РУКОВОДСТВО АДМИНИСТРАТОРА

Версия 1.0

Листов 145

2023

АННОТАЦИЯ

Настоящий документ является руководством администратора Операционной системы (ОС) Аврора релиз 4.0.2 update 4.

Настоящий документ содержит описание доступных администратору функциональных возможностей мобильного устройства (МУ), функционирующего под управлением ОС Аврора[1].

Администратор имеет доступ к функциям и настройкам ОС Аврора, описанным как в настоящем документе, так и в документе «Руководство пользователя».

ПРИМЕЧАНИЯ:

1. Перед началом работы администратору необходимо ознакомиться с положениями настоящего документа, а также с информацией, приведенной в документе «Руководство пользователя»;

2. Внешний вид интерфейса МУ может отличаться от приведенного на рисунках в настоящем документе. Снимки экрана являются примером и представлены в документе для общего ознакомления с интерфейсом МУ.

[1] Описание различных способов установки ОС Аврора на МУ приведено в соответствующих документах предприятия-разработчика, которые предназначены для использования Производителями МУ и авторизованными Сервисными центрами производителя.

СОДЕРЖАНИЕ

1. ВВОД В ЭКСПЛУАТАЦИЮ

ОС Аврора представляет собой защищенную мобильную многозадачную ОС для мобильных применений под аппаратные платформы на базе процессоров с архитектурой ARM и имеет различные версии исполнения, описание которых приведено в таблице (Таблица 7).

ВНИМАНИЕ! ОС Аврора в сертифицированной версии может быть использована, но не ограничиваться, в следующих системах и объектах:

– в государственных информационных системах, не содержащих информации, составляющей государственной тайны, до 1 класса защищенности включительно в соответствии с документом «Требования о защите информации, не составляющей государственную тайну, содержащейся в государственных информационных системах», утвержденным приказом ФСТЭК России от 11 февраля 2013 г. № 17;

– в информационных системах персональных данных до 1 уровня защищенности включительно в соответствии с документом «Состав и содержание организационных и технических мер по обеспечению безопасности персональных данных при их обработке в информационных системах персональных данных», утвержденным приказом ФСТЭК России от 18 февраля 2013 г. № 21;

– в автоматизированных системах управления до 1 класса защищенности включительно в соответствии с документом «Требования к обеспечению защиты информации в автоматизированных системах управления производственными и технологическими процессами на критически важных объектах, потенциально опасных объектах, а также объектах, представляющих повышенную опасность для жизни и здоровья людей и для окружающей природной среды», утвержденным приказом ФСТЭК России от 14 августа 2014 г. № 31;

– на значимых объектах критической информационной инфраструктуры до 1 категории включительно в соответствии с документом «Требования по обеспечению безопасности значимых объектов критической информационной инфраструктуры Российской Федерации», утвержденным приказом ФСТЭК России от 25 декабря 2017 г. № 239.

1.1. Ограничения по эксплуатации

Администратору необходимо соблюдать следующие правила и ограничения по эксплуатации МУ:

– не допускать установку любого программного обеспечения (ПО), поставляемого в отличном от RPM виде (самостоятельное копирование файлов, установка ПО из архивов, установка ПО не в штатные каталоги из RPM-пакетов и т.п.);

– исключить подключение МУ к недоверенным точкам доступа беспроводных интерфейсов (WLAN, Bluetooth®) и беспроводным МУ;

ПРИМЕЧАНИЕ. Перечень доверенных точек доступа должен быть сформирован на месте эксплуатации оператором информационной системы (ИС).

– исключить передачу конфиденциальной речевой и иной информации (SMS, MMS) посредством МУ по протоколу GSM;

– предусмотреть меры, обеспечивающие отсутствие компьютерных вирусов на средствах вычислительной техники, к которым подключается МУ.

1.2. Учетные записи ролей

В ОС Аврора реализован многопользовательский режим работы, который позволяет использовать МУ нескольким учетным записям с различными ролями.

Роль – совокупность прав доступа, на основе которых определяется возможность выполнения того или иного действия в ОС Аврора.

Учетная запись роли — это хранящаяся на МУ совокупность данных о пользователе, необходимая для его аутентификации и предоставления доступа к его личным данным и настройкам.

ПРИМЕЧАНИЕ. В зависимости от выбранной роли возможности МУ, функционирующего под управлением ОС Аврора, могут отличаться.

На МУ, функционирующем под управлением ОС Аврора, могут быть созданы одновременно до 7 учетных записей:

– учетная запись с ролью администратора, которая обладает расширенными функциональными возможностями, при этом:

- не может быть удалена с МУ;
- не обладает правами суперпользователя;
- любые изменения настроек, выполненные под такой ролью, будут применимы ко всем учетным записям МУ;

– до 6 учетных записей с ролью пользователя, создание которых выполняется администратором в системных настройках МУ (п. 1.2.1).

ПРИМЕЧАНИЕ. По умолчанию при первом включении МУ загружается в режиме администратора.

1.2.1. Создание учетной записи пользователя

Рисунок 1

Для создания новой учетной записи пользователя администратору необходимо выполнить следующие действия:

– открыть меню настроек касанием значка на Экране приложений (Рисунок 1);

– в меню системных настроек коснуться пункта «Пользователи» ;

– на странице «Пользователи» коснуться пункта «Добавить пользователя» (Рисунок 2);

– на отобразившейся странице ввести имя и логин новой учетной записи пользователя (Рисунок 3);

– установить квоту на использование дискового пространства, перемещая соответствующий слайдер вправо для увеличения квоты либо влево для уменьшения (Рисунок 3);

– коснуться кнопки «Подтвердить» для сохранения изменений либо кнопки «Отменить» для отмены операции;

– в случае необходимости сохранения изменений подтвердить действие вводом текущего кода безопасности, в результате чего на странице «Пользователи» отобразится строка с созданным пользователем (Рисунок 4).

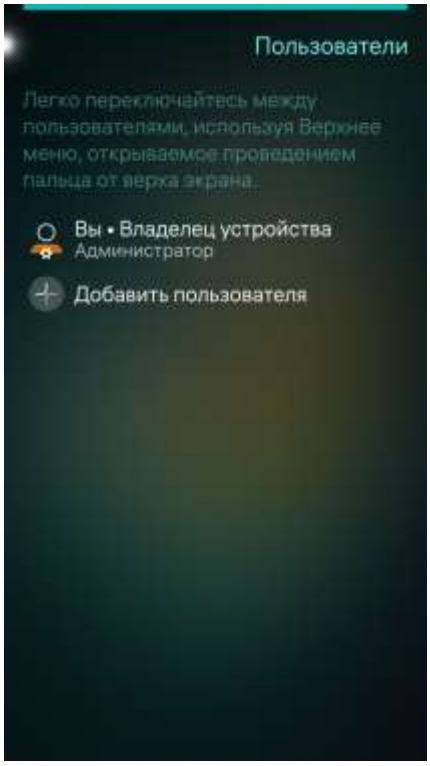

Рисунок 2

Рисунок 3

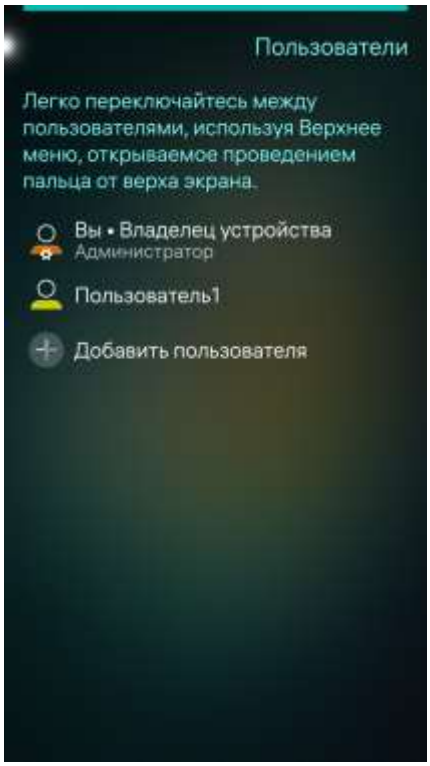

Рисунок 4

ПРИМЕЧАНИЯ:

1. Минимальные и максимальные значения для задания квоты зависят от конструктивных особенностей МУ;

2. Подробная информация о квотировании постоянной памяти приведена в п. 4.7.2.

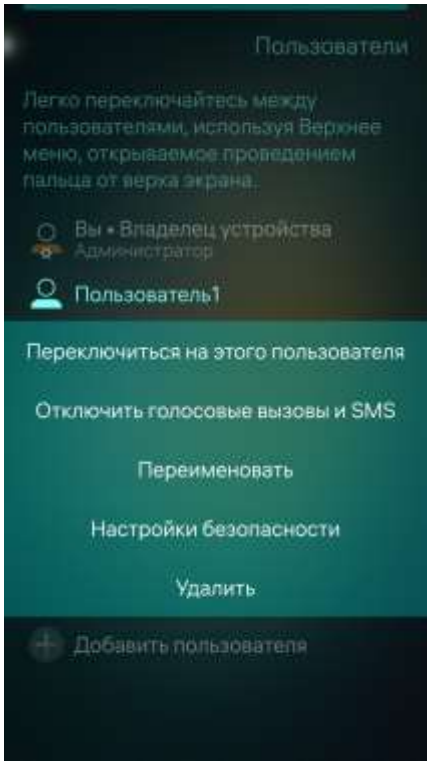

Рисунок 5

При работе с учетными записями пользователей администратору доступны следующие действия (Рисунок 5):

– переключение между учетными записями ролей (п. 1.2.2);

– управление голосовыми вызовами и SMS (п. 1.2.3);

– переименование учетной записи пользователя (п. 1.2.4);

– настройки безопасности (подраздел 1.4);

– удаление учетной записи пользователя (п. 1.2.5).

1.2.2. Переключение между учетными записями

Просмотреть под какой учетной записью загружено МУ, а также выполнить переключение между учетными записями можно как в верхнем меню (Рисунок 6 – Рисунок 7), так и в системных настройках (Рисунок 8).

Для проверки активной учетной записи с помощью верхнего меню необходимо выполнить следующие действия:

– включить экран МУ;

– открыть верхнее меню, проведя от верхнего края Экрана блокировки к нижнему. В левом нижнем углу будет отображаться текущая роль: «Владелец устройства», если МУ загружено в режиме администратора, либо имя пользователя, если МУ загружено в режиме пользователя (Рисунок 6).

При нахождении в режиме пользователя с помощью верхнего меню можно осуществить переход к учетной записи администратора, выполнив следующие действия:

– открыть верхнее меню, проведя от верхнего края Экрана блокировки к нижнему;

– коснуться поля с именем учетной записи пользователя;

– в раскрывающемся списке выбрать «Владелец устройства» (Рисунок 7);

– дождаться, когда ОС Аврора переключится на учетную запись администратора;

– процесс переключения займет несколько секунд, далее необходимо убедиться, что текущая роль «Владелец устройства»: слева от названия роли «Владелец устройства» должна отображаться пометка «Вы» (Рисунок 8).

Рисунок 6

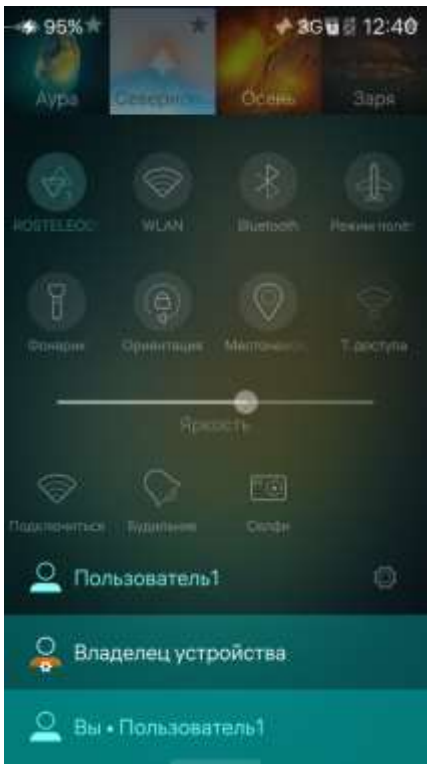

Рисунок 7

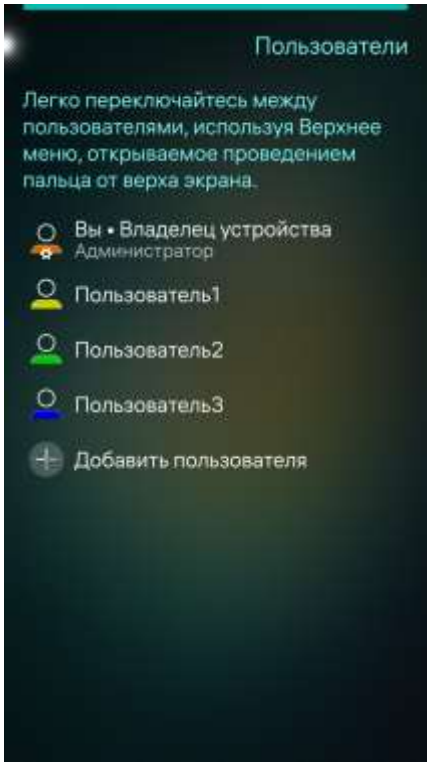

Рисунок 8

Для проверки активной учетной записи с помощью системных настроек необходимо выполнить следующие действия:

– коснуться пункта «Пользователи» в меню системных настроек, в результате чего отобразится страница «Пользователи» со списком учетных записей пользователей, созданных на МУ (Рисунок 8);

При работе МУ в режиме администратора в поле «Владелец устройства» отображается пометка «Вы», означающая, что текущий пользователь МУ наделен ролью администратора (Рисунок 8). Для перехода в режим пользователя необходимо выполнить следующие действия:

– коснуться строки с именем одной из учетных записей пользователей, созданных на МУ;

– в контекстном меню коснуться пункта «Переключится на этого пользователя» (см. Рисунок 5);

– процесс переключения займет несколько секунд, далее необходимо ввести код безопасности учетной записи пользователя, на которого происходит переключение;

– при работе МУ в режиме пользователя в поле «[Имя пользователя]» отображается пометка «Вы» (Рисунок 9).

При нахождении в режиме пользователя с помощью меню системных настроек можно осуществить переход к учетной записи администратора, выполнив следующие действия:

– в перечне пользователей коснуться строки с пометкой «Владелец устройства»;

– в контекстном меню коснуться пункта «Переключится на этого пользователя» (Рисунок 10);

– процесс переключения займет несколько секунд, далее необходимо ввести кода безопасности администратора.

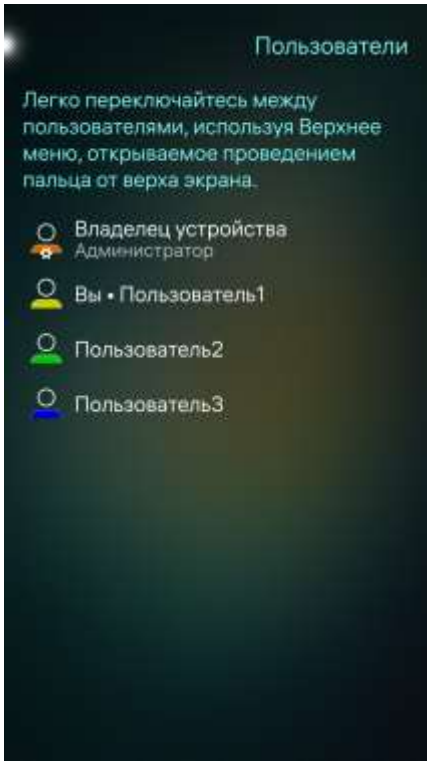

Рисунок 9

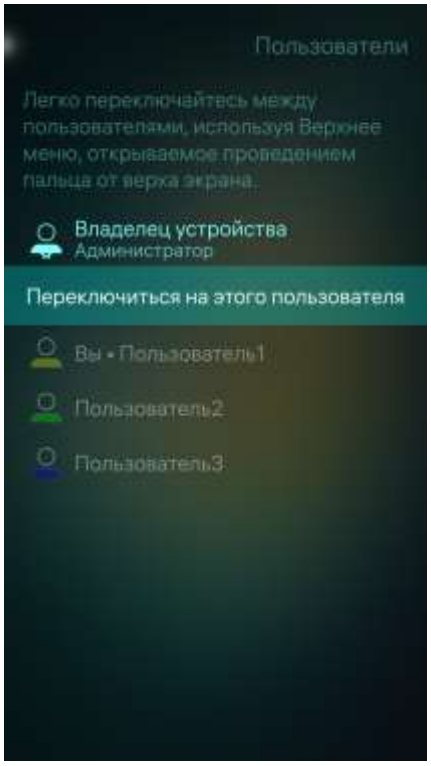

Рисунок 10

1.2.3. Управление исходящими голосовыми вызовами и SMS

Для отключения исходящих голосовых вызовов и SMS необходимо в контекстном меню коснуться пункта «Отключить голосовые вызовы и SMS» (см. Рисунок 5).

Далее у выбранного пользователя отобразится предупреждающий значок (Рисунок 11) и его возможности в МП «Телефон» и МП «Сообщения» будут ограничены.

Для включения исходящих голосовых вызовов и SMS администратору необходимо в контекстном меню коснуться пункта «Включить голосовые вызовы и SMS» (Рисунок 12), после чего предупреждающий значок перестанет отображаться у выбранного пользователя, и его возможности в МП «Телефон» и МП «Сообщения» будут восстановлены.

ПРИМЕЧАНИЕ. Подробное описание работы указанных МП, а также сообщений, выводимых на экран МУ при отключении голосовых вызовов и SMS, приведено в документе «Руководство пользователя».

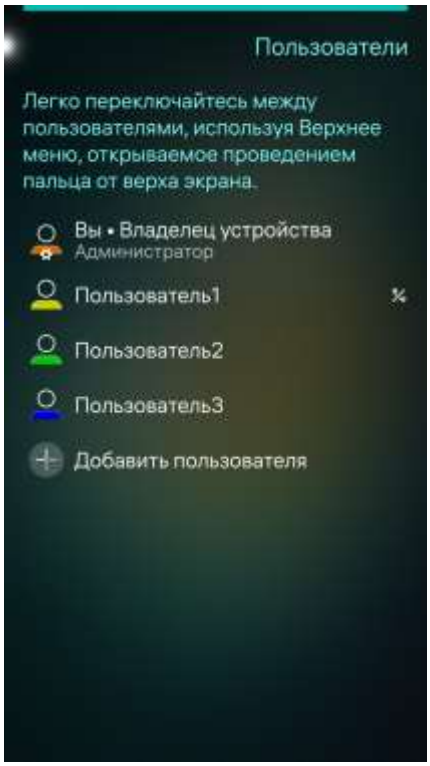

Рисунок 11

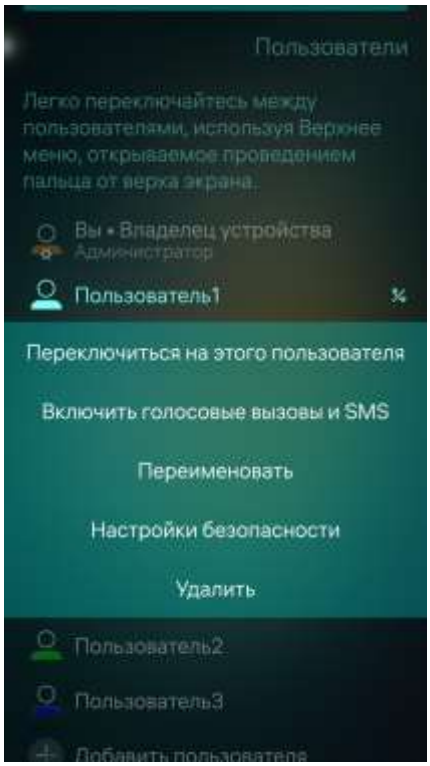

Рисунок 12

1.2.4. Переименование учетной записи

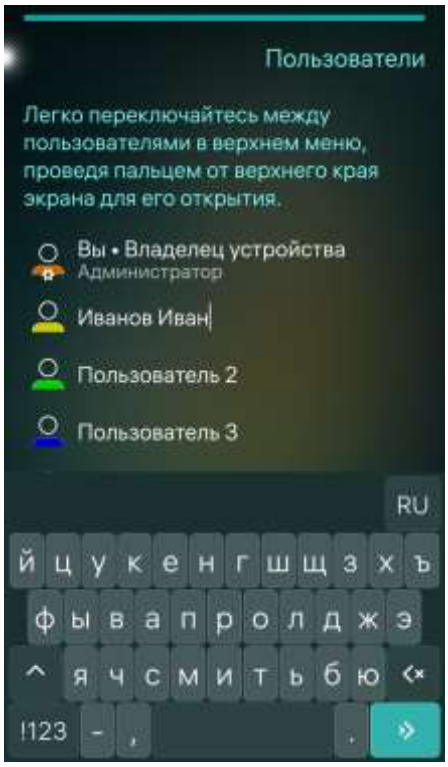

Рисунок 13

Для переименования учетной записи пользователя необходимо выполнить следующие действия:

– коснуться строки с именем учетной записи, которую необходимо переименовать;

– в контекстном меню коснуться пункта «Переименовать» (см. Рисунок 5);

– ввести новое имя учетной записи пользователя, например, Иванов Иван (Рисунок 13);

– коснуться значка » для подтверждения действия.

В результате учетная запись пользователя будет переименована.

ПРИМЕЧАНИЕ. При переименовании учетной записи изменяется только ее имя, логин остается неизменным.

1.2.5. Удаление учетной записи пользователя

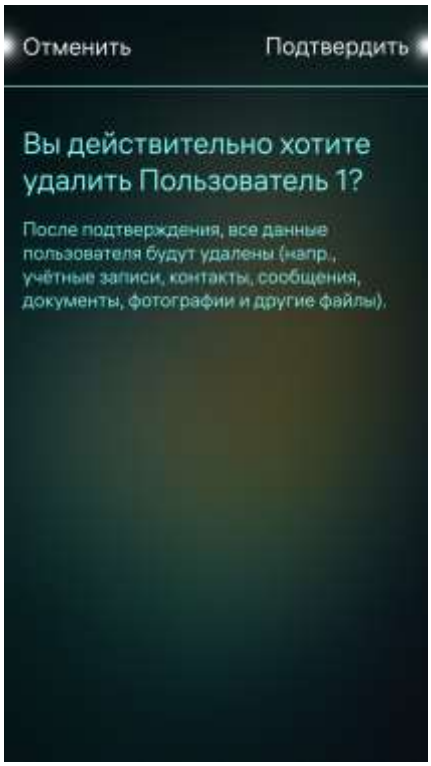

Рисунок 14

Для удаления учетной записи пользователя необходимо выполнить следующие действия:

– коснуться строки с именем учетной записи, которую необходимо удалить;

– в контекстном меню коснуться пункта «Удалить» (см. Рисунок 5);

– на отобразившейся странице коснуться кнопки «Подтвердить» для удаления учетной записи пользователя либо кнопки «Отменить» для отмены операции (Рисунок 14);

– для удаления подтвердить действие вводом текущего кода безопасности.

В результате с МУ будет удалена учетная запись и данные пользователя.

1.3. Установка даты и времени

ПРИМЕЧАНИЕ. Описание основных шагов первоначальной настройки МУ приведено в документе «Руководство пользователя», при этом установка и последующая настройка даты и времени доступна только администратору.

ВНИМАНИЕ! При первом включении МУ необходимо дважды ввести код безопасности.

Код безопасности будет запрашиваться и использоваться для:
- шифрования домашнего каталога пользователя (Рисунок 15);
- разблокировки МУ (Рисунок 16).

ПРИМЕЧАНИЕ. Шифрование раздела с домашними директориями пользователей (подраздел 4.2.2) происходит в следующих случаях:
- при первом включении МУ;
- после сброса МУ до заводского состояния (подраздел 2.2).

Рисунок 15

Рисунок 16

ПРИМЕЧАНИЕ. Изменения настроек в пункте меню «Время и дата» применяются ко всем учетным записям ролей, созданным на МУ.

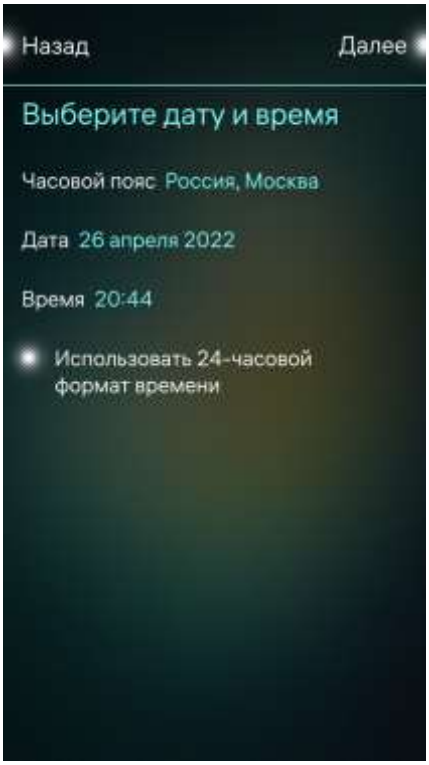

Рисунок 17

Для установки даты и времени необходимо убедиться, что все поля заполнены корректно, после чего коснуться кнопки «Далее» (Рисунок 17) либо изменить значения заполненных полей, выполнив следующие действия:

– изменить значения параметров «Часовой пояс», «Дата», «Время», коснувшись соответствующих полей;

– выбрать формат времени, коснувшись соответствующего переключателя;

ПРИМЕЧАНИЕ. Для активации переключателя достаточно коснуться поля, в котором он расположен: переключатель начнет светиться ярче, чем в состоянии по умолчанию (неактивном).

– коснуться кнопки «Далее».

Для последующей настройки часового пояса, текущих даты и времени необходимо выполнить следующие действия:

– в меню настроек системы коснуться пункта «Время и дата» , в результате чего отобразится страница настройки даты и времени (Рисунок 18);

– для автоматического обновления даты/времени коснуться переключателя «Автоматическое обновление времени» (Рисунок 19). Значения полей часового пояса, даты и времени станут недоступными для редактирования. В дальнейшем обновление даты/времени будет происходить автоматически;

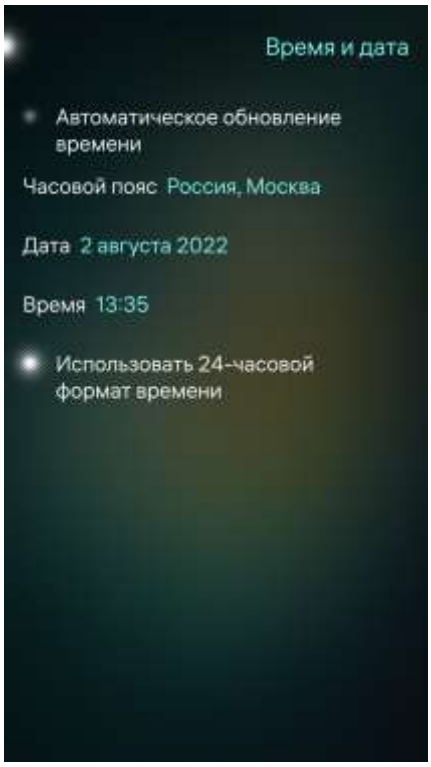

Рисунок 18

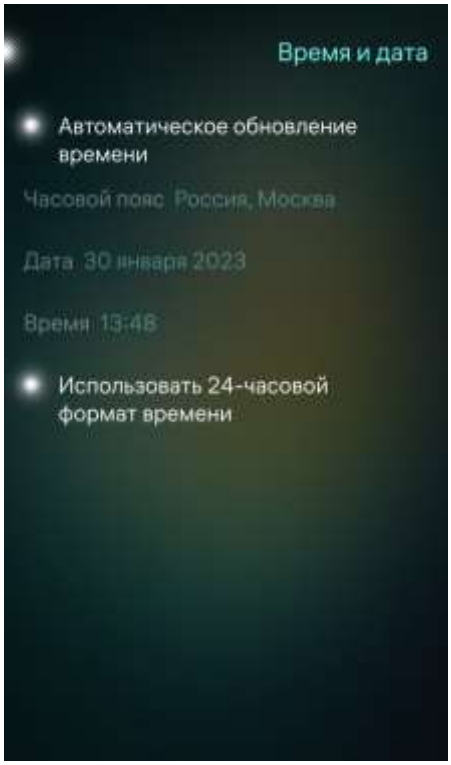

Рисунок 19

ПРИМЕЧАНИЕ. Для установки часового пояса, времени и даты вручную опция автоматического обновления времени должна быть выключена.

В случае необходимости задать часовой пояс, время и дату вручную требуется выполнить следующие действия:

– для установки часового пояса: коснуться поля «Часовой пояс» (см. Рисунок 18) и на открывшейся странице выбрать необходимое значение. Для ускорения процесса выбора можно воспользоваться полем поиска (Рисунок 20). Далее выбрать формат времени, коснувшись переключателя «Использовать 24-часовой формат» (см. Рисунок 18) для отображения времени в соответствующем формате. Если данный пункт не активирован, время будет отображаться в 12-часовом формате с уточнением «до полудня» или «после полудня»;

– для установки даты: коснуться поля «Дата» (см. Рисунок 18), на открывшейся странице коснуться текущей даты и выбрать из списка текущий год, месяц и число (Рисунок 21). Далее коснуться кнопки «Подтвердить» для сохранения даты либо кнопки «Отменить» для отмены операции. В случае подтверждения выбранная дата отобразится на странице настройки даты и времени, в случае отмены дата останется прежней;

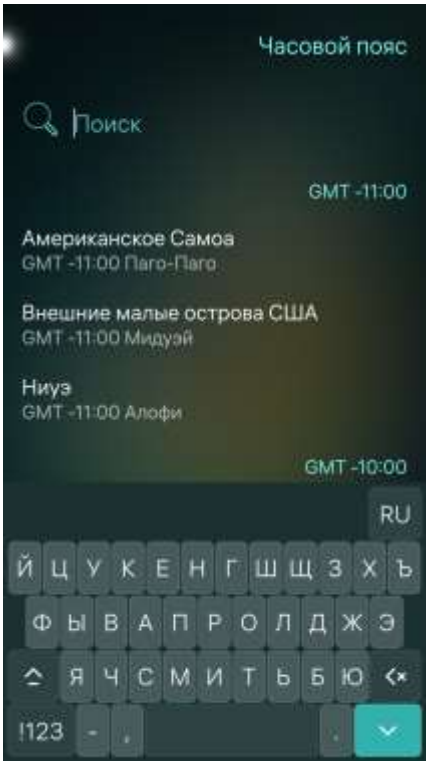

Рисунок 20

Рисунок 21

Рисунок 22

– для установки времени: коснуться поля «Время» (см. Рисунок 18), в результате чего отобразится циферблат, метка во внутреннем круге которого играет роль часовой стрелки, во внешнем — минутной (Рисунок 22). Для установки необходимого значения следует поочередно коснуться каждой из меток значка и, передвигая ее по или против часовой стрелки, установить в позиции, соответствующей текущему времени;

– коснуться кнопки «Подтвердить» для сохранения установленного времени либо кнопки «Отменить» для отмены операции. В случае подтверждения выбранная дата отобразится на странице настройки даты и времени, в случае отмены дата останется прежней.

1.4. Настройки безопасности

1.4.1. Настройка блокировки МУ

ПРИМЕЧАНИЯ:

1. Изменения настроек в пункте меню «Блокировка устройства» применяются ко всем учетным записям ролей, созданным на МУ;

2. Случаи, при которых код безопасности может быть запрошен, описаны в подразделе 4.2

Блокировка МУ позволяет обеспечить доступ к данным, хранящимся на МУ, только администратору.

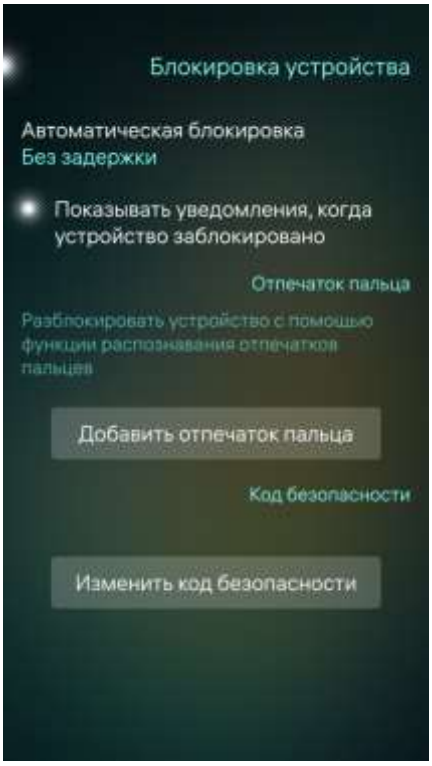

Рисунок 23

Для настройки блокировки МУ необходимо выполнить следующие действия:

– коснуться пункта «Блокировка устройства» в меню настроек безопасности, в результате чего отобразится страница с настройками блокировки МУ (Рисунок 23);

– коснуться поля «Автоматическая блокировка» и на открывшейся странице выбрать время до автоматической блокировки МУ (Рисунок 24, Рисунок 25), коснувшись соответствующих полей и подтвердив действие вводом текущего кода безопасности;

ВНИМАНИЕ! Варианты значений в поле «Автоматическая блокировка» отличаются в зависимости от версии ОС Аврора (Рисунок 24, Рисунок 25).

ВНИМАНИЕ! Если установленное время блокировки превышает время спящего режима, экран может быть неактивен, при этом МУ не будет заблокировано. Для дальнейшей работы с МУ необходимо нажать кнопку питания и продолжить работу без ввода кода безопасности.

ПРИМЕЧАНИЕ. При превышении количества попыток ввода неверного кода безопасности МУ автоматически будет заблокировано. Время блокировки является фиксированным и составляет 15 минут.

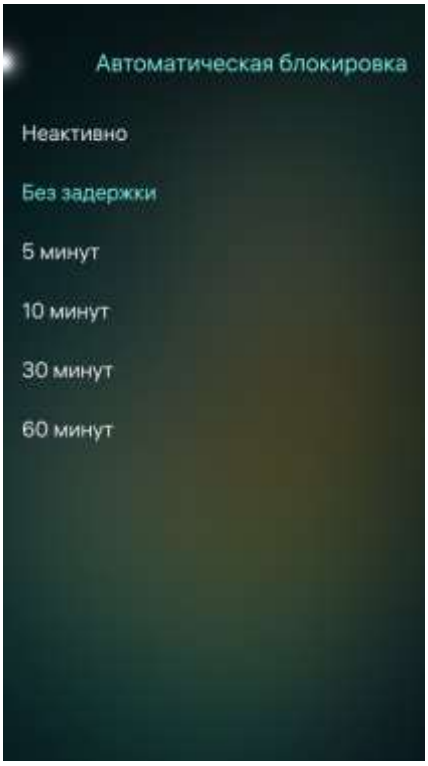

Рисунок 24

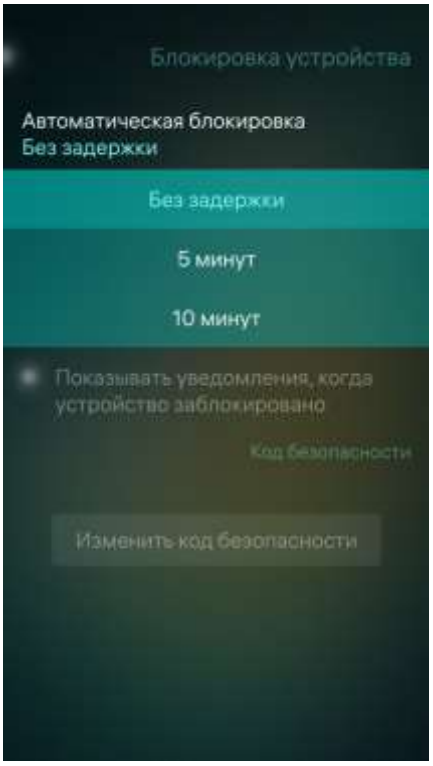

Рисунок 25

Рисунок 26

Для изменения текущего кода безопасности необходимо выполнить следующие действия:

– коснуться кнопки с соответствующим названием (см. Рисунок 25) и подтвердить действие вводом текущего кода безопасности;

– дважды ввести новый код безопасности.

На МУ предусмотрена возможность настроить блокировку с помощью функции распознавания отпечатка пальца.

ПРИМЕЧАНИЕ. Наличие указанной функции и расположение датчика отпечатка пальца зависит от конструктивных особенностей МУ, а также от версии ОС Аврора. Подробная информация о настройке блокировки с помощью функции распознавания отпечатка пальца приведена в документе «Руководство пользователя».

1.4.2. Настройка парольной политики (код безопасности)

ПРИМЕЧАНИЕ. Изменения настроек парольной политики будут применены ко всем учетным записям, созданным на МУ.

Для перехода к настройкам парольной политики необходимо выполнить следующие действия:

– в меню настроек системы коснуться пункта «Пользователи» , в результате чего отобразится страница «Пользователи» с представленным списком пользователей, созданных на МУ и открыть меню действий;

– коснуться пункта «Общие настройки» (Рисунок 27), в результате чего отобразится страница «Общие настройки» (Рисунок 28);

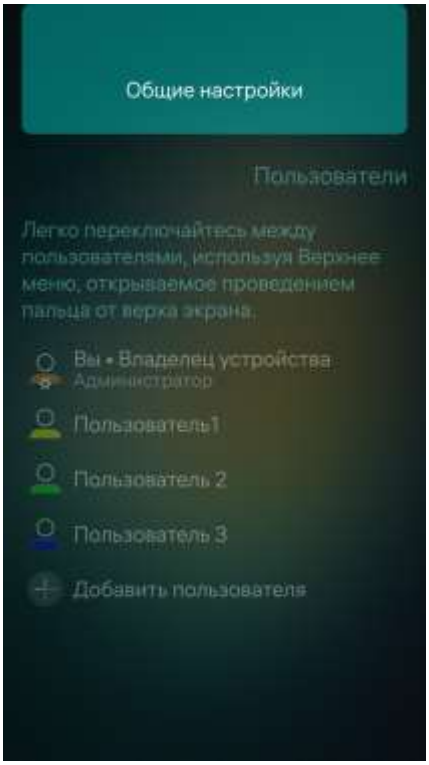

Рисунок 27

Рисунок 28

– коснуться пункта «Настройка политики паролей» и на отобразившейся странице коснуться поля «Настройка политики паролей» (Рисунок 29).

