БЕЗОПАСНОСТЬ

операционной системы специального назначения

Astra Linux Special Edition

версия 1.6

СДЕЛАНО В РОССИИ

Горячая линия–Телеком

П. В. Буренин, П. Н. Девянин, Е. В. Лебеденко, В. Г. Проскурин, А. Н. Цибуля

БЕЗОПАСНОСТЬ

операционной системы специального назначения

Astra Linux Special Edition

Под редакцией доктора технических наук, профессора П. Н. Девянина

3-е издание, переработанное и дополненное

Рекомендовано Федеральным учебно-методическим объединением в системе высшего образования по УГСНП «Информационная безопасность» в качестве учебного пособия для студентов, обучающихся по направлениям подготовки и специальностям УГСНП «Информационная безопасность»

Москва Горячая линия Телеком 2019

Обеспечение безопасности операционных систем семейства Linux

1.1. Понятие защищённой (доверенной) операционной системы

Существуют два основных подхода к трактовке понятия защищённой автоматизированной системы (АС), применимых к операционным системам (ОС). Первый подход подразумевает, что защищённость ОС обеспечивается при реализации некоторых заданных изначально (например, в соответствующих стандартах или руководящих документах [9, 61, 83]) требований по безопасности, включая наличие некоторого набора механизмов защиты, проверку отсутствия предопределённого перечня уязвимостей и т. п. При втором подходе рассматривается возможность использования ОС в составе АС, которые считаются их владельцами (пользователями) критическими, в связи с чем в ОС должен обеспечиваться комплекс средств защиты информации (СЗИ), адекватный угрозам безопасности именно этих систем. На первый взгляд, эти два подхода не противоречат друг другу, так как вряд ли в критических АС будут применяться решения, не реализующие хотя бы минимальные требования по безопасности. Вместе с тем ОС при выполнении всех требований по безопасности может быть контролируема извне, например её разработчиком. В указанных условиях второй подход подразумевает, что под защищёнными подразумеваются решения, которые в англоязычной литературе обозначаются термином trusted, или, в отечественной интерпретации — доверенные.

Таким образом, защищённой (доверенной) целесообразно считать ОС, которая не только реализует заданные априорно требования безопасности, но и адекватна угрозам безопасности, специфичным для отечественных АС, в том числе для которой отсутствует возможность несанкционированного влияния на её работу извне, при этом владелец (пользователь) защищённой ОС должен иметь однозначное представление об алгоритме функционирования её защитных механизмов во всех режимах работы

Это требование становится все более актуальным в последние годы. Автоматическое обновление, автоматическое оповещение разработчиков о программных ошибках, разнообразные онлайн-сервисы существенно повышают потребительские качества прикладного и системного программного обеспечения ОС, но, с другой стороны, создают все больше возможностей для производителей ПО, в том числе ОС, контролировать действия пользователей [98, 126]. Для целого ряда применений защищённых ОС вопрос доверия к её разработчику оказывается более значимым, чем вопрос об объёме и качестве реализации в данной ОС стандартных механизмов обеспечения безопасности.

Именно этими соображениями можно объяснить рост интереса к отечественным защищённым ОС, отмечающийся в последнее время в Российской Федерации со стороны органов государственной власти и предприятий промышленности. Дополнительным

стимулом по данному направлению стало принятое Правительством Российской Федерации решение об установлении запрета на допуск ПО, происходящего из иностранных государств, для целей осуществления закупок для обеспечения государственных и муниципальных нужд [50], планируемое «законодательное закрепление норм, обеспечивающих преференции для компьютерного, серверного и телекоммуникационного оборудования и ПО отечественного производства при осуществлении закупок для государственных и муниципальных нужд, а также при предоставлении различных форм государственной поддержки» [88]. Высказываются прогнозы, что в случае утверждения и принятия Программы развития российского сегмента сети Интернет «к 2025 году все государственные учреждения и стратегические предприятия будут оснащены компьютерами на российской элементной базе с отечественной операционной системой на борту» [62].

В современных условиях перспективная отечественная защищённая ОС должна отвечать следующим требованиям:

• соответствовать требованиям обеспечения технологической независимости (импортозамещения) Российской Федерации в важнейших областях информатизации, телекоммуникации и связи;

• быть пригодной к функционированию в компьютерных сетях, как изолированных, так и подключённых к сети Интернет (или иным телекоммуникационным сетям), в том числе ориентированных на обработку информации, отнесённой к государственной тайне, или персональных данных;

• реализовывать современные механизмы обеспечения информационной безопасности, учитывающие возможность обработки в данной ОС информации, отнесённой к государственной тайне, как с точки зрения удовлетворения формальных требований соответствующих нормативных документов и стандартов, так и с точки зрения обеспечения реальной защиты от актуальных угроз безопасности.

