Состояние модуля lockdown задается в файле /sys/kernel/security/lockdown, который по умолчанию имеет следующий вид:

```
[none] integrity confidentiality
```

В данном файле приведены возможные состояния модуля lockdown, а его текущее состояние отображается в квадратных скобках ([none] — модуль отключен).

После загрузки ОС модуль lockdown по умолчанию отключен.

Активировать модуль lockdown в нужном режиме работы можно в процессе функционирования ОС путем записи соответствующего режима в файл /sys/kernel/security/lockdown командой:

```
sudo echo <режим_модуля> > /sys/kernel/security/lockdown
```

где <режим_модуля> — integrity (для активации режима integrity) или confidentiality (для активации режима confidentiality), при этом:

- если модуль был отключен, то возможно активировать любой режим;
- если модуль работал в режиме `integrity`, то возможно только активировать режим `confidentiality`;
- если модуль работал в режиме `confidentiality`, то изменить режим или отключить модуль невозможно.

Изменения в режиме работы модуля lockdown сохраняются до перезагрузки ОС.

17. УСЛОВИЯ ЭКСПЛУАТАЦИИ ОС

В целях защиты информации от несанкционированного доступа, включая обеспечение конфиденциальности, целостности, доступности информации и устойчивого функционирования информационных систем, необходимо обеспечить условия эксплуатации ОС согласно 17.1-17.4.

17.1. Обеспечение безопасности среды функционирования

В целях обеспечения безопасности среды функционирования ОС на объектах информатизации должны быть выполнены следующие требования:

1) СВТ, предназначенное для установки и функционирования ОС, должно соответствовать требованиям, указанным в разделе 2 документа РУСБ.10015-37 31 01;

2) для создания защищенной среды виртуализации изделие должно применяться совместно с аппаратным обеспечением, поддерживающим соответствующие технологии VT/SVM, в том числе при необходимости предоставления доступа виртуальным машинам к физическим устройствам;

3) установка, управление и конфигурирование ОС должны проводиться в соответствии с эксплуатационной документацией;

4) должна быть обеспечена защита от осуществления действий, направленных на нарушение физической целостности СВТ, на котором функционирует ОС;

5) должна быть обеспечена доверенная загрузка ОС. Доверенную загрузку ОС рекомендуется выполнять с помощью сертифицированных средств доверенной загрузки или аппаратно-программных модулей доверенной загрузки. При технической невозможности или нецелесообразности использования таких средств должны быть приняты организационно-технические меры, предотвращающие возможность доступа пользователя к ресурсам СВТ в обход механизмов защиты ОС (должна быть исключена возможность загрузки альтернативной операционной системы на СВТ, а также модификация модулей загружаемой ОС);

6) при использовании сетевого взаимодействия необходимо обеспечить доверенный канал передачи информации между СВТ, на которых установлена ОС (например, контроль несанкционированного подключения к ЛВС в пределах контролируемой зоны, защищенная передача сетевого трафика за пределами контролируемой зоны);

7) должны быть определены лица, отвечающие за эксплуатацию и безопасность ОС, которым будут предоставлены административные полномочия в ОС. Данные лица не должны считаться нарушителем в модели угроз безопасности информации;

8) лица, ответственные за эксплуатацию ОС, должны обеспечить, чтобы аутентификационная информация для каждой учетной записи пользователя ОС содержалась в

тайне и была недоступна лицам, не уполномоченным использовать данную учетную запись.

17.2. Указания по эксплуатации ОС

17.2.1. Перед началом эксплуатации ОС администратор безопасности должен обеспечить следующие условия:

1) 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 указанном файле в секции `Primary block` первой строкой добавить:

```
auth required pam_securetty.so
```

2) используя графическую утилиту `fly-admin-smc` («Управление политикой безопасности», см. электронную справку), запущенную от имени администратора через механизм `sudo`, в категории «Политики учетной записи» задать параметры для настройки политики паролей.

Задать данные настройки можно также путем корректировки файлов сценариев:

а) в файле `/etc/pam.d/common-password` в строке `password requisite pam cracklib.so` установить значение `minlen=<минимальная_длина_пароля>` и добавить параметры `dcredit=-1`, `ucredit=-1` и `lcredit=-1` (данные параметры устанавливают требования к сложности пароля);

б) в файле `/etc/login.defs` для переменной `PASS_MAX_DAYS` установить требуемое значение максимального срока действия пароля в днях.