ВНИМАНИЕ! Наличие функции выбора предустановленной политики паролей зависит от версии ОС Аврора.

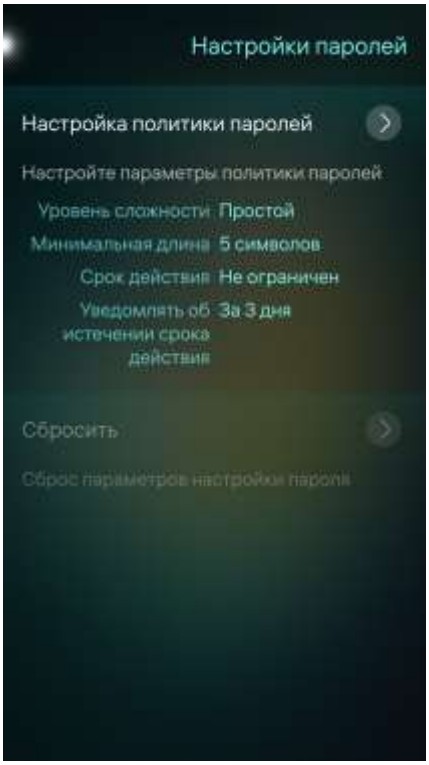

Рисунок 29

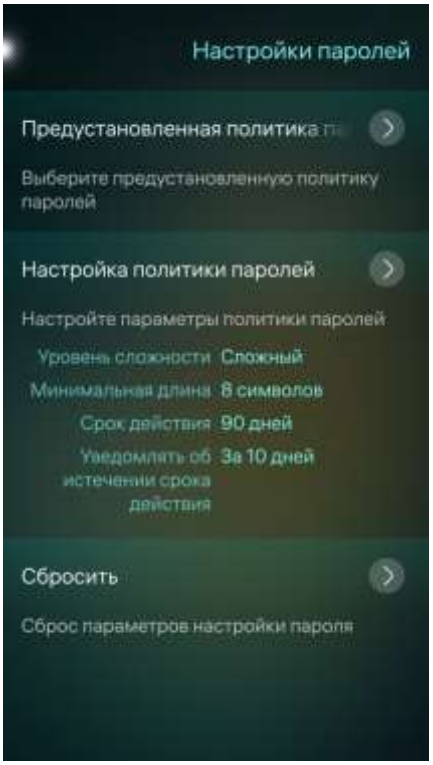

Рисунок 30

Для выбора политики паролей необходимо выполнить следующие действия:

– коснуться поля «Предустановленная политика паролей» (Рисунок 30);

– на открывшейся странице «Предустановка политики паролей» выбрать необходимую политику касанием соответствующего поля (Рисунок 31), в результате чего будет установлена необходимая политика паролей с заданными параметрами.

Для настройки политики паролей на открывшейся странице «Настройки парольной политики» необходимо задать необходимые параметры для кода безопасности (Рисунок 32):

– уровень сложности;
– длина;
– количество попыток;
– срок действия;
– уведомление об истечении срока действия.

Рисунок 31

Рисунок 32

Для настройки уровня сложности кода безопасности необходимо выполнить следующие действия (Рисунок 33):

– коснуться поля «Уровень сложности» и в раскрывающемся списке выбрать значение «Простой», состоящий только из цифр, либо «Сложный», который должен содержать цифру, букву, заглавную букву и специальный символ;

ПРИМЕЧАНИЕ. В случае выбора значения «Сложный» подтвердить действие вводом текущего кода безопасности.

Для настройки длины кода безопасности необходимо (Рисунок 34) установить количество символов, перемещая слайдер «Минимальная длина пароля» влево для уменьшения количества входящих в код безопасности символов либо вправо для увеличения количества символов и подтвердить действие вводом текущего кода безопасности.

ПРИМЕЧАНИЕ. Предусмотрена возможность установить длину кода безопасности от 5 до 12 символов.

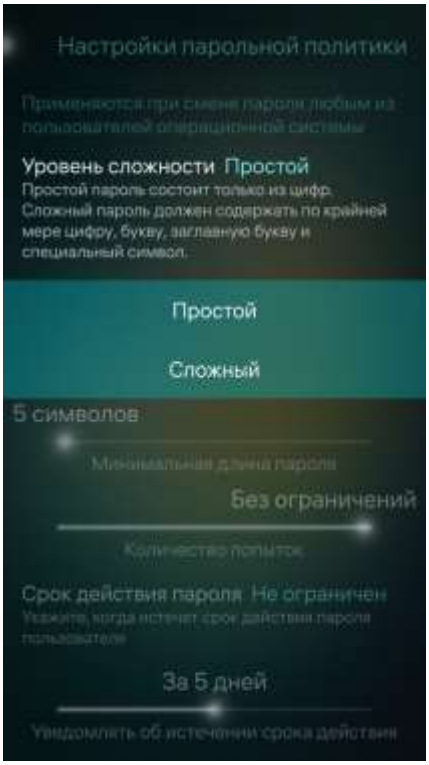

Рисунок 33

Рисунок 34

Для установки количества попыток ввода кода безопасности необходимо переместить слайдер «Количество попыток»:

– влево для уменьшения (минимальное значение: 4 попытки);

– вправо для увеличения (максимальное значение: 10 попыток либо «Без ограничений»).

ВНИМАНИЕ! Максимальное значение слайдера «Количество попыток» зависит от версии ОС Аврора.

Для задания срока действия кода безопасности необходимо выполнить следующие действия:

– коснуться поля «Срок действия пароля» (см. Рисунок 34) и на отобразившейся странице выбрать одно из значений (Рисунок 35) и подтвердить действие вводом текущего кода безопасности.

ВНИМАНИЕ! По требованиям безопасности ИС, в которых планируется использование МУ, функционирующего под управлением ОС Аврора, срок действия пароля должен быть установлен в значение «180 дней».

Для задания времени уведомления об истечении срока действия кода безопасности необходимо выполнить следующие действия (Рисунок 36):

– переместить слайдер «Уведомлять об истечении срока действия» влево для уменьшения количества дней либо вправо для увеличения количества дней, за которое пользователь начнет получать уведомления об истечении срока действия кода безопасности и подтвердить действие вводом текущего кода безопасности.

ПРИМЕЧАНИЕ. Предусмотрена возможность установить значение от 0 (Никогда) до 10 дней.

Рисунок 35

Рисунок 36

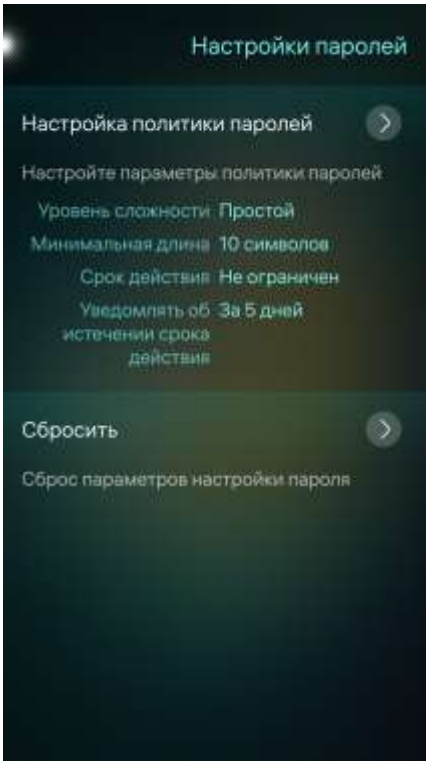

Рисунок 37

Для сброса парольной политики необходимо на странице «Настройка паролей» коснуться поля «Сбросить» и подтвердить действие вводом текущего кода безопасности (Рисунок 37).

1.4.3. Настройка безопасности учетной записи пользователя

ПРИМЕЧАНИЕ. Изменения настроек безопасности применяются только к выбранной учетной записи пользователя.

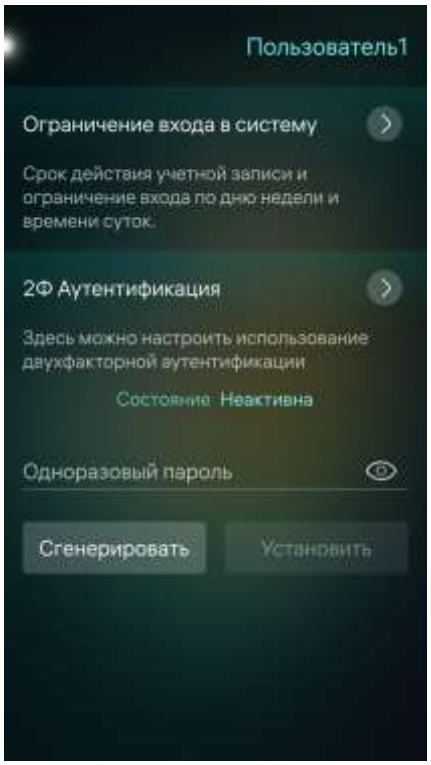

Рисунок 38

Для настройки безопасности учетной записи пользователя необходимо выполнить следующие действия:

– коснуться одной из созданных на МУ учетных записей пользователя (см. Рисунок 5);

– в контекстном меню коснуться пункта «Настройки безопасности»;

– на отобразившейся странице «[Имя учетной записи пользователя]» (Рисунок 38) можно выполнить следующие действия:

- задать для учетной записи пользователя ограничения входа в систему (пп. 1.4.3.1);
- включить и настроить двухфакторную аутентификацию (2ФА) (пп. 1.4.3.2);
- задать одноразовый пароль (пп. 1.4.3.3).

1.4.3.1. Ограничения входа в систему

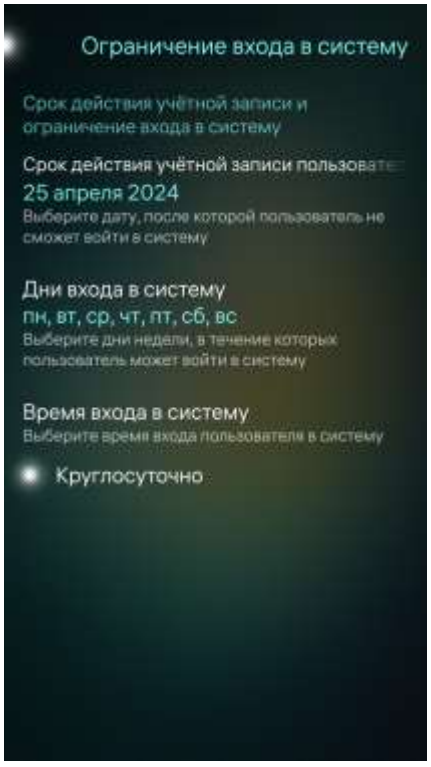

Рисунок 39

При необходимости установить для учетной записи пользователя ограничения входа в систему следует коснуться пункта с соответствующим названием и на открывшейся странице настроить следующие параметры (Рисунок 39):

– срок действия учетной записи пользователя;

– дни входа в систему;

– время входа в систему.

Для настройки срока действия учетной записи пользователя необходимо коснуться поля «Срок действия учетной записи пользователя» и на отобразившейся странице выбрать дату, после которой пользователь не сможет войти в систему (см. Рисунок 21), а также подтвердить действие вводом текущего кода безопасности.

Для выбора дней, в которые пользователь сможет войти в систему, необходимо коснуться поля «Дни входа в систему» и на отобразившейся странице выбрать дни недели, в течение которых пользователь сможет войти в систему (Рисунок 40), а также подтвердить действие вводом текущего кода безопасности.

Для настройки времени входа в систему необходимо выполнить следующие действия:

– коснуться переключателя «Круглосуточно» для его деактивации, в результате чего отобразятся поля ввода диапазона времени, в течение которого пользователь сможет войти в систему (Рисунок 41);

ПРИМЕЧАНИЕ. Для активации переключателя достаточно коснуться поля, в котором он расположен: переключатель начнет светиться ярче, чем в состоянии по умолчанию (неактивном).

– коснуться соответствующих полей для установки интервала времени, в течение которого пользователь сможет войти в систему (подраздел 1.3) и подтвердить действие вводом текущего кода безопасности.

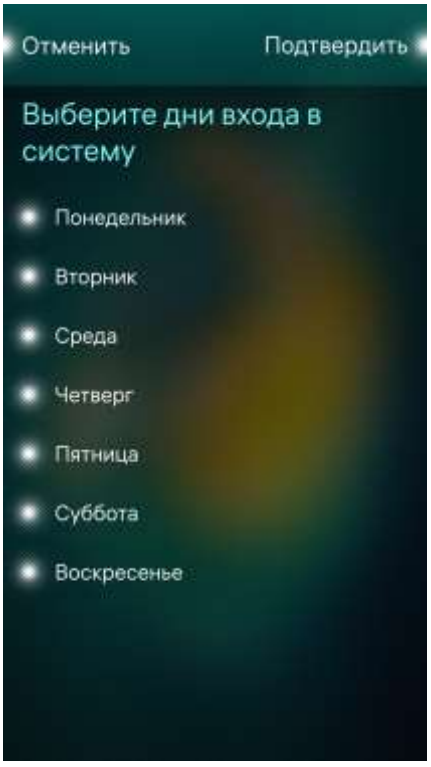

Рисунок 40

Рисунок 41

1.4.3.2. Настройка 2ФА

2ФА – процесс подтверждения подлинности пользователя с помощью применения нескольких различающихся факторов, которые в ОС Аврора применяются:

– в качестве первого фактора: пароль;

– в качестве второго фактора: токен, содержащий уникальную информацию пользователя.

ПРИМЕЧАНИЯ:

1. Информация о состоянии 2ФА отображается на странице «[Имя пользователя]» в пункте меню «Двухфакторная аутентификация»;

2. Для настройки и активации 2ФА администратору необходимо использовать USB смарт-карту (токен).

В ОС Аврора для 2ФА поддерживаются следующие токены:

– по предоставлению сертификата открытого ключа, расположенного на программно-аппаратном комплексе аутентификации и хранения информации «Рутокен» версии 4 (ЭЦП PKI) (сертификат ФСТЭК России №3753);

– средства аутентификации и безопасного хранения информации пользователей JaCarta (сертификат ФСТЭК России №3449).

ВНИМАНИЕ! Использование для 2ФА токена, отличного от указанных, не предусматривается.

1.4.3.2.1. Правила настройки и использования 2ФА

Необходимо учитывать следующие основные правила настройки и использования 2ФА:

– эксплуатацию токена следует осуществлять согласно требованиям, указанными в соответствующей документации на него;

– для обеспечения подключения к МУ и последующей настройки токена требуется использовать специализированный USB OTG переходник, который не входит в комплект поставки МУ;

– политика безопасности может запрещать применение внешних USB-устройств, для этого необходимо дополнительно проверить установленное ограничение действующей в ОС Аврора политики безопасности (подраздел 4.3.6);

– при работе с токеном потребуется дополнительный пароль для доступа в защищенную область памяти токена, в которую производится назначение и сохранение аутентификационной информации пользователя;

– использование 2ФА доступно для всех учетных записей пользователей (п. 1.2.1), созданных на МУ.

ПРИМЕЧАНИЕ. Проверка токена при входе пользователя происходит однократно – только при первичном входе.

1.4.3.2.2. Предварительная подготовка токена

Настройка токена происходит на электронно-вычислительной машине (ЭВМ) ОС Linux, на которой предварительно должен быть установлен пакет opensc.

ПРИМЕЧАНИЕ. Для настройки 2ФА токен должен иметь формат PKCS#15.

В случае если токен имеет другой формат, для переинициализации токена в формат PKCS#15 на ЭВМ необходимо выполнить следующие команды:

```
pkcs15-init --erase-card -p rutoken_ecp
pkcs15-init --create-pkcs15 --so-pin "87654321" --so-puk ""
pkcs15-init --store-pin --label "User PIN" --auth-id 02 --pin
"12345678" --puk "" --so-pin "87654321" --finalize
```

ВНИМАНИЕ! После переинициализации токена все данные с него будут удалены.

1.4.3.2.3. Включение и выключение 2ФА

ПРИМЕЧАНИЕ. Включение 2ФА возможно только для выбранной учетной записи пользователя.

Для включения 2ФА необходимо выполнить следующие действия:

– коснуться одной из созданных учетных записей пользователя (см. Рисунок 20);

– в контекстном меню коснуться пункта «Настройки безопасности»;

– отобразится страница «[Имя учетной записи пользователя]» (см. Рисунок 32), на которой необходимо коснуться пункта «2Ф Аутентификация»;

– на отобразившейся странице коснуться кнопки «Начать настройку» для настройки 2ФА (Рисунок 42);

– подключить токен (смарт-карту). После успешного подключения на экране МУ отобразится соответствующее сообщение (Рисунок 43);

– на отобразившейся странице «Инициализация смарт-карты» коснуться кнопки «Ввести пароль» для ввода пароля от токена (смарт-карты) либо коснуться кнопки «Попробуйте другую смарт-карту» для подключения другого токена (Рисунок 44);

– в поле ввода ввести пароль от токена(смарт-карты) и коснуться значка  (Рисунок 45);

– после успешной инициализации токена (смарт-карты) на экране отобразится соответствующее сообщение;

– коснуться кнопки «Подтвердить» для подтверждения либо кнопки «Отменить» для отмены операции (Рисунок 46);

– на отобразившейся странице коснуться кнопки «Завершить» для завершения настройки 2ФА (Рисунок 47), после чего значение поля «Состояние» изменится на «Активна» (Рисунок 49).

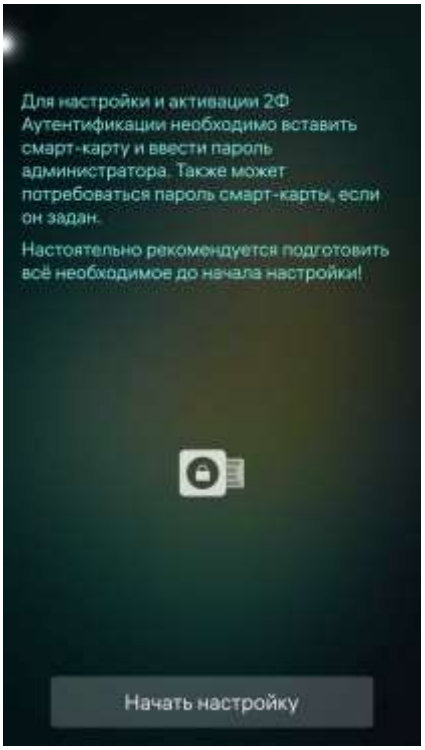

Рисунок 42

Рисунок 43

Рисунок 44

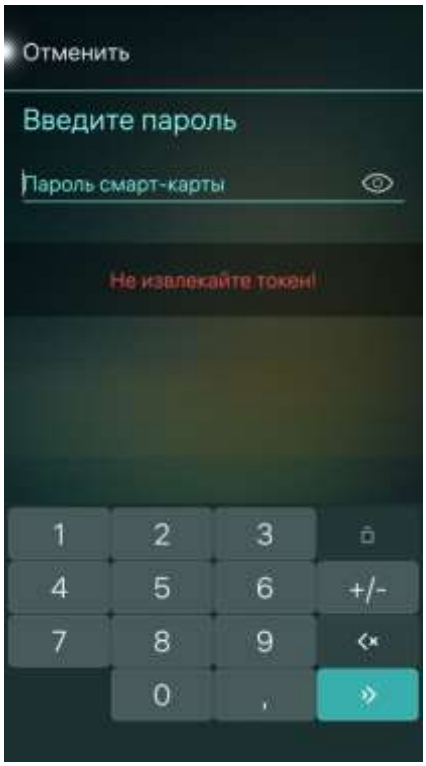

Рисунок 45

Рисунок 46

Рисунок 47

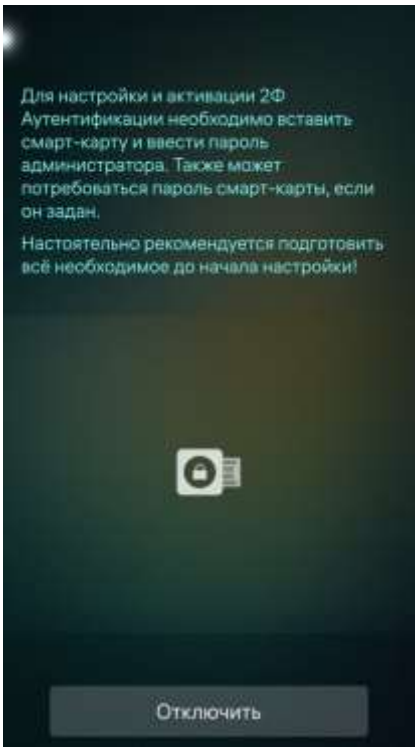

Рисунок 48

Для выключения 2ФА необходимо выполнить следующие действия:

– на странице «[Имя учетной записи пользователя]» (см. Рисунок 38) коснуться пункта «2Ф Аутентификация»;

– на отобразившейся странице коснуться кнопки «Отключить» для отключения токена (Рисунок 48), после чего значение поля «Состояние» изменится на «Неактивна» (см. Рисунок 38).

1.4.3.3. Задание одноразового пароля для учетной записи пользователя

Одноразовый пароль требуется для выполнения входа в систему при первом включении учетной записи пользователя.

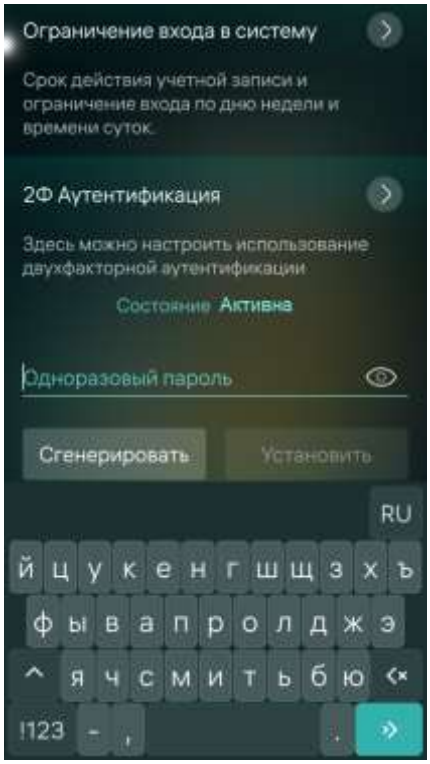

Рисунок 49

ПРИМЕЧАНИЕ. Подробная информация о входе в учетную запись пользователя с помощью одноразового пароля приведена в документе «Руководство пользователя».

Для задания одноразового пароля учетной записи пользователя необходимо выполнить следующие действия:

– коснуться кнопки «Сгенерировать» (Рисунок 49) для получения пароля, сгенерированного случайным образом либо установить курсор в поле «Одноразовый пароль», после чего задать пароль;

– при необходимости коснуться значка для отображения пароля;

– коснуться кнопки «Установить» (Рисунок 49) и подтвердить действие вводом текущего кода безопасности.

1.5. Настройка МУ

Администратору МУ доступны следующие возможности:

– настройка USB-подключения (п. 1.5.1);

– настройка PIN-кода для SIM-карты (п. 1.5.2);

– настройка МП (п. 1.5.3).

1.5.1. Настройка USB-подключения

Для настройки USB-подключения необходимо выполнить следующие действия:

– коснуться пункта «USB» в меню настроек сети;

– в открывшемся окне настроек USB-подключения (Рисунок 50) коснуться поля «Режим USB по умолчанию» и выбрать необходимое значение из раскрывающегося списка (Рисунок 50).

ПРИМЕЧАНИЯ:

1. В случае выбора пункта «Всегда спрашивать» при подключении МУ к ЭВМ с помощью USB-кабеля на МУ отобразится окно с выбором режима USB-подключения (Рисунок 51);

2. Значение «Режим разработчика» отображается только при активации соответствующих переключателей в разделе «Средства разработчика» системных настроек (подраздел 3.1).

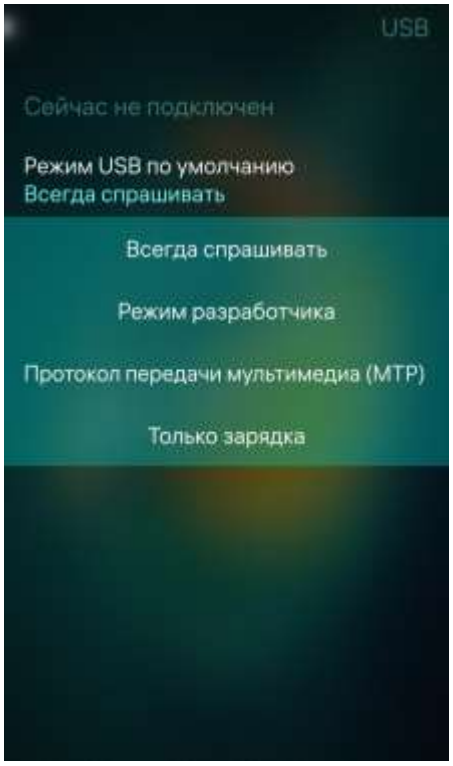

Рисунок 50

Рисунок 51

1.5.2. Настройка PIN-кода для SIM-карты

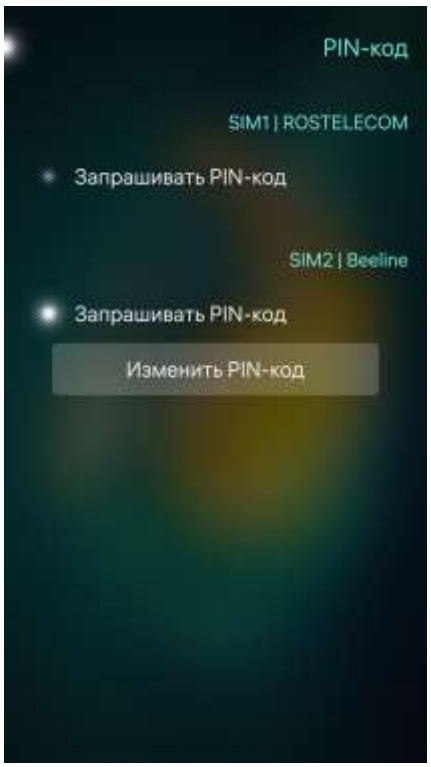

Рисунок 52

Защита установленной на МУ SIM-карты обеспечивается с помощью PIN-кода, который можно активировать/деактивировать отдельно для каждой из SIM-карт.

ПРИМЕЧАНИЕ. В зависимости от конструктивных особенностей МУ допускается установка до двух SIM-карт.

Для настройки PIN-кода необходимо выполнить следующие действия (Рисунок 52):

– коснуться пункта «PIN-код» в меню настроек безопасности;

– коснуться переключателя «Запрашивать PIN-код» тех SIM-карт, которые необходимо защитить вводом PIN-кода либо снять защиту.

После активации PIN-кода он будет запрашиваться при каждом включении МУ (Рисунок 53). В случае трехкратного ввода неверного PIN-кода SIM-карта будет заблокирована и для ее разблокировки потребуется PUK-код, для ввода которого предоставляется 10 попыток (Рисунок 54).

ПРИМЕЧАНИЕ. PUK-код предоставляется оператором сотовой связи.

После ввода верного PUK-кода отобразится страница для изменения PIN-кода (см. Рисунок 52), на которой необходимо выполнить следующие действия:

– коснуться кнопки «Изменить PIN-код»;

– внести соответствующие изменения в разделе SIM-карты, которую требуется защитить вводом PIN-кода.

Рисунок 53

Рисунок 54

1.5.3. Настройки МП

ПРИМЕЧАНИЕ. Администратор имеет доступ к дополнительным настройкам МП, описанным как в настоящем документе, так и в документе «Руководство пользователя».

Для дополнительной настройки МП необходимо выполнить следующие действия:

– открыть меню настроек касанием значка на Экране приложений (см. Рисунок 1);

– коснуться раздела «Приложения» для перехода к странице с выбором МП;

– на странице настраиваемых МП коснуться значка соответствующего МП и задать требуемые настройки.

Рисунок 55

МП «Телефон»:

– сбросить счетчики звонков (Рисунок 55);

– включить опцию «Запись разговора» (Рисунок 56);

– просмотреть информацию о записанных вызовах (Рисунок 57).

Для выполнения действий над записанными вызовами следует коснуться поля с количеством записанных вызовов и на отобразившейся странице в контекстном меню вызова коснуться пункта с необходимым действием.

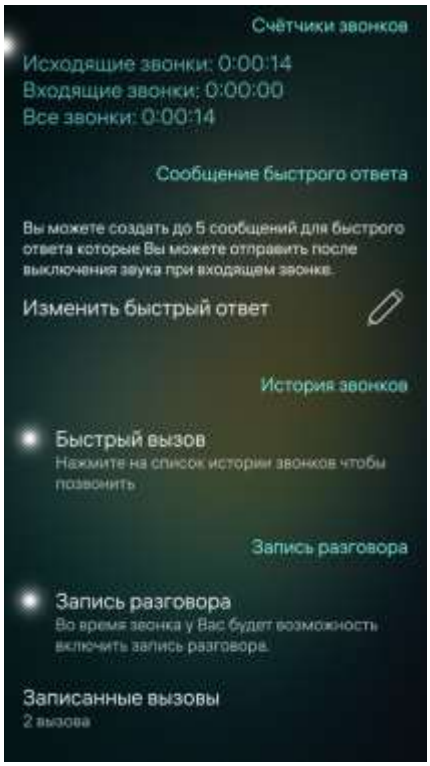

Рисунок 56

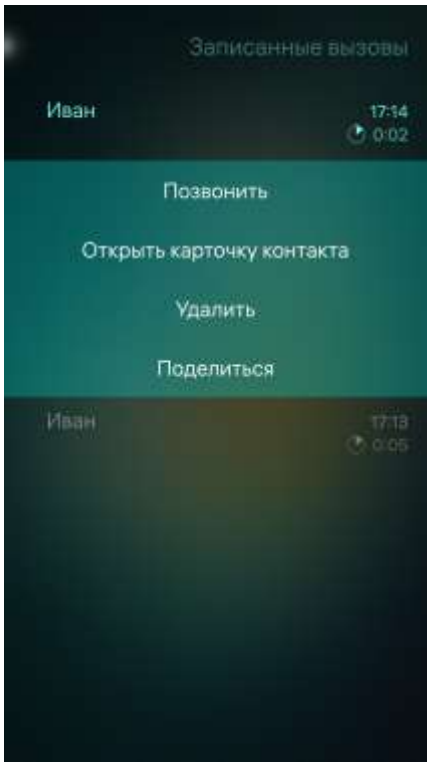

Рисунок 57

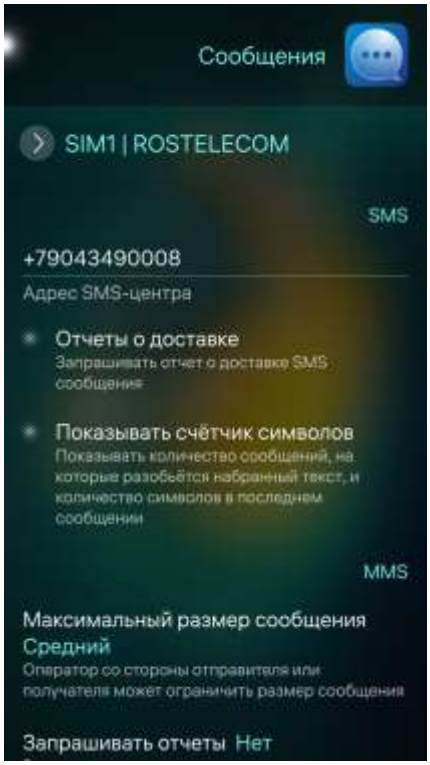

Рисунок 58

МП «Сообщения» (Рисунок 58):

– просмотреть адрес SMS-центра;
– включить опцию «Отчеты о доставке».

2. ВЫПОЛНЕНИЕ ПРОГРАММЫ

2.1. Настройка обновлений ОС Аврора

Обновление ОС Аврора осуществляется администратором либо локально вручную, либо удаленно с использованием Прикладного программного обеспечения «Аврора Центр» (ППО).

ПРИМЕЧАНИЕ. Для получения информации по обновлению ОС Аврора с использованием ППО следует обратиться к соответствующей документации на ППО, размещенной на веб-сайте: https://auroraos.ru/documentation/#!/tab/452394980-2.

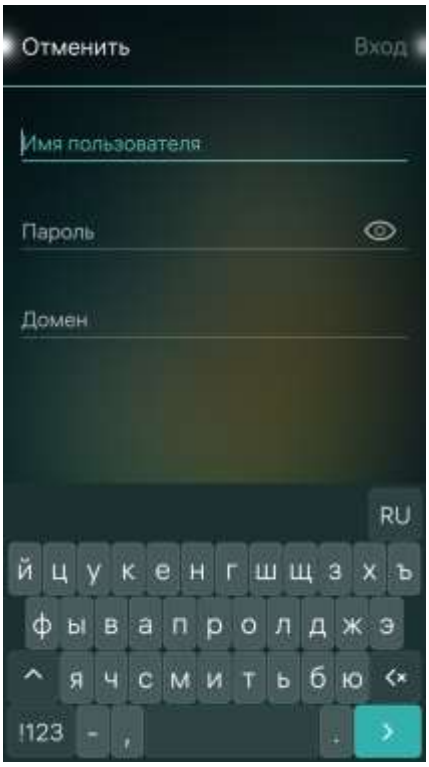

Рисунок 59

Для настройки обновлений ОС необходимо выполнить следующие действия:

– в меню системных настроек коснуться пункта «Обновления Аврора ОС» , в результате чего отобразится страница с настройками обновления;

– коснуться переключателя «Включить защищенные обновления» (Рисунок 60);

ВНИМАНИЕ! Для получения имени пользователя и пароля необходимо обратиться на электронную почту: dev-support@omp.ru.

– на открывшейся странице заполнить поля (Рисунок 59) для регистрации доступа к репозиториям;

– коснуться кнопки «Вход» для выполнения входа либо кнопки «Отменить» для отмены операции.

Ранее установленную версию ОС Аврора можно обновить до текущей локально через графический интерфейс, выполнив следующие действия:

– на странице с настройками обновления коснуться значка для проверки доступных обновлений, в результате чего отобразится сообщение: «Нет доступных обновлений», а также дата и время последней проверки (Рисунок 60) либо доступное обновление (Рисунок 61);

Рисунок 60

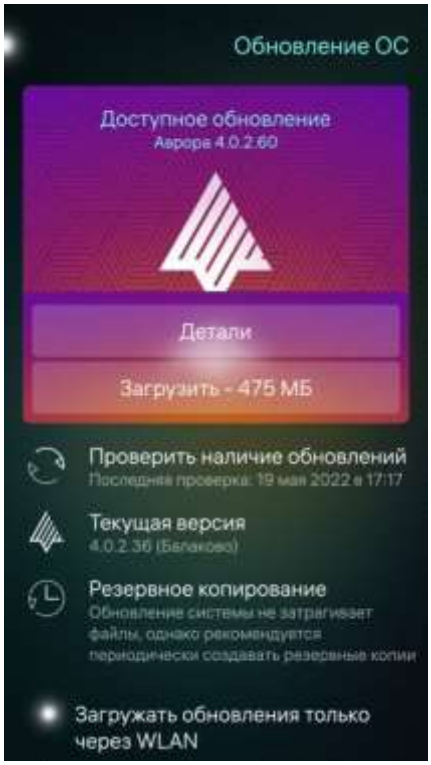

Рисунок 61

– при наличии доступного обновления коснуться кнопки «Загрузить – [Размер обновления]» для его загрузки. В случае необходимости загрузку обновления можно отменить касанием кнопки «Отменить загрузку» (Рисунок 62);

– коснуться кнопки «Детали» и на открывшейся странице ознакомится с подробной информацией об обновлении (Рисунок 63).

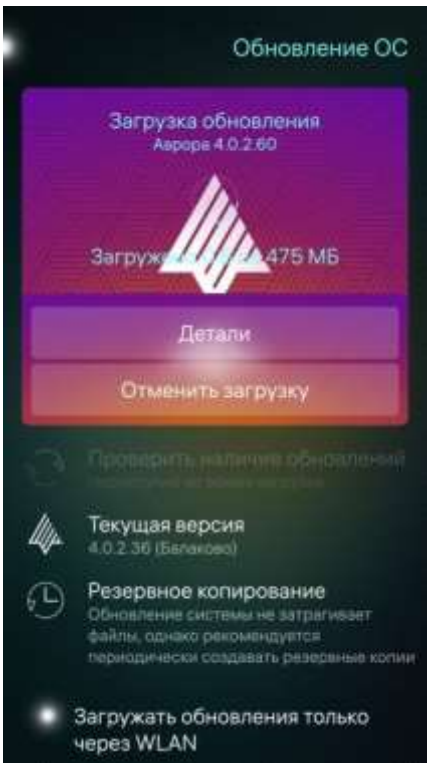

Рисунок 62

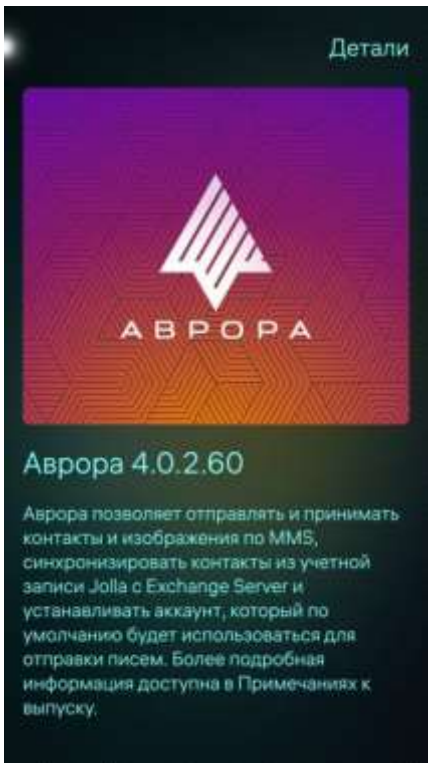

Рисунок 63

– после успешной загрузки обновления коснуться кнопки «Установить обновление» (Рисунок 64);

– на открывшейся странице коснуться кнопки «Установить» (Рисунок 65), после чего МУ автоматически будет перезагружено, либо кнопки «Отменить» для отмены операций.