Разработчик такой ОС должен обладать развитой инфраструктурой разработки и сопровождения её прикладного и системного ПО. Должны быть реализованы механизмы, обеспечивающие доверие к разработчику ОС, возможность научного обоснования (хотя бы неформального) безопасности реализуемых в ней программно-технических решений.

1.2. Обзор защищённых операционных систем семейства Linux

Принято считать, что ОС *Linux* (точнее, *GNU/Lunix*) [41] была создана в 1991 г. 21-летним финским программистом Линусом Торвальдсом. Фактически он переписал с нуля ядро ОС *Minix*, ничем не примечательного «клона» ОС семейства *UNIX*.Долгое время единственным преимуществом ОС *Linux* по сравнению с другими *UNIX*-системами была

лицензионная чистота программного кода ОС *Linux*, позволяющая разворачивать на её базе самые разнообразные информационные системы, не беспокоясь о потенциальных сложностях с лицензиями. Примерно до 2000 г. ОС *Linux* ничем не выделялась среди других ОС семейства *UNIX*, заметно уступая многим из них по производительности и надёжности.

Долгое время открытость программного кода ОС *Linux* оставалась единственным фактором, делавшим данную ОС привлекательной для разработчиков прикладного ПО. Однако со временем разнообразие ПО, адаптированного под ОС *Linux*, достигло некоторой «критической массы», и ситуация изменилась. По мере того как ОС *Linux* становилась де-факто стандартом в мире ОС семейства *UNIX* ,все больше программистов разрабатывали прикладное и системное ПО, предназначенное для работы под управлением ОС *Linux*, а компоненты ОС подвергались все более тщательному тестированию и оптимизации. Начиная с какого-то момента, нарастание популяр-
ности ОС *Linux* стало самоподдерживающимся процессом, и в настоящее время эти ОС составляют более 90% ОС семейства*UNIX*.*Linux*-системы лидируют на рынках смартфонов, самых мощных суперкомпьютеров, занимают примерно половину рынка встраиваемых систем, имеют значительную долю рынка нетбуков [40].

При сравнении ОС семейства *Linux* с более популярными ОС семейств *Microsoft Windows* или *Mac OS* бросается в глаза характерная особенность *Linux*-систем — они первоначально в большей мере были ориентированы на профессиональных высококвалифицированных пользователей. Заметная доля действий, необходимых для настройки системы, а в некоторых случаях и для приведения её в работоспособное состояние, выполняется путём ручного редактирования конфигурационных файлов, редактирования или написания с нуля различных скриптов и т.д. По этой причине ранние версии ОС семейства *Linux* были практически недоступны массовому пользователю, сейчас этот недостаток в основном преодолён современные версии ОС семейства *Linux* требуют лишь минимального обучения. Часто пользователи даже не знают, что работают именно с ОС этого семейства, так, например, популярная мобильная ОС *Android* фактически представляет собой пакет системного ПО, развёрнутого на платформе ОС семейства *Linux*.

Среди многих пользователей ОС семейства *Linux* бытует мнение , что данная ОС отличатся необычно мощно и практически мощной и практически неуязвимой подсистемой защиты. В качестве аргументов в поддержку этой точки зрения обычно приводят наличие в них базовых механизмов дискреционного управления доступом, аутентификации и аудита, которые аналогичны или даже уступают соответствующим механизмам других ОС. Также встречается аргументация, что практически полное отсутствие вредоносного ПО для ОС семейства *Linux* является следствием того, что эта система значительно лучше защищена против вирусных атак, чем, например, ОС семейства *Microsoft Windows*. Это совершенно

неверно. Действительно, вирусы для ОС семейства *Linux* практически не встречаются «в дикой природе» хакерского сообщества, но это обусловлено отнюдь не высокой защищённостью этих ОС. Программисту средней квалификации несложно убедиться, что написание компьютерного вируса или иной вредоносной программы для ОС семейства *Linux* ничуть не сложнее (за исключением некоторых узких классов вредоносных программ), чем написание аналогичной программы, например, для ОС семейства *Microsoft Windows* . Незначительное число *Linux*-вирусов объясняется в основном тем, что разработчики вредоносного ПО предпочитают ориентироваться на более популярные программные платформы, для которых нелегальная деятельность киберпреступников приносит наибольший доход. . Вазовые средства безопасности ОС семейства *Linux* унаследованы от ранних версий ОС семейства *UNIX*, разрабатывавшихся в начале 70-х голов прошлого столетия. Исходя из требований обратной совместимости со старыми версиями, ОС семейства *Linux* продолжают поддерживать целый ряд устаревших защитных механизмов и концепций. В частности, в подсистемах защиты большинства этих ОС присутствуют следующие «анахронизмы»:

- все объекты (сущности) доступа должны интерпретироваться как файловые объекты, атрибуты защиты других типов объектов не могут корректно описываться штатными средствами ОС;
- не поддерживаются глобальные уникальные идентификаторы учётных записей пользователей, все идентификаторы пользователей и групп уникальны только в пределах одного экземпляра ОС;
- набор прав доступа субъектов (процессов) к сущностям (файлам, каталогам) сильно ограничен, поддерживаются только три права доступа: чтение, запись и выполнение, а также задаётся владелец каждой сущности;
- полномочия суперпользователя тооф практически неограничены;
- отсутствуют механизмы автоматического назначения атрибутов защиты вновь создаваемым сущностям на основе атрибутов защиты контейнеров (каталогов), где эти сущности создаются;
- для динамического изменения полномочий субъектов доступа применяется неудобный и потенциально опасный механизм *SUID/SGID*;
- не поддерживаются механизмы олицетворения субъектов доступа, осуществляющих клиентский доступ к процессу-серверу;
- поддерживаемые средства минимизации полномочий пользователей крайне примитивны;
- не поддерживается мандатный контроль целостности;
- не поддерживается изолированная программная среда, даже частично.

Отдельно стоит отметить проблемы безопасности графического интерфейса *X Windows System*, используемого в современных версиях ОС семейства *Linux* для взаимодействия с процессами пользователей. Энтузиасты ОС семейства *Linux* любят критиковать ОС семейства *Microsoft Windows* за недостаточную защищённость её графической подсистемы, например: «В ОС *Microsoft Windows NT* любой процесс независимо от уровня своих привилегий может послать сообщение окну другого процесса (в том числе и более привилегированного!), причём нет никакой возможности установить от отправителя сообщения!... Находим окно какого-нибудь привилегированного приложения (а такая возможность у нас есть), получаем дескриптор интересующего нас элемента управления (кнопки, пункта меню, строки редактирования) и... эмулируем ввод пользователя!!! Привилегированный процесс все сделает за нас, так ничего при этом и не заподозрив!» [28].

На самом деле указанная проблема характерна не только для ОС семейства *Microsoft Windows* . Ещё в 1994 г. Р. Браатен в нашумевшем в своё время сообщении в конференции *comp.security.unix* [106] сформулировал угрозу похищения графическим приложением *X Windows System* учет конфиденциальной информации, адресованной другому графическому приложению. В ОС семейства*Microsoft Windows* , начиная с ОС *Microsoft Vista*, введён мандатный контроль целостности графических сущностей, значительно повысивший защищённость графической подсистемы, однако в *X Windows System* ничего подобного не произошло [60].

Среди администраторов безопасности ОС семейства *Linux* распространено мнение, что для её превращения в высокозащищённую ОС достаточно установить пакет *Security Enhanced Linux* (*SELinux*) специально разработанный для этой цели в АНБ США [236]. Данный пакет широко и успешно применяется для реализации замкнутой программной среды, однако попытки его применить для реализации более сложных моделей управления доступом, как правило, кончаются неудачей, так как требуют существенных усилий по разработке соответствующих политик. На рис. 1.1 показан типичный вид рабочего стола после включения *MLS*-политики[17,,103] *SELinu*x в *Fedora Linux*.

Попытки построить на базе ОС семейства *Linux* защищенную ОС неоднократно предпринимались как в России, так и за рубежом. Исторически можно считать, что первым проектом в данном направлении является ОС *Linux-Mandrake Russian Edition* ,разрабатывавшаяся группой энтузиастов в 1999-2000 гг. и в дальнейшем «выросшая» в проект ОС *ALT Linux*, поддерживаемый компанией «Альт Линукс». Начиная с 2005 г., дистрибутив ОС *ALT Linux* является полностью самостоятельным [97]. Подсистема безопасности ОС *ALT Linux* несколько интересных нововведений (раздельное хранение аутентификационных данных разных пользователелей, минимизация количества *SUID*. и SGID-программ), которые однако, не оказывали существенного влияния на общую

защищенность ОС. В настоящее время этот проект реализуется ООО «Базальт СПО», выпускающей ОС Альт, некоторые дистрибутивы которой реализуют мандатное управление доступом [1] на основе пакета