Запрет повторного использования заданного числа последних паролей можно установить путем корректировки файла `/etc/pam.d/common-password`. Для этого в указанном файле в строке `...pam_unix.so` добавить параметр `remember=<число_паролей>`;

3) используя графическую утилиту `fly-admin-smc` («Управление политикой безопасности», см. электронную справку), запущенную от имени администратора через механизм `sudo`, в категории «Политики учетной записи» задать значения для настройки блокировки при неуспешных входах пользователя в систему. Данные параметры также могут быть установлены путем корректировки файла `/etc/pam.d/common-auth` (параметры `deny`, `lock_time` и `unlock_time`);

4) функции разграничения доступа, мандатного контроля целостности и гарантированного уничтожения (стирания) информации в полном объеме реализованы только

для файловых систем ext2/ext3/ext4/XFS, поддерживающих расширенные (в т.ч. мандатные) атрибуты файловых объектов. При использовании других файловых систем необходимо учитывать данные ограничения. Установку ОС следует выполнять на системный раздел с файловой системой ext4/XFS;

5) при необходимости установки в систему внешних модулей уровня ядра ОС, такие модули должны быть получены из доверенного источника, и перед их установкой с ОС должна осуществляться проверка их целостности;

6) использование файловой системы NFS для доступа к сетевым дискам возможно при выполнении следующих условий:

а) сетевые диски не предназначены для хранения файловых объектов, требующих обеспечения мандатного управления доступом к ним;

б) версия используемого протокола NFS не ниже 3;

7) задать значение времени неактивности для блокировки экрана, отредактировав (или создав, если отсутствует) файл `usr/share/fly-wm/theme.master/themerc`, указав в нем строки:

```
[Variables]
ScreenSaverDelay=<время_неактивности_в_секундах>
LockerOnSleep=true
LockerOnDPMS=true
LockerOnLid=true
LockerOnSwitch=true
```

17.2.2. При использовании механизма гарантированного удаления данных должны дополнительно быть выполнены следующие условия:

1) при использовании разделов подкачки в ОС необходимо активировать в файле `/etc/parsec/swap_wiper.conf` их очистку согласно 8.1.

17.2.3. При использовании мандатного контроля целостности должны дополнительно быть выполнены следующие условия:

1) при использовании инструмента `qemu-ga` (из состава пакета `qemu-guest-agent`) значения параметров `-p` (в том числе при использовании опции `-m unix-listen`) и `-t` должны указывать на сокеты и файлы, уровни целостности которых совпадают с уровнем целостности запускаемого процесса.

17.2.4. При использовании мандатного управления доступом должны дополнительно быть выполнены следующие условия:

1) с использованием средств управления запуском служб должна быть отключена служба `gpm` для поддержки мыши в консольном режиме;

2) в случае необходимости использования мандатного управления доступом на сетевых дисках, их монтирование в файловую систему ОС должно осуществляться только с использованием файловой системы CIFS, поддерживающей расширенные (в т.ч. мандатные) атрибуты пользователей;

3) в конфигурационном файле защищенного сервера печати из состава изделия `/etc/cups/cupsd.conf` не допускается установка значения `None` параметра `DefaultAuthType` (отключение аутентификации) и внесение изменений в параметры политики PARSEC, не соответствующих эксплуатационной документации;

4) при использовании защищенного сервера СУБД из состава ОС в режиме мандатного управления доступом необходимо в конфигурационном файле кластера `postgresql.conf` для параметра `enable_bitmapscan` установить значение `off` и для параметра `ac_ignore_socket_maclabel` установить значение `false`;

5) при использовании защищенного сервера СУБД из состава ОС в режиме мандатного управления доступом не допускается отключать аутентификацию субъектов доступа установкой в конфигурационном файле кластера `pg_hba.conf` режима `trust` (без аутентификации);

6) при использовании защищенного комплекса программ электронной почты из состава ОС в режиме мандатного управления доступом конфигурационные параметры агента передачи электронной почты Exim и агента доставки электронной почты Dovecot не должны допускать отправку и прием сообщений электронной почты без аутентификации;

7) для штатной работы ОС не допускается отключение администратором системы расширения `XPARSEC` у X-сервера (использование отладочной опции `XPARSEC=Disable` в конфигурационных файлах X-сервера, см. документ РУСБ.10015-37 31 01).