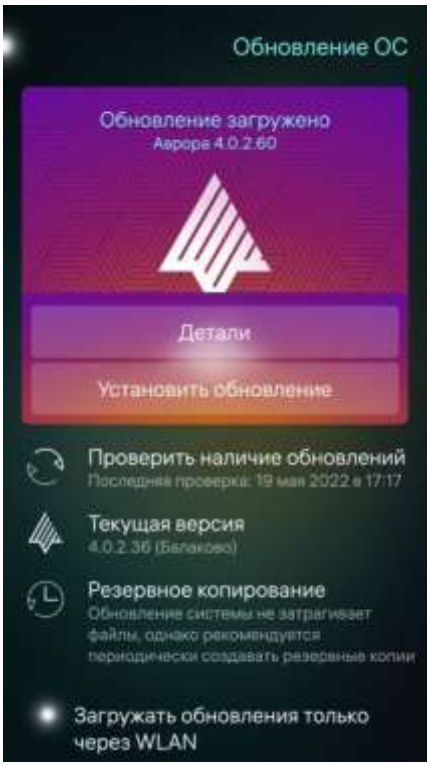

Рисунок 64

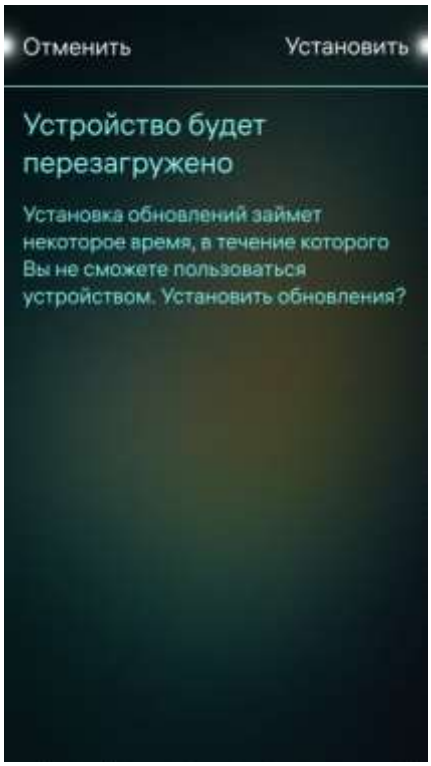

Рисунок 65

На странице «Обновление ОС» администратору также доступны следующие действия (см. Рисунок 60):

– просмотреть информацию о текущей версии ОС ;

– выполнить резервное копирование перед обновлением ОС, коснувшись значка ;

ПРИМЕЧАНИЕ. Подробная информация о создании резервной копии приведена в документе «Руководство пользователя».

– активировать либо деактивировать переключатель «Загружать обновления только через WLAN».

2.2. Сброс настроек МУ

ПРИМЕЧАНИЕ. Сброс настроек МУ – это процесс удаления всех данных, после которого МУ возвращается к заводскому состоянию, т.е. к версии ОС, установленной производителем МУ.

Для сброса настроек МУ до заводского состояния необходимо выполнить следующие действия:

– открыть меню настроек системы касанием значка на Экране приложений;

– в подразделе «Информация» коснуться пункта «Сбросить устройство» ;

– коснуться кнопки «Очистить устройство» (Рисунок 66);

– коснуться кнопки «Подтвердить» для подтверждения сброса настроек МУ либо кнопки «Отменить» для отмены операции (Рисунок 67);

– при необходимости коснуться переключателя «Автоматически перезагрузить устройство после сброса» для последующей перезагрузки МУ;

– при необходимости коснуться переключателя «Стереть все данные» для удаления данных.

ПРИМЕЧАНИЕ. После перезагрузки МУ произойдет сброс настроек до заводского состояния с последующим запуском мастера первоначальной настройки, подробное описание работы с которым приведено в документе «Руководство пользователя».

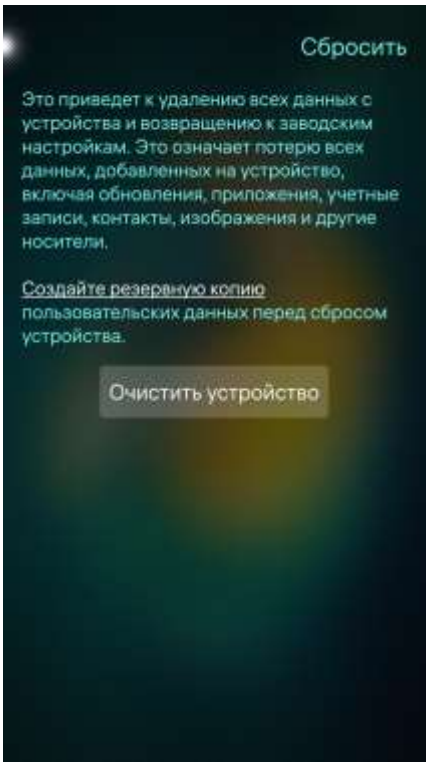

Рисунок 66

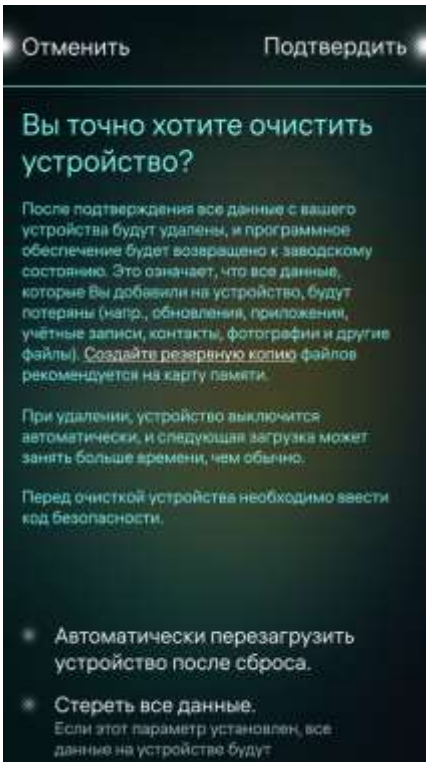

Рисунок 67

2.3. Установка стороннего ПО

Администратор имеет возможность устанавливать на МУ сторонние программы, не являющиеся встроенными в ОС Аврора.

Сторонние разработчики могут использовать официальный набор инструментов разработки ПО для ОС Аврора, подробная информация о котором приведена на веб-сайте: https://community.omprussia.ru.

При установке сторонних МП администратору необходимо убедится в выполнении следующих условий:

– пакет программ устанавливается штатным образом;
– необходимые исполняемые файлы программы запускаются;
– штатное поведение и выполнение программы сохраняется и после перезагрузки ОС Аврора.

Установка и управление сторонним ПО может осуществляться администратором следующими способами:

– локально:
  - через графический интерфейс МП «Файлы» (п. 2.3.1);
  - в МП «Terminal» (п. 2.3.2).
– удаленно:
  - принудительная установка МП с помощью политики;
  - установка МП с помощью ППО.

ПРИМЕЧАНИЕ. Подробная информация по использованию ППО приведена в соответствующей документации, расположенной на веб-сайте: https://auroraos.ru/documentation/#!/tab/452394980-2.

2.3.1. Установка ПО с помощью графического интерфейса МП «Файлы»

Для установки стороннего ПО локально через графический интерфейс с помощью МП «Файлы» необходимо выполнить следующие действия:

– открыть МП «Файлы», коснувшись значка  на Экране приложений;
– коснуться установочного файла МП;
– в диалоговом окне коснуться кнопки «Установить» (Рисунок 68).

После успешной установки на экране МУ отобразится уведомление (Рисунок 69), а значок установленного МП будет отображаться на Экране приложений.

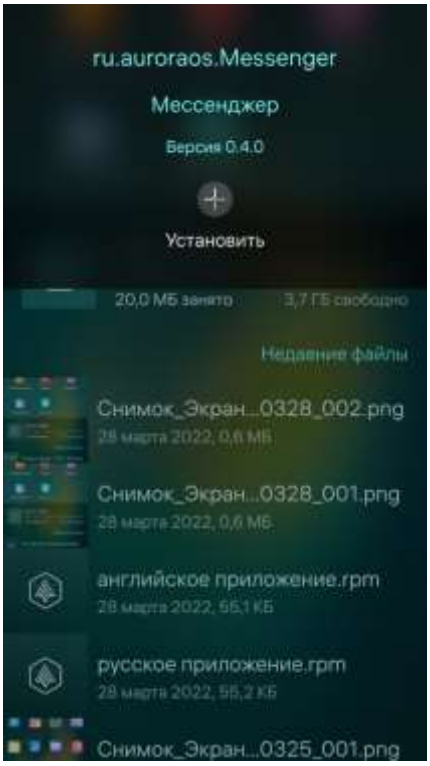

Рисунок 68

Рисунок 69

2.3.2. Установка ПО с помощью МП «Terminal»

Для установки стороннего ПО локально через графический интерфейс с помощью МП «Terminal» необходимо получить права суперпользователя (п. 2.4.2).

Управление МП осуществляется в МП «Terminal» с помощью менеджера RPM-пакетов посредством выполнения следующих команд:

– для установки скачанного RPM-пакета:

```
#rpm -ihv имя_пакета
```

– для удаления RPM-пакета:

```
#rpm -e имя_пакета
```

– для просмотра перечня установленных RPM-пакетов:

```
#rpm -qa
```

– для подробного ознакомления с возможностями менеджера RPM-пакетов:

```
#rpm -help
```

ПРИМЕЧАНИЕ. Для получения информации о возникающих ошибках и нештатных ситуациях в процессе управления МП необходимо использовать МП «Журнал» и ознакомиться с представленной в нем информацией. Подробное описание работы МП приведено в документе «Руководство пользователя».

2.4. Тонкая настройка

ВНИМАНИЕ! Администратору необходимо использовать МП «Terminal» только для осуществления действий по тонкой настройке МУ в соответствии с настоящим документом. Использование МП «Terminal» для других целей ЗАПРЕЩАЕТСЯ.

Тонкая настройка предназначена для выполнения сервисных операций, недоступных из графического интерфейса (например, настройка прав доступа, исправление конфигурационных файлов и т.д.)

МП «Terminal» является инструментом предоставления доступа к командной строке, где имеются следующие возможности:

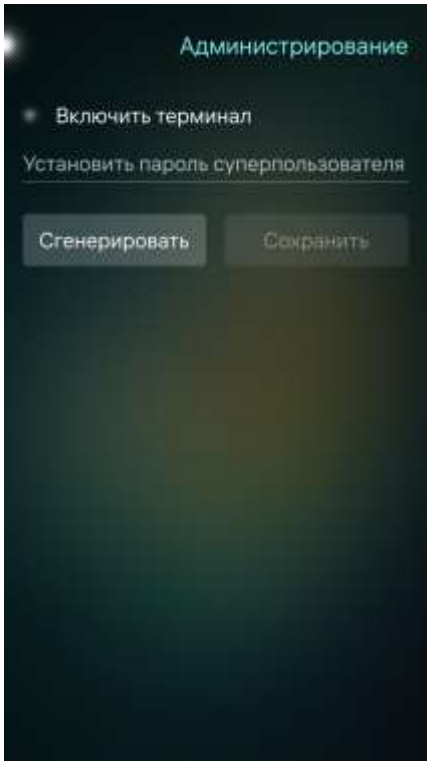

Рисунок 70

– вывод потока данных, а также диагностических и отладочных сообщений в текстовом виде;

– выполнение сервисных действий и более тонкую настройку МУ (в т.ч. с использованием прав суперпользователя).

ПРИМЕЧАНИЕ. По умолчанию МП «Terminal» не отображается на Экране приложений.

Для доступа к МП «Terminal» необходимо выполнить следующие действия:

– коснуться пункта «Администрирование» в меню системных настроек;

– на открывшейся странице коснуться переключателя «Включить терминал» (Рисунок 70) для отображения МП на Экране приложений;

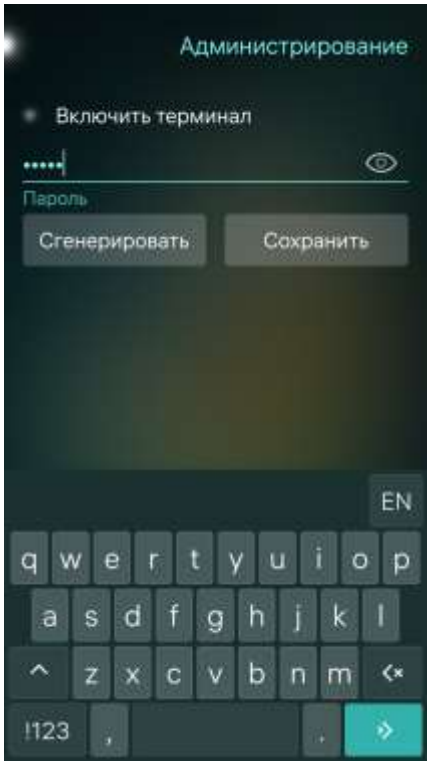

Рисунок 71

– коснуться поля ввода и ввести желаемый пароль либо кнопки «Сгенерировать» для получения пароля, сгенерированного случайным образом, который в дальнейшем будет использоваться для получения прав суперпользователя;

ПРИМЕЧАНИЕ. Необходимо запомнить заданный пароль.

– коснуться кнопки «Сохранить» (Рисунок 71), в результате чего МП «Terminal» отобразится на Экране приложений (см. Рисунок 1).

ПРИМЕЧАНИЕ. МП «Terminal» доступно под учетной записью администратора.

2.4.1. Настройка интерфейса МП «Terminal»

Для работы с МП «Terminal» его необходимо запустить, проведя по экрану снизу вверх и на Экране приложений коснувшись значка  .

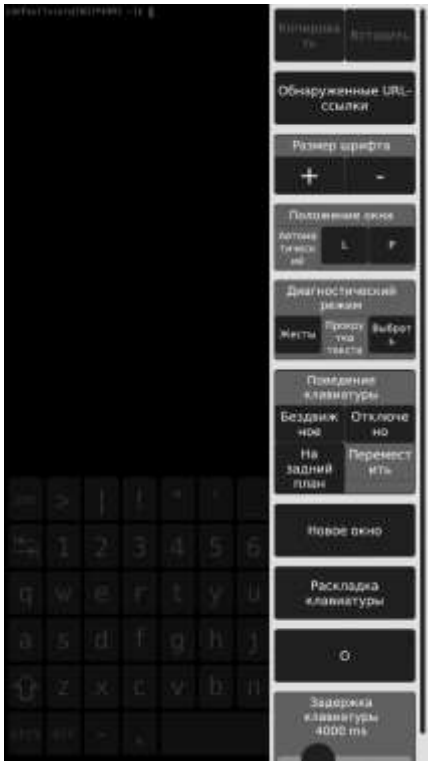

Рисунок 72

В интерфейсе МП «Terminal» необходимо коснуться значка  для отображения меню с настройками интерфейса, в котором доступны следующие возможности (Рисунок 72):

– копирование или вставка фрагментов текста;
– поиск URL-ссылок;
– создание нового окна;
– выбор языка интерфейса;
– просмотр информации о МП «Terminal»;
– увеличение и уменьшение размера шрифта;
– выбор ориентации окна;
– выбор действия при касании экрана;
– выбор способа отображения клавиатуры;
– установка времени задержки клавиатуры.

ПРИМЕЧАНИЕ. Выполнение данных действий осуществляется посредством касания соответствующих кнопок.

2.4.2. Получение прав суперпользователя

Рисунок 73

Права суперпользователя предоставляют администратору возможность работать в привилегированном режиме и выполнять любые операции в ОС Аврора.

Для получения прав суперпользователя необходимо открыть МП «Terminal» и выполнить следующие действия (Рисунок 73):

– провести по экрану снизу вверх и на Экране приложений коснуться значка  ;

– в МП выполнить команду: `devel-su;`

– указать заданный ранее пароль, в результате чего будет выполнен переход в режим суперпользователя.

3. СРЕДСТВА РАЗРАБОТЧИКА

ВНИМАНИЕ! Запрещается активация режима разработчика на МУ, функционирующем под управлением ОС Аврора в сертифицированной версии и предназначенном для использования в ИС, аттестованных по требованиям обеспечения безопасности в соответствии с законодательством Российской Федерации.

Режим разработчика предоставляет администратору доступ к расширенному функционалу настроек.

ВНИМАНИЕ! Режим разработчика невозможно отключить после активации. Для деактивации режима разработчика необходимо сбросить МУ до заводских настроек (подраздел 2.2).

3.1. Активация режима разработчика

ПРИМЕЧАНИЕ. Перед активацией режима разработчика необходимо убедиться в наличии на МУ доступа к сети Интернет.

Для активации режима разработчика необходимо выполнить следующие действия:

– коснуться пункта «Средства разработчика»  в меню системных настроек, в результате чего откроется страница «Инструменты разработчика»;

– коснуться переключателя «Режим разработчика» (Рисунок 74) и подтвердить действие вводом текущего кода безопасности;

– на открывшейся странице ознакомится с «Условиями разработчика» и коснуться кнопки «Подтвердить» для подтверждения активации режима разработчика либо кнопки «Отменить» для отмены операции (Рисунок 75).

Рисунок 74

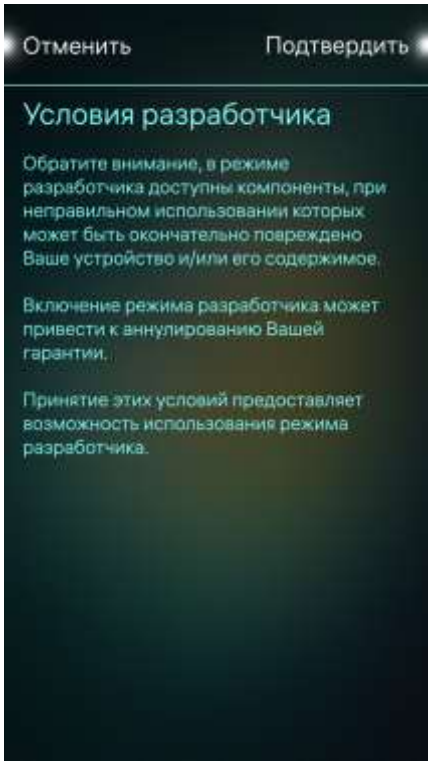

Рисунок 75

3.2. Инструменты разработчика

ПРИМЕЧАНИЕ. SSH-пароль используется для получения прав суперпользователя.

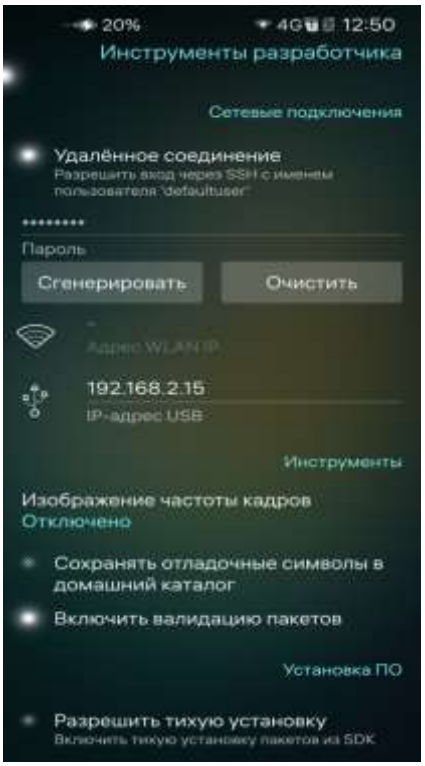

Рисунок 76

Для работы с инструментами разработчика необходимо выполнить следующие действия:

– активировать режим разработчика (подраздел 3.1);

– разрешить вход по SSH-паролю, коснувшись переключателя «Удаленное соединение» (Рисунок 76) и подтвердив действие вводом текущего кода безопасности;

– задать либо сгенерировать пароль, коснувшись поля «Сгенерировать», после чего коснуться кнопки «Сохранить» и подтвердить действие вводом текущего кода безопасности (Рисунок 76);

ПРИМЕЧАНИЕ. Поле «Установить пароль для SSH и доступа» отображается после активации переключателя «Удаленное соединение».

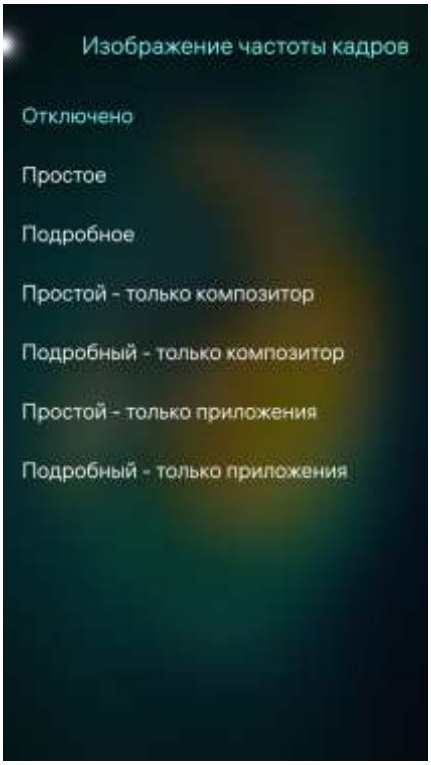

Рисунок 77

– настроить отображение частоты кадров запущенных МП, коснувшись поля «Изображение частоты кадров» в подразделе «Инструменты», и на отобразившейся странице коснуться пункта «Простое» или «Подробное» либо коснуться поля «Отключено» для отключения диагностики (Рисунок 77);

– разрешить либо запретить сохранение отладочных символов в домашнем каталоге, коснувшись переключателя «Сохранять отладочные символы в домашний каталог» в подразделе «Инструменты» (см. Рисунок 76);

– включить либо отключить валидацию пакетов, коснувшись переключателя «Включить валидацию пакетов» в подразделе «Инструменты» (см. Рисунок 76);

– разрешить либо запретить тихую установку пакетов из SDK, коснувшись переключателя «Разрешить тихую установку» в подразделе «Установка ПО» (см. Рисунок 76) и подтвердить действие вводом текущего кода безопасности.

4. МЕХАНИЗМЫ БЕЗОПАСНОСТИ

Реализованные в ОС Аврора функции безопасности можно настроить с помощью соответствующих интерфейсов, описание которых приведено в настоящем разделе.

4.1. Регистрация событий безопасности (аудит)

4.1.1. Системное журналирование

4.1.1.1. Общая информация

Системное журналирование – сервис для регистрации и хранения информации о важных программных и аппаратных событиях, а также их дальнейшего анализа в случае некорректной работы системы.

ОС Аврора осуществляет регистрацию и хранение следующей информации:

– сообщений из системного журнала (`syslog`) и ядра (`kernel log`);

– сообщений, выводимых процессами служб на стандартные потоки вывода (`stdout`);

– ошибок (`stderr`).

Полученная информация индексируется и хранится в системном журнале, который находится во временном каталоге и после перезагрузки не сохраняется.

4.1.1.2. Сохранение событий системы в постоянную память МУ

Для сохранения событий системы в постоянную внутреннюю память МУ необходимо выполнить следующие действия:

– открыть МП «Terminal» и выполнить команду:

```
vi /etc/systemd/journald.conf;
```

– установить параметры в следующие значения:

```
Storage=persistent;
SystemMaxUse=500M;
RuntimeMaxUse=1M;
```

– перезагрузить МУ для сохранения изменений.

Для выгрузки файла журнала необходимо выполнить команду: `journalctl -a > j.log`, при этом потребуется создать файл лога, который будет иметь название: `j.log`, которое при необходимости можно изменить, и содержать все события (`-a = all`).

4.1.2. Сервис sdjd

4.1.2.1. Общая информация

Для надежного хранения набора сообщений, относящихся к системе защиты информации, используется сервис по сбору и регистрации событий безопасности `sdjd`. Сервис сохраняет отдельные сообщения в файл `/var/log/sdjd-v2.log`, который при заполнении до максимального размера (50 МБ) перезаписывает предыдущие сообщения новыми.

Для определения событий регистрации сервисом sdjd используется конфигурационный файл `/etc/omp/sdjd.conf`, в котором с помощью редактирования можно задать неактуальным событиям статус false.

В ОС Аврора с использованием сервиса `sdjd` регистрируются и долговременно хранятся следующие типы событий безопасности:

– результат попытки входа в систему;
– блокирование интерактивного сеанса как по запросу пользователя, так и по истечении установленного периода неактивности пользователя;
– блокирование доступа после установленного количества неуспешных попыток ввода аутентификационной информации (п. 1.4.1);
– истечение срока действия кода безопасности;
– смена кода безопасности;
– запуск процедуры и результат проверки контроля целостности (КЦ);
– получение отрицательного результата проверки;
– автоматическая блокировка МУ;
– результат попытки установки, удаления или обновления RPM-пакетов;
– подключение и отключение внешних носителей информации;
– переполнение журнала событий безопасности;
– включение, перезагрузка и выключение МУ;
– добавление правил сетевого фильтра;
– запуск, завершение и изменение конфигурации службы аудита;
– изменение системного времени;
– нештатное завершение системы;
– попытки доступа к файлам, находящимся в процессе регистрации;
– сбой в механизме изоляции процессов;
– ошибки валидации RPM-пакетов;
– создание, переключение и удаление учетной записи пользователя;
– инициализация и результат прохождения 2ФА;
– события антивируса;
– события Доверенной среды исполнения Аврора (при наличии);
– запуск МП «Журнал»;
– обнаружение нарушения целостности сторонних файлов;

– изменение парольной и пользовательской политик;
– включение режима разработчика;
– включение и выключение удаленного доступа;
– разрешение и запрет доступа к МП «Terminal».

ПРИМЕЧАНИЕ. Полный список событий безопасности, регистрируемых сервисом `sdjd` в ОС Аврора, доступен на веб-сайте: https://community.omprussia.ru/documentation/4.0/software_development/reference/sdjd/events.html.

Каждое событие содержит следующую информацию:
– уникальный идентификатор события;
– тип события;
– время регистрации события;
– уровень важности сообщения;
– опциональный текст сообщения (или пустая строка);
– PID процесса-отправителя;
– PID родительского процесса для процесса-отправителя;
– UID процесса-отправителя;
– Effective UID процесса-отправителя;
– Saved UID процесса-отправителя;
– File system UID процесса-отправителя;
– Real GID процесса-отправителя;
– Effective GID процесса-отправителя;
– Saved GID процесса-отправителя;
– File system GID процесса-отправителя;
– Supplementary groups процесса-отправителя;
– эффективные привилегии процесса-отправителя;
– полный путь к исполняемому файлу процесса-отправителя;
– контекст безопасности процесса-отправителя (текущая роль или метка SELinux);
– текстовое представление идентификатора события для удобства отладки.

Администратор может определить список объектов ФС, попытки доступа к которым будут регистрироваться в файле `/usr/share/security-audit/security-audit-rules.conf`, добавив правило аудита для наблюдаемого объекта отдельной строкой в файл `/usr/share/security-audit/security-audit-rules.conf`. Попытки доступа к объектам ФС, находящимся в каталоге `/home` и его подкаталогах, не регистрируются.

Все регистрируемые события безопасности отображаются с указанием времени и цветовой индикацией в журнале событий, просмотр которого осуществляется с помощью МП «Журнал».

ПРИМЕЧАНИЕ. Пользователь имеет возможность сохранить журнал событий безопасности системы в постоянную внутреннюю память МУ в шифрованном виде.

4.1.2.2. Просмотр сообщений аудита

Рисунок 78

Для просмотра сообщений аудита и доступа к ним необходимо использовать следующие инструменты:

– программы `journalctl` и `dmesg`, имеющие интерфейс командной строки;

– утилиту `sdjd-dump`, позволяющую просмотреть события безопасности в МП «Terminal»;

– МП «Журнал» (`/usr/bin/log-viewer`), в графическом интерфейсе которого отображаются записи о событиях, сохраняемых сервисом `sdjd`.

ПРИМЕЧАНИЕ. В МП «Журнал» отображаются события аудита (Рисунок 78). Подробное описание работы МП приведено в документе «Руководство пользователя».

4.1.3. Сервис reports

Сервис reports генерирует единый системный отчет в виде архива с файлами конфигурации и логами работ различных компонентов ОС Аврора, который может быть зашифрован.

4.1.3.1. Описание системного отчета

Системный отчет имеет следующий формат: `tar.bz`.

Имя системного отчета имеет следующую маску: `reports-YYYYMMDD-HHMMSS.tar.bz`.

Подробное описание процедуры формирования отчета приведено в документе «Руководство пользователя».

Описание текущего набора модулей приведено в таблице (Таблица 1).

Таблица 1

Модуль	Описание
`agnss`	Информация о A-GNSS
`available-packages`	Список доступных и установленных пакетов
`battery`	Информация об аккумуляторе
`bluetooth`	Информация о Bluetooth®

Модуль	Описание
cellular	Информация SIM-картах
certificates	Информация об установленных в систему сертификатов (ima, kernel, rpm DB)
disks	Информация о дисках и занятом/свободном пространстве
dmesg	Системные сообщения ядра
emm	Информация о папке /etc/emm
info	Общая информация о МУ: – контрольная сумма загрузочного раздела; – информация об AIDE; – список дополнительных функций; – информация о релизе ОС; – информация о HW релизе ОС; – версия ядра
integrityd	Конфигурационные файлы сервиса integrityd
internet	Информация об активном интернет-соединении
logcat	Системные сообщения
memory	Информация о памяти
mount-points	Текущие точки монтирования
policy	Информация о папке /etc/policy
process-list	Текущий список процессов
push-daemon	Информация о папках /etc/xdg/push-daemon и /var/lib/push-daemon
rpms	Список установленных пакетов
screen	Информация об экране МУ
sdjd	События безопасности
ssu	Информация о репозиториях
system-journal-json	Системный журнал в формате JSON
system-journal-log	Системный журнал в текстовом виде
system	Краткая информация о системе: – режим экономии аккумулятора; – подключение к сети Интернет; – статус подключения; – активность сети WLAN
systemboot	Информация о системной загрузке
systemctl	Список сервисов
systemupdate	Лог обновления

4.1.3.2. Генерация системного отчета

4.1.3.2.1. Генерация системного отчета с помощью МП «Terminal»

Системный отчет можно сгенерировать в МП «Terminal» с помощью следующих команд.

– генерация системного отчета с сохранением в папке `/var/reports`:

```
generate-reports.sh true
```

– генерация системного отчета с сохранением во временной папке `/run/reports`, которая будет очищена после перезагрузки МУ:

```
generate-reports.sh false
```

– генерация зашифрованного системного отчета в указанной папке:

```
generate-encrypted-reports.sh /home/defaultuser/Documents/
```

ПРИМЕЧАНИЕ. При неуспешной генерации зашифрованного отчета будет создан обычный системный отчет.

4.1.3.2.2. Генерация системного отчета с помощью МП «Журнал»

Системный отчет можно сгенерировать в МП «Журнал». Подробное описание генерации системного отчета с помощью МП «Журнал» приведено в документе «Руководство пользователя».

4.1.3.2.3. Зашифрованный системный отчет

Шифрование системного отчета происходит только с установленным на МУ клиентским сертификатом. При отсутствии данного сертификата, вне зависимости от версии ОС Аврора, системный отчет не будет зашифрован.

В ОС Аврора используется блочное шифрование GOST 28147-89.

Шифрованный отчет имеет следующий формат: `tar.bz.cms`.

Имя шифрованного отчета имеет следующую маску: `reports-YYYYMMDD-HHMMSS.tar.bz.cms`.

Дополнительно формируется файл с информацией о сертификате, публичный ключ которого был использован для шифрования отчета. Он создается рядом и имеет формат txt: `reports-YYYYMMDD-HHMMSS.tar.bz.txt`. Время создания в названии у этих файлов идентичное.

Пример файла с информацией:

```
cat reports-20220329-104949.tar.bz2.txt
```

Вывод команды:

```
Subject: OMP Test
Group: developer
Subgroup: regular
Key ID:
7003efe7156bd53a2e88c2cb3c0d43e9786760c53721933dd5052fdaf443dbfb
```

Отчет можно расшифровать соответствующим ключом и сертификатом с помощью OpenSSL. Ключ расшифровки определяется в поле «Key ID» файла формата .txt, расположенного с архивом. Также потребуется подключить гостовый движок[2], дополнительно устанавливаемый в систему:

```
openssl cms -decrypt -in /home/defaultuser/Documents/reports-20220228-173753.tar.bz2.cms -recip system-developer-cert.crt  -inkey system-developer-key.pem -inform DER > reports-20220228-173753.tar.bz2
```

4.2. Идентификация и аутентификация

4.2.1. Общая информация

В ОС Аврора реализована подсистема идентификации и аутентификации, предназначенная для обеспечения однозначной и непротиворечивой:

– идентификации объектов доступа ОС Аврора (файлов и папок);

– идентификации субъектов доступа ОС Аврора (системных процессов, процессов в пространстве ядра, а также процессов или программ, запущенных от имени пользователей);

– идентификации учетных записей пользователей ОС Аврора (с ролью администратора и с ролью пользователя);

– аутентификации пользователей ОС Аврора, как с использованием аутентификации с помощью пароля (характеристики пароля и значения политики паролей, такие как сложность алфавита, длина, сроки действия и т.п. могут быть изменены администратором), так и с использованием 2ФА (пп. 1.4.3.2).

Для работы с ОС Аврора администратору присваиваются следующие идентификаторы:

– символьный: defaultuser;

– числовой: 100000.

Для выполнения функций администрирования используются интерфейсы функциональных возможностей безопасности (ИФБО) ОС Аврора, которые представляют собой конфигурационные файлы либо команды оболочки.

В ОС Аврора для исполняемых файлов применяется формат, позволяющий установить режим доступа к сегментам в адресном пространстве процесса. С помощью `seccomp-bpf` можно запретить некоторые системные вызовы, например: `mount/umount`, `ptrace`, `kexec` и др.

Для усиления защиты МУ при первом включении рекомендуется установить и периодически изменять код безопасности, который:

– в случае шифрования раздела с домашними директориями (п. 4.2.2) пользователей будет храниться в LUKS-слоте, соответствующем идентификатору пользователя;

[2]Реализация криптоалгоритмов российского ГОСТ для OpenSSL.

– в иных случаях будет храниться в виде свертки, полученной с помощью алгоритма криптографического преобразования `sha1`.

В целях предотвращения несанкционированного доступа к МУ, функционирующему под управлением ОС Аврора, код безопасности (см. п. 1.4.1) требуется для подтверждения выполнения следующих действий:

– создания учетных записей ролей;
– настройки парольной политики;
– задания ограничений входа в систему;
– включения и настройки 2ФА;
– задания одноразового пароля;
– изменения настроек блокировки;
– разрешения установки стороннего ПО;
– установки SSH-пароля;
– активации и настройки режима разработчика;
– просмотра данных учетных записей;
– сброса настроек МУ.

ПРИМЕЧАНИЕ. Подробная информация о задании кода безопасности приведена в документе «Руководство пользователя».

4.2.2. Шифрование раздела с домашними директориями

В ОС Аврора раздел с домашними директориями шифруется с помощью алгоритма aes-xts-plain64 512-битным мастер-ключом, для хранения которого используется LUKS-заголовок, состоящий из следующих 8 слотов:

– 1 слот используется учетной записью администратора;
– 6 слотов доступны создаваемым учетным записям пользователя;
– 1 слот резервируется для смены паролей.

Идентификатор учетной записи пользователя определяет номер используемого слота, при этом в каждом из слотов хранится мастер-ключ, шифрованный паролем соответствующей учетной записи пользователя с помощью алгоритма PBKDF. Количество итераций алгоритма подбирается таким образом, чтобы проверка 1 комбинации выполнялась приблизительно 1 секунду.

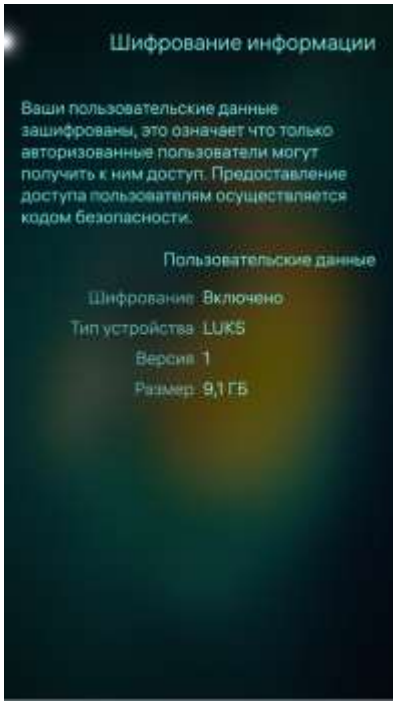

Рисунок 79

Выполнение следующих действий требует ввода корректного кода безопасности из слота LUKS-заголовка, соответствующего идентификатору выбранной учетной записи пользователя:

– разблокировка домашней директории при загрузке МУ;

– разблокировка UI Lipstick.