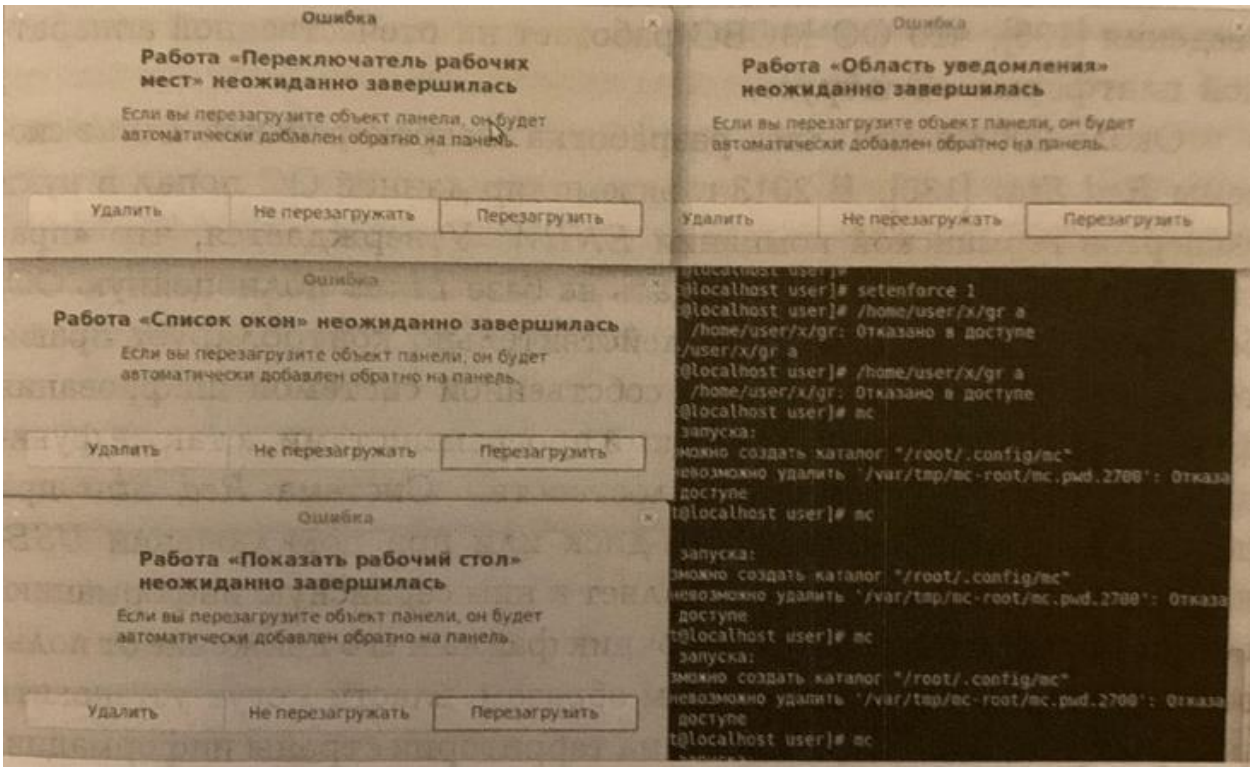


Рисунок 1.Пример результата включения MLS-политики SELinux в Fedora Linux

SELinux [136]. При этом ОС Альт СП 8 является несомненно лучшей реализацией пакета *SELinux* среди всех известных авторам ОС. Тем не менее экспериментально показано, что эта ОС имеет ряд потенциальных уязвимостей в подсистеме мандатного управления доступом, связанные с реализацией информационных потоков по времени: возможна утечка конфиденциальной информации через устройства */dev/pts/*,*через системные журналы, через уязвимости графической подсистемы.

Какое-то время самой защищённой российской ОС семейства *Linux* являлась Мобильная система Вооружённых Сил (МСВС),разработанная по заказу Минобороны России Всероссийским научно-исследовательским институтом автоматизации управления в непромышленной сфере (ВНИИНС) на основе ОС *Red Hat Enterprise Linux*[6]. ОС МСВС была принята на снабжение вооружённых сил в 2002 г. На протяжении многих лет она широко применялась в самых различных компьютерных системах военного и двойного назначения, существуют её настольные и серверные версии, предназначенные для функционирования на обычных бытовых компьютерах.

Последняя достоверно существующая (о более новых версиях авторами информации не найдено) версия МСВС 5.0, сертифицированная в 2011 г, содержит ядро *Linux* версии 2.6.32 и *glibc* версии 2.5 сборки 2006 г. Утверждается [78], что установка дополнительного ПО в

данную ОС серьёзно затруднена. Осенью 2017 г. появились сведения, что ОС МСВС работает на отечественной аппаратурной платформе «Эльбрус».

Около 2003 г. началась разработка северокорейской *Linux*-системы *Red Star*[130]. В 2013 г. экземпляр данной ОС попал в руки экспертов германской компании *ENRW*. Утверждается, что «правительству КНДР удалось создать на базе *Linux* полноценную ОС,большую часть кода которой действительно контролирует правительство». *Red Star* обладает собственной системой шифрования файлов, разработанной корейскими программистами, а также функциями защиты собственной целостности. Система *Red Star* при попадании файлов на жёсткий диск или при подключении *USB* накопителя к компьютеру добавляет к ним сервисную информацию, позволяющую отслеживать источник файла и его движение от пользователя к пользователю. Таким образом, власти могут установить распространителя запрещённой на территории страны информации.