17.2.5. В целях обеспечения удаленного доступа пользователей с использованием сетей связи общего пользования к средствам виртуализации из состава ОС должны применяться средства криптографической защиты информации, прошедшие процедуру оценки соответствия в соответствии с законодательством Российской Федерации.

17.2.6. Запрещается использование образа контейнера при выявлении в нем уязвимостей.

17.3. Условия применения ПО

17.3.1. ПО, функционирующее в среде ОС, не должно изменять алгоритм принятия решения о предоставлении доступа субъектов (пользователей) к объектам ОС и содержать скрытых или явных возможностей, позволяющих:

1) подменять файлы образа ядра `vmlinuz-*` и образов временной файловой системы начальной загрузки `/boot/initrd.img-*` (в случае отсутствия явной необходимости обновления отдельных компонентов, входящих в состав образа), находящиеся в каталоге `/boot`, файлы модулей ядра, находящиеся в каталоге `/lib/modules`, и прочие файлы, входящие в состав изделия или стороннего ПО;

2) статически или динамически изменять таблицу системных вызовов и поля структуры типа `security_operations` и иных структур типа `*security*`;

3) переопределять основной процесс ОС в конфигурационном файле загрузчика изделия путем установки параметра `init=<полный_путь_к_исполняемому_файлу>`;

4) изменять параметры аутентификации в конфигурационных файлах PAM-сценариев, находящихся в каталоге `/etc/pam.d`, результатом чего может являться снижение установленного уровня доверия к результатам идентификации и аутентификации;

5) устанавливать подгружаемые модули аутентификации (PAM-модули), определяющие мандатные атрибуты сессии пользователя с использованием функций API библиотеки `libparsec-mac` подсистемы безопасности PARSEC, имена которых начинаются с префиксов `mac_set_`, `parsec_` или `pdp_`;

6) устанавливать PAM-модули, определяющие привилегии PARSEC сессии пользователя с использованием функций API библиотеки `libparsec-cap` подсистемы безопасности PARSEC, имена которых начинаются с префиксов `mcap` или `parsec_`;

7) устанавливать интерпретаторы команд, заменяющие интерпретаторы, входящие в состав изделия (bash, dash, PHP, Perl, Python, TCL, Ruby);

8) получать доступ к памяти других процессов ПО с использованием привилегии `CAP_SYS_PTRACE` и системного вызова `ptrace`;

9) изменять системное время (если наличие возможностей по изменению времени не предусмотрено функциональным назначением ПО);

10) динамически изменять сегмент кода ядра ОС и использовать неэкспортируемые символы ядра ОС;

11) осуществлять доступ к файлу `ctl` файловой системы `parsecfs` посредством системного вызова `ioctl`, минуя системные функции API-библиотек `libparsec-*`.

17.3.2. Для ПО, функционирующего в среде ОС, должны соблюдаться следующие условия:

1) необходимость и корректность использования в ПО привилегий администратора, прикладного программного интерфейса подсистемы безопасности PARSEC, мандат-

ных привилегий, функционирования в пространстве ядра или только на высоком уровне целостности должны быть обоснованы в документации на данное ПО;

2) для ПО, использующего модуль `wsgi_mod` для Apache из состава изделия, в документации разработчика ПО должен быть определен перечень сценариев работы модуля `wsgi_mod`. При этом должны быть исключены сценарии работы `wsgi_mod` в режимах, предусматривающих работу вне контекста пользователя и изменяющих процесс аутентификации.

17.3.3. При использовании защищенного сервера печати из состава изделия установка и использование альтернативных серверов печати должны быть исключены.

17.3.4. При использовании механизма мандатного управления доступом дополнительно должны соблюдаться следующие условия:

1) ПО при обработке запросов пользователей не должно взаимодействовать с защищенной СУБД из состава изделия от имени пользователя с привилегиями `PARSEC_CAP_SETMAC` и `PARSEC_CAP_CHMAC`, позволяющими изменять мандатные атрибуты защищаемых объектов в СУБД;

2) ПО, содержащее сетевые службы (программы, обрабатывающие запросы пользователей с применением протоколов стека TCP/IP), которые используют привилегии (`PARSEC_CAP_SETMAC`, `PARSEC_CAP_CHMAC` и `PARSEC_CAP_PRIV_SOCK`) и прикладной программный интерфейс подсистемы безопасности PARSEC из состава изделия, должно пройти сертификационные исследования, подтверждающие корректность реализации указанных служб;

3) программный компонент RabbitMQ не применять при одновременной обработке данных с различными уровнями конфиденциальности либо для обеспечения взаимодействия процессов с различными уровнями доступа;

4) ПО, обрабатывающее одновременно данные с различными уровнями конфиденциальности, не должно использовать программные пакеты Erlang.