Для отображения страницы «Шифрование информации» (Рисунок 79) необходимо выполнить следующие действия:

– открыть меню настроек системы касанием значка на Экране приложений;

– в подразделе «Безопасность» коснуться пункта меню «Шифрование» .

4.3. Управление доступом

В ОС Аврора реализованы следующие механизмы управления доступом:

– дискреционное разграничение прав доступа (DAC), (п. 4.3.1-4.3.7) основанное на следующих сущностях:

- биты разрешений;
- списки контроля доступа (POSIX ACL) и расширенные атрибуты стандарта POSIX 1e;
- возможности и/или перечни возможностей – `capabilities`;

– долевое разграничение доступа предусматривает активацию политик безопасности (п. 4.3.7), которые:

- доступны для управления только администратору;
- предназначены для управления функциями ОС Аврора.

Политику дискреционного разграничения прав доступа (п. 4.3.1-4.3.7) рекомендуется применять при необходимости:

– разрешить или запретить доступ учетной записи пользователя к объекту ФС (МП, файл МУ, служба и т. п.);

– ограничить для какого-либо процесса возможности выполнять определенные действия.

ПРИМЕЧАНИЕ. Настройка политики дискреционного разграничения прав доступа действует только в отношении ФС EXT4, которая используется в ОС Аврора по умолчанию. Стандартный набор прав в виде битовой маски поддерживается ОС Аврора для служебных ФС: `tmpfs`, `sysfs`, `procfs`.

Принцип работы политики дискреционного разграничения прав доступа представлен на рисунке (Рисунок 80).

Рисунок 80

Дискреционное разграничение прав доступа осуществляется каждый раз при попытке обращения субъекта к объекту, при этом обе сущности содержат соответствующие атрибуты безопасности, принадлежащие дискреционной модели. Ядро ОС Аврора проверяет уровень доступа, базирующийся на данных атрибутах с одной стороны и правилами разграничения доступа (ПРД) с другой стороны.

Дискреционное разграничение прав доступа позволяет владельцу объекта определять, кто обладает полномочиями осуществлять доступ к объекту.

Субъектами в реализации дискреционного разграничения прав доступа выступают обычные процессы ОС Аврора, которые представлены программной структурой ядра `task_struct`.

Все нижеименованные сущности, такие как обычные файлы, файлы МУ (как блочные, так и символьные), директории, файлы сокетов (пайпов) с точки зрения дискреционного разграничения прав доступа ядра ОС Аврора являются объектами доступа, следовательно, также попадают под действие политик разграничения.

Атрибуты субъектов доступа используются для осуществления решений политики дискреционного разграничения прав доступа. В качестве данных атрибутов для субъектов доступа (процессов) обязательно применяются следующие:

– эффективный идентификатор (число) владельца процесса (PID);
– эффективный идентификатор (число) группы процесса (GID);
– метка возможности или возможность (`capability`) стандарта POSIX.1e;
– членство процесса в дополнительных (не основных) группах (при необходимости).

Указанные атрибуты субъекта доступа хранятся в программной структуре ядра `task_structc` и оцениваются системными вызовами.

Атрибуты объектов доступа также используются для осуществления решений политики дискреционного разграничения прав доступа. В качестве данных атрибутов объекта доступа применяются следующие:

– владелец объекта;

– владелец группы объекта;

– биты прав доступа;

– дополнительные атрибуты списков контроля доступа в соответствии со стандартом POSIX.1e ACLs.

Атрибуты объектов доступа сохраняются и учитываются ядром ОС Аврора, а также хранятся в индексных дескрипторах ФС, размещенной на дисковом пространстве МУ.

Правила дискреционного разграничения прав доступа определяют:

– каким образом конкретный процесс (субъект) может осуществлять попытку получения доступа к объекту, основываясь на указанных выше атрибутах безопасности;

– каким образом и при каких условиях атрибуты безопасности субъекта и объекта принимают новые значения.

4.3.1. Общее описание привилегий

Привилегия программы базируется на следующих значениях атрибутов процесса:

– атрибут эффективного UID (PID);

– атрибут эффективной группы процесса (GID);

– набор (список) членства дополнительных групп процесса (при наличии), которые субъект доступа (процесс) имеет в любое время жизненного цикла.

Любой процесс с эффективным идентификатором 0 считается привилегированным и обладает возможностью не применять политику дискреционного разграничения прав доступа в ОС Аврора.

Учетная запись суперпользователя ассоциирована с идентификатором 0. Однако технически ядро ОС Аврора определяет учетные записи пользователей по числовым идентификаторам, следовательно, технически возможно иметь учетную запись пользователя с идентификатором 0, обладающего именем, отличным от суперпользователя.

Кроме того, ядро ОС Аврора имеет соответствующий механизм оценки прав, базирующийся на требованиях открытого стандарта POSIX1.e ACL для списков контроля доступа и возможностей (`capabilities`), который также позволяет преодолевать ограничения политики дискреционного разграничения прав доступа для субъекта с атрибутом эффективного идентификатора 0.

Поскольку для выполнения проверки прав доступа процессы разделяют на две категории: привилегированные (ID эффективного пользователя равен 0, как у

суперпользователя) и непривилегированные (ID эффективного пользователя не равен 0), для привилегированных процессов выполняются не все проверки прав в ядре, а для непривилегированных процессов выполняется полная проверка на основе возможностей процесса (обычно эффективного UID, эффективного GID и списка дополнительных групп).

В ядре ОС Аврора все привилегии, связанные с суперпользователем, разделены на несколько частей, называемых возможностями (`capabilities`), которые можно разрешать и запрещать независимо друг от друга. Возможности также являются атрибутом нити.

Например, если какой-либо процесс пытается создать специальный файл МУ, используя для этого системный вызов `mknod` (), ядро ОС Аврора помимо проверки дискреционных прав выполняет следующие дополнительные проверки:

– равно ли 0 значение эффективного идентификатора процесса (субъекта доступа);

– разрешено ли процессу создавать специальные файлы с помощью совершения соответствующего системного вызова.

ОС Аврора обеспечивает для политики дискреционного разграничения прав доступа применение следующих свойств:

– принудительная реализация получения субъектами (процессами) доступа к объектам только в рамках полномочий, определяемых ПРД;

– наличие способности определять (назначать) политику доступа только для субъектов, имеющих на это права, как определено политикой полномочий для них;

– отсутствие любых ограничений для субъекта, имеющего идентификатор 0.

Атрибуты дискреционного разграничения прав доступа связываются непосредственно с объектом доступа в момент его создания. Действие этих атрибутов распространяется на объект до его уничтожения либо до тех пор, пока атрибуты не будут изменены.

Атрибуты существуют и действуют для любого типа объекта, хранятся в метаданных файлов и реализованы в виде следующих сущностей:

– биты разрешений (базовые атрибуты);

– списки контроля доступа (ACL и расширенные атрибуты);

– возможности (`capabilities`).

Субъект, идентификатор которого совпадает с идентификатором владельца объекта, может изменять атрибуты объекта, включая базовые и расширенные биты прав (за исключением случаев, когда ФС находится в режиме «только для чтения»).

ПРИМЕЧАНИЯ:

1. Возможность изменять владельца объекта доступна только суперпользователю;
2. Возможность изменять группу объекта ограничена только для владельца и для пользователя с идентификатором 0 (суперпользователь).

В случае необходимости присвоить (изменить) новый идентификатор группы для объекта, он должен совпадать с текущим значением основной группы субъекта или являться значением группы, членом которой является субъект.

Суперпользователь уполномочен определять любую группу для объекта, изменять владельца и любые биты прав, включая расширенные атрибуты и возможности.

Стандартный набор прав доступа реализован в виде битовой маски прав, налагаемых на каждый идентифицированный объект. Данный стандартизированный подход и стандартный механизм, реализуемый ОС Аврора, определены в текущей спецификации UNIX.03.

Индивидуальные биты прав используются для указания права на чтение (`r` или 4), запись (`w` или 2) или выполнение (`x` или 1). Они определяются отдельно для владельца объекта, группы, к которой принадлежит объект и всех остальных, т.е. субъектов, идентификаторы которых не совпадают ни с идентификатором владельца, ни с текущим идентификатором группы объекта.

4.3.2. ПО с повышенным уровнем привилегий

Программы, функционирующие в системе и использующие повышенный уровень привилегий, могут быть разделены на 3 типа:

– программы или системные службы, запущенные системным инициализатором или ядром. Например, программы `systemd` или `crond`;

– программы, запущенные от имени суперпользователя с целью выполнения функций администрирования или обслуживания. Например, запуск программы `iptables` для управления МЭ;

– программы, имеющие установленный эффективный бит смены идентификатора (`setuid`), если бит указывает на принадлежность пользователю с идентификатором 0.

Таким образом, все ПО, которое работает в системе, требует наличия повышенных привилегий и содержит или реализует те или иные функции безопасности, можно отнести к функциональным возможностям безопасности (ФБО) либо ИФБО ОС Аврора.

Остальное ПО, работающее в системе с повышенными привилегиями либо без наличия таковых, при этом не реализующее никаких функций безопасности, является не ФБО или ИФБО, а непривилегированным ПО.

В корректно обслуживаемой системе любое непривилегированное ПО находится под действием ФБО ОС Аврора и не имеет возможности выйти из-под его действия. Однако при этом предполагается, что любое непривилегированное ПО не является доверенным.

Привилегированное ПО, которое не содержит ФБО (например, внешние модули уровня ядра, реализующие определенные функции, к примеру поддержку специализированного оборудования), должно быть надежно изолировано от использования с помощью ФБО ОС Аврора для предотвращения доступа к нему в обход требуемых полномочий.

4.3.3. Стандартные биты прав доступа

В ОС Аврора реализованы 3 набора по 3 бита, которые определяют политику доступа к объекту для 3 категорий пользователей (субъектов):

– пользователь-владелец объекта;

– пользователи, входящие в группу, к которой принадлежит объект;

– все остальные пользователи (не входящие в указанное множество).

3 бита в каждом наборе определяют права доступа для каждого пользователя, состоящего в 1 из указанных выше категорий. При этом:

– 1 бит отводится для чтения (`r` или 4), которое рассматривается как поток данных от объекта (файла, каталога) к субъекту (процесс, пользователь);

– 1 бит отводится для записи или удаления (`w` или 2), которое рассматривается как поток данных от субъекта (процесс, пользователь) к объекту (файл, каталог);

– 1 бит отводится для исполнения (`x` или 1), которое рассматривается как процесс загрузки в оперативную память текста программы и следование инструкциям функций программы, поэтому каждый субъект получает доступ к каждому объекту, базируясь на комбинации этих битов.

Например:

– `rwx` – определяет права на совершение всех операций (чтение, запись и выполнение);

– `r-x` – определяет права только на чтение и выполнение;

– `r` – определяет право только на чтение;

– `---` – определяет отсутствие любых прав доступа.

При попытке доступа субъекта к объекту решение о предоставлении доступа или отказе в доступе базируется на следующем алгоритме:

– является ли идентификатор пользователя (субъекта) равным 0. Если является, принимается решение о гарантировании любого доступа на чтение/запись, игнорируя биты прав. Доступ на выполнение также гарантируется для этого идентификатора субъекта, если бит исполнения стоит хотя бы для какого-нибудь субъекта. Дальнейшие проверки производиться не будут;

– если идентификатор субъекта доступа совпадает с указанным идентификатором владельца объекта, а биты разрешений установлены в определенные значения, решение о доступе или отказе в доступе будет принято в соответствии со значениями битов. Дальнейшие проверки производиться не будут;

– если идентификатор одной из групп субъекта доступа (первичной или любой другой) совпадает с идентификатором группы объекта доступа, а биты прав (разрешений) для группы установлены в какое-либо значение, решение о доступе или отказе в доступе будет принято в соответствии со значениями битов. Дальнейшие проверки производиться не будут;

– если идентификатор субъекта доступа или идентификатор любой из групп, в которой состоит субъект доступа, не совпадает со значениями атрибутов владельца и

группы для объекта доступа, оцениваются значения битов прав для объекта согласно категории иных пользователей. Если битовая маска прав для категории иных пользователей установлена в какое-либо значение, решение о доступе принимается согласно установленным в ней значениям и субъекту предоставляется доступ к объекту.

ПРИМЕЧАНИЕ. Если никакое из описанных условий не выполнено и идентификатор субъекта доступа не равен 0, попытка доступа блокируется.

4.3.4. Расширенные атрибуты и списки контроля доступа

ОС Аврора поддерживает расширенный механизм (атрибуты) объектов, а также списки контроля доступа в соответствии с требованиями стандарта POSIX для ФС EXT4. Данный механизм позволяет системе предоставлять наиболее тонко структурируемый доступ субъектов к объектам.

К перечню расширенных атрибутов относятся следующие:

– `A – (no atime updates)` время получения доступа к файлу не будет обновляться, что благоприятно повлияет на производительность ФС в случае слишком частых обращений к файлу;

– `a – (append only)` в файл можно только дописывать, но нельзя удалять/переименовывать (удобно для сообщений о событиях). Если атрибут установлен на каталог, то находящиеся в нем файлы нельзя удалять, можно только создавать новые и модифицировать существующие;

– `c – (compressed)` ядро производит прозрачное сжатие информации файла на диске, а при доступе возвращаются несжатые данные;

– `D – (synchronous directory updates)` при модификации директории изменения синхронно записываются на диск;

– `d – (no dump)` игнорировать при создании резервной копии программой `dump`;

– `i – (immutable)` бит, запрещающий любые изменения файла (удалять, переименовывать и модифицировать). Если атрибут установлен на директорию, то находящиеся в ней файлы можно модифицировать, при этом удалять или создавать новые файлы невозможно;

– `j – (data journalling)` если ФС смонтирована с параметрами `data=ordered` или `data=writeback`, данные файла с этим атрибутом сохраняются сначала в журнал ФС, и лишь затем в файл. Атрибут устанавливается и снимается только суперпользователем и не действует при монтировании с параметром `data=journal` (т.к. в этом случае данные также сохраняются сначала в журнал и атрибут не имеет смысла);

– `s – (secure deletion)` полное удаление файла (место на диске, где он находился, заполняется нулями);

– `S – (synchronous updates)` прямая запись на диск без кэширования (обновление в файле происходит на диске синхронно с МП, изменяющим данный файл);

– u – (undeletable) при удалении файла с таким атрибутом его содержимое сохраняется, что позволяет успешно использовать инструменты для восстановления удаленных файлов;

– T – каталог с таким атрибутом считается расположенным на вершине иерархии директорий с целью использования метода распределения блоков Орлова;

– t – в конец файла с таким атрибутом невозможно присоединить другой файл (tail-merging). На момент написания ext2 и ext3 не поддерживали tail-merging (не считая экспериментальных патчей);

– E – указывает на наличие ошибок при сжатии файла. Невозможно установить/снять с помощью chattr, можно лишь просмотреть командой lsattr;

– e – указывает, что файл использует дополнения для размещения блоков на диске. Невозможно установить/снять с помощью chattr, можно лишь просмотреть командой lsattr;

– I – указывает, что каталог был проиндексирован при использовании htree. Невозможно установить/снять с помощью chattr, можно лишь просмотреть командой lsattr;

– H – указывает, что файл хранит свои блоки в единицах ФС, а не в единицах секторов, это означает, что файл имеет размер более 2TB (или когда-то занимал). Невозможно установить/снять с помощью chattr, можно лишь просмотреть командой lsattr;

– X – указывает на наличие возможности получить прямой непосредственный доступ к сжатому файлу. Невозможно установить/снять с помощью chattr, можно лишь просмотреть командой lsattr;

– Z – указывает, что сжатый файл is dirty. Нельзя установить/снять с помощью chattr, можно лишь просмотреть командой lsattr.

Просмотр атрибутов осуществляется с помощью ИФБО lsaatr, а установка или модификация осуществляется с помощью ИФБО chattr. При этом атрибуты для объекта сохраняются в метаданных объекта (i-node) ФС EXT4.

Расширенные атрибуты ACL содержат следующую информацию:

– признак (метка) типа значения списка контроля доступа;

– уточняющее значение (уточняет предыдущий признак);

– набор прав, который определяет дискреционные права для субъекта, используя сначала признак типа, а затем его уточняющее значение.

Существуют следующие признаки:

– ACL_GROUP – определяет возможность доступа для субъекта, у которого идентификатор группы (или любой из идентификаторов группы субъекта) совпадает со значением, указанным для группы в списке контроля доступа объекта в качестве уточняющего значения;

– ACL_GROUP_OBJ – определяет возможность доступа для субъекта, у которого идентификатор группы (или любой из идентификаторов группы субъекта) совпадает со значением идентификатора группы объекта;

– ACL_MASK – определяет максимально возможное значение битов прав для группы или групп;

– ACL_OTHER – определяет права доступа для субъектов, которые не могут быть соотнесены с любым значением идентификаторов, указанных в списке контроля доступа;

– ACL_USER – определяет права для субъекта, идентификатор которого указан как уточняющее значение владельца объекта;

– ACL_USER_OBJ – определяет права доступа для объекта, идентификатор которого совпадает со значением владельца объекта.

Уточняющее значение – это значение, требуемое признаком ACL_GROUP и ACL_USER. Оно может содержать идентификатор пользователя или группы, для которых будет приниматься решение о предоставлении доступа в соответствии с битами прав.

Права доступа для списков контроля доступа ACL – существуют как биты значений прав, означающих чтение (r), запись/удаление (w) или выполнение/поиск (x).

Отношения, возникающие при принятии решений в момент сравнения базовых прав (битовой маски) и расширенных прав (списков контроля доступа), состоят в следующем:

– список контроля доступа ACL считается обязательным или минимально необходимым для типов ACL_USER_OBJ, ACL_GROUP_OBJ и ACL_OTHER;

– типы ACL_GROUP или ACL_USER считаются расширенными и оцениваются при наличии;

– по умолчанию (если не указано иное) тип ACL_USER_OBJ, ACL_GROUP_OBJ и ACL_OTHER принимают значения, совпадающие с базовым значением битовой маски владельца и группы объекта;

– если указаны уточняющие значения для типов ACL_GROUP и ACL_USER, как минимум одно значение типа ACL_MASK должно быть указано. Иначе тип ACL_MASK также считается необязательным, и его значение может быть не указано.

Проверка расширенных прав при определении попытки доступа субъекта к объекту, на котором установлены расширенные атрибуты ACL, происходит по следующему алгоритму:

1) ЕСЛИ: идентификатор субъекта доступа совпадает с идентификатором владельца объекта доступа;

ТОГДА: оцениваются значения запрошенных прав, установленных в объеме для типа ACL_USER_OBJ, и субъект получает доступ к объекту в объеме указанных прав;

ИНАЧЕ: доступ блокируется;

2) ЕСЛИ: идентификатор субъекта доступа совпадает с любым указанным идентификатором владельца объекта доступа, упомянутым для типа ACL_USER;

ТОГДА: оцениваются значения запрошенных прав, установленные в объеме сначала для типа ACL_USER, а затем для типа ACL_MASK, и субъект получает доступ к объекту в объеме указанных прав;

ИНАЧЕ: доступ блокируется;

3) ЕСЛИ: основной или добавочный идентификатор группы субъекта доступа совпадает с любым указанным идентификатором группы объекта доступа, упомянутым для типа сначала ACL_GROUP_OBJ, а затем ACL_GROUP;

ТОГДА: оцениваются права для признака типа ACL_MASK,

И ЕСЛИ: права доступа содержат запрошенные значения для любого из типов ACL_GROUP_OBJ, ACL_GROUP, ACL_MASK;

ТОГДА: доступ предоставляется;

ИНАЧЕ: доступ блокируется.

4.3.5. Механизм разрешений наборов возможностей

ОПИСАНИЕ: для выполнения проверки прав доступа субъекты доступа разделяют на 2 категории: привилегированные (ID эффективного пользователя равен 0, как у суперпользователя), и непривилегированные (ID эффективного пользователя не равен 0).

СПИСОК РАЗРЕШЕНИЙ: возможности, реализованные в составе ОС Аврора, а также операции или поведение, разрешаемые данными возможностями:

– CAP_AUDIT_CONTROL – позволяет включать или выключать аудит ядра, изменять фильтрующие правила аудита, получать состояние аудита и фильтрующие правила;

– CAP_AUDIT_WRITE – позволяет записывать данные в журнал аудита ядра;

– CAP_BLOCK_SUSPEND – позволяет использовать возможности, способные приводить к блокированию приостановки системы (epoll(7) EPOLLWAKEUP, /proc/sys/wake_lock);

– CAP_CHOWN – позволяет выполнять произвольные изменения файловых UID и GID;

– CAP_DAC_OVERRIDE – позволяет пропускать проверки доступа к файлу на чтение, запись и выполнение;

– CAP_DAC_READ_SEARCH – позволяет пропускать проверки доступа к файлу на чтение и доступа к каталогу на чтение и выполнение; вызывать функцию open_by_handle_at();

– CAP_FOWNER – позволяет выполнять следующие действия:

- пропускать проверки доступа для операций, которые обычно требуют совпадения UID ФС процесса и UID файла (например, chmod, utime), исключая операции, охватываемые CAP_DAC_OVERRIDE и CAP_DAC_READ_SEARCH;
- устанавливать расширенные атрибуты произвольных файлов;
- устанавливать списки контроля доступа (ACL) произвольных файлов;
- игнорировать закрепляющий бит при удалении файла;
- задавать O_NOATIME для произвольных файлов в open и fcntl;

– CAP_FSETID – позволяет не очищать биты режима set-user-ID и set-group-ID при изменении файла, а также устанавливать бит set-group-ID на файл, у которого

GID не совпадает с битом ФС или любыми дополнительными GID вызывающего процесса;

– CAP_IPC_LOCK – позволяет блокировать память (mlock(2), mlockall(2), mmap(2), shmctl(2));

– CAP_IPC_OWNER – позволяет не выполнять проверки доступа для операций с объектами System V IPC;

– CAP_KILL – позволяет не выполнять проверки при отправке сигналов. К данной возможности относится использование ioctl(2) с операцией KDSIGACCEPT;

– CAP_LEASE – позволяет устанавливать аренду на произвольные файлы;

– CAP_LINUX_IMMUTABLE – позволяет устанавливать inode-флаги FS_APPEND_FL и FS_IMMUTABLE_FL;

– CAP_MKNOD – позволяет создавать специальные файлы с помощью mknod(2);

– CAP_NET_ADMIN – позволяет выполнять следующие сетевые операции:

- настройка интерфейса;
- управление IP МЭ, трансляцией адресов и ведением учета;
- изменение таблицы маршрутизации;
- привязка к любому адресу для прозрачного проксирования;
- назначение типа сервиса;
- очистка статистики драйвера;
- включение режима захвата (promiscuous);
- включение многоадресных рассылок (multicasting);
- использование setsockopt(2) для включения следующих параметров сокета: SO_DEBUG, SO_MARK, SO_PRIORITY (для приоритетов вне диапазона 0-6), SO_RCVBUFFORCE и SO_SNDBUFFORCE;

– CAP_NET_BIND_SERVICE – позволяет привязывать сокет к привилегированным портам домена сети Интернет (номера портов меньше 1024);

– CAP_NET_BROADCAST – (не используется) позволяет осуществлять широковещание с сокета и прослушивание многоадресных рассылок;

– CAP_NET_RAW позволяет выполнять следующие действия:

- использовать сокеты RAW и PACKET;
- привязываться к любому адресу для прозрачного проксирования;

– CAP_SETGID – позволяет выполнять произвольные действия с GID процесса и списком дополнительных GID, а также подделывать GID при передаче возможностей сокета через доменные сокеты UNIX; записывать отображение ID группы в пользовательское пространство имен;

– CAP_SETFCAP – позволяет назначать файловые возможности, при этом:

- если файловые возможности не поддерживаются: предоставлять и отзывать любую возможность в списке разрешенных возможностей вызывающего или любого другого процесса (данное свойство CAP_SETPCAP недоступно, если ядро

собрано с поддержкой файловых возможностей, т.к. CAP_SETPCAP имеет полностью другую семантику у таких ядер);

- если файловые возможности поддерживаются: добавлять любую возможность из ограничивающего набора вызывающей нити в ее наследуемый набор; отзывать возможности из ограничивающего набора (с помощью prctl(2) с операцией PR_CAPBSET_DROP); изменять флаги securebits;

– CAP_SETUID – позволяет выполнять произвольные действия с UID процесса (setuid(2), setreuid(2), setresuid(2), setfsuid(2)); подделывать UID при передаче возможностей сокета через доменные сокеты UNIX; записывать отображение ID пользователя в пользовательское пространство имен;

– CAP_SYS_ADMIN позволяет выполнять следующие действия:

- решать задачи управления системой: quotactl(2), mount(2), umount(2), swapon(2), swapoff(2), sethostname(2) и setdomainname(2);
- выполнять привилегированные операции syslog(2) (для данных операций необходимо использовать CAP_SYSLOG);
- выполнять команду VM86_REQUEST_IRQ vm86(2);
- выполнять операции IPC_SET и IPC_RMID над произвольными объектами System V IPC;
- перезаписывать ограничения ресурса RLIMIT_NPROC;
- выполнять операции над расширенными атрибутами trusted и security;
- использовать lookup_dcookie(2);
- использовать ioprio_set(2) для назначения классов планирования ввода-вывода IOPRIO_CLASS_RT и IOPRIO_CLASS_IDLE;
- подделывать PID при передаче возможностей сокета через доменные сокеты UNIX;
- превышать /proc/sys/fs/file-max, системное ограничение на количество открытых файлов в системных вызовах, открывающих файлы (например, accept(2), execve(2), open(2), pipe(2));
- задействовать флаги CLONE_*, создающие новые пространства имен с помощью clone(2) и unshare(2) (для создания пользовательских пространств имен не требуется никаких иных возможностей);
- вызывать perf_event_open(2);
- получать доступ к информации о привилегированном событии perf;
- вызывать setns(2), требуется CAP_SYS_ADMIN в пространстве имен назначения;
- вызывать fanotify_init(2);
- вызывать bpf(2);
- выполнять операции KEYCTL_CHOWN и KEYCTL_SETPERM в keyctl(2);
- выполнять операцию MADV_HWPOISON в madvise(2);

- задействовать TIOCSTI в `ioctl(2)` для вставки символов во входную очередь терминала, отличного от управляющего терминала вызывающего;
- задействовать устаревший системный вызов `nfsservctl(2)`;
- задействовать устаревший системный вызов `bdflush(2)`;
- выполнять различные привилегированные операции `ioctl(2)` над блочными МУ;
- выполнять различные привилегированные операции `ioctl(2)` над ФС;
- выполнять административные операции над драйверами МУ;

– `CAP_SYS_BOOT` – позволяет использовать `reboot(2)` и `kexec_load(2)`;

– `CAP_SYS_CHROOT` – позволяет использовать `chroot(2)`;

– `CAP_SYS_MODULE` – позволяет загружать и выгружать модули ядра;

– `CAP_SYS_NICE` позволяет выполнять следующие действия:

- повышать значение уступчивости процесса (`nice(2)`, `setpriority(2)`) и изменять значение уступчивости у произвольных процессов;
- назначать политики планирования реального времени для вызывающего процесса, а также политики планирования и приоритеты для произвольных процессов (`sched_setscheduler(2)`, `sched_setparam(2)`, `shed_setattr(2)`);
- выполнять привязку к электронной подписи (ЭП) для произвольных процессов (`sched_setaffinity(2)`);
- назначать класс планирования ввода-вывода и приоритет для произвольных процессов (`ioprio_set(2)`);
- применять `migrate_pages(2)` к произвольным процессам для их перемещения на произвольные узлы;
- применять `move_pages(2)` к произвольным процессам;
- использовать флаг `MPOL_MF_MOVE_ALL` в `mbind(2)` и `move_pages (2)`;

– `CAP_SYS_PACCT` – позволяет использовать `acct`(2);

– `CAP_SYS_PTRACE` – позволяет выполнять следующие действия:

- трассировать любой процесс с помощью `ptrace`(2);
- применять `get_robust_list(2)` к произвольным процессам;
- перемещать данные в/из памяти произвольного процесса с помощью `process_vm_readv(2)` и `process_vm_writev(2)`;
- изучать процессы с помощью `kcmp(2)`;

– CAP_SYS_RAWIO – позволяет выполнять следующие действия:

- выполнять операции ввода-вывода из портов (`iopl(2) и ioperm(2)`);
- разрешать доступ к `/proc/kcore`;
- задействовать операцию FIBMAP в `ioctl(2)`;
- открывать МУ для доступа к специальным регистрам x86 (MSR);
- обновлять `/proc/sys/vm/mmap_min_addr`;
- создавать отображения памяти по адресам, меньше значения, заданного в `/proc/sys/vm/mmap_min_addr`;

- отображать файлы в /proc/bus/pci;
- открывать /dev/mem и /dev/kmem;
- выполнять различные команды МУ SCSI;
- выполнять определенные операции с МУ hpsa(4) и cciss(4);
- выполнять некоторые специальные операции с другими устройствами;

– CAP_SYS_RESOURCE – позволяет выполнять следующие действия:

- использовать зарезервированное пространство ФС ext2;
- совершать вызовы ioctl(2), управляющие журналированием ext3;
- превышать ограничение дискового пространства;
- увеличивать ограничения по ресурсам;
- перезаписывать ограничение ресурса RLIMIT_NPROC;
- превышать максимальное количество консолей при выделении консоли;
- превышать максимальное количество раскладок;
- использовать более, чем 64 ГЦ прерывания из часов реального времени;
- назначать значение msg_qbytes очереди сообщений System V больше ограничения /proc/sys/kernel/msgmnb;
- превышать ограничение /proc/sys/fs/pipe-size-max при назначении вместимости канала с помощью команды F_SETPIPE_SZ y fcntl(2);
- использовать F_SETPIPE_SZ для увеличения вместимости канала больше, чем ограничение, задаваемое в /proc/sys/fs/pipe-max-size;
- превышать ограничение /proc/sys/fs/mqueue/queues_max при создании очередей сообщений POSIX; задействовать операцию PR_SET_MM в prctl(2);
- устанавливать /proc/PID/oom_score_adj в значение меньшее, чем последнее установленное значением процессом с помощью CAP_SYS_RESOURCE;

– CAP_SYS_TIME – позволяет настраивать системные часы (settimeofday(2), stime(2), adjtimex(2)) и часы реального времени (аппаратные);

– CAP_SYS_TTY_CONFIG – позволяет использовать vhangup(2) и задействовать различные привилегированные операции ioctl(2) с виртуальными терминалами;

– CAP_SYSLOG – позволяет выполнять следующие действия:

- выполнять привилегированные операции syslog(2);
- просматривать адреса ядра, отображаемые в /proc и других интерфейсах, когда значение /proc/sys/kernel/kptr_restrict равно 1;

– CAP_WAKE_ALARM – позволяет вызывать что-либо при пробуждении системы (устанавливать таймеры CLOCK_REALTIME_ALARM и CLOCK_BOOTTIME_ALARM).

НАБОРЫ ВОЗМОЖНОСТЕЙ СУБЪЕКТА

Каждая нить имеет наборы возможностей, содержащие 0 и/или более следующих возможностей:

1) Разрешенные действия (Permitted). Permitted – ограничивающий набор эффективных возможностей, которыми наделяется нить. Данный набор также ограничивает список возможностей, которые могут быть добавлены в наследуемый

набор для нити, не имеющей возможностей CAP_SETPCAP в своем эффективном наборе. Если нить сбрасывает возможность в своем разрешительном наборе, она не сможет получить ее обратно (если только не выполняется execve(2) для программы с set-user-ID-root или программа, у которой соответствующие возможности файла предоставляют эту возможность);

2) Наследуемые действия (Inheritable). Inheritable – набор наследуемых возможностей, которые остаются таковыми при выполнении любой программы и сохраняются при вызове execve(2). Наследуемые возможности добавляются в разрешительный набор, если выполняющаяся программа имеет соответствующие установленные биты в файловом наследуемом наборе. Если выполнение происходит не от имени суперпользователя, наследуемые возможности не сохраняются после execve(2), поэтому МП, которым необходимо выполнять вспомогательные программы с повышенными возможностями, требуется использовать наружные возможности (ambient capabilities);

3) Эффективные действия (Effective). Effective – данный набор возможностей используется ядром при выполнении проверок прав нити.

С помощью capset(2) нить может изменять свои наборы возможностей.

Файл (ИФБО) /proc/sys/kernel/cap_last_cap содержит числовое значение самой большой возможности, поддерживаемой работающим ядром. Это можно использовать для определения наибольшего бита, который может быть установлен в наборе возможностей.

ФАЙЛОВЫЕ НАБОРЫ ВОЗМОЖНОСТЕЙ

В ОС Аврора связь наборов возможностей с исполняемым файлом поддерживается с помощью setcap(8). Наборы возможностей файла хранятся в расширенном атрибуте, для записи в который требуется возможность CAP_SETFCAP. Наборы файловых возможностей вместе с наборами возможностей нити определяют наборы возможностей нити после выполнения execve(2).

Существуют следующие файловые наборы возможностей:

– permitted (ранее называвшийся forced) – возможности автоматически разрешаются нити независимо от ее унаследованных возможностей;

– inheritable (ранее называвшийся allowed) – набор объединяется (AND) с унаследованным набором нити, чтобы определить, какие унаследованные возможности будут включены в ее разрешительный набор после execve(2);

– effective – в действительности представляет собой не набор, а одиночный бит. Если бит включен, то при вызове execve(2) все новые разрешенные возможности нити будут также добавлены в эффективный набор. Если бит выключен, то после execve(2) ни одна из новых разрешенных возможностей не будет добавлена в новый эффективный набор.

Включение эффективного файлового бита подразумевает, что любая файловая разрешительная или наследуемая возможность, позволяющая нити получить соответствующую разрешительную возможность при execve(2), также получит ее в эффективном наборе.

Поэтому если при назначении возможностей файлу (setcap(8), cap_set_file(3), cap_set_fd(3)) указать эффективный флаг как включенный для любой возможности, эффективный флаг должен также быть указан включенным для всех остальных возможностей, для которых включен соответствующий разрешительный или наследуемый флаги.

Преобразование возможностей при execve().

При execve(2) ФБО вычисляют новые возможности субъекта доступа (процесса) по следующему алгоритму:

```
P'(ambient) = (привилегированный файл) ? 0 : P(ambient)
P'(permitted) = (P(inheritable) & F(inheritable)) |
(F(permitted) & cap_bset) | P'(ambient)
P'(effective) = F(effective) ? P'(permitted) : P'(ambient)
P'(inheritable) = P(inheritable) [т. е., не изменяется],
```

где:

– P – значение набора возможностей нити до execve(2);

– P' – значение набора возможностей после execve(2);

– F – файловый набор возможностей;

– cap_bset – значение ограничивающего набора возможностей;

– привилегированный файл – файл, имеющий возможности, или для него установлен бит set-user-ID или set-group-ID.

ВОЗМОЖНОСТИ И ВЫПОЛНЕНИЕ ПРОГРАММЫ СУПЕРПОЛЬЗОВАТЕЛЕМ

Для предоставления полного набора возможностей суперпользователю в execve(2) необходимо выполнение следующих правил:

– если выполняется программа с установленным битом set-user-ID-root или ID реального пользователя процесса равен 0 (суперпользователь), предоставляются полные файловые и наследуемые наборы возможностей (т.е. разрешены все возможности);

– если выполняется программа с установленным битом set-user-ID-root, эффективный файловый бит равен единице (установлен).

Результат указанных правил, объединенных с преобразованиями возможностей, следующий: когда процесс выполняет execve(2) для программы с битом set-user-ID-root или когда процесс с эффективным UID 0 выполняет execve(2), он получает все возможности из разрешительного и эффективного наборов, за исключением тех, которые отменены ограничивающим набором возможностей. Это предоставляет семантику, совпадающую с обычными системами UNIX.

ОГРАНИЧИВАЮЩИЙ НАБОР ВОЗМОЖНОСТЕЙ

Ограничивающий набор возможностей – это механизм безопасности, который можно использовать для ограничения возможностей, доступных при `execve(2)`, следующим образом:

– при `execve(2)` ограничивающий набор возможностей складывается (AND) с файловым разрешительным набором возможностей, и результат этой операции назначается разрешительному набору возможностей нити. Таким образом, ограничивающий набор возможностей ограничивает разрешенные возможности, которые может предоставить исполняемый файл;

– ограничивающий набор возможностей представляет собой перечень, который нить может добавить в свой наследуемый набор с помощью `capset(2)`. Это означает, что если возможность отсутствует в ограничивающем наборе, нить не может добавить эту возможность в свой наследуемый набор, даже если она присутствует в разрешительном наборе, следовательно, не может сохранить данную возможность в разрешительный набор при вызове `execve(2)` для файла, имеющего возможность в своем наследуемом наборе.

Ограничивающий набор скрывает файловые разрешительные возможности, но не наследуемые возможности. Если нить имеет в своем наследуемом наборе возможность, которая отсутствует в ограничивающем наборе, нить по-прежнему обладает этой возможностью в своем разрешительном наборе при выполнении файла, имеющего возможность в своем наследуемом наборе.

Только процесс 1 может задавать возможности в ограничивающем наборе возможностей, помимо этого суперпользователь (точнее, программы с возможностью `CAP_SYS_MODULE`) могут лишь удалять возможности из набора.

Ограничивающий набор наследуется при `fork(2)` от нити родителя и сохраняется при `execve(2)`.

Нить может удалять возможности из своего ограничивающего набора с помощью вызова `prctl(2)` с операцией `PR_CAPBSET_DROP` при наличии возможности `CAP_SETPCAP`.

После удаления возможности из ограничивающего набора восстановить его невозможно. Нить может определить наличие возможности в своем ограничивающем наборе с помощью вызова `prctl(2)` с операцией `PR_CAPBSET_READ`.