В 2006 г. на Украине компанией «Майлинукс» была разработана первая украинская ОС, названная *myLinux* [5]. Весной 2014 г. был анонсирован [84] проект другой украинской ОС, которая будет построена на основе ОС семейства *Linux* , откуда «будут вырезаны продукты и приложения, которые смогут повлиять на её безопасность». Предполагалось, что к концу лета того же года проект будет представлен на рассмотрение Совета национальной безопасности и обороны Украины, однако никаких сообщений, что это действительно произошло, в доступных источниках не обнаруживается.

В 2006 г. в Китае начались поставки в военные и государственные учреждения ОС *Kylin*, разработанной китайским Национальным университетом оборонных технологий на базе ОС *FreeBSD*[118]. Название ОС означает мифическое животное, часто упоминающееся в китайском фольклоре наряду с драконом и фениксом. Дистрибутив ОС *Kylin* некоторое время был доступен для скачивания. По мнению экспертов [119] данная ОС не содержит каких-либо выдающихся защитных механизмов, никто не упоминает о поддержке в ней мандатного управления доступом, мандатного контроля целостности, изолированной программной среды и т. п. В 2013 г.официально стартовал проект ОС *Ubuntu Kylin*, по всей видимости, не имеющий ничего общего с ОС *Kylin*, кроме названия. ОС *Ubuntu Kylin* представляет собой версию ОС *Ubuntu Linux* с полнофункциональной поддержкой китайского языка и несколькими предустановленными приложениями, специфичными для Китая (лунный календарь, упрощённый:лоступ к китайским социальным сетям, поставщикам китайской музыки и т, п.). Есть сведения [144], что обычный дистрибутив *Ubuntu* легко превращается в ОС *Ubuntu Kylin* в результате установки нескольких дополнительных пакетов ПО. Нет никаких сведений, что подсистема безопасности данной ОС чем-либо отличается от подсистемы безопасности ОС *Ubuntu Linux*. При этом ОС *Ubuntu Kylin* эксплуатируется не только в Китае.

Около 15 тыс. компьютеров с предустановленной данной ОС были поставлены на Украину в качестве гуманитарной помощи [120].

Семейство отечественных ОС «РОСА» производится с 2009 г, группой компаний «РОСА» изначально на основе ОС *Mandriva Linux* [132]. Семейство ОС «РОСА» включает в себя сертифицированные защищённые ОС «РОСА Хром» и «РОСА Никель», в которых заявлена поддержка мандатного управления доступом [64,65], а также «РОСА Кобальт» и несколько свободно распространяе-мых дистрибутивов, потребительские качества которых оцениваются пользователями довольно высоко [51]. Весной 2015 г, «НТЦ ИТ РОСА» вошла в число пяти российских компаний, подавших заявку на государственную поддержку отечественных программных продуктов [32]. В 2018 г. появились сведения о разработке дистрибутива *ROSA Enterprise Server* , основанного на *Red Hat Enterprise Linux* [31]. Сведений о существенных научно обоснованных технических решениях в области информационной безопасности, заложенных в
основу разработки данной ОС, обнаружить не удалось.

В 2010 г. знаменитая польская исследовательница безопасности ОС Йоанна Рутковска объявила, что ведёт разработку защищённой ОС *Qubes*, основанной на ОС *Fedora Linux* [129]. Дополнительные средства защиты данной ОС основаны на инкапсуляции прикладных и системных программ в отдельные виртуальные машины, взаимодействие которых реализуется посредством гипервизора *Xen*. Передача данных между виртуальными машинами осуществляется на основе общесистемной политики безопасности, потенциально способной поддерживать мандатное управление доступом, мандатный контроль целостности, изолированную программную среду и т.п. Для пользователя наличие этой политики становится заметным только в том случае, если он попытается её нарушить, иначе работа пользователя выполняется как в обычной ОС *Fedora Linux*. Окна программ, принадлежащих к разным виртуальным машинам, выполняются на
общем рабочем столе, как при включённом *seamless mode* в ПО *Virtual Box* (в русскоязычной локализации этот режим называется «Режим интеграции дисплея».

Позже похожая идея была реализована отечественной компанией «НеоБИТ», изготовившей гибридную ОС *Linux over Febos,* в которой прикладные программы выполняются в виртуальных машинах ОС *Linux*, выступающих, в свою очередь, в роли прикладных программ ОС «Фебос» — собственной разработки «НеоБИТ»,не имеющей отношения к семейству ОС *Linux* |8]. Фактически, ОС «Фебос», насколько можно судить по доступным описаниям, не содержит в себе ничего, кроме микроядра, гипервизора и монитора безопасности, все прикладные интерфейсы вынесены в виртуальные машины ОС *Linux*.