17.4. Условия исключения скрытых каналов

Вопросы исключения скрытых каналов должны рассматриваться при построении информационных систем, обрабатывающих информацию, отнесенную к государственным секретам, и могут рассматриваться при построении информационных систем, обрабатывающих информацию, распространение и (или) предоставление которой ограничено, не отнесенной к государственным секретам.

Для исключения скрытых каналов (информационных потоков) при защите от угрозы целостности информации используется мандатный контроль целостности (см. 4.9), который в условиях установленных в компьютерной системе уровней целостности (уровней доверия)

обеспечивает невозможность записи отправителем с низким уровнем доверия информации в объекты с более высоким уровнем доверия.

Для исключения возможности использования отправителем и получателем общего буфера обмена в ОС обеспечивается изоляция процессов (см. 7.1) в совокупности с очисткой областей оперативной памяти при освобождении или выделении (см. 8.1), а также обеспечение запуска процессов в замкнутой относительно остальных процессов среде по памяти, в частности, использование Киоск-2 (см. 16.2) и Xephyr в графической подсистеме, обеспечивающих изоляцию буфера обмена.

Для исключения возможности запуска программ, не участвующих в процессе обработки информации, должен быть активирован механизм замкнутой программной среды и настроен для работы в штатном режиме.

Таким образом, условиями исключения скрытых каналов является реализуемая ОС политика дискреционного управления доступом (см. раздел 3), мандатного управления доступом и мандатного контроля целостности (см. раздел 4), которая является неотъемлемой частью модели политики безопасности ОС. С учетом указанных условий эксплуатации скрытые каналы по памяти не могут быть реализованы, а по времени крайне затруднены, т.е. имеют низкие пропускную способность и вероятность возникновения условий для реализации.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

БД — база данных
ЕПП — единое пространство пользователей
КСЗ — комплекс средств защиты
МКЦ — мандатный контроль целостности
НСД — несанкционированный доступ
ОС — операционная система специального назначения «Astra Linux Special Edition»
ПО — программное обеспечение
ПРД — правила разграничения доступа
СВТ — средство вычислительной техники
СЗИ — средства защиты информации
СЗФС — сетевая защищенная файловая система
СПО — специальное программное обеспечение
СУБД — система управления базами данных
ФС — файловая система
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)
DNS — Domain Name System (система доменных имен)
FTP — File Transfer Protocol (протокол передачи файлов)
GID — Group Identifier (идентификатор группы)
IP — Internet Protocol (межсетевой протокол)
IPC — InterProcess Communication (межпроцессное взаимодействие)
LDAP — Lightweight Directory Access Protocol (легковесный протокол доступа к сервисам каталогов)
MAC — Mandatory Access Control (мандатное управление доступом)
NFS — Network File System (сетевая файловая система)
NIS — Network Information Service (сетевая информационная служба)

NSS — Name Service Switch (служба для организации унифицированного доступа к информации о пользователях, группах, сетевых именах, службах и т.п. на основе конфигурации различных источников, таких как: локальные файлы, различные средства сетевой идентификации (DNS, NIS), а также LDAP)

PAM — Pluggable Authentication Modules (подключаемые модули аутентификации)

PID — Process Identifier (идентификатор процесса)

RSA — Rivest Shamir Adelman (алгоритм по схеме открытого ключа)

SMB — Server Message Block (блок сообщений сервера)

SQL — Structured Query Language (язык структурированных запросов)

TCP — Transmission Control Protocol (протокол передачи данных)

UDP — User Datagram Protocol (протокол управления передачей данных)

UID — User Identifier (идентификатор пользователя)

Лист регистрации изменений

Изм.	Номера листов (страниц)				Всего листов (страниц) в документе	Номер документа	Входящий номер сопроводительного документа и дата	Подпись	Дата
	измененных	замененных	новых	аннулированных					
1		Все				РУСБ.06-22 от 12.09.22			