Удаление возможностей из ограничивающего набора доступно, только если ядро собрано с поддержкой файловых возможностей.

Для сохранения привычной семантики при переходе от 0 к ненулевым пользовательским ID ФБО осуществляют следующие изменения наборов возможностей нити при изменении у нити реального, эффективного, сохраненного ID и пользовательского ID ФС (с помощью `setuid(2)`, `setresuid(2)` или подобных):

– если ранее реальный, эффективный или сохраненный пользовательский ID не был равен 0 и в результате изменения UID все эти ID получили ненулевое значение, то все возможности удаляются из разрешительного и эффективного наборов;

– если эффективный пользовательский ID изменяется с 0 на ненулевое значение, то все возможности удаляются из эффективного набора;

– если эффективный пользовательский ID изменяется с ненулевого значения на 0, то разрешительный набор копируется в эффективный набор;

– если пользовательский ID ФС изменяется с 0 на ненулевое значение, то из эффективного набора удаляются следующие возможности: `CAP_CHOWN`, `CAP_DAC_OVERRIDE`, `CAP_DAC_READ_SEARCH`, `CAP_FOWNER`, `CAP_FSETID`, `CAP_LINUX_IMMUTABLE`, `CAP_MAC_OVERRIDE` и `CAP_MKNOD`. Если пользовательский ID ФС изменяется с ненулевого значения на 0, то любая из возможностей, включенных в разрешительный набор, включается в эффективном наборе.

Если нить, у которой один или более пользовательских ID равен 0, стремится предотвратить удаление разрешительных возможностей при сбросе всех пользовательских ID в ненулевые значения, она может использовать вызов `prctl(2)` с операцией PR_SET_KEEPCAPS или флагом безопасности `SECBIT_KEEP_CAPS`, описанным далее.

Нить может получать и изменять свои наборы возможностей с помощью системных вызовов `capget(2)` и `capset(2)`. Однако для этой цели предпочтительнее использовать `cap_get_proc(3)` и `cap_set_proc(3)` из пакета `libcap`.

При изменении наборов нити применяются следующие правила:

– если вызывающий не имеет возможности `CAP_SETPCAP`, то новый наследуемый набор должен быть поднабором комбинации существующего наследуемого и разрешительного наборов;

– новый наследуемый набор должен быть поднабором комбинации существующего наследуемого и ограничивающего наборов;

– новый разрешительный набор должен быть поднабором существующего разрешительного набора (т.е. невозможно приобрести разрешительные возможности, которых нить не имеет);

– новый эффективный набор должен быть поднабором нового разрешительного набора.

ФЛАГИ SECUREBITS: ОРГАНИЗАЦИЯ ИСКЛЮЧИТЕЛЬНО ОКРУЖЕНИЯ

В ОС Аврора реализован набор флагов `securebits` (для каждой нити), который можно использовать для отключения специальных действий возможностей для UID 0 (суперпользователь). К этим флагам относятся:

– `SECBIT_KEEP_CAPS` – позволяет нити, у которой один и более UID равен 0, сохранить свои возможности при изменении всех ее UID на ненулевые значения. Если флаг не установлен, изменение UID приведет к утрате нитью всех возможностей. Данный флаг всегда сбрасывается при `execve(2)` (и предоставляет те же возможности, что и старый вызов `prctl(2)` с операцией `PR_SET_KEEPCAPS`);

– `SECBIT_NO_SETUID_FIXUP` – не позволяет ядру изменить наборы возможностей при изменении эффективного UID и UID ФС с 0 на ненулевое значение;

– SECBIT_NOROOT – в случае установки данного флага ядро не предоставляет возможности при исполнении программы, имеющей бит set-user-ID-root, или когда процесс с эффективным или реальным UID равным 0 вызывает execve(2);

– SECBIT_NO_CAP_AMBIENT_RAISE – запрещает повышение наружных возможностей посредством prctl(2) с операцией PR_CAP_AMBIENT_RAISE.

Каждый из перечисленных выше базовых флагов имеет дополнительный флаг блокировки, установка любого из которых является необратимой и запрещает дальнейшие изменения соответствующего базового флага.

Флаги блокировки:

– SECBIT_KEEP_CAPS_LOCKED;

– SECBIT_NO_SETUID_FIXUP_LOCKED;

– SECBIT_NOROOT_LOCKED;

– SECBIT_NO_CAP_AMBIENT_RAISE.

Флаги securebits можно изменять и получать с помощью вызова prctl(2) с операциями PR_SET_SECUREBITS и PR_GET_SECUREBITS. Для изменения флагов требуется возможность CAP_SETPCAP.

Флаги securebits наследуются дочерними процессами. При execve(2) все флаги сохраняются, за исключением SECBIT_KEEP_CAPS, который всегда сбрасывается.

МП может использовать следующий вызов для собственной блокировки и помещения всех своих потомков в окружение, где имеется только 1 способ добавления прав – запуск программы со связанными с ней файловыми возможностями:

```
prctl(PR_SET_SECUREBITS,
SECBIT_KEEP_CAPS_LOCKED |
SECBIT_NO_SETUID_FIXUP |
SECBIT_NO_SETUID_FIXUP_LOCKED |
SECBIT_NOROOT |
SECBIT_NOROOT_LOCKED);
```

4.3.6. Linux Security Modules

Дополнительно для кастомизации и уточнения применяемых ПРД может использоваться фреймворк Linux Security Modules.

Фреймворк является частью ядра и позволяет регистрировать процедуры, которые будут выполняться в случае успешного завершения стандартной процедуры проверки дискреционного доступа перед предоставлением фактического доступа к объекту.

Посредством Linux Security Modules в ядре ОС Аврора подключен модуль безопасности diehard, который не позволяет непривилегированному пользователю прервать выполнение процесса, запущенного от его же имени. Пользовательский процесс, не имеющий выделенной группы 777, не может послать сигнал процессу, имеющему такую группу, даже если процессы выполняются с одинаковым EUID. Исключением являются процессы, имеющие capability CAP_KILL, – на них не распространяются ограничения, накладываемые модулем diehard.

Заголовочный файл include/linux/security.h содержит описание структуры security_ops, представляющей собой список заранее определенных и документированных callback-функций, которые доступны модулю безопасности для выполнения проверок. По умолчанию данные функции в основном возвращают 0, разрешая любые действия. Однако некоторые используют модуль безопасности POSIX. В данной структуре специфический для ОС Аврора модуль diehard зарегистрирован следующим образом.

```
#ifdef CONFIG_SECURITY_DIEHARD

extern int diehard_task_kill(struct task_struct *p, struct siginfo
*info,
                    int sig, u32 secid);

#else /* !CONFIG_SECURITY_DIEHARD */

static inline int diehard_task_kill(struct task_struct *p,
                         struct siginfo *info, int sig, u32 secid)
{
     return 0;
}

#endif /* !CONFIG_SECURITY_DIEHARD */
```

Исходный текст модуля diehard (файл security/diehard/diehard-lsm.c):

```
#define DIE_HARD_GROUP 777

static const int zero = 0, one = 1;
static int enabled = 0;

int
diehard_task_kill(struct task_struct *p, struct siginfo *info, int sig,
                  u32 secid)
{
     const struct cred *cred  = current_cred();
     const struct cred *tcred = __task_cred(p);

     if (!enabled)
          return 0;

     if (ns_capable(tcred->user_ns, CAP_KILL))
          return 0; /* Privileged process are not affected by us */

     if (!gid_eq(cred->gid,  DIE_HARD_GROUP) &&
         !gid_eq(cred->egid, DIE_HARD_GROUP) &&
         !gid_eq(cred->sgid, DIE_HARD_GROUP) &&
         groups_search(tcred->group_info, DIE_HARD_GROUP) &&
         !groups_search(cred->group_info, DIE_HARD_GROUP))
          return -EACCES;

     return 0;
```

```
}

#ifndef CONFIG_SECURITY_DIEHARD_STACKED

static struct security_operations diehard_ops = {
        .name      = "diehard",
        .task_kill = diehard_task_kill,
};

#endif /* CONFIG_SECURITY_DIEHARD_STACKED */

#ifdef CONFIG_SYSCTL

struct ctl_path diehard_sysctl_path[] = {
     { .procname = "kernel", },
     { .procname = "diehard", },
     { }
};

static struct ctl_table diehard_sysctl_table[] = {
{
.procname       = "enabled",
.data           = &enabled,
.maxlen         = sizeof(int),
.mode           = 0644,
.proc_handler   = proc_dointvec_minmax,
.extra1         = (void *)&zero,
.extra2         = (void *)&one,
},
{ }
};

#endif /* CONFIG_SYSCTL */

static __init int
diehard_init(void)
{
#ifndef CONFIG_SECURITY_DIEHARD_STACKED
        if (!security_module_enable(&diehard_ops))
                return 0;
#endif /* CONFIG_SECURITY_DIEHARD_STACKED */

printk(KERN_INFO "DieHard: starting.\n");

#ifndef CONFIG_SECURITY_DIEHARD_STACKED
if (register_security(&diehard_ops))
panic("DieHard: kernel registration failed.\n");
#endif /* CONFIG_SECURITY_DIEHARD_STACKED */

#ifdef CONFIG_SYSCTL
if (!register_sysctl_paths(diehard_sysctl_path, diehard_sysctl_table))
panic("DieHard: sysctl registration failed.\n");
```

```
#endif /* CONFIG_SYSCTL */

return 0;
}

security_initcall(diehard_init);
```

Примером использования описанного механизма может служить необходимость обеспечить невозможность непривилегированного пользователя прервать выполнение антивирусного средства.

4.3.7. Управление политиками безопасности

Для определения возможностей учетной записи пользователя по использованию ресурсов и функциональных возможностей ОС Аврора применяется ролевая модель, на общесистемном уровне позволяющая администратору задать ограничения на использование функционала ОС посредством политик безопасности.

ПРИМЕЧАНИЕ. Настройка управления доступом, а также изменение данной настройки доступны только администратору либо через MDM-систему.

Для перехода к настройкам политик безопасности необходимо выполнить следующие действия:

– открыть меню настроек касанием значка на Экране приложений;

– коснуться пункта меню «Политики безопасности в подразделе «Безопасность» , в результате чего отобразится одноименная страница управления политиками безопасности (Рисунок 81).

– быстрый поиск политики безопасности с помощью ввода первых букв ее названия в поле «Поиск»;

– включение и выключение политики безопасности касанием переключателя справа от выбранной политики;

ПРИМЕЧАНИЕ. При активации переключатель начнет светиться ярче, чем в состоянии по умолчанию (неактивном).

– блокировка и разблокировка политики безопасности касанием соответствующего значка слева от выбранной политики либо касанием поля, в котором расположена выбранная политика.

ПРИМЕЧАНИЕ. Значок указывает на то, что политика разблокирована (доступна), значок указывает на то, что политика заблокирована (недоступна).

В случае если какая-либо из политик заблокирована администратором, при попытке доступа к ней отобразится сообщение «Отключено при помощи Aurora Device Manager» (Рисунок 81).

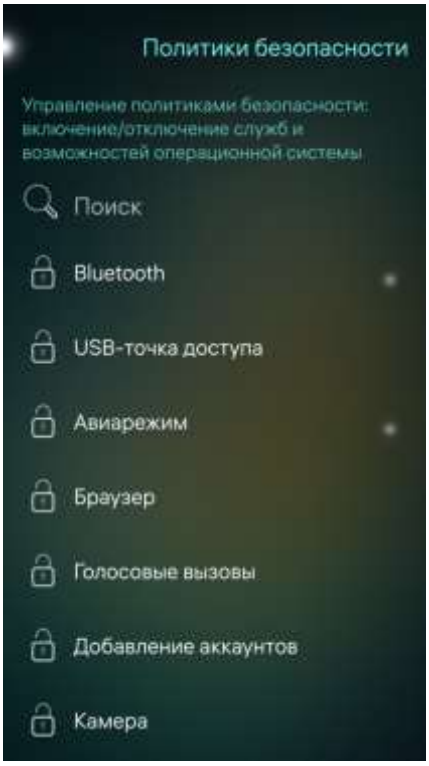

Рисунок 81

Рисунок 82

Наименование и описание управляемых политик безопасности приведено в таблице (Таблица 2).

ВНИМАНИЕ! Состояние политики по умолчанию отличается в зависимости от версии ОС Аврора.

Таблица 2

№	Название политики	Описание политики	Атрибут безопасности	Политика по умолчанию	Управления политикой
1	Bluetooth	Использование интерфейса Bluetooth®	BluetoothToggleEnabled	Заблокирована	GUI
2	USB точка доступа	Использование МУ в качестве USB-модема	UsbConnectionSharingEnabled	Разблокирована	GUI
3	Авиарежим	Использование режима полета	FlightModeToggleEnabled	Разблокирована	GUI
4	Браузер	Работа пользователя с браузером	BrowserEnabled	Разблокирована	GUI
5	Голосовые вызовы	Работа с голосовыми вызовами	VoiceCallEnabled	Разблокирована	GUI

6	Добавление аккаунтов	Добавление данных учетных записей	AccountCreationEnabled	Разблокирована	GUI
7	Камера	Использование камеры	CameraEnabled	Разблокирована	GUI
8	Микрофон	Использование микрофона	MicrophoneEnabled	Разблокирована	GUI
9	Настройка VPN	Управление VPN-соединениями	VpnConnectionSettingsEnabled	Разблокирована	GUI
10	Настройка VPN-соединения	Настройка/редактирование VPN-соединений	VpnConfigurationSettingsEnabled	Разблокирована	GUI
11	Настройки SIM-карт	Управление SIM-картами	SimSlotsSettingsEnabled	Разблокирована	GUI
12	Настройки WLAN	Использование сети WLAN	WlanToggleEnabled	Разблокирована	GUI
13	Настройки геолокации	Использование служб местоположения	LocationSettingsEnabled	Разблокирована	GUI
14	Настройки глобальных сервисов	Настройки глобальных сервисов	GlobalServiceSettingsEnabled	Разблокирована	GUI
15	Настройка даты и времени	Изменение настроек времени и даты	DateTimeSettingsEnabled	Разблокирована	GUI
16	Настройки мобильной сети	Настройки мобильной сети	MobileNetworkSettingsEnabled	Разблокирована	GUI
17	Настройки прокси	Настройка прокси-сервера	NetworkProxySettingsEnabled	Заблокирована	GUI
18	Общий доступ к Интернету	Использование МУ в качестве беспроводной точки доступа	InternetSharingEnabled	Разблокирована	GUI
19	Передача файлов на ПК (MTP)	Передача файлов по протоколу MTP	UsbMtpEnabled	Заблокирована	GUI

20	Сброс к заводским настройкам	Выполнение сброса ОС к заводским настройкам	DeviceResetEnabled	Разблокирована	GUI
21	Снимки экрана	Создание снимков экрана	ScreenshotEnabled	Разблокирована	GUI
22	Сообщения	Отправка SMS	SMSEnabled	Разблокирована	GUI
23	Установка приложений из файлового менеджера	Установка МП	ApplicationInstallationEnabled	Заблокирована	GUI
24	-	Использование Android Debug Bridge	UsbAdbEnabled	Разблокирована	policy.conf
25	-	Настройка мобильных точек доступа	MobileDataAccessPointSettingsEnabled	Разблокирована	policy.conf
26	-	Работа с обновлением ОС	OsUpdatesEnabled	Разблокирована	policy.conf
27	-	Принятие решений о загрузке сторонних пакетов	SideLoadingSettingsEnabled	Разблокирована	policy.conf
28	-	Активация режима разработчика	DeveloperModeSettingsEnabled	Разблокирована	policy.conf
29	-	Использование режима USB Mass Storage	UsbMassStorageEnabled	Разблокирована	policy.conf
30	-	Работа со статистикой интернет-данных	NetworkDataCounterSettingsEnabled	Разблокирована	policy.conf
31	-	Работа со статистикой звонков	CallStatisticsSettingsEnabled	Разблокирована	policy.conf

32	-	Изменение типа технологии мобильной передачи данных	CellularTechnologySettingsEnabled	Разблокирована	policy.conf
33	-	Режим разработчика	UsbDeveloperModeEnabled	Разблокирована	policy.conf
34	-	Режим сетевого адаптера	UsbHostEnabled	Разблокирована	policy.conf
35	-	Режим отладки	UsbDiagnosticModeEnabled	Разблокирована	policy.conf

4.4. Ограничение программной среды

В ОС Аврора ограничения программной среды могут задаваться администратором с помощью:

– управления службами посредством `systemd`, подробное описание которого приведено в приложении (Приложение 2);

– изменения лимитов, описание которых приведено в приложении (Приложение 3).

В ОС Аврора при установке ПО выполняются следующие проверки:

– содержания RPM-пакета на предмет исполняемых сценариев в разделе `postinstall`;

– установки программ без применения битов `suid-bit` и `sgid-bit`.

4.4.1. Механизм подписи RPM-пакетов

4.4.1.1. Подпись и проверка подписи RPM-пакета

В ОС Аврора используется механизм подписи RPM-пакетов и его содержимого, при этом:

– для пакетов отключены и не используются GPG-подписи;

– подпись разработчика подписывает RPM-пакет целиком;

– клиентская подпись подписывает область подписи разработчика;

– каждая последующая подпись добавляется в конец предыдущей и подписывает ее.

Доступ к функциям менеджера RPM-пакетов осуществляется администратором с помощью МП «Terminal» (п. 2.3.2).

Описание интерфейса команды `rpm` приведено в приложении (Приложение 1).

Каждый RPM-пакет имеет название, состоящее из следующих частей:

– название программы;

– версия программы;

– номер выпущенной версии. Обозначает количество пересборок программы одной и той же версии или дистрибутив, под который собран RPM-пакет (например, mdv (Mandriva Linux) или fc4 (Fedora Core 4));

– архитектура, под которую собран RPM-пакет (armv7hl, i386, ppc и т. д.).

Собранный RPM-пакет обычно имеет следующий формат названия:

```
<название>-<версия>-<релиз>.<архитектура>.rpm
```

Например:

```
nano-0.98-2.i386.rpm
```

RPM-пакет может содержать только исходные коды, при этом информация об архитектуре отсутствует и заменяется на `src`.

Например:

```
libgnomeuimm2.0-2.0.0-3.src.rpm
```

Библиотеки распространяются в 2 отдельных пакетах: первый содержит собранный код, второй (обычно к нему добавляют `-devel`) содержит заголовочные файлы, а также файлы, требуемые для разработки.

Необходимо, чтобы версии двух пакетов совпадали, в противном случае библиотеки могут работать некорректно. Пакеты с расширением `noarch.rpm` не зависят от конкретной архитектуры МУ и обычно содержат графику, архитектурно независимые скрипты, а также тексты, используемые другими программами.

RPM-пакет обеспечивает:

– легкость удаления и обновления ПО;

– популярность – часто ПО собирается именно в RPM, необходимость сборки программы из исходных кодов отсутствует;

– неинтерактивную установку – процесс установки/обновления/удаления легко автоматизируется;

– проверку целостности пакетов с помощью КС и подписей;

– `DeltaRPM` – аналог набора изменений, позволяющий обновить установленное ПО с минимальной затратой трафика;

– возможность аккумуляции опыта сборщиков в spec-файле;

– относительную компактность spec-файлов за счет использования макросов.

Специфика работы RPM-пакетов в составе ОС Аврора связана с переработанным механизмом КЦ (подраздел 4.9).

ПРИМЕЧАНИЕ. В составе ОС Аврора допускается установка только подписанных RPM-пакетов.

Подпись RPM-пакета проверяется в момент его установки и хранится в ОС Аврора после окончания процесса установки в особой базе данных (БД), называемой OMPCERT. Подписи RPM-пакета формируются с помощью алгоритма ГОСТ Р 34.10-2012.

Подпись хранится в стандартном для библиотеки OpenSSL формате CMS и содержит следующие данные:

– хеш содержимого пакета или подписи предыдущего уровня с применением функции хеширования и длиной хеш-кода 256 бит по ГОСТ Р 34.11-2012;

– дату подписания пакета;

– подпись указанных выше полей с применением алгоритма ГОСТ Р 34.10-2012 и функции хеширования ГОСТ Р 34.11-2012, длина выхода 256 бит;

– сертификат субъекта подписи пакета.

ПРИМЕЧАНИЕ. Для проверки подписи в процессе установки RPM-пакета имеет значение структура и состав сертификата ключа проверки ЭП. Сертификат должен быть выдан клиенту (субъекту) предприятием-разработчиком.

Процесс получения субъектом сертификата состоит из следующих этапов:

– генерация ключевой пары, защищенной паролем;

ПРИМЕЧАНИЕ. Подробная информация приведена на веб-сайте: https://community.omprussia.ru/.

– создание запроса на сертификат с указанием в обязательном порядке наименования клиента;

– отправка запроса на получение сертификата предприятию-разработчику и получение сертификата с присвоенной меткой группы безопасности;

– подписание RPM-пакета, полученного сертификата и защищенного ключа подписи.

Метка группы безопасности является строкой и может быть произвольной, однако предприятие-разработчик ОС Аврора при выдаче сертификата использует определенный набор меток. Обработка подписанного пакета зависит от присвоенной сертификату метки группы безопасности.

В начале эксплуатации ОС Аврора не связана с каким-либо субъектом. При установке первого пакета устанавливается ключ клиента (`client certificate key id`) и создается привязка ОС Аврора к клиенту (субъекту).

ПРИМЕЧАНИЕ. На ОС Аврора может быть установлен только RPM-пакет с подписью клиента.

Для установки стороннего ПО на МУ, функционирующее под управлением ОС Аврора, RPM-пакет должен иметь следующие подписи:

– подпись разработчика, которая является обязательной и позволяет идентифицировать автора пакета, а также используется для подписи пакета и исполняемых файлов внутри него;

ПРИМЕЧАНИЕ. Без подписи разработчика невозможно установить МП на МУ.

– подпись клиента, которая дополнительно защищает и контролирует установку стороннего ПО на МУ. Все МП, которые будут использоваться предприятием-разработчиком, необходимо подписывать подписью клиента.

ПРИМЕЧАНИЕ. На МУ можно установить только ПО, прошедшее внутренний контроль и подписанное собственной подписью клиента.

Для подписи МП требуются 2 ключевые подписанные пары и 2 сертификата (Таблица 3).

Таблица 3

Назначение	Алгоритм	Имя файла закрытого ключа по умолчанию	Имя файла запроса на сертификат по умолчанию	Имя файла сертификата по умолчанию
Подпись бинарных файлов и библиотек внутри RPM-пакета	RSA 2048	binaries-key.pem	binaries-csr.pem	`binaries-cert.pem`
Подпись RPM-пакетов	ГОСТ Р 34.10-2012 (256 бит)	packages-key.pem	packages-csr.pem	`packages-cert.pem`

ПРИМЕЧАНИЕ. Генерацию ключевых пар и запросы на сертификаты необходимо запускать внутри `build-engine`, подробная информация о котором приведена на веб-сайте: https://community.omprussia.ru/.

Пример команды для генерации ключевых пар и запросов на сертификаты:

```
customer-gen-csrs \ --common-name "developer company name" \ --
binaries-key binaries-key.pem \ --packages-key packages-key.pem
```

В процессе выполнения команды будут запрошены пароли для шифрования файлов с закрытыми ключами и в рабочей директории скрипта будут созданы файлы запросов `binaries-csr.pem` и `packages-csr.pem`.

Файлы запросов (не файлы ключей) необходимо передать предприятию-разработчику и получить взамен подписанные файлы сертификатов.

ПРИМЕЧАНИЕ. При проведении финального тестирования созданных МП потребуется сертификат сторонней организации на подпись RPM-пакета.

Сертификат сторонней организации необходимо дополнительно запросить у предприятия-разработчика. Создавать дополнительные ключи и запросы на сертификат не требуется. В дальнейшем именем по умолчанию для файла сертификата сторонней организации будет считаться `packages-client-cert.pem`.

Ввиду отсутствия механизмов для изменения привязки в дальнейшей эксплуатации необходимо выполнить одно из следующих действий:

– повторно установить ОС Аврора на МУ;

– установить пакет, удаляющий привязку, после чего повторно привязать ОС Аврора к клиенту с помощью пакета, который должен быть подписан подписью клиента (к клиенту должен быть привязан экземпляр ОС Аврора).

ПРИМЕЧАНИЕ. Для проверки подписи RPM-пакета и подписи файлов IMA используется единый сертификат, т.е. для подписи пакета и его содержимого используется одна подпись.

Механизм безопасности IMA обеспечивает отсутствие возможности запуска на МУ для неподписанных исполняемых файлов RPM-пакета, а также для исполняемых файлов RPM-пакета, подписанного неверной подписью либо подписью, которая верна, но отличается от текущего корневого сертификата.

ВНИМАНИЕ! Поддерживаются алгоритмы для: IMA SHA256, RSA2048 и ГОСТ 34.10 2012.

ПРИМЕЧАНИЕ. При отзыве скомпрометированных ключей происходит отзыв ключа, а не сертификата.

В целях разделения зоны и поддержания глубины интеграции сторонних RPM-пакетов имеются дополнительные подгруппы для подписей: Regular, Extended, MDM, Antivirus, которые различаются набором правил и разрешений по расположению и взаимодействию файлов в ОС, а также использованием взаимосвязанных компонентов.

ПРИМЕЧАНИЕ. Для упрощения процесса у разработчика сохраняется возможность отключения валидации пакетов (проверки на предмет корректности), однако при этом основные критические для системы проверки останутся активными.

Группы (метки) безопасности RPM-пакетов: после привязки экземпляра ОС Аврора невозможна установка RPM-пакетов, не подписанных клиентом либо подписанных другим клиентом.

RPM-пакет считается подписанным клиентом, если сертификат последней подписи пакета имеет метку группы безопасности `client`.

Для группы безопасности требуется следующее:

– обязательная подпись разработчика, метка – `developer`;

– опциональная подпись специализированной лаборатории, утверждающая безопасность RPM-пакета, метка – `seclab`;

– обязательная подпись клиента, метка – `client`.

Подписи накладываются следующим образом:

1. Разработчик подписывает RPM-пакет, который он произвел;

2. RPM-пакет с подписью разработчика может быть направлен в лабораторию для дополнительной независимой проверки. В этом случае лаборатория подписывает пакет с подписью разработчика своим ключом;

3. RPM-пакет в обязательном порядке подписывается подписью клиента (владельцем ОС Аврора, субъектом) для его успешной установки в ОС Аврора, при этом разработчик подписывает непосредственно RPM-пакет, а каждый последующий субъект подписывает предыдущую подпись (т.е. лаборатория подписывает подпись разработчика, клиент подписывает подпись лаборатории), таким образом

организована иерархия подписей, в которой на каждом из этапов проверки можно выявить расхождения.

Корневой сертификат предприятия-разработчика: для установки доверия между сертификатами и системой предлагается иерархическая связь между сертификатами, которые используются для подписи пакетов. Разработчик ОС Аврора посредством утилиты `tk26sig` генерирует сертификат, который считается корневым. Далее предприятие-разработчик, используя данный сертификат, выдает сертификаты своим клиентам на основе их запроса это могут быть ключи разработчика, клиента и любые другие. Таким образом, всегда возможно проследить связь между цепочкой сертификатов: если выданный клиентам сертификат не выдан предприятием-разработчиком, он считается недействительным, соответственно, все попытки установить RPM-пакеты, подписанные сертификатами, выданными не предприятием-разработчиком, будут неудачными.

Путь до корневого сертификата предопределен на программной уровне: `/etc/rpm/rootcacert-omp.pem`.

Сертификат предприятия-разработчика: среди всех сертификатов для группы `developer` наиболее важным является сертификат, выданный предприятию-разработчику для разработки ОС Аврора. Такими сертификатами подписываются все продукты предприятия-разработчика и все системные RPM-пакеты.

Пакет, подписанный данным сертификатом, не подвергается процессу валидации (т.к. такие пакеты всегда считаются доверенными). Для отличия сертификатов идентификатор публичного ключа данного сертификата устанавливается в ФС по пути `/etc/rpm/system-developer-keyid` и при установке каждого пакета происходит проверка публичного ключа сертификата предприятия-разработчика, которым подписан пакет с ключом, расположенным в ФС.

Внутренняя структура подписей: подписи файла IMA интегрируются в блок подписи предприятия-разработчика, подпись IMA переносится из заголовка пакета. Только первая подпись (не весь файл) заверяется второй подписью.

Общий формат подписанного RPM-пакета приведен на рисунке (Рисунок 81).

Рисунок 81

Валидация RPM-пакетов: RPM-пакеты валидируются сценарием, находящимся в проекте `rpm-validator`. Процесс валидации проходят не все RPM-пакеты, а только не удовлетворяющие условия безопасности.

RPM-пакеты, подписанные ключом предприятия-разработчика, не проходят процесс валидации ввиду невозможности корректного обновления RPM-пакетов, входящих в состав ОС Аврора.

При установке через `librpm` RPM-пакет проходит следующие стадии:

– внутренние проверки;

– валидация плагином `rpm-plugin-validation`.

Плагин `rpm-plugin-validation`, вызванный перед установкой RPM-пакета, последовательно выполняет следующие действия:

– проверка подписей и сертификатов RPM-пакета;

– проверка привязки системы к определенному клиенту. При отсутствии данной проверки выполняется попытка привязки к клиенту, подпись которого стоит на RPM-пакете. Отсутствие подписи трактуется как ошибка установки;

– если RPM-пакет является обновлением, то проверяется, что сертификат разработчика не изменился (т.е. не изменился ли разработчик). Изменение сертификата предприятия-разработчика считается ошибкой установки;

– если RPM-пакет подписан не предприятием-разработчиком, выполняется валидация пакета. Неуспешная валидация считается ошибкой установки;

– вставка сертификата в системный IMA keyring. Неуспешный процесс считается ошибкой установки;

– вставка информации о подписях и сертификатах RPM-пакета в БД `rpmsign-external`. Неуспешный процесс считается ошибкой установки.

По завершении обработки RPM-пакета каждое действие создает сообщение аудита в `sdjd` (система аудита предприятия-разработчика (подраздел 4.1)) об успехе или неуспехе при установке или удалении RPM-пакета.

При неуспешной установке RPM-пакета данные из БД OMPCERT удаляются, т.к. в `librpm` не существует способа возврата транзакции после установки.

При установке любого RPM-пакета необходимо учитывать следующее:

– каждый RPM-пакет должен иметь как минимум подпись предприятия-разработчика пакета и подпись клиента, если RPM-пакет для установки получен из доверенного источника (например, из магазина МП);

– исполняемые файлы и загружаемые библиотеки, входящие в состав RPM-пакета, также должны быть подписаны ключом при сборке пакета в системе сборки.

Для проверки подписи RPM-пакета необходимо выполнить команду:

```
ompcert-cli verify someapplication.rpm -r rootcacert-omp.pem
```

Для просмотра важных атрибутов подписи (имени субъекта, метки и ID ключа) необходимо выполнить команду:

```
ompcert-cli dump someapplication.rpm
```

4.4.1.2. Подпись МП

Для подписи МП необходимо выполнить команду:

```
customer-sign \     --binaries-key binaries-key.pem \     --packages-
key packages-key.pem \     --packages-cert packages-cert.pem \
sampleapp.rpm
```

где:

– `sampleapp.rpm` – пакет, содержащий ПО;

– `binaries-key.pem` – закрытый ключ подписи бинарных файлов;

– `packages-key.pem` – закрытый ключ подписи пакетов;

– `packages-cert.pem` – сертификат подписи пакетов.

ВНИМАНИЕ! В процессе подписи будут запрошены пароли от файлов с закрытыми ключами для подписи бинарных файлов и RPM-пакетов.

Если требуется подпись от сторонней организации, необходимо выполнить команду:

```
ompcert-cli sign sampleapp.rpm packages-key.pem packages-client-
cert.pem
```

где:

– `sampleapp.rpm` – пакет, содержащий ПО;

– `packages-key.pem` – закрытый ключ подписи пакетов;

– `packages-client-cert.pem` – сертификат подписи пакетов от имени клиента.

Для изоляции МП используются песочницы, описание которых приведено в подразделе 4.5.

4.4.1.3. Проверка подписи бинарных файлов

В подписи RPM-пакета хранится секция с подписями IMA для каждого файла, содержащегося в пакете. Во время установки содержимого пакета на каждый файл устанавливается расширенный атрибут `security.ima`, куда помещается соответствующая данному файлу подпись IMA.

При запуске бинарного файла происходит обязательная проверка подписи IMA с помощью подсистемы ОС Linux IMA/EVM и сертификата, перемещенного в IMA keyring при установке RPM-пакета. В случае неуспешной проверки бинарный файл не будет запущен.

Для проверки подписи бинарных файлов необходимо выполнить команду:

```
rpm -q --qf  '[%{FILENAMES}\n%{FILESIGNATURES}\n]' package.rpm | grep -A1 /usr/bin/ | tail -n 1 | cut -c 7-14
```

где `package.rpm` – имя файла подписанного пакета.

Результатом работы программы будет 4 байта, например: `de39e183`.

Такая же последовательность цифр должна присутствовать в выводе команды `cat /proc/keys`, если сертификат подписи бинарных файлов был добавлен в папку `/etc/keys/ima`.

ПРИМЕЧАНИЕ. При проверке подписи МП может потребоваться перезагрузка МУ.

4.4.2. Работа с сертификатами

4.4.2.1. Добавление корневого сертификата УЦ

Корневой сертификат удостоверяющего центра (УЦ) представляет собой файл, содержащий шифрованную сервисную информацию об УЦ. На основе данного файла строится цепочка доверия сертификатам. Получив доступ к шифрованной информации, криптопровайдер подтверждает подлинность личной ЭП. Таким образом, любая ЭП, выпущенная УЦ, работает корректно только при наличии корневого сертификата.

Для добавления корневого сертификата УЦ в доверенные необходимо выполнить следующие действия:

– подключить МУ к ЭВМ с помощью USB-кабеля;

– выбрать режим «Протокол передачи данных (MTP)»;

– скопировать с ЭВМ и перенести на МУ сертификат УЦ;

– открыть МП «Terminal» и выполнить следующие команды:

```
devel-su
cp <путь к файлу> /etc/pki/ca-trust/source/anchors/
update-ca-trust
```

Загрузить корневой сертификат УЦ на МУ также возможно с помощью сети Интернет, выполнив в МП «Terminal» следующие команды:

```
devel-su
curl -o /etc/pki/ca-trust/source/anchors/root_ca.crt "https:
//url_сертификата/root_ca.crt"
update-ca-trust
```

4.4.2.2. Проверка сертификатов

Для проверки сертификатов необходимо выполнить следующие действия:

– загрузить корневые сертификаты с помощью команд:

```
curl -L http://community.omprussia.ru/files/doc/rootcacert-omp.pem -o
rootcacert-omp.pem
curl -L http://community.omprussia.ru/files/doc/ima-root-ca.x509.pem -
o ima-root-ca.x509.pem
```

– проверить сертификат подписи бинарных файлов с помощью команды:

```
echo "test" > testfile openssl smime -sign \     -in testfile \     -
signer binaries-cert.pem \     -inkey binaries-key.pem \     -out
testfile.sig openssl smime -verify \     -in testfile.sig \     -signer
binaries-cert.pem \     -CAfile ima-root.ca.x509.pem
```

– проверить сертификат подписи пакетов с помощью команды:

```
echo "test" > testfile openssl smime -sign \     -in testfile \     -
signer packages-cert.pem \     -inkey packages-key.pem \     -out
testfile.sig.gost openssl smime -verify \     -in testfile.sig.gost \ -
signer packages-cert.pem \     -CAfile rootcacart-omp.pem
```

– проверить сертификат сторонней организации с помощью команды:

```
echo "test" > testfile-client openssl smime -sign \     -in testfile-
client \     -signer packages-client-cert.pem \     -inkey packages-
key.pem \     -out testfile-client.sig.gost openssl smime -verify \   -
in testfile-client.sig.gost \     -signer packages-client-cert.pem \  -
CAfile rootcacart-omp.pem
```

4.5. Изоляция процессов

4.5.1. Изоляция адресных пространств

Решение задачи изоляции адресных пространств процессов основано на архитектуре ядра ОС Аврора, которое обеспечивает собственное изолированное адресное пространство для каждого процесса в системе.

Используемый механизм изоляции основан на страничном механизме защиты памяти, а также механизме трансляции виртуального адреса в физический, поддерживаемом модулем управления памятью. Одни и те же виртуальные адреса, с которыми работает процессор, преобразуются в разные физические адреса для разных адресных пространств. При этом процесс не может несанкционированным образом получить доступ к пространству другого процесса, т.к. непривилегированный пользовательский процесс лишен возможности работать с физической памятью напрямую.

ПРИМЕЧАНИЕ. Механизм разделяемой памяти позволяет получить доступ к одному и тому же участку памяти и находится под контролем политики дискреционного разграничения прав доступа.

Адресное пространство ядра защищено от пользовательских процессов с использованием механизма страничной защиты. Страницы пространства ядра являются привилегированными, и доступ к ним из непривилегированного кода вызывает исключение процессора, который обрабатывается ядром ОС корректным образом.

Единственным санкционированным способом доступа к ядру ОС из пользовательской программы является механизм системных вызовов, который гарантирует возможность выполнения пользователем только санкционированных действий.

Дополнительные механизмы изоляции процессов обеспечиваются применяемыми технологиями контейнеризации не только друг от друга, но и от внешних ресурсов, а также внешних ресурсов от процессов.

Программные средства разрешают процессу лишь определенный перечень действий, выполняемых по отношению к другим процессам и периферийным устройствам, включая постоянное запоминающее устройство, который определяется при установке пакета, содержащего исполняемый файл.

ПРИМЕЧАНИЕ. Подробное описание привилегированных и непривилегированных процессов приведено в подразделе 4.3.