ОС «Заря» разработана АО «Центральный научно-исследовательский институт экономики информатики и систем управления» (ЦНИИ ЭИСУ) [89] по заказу Минобороны

России в 2013 г. [92]. Данная ОС основана на ОС *CentOs Linux* и ОС *Red Hat Enterprise Linux* , доступные в сети Интернет схемы её архитектуры не содержат никаких уникальных особенностей, существенно отличающих её от других ОС семейства *Linux*. Некоторые источники позиционируют ОС «Заря» как следующее поколение ОС МСВС [82]. Утверждается, что ОС «Заря» совместима с пакетами ПО *Libre Office* , *GIMP* и *Chromium*, существуют версии ОС для настольных, серверных и встраиваемых систем, версия «Заря-ЦОД» для построения центров обработки данных, «Заря РВ» — дистрибутив, предназначенный для построения ОС реального времени «для встраиваемых систем, работающих без участия человека, которые должны обрабатывать информацию в режиме реального времени» [48], а также универсальная интеграционная шина (УИН!) «Заря», предназначенная для построения систем информационного обмена в задачах АСУ [47]. Никаких демонстрационных версий ОС «Заря» не обнаружено, а доступные в сети Интернет снимки экранов неинформативны. В 2015 г ЦНИИ ЭИСУ, кроме того, объявило о создании на базе ядра ОС тих мобильной ОС «РоМОС», которая «подразумевает отсутствие утечки информации, исключает различные закладки, делает невозможной установку шпионского ПО, прослушку» [67].

Обоснований этих утверждений в доступных источниках не обнаруживается. В сентябре 2015 г. ЦНИИ ЭИСУ, объявило о готовности серийному производству новой ОС «Рассвет», представляющей собой развитие серии специализированных ОС «Заря» [3].

В 2013-2014 гг. компанией «Ред Софт» по заказу ФССП Россий разработана ОС «ГосЛинукс», также основанная на ОС *CentOS* которая позиционируется как защищённая ОС с функциями, позволяющими «приставу обрабатывать персональные данные должников и взыскателей без дополнительных средств защиты информации, а также применять электронную подпись для издания документов в электронном виде» [30]. На официальном сайте ФСПП России

утверждается, что «основные доработки, выполненные подрядчиком, касались криптографической подсистемы и предконфитурации встроенных средств защиты информации» [46]. Позже той же компанией была разработана ОС «Ред ОС», которая, как утверждает разработчик, «имеет в своём составе специализированную подсистему распределённого аудита, которая позволяет отслеживать критичные события безопасности в корпоративной сети и предоставляет IT-администратору необходимые инструменты для оперативного реагирования на инциденты ИБ» [125]. О иных функциях безопасности, выходящих за рамки типичных для других ОС семейства *Linux*,ничего не сообщается.

Не позднее 2014 г. корпорацией «Росатом» создана так называемая «Защищённая операционная система для суперЭВМ» (ЗОС), основанная на программной платформе ОС семейства *Linux*. Утверждается, что данная ОС сертифицирована ФСТЭК России и «средства

защиты информации в ЗОС отвечают требованиям технических условий, что позволяет применять системное ПО в автоматизированных системах класса защищенности 1В-С» [27]. Заявленный перечень реализованных в ЗОС функций безопасности также является типичным для *Linux*систем.

ОС «Эльбрус» разработана АО МЦСТ [39] для вычислительных комплексов, построенных на основе отечественной аппаратной платформы «Эльбрус». Компания-разработчик декларирует возможность применения ОС «Эльбрус» в качестве ОС реального времени, утверждает, что ‹в ядро программной платформы «Эльбрус» встроен комплекс СЗИ от несанкционированного доступа, который позволяет использовать ОС для построения автоматизированных систем, отвечающих самым высоким требованиям информационной безопасности» [125]. Подробности реализации данного комплекса не приводятся.

Помимо вышеперечисленных ОС, существует целый ряд защищённых ОС семейства *Linux*, о которых известно либо мало, либо практически ничего, кроме названия. К этой группе относятся, в частности, отечественные ОС *Saifish Mobile OS RUS*[66], ОС «ОСь» [38], ОС «Стрелец», ОС «ОСнова» [37|.

В целом, несмотря на очевидные недостатки безопасности ОС семейства *Linux*, широкий спектр не всегда удачных разработок защищённых ОС на их основе, в настоящее время ОС этого семейства по прежнему являются почти идеальной платформой для создания отечественной защищённой ОС. Используемые в ОС семейства *Linux* иногда примитивные и устаревшие механизмы защиты позволяют без кардинальной переработки, блокирования или учёта их особенностей реализовывать в ОС современные механизмы мандатного и ролевого управления доступом, мандатного контроля целостности, и добиваться строгого теоретического обоснования безопасности и верификации полученного решения. Таким образом, сочетание высокой надёжности, приемлемых потребительских качеств, открытого исходного кода и возможности сравнительно лёгкой доработки механизмов защиты ОС семейства *Linux* позволяют построить на
их базе высокозащищённую отечественную ОС ценой существенно меньших усилий, чем если бы она создавалась с нуля или строилась на других платформах.