4.5.2. Изоляция МП с использованием песочниц

В ОС Аврора реализована изоляция МП с использованием песочниц, основанных на свободно распространяемом продукте Firejail, который использует стандартные для ОС Linux механизмы `namespace` и `seccomp-bpf`, позволяющие определять список доступных для МП системных вызовов.

Firejail представляет собой легковесную песочницу и подходит для использования в МУ, при этом все пользовательские МП запускаются через sailjail-обертку над Firejail. Дополнительно для изоляции используется `xdg-dbus-proxy`, фильтрующий прокси для D-Bus.

Механизм `seccomp-bpf` запрещает некоторые системные вызовы, например: `mount/umount`, `ptrace`, `kexec` и др.

По умолчанию в `mount namespace` доступ к ФС ограничивается до:

– доступа на чтение и запись к данным конкретного МП (`$XDG_DATA_HOME`, `$XDG_CONFIG_HOME`, `$XDG_CACHE_HOME`);

– доступа только на чтение к `/usr/share/$ApplicationName`;

– доступа только на чтение к некоторым директориям и файлам в `/etc`, `/usr/share`, `/var/lib` и т.д.

Доступ к ФС может быть расширен при запросе соответствующих разрешений, описанных с помощью правил Firejail, например:

– разрешение «Pictures» предоставляет доступ на чтение и запись к директории пользователя `~/Pictures`, а также к кэшу превью;

– разрешение «RemovableMedia» предоставляет доступ к съемным носителям (`/media/$USER/`) и т.д.

В `net namespace` по умолчанию добавляется только `loopback device`.

Работа в песочнице позволяет определить МП разрешения на доступ к ресурсам, обусловленные элементами, описание которых приведено в таблице (Таблица 4).

Таблица 4

№	Элемент	Описание
1	`Accounts`	Просмотр, модификация и синхронизация учетных записей
2	`Ambience`	Установка и редактирование атмосфер
3	`AppLaunch`	Запуск и остановка сервисов `systemd`
4	`ApplicationInstallation`	Установка и удаление МП
5	`Audio`	Воспроизведение и запись аудио, изменение конфигурации
6	`Bluetooth`	Подключение и использование Bluetooth®-устройств
7	`Calendar`	Просмотр и модификация событий календаря
8	`CallRecordings`	Доступ к записанным вызовам
9	`Camera`	Доступ к камере, съемка фото и видео
10	`CommunicationHistory`	Доступ к истории вызовов и сообщений
11	`Contacts`	Просмотр и модификация данных контактов
12	`Documents`	Доступ к каталогу «Documents»
13	`Downloads`	Доступ к каталогу «Downloads»
14	`E-mail`	Чтение и отправка писем из электронной почты, доступ к вложениям
15	`Internet`	Использование сети Интернет
16	`Location`	Использование геопозиционирования
17	`MediaIndexing`	Доступ к перечню файлов на МУ
18	`Messages`	Доступ к чтению и отправке SMS
19	`Microphone`	Запись аудио с помощью микрофона
20	`Music`	Доступ к каталогу «Music», плейлистам и обложкам
21	`NFC`	Подключение и использование NFC-устройств
22	`Phone`	Осуществление вызовов напрямую или через пользовательский интерфейс
23	`Pictures`	Доступ к каталогу «Pictures»
24	`PublicDir`	Доступ к каталогу «Public»

№	Элемент	Описание
25	RemovableMedia	Использование карт памяти и USB
26	Synchronization	Доступ к каркасу синхронизации
27	UserDirs	Доступ к каталогам «Documents», «Downloads», «Music», «Pictures», «Public» и «Video»
28	Videos	Доступ к каталогу «Videos»
29	WebView	Для использования Gecko WebView
30	AccessSecurityLog	Доступ к регистрационному журналу
31	DeviceInfo	Извлечение данных о МУ
32	LogSecurityEvents	Запись в регистрационный журнал
33	PushNotifications	Чтение push-уведомлений
34	Reports	Генерирование архива с системными отчетами
35	SecureStorage	Хранение шифрованных файлов
36	UserStatus	Извлечение списка пользователей системы

Для каждого МП, выполняющегося в песочнице, может быть создан профиль, определяющий его возможности и поведение. Например, профиль для МП «Погода» может выглядеть следующим образом:

```
# Firetail > Firejail profile for /usr/bin/omp-weather

### security filters
caps.drop all
nonewprivs
seccomp

### network
protocol unix,

### environment
shell none

### baseline
include permission-baseline.inc

### application
private-bin omp-weather,
private-etc passwd,group,location,dconf,xdg,fonts,system-fips,selinux,
private-tmp
dbus-user.own ru.omprussia.weather
whitelist /usr/share/omp-weather

# Settings
include sessionbus-com.jolla.settings.inc

# Connman
include systembus-net.connman.inc
```

4.6. Защита памяти

4.6.1. Очистка памяти

Очистка оперативной памяти основана на архитектуре ядра ОС Аврора и гарантирует, что обычный непривилегированный процесс не может получить данные чужого процесса, если это явно не разрешено ПРД.

Средства взаимодействия между процессами контролируются с помощью ПРД, и процесс не может получить доступ к неочищенной памяти (как оперативной, так и дисковой). Ядро выделяет каждому процессу виртуальное адресное пространство, которое транслируется в физические адреса памяти с поддержкой рандомизации.

Доступные для пользовательского процесса функции выделения и распределения памяти осуществляют выполнение режима инициализации, при котором происходит обнуление ячеек памяти. Таким образом, ядро и системная библиотека `libc` гарантируют получение процессом только очищенных страниц памяти без остаточной информации.

Очистка памяти на внешних носителях (eMMC) основана на реализации механизма `secdel`, который очищает на носителе неиспользуемые блоки ФС непосредственно при их освобождении с помощью перезаписи их маскирующей последовательностью.

4.6.2. Перезапись остаточной информации

В ОС Аврора реализована функция перезаписи остаточной информации, срабатывающая каждый раз при удалении файла. Процесс перезаписи происходит автоматически и скрыт для пользователя. В случае необходимости осуществить принудительную перезапись случайными или специальными битовыми последовательностями администратор имеет возможность самостоятельно запустить процедуру удаления файла и/или каталога.

Данная процедура приводит к снижению производительности, особенно при удалении большого файла: чем файл больше, тем снижение существеннее и заметнее, т.к. на период гарантированного удаления все остальные операции записи на носитель информации приостанавливаются и помещаются в очередь до завершения удаления.

ОС Аврора функционирует на МУ, память которых состоит из набора (множества) ячеек. Ввиду конструктивных особенностей МУ каждая ячейка имеет большое и конечное количество циклов перезаписи, т.е. постоянные операции по чтению и записи с/на USB-накопитель влекут увеличение счетчика обращений (количество которых ограничено на аппаратном уровне) и, как следствие, постепенное снижение срока службы накопителя.

Нерациональное использование многократной перезаписи ОС Аврора влечет снижение производительности на время проведения операции удаления и снижение срока службы накопителя данных МУ.

Для многократной перезаписи используется утилита командной строки wipe, доступ к которой осуществляется с помощью МП «Terminal» (подраздел 2.4).

Подробное описание интерфейса wipe приведено в приложении (Приложение 4).

4.7. Обеспечение надежного функционирования

4.7.1. Надежные метки времени

Метка времени считается надежной при условии невозможности внести в нее изменения после создания, если целостность данной метки не была нарушена.

Надежные метки времени обеспечиваются архитектурой хронометража ОС, которая представляет собой набор структур данных и функций ядра, имеющих отношение к отслеживанию хода времени и базирующихся на:

- счетчике отметок времени (TSC);
- программируемом таймере интервалов (PIT);
- APIC-таймере;
- высокоточном таймере событий (HPET).

Начальные значения системных таймеров задаются по таймеру реального времени МУ, а базовые интервалы вычисляются при инициализации системы.

В качестве низкоуровневого механизма ядра, обеспечивающего функционирование надежных меток времени, используется специальный набор структур данных ядра (`timespec xtime`), функций ядра `do_gettimeofday()` и низкоуровневых системных вызовов `clock_stettime(), clock_gettime()`.

4.7.2. Квотирование постоянной памяти

Квотирование — это разделение ограниченного дискового пространства МУ между учетными записями пользователей, позволяющее создавать одновременно несколько учетных записей пользователей на МУ (подраздел 1.2).

В ОС Аврора предусмотрена возможность выделения квоты – объема дискового пространства для учетной записи пользователя при ее создании, при этом у учетной записи пользователя отсутствует возможность претендовать на больший объем постоянной памяти, а выделенное ей пространство будет недоступно для других учетных записей пользователей даже в случае, если оно свободно.

ВНИМАНИЕ! Объем квоты задается при создании учетной записи пользователя и не может быть изменен впоследствии.

Для того, чтобы задать учетной записи пользователя квоту на использование дискового пространства, необходимо на странице создания учетной записи пользователя установить количество выделяемых ему ГБ, перемещая соответствующий слайдер вправо для увеличения квоты либо влево для уменьшения (см. Рисунок 3).

ОС Аврора автоматически вычисляет допустимые пределы квотирования таким образом, чтобы можно было создать до 7 учетных записей пользователей, при этом квоты устанавливаются следующим образом:

– нижняя граница квоты задается в 2 ГБ и менее, если общий объем встроенной памяти на МУ менее 14 ГБ;

– верхняя граница выбирается с расчетом, чтобы для каждой учетной записи пользователя оставался хотя бы минимальный объем памяти.

Для вычисления границ квоты используются следующие переменные:

– `AllSpace` – общий объем дискового пространства МУ;

– `Nusr` – количество учетных записей, созданных на МУ;

– `Qi` – объем дискового пространства МУ, занятый созданными учетными записями;

– `UsrAvSpace = AllSpace – 2 GB` – 2ГБ выделяется для учетной записи администратора;

– `MinQuote = min (UsrAvSpace/6 , 2 GB)` – выделяется минимальная квота 2 ГБ, при наличии небольшого объема пространства выделяется весь доступный объем, разделенный на 6 учетных записей пользователей;

– `MaxQuote = UsrAvSpace - sum (Qi) - MinQuote x (6 - Nusr)` – максимальная квота, которую можно выделить создаваемой на МУ учетной записи пользователя: все доступное пространство за вычетом суммы уже выделенных квот, а также количества учетных записей пользователей, которые могут быть созданы в будущем, при этом квота для текущей учетной записи на будущее не резервируется.

ПРИМЕЧАНИЕ. Максимальные квоты учетных записей пользователя задаются администратором в конфигурационном файле `/etc/security/limits.conf` (Приложение 3). Также для управления аппаратными квотами из различных пользовательских программ используются следующие системные вызовы: `setrlimit(), getrlimit(), prlimit() и nice()`.

Примеры разбиения дискового пространства (в ГБ) для некоторых поддерживаемых МУ приведены в таблице (Таблица 5).

Таблица 5

МУ	Общий объем	Минимальный объем, доступный пользователю	Максимальный объем, доступный пользователю
Mashtab TrustPhone T1	64	2	32
Qtech QMP-M1-N-IP	32	2	10
Aquarius NS220 v2.1	16	1	1

4.8. Фильтрация сетевого потока

4.8.1. Общая информация

Фильтрация сетевых потоков в ОС Аврора осуществляется с помощью встроенного в ядро ОС фильтра сетевых пакетов `netfilter` и монитора обращений, контролирующего сетевой стек IPv4.

Администратор при помощи утилиты `iptables` может задавать модулю ядра `netfilter` правила (или цепочки) фильтрации в соответствии с атрибутами отправителя и получателя сетевых пакетов, а также атрибутами передаваемой информации в IP-заголовках пакетов.

4.8.2. Межсетевое экранирование

Функции МЭ в ОС Аврора реализованы в службе `connman`, использующей механизмы `iptables` для разграничения сетевых потоков данных.

Конфигурация по умолчанию поставляется в пакете `connman-configs-core`, собираемом из пакета `omp-common-configurations`.

4.8.3. Путь к файлам конфигурации МЭ

Новые настройки `iptables` поддерживаются в `connman`, начиная с версии 1.32+git41. Настройки, описанные в настоящем документе, хранятся в следующих файлах:

– `/etc/connman/firewall.conf – [1];`

– `/etc/connman/firewall.d/*firewall.conf – [2]`,

где [1] – основная конфигурация, [2] – каталог с файлами, оканчивающимися на «firewall.conf» и расположенными в алфавитном порядке.

4.8.3.1. Правила конфигурации МЭ

Бинарный пакет «connman-configs-core» устанавливает правила в следующих файлах:

1) `/etc/connman/firewall.conf`:

– IPv4:

- разрешает установленные и связанные пакеты (`-m conntrack --ctstate RELATED,ESTABLISHED`);
- разрешает все входящие пакеты для loopback-интерфейса (требуется `connman` для разрешения доменных имен);
- по умолчанию блокирует весь входящий трафик (таблица `filter` INPUT-цепочка);

– IPv6:

- разрешает установленные и связанные пакеты (`-m conntrack --ctstate RELATED,ESTABLISHED`);
- разрешает все входящие пакеты для loopback-интерфейса (требуется `connman` для разрешения доменных имен для IPv6-протокола);

- по умолчанию блокирует весь входящий трафик (таблица `filter` INPUT-цепочка);

ПРИМЕЧАНИЕ. В ОС Аврора IPv6-поддержка в `iptables` не заявлена.

2) `/etc/connman/firewall.d/10-block-icmp-firewall.conf`:

– IPv4:

- разрешает входящие ICMP-пакеты, отличные от ping-пакетов (т.е. блокировать только входящие ping-пакеты);
- разрешает исходящие ICMP-пакеты, отличные от эхо-ping-пакетов (т.е. блокировать только исходящий эхо-ответ).

– IPv6:

- разрешает входящие ICMP-пакеты, отличные от ping-пакетов (т.е. блокировать только входящие ping-пакеты);
- разрешает исходящие ICMP-пакеты, отличные от эхо-ping-пакетов (т.е. блокировать только исходящий эхо-ответ);

3) `/etc/connman/firewall.d/11-allow-dccp-non-privileged-ports-firewall.conf`:

– для IPv4 и IPv6:

- разрешает входящий DCCP-трафик на портах 1024:65535;

4) `/etc/connman/firewall.d/11-allow-sctp-non-privileged-ports-firewall.conf`:

– для IPv4 и IPv6:

- разрешает входящий SCTP-трафик на портах 1024:65535;

5) `/etc/connman/firewall.d/11-allow-tcp-non-privileged-ports-firewall.conf`:

– для IPv4 и IPv6:

- разрешает входящий TCP-трафик на портах 1024:65535.

6) `/etc/connman/firewall.d/11-allow-udp-non-privileged-ports-firewall.conf`:

– для IPv4 и IPv6:

- разрешает входящий UDP-трафик на портах 1024:65535;
- разрешает входящий UDP Lite-трафик на портах 1024:65535.

7) `/etc/connman/firewall.d/12-allow-ipsec-firewall.conf`:

– для IPv4 и IPv6:

- разрешает весь входящий IPSec Authentication Header (AH)-трафик;
- разрешает весь входящий IPSec Encapsulating Security Payload (ESP)-трафик;

8) `/etc/connman/firewall.d/12-allow-ipv6-mobility-firewall.conf`:

– только для IPv6:

- разрешает весь входящий IPv6 Mobility Header (MH)-трафик.

4.8.3.2. Точка доступа

Конфигурация режима «Точка доступа» реализована как дополнительная опция. При включенном режиме применяются те же правила по умолчанию, что и при приеме всего трафика. Более строгие правила фильтрации трафика могут быть добавлены с помощью меню настроек. С этой целью в файлы конфигурации добавлена группа [tethering], аналогичная динамическим правилам для групп.

ПРИМЕЧАНИЕ. Правила «Точка доступа» применяются только для режима передачи данных посредством сети WLAN. Для USB-режима правила по умолчанию применяются независимо от конфигурации режима «Точка доступа».

Правила режима «Точка доступа» должны быть целостными и завершенными. При наличии хотя бы одного набора правил режима «Точка доступа» никакие правила по умолчанию не будут добавлены, т.к. они переопределяют пользовательские правила и разрешают весь трафик.

Поэтому, например, могут быть включены следующие входящие правила, разрешающие только DHCP и DNS `/etc/connman/firewall.d/42-tethering-firewall.conf`:

```
[tethering]

IPv4.INPUT.RULES = -p udp -m udp --dport 53 -j ACCEPT; -p tcp -m tcp --
dport 53 -j ACCEPT; -p udp -m udp --dport 67 -j ACCEPT

IPv6.INPUT.RULES = -p udp -m udp --dport 53 -j ACCEPT; -p tcp -m tcp --
dport 53 -j ACCEPT; -p udp -m udp --dport 67 -j ACCEPT
```

4.8.3.3. Файлы конфигурации

4.8.3.3.1. Формат файла

Каждая группа определяется с помощью квадратных скобок – [], например, [General]. Каждый ключ регистрозависим и написан простым текстом без тегов, за ним следует символ «равно» (=). Это означает, что все ключи до начала следующей группы [] принадлежат этой группе.

Например:

```
IPv4.INPUT.RULES = -p tcp -m tcp -j ACCEPT
```

Каждый ключ правил может существовать в одной группе только один раз (проверку осуществляет анализатор файлов ключей glib, обрабатывается только первый ключ).

Разделитель для правил – точка с запятой (;).

Правила могут быть выключены при помощи символа «#» в начале правила.

Например, выключить первое правило `--dport 23` и применить правило `--dport 24`:

```
#-p udp -m udp --dport 23 -j ACCEPT; -p udp -m udp --dport 24 -j ACCEPT
```

4.8.3.3.2. Порядок обработки

Первым загружается файл основной конфигурации (`/etc/connman/firewall.conf`), далее загружаются остальные файлы конфигурации из `/etc/connman/firewall.d` в алфавитном порядке.

Порядок обработки правил:

– ключи из файлов загружаются в алфавитном порядке, поэтому правила в файле, например, с префиксом 00, будут обработаны и добавлены первыми;

– правила из файла основной конфигурации добавляются в `iptables` последними и считаются основными;

– если включены динамические правила, то они добавляются в начало в `iptables`.

Например:

1) Файлы конфигурации и порядок их обработки:

– 1: основной файл `firewall.conf` – содержит [General] (Общие) правила и устанавливает POLICY (Политику);

– 2: `firewall.d/10-firewall.conf` – содержит WLAN-правила;

– 3: `firewall.d/20-firewall.conf` – содержит [General] (Общие) правила;

– 4: `firewall.d/30-firewall.conf` – содержит [General] (Общие) и WLAN-правила;

2) Правила МЭ при запуске `connman`:

– применяется Политика из пункта 1;

– порядок применения Правил:

- правила из пункта 3 [General];
- правила из пункта 4 [General];
- правила из пункта 1 [General];

3) Правила МЭ после включения WLAN:

– применяется Политика из пункта 1;

– порядок применения Правил:

- правила из пункта 2 [WLAN];
- правила из пункта 4 [WLAN];
- правила из пункта 3 [General];
- правила из пункта 4 [General];
- правила из пункта 1 [General].

При использовании нескольких разных правил:

– правила из раздела [General] (Общие). Последнее определение POLICY (Политики) перезаписывает предыдущие;

– правила из каждого типа группы добавляются к существующим правилам.

При добавлении или удалении файлов конфигурации необходимо выполнить команду:

```
systemctl reload connman
```

– если установлен новый пакет, достаточно выполнить перезагрузку;

– правила из нового файла конфигурации добавляются во внутренние списки по порядку, однако:

- порядок в `iptables` корректируется динамическими правилами после восстановления соединения (например, WLAN);
- порядок в `iptables` корректируется [General] (Общими) правилами после перезапуска `connman`: применяется только при наличии правил, которые должны быть добавлены в определенную позицию на основе порядка, определяемого именем файла.

При изменении существующего файла конфигурации необходимо выполнить команду:

```
systemctl restart connman
```

Обнаружение изменений в файлах конфигурации будет реализовано в следующих версиях.

4.8.3.3.3. Группы и ключи

Общие группы [General]:

– IPv4.INPUT.RULES = #Набор правил в таблице `filter`, INPUT-цепочка для протокола IPv4;

– IPv4.OUTPUT.RULES = #Набор правил в таблице `filter`, OUTPUT-цепочка для протокола IPv4;

– IPv4.FORWARD.RULES = #Набор правил в таблице `filter`, FORWARD-цепочка для протокола IPv4;

– IPv4.INPUT.POLICY = #Политика по умолчанию для таблицы `filter`, INPUT-цепочка (может быть ACCEPT или DROP);

– IPv4.OUTPUT.POLICY = #Политика по умолчанию для таблицы `filter`, OUTPUT-цепочка (может быть ACCEPT или DROP);

– IPv4.FORWARD.POLICY = #Политика по умолчанию для таблицы `filter`, FORWARD-цепочка (может быть ACCEPT или DROP);

– IPv6.INPUT.RULES = #Набор правил в таблице `filter`, INPUT-цепочка для протокола IPv6;

– IPv6.OUTPUT.RULES = #Набор правил в таблице `filter`, OUTPUT-цепочка для протокола IPv6;

– IPv6.FORWARD.RULES = #Набор правил в таблице `filter`, FORWARD-цепочка для протокола IPv6;

– IPv6.INPUT.POLICY = #Политика по умолчанию для таблицы `filter`, INPUT-цепочка для протокола IPv6 (может быть ACCEPT или DROP);

– IPv6.OUTPUT.POLICY = #Политика по умолчанию для таблицы `filter`, OUTPUT-цепочка для протокола IPv6 (может быть ACCEPT или DROP);

– IPv6.FORWARD.POLICY = #Политика по умолчанию для таблицы `filter`, FORWARD-цепочка для протокола IPv6 (может быть ACCEPT или DROP).

Динамические группы по типам устройств:

– группа активируется при вызове службы `connman` данным типом устройства (например, WLAN-подключение);

– каждый тип правила будет содержать в себе интерфейс сервиса (например, для INPUT-правил будет установлен параметр `-i <interface>`);

– группа может содержать следующие ключи (такие же, как в разделе Общие [General]:

- IPv4.INPUT.RULES =;
- IPv4.OUTPUT.RULES =;
- IPv4.FORWARD.RULES =;
- IPv6.INPUT.RULES =;
- IPv6.OUTPUT.RULES =;
- IPv6.FORWARD.RULES =;

– предопределенные группы:

- [unknown] – не используется или не поддерживается, но является частью внутренних типов устройств `connman`;
- [system];
- [ethernet];
- [wifi];
- [bluetooth];
- [cellular];
- [gps];
- [vpn];
- [gadget];
- [p2p];
- [tethering] – настройки, применяемые для режима «Точка доступа».

ПРИМЕЧАНИЕ. При наличии проблем с режимом «Точка доступа» необходимо добавить следующие строки в группу Общие [General]: например, `95-tether-override-firewall.conf`:

```
[General]
IPv4.INPUT.RULES = -i tether -j ACCEPT
IPv6.INPUT.RULES = -i tether -j ACCEPT
```

4.8.3.3.4. Формат правил

Выполняются следующие правил iptables-формата (https://www.frozentux.net/iptables-tutorial/iptables-tutorial.html):

– опции проверяются для каждого протокола и/или типа соответствия;

– значения опций проверяются на величину и тип значения;

– если правило не существует в iptables, оно игнорируется анализатором правил;

– отрицания поддерживаются как в iptables.

Поскольку в первой версии МЭ поддерживаются не все возможности переключателей (switches) для iptables, переключатели, которые connman не может добавить, игнорируются iptables:

– модификаторы цепочек: -A, -D, -X, -F, -I, -P, -E, -R, -Z (и их длинные эквиваленты):

- новые или существующие цепочки не изменяются правилами. Все правила МЭ подчиняются логике connman, в которой подобные модификаторы не предусмотрены;
- в случае возникновения необходимости такие модификаторы могут быть добавлены в дальнейшем;

– указатели места назначения для DNAT: --to-destination, --from-destination;

– переключатели фрагментирования: -f, --fragment;

– переключатель версии IP-протокола: --ipv4, -4, --ipv6, -6;

– переключатели соответствия (-m), которые не поддерживаются:

- IPv4: -m comment, -m state, -m iprange, -m recent, -m owner, -m sctp, -m dccp, -m hashlimit, -m icmpv6/ipv6-icmp;
- IPv6: -m comment, -m state, -m iprange, -m recent, -m owner, -m ttl, -m sctp, -m dccp, -m mh, -m hashlimit, -m frag, -m icmp;
- вышеуказанные переключатели вызывали сбои в iptables или неправильно определяли версию IP-протокола.

Цели МЭ (-j TARGET) соответствуют целям iptables по умолчанию: ACCEPT, DROP, REJECT, LOG и QUEUE.

Протоколы МЭ (-p protocol) аналогичны протоколам iptables: tcp, udp, udplite, icmp, icmpv6, ipv6-icmp, esp, ah, sctp, mh (только для IPv6) и специальному ключевому слову all.

Каждое правило:

– должно иметь минимум 1 цель (-j/--jump TARGET or -g/--goto TARGET);

– может иметь от 0 до 1 соответствия по протоколу (-p/--protocol protocol);

– для SCTP-, DCCP- и MH-протоколов, механизм соответствия протокола не может быть использован, например:

```
-p sctp -m sctp --dport 22 -j DROP (не будет работать, но)
-p sctp --dport 22 -j DROP (будет работать)
```

– может иметь от 0 до 2 указателей соответствия (-m/--match match).

Например, чтобы разрешить для telnet одну попытку соединения в секунду, необходимо установить:

```
-p udp -m udp --dport 23 -m limit --limit 1/second --limit-burst 1 -j ACCEPT
```

– может иметь:

- от 0 до 2 переключателей портов с опцией обычного порта (одно и то же направление не может использоваться дважды). С модификатором протокола (`-m <protocol>`): `--destination-port`, `--dport`, `--source-port`, `--sport`;
- от 0 до 1 переключателя портов с мультипортом. Вместе с указателем соответствия `-m multiport` возможно использовать: `--destination-ports`, `--dports`, `--source-ports`, `--sports`, `--port`, `--ports` и переключатели;

– может иметь от 0 до 2 указателей места назначения (одно и то же направление не может использоваться дважды). `--source`, `--src`, `-s`, `--destination`, `--dst`, `-d`;

– каждое правило раздела [General] (Общие) может иметь от 0 до 2 переключателей интерфейса (одно и то же направление не может использоваться дважды). `--in-interface`, `-i`, `--out-interface`, `-o`. Переключатели интерфейса игнорируются в динамических правилах, предназначенных для типов услуг (`service types`).

4.8.3.3.5. Устранение неполадок

При работе с МП в случае возникновения проблемы сети/iptables (МЭ) необходимо выполнить следующие действия:

1) Настроить базовый мониторинг работы iptables для следующих протоколов:

– IPv4:

```
watch -n 1 iptables -t filter -L -v -n
```

– IPv6:

```
watch -n 1 ip6tables -t filter -L -v -n
```

2) Для выхода из программы нажать сочетание клавиш «Ctrl» и «C».

3) Проверить вывод и количество сброшенных пакетов в INPUT-цепочке (политика DROP).

4) Запустить МП и проверить, какие счетчики увеличивают свое значение.

ПРИМЕЧАНИЕ. Если сетевые пакеты сбрасываются, счетчик DROP будет увеличивать свое значение. Счетчики пакетов для каждого правила будут увеличиваться, если пакеты соответствуют этому правилу.

5) Если проблема в работе МП возникла из-за сбрасывания пакетов, необходимо выполнить следующие действия:

– для временного решения (действует до перезагрузки системы), разрешающего весь трафик по умолчанию следующих протоколов:

- для IPv4 выполнить команду:

```
iptables -t filter -P INPUT ACCEPT
```

- для IPv6 выполнить команду:

```
ip6tables -t filter -P INPUT ACCEPT
```

– для решения на постоянной основе создать файл `/etc/connman/firewall.d/99-accept-all-firewall.conf`, добавить в него следующие строки:

```
[General]
IPv4.INPUT.POLICY = ACCEPT
IPv6.INPUT.POLICY = ACCEPT
```

– и перезапустить МП `connman`:

```
systemctl restart connman
```

Если система не загружается, необходимо перейти в режим восстановления и записать в файл опции, указанные на шаге 4, для решения на постоянной основе.

4.9. Контроль целостности

Подсистема КЦ ОС Аврора служит для проверки неизменности среды выполнения, оповещения пользователя об изменении критически важных компонентов, а также запрещает загрузку и использование системы при нарушении ее целостности.

В ОС Аврора КЦ реализован с помощью программного компонента `integrity`, который представляет собой демон для управления и проверки целостности данных. В отличие от других инструментов `integrity` использует ЭП (IMA и `secureboot`) как источник доверия.

ПРИМЕЧАНИЕ. Каждый устанавливаемый в ОС Аврора пакет ПО должен иметь цифровую подпись (п. 2.3.2).

`Integrityd` отвечает за управление и верификацию целостности объектов, при этом под объектами понимаются обычные файлы, ELF-файлы и разделы. Объекты могут объединяться в группы, также может быть установлена иерархия групп.

Информация о запуске/завершении процедуры КЦ и ее результатах записывается в системный журнал, просмотр которого осуществляется с помощью МП «Журнал».

Автоматически, без необходимости специальных действий со стороны администратора, ОС Аврора отслеживает следующее:

– целостность устанавливаемых пакетов ПО формата .rpm (п. 2.3.2);
– целостность загружаемых внешних модулей уровня ядра;
– целостность всех исполняемых файлов при попытке их запуска;
– целостность разделов.

В случае необходимости выполнить КЦ произвольного файла, ELF-файла, разделов или группы разделов следует использовать утилиту `integrityd`, доступную в МП «Terminal», которая применяет ЭП как источник доверия.

ВНИМАНИЕ! Перед выполнением КЦ требуется обратить внимание на следующее:

– проверка целостности указанных файлов по умолчанию производится каждый раз при загрузке ОС Аврора, а также в 00:00 часов один раз в сутки;

– время загрузки ОС Аврора увеличивается пропорционально количеству файлов, требуемых для проверки целостности, т.е. чем больше список файлов на проверку, тем больше времени занимает проверка целостности;

– при проверке целостности выполняются математические операции, что приводит к повышению использования ресурсов процессора, следовательно, к увеличению расхода заряда аккумулятора МУ.

ПРИМЕЧАНИЕ. В результате получения от сервиса `integrityd` отрицательного статуса проверки целостности, который свидетельствует о нарушении целостности, доступ к ОС Аврора блокируется компонентом `securityd`. В этом случае необходимо передать МУ администратору для повторной установки ОС Аврора.

5. РЕКОМЕНДАЦИИ ПО УСТРАНЕНИЮ ВОЗМОЖНЫХ ОШИБОК

Действия по устранению возможных ошибок приведены в таблице (Таблица 6).

Таблица 6

№	Ошибка	Причина/рекомендации по устранению
1	Невозможно выполнить обновления, т.к. не работает кнопка «Загрузить»	Для получения обновления ОС Аврора требуется доступ к определенным ресурсам предприятия-разработчика. Такой доступ предоставляется клиентам, оформившим техническую поддержку, включающую услугу обновления ОС. Более подробная информация доступна по электронной почте support@omp.ru
2	Не получается установить МП	Установка МП на МУ, функционирующее под управлением ОС Аврора, возможна следующими способами: – с использованием ППО «Аврора Центр», развернутой для компании-клиента: внутри периметра организации или у внешнего поставщика (ИТ-интегратора); – непосредственно на МУ в режиме разработчика. Для переключения МУ в данный режим необходимо выполнить операции, описанные в документе «Руководство пользователя»; – из графического интерфейса ОС Аврора; – с использованием МП «Terminal»
3	Ошибка сертификата	Один из доверенных сертификатов, входящих в третье поколение ОС, устарел, в результате чего была утрачена доверенность ресурсов, подписанных такими сертификатами
4	При получении обновлений с ППО «Аврора Центр» отображается сообщение «Appmanager is busy»	Проблемы с сетевым доступом МУ к ППО. Если при скачивании МП происходит сетевой разрыв, то последующие МП становятся в очередь и их установка не происходит
5	Отображается сообщение «Управление обновлением ОС запрещено»	Установленный на МУ mdm-клиент (например, клиент ППО) запрещает обновления через интерфейс МУ

№	Ошибка	Причина/рекомендации по устранению
6	МУ заблокировано, требуется код доступа	Необходимо обратиться к администратору МУ либо оператору ППО для сброса пароля МУ, подключенного к сети Интернет
7	При установке RPM на МУ возникает ошибка	Несовместимость ключей в ОС и ключей, которыми подписано МП. Необходимо переподписать МП либо использовать ОС с ключами, соответствующими МП
8	Пользователь забыл код безопасности МУ	Необходимо обратиться к администратору для сброса пароля МУ
9	Администратор забыл код безопасности МУ	Необходимо последовательно выполнить следующие действия: – ввести МУ в Recovery Mode (выключить МУ одновременным нажатием кнопки уменьшения громкости и кнопки включения МУ), в результате чего на экране отобразится строка «Recovery: Connect USB cable and open telnet to address 10.42.66.66»; – подключить МУ к ЭВМ через USB-кабель; – открыть терминал и ввести команду: `telnet 10.12.66.66`; – выбрать пункт «Reset device to factory state»; нажать «Enter» 5 раз, после чего ожидать «Wipe Device»
10	После загрузки обновлений МП на МУ оно не обновилось	Убедившись, что МУ имеет доступ к сети Интернет и заряд аккумулятора МУ составляет не менее 50%, необходимо осуществить принудительное обновление, выполнив следующие действия: – открыть Экран приложений, проведя по Домашнему экрану снизу вверх; – перейти в пункт меню системных настроек «Обновления ОС Аврора», после чего выбрать пункт «Проверить наличие обновлений». Если после выполнения описанных действий МП не было установлено либо обновлено, необходимо провести анализ системных сообщений (подраздел 4.1)
11	Невозможно сбросить МУ к заводским настройкам	Данная функция отключена с помощью политики безопасности, необходимо отключить данную политику в меню безопасности (подраздел 4.3)
12	Недоступны настройки даты и времени	Данная функция отключена с помощью политики безопасности, необходимо отключить данную политику в меню безопасности (подраздел 4.3)

ПЕРЕЧЕНЬ ТЕРМИНОВ И СОКРАЩЕНИЙ

В настоящем документе приняты следующие термины и сокращения (Таблица 7).

Таблица 7

Термин/ Сокращение	Расшифровка
Администратор	Пользователь, обладающий правами на выполнение операций, связанных с администрированием системы
БД	База данных
Версия ОС Аврора	1. Корпоративная версия - исполнение ОС Аврора, предназначенное для организации доверенных мобильных рабочих мест, на которых не происходит обработка информации, подлежащей защите в соответствии с законодательством РФ; 2. Сертифицированная версия - исполнение ОС Аврора, прошедшее сертификационные испытания в системе сертификации нормативных регуляторов РФ (ФСТЭК России, ФСБ России), имеющее соответствующий комплект программных документов и готовые к серийному производству. Предназначены для организации доверенных мобильных рабочих мест, на которых происходит обработка информации, подлежащей защите в соответствии с законодательством РФ. Могут использоваться в ГИС, на объектах КИИ и в иных регулируемых ИС
2ФА	Двухфакторная аутентификация
ИС	Информационная система
ИФБО	Интерфейс функциональных возможностей безопасности
Квота	Объем дискового пространства, выделяемого администратором для записи данных учетных записей пользователей
КЦ	Контроль целостности
МП	Мобильное приложение
МУ	Мобильное устройство
МЭ	Межсетевое экранирование
ОС	Операционная система

Термин/ Сокращение	Расшифровка
Переключатель	Элемент интерфейса ОС Аврора, представляющий собой светящуюся точку, расположенную в поле, и позволяющий выбрать одно из состояний, чаще всего включение или выключение. При активации переключателя точка начинает светиться ярче, чем в неактивном состоянии
Пользователь	Лицо, использующее систему для выполнения заложенных в ней функций
Предприятие-разработчик	Общество с ограниченной ответственностью «Открытая мобильная платформа» (ООО «Открытая мобильная платформа»)
ПО	Программное обеспечение
ППО	Прикладное программное обеспечение «Аврора Центр»
ПРД	Правила разграничения доступа
Суперпользователь	Пользователь, обладающий правами на выполнение всех без исключения операций в системе (в системе имеет логин «root»)
Токен	Аутентификационные данные, которые выдаются пользователю после успешной авторизации и являются ключом для доступа к службам
УЦ	Удостоверяющий центр
ФБО	Функциональные возможности безопасности
ФС	Файловая система
ЭВМ	Электронно-вычислительная машина
ЭП	Электронная подпись
ACL	Access Control List – список контроля доступа
DAC	Discretionary Access Control – дискреционное разграничение прав доступа
eMMC	Embedded Multimedia Memory Card – встроенная мультимедийная карта памяти
GID	Group IDentifier – идентификатор группы
GUI	Graphical User Interface - разновидность пользовательского интерфейса, в котором элементы интерфейса (меню, кнопки, значки, списки), представленные пользователю на дисплее, исполнены в виде графических изображений
HPET	High Precision Event Timer - таймер событий высокой точности
IMA	Integrity Measurement Architecture - механизм проверки подписи бинарных файлов
MD5	Message Digest 5 – 128-битный алгоритм хеширования

Термин/ Сокращение	Расшифровка
MTP	Media Transfer Protocol – основанный на PTP аппаратно-независимый протокол, разработанный компанией Microsoft для подключения цифровых плееров к компьютеру
NFC	Near field communication - технология беспроводной передачи данных малого радиуса действия, которая дает возможность обмена данными между устройствами, находящимися на расстоянии около 10 сантиметров
PID	Process IDentifier – идентификатор процесса
PIN-код	Personal Identification Number - персональный код, состоящий из 4 цифр, предназначенный для получения доступа к SIM-карте и предотвращающий ее несанкционированное использование
PUK-код	Personal Unlock Key - дополнительный код, состоящий из 8 цифр и применяемый для разблокировки SIM-карты после неудачного ввода значения PIN-кода 3 раза подряд
RPM	Red Hat Package Manager – менеджер пакетов Red Hat обозначает две сущности: формат пакетов ПО (RPM-пакет) и программа, созданная для управления этими пакетами. Программа позволяет устанавливать, удалять и обновлять ПО
RPM-пакет	Файл формата RPM, позволяющий устанавливать, удалять и обновлять приложение на МУ
SIM	Subscriber Identification Module – модуль идентификации абонента
SSU	SourceSafe для Unix – утилита, обеспечивающая доступ из командной строки к локальным и удаленным репозиториям Source Safe/VSS через TCP
SSH	Secure SHell — сетевой протокол прикладного уровня, позволяющий производить удаленное управление ОС и туннелирование TCP-соединений (например, для передачи файлов)
UID	User Identifier – идентификатор пользователя
USB	Universal Serial Bus – универсальная последовательная шина
VPN	Virtual Private Network — виртуальная частная сеть, обобщенное название технологий, позволяющих обеспечить одно или несколько сетевых соединений (логическую сеть) поверх другой сети (например, сети Интернет)
WLAN	Wireless Local Area Network – беспроводная локальная сеть

ПРИЛОЖЕНИЕ 1

Описание команды RPM

ИМЯ: `rpm` – менеджер RPM-пакетов.

ОБЗОР

ЗАПРОС И ПРОВЕРКА ПАКЕТОВ:

```
rpm {-q|--query} [опции-выбора] [опции-запроса]
rpm {-V|--verify} [опции-выбора] [опции-проверки]
rpm --import PUBKEY ...
rpm {-K|--checksig} [--nosignature] [--nodigest]
PACKAGE_FILE ...
```

УСТАНОВКА, ОБНОВЛЕНИЕ И УДАЛЕНИЕ ПАКЕТОВ:

```
rpm {-i|--install} [опции-установки] PACKAGE_FILE ...
rpm {-U|--upgrade} [опции-установки] PACKAGE_FILE ...
rpm {-F|--freshen} [опции-установки] PACKAGE_FILE ...
rpm {-e|--erase} [--allmatches] [--nodeps] [--noscripts]
[--notriggers] [--repackage] [--test] PACKAGE_NAME ...
```

РАЗНОЕ:

```
rpm {--initdb|--rebuilddb}
rpm {--addsign|--resign} PACKAGE_FILE ...
rpm {--querytags|--showrc}
rpm {--setperms|--setugids} PACKAGE_NAME ...
```

ОПЦИИ ВЫБОРА:

```
[PACKAGE_NAME] [-a,--all] [-f,--file FILE]
[-g,--group GROUP] {-p,--package PACKAGE_FILE]
[--fileid MD5] [--hdrid SHA1] [--pkgid MD5] [--tid TID]
[--querybynumber HDRNUM] [--triggeredby PACKAGE_NAME]
[--whatprovides CAPABILITY] [--whatrequires CAPABILITY]
```

ОПЦИИ ЗАПРОСА:

```
[--changelog] [-c,--configfiles] [-d,--docfiles] [--dump]
[--filesbypkg] [-i,--info] [--last] [-l,--list]
[--provides] [--qf,--queryformat QUERYFMT]
[-R,--requires] [--scripts] [-s,--state]
[--triggers,--triggerscripts]
```

ОПЦИИ ПРОВЕРКИ:

```
[--nodeps] [--nofiles] [--noscripts]
[--nodigest] [--nosignature]
[--nolinkto] [--nomd5] [--nosize] [--nouser]
[--nogroup] [--nomtime] [--nomode] [--nordev]
```

ОПЦИИ УСТАНОВКИ:

```
[--aid] [--allfiles] [--badreloc] [--excludepath OLDPATH]
[--excludedocs] [--force] [-h,--hash]
[--ignoresize] [--ignorearch] [--ignoreos]
```

```
[--includedocs] [--justdb] [--nodeps]
[--nodigest] [--nosignature] [--nosuggest]
[--noorder] [--noscripts] [--notriggers]
[--oldpackage] [--percent] [--prefix NEWPATH]
[--relocate OLDPATH=NEWPATH]
[--repackage] [--replacefiles] [--replacepkgs]
[--test]
```

ОПИСАНИЕ: RPM – это менеджер пакетов, который может быть использован для сборки, установки, запроса, проверки, обновления и удаления отдельных программных пакетов. RPM-пакет содержит архив с файлами и метаданные, используемые для установки и удаления файлов данного архива. Метаданные включают сценарии, атрибуты файлов и информацию с описанием RPM-пакета. RPM-пакеты могут быть двух типов: двоичные, используемые для объединения устанавливаемых программ, и пакеты с исходными кодами, включающие исходные коды и описание способа формирования двоичного пакета.

Необходимо указать один из следующих основных режимов работы: Запрос, Проверка, Проверка подписи, Установка/Обновление/Освежение, Удаление, Инициализация БД, Перестройка БД, Повторная подпись, Добавление подписи, Задание владельцев/групп, Отображение тегов запроса, Отображение конфигурации.

ОБЩИЕ ОПЦИИ:

Следующие опции могут быть использованы во всех режимах:

– `-?, --help` – выводит детальную справку об использовании;

– `--version` – выводит одну строку, содержащую номер версии RPM;

– `--quiet` – выводит как можно меньше сообщений, отображаются только сообщения об ошибках;

– `-v` – выводит подробную информацию, отображаются сообщения о выполнении всех шагов;

– `-vv` – выводит большой объем отладочной информации;

– `--rcfile FILELIST` – каждый из файлов разделенного двоеточиями списка файлов `FILELIST` последовательно прочитывается RPM на предмет конфигурационной информации. В этом списке существует только первый файл; все знаки тильда будут заменены значением `$HOME`. По умолчанию список файлов `FILELIST` выглядит как `/usr/lib/rpm/rpmrc:/usr/lib/rpm/redhat/rpmrc:/etc/rpmrc:~/.rpmrc`;

– `--pipe CMD` – перенаправляет вывод rpm на вход команды CMD;

– `--dbpath DIRECTORY` – использует БД RPM в каталоге `DIRECTORY` вместо пути по умолчанию `/var/lib/rpm`;

– `--root DIRECTORY` – использует для всех операций ФС с корнем в `DIRECTORY`.

ПРИМЕЧАНИЕ. БД внутри `DIRECTORY` будет использоваться для проверки зависимостей, и все сценарии `%post` и `%prep` будут исполняться после `chroot(2)` в `DIRECTORY`.

ОПЦИИ УСТАНОВКИ И ОБНОВЛЕНИЯ:

Общая форма команды установки RPM-пакета:

```
rpm {-i|--install} [опции-установки] PACKAGE_FILE ...
```

Общая форма команды обновления RPM-пакета:

```
rpm {-U|--upgrade} [опции-установки] PACKAGE_FILE ...
```

Таким образом выполняется установка или обновление уже установленного пакета до новой версии, при этом все другие версии удаляются после установки нового RPM-пакета. Если предыдущая версия RPM-пакета уже установлена, для обновления RPM-пакета необходимо выполнить команду:

```
rpm {-F|--freshen} [опции-установки] PACKAGE_FILE ...
```

Параметр `PACKAGE_FILE` может быть указан как адрес `ftp` или `http URL`, в таком случае пакет будет скачан перед установкой.

ОПЦИИ `FTP/HTTP` для получения информации о работе RPM-пакетов с поддержкой `ftp` или `http`:

– `--aid` –при необходимости добавляет предложенные RPM-пакеты в набор транзакции;

– `--allfiles` – устанавливает или обновляет все файлы с флагом `missingok` в RPM-пакете независимо от их существования;

– `--badreloc` – используется вместе с `--relocate`, разрешает перемещение всех путей файлов, а не только тех, для которых старые пути `OLDPATH` включены в заметки по перемещению (`relocation hint`) в бинарном пакете;

– `--excludepath OLDPATH` – не устанавливает файлы, названия которых начинаются с `OLDPATH`;

– `--excludedocs` – не устанавливает файлы, помеченные как документация (включающие `man` страницы и документы `texinfo`);

– `--force` – эквивалентно использованию `--replacepkgs`, `--replacefiles` и `–oldpackage`;

– `-h, --hash` – выводит 50 отметок при распаковке архива. Используется с `-v|--verbose` для удобства отображения;

– `--ignoresize` – не проверяет подключенные ФС на наличие необходимого места на диске перед установкой данного RPM-пакета;

– `--ignorearch` – разрешает установку или обновление, даже если архитектуры бинарного пакета и узла не совпадают;

– `--ignoreos` – разрешает установку или обновление, даже если ОС Аврора бинарного пакета и узла не совпадают;

– `--includedocs` – устанавливает файлы с документацией. Это поведение задано по умолчанию;

– `--justdb` – обновляет только информацию в базе, но не в ФС;

– `--nodigest` – не проверяет при чтении дайджест RPM-пакета или заголовка;

– `--nosignature` – не проверяет при чтении подпись RPM-пакета или заголовка;

– `--nodeps` – не проверяет зависимости перед установкой или обновлением RPM-пакета;

– `--nosuggest` – не предлагает RPM-пакеты для разрешения отсутствующих зависимостей;

– `--noorder` – не выполняет переупорядочивание RPM-пакетов для установки. Список пакетов обычно переупорядочивается для удовлетворения зависимостей;

– `--noscripts; --nopre; --nopost; --nopreun; --nopostun` – не выполняет скриптлеты с указанным именем. Опция `--noscripts` эквивалентна `--nopre --nopost --nopreun --nopostun` и выключает исполнение соответствующих `%pre, %post, %preun` и `%postun` скриптлетов;

– `--notriggers; --notriggerin; --notriggerun; --notriggerpostun` – не выполняет никакие триггерные скриптлеты с указанным именем. Опция `--notriggers` эквивалентна `--notriggerin --notriggerun --notriggerpostun` и выключает исполнение соответствующих `%triggerin, %triggerun` и `%triggerpostun` скриптлетов;

– `--oldpackage` – разрешает обновить или заменить новый пакет более старой версией;

– `--percent` – выводит информацию в процентах по мере распаковки файлов из архива пакета. Предназначена для упрощения вызова rpm из других утилит;

– `--prefix NEWPATH` – для перемещаемых бинарных пакетов, преобразовывает все пути файлов, которые начинаются с инсталляционного префикса в заметках, по перемещению (`relocation hint`) на `NEWPATH`;

– `--relocate OLDPATH=NEWPATH` – для перемещаемых бинарных пакетов. В заметках по перемещению (`relocation hint`) преобразовать все пути файлов, которые начинаются с `OLDPATH`, на `NEWPATH`. Данная опция может быть использована несколько раз, если требуется переместить несколько путей `OLDPATH` в пакете;

– `--repackage` – переупаковывает файлы перед удалением. Ранее установленный RPM-пакет будет назван в соответствии с макросом `%_repackage_name_fmt` и создан в каталоге, названном по значению макроса `%_repackage_dir` (значение по умолчанию – `/var/spool/repackage`);

– `--replacefiles` – устанавливает RPM-пакеты, даже если они заменяют файлы от других установленных RPM-пакетов;

– `--replacepkgs` – устанавливает RPM-пакеты, даже если они уже установлены в системе;

– `--test` – не устанавливая RPM-пакеты, выполняет проверку и сообщает о потенциальных конфликтах.

ОПЦИИ УДАЛЕНИЯ:

Общая форма команды удаления rpm:

```
rpm {-e|--erase} [--allmatches] [--nodeps] [--noscripts] [--notriggers]
[--repackage] [--test] PACKAGE_NAME ...
```

Также могут быть использованы следующие опции:

– `--allmatches` – удаляет все версии RPM-пакета, совпадающие с `PACKAGE_NAME`. Обычно при наличии нескольких RPM-пакетов, совпадающих с `PACKAGE_NAME`, возникает ошибка;

– `--nodeps` – не проверяет зависимости перед удалением RPM-пакетов;

– `--noscripts`; `--nopreun`; `--nopostun` – не выполняет скриптлеты с указанными именами. Наличие параметра --noscripts при удалении пакетов эквивалентно `--nopreun` `--nopostun` и выключает исполнение соответствующих скриптлетов %preun и `%postun`;

– `--notriggers`; `--notriggerun`; `--notriggerpostun` – не выполняет никакие триггерные скриптлеты с указанным именем. Опция --notriggers эквивалентна `--notriggerun` `--notriggerpostun` и выключает исполнение соответствующих `%triggerun` и %triggerpostun скриптлетов;

– `--repackage` – переупаковывает файлы перед удалением. Ранее установленный RPM-пакет будет назван в соответствии с макросом `%_repackage_name_fmt` и создан в каталоге, названном по значению макроса `%_repackage_dir` (значение по умолчанию – `/var/spool/repackage`);

– `--test` – не удаляя RPM-пакеты, выполняет проверку. Данный параметр удобно использовать при отладке совместно с опцией `-vv`.

ОПЦИИ ЗАПРОСА:

Общая форма команды запроса rpm:

```
rpm {-q|--query} [select-options] [query-options]
```

Существует возможность задать формат вывода информации об RPM-пакете. Для этого необходимо использовать следующий параметр с указанием строки формата `QUERYFMT` после него:

```
 --qf|--queryformat QUERYFMT
```

Форматирование запроса – это измененная версия стандартного механизма `printf(3)`, которая формируется из статических строк (в т.ч. могут включать стандартные для языка C escape-последовательности для перевода строки, табуляции и других специальных символов) и меток форматирования `printf(3)`. Указатель формата может быть пропущен и заменен на имя выводимого тега заголовка, заключенного в фигурные скобки {},т.к. тип выводимой информации известен RPM заранее. Имена тегов нечувствительны к регистру, и префикс `RPMTAG_` в имени тега можно опускать.

Альтернативные форматы вывода могут быть заданы при помощи задания типа вывода: `typetag` после имени тега.

Поддерживаются следующие типы:

- `:armor` – упаковывает публичный ключ в ASCII вид;
- `:base64` – кодирует двоичные данные в формат base64;
- `:date` – использует формат `strftime(3)` "%c";
- `:day` – использует формат `strftime(3)` "%a %b %d %Y";
- `:depflags` – форматирует флаги зависимостей;
- `:fflags` – форматирует флаги файлов;
- `:hex` – в шестнадцатеричном виде;
- `:octal` – в восьмеричном виде;
- `:perms` – форматирует права доступа файлов;
- `:shescape` – экранирует одиночные кавычки для применения в сценариях;
- `:triggertype` – выводит суффикс триггера.

Например, для вывода только имени RPM-пакета при запросе можно использовать строку формата `%{NAME}`. Для вывода имени RPM-пакетов и информации о дистрибутиве в две колонки можно использовать `%-30{NAME}%{DISTRIBUTION}`. Команда `rpm` отобразит список всех тегов, с которыми она может работать при ее вызове с параметром `--querytags`.

Существуют 2 набора параметров для выполнения запросов – выбор пакетов и указание информации.

ОПЦИИ ВЫБОРА ПАКЕТОВ:

- `PACKAGE_NAME` – выполняет запрос к установленному RPM-пакету с именем `PACKAGE_NAME`;
- `-a, --all` – выполняет запрос ко всем установленным RPM-пакетам;
- `-f, --file FILE` – выполняет запрос к RPM-пакету, владельцу файла FILE;
- `--fileid MD5` – выполняет запрос к RPM-пакету, содержащему указанный идентификатор файла, т.е. алгоритм MD5 дайджест содержимого файла;
- `-g, --group GROUP` – выполняет запрос к RPM-пакету с группой GROUP;
- `--hdrid SHA1` – выполняет запрос к RPM-пакету, содержащему указанный идентификатор заголовка, т.е. SHA1 дайджест неизменной части заголовка;
- `-p, --package PACKAGE_FILE` – выполняет запрос к (неустановленному) RPM-пакету в файле `PACKAGE_FILE`. Параметр `PACKAGE_FILE` может быть указан в виде адреса `ftp` или `http URL`, в результате чего заголовок пакета будет скачан и опрошен. Необходимо обратиться к ОПЦИЯМ FTP/HTTP за информацией о поддержке в `rpm` работы с `ftp` и `http`. Если аргумент (аргументы) `PACKAGE_FILE` не является бинарным пакетом, он будет интерпретирован как `ASCII` манифест пакета. В нем разрешено применение комментариев, начинающихся с '#'. Каждая из строк файла манифеста пакета может содержать разделенные запятыми `glob` выражения, включая адреса URL с внешними glob-выражениями (они будут развернуты в пути), которые будут подставлены вместо обращения пакета как дополнительный аргумент `PACKAGE_FILE` в запросе;

– `--pkgid MD5` – выполняет запрос к RPM-пакету, содержащему указанный идентификатор, т.е. MD5 дайджест объединенного содержимого заголовка и тела пакета;

– `--querybynumber HDRNUM` – выполняет запрос напрямую `HDRNUM'` записи в БД и используется только для отладки;

– `--specfile SPECFILE` – обрабатывает и выполняет запрос к `SPECFILE` файлу, как если бы это был RPM-пакет. Несмотря на то, что доступна не вся информация (например, список файлов), данный тип запросов позволяет использовать `rpm` для извлечения информации из `spec` файлов без написания специализированного анализатора таких файлов;

– `--tid TID` – выполняет запрос к RPM-пакету, содержащему указанный идентификатор транзакции TID. В качестве идентификатора используется временная метка `UNIX`. Все RPM-пакеты, установленные или удаленные в составе одной транзакции, будут иметь один и тот же идентификатор;

– `--triggeredby PACKAGE_NAME` – выполняет запрос к RPM-пакетам, которые вызывают срабатывание триггера пакета `PACKAGE_NAME`;

– `--whatprovides CAPABILITY` – выполняет запрос ко всем RPM-пакетам, предоставляющим функциональность `CAPABILITY`;

– `--whatrequires CAPABILITY` – выполняет запрос ко всем RPM-пакетам, которые требуют `CAPABILITY` для корректной работы.

ОПЦИИ ЗАПРОСА ПАКЕТОВ:

– `--changelog` – выводит информацию об изменениях в RPM-пакете;

– `-c, --configfiles` – выводит только конфигурационные файлы (подразумевает `-l`);

– `-d, --docfiles` – выводит только файлы документации (подразумевает `-l`);

– `--dump` – распечатывает информацию о файле в виде: `path size mtime md5sum mode owner group isconfig isdoc rdev symlink` – опция должна быть использована совместно с одной из `-l, -c, -d`;

– `--filesbypkg` – выводит все файлы во всех выбранных пакетах;

– `-i, --info` – выводит информацию об RPM-пакете, включая имя, версию и описание. Будет использована –`queryformat`, если указана;

– `--last` – упорядочивает вывод RPM-пакетов по времени установки таким образом, чтобы последние пакеты были выведены в начале;

– `-l, --list` – выводит список файлов в RPM-пакете;

– `--provides` – выводит функциональность (`capabilities`), предоставляемую RPM-пакетом;

– `-R, --requires` – выводит RPM-пакеты, от которых зависит данный пакет;

– `--scripts` – выводит скриптлет(ы) RPM-пакета, который используется как часть процесса установки или удаления RPM-пакетов;

– `-s, --state` – выводит состояние (`state`) файлов в RPM-пакете (подразумевается -l). Состояние каждого файла в пакете может быть одним из следующих: нормальное (`normal`), не установлен (`not installed`) или заменен (`replaced`);

– `--triggers, --triggerscripts` – отображает сценарии триггеров (если существуют), входящие в состав RPM-пакета.

ОПЦИИ ПРОВЕРКИ:

Общая форма команды проверки RPM:

```
rpm {-V|--verify} [опции-выбора] [опции-проверки]
```

Данная операция сравнивает информацию о файлах, установленных из RPM-пакета, с информацией о них из метаданных пакета, хранимых в БД RPM. Также при проверке сравниваются размер, MD5-сумма, права доступа, тип, владелец и группа каждого файла, при это любые расхождения будут отображены. Файлы, которые не были установлены вместе с RPM-пакетом, например, файлы документации, исключенные при помощи опции `--excludedocs`, игнорируются без предупреждения.

Опции выбора RPM-пакетов аналогичны запросу пакетов (включая файлы обращения пакета в качестве аргумента).

Опции, уникальные для режима проверки:

– `--nodeps` – не проверяет зависимости RPM-пакетов;

– `--nodigest` – не проверяет при чтении дайджест RPM-пакета или заголовка;

– `--nofiles` – не проверяет атрибуты файлов RPM-пакетов;

– `--noscripts` – не выполняет скриптлет `%verifyscript` (при его наличии);

– `--nosignature` – не проверяет подпись RPM-пакета или заголовка при чтении;

– `--nolinkto; --nomd5; --nosize; --nouser; --nogroup; --nomtime; --nomode;`

– `--nordev` – не проверяет соответствующие атрибуты файлов.

Формат вывода представляет собой строку из 8 символов и маркера из заголовка RPM-пакета, за которыми следует имя файла.

Возможные маркеры атрибутов:

– `c %config` - конфигурационный файл;

– `d %doc` - файл документации;

– `g %ghost` - файл (т.е. содержимое файла, не включено в состав RPM-пакета);

– `l %license` - файл с лицензией;

– `r %readme` - файл `readme`.

Каждый из 8 символов отражает результат проверки атрибутов файлов со значением того же атрибута, записанного в БД. Символ «.» означает, что проверка прошла, а символ «?» означает, что проверка не может быть выполнена (например, права доступа к файлу не позволяют провести чтение). В противном случае будут

отображены символы, выделенные жирным и отображающие проверки соответствующего `--verify` теста:

- `S` – отличается размер (Size) файла;
- `M` – отличается режим (Mode) доступа (включая права доступа и тип файла);
- `5` – отличается контрольная MD5 сумма;
- `D` – отличается старший/младший номер файла устройства (Device);
- `L` – отличается путь ссылки при `readLink(2)`;
- `U` – отличается владелец (User);
- `G` –отличается групповое владение (Group);
- `T` – отличается время изменения (mTime).

ОПЦИИ ПЕРЕСТРОЙКИ БД:

Общая форма команды перестройки БД rpm:

```
rpm {--initdb|--rebuilddb} [-v] [--dbpath DIRECTORY] [--root DIRECTORY]
```

Следует использовать `--initdb` для создания новой БД и `--rebuilddb` для перестройки индексов БД на основании заголовков установленных пакетов:

- `SHOWRC` – команда;
- `rpm --showrc` – показывает значения, которые rpm будет использовать для всех опций, установленных в конфигурационных файлах `rpmrc` и macros.

ОПЦИИ FTP/HTTP:

RPM может выступать в качестве клиента FTP и/или HTTP, таким образом, пакеты могут быть опрошены или установлены из сети Интернет. Пакеты для операций установки, обновления или запроса могут быть указаны в виде адреса `ftp` или `http` URL:

```
ftp://USER:PASSWORD@HOST:PORT/path/to/package.rpm
```

Если параметр: `PASSWORD` пропущен, будет запрошен пароль (один раз для пары пользователь/узел). Если пропущены оба параметра (`user` и `password`), будет выполнено анонимное подключение ftp. Во всех случаях используется пассивный (PASV) режим передачи файлов `ftp`.

RPM допускает следующие опции для адресов ftp URL:

- `--ftpproxy HOST` - узел HOST будет использован как прокси-сервер для всех операций передачи данных `ftp`, что позволяет работать с `ftp` через МЭ, на которых запущена служба прокси. Данная опция может быть также указана при настройке макроса `%_ftpproxy`;
- `--ftpport PORT` – номер TCP порта PORT будет использован для `ftp` подключения к `ftp` прокси-серверу вместо порта по умолчанию. Данная опция может быть также указана при настройке макроса `%_ftpport`.

RPM допускает следующие опции для адресов http URL:

- `--httpproxy HOST` – узел HOST будет использован как прокси-сервер для всех операций передачи данных `http`. Данная опция может быть также указана при настройке макроса `%_httpproxy`;

– `--httpport PORT` – номер TCP порта PORT будет использован для `http` подключения к http прокси-серверу вместо порта по умолчанию. Данная опция может быть также указана при настройке макроса `%_httpport`.

ВОПРОСЫ СОВМЕСТИМОСТИ: выполнение `rpmbuild`.

Режимы сборки вынесены в программу `/usr/bin/rpmbuild`, однако совместимость, обеспечиваемая при помощи приведенных ниже `popt` псевдонимов, не является совершенной, в связи с чем совместимость режимов сборки через псевдонимы `popt` удалена из RPM. Требуется установить пакет `rpmbuild` и просмотреть документацию в `rpmbuild(8)` по всем режимам сборки RPM, ранее приведенную в `rpm(8)`.

В случае необходимости вызывать `rpmbuild` из командной строки `rpm` требуется добавить следующие строки в `/etc/popt`,:

```
rpm exec --bp rpmb -bp
rpm exec --bc rpmb -bc
rpm exec --bi rpmb -bi
rpm exec --bl rpmb -bl
rpm exec --ba rpmb -ba
rpm exec --bb rpmb -bb
rpm exec --bs rpmb -bs
rpm exec --tp rpmb -tp
rpm exec --tc rpmb -tc
rpm exec --ti rpmb -ti
rpm exec --tl rpmb -tl
rpm exec --ta rpmb -ta
rpm exec --tb rpmb -tb
rpm exec --ts rpmb -ts
rpm exec --rebuild rpmb --rebuild
rpm exec --recompile rpmb --recompile
rpm exec --clean rpmb --clean
rpm exec --rmsource rpmb --rmsource
rpm exec --rmspec rpmb --rmspec
rpm exec --target rpmb --target
rpm exec --short-circuit rpmb --short-circuit
```

ФАЙЛЫ: конфигурация `rpmrc`:

```
/usr/lib/rpm/rpmrc
/usr/lib/rpm/redhat/rpmrc
/etc/rpmrc
~/.rpmrc
Конфигурация макросов:
/usr/lib/rpm/macros
/usr/lib/rpm/redhat/macros
/etc/rpm/macros
~/.rpmmacros
```

БД:

```
/var/lib/rpm/Basenames;
/var/lib/rpm/Conflictname;
/var/lib/rpm/Dirnames;
```

```
/var/lib/rpm/Filemd5s;
/var/lib/rpm/Group;
/var/lib/rpm/Installtid;
/var/lib/rpm/Name;
/var/lib/rpm/Packages;
/var/lib/rpm/Providename;
/var/lib/rpm/Provideversion;
/var/lib/rpm/Pubkeys;
/var/lib/rpm/Removed;
/var/lib/rpm/Requirename;
/var/lib/rpm/Requireversion;
/var/lib/rpm/Sha1header;
/var/lib/rpm/Sigmd5;
/var/lib/rpm/Triggername;
```

Временные файлы: `/var/tmp/rpm*`.

ПРИЛОЖЕНИЕ 2

Управление службами с помощью systemd

В ОС Аврора `systemd` используется в качестве системы инициализации по умолчанию, которая разработана с учетом обратной совместимости со сценариями инициализации `SysV` и предлагает следующие возможности:

- параллельный запуск системных служб во время загрузки системы;
- активация демонов по требованию;
- поддержка снимков состояния системы;
- логика управления службами на основе зависимостей.

ИМЯ: `systemd, init` – служба инициализации ОС Аврора.

ОБЗОР:

```
systemd [OPTIONS...]
init [OPTIONS...] {COMMAND}
```

ОПИСАНИЕ: `systemd` представляет собой системный и сервисный менеджер для ОС Аврора. При выполнении в качестве первого процесса загрузки (PID-1) он выступает в роли системы инициализации, которая запускает и поддерживает сервисы пользовательского пространства. Для совместимости с SysV, если `systemd` вызывается как `init` и PID которого не равен 1, он будет выполнять `telinit` и передавать все аргументы командной строки без изменений. Это означает, что `init` и `telinit` в основном эквивалентны, если вызываются из нормального сеанса. При запуске в качестве системной инстанции `systemd` интерпретирует файл конфигурации `system.conf`, в противном случае - `user.conf`.

ОПЦИИ:

- `-h, --help` – печатает короткий текст помощи и выходит;
- `--test` – определяет последовательность запуска, выводит ее и выходит. Данная опция полезна только для отладки;
- `--dump-configuration-items` – выводит краткий, при этом исчерпывающий список элементов конфигурации, используемых в настроечных файлах модулей;
- `--introspect=` – извлекает данные самоанализа D-Bus интерфейса. В основном это полезно во время инсталляции для получения данных, пригодных для репозитория D-Bus интерфейсов. При необходимости имя интерфейса может быть указано для данных самоанализа. Если оно пропущено, данные самоанализа выводятся для всех интерфейсов;
- `--unit=` – устанавливает по умолчанию модуль для активации при запуске. Если не указано, по умолчанию – default.target;

– `--system`, `--user` – направляет сигнал `systemd` запустить системную инстанцию (соответственно, пользовательскую инстанцию), даже если ID процесса не равен 1 (соответственно равен 1), т. е. `systemd` не выполняется (соответственно выполняется) в качестве процесса инициализации. Обычно необходимость задавать эти опции отсутствует, т.к. `systemd` автоматически определяет режим своего запуска. Следовательно, эти варианты мало полезны, кроме случая отладки;

ВНИМАНИЕ! Загрузка и поддержка полной системы с `systemd`, запущенной в режиме `--system` при PID, не равном 1, не обеспечивается. На практике задание опции `--system` явно имеет смысл только в сочетании с опцией `–test`.

– `--dump-core` – дамп ядра при аварийном отказе. Опция не имеет эффекта в случае запуска в пользовательской инстанции;

– `--crash-shell` – запускает `shell` при сбое. Опция не имеет эффекта в случае запуска в пользовательской инстанции;

– `--confirm-spawn` – запрашивает подтверждение при размножении процессов. Опция не имеет эффекта в случае запуска в пользовательской инстанции;

– `--show-status=` – отображает сжатую статусную информацию о сервисах при загрузке. Опция не имеет эффекта при выполнении в пользовательской инстанции. Принимает булевый аргумент, который может быть пропущен, что интерпретируется как `true`;

– `--sysv-console=` – определяет, будет ли вывод SysV скрипта инициализации направлен на консоль. Опция не имеет эффекта при выполнении в пользовательской инстанции. Принимает булевый аргумент, который может быть пропущен, что интерпретируется как true;

– `--log-target=` – задает приемник для сообщений о событиях. Аргумент должен быть одним из следующих: `console`, `syslog`, `kmsg`, `syslog-or-kmsg`, `null`;

– `--log-level=` – задает уровень журналирования. В качестве аргумента принимается числовой уровень журналирования или хорошо известные `syslog(3)` символические имена (в нижнем регистре):`emerg`, `alert`, `crit`, `err`, `warning`, `notice`, `info`, `debug`;

– `--log-color=` – выделяет важные журнальные сообщения. Принимает булевый аргумент, который может быть пропущен, что интерпретируется как `true`;

– `--log-location=` – включает расположения в коде в журнальные сообщения. Это наиболее значимо для целей отладки. Принимает булевый аргумент, который может быть пропущен, что интерпретируется как `true`;

– `--default-standard-output=`, `--default-standard-error=` -задает вывод по умолчанию или вывод ошибок для всех сервисов и сокетов, т. е. управляет значением по умолчанию для `StandardOutput=` и для `StandardExecute`. Воспринимает одно из следующих значений: `inherit`, `null`, `tty`, `syslog`, `syslog+console`, `kmsg`, `kmsg+console`. Если аргумент пропущен, по умолчанию считается `inherit`.

АРХИТЕКТУРА: systemd предусматривает систему зависимостей между разными элементами, называемыми «units» (юниты). Юниты инкапсулируют объекты, важные для загрузки системы и поддержки.

Большинство юнитов настроены в файлах конфигурации юнитов, синтаксис и базовый набор опций которых описан в systemd.unit (5), однако некоторые создаются автоматически из других конфигураций или динамически из системных структур.

Юниты могут быть active (что подразумевает started, bound, plugged in, ... в зависимости от типа устройства), или inactive (что подразумевает stopped, unbound, unplugged, ...), а также находиться в процессе активации или деактивации, т.е. между двумя состояниями (эти состояния называются activating, deactivating). Также возможно специальное состояние failed, схожее с inactive и возникающее, когда сервис потерпел неудачу на некотором пути (процесс возвратил код ошибки на выходе, потерпел крах или тайм-аут операции). При возникновении данного состояния причина регистрируется в журнале для дальнейшего использования.

ВНИМАНИЕ! Различные типы юнитов могут иметь ряд дополнительных подсостояний, отображаемых на 5 обобщенных состояний юнитов.

Доступны следующие типы юнитов:

1) Service units, управляющие демонами и процессами, из которых они состоят. За дополнительной информацией необходимо обращаться к systemd.service(5);

2) Socket units, инкапсулирующие локальные IPC или сетевые сокеты в системе, полезны при активации на основе сокетов. За дополнительной информацией о socket units необходимо обращаться к systemd.socket(5), об активации на основе сокетов и других формах активации – к daemon(7);

3) Target units – полезны для группировки юнитов или обеспечения хорошо известных точек синхронизации в процессе загрузки. За дополнительной информацией необходимо обращаться к systemd.target(5);

4) Device units – представляют МУ ядра в systemd и могут быть использованы для выполнения активации на основе устройств. За дополнительной информацией необходимо обращаться к systemd.device(5);

5) Mount units – управляют точками монтирования в ФС. За дополнительной информацией необходимо обращаться к systemd.mount(5);

6) Automount units – обеспечивают возможности автомонтирования для монтирования ФС по требованию, а также параллелизации загрузки. За дополнительной информацией необходимо обращаться к systemd.automount(5);

7) Snapshot units – могут быть использованы для временного сохранения состояний множества юнитов systemd, которые позднее могут быть восстановлены при активации сохраненного snapshot unit. За дополнительной информацией необходимо обращаться к systemd.snapshot(5);

8) Timer units – используются для запуска активации других юнитов, основываясь на таймерах. За дополнительной информацией необходимо обращаться к systemd.timer(5);

9) Swap units – схожи с mount units, инкапсулируют разделы или файлы подкачки памяти ОС Аврора, описание которых приведено в systemd.swap(5);

10) Path units – могут быть использованы для активации других сервисов при изменении и модификации объектов ФС. За дополнительной информацией необходимо обращаться к systemd.path(5).

Юниты именуются так же, как их конфигурационные файлы. Некоторые юниты имеют собственную особенную семантику.

Systemd известны различные виды зависимостей, в т.ч. зависимости требования положительные и отрицательные (например, Requires= и Conflicts=), а также зависимости очередности (After= и Before=). Зависимости очередности и требования ортогональны. Если между двумя юнитами существует только зависимость требования (например, foo.service требует bar.service), но не зависимость очередности (например, после foo.service bar.service), и оба этих юнита запрошены на запуск, они будут запускаться параллельно. Наличие обеих зависимостей требования и очередности между двумя юнитами является общим шаблоном. Кроме того, большинство зависимостей неявно создается и поддерживается systemd.

Также имеется возможность вручную объявлять дополнительные зависимости, однако это не является обязательным.

Прикладные программы и юниты (через зависимости) могут потребовать изменения состояния юнитов. В systemd эти требования инкапсулируются в «jobs» и поддерживаются в job-очереди. Jobs могут успешно выполниться или потерпеть неудачу, их выполнение упорядочено в соответствии с зависимостью очередности юнитов, на которую они были запланированы.

При загрузке systemd активирует юнит типа target default.target, работа которого заключается в активации загружаемых сервисов и других юнитов, загружаемых с помощью зависимостей. Обычно имя данного юнита представляет собой псевдоним: ссылка на graphical.target (для полнофункциональных загрузок в пользовательский интерфейс) или на multi-user.target (для ограниченной только консолью загрузки при использовании во встроенных или серверных средах или им подобных; подмножество graphical.target).

Администратор имеет возможность сделать его псевдонимом к любому другому юниту типа target. За дополнительной информацией о юнитах типа target необходимо обращаться к systemd.special(7).

Процессы-потомки systemd помещаются в отдельные группы управления ОС Аврора с такими же именами, как у юнитов, к которым они принадлежат, в собственной иерархии systemd, который использует это для эффективного отслеживания процессов. Информация о группе управления поддерживается в ядре и доступна с помощью иерархии ФС (под /sys/fs/cgroup/systemd/) или в таких

инструментах, как ps(1) (ps xawf -eo pid,user,cgroup,args особенно полезна для получения списка всех процессов и юнитов systemd, которым они принадлежат).

Systemd в большой степени совместима с системой SysV init: SysV init скрипты поддерживаются и читаются как файлы конфигурации альтернативного (хотя и ограниченного) формата. Обеспечивается SysV /dev/initctl интерфейс, доступна совместимая реализация различных SysV клиентских инструментов, а также поддерживаются различные устоявшиеся UNIX-функциональности, такие как /etc/fstab или БД utmp.

Systemd имеет минимальную систему транзакций: в случае необходимости запустить или выключить юнит systemd и все его зависимости будут добавлены во временную транзакцию. Далее следует проверка на предмет того, является ли транзакция последовательной (является ли порядок всех юнитов незацикленным):

– если нет, systemd постарается исправить это и удалит из транзакции jobs, не являющиеся необходимыми, что может предотвратить зацикливание. Также systemd пытается подавить в транзакции jobs, способные остановить запущенный сервис и не являющиеся необходимыми. Кроме того проверяется, являются ли jobs из транзакции противоречивыми jobs, которые уже находятся очереди. Если да, транзакция, вероятно, будет прервана;

– если да, транзакция является последовательной и ее влияние сведено к минимуму, она объединяется со всеми уже ожидающими обработки jobs и добавляется в очередь выполнения. Это означает, что до выполнения затребованной операции systemd будет проверять, что операция имеет смысл, исправлять ее, если возможно, и, если она действительно не может работать, отменит ее.