1.3. Архитектура, назначение и области применения ОССН

1.3.1. Назначение ОССН

Доктрина информационной безопасности Российской Федерации [25] определяет ряд стратегических целей в таких областях, как оборона страны, государственная и общественная безопасность, экономическая сфера, наука, технологии и образование. В

общем виде эти цели определяют вектор развития организационных и технологических решений, направленных, в том числе на:

• обеспечение безопасности информации, обрабатываемой в информационных системах на территории Российской Федерации;

• обеспечение устойчивого и бесперебойного функционирования

информационной инфраструктуры, в первую очередь критической информационной инфраструктуры Российской Федерации;

• повышение безопасности функционирования объектов информационной инфраструктуры, в том числе в целях обеспечения устойчивого взаимодействия государственных органов;

• повышение безопасности функционирования образцов вооружения, военной и специальной техники и автоматизированных систем управления.

Эффективное развитие и совершенствование этих решений базируется, в первую очередь:

• на проведении научных исследований и реализации опытных коммерческих разработок в целях создания перспективных информационных технологий и средств обеспечения информационной безопасности;

• создании и внедрении информационных технологий, изначально устойчивых к различным видам воздействия.

При этом важнейшим аспектом, актуальным в современных реалиях глобализации информационных технологий, связанной с трансграничным характером их внедрения и использования, является минимизация, а в перспективе и полная ликвидация зависимости отечественных технологических решений от зарубежных информационных технологий и средств обеспечения информационной безопасности. Очевидными факторами достижения этого являются:

• создание, развитие и внедрения во все сферы, связанные с применением информационных технологий, отечественных разработок;

• совершенствование методов и способов создания средств и оказания услуг на основе информационных технологий с использованием отечественных разработок, удовлетворяющих требованиям информационной безопасности;

• обеспечение конкурентоспособности российских информационных технологий и развитие научно-технического потенциала в области обеспечения информационной безопасности.

Таким образом, вопросы ограничения применения в рамках АС (в первую очередь функционирующих в интересах органов государственной власти) продукции иностранного производства, для которой существуют отечественные аналоги, являются особенно

актуальными для современных условий. В частности, изменения, внесённые в Федеральные законы «Об информации, информационных технологиях и о защите информации» [85] и «О контрактной системе в сфере закупок товаров, работ, услуг для обеспечения государственных и муниципальных нужд» [86], имеют прямое отношение к курсу на импортозамещение, базирующемуся на тенденциях развития рынка российского ПО.

ОССН в полной мере отражает рассмотренные выше аспекты.Являясь результатом планомерного процесса по формированию и совершенствованию системной программной платформы для автоматизированных систем в защищённом исполнении (АСЗИ), ОССН представляет собой симбиоз решений, базирующихся на концепции свободного ПО, органично дополненных совокупностью программных модулей и информационных структур, являющихся авторскими разработками АО «НПО РусБИТех». При этом решения, заложенные в основу ОССН, представляют собой результат как научных исследований, прошедших тщательную научно-техническую верификацию, так и проведения опытно-конструкторских работ ,направленных на интеграцию ОССН, как системной платформы, в широкий спектр АСЗИ, что подтверждается соответствующими процедурами по сертификации СЗИ.

Актуальная в настоящее время версия 1.6 релиза ОССН «Смоленск», ориентированного на применение в классе вычислительных систем, поддерживающих процессорную архитектуру x86-64, представляет очередной этап эволюционного развития системной программной платформы ОССН. Совершенствование функциональных возможностей этой версии основано, как минимум на двух важных тенденциях:

• использование обратной связи с потребителями, активно эксплуатирующими системы в составе широкого спектра АСЗИ,функционирующих на разных вариантах платформы x86-64:

• упреждающий учёт тенденций развития современных АСЗИ,а именно — увеличение роли технологий виртуализации платформ в интересах создания гибких, масштабируемых, высокопроизводительных и отказоустойчивых инфраструктур, а также решение задач кроссплатформенной интеграции и межплатформенной миграции доменных инфраструктур, развёртываемых в рамках крупных корпоративных информационных систем.

При этом, наряду с поддержкой указанных тенденций, важнейшим аспектом развития ОССН является совершенствование её компонентов, связанных с функциями защиты информации. В этом направлении разработчики ОССН опираются на строгое соответствие национальным стандартам в области защиты информации, а также создания и модернизации АСЗИ, включая ГОСТ 51583-2014 «Защита информации. Порядок создания автоматизированных систем в защищённом исполнении. Общие положения» [12], ГОСТ Р58256-2018 «Защита информации. Управление потоками информации в информационной

системе. Формат классификационных меток» [13], требования системы сертификации СЗИ, определённые ФСТЭК России в «Положении о системе сертификации средств защиты информации» [49].

Версия 1.6 ОССН релиза «Смоленск» имеет следующие сертификаты соответствия участников Системы сертификации СЗИ по требованиям безопасности информации (табл. 1.1).