Systemd содержит встроенную реализацию задач, которые должны быть выполнены как часть процесса загрузки. Например, устанавливает имя хоста или конфигурирует петлевое устройство сети, настраивает и монтирует различные ФС API, такие как /sys или /proc.

Для получения дополнительной информации о понятиях и идеях, заложенных в systemd, необходимо обращаться к Original Design Document[2].

ВНИМАНИЕ! Некоторые интерфейсы, предоставляемые systemd, содержат Interface Stability Promise[3].

КАТАЛОГИ ИФБО:

Каталоги системных юнитов

Системный администратор systemd читает конфигурацию юнитов из различных каталогов. Пакеты, желающие установить файлы юнитов, должны разместить их в каталог, возвращаемый командой:

```
pkg-config systemd --variable=systemdsystemunitdir
```

Другими проверяемыми каталогами являются /usr/local/lib/systemd/system и /usr/lib/systemd/system.

Пользовательская конфигурация всегда имеет приоритет. `pkg-config systemd --variable=systemdsystemconfdir` возвращает путь к каталогу системной конфигурации. Пакеты должны изменять содержимое этих каталогов только с помощью команд `enable` и `disable` инструмента `systemctl(1)`.

Каталоги пользовательских юнитов

Подобные правила применяются для каталогов пользовательских юнитов, однако находить юниты в данном случае следует по спецификации XDG Base Directory [4]. МП должны поместить свои файлы юнитов в каталог, возвращенный командой:

```
pkg-config systemd --variable=systemduserunitdir
```

Глобальная конфигурация выполнена в каталоге, о котором сообщает команда `pkg-config systemd --variable=systemduserconfdir`. Команды `enable` и `disable` инструмента `systemctl(1)` могут обращаться с обоими видами юнитов: глобальными (т. е. для всех пользователей) и частными (для одного пользователя), разрешая/запрещая их.

Каталоги скриптов SysV init

Каталоги скриптов `SysV init` располагаются между дистрибутивами. Если `systemd` не может найти родной `unit` файл для требуемого сервиса, он будет осуществлять поиск скрипта `SysV init` того же самого имени (с удаленным суффиксом `.service`).

Каталог механизма ссылок SysV runlevel

Местоположение каталога механизма ссылок `SysV runlevel` различно между дистрибутивами. `Systemd` учитывает механизм ссылок, определяя, должен ли сервис быть включен.

ВНИМАНИЕ! `service unit` со встроенным `unit` конфигурационным файлом не может быть запущен при активизации его в механизме ссылок `SysV runlevel`.

СИГНАЛЫ:

– `SIGTERM` – после получения данного сигнала системный менеджер `systemd` сериализует его состояние, перезапускается и десериализует сохраненное состояние повторно. В основном это эквивалентно следующему: `systemctl daemon-reexec`;

– `systemctl daemon-reexec` – пользовательский менеджер `systemd` запустит `exit.target unit`, когда примет данный сигнал. В основном это эквивалентно: `systemctl --user start exit.target`;

– `SIGINT` – после получения данного сигнала системный менеджер `systemd` запустит юнит `ctrl-alt-del.target`. В основном это эквивалентно: `systemctl start ctl-alt-del.target`;

– `systemctl start ctl-alt-del.target` – пользовательский менеджер `systemd` поступит с данным сигналом тем же способом, что и с сигналом `SIGTERM`;

– `SIGWINCH` – получив данный сигнал, системный менеджер запустит юнит `kbrequest.target`. В основном это эквивалентно: `systemctl start kbrequest.target`;

– systemctl start kbrequest.target – пользовательский менеджер systemd игнорирует данный сигнал;

– SIGPWR – получив данный сигнал, системный менеджер запустит юнит sigpwr.target. В основном это эквивалентно: systemctl start sigpwr.target;

– SIGUSR1 – получив данный сигнал, системный менеджер будет пытаться присоединиться к шине D-Bus;

– SIGUSR2 -получив данный сигнал, системный менеджер целиком запишет в сообщение о событиях свое состояние в читаемой форме. Записанные данные будут такие же, как выводимые командой: systemctl dump;

– SIGHUP – полная перезагрузка конфигурации демона. В основном это эквивалентно: systemctl daemon-reload;

– SIGRTMIN+0 – входит в режим по умолчанию, запускает юнит default.target. В основном это эквивалентно: systemctl start default.target;

– SIGRTMIN+1 – входит в спасательный режим, запускает юнит rescue.target. В основном это эквивалентно: systemctl isolate rescue.target;

– SIGRTMIN+2 – входит в аварийный режим, запускает юнит emergency.service. В основном это эквивалентно: systemctl isolate emergency.service;

– SIGRTMIN+3 – останавливает МУ, запускает юнит halt.target. В основном это эквивалентно: systemctl start halt.target;

– SIGRTMIN+4 – выключение питания МУ, запуск юнита poweroff.target. В основном это эквивалентно systemctl start poweroff.target;

– SIGRTMIN+5 – перезапуск МУ, запуск юнита reboot.target. В основном это эквивалентно: systemctl start reboot.target;

– SIGRTMIN+6 – перезапуск МУ с помощью kexec, запуск юнита kexec.target. В основном это эквивалентно: systemctl start kexec.target;

– SIGRTMIN+13 – немедленная остановка работы МУ;

– SIGRTMIN+14 – немедленное отключение питания МУ;

– SIGRTMIN+15 – немедленная перезагрузка МУ;

– SIGRTMIN+16 – немедленная перезагрузка МУ с kexec;

– SIGRTMIN+20 – делает возможным вывод статусных сообщений на консоль, в т.ч. вызванную параметром systemd.show_status=1 в командной строке ядра;

– SIGRTMIN+21 – отмена возможности вывода статусных сообщений на консоль, в т.ч. вызванную параметром systemd.show_status=0 в командной строке ядра.

ПЕРЕМЕННЫЕ СРЕДЫ И ОКРУЖЕНИЕ:

– $SYSTEMD_LOG_LEVEL – systemd читает log level из этой переменной окружения. Данная настройка может быть переопределена с помощью опции --log-level=;

– $SYSTEMD_LOG_TARGET – systemd читает целевой журнал из этой переменной окружения. Данная настройка может быть переопределена с помощью опции --log-target=;

– $SYSTEMD_LOG_COLOR – определяет, будет ли systemd выделять цветом важные сообщения системного журнала. Данная настройка может быть переопределена с помощью опции --log-color=;

– $SYSTEMD_LOG_LOCATION – определяет, будет ли systemd выводить номер строки в исходном коде вместе с журнальными сообщениями. Это поведение может быть переопределено с помощью опции --log-location=;

– $XDG_CONFIG_HOME, $XDG_CONFIG_DIRS, $XDG_DATA_HOME, $XDG_DATA_DIRS – пользовательский менеджер systemd использует эти переменные в соответствии с XDG Base Directory спецификацией [4], чтобы найти ее конфигурацию;

– $SYSTEMD_UNIT_PATH – задает, где systemd ищет файлы юнитов;

– $SYSTEMD_SYSVINIT_PATH – задает, где systemd ищет скрипты SysV init;

– $SYSTEMD_SYSVRCND_PATH – задает, где systemd ищет скрипты системы ссылок runlevel SysV init.;

– $LISTEN_PID, $LISTEN_FDS – устанавливает systemd для контролируемых процессов во время активации на основе сокетов. Подробнее в sd_listen_fds(3);

– $NOTIFY_SOCKET – устанавливает systemd для контролируемых процессов с целью уведомления о статусе и завершении запуска. Подробнее в sd_notify(3).

ПАРАМЕТРЫ КОНФИГУРАЦИИ, ПЕРЕДАВАЕМЫЕ ОТ ЯДРА ОС АВРОРА:

Если systemd запущен в системной инстанции, он разбирает несколько аргументов командной строки ядра:

– systemd.unit= – переопределяет юнит, который активируется при загрузке – по умолчанию default.target. Это может быть использовано для временной загрузки в другой загрузочный юнит, как, например, rescue.target или emergency.service. Подробнее об этих юнитах в systemd.special(7);

– systemd.dump_core= – принимает булевый аргумент. Если true, то systemd создает core dump, когда работает некорректно. В противном случае core dump не создается. По умолчанию true;

– systemd.crash_shell= – принимает булевый аргумент. Если true, то systemd порождает shell, когда рушится. В противном случае shell не порождается. По умолчанию false из соображений безопасности, т.к. shell не защищен парольной аутентификацией;

– systemd.crash_chvt= – принимает аргумент целое число. Если оно положительное, systemd активирует заданный виртуальный терминал, когда рушится. По умолчанию -1;

– systemd.confirm_spawn= – принимает булевый аргумент. Если true, запрашивается подтверждение, когда порождается процесс. По умолчанию false;

– systemd.show_status= – принимает булевый аргумент. Если true, выводит на консоль в сжатом виде изменения статуса сервисов в течение загрузки. По умолчанию true;

– systemd.sysv_console= – принимает булевый аргумент. Если true, вывод скрипта SysV init будет направлен на консоль. По умолчанию true, но, если quiet передан как опция командной строки ядра, по умолчанию false;

– управляет выводом в сообщение о событиях с тем же эффектом, что и описанные выше переменные окружения: $SYSTEMD_LOG_TARGET, $SYSTEMD_LOG_LEVEL, $SYSTEMD_LOG_COLOR, $SYSTEMD_LOG_LOCATION:

```
systemd.log_target=, systemd.log_level=, systemd.log_color=,
systemd.log_location=
```

– systemd.default_standard_output=,systemd.default_standard_error= – управляет стандартным выводом/выводом ошибки для сервисов по умолчанию с тем же эффектом, что и аргументы командной строки, описанные выше --default-standard-output= и --default-standard-error= соответственно.

СОКЕТЫ И КАНАЛЫ:

– /run/systemd/notify – сокет уведомления состояния демона. Этот сокет AF_UNIX datagram используется для реализации логики уведомления демона, которое осуществляется посредством sd_notify(3);

– /run/systemd/logger – используется внутренне юнитом systemd-logger.service, для systemd вводит понятие системных юнитов. Данные юниты представлены конфигурационными файлами, расположенными в одном или нескольких каталогах, описание которых приведено в таблице (Таблица 2.1), и в сжатом виде аккумулируют информацию о системных службах, слушающих сокетах, сохраненных снимках состояния системы, а также о других объектах, имеющих отношение к системе инициализации.

Таблица 2.1

Каталог	Описание
/usr/lib/systemd/system/	Файлы юнитов systemd, распределенные установленными RPM-пакетами
/run/systemd/system/	Файлы юнитов systemd, созданные в процессе работы. Данный каталог имеет приоритет над каталогом с установленными служебными файлами юнитов
/etc/systemd/system/	Файлы юнитов systemd, созданные командой systemctl enable, а также файлы юнитов, добавленные для расширения службы. Данный каталог имеет приоритет над каталогом с файлами юнитов, созданными в процессе работы

Полный список доступных типов юнитов systemd приведен в таблице (Таблица 2.2).

Таблица 2.2

Тип юнита	Расширение файла	Описание
Служба	.service	Системная служба
Цель	.target	Группа юнитов systemd
Автоматическое монтирование	.automount	Точка автоматического монтирования ФС
Устройство	.device	Файл устройства, распознаваемого ядром
Монтирование	.mount	Точка монтирования ФС
Путь	.path	Файл или каталог в ФС
Область	.scope	Процесс, созданный извне
Срез	.slice	Группа иерархически организованных юнитов, управляющих системными процессами
Снимок	.snapshot	Сохраненное состояние менеджера systemd
Сокет	.socket	Сокет для обмена информацией между процессами
Подкачка	.swap	Устройство или файл подкачки
Таймер	.timer	Таймер systemd

Переопределение параметров systemd по умолчанию осуществляется с помощью файла system.conf.

Конфигурация systemd по умолчанию определяется во время компиляции и доступна для просмотра в файле /etc/systemd/system.conf. При необходимости изменить параметры по умолчанию и глобально переопределить отдельные значения юнитов systemd следует использовать данный файл.

Для переопределения значения предела истечения времени ожидания, по умолчанию равного 90 секундам, необходимо использовать параметр DefaultTimeoutStartSec, для которого потребуется указать значение в секундах:

```
DefaultTimeoutStartSec=<требуемое_значение>
```

Основные предлагаемые возможности

В ОС Аврора система systemd и service manager осуществляют активацию на основе сокетов – во время загрузки systemd создает слушающие сокеты для всех системных служб, поддерживающих данный тип активации, и передает их соответствующим службам, как только они начинают работу. Это позволяет systemd осуществлять параллельный запуск служб и делает возможным перезапуск службы без потери сообщений, отправленных ей в то время, пока она была недоступна: соответствующий сокет остается доступным, и все сообщения помещаются в очередь.

Для активации на основе сокетов systemd использует юниты сокетов:

– активация на основе шины – системные службы, использующие D-Bus для обмена информацией между процессами; можно запустить по требованию, как только

клиентское МП в первый раз попробует обменяться с ними информацией. Для активации на основе шины systemd использует файлы службы D-Bus;

– активация на основе МУ – системные службы, поддерживающие активацию на основе устройств; можно запустить по требованию при подключении определенного типа оборудования или если это оборудование становится доступно. Для активации на основе устройств systemd использует юниты МУ;

– активация на основе путей – системные службы, поддерживающие активацию на основе путей; можно запустить по требованию, когда определенный файл или каталог изменят свой статус. Для активации на основе путей systemd использует юниты пути;

– снимки состояния системы – systemd может временно сохранить текущее состояние всех юнитов или повторно активировать предыдущее состояние системы из динамически созданного снимка. Для хранения текущего состояния системы systemd использует динамически созданные юниты снимков;

– управление точками монтирования и автомонтирования – systemd наблюдает и управляет точками монтирования и автоматического монтирования. Для точек монтирования systemd использует юниты точек монтирования, а для точек автомонтирования – юниты автомонтирования;

– агрессивная параллелизация – используя активацию на основе сокетов, systemd может запускать системные службы параллельно, как только слушающие сокеты окажутся на месте. В сочетании с системными службами, поддерживающими активацию по требованию, параллельная активация значительно сокращает время, необходимое для загрузки системы;

– транзакционная логика активации юнитов – перед активацией или отключением юнита systemd рассчитывает его зависимости, создает временную транзакцию и проверяет ее целостность. Если транзакция будет нецелостной, то перед тем, как выдать сообщение об ошибке, systemd автоматически пытается скорректировать транзакцию и удалить из нее задачи, не имеющие критического значения;

– обратная совместимость с системой инициализации SysV – systemd поддерживает сценарии инициализации SysV, что описано в базовой спецификации проекта Linux Standard Base. Это облегчает переход к служебным юнитам systemd.

Изменения совместимости

Системы systemd и service manager разработаны таким образом, чтобы обеспечивать базовую совместимость с SysV и Upstart. Наиболее значительные изменения в совместимости относительно ОС Аврора и ряда предыдущих дистрибутивов Linux:

– systemd имеет ограниченную поддержку уровней выполнения, предоставляя несколько юнитов цели, которые можно напрямую сопоставить с этими уровнями выполнения. Однако сопоставление возможно не для всех целей systemd, поэтому из соображений совместимости в состав systemd входит команда runlevel, которая

может вернуть N для обозначения неизвестного уровня выполнения. По возможности рекомендуется по избегать использования данной команды;

– утилита `systemctl` не поддерживает произвольные команды. В дополнение к стандартным командам, таким, как `start`, `stop` и `status`, авторы сценариев инициализации `SysV` реализовали поддержку для любого числа произвольных команд в качестве дополнительного функционала. Например, в Red Hat Enterprise Linux 6 сценарий `init` для `iptables` можно было выполнять внутри команды `panic`, что включало режим panic и перенастраивало систему на немедленный сброс всех входящих и исходящих пакетов. Ничего из этого не поддерживается в `systemd`, и `systemctl` принимает только команды, перечисленные в документации;

– утилита `systemctl` не обменивается информацией со службами, которые не были запущены `systemd`. Когда `systemd` запускает системную службу, идентификатор ее главного процесса сохраняется для возможности его отслеживания. Затем утилита `systemctl` использует этот идентификатор для опроса и управления службой. Соответственно, если пользователь запускает определенный демон напрямую из командной строки, `systemctl` не в состоянии определить его текущий статус или остановить его выполнение;

– `systemd` останавливает только работающие службы. После инициации последовательности `shutdown` Red Hat Enterprise Linux 6 и более ранние релизы системы использовали символьные ссылки, расположенные в каталоге `/etc/rc0.d/`, для остановки всех доступных системных служб вне зависимости от их статуса. С `systemd` во время выполнения последовательности `shutdown` останавливаются только работающие службы;

– системные службы не могут читать информацию из стандартного потока `input`. Когда `systemd` запускает службу, стандартный ввод службы подключается к `/dev/null` для предотвращения любого взаимодействия с пользователем;

– системные службы не наследуют контекст (такой, как переменные окружения `HOME` и `PATH`) от вызывающего их пользователя или его сеанса. Каждая служба работает в чистом контексте выполнения;

– во время загрузки сценария инициализации `SysV` `systemd` читает информацию о зависимостях, закодированную в заголовке Linux Standard Base, и интерпретирует ее во время процесса работы;

– для предотвращения зависания системы по вине неправильно работающей службы все операции с юнитами служб подпадают под действие лимита истечения времени ожидания, равного 5 минутам. Это значение строго настроено для всех служб, создаваемых сценариями инициации, и не может быть изменено. Однако для увеличения предела истечения времени ожидания на каждую службу можно использовать индивидуальные конфигурационные файлы.

Управление системными службами

В версиях ОС Linux, в которых применялись `SysV init` или `Upstart`, используемые сценарии инициализации хранились в каталоге `/etc/rc.d/init.d/`.

Данные сценарии были написаны на языке Bash и предоставляли системным администраторам возможность управления состоянием служб или демонов в системе. В ОС Аврора эти сценарии инициализации были заменены на юниты служб, которые имеют расширение `.service` и служат для тех же целей, что и сценарии инициализации. Для просмотра, запуска, остановки, перезапуска, включения и отключения системных служб используется команда `systemctl`.

Команды `service` и `chkconfig` также доступны в системе и работают, как ожидается, однако включены только из соображений совместимости, и их использования следует избегать.

Файлы конфигураций из каталога `/etc/systemd/system/` имеют больший приоритет, чем юниты из каталога `/usr/lib/systemd/system/`. Поэтому, если конфигурационные файлы содержат параметр, который можно указать только один раз, такой, как Description или ExecStart, то изначальное значение этого параметра будет предопределено.

ВНИМАНИЕ! В выводе команды `systemd-delta` переопределенные юниты всегда помечены как [EXTENDED], даже если в целом некоторые параметры фактически переопределяются.

Определение службы можно расширить, что описывается на странице руководства `systemd.unit(5)`. Используя в качестве примера `tftp.service`, необходимо создать новый подкаталог `tftp.service.d` в каталоге `/etc/systemd/system`, а затем в этом подкаталоге создать файл `conf`, расширяющий (или переопределяющий) параметры службы.

Количество открытых дескрипторов файлов ограничено числом 500.000:

```
# mkdir -p /etc/systemd/system/tftp.service.d/
# cat >/etc/systemd/system/tftp.service.d/filelimit.conf <<EOF
[Service]
LimitNOFILE=500000
EOF
```

Изменения применяются после перезагрузки конфигурации демона и перезапуска службы:

```
# systemctl daemon-reload
# systemctl restart tftp.service
```

Команды `systemd-delta` и `systemctl status tftp.service` показывают, что определение службы было добавлено:

```
# systemd-delta --type=extended
[EXTENDED]
/usr/lib/systemd/system/tftp.service → /etc/systemd/system
/tftp.service.d/filelimit.conf
1 overridden configuration file found.
# systemctl status tftp.service
 tftp.service - Tftp Server
Loaded: loaded (/usr/lib/systemd/system/tftp.service; indirect; vendor
```

```
preset: disabled)
Drop-In: /etc/systemd/system/tftp.service.d
 └──filelimit.conf
...
```

Возможные лимиты описаны в следующих разделах страницы руководства `systemd.exec(5)`:

```
LimitCPU=, LimitFSIZE=, LimitDATA=, LimitSTACK=, LimitCORE=,
LimitRSS=,
LimitNOFILE=, LimitAS=, LimitNPROC=, LimitMEMLOCK=, LimitLOCKS=,
LimitSIGPENDING=, LimitMSGQUEUE=, LimitNICE=, LimitRTPRIO=,
LimitRTTIME=

These settings control various resource limits for executed
processes. See setrlimit(2) for details. Use the string infinity
to
configure no limit on a specific resource.
```

ПРИЧИНА: лимиты, указанные в файлах `/etc/security/limits.conf` или `/etc/security/limits.d/*.conf`, настраиваются PAM в начале сеанса регистрации, что указано в следующей записи файла `/etc/pam.d/system-auth-ac` :

```
session        required       pam_limits.so
```

ПРИМЕЧАНИЕ. Демоны, запускаемые `systemd`, не используют сеанс регистрации PAM лимиты которых можно настраивать только в файле юнита службы.

ПРИЛОЖЕНИЕ 3

Изменение лимитов

Описание интерфейса nice

ИМЯ: `nice` – запускает программу с заданием приоритета.

ОБЗОР: `nice` [ПАРАМЕТР] [КОМАНДА [АРГУМЕНТ]...]

ОПИСАНИЕ: запускает КОМАНДУ с указанием приоритета ее выполнения. Без указания КОМАНДА выдает текущий приоритет работы. `ADJUST` по умолчанию равен 10. Диапазон приоритетов расположен от -20 (наивысший) до 19 (наименьший):

- `-ADJUST` – увеличивает приоритет на `ADJUST`;
- `-n, --adjustment=ADJUST` – то же, что и `-ADJUST`;
- `--help` – выдает информацию и завершает работу;
- `--version` – выдает информацию о версии и завершает работу.

Описание интерфейса renice

ИМЯ: `renice` – изменение приоритета выполняющихся процессов.

ОБЗОР:

```
renice priority pid...
renice priority [-p pid... ] [-g pgid...] [-u username]
```

ОПИСАНИЕ: команда `renice` используется для изменения приоритетов выполняющихся процессов. Новое значение приоритета задается числовой константой priority, которая может принимать значения в диапазоне от -20 до +20. Непривилегированные пользователи могут установить значение приоритета только в пределах 0-20, в то время как для системного администратора доступен весь диапазон. Наиболее типичными значениями являются следующие: 19 – процессы выполняются только в случае отсутствия у процессора спешной работы; 0 – базовый приоритет равноправных процессов, выполняющихся под управлением диспетчера; отрицательное значение – для выполнения наиболее спешных и безотлагательных работ.

Описание интерфейса kill

ИМЯ: `kill` – посылает сигнал процессу или выводит список допустимых сигналов.

ОБЗОР:

```
kill [-s СИГНАЛ | -СИГНАЛ] PID...
kill -l [СИГНАЛ]...
kill -t [СИГНАЛ]...
```

ОПИСАНИЕ: посылает сигнал процессу или выводит список допустимых сигналов.

Аргументы, обязательные для полных вариантов опций, являются обязательными также и для кратких вариантов:

- `-s, --signal=СИГНАЛ, -СИГНАЛ` – имя или номер посылаемого сигнала;

– `-l, --list` – вывести имена сигналов или вывести имя сигнала, соответствующее номеру, и наоборот;

– `-t, --table` – вывести информацию о сигналах в виде таблицы;

– `--help` – вывести справку и завершить работу;

– `--version` – вывести информацию о версии и завершить работу.

СИГНАЛ: может указываться в виде имени (например, «HUP») или номера (например, «1»). Также в качестве сигнала можно указывать код выхода, который программа должна сообщить системе при завершении.

PID – числовой идентификатор процесса. Если число отрицательное, оно определяет группу процесса.

Описание интерфейса /etc/security/limits.conf

ИМЯ: файл ограничения ресурсов.

ФОРМАТ: группа/пользователь лимит (жесткий/мягкий) параметр значение.

ОПИСАНИЕ ПАРАМЕТРОВ:

– `core` – размер `core` файлов (КБ);

– `data` – максимальный размер данных (КБ);

– `fsize` – максимальный размер файла (КБ);

– `memlock` – максимальное заблокированное адресное пространство (КБ);

– `nofile` – максимальное количество открытых файлов;

– `rss` – максимальный размер памяти для резидент-программ (КБ);

– `stack` – максимальный размер стека (КБ);

– `cpu` – максимальное процессорное время (MIN);

– `nproc` – максимальное количество процессов;

– `as` – ограничение адресного пространства (КБ);

– `maxlogins` – максимальное число одновременных регистраций в системе;

– `maxsyslogins` – максимальное количество учетных записей;

– `priority` – приоритет запущенных процессов;

– `locks` – максимальное количество файлов, блокируемых пользователем;

– `sigpending` – максимальное количество сигналов, которые можно передать процессу;

– `msgqueue` – максимальный размер памяти для очереди POSIX сообщений (`bytes`);

– `nice` – максимальный приоритет, который можно выставить: [-20, 19];

– `rtprio` – максимальный приоритет времени выполнения.

ПРИЛОЖЕНИЕ 4

Описание интерфейса wipe

ИСПОЛЬЗОВАНИЕ: wipe [опции] файлы.

ОПЦИИ:

– `-a` – прерывает при ошибке;

– `-b <buffer-size-lg2>` – устанавливает размер индивидуального буфера ввода/вывода, указав его логарифм по основанию 2. Могут быть выделены до 30 этих буферов;

– `-c` – совершает `chmod()` на защищенных от записи файлах;

– `-D` – следует символическим ссылкам (конфликтует с `-r`);

– `-e` – использует точный размер файла: не округляет размер файла для стирания возможного мусора, остающегося на последнем блоке;

– `-f` – форсирует, т.е. не запрашивает подтверждения;

– `-F` –не стирает имена файлов;

– `-h` – показывает справку;

– `-i` – информативный (вербальный) режим;

– `-k` – сохраняет файлы, т. е. после перезаписи файлы не удаляются;

– `-l <длина>` – устанавливает длину стирания на `<длину>` байтов, где `<длина>` это целое число, за которым следует `K (Kilo:1024)`, `M (Mega:K^2)` или `G (Giga:K^3)`;

– `-M (l|r)` – устанавливает алгоритм `PRNG` для заполнения блоков (и порядка проходов);

– `l` – использует вызов библиотеки `random()`;

– `a` – использует алгоритм шифрования `arcfour`;

– `-o <сдвиг>` – устанавливает сдвиг очистки на `<сдвиг>`, где `<сдвиг>` имеет тот же формат, что и `<длина>`;

– `-P <проходы>` – устанавливает количество проходов для очистки имени файла. По умолчанию это 1;

– `-Q <количество>` – устанавливает количество проходов для быстрой очистки;

– `-q` – быстрая очистка, менее безопасная, по умолчанию 4 случайных прохода;

– `-r` – рекурсия по каталогам, переход по символическим ссылкам осуществляться не будет;

– `-R` –– устанавливает устройство рандомизации (или команду сидов рандомизации `-S c`);

– `-S` – (`r|c|p`) – метод рандомизации сидов;

– `r` – считывает с устройства рандомизации (надежно);

– `c` – считывает из вывода команды рандомизации сидов;

– `p` – использует `pid()`, `clock()` и т.д. (самый слабый вариант);

– `-s` – тихий режим – подавлять весь вывод;

– `-T <попытки>` – устанавливает максимальное число попыток для свободного поиска имени файла; по умолчанию это 10;

– `-v` – отображает информацию о версии;

– `-Z` – не стирает имя файла;

– `-X <число>` – пропускает число проходов (полезно для продолжения операции очистки);

– `-x <pass1,pass2,...>` – задает очередь проходов;

Руководство по wipe

ИМЯ: `wipe` – безопасное стирание файлов.

ОБЗОР: `wipe [опции] path1 path2 ... pathn`

ОПИСАНИЕ: `wipe` несколько раз перезаписывает специальные паттерны на удаляемый файл, используя вызов `fsync()` и/или `O_SYNC` бит для принудительного доступа к диску. В нормальном режиме используются 34 образца (из которых 8 являются рандомными). Нормальный режим делает 35 проходов (0-34). Быстрый режим позволяет использовать только 4 прохода с рандомными паттернами, что намного менее безопасно.

Журналируемые ФС, такие как Ext3, Ext4 или ReiserFS, используются по умолчанию. На них отсутствуют программы удаления, которые могут надежно зачистить файлы, поскольку чувствительные данные и метаданные могут быть записаны в журнал, к которому сложно получить доступ.

Стирание на NFC или на журналируемых ФС (ReiserFS и т.д.) не будет работать, поэтому рекомендуется вызывать `wipe` напрямую на соответствующее блочное МУ с соответствующими опциями.

ВНИМАНИЕ! Выполнять вызов `wipe` напрямую на соответствующее блочное МУ с соответствующими опциями рекомендуется только в случае крайней необходимости.

Далее следует задать правильные опции, в частности: не рекомендуется стирать целый жесткий диск (`wipe -kD /dev/hda`), т.к. это уничтожит главную загрузочную запись. Предпочтительна очистка разделов (`wipe -kD /dev/hda2`) при условии создания резервных копий всех данных.

ОПЦИИ КОМАНДНОЙ СТРОКИ:

– `f` (форсированно; отключить запрос подтверждения) – по умолчанию `wipe` запрашивает подтверждение с указанием количества регулярных и специальных файлов и директорий, указанных в командной строке. Для подтверждения необходимо ввести «yes», для отмены – «no». Предусмотрена возможность отключить запрос подтверждения опцией `-f`;

– r (рекурсивно в поддиректориях) – позволяет удалить все дерево каталогов. Переход по символическим ссылкам не осуществляется;

– c (chmod если необходим) – если на файл или директорию для стирания не установлены права записи, будет сделан chmod для установки разрешений;

– i (информационный, вербальный режим) – для включения вывода отчетов в стандартный вывод (stdout). По умолчанию все данные записываются в стандартный вывод ошибок (stderr);

– s (тихий режим) – подавляются все сообщения, кроме запросов подтверждения и сообщений ошибок;

– q (быстрое стирание) – в случае использования данной стадии wipe будет делать (по умолчанию) только 4 прохода, записывая случайные данные, на каждый файл;

– Q <количество-проходов> – устанавливает количество проходов для быстрой очистки, по умолчанию 4. Данная опция требует -q;

– a (остановить при ошибке) – программа выйдет с EXIT_FAILURE при возникновении нефатальной ошибки;

– R (установить устройство генерации случайных чисел или команду сидов рандомных данных) – данной опцией, которая требует аргумента, возможно указать альтернативу устройству /dev/random или команду, стандартный вывод которой будет хеширован с использованием MD5-хеша. Это различие может быть сделано опцией -S;

– S (метод случайных сидов) – данная опция принимает односимвольный аргумент, определяющий, как используется устройство рандомизации/аргумент сидов рандомизации. Устройством рандомизации по умолчанию является /dev/random. Его можно установить, используя опцию -R.

Односимвольными аргументами могут являться:

– r – если необходимо, чтобы аргумент интерпретировался как обычное файловое/символьное устройство. Будет работать с /dev/random, также должен работать с FIFO и подобным;

– c – если необходимо, чтобы аргумент выполнялся как команда. Вывод из команды будет хеширован с использованием алгоритма MD5 для обеспечения требуемого сида;

– p – если необходимо, чтобы wipe получала сиды хешированием переменных окружения, текущей даты и времени, ID процессов и т.д. (аргумент устройства рандомизации не используется). Это наименее безопасно;

– M (выбрать алгоритм генерации псевдослучайных чисел) – во время случайных проходов wipe перезаписывает целевые файлы потоком бинарных данных, созданных следующими алгоритмами по выбору;

– `l` – будет использовать (в зависимости от системы пользователя) псевдорандомную библиотеку генератора `random()` или `rand()`. Следует обратить внимание, что на большинстве систем `rand()` представляет собой линейный конгруэнтный генератор, который является крайне слабым. Выбор делается во время компилирования и определяется `HAVE_RANDOM`;

– `a` – будет использовать поток шифра Arcfour как PRNG. Arcfour совместим с шифром RC4. Это означает, что при том же ключе Arcfour является в точности таким же потоком, как RC4.

– `r` – будет использовать свежий алгоритм RC6 как PRNG; RC6 отпирается 128-битными сидами, затем нулевой блок многократно зашифровывается для получения псевдослучайного потока. Это должно быть относительно безопасно. RC6 с 20 кругами медленнее, чем `random()`, опция компилирования WEAK_RC6 позволяет использовать более быструю 4-круговую версию RC6. Для получения возможности использовать RC6 `wipe` должен быть скомпилирован с указанным `ENABLE_RCK`; В `Makefile` приведены предупреждения по патентным вопросам.

Во всех случаях PRNG распространяется с данными, собранными из устройства рандомизации:

– `l <длина>` – ввиду возможного наличия некоторых проблем в определении действительного размера блочного устройства (т.к. некоторые устройства не имеют фиксированного размера, например, дискеты или кассеты) может потребоваться указать размер устройства вручную. <длина> – это емкость МУ, выраженная в числе байт. Можно использовать K (кило) для указания умножения на 1024, M (мега) для выражения умножения на 1048576, G (гига) для умножения на 1073741824 и b (блок) для умножения на 512:

```
1024 = 2b = 1K
20K33 = 20480+33 = 20513
114M32K = 114*1024*1024+32*1024.
```

– `<сдвиг>` – позволяет указать сдвиг внутри стираемого файла или устройства. Синтаксис <сдвига> такой же, как и для опции `-l`;

– `e` – использует точный размер файла: не округлять размер файла для стирания оставшегося мусора в последнем блоке;

– `Z` – не стирает размеры файлов повторным уменьшением вдвое файлового размера. Данные попытки предпринимаются только на обычных файлах, т. е. опция бесполезна при использовании `wipe` для очистки блочного или специального устройства;

– `X <число>` – пропускает заданное число проходов. Полезно для продолжения очистки с заданной точки, например, когда при очистке большого диска необходимо прервать операцию. Используется с `-x`;

– x <pass1>,…,<pass35> – указывает порядок проходов. Когда wipe прерывается, она будет печатать текущую, выбранную случайным образом, пермутацию порядка прохода и номера прохода в качестве соответствующих аргументов -x и -X;

– F – не стирает имена файлов. Обычно wipe пытается скрыть имена файлов, переименовывая их; это не гарантирует, что физическое расположение, содержащее старые имена файлов, будет перезаписано. После переименования файла единственным способом убедиться, что имя изменено, является физическое осуществление вызова sync (), который вымывает дисковые кэши ФС, в то время как для добавления и записи кэша может использоваться бит O_SYNC для синхронизации ввода/вывода для одного файла. Т.к. sync () медленный, вызов sync () после каждого переименования делает очистку имен файлов также медленной;

– k – сохраняет файлы: не удаляет файлы после их перезаписи. Полезно при необходимости стереть устройство, сохранив при этом специальный файл устройства. Это подразумевает -F;

– D – следует символическим ссылкам: по умолчанию wipe никогда не следует символическим ссылкам. Однако, если был указан -D, wipe согласится на стирание целей, на которые указывают символические ссылки. Невозможно одновременно указать опции -D и -r (рекурсия);

– v – показывает информацию о версии и выход;

– h – показывает справку.

ФАЙЛЫ: по умолчанию используется /dev/random в качестве сида (источника) генератора псевдослучайных чисел.

ПЕРЕМЕННЫЕ ОКРУЖЕНИЯ

Если установлена WIPE_SEEDPIPE, wipe будет выполнять указанную в ней команду (используя popen()), хешировать вывод команды с алгоритмом MD5 message-digest для получения 128-битного сида для PRNG. Например, на системах с отсутствующем устройством /dev/random эта переменная должна быть установлена в /etc/profile в сценарий оболочки, содержащий различные команды, такие как ls, ps, who, last и т.д., которые запускаются асинхронно, чтобы получить вывод насколько возможно менее предсказуемым.

Примеры запуска wipe

Следующая команда рекурсивно (-r) удалит все в директории private, при этом включено принудительное удаление и отключен запрос подтверждения (-f), показан прогресс процесса удаления (-i):

```
wipe -rfi private/*
```

Удалить каждый файл и каждую директорию (опция -r) в директории /home/berke/plaintext/, а также саму директорию /home/berke/plaintext/.

Обычные файлы будут удалены в 34 прохода, и их размер будет уменьшаться вдвое случайное количество раз. Специальных файлов (символьных и блочных устройств, FIFO) не будет. Все элементы директории (файлы, специальные файлы и директории) будут переименованы 10 раз и затем удалены. Элементы с неподходящими разрешениями будут выполнять `chmod()` (опция `-c`). Все указанные операции будут происходить без подтверждения пользователя (опция `-f`):

```
wipe -rcf /home/berke/plaintext/
```

Блочное устройство `/dev/hda3` соответствует 3 разделу главного диска на первичном IDE интерфейсе, которое будет удалено в быстром режиме (опция `-q`), т.е. 4 случайными проходами. Индексные дескрипторы не будут переименовываться или удаляться (опция `-k`). Перед запуском программа потребует ввести «yes»:

```
wipe -kq /dev/hda3
```

Поскольку `wipe` никогда не следует по символьным ссылкам, если нет явного указания, при необходимости удалить `/dev/floppy`, который может оказаться символьной ссылкой `/dev/fd0u1440`, потребуется указать опцию -D. Перед запуском программа потребует ввести «yes»:

```
wipe -kqD /dev/floppy
```

В данном случае `wipe` рекурсивно (опция `-r`) уничтожит все в `/var/log`, кроме `/var/log`. Программа не будет пытаться выполнить `chmod()`. Она будет вербальной (опция `-i`) и не потребует ввести «yes» в результате опции `-f`:

```
wipe -rfi >wipe.log /var/log/*
```

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

№ версии документа	Описание изменения	ФИО	Дата
1.0	Начальная версия	Аносов С. Белянкина Н Андронова К.	17.05.2023 г.