Указанные сертификаты подтверждают возможность использования ОССН в составе АСЗИ различных классов защищённости с заданными уровнями защиты от несанкционированного доступа к информации (НДС) и наличия недекларируемых возможностей (НДВ)

Таблица 1.1

Сертификаты ОССН

Участник системы сертификации	Номер сертификата	Дата получения сертификата	Срок действия Сертификата
Федеральная служба безопасности Российской Федерации	СФ/014-2961	09.09.2016	31.08.2021
Федеральная служба по техническому и экспортному контролю Российской Федерации	2557	27.01.2012	27.01.2021
Министерство обороны Российской Федерации	1339	24.09.2010	17.04.2023

в составе компонентов системы, в том числе АСЗИ, обрабатывающих информацию, содержащую сведения, составляющие государственную тайну с грифом не выше «совершенно секретно». Предельный уровень защиты версии 1.6 релиза ОССН сертифицирован для использования в многопользовательских АСЗИ, пользователи которых имеют разные полномочия по доступу к обрабатываемой информации (Группа 16), по классу 3 защиты от НСА и уровню 2 контроля отсутствия НДВ. Кроме того, ОССН версии 1.6 сертифицирована на соответствие требованиям профиля защиты ОС общего назначения (типа «А») второго класса защиты [61].

В общем же случае АСЗИ, реализованные с использованием ОССН, могут обеспечивать конфиденциальность информации применительно к сведениям конфиденциального характера в соответствии с [35], а именно:

• о фактах, событиях и обстоятельствах частной жизни гражданина, позволяющие идентифицировать его личность (персональные данные), за исключением сведений, подлежащих распространению в средствах массовой информации в установленных федеральными законами случаях,

• составляющим тайну следствия и судопроизводства, о лицах, в отношении которых в соответствии нормативными правовыми актами Российской Федерации принято решение о применении мер государственной защиты, а также о мерах государственной зашиты указанных лиц;

• служебным сведениям, доступ к которым ограничен органами государственной власти в соответствии с Гражданским кодексом Российской Федерации и Федеральными законами (служебная, тайна);

•связанным с профессиональной деятельностью, доступ к которым ограничен в соответствии с Конституцией Российской Федерации и федеральными законами (врачебная, нотариальная, адвокатская тайна, тайна переписки, телефонных переговоров, о почтовых отправлений, телеграфных или иных сообщений и т.д.);

• связанным с коммерческой деятельностью, доступ к которым в ограничен в соответствии с Гражданским кодексом Российской Федерации и федеральными законами (коммерческая тайна);

• о сущности изобретения, полезной модели или промышленного и образца до официальной публикации информации о них;

• содержащимся в личных делах осуждённых, а также о принудительном исполнении судебных актов, актов других органов и должностных лиц.

Наряду с обеспечением защиты информации, содержащей сведения, составляющие государственную тайну, а также информации, содержащей сведения конфиденциального характера, ОССН отвечает требованиям Федерального закона «О безопасности критической информационной структуры Российской Федерации» [87] в рамках статей 10 и 11 (предотвращение неправомерного доступа к информации, обрабатываемой значимым объектом критической информационной инфраструктуры, уничтожения такой информации, её модифицирования, блокирования, копирования, предоставления и распространения, а также иных неправомерных действий в отношении такой информации).

Состав и функциональные возможности компонентов ОССН в полной мере соответствуют требованиям, определенным в постановлении Правительства Российской Федерации «Об установлении запрета на допуск программного обеспечения, происходящего из иностранных государств, для целей осуществления закупок для обеспечения государственных и муниципальных нужд [50], разработанного в рамках реализации программы Правительства Российской Федерации по импортозамещению. В частности,

сведения об ОССН включены в единый реестр российских программ для электронных вычислительных машин и баз данных [26].

В контексте стандартизации и сертификации функциональных решений, заложенных в ОССН, важным аспектом её развития является верификация (формальное доказательство) непротиворечивости изложенной в главе 2 настоящего учебного пособия мандатной сущностно-ролевой ДП-модели безопасности управления доступо информационными потоками (МРОСЛ ДП-модели), положенной в основу комплекса СЗИ, и реальной его функциональной спецификацией, реализованной в конкретном релизе ОССН. Нормативной основой такой верификации являются ГОСТ Р ИСО, МЭК 15408-3-2013 «Методы и средства обеспечения безопасности. Критерии оценки безопасности информационных технологий» [9] и утверждённые в 2018 г ФСТЭК России «Требования по безопасности информации, устанавливающие уровни доверия к средствам технической защиты информации и средствам обеспечения безопасности информационных технологий» [83], определяющие следующие требования к процессу разработки критически важного с точки зрения защиты информации ПО:

• проведение формального моделирования политики безопасности;

• верификация непротиворечивости модели политики безопасности с получением оценок недостижимости небезопасных состояний системы, в рамках которой эта политика безопасности реализуется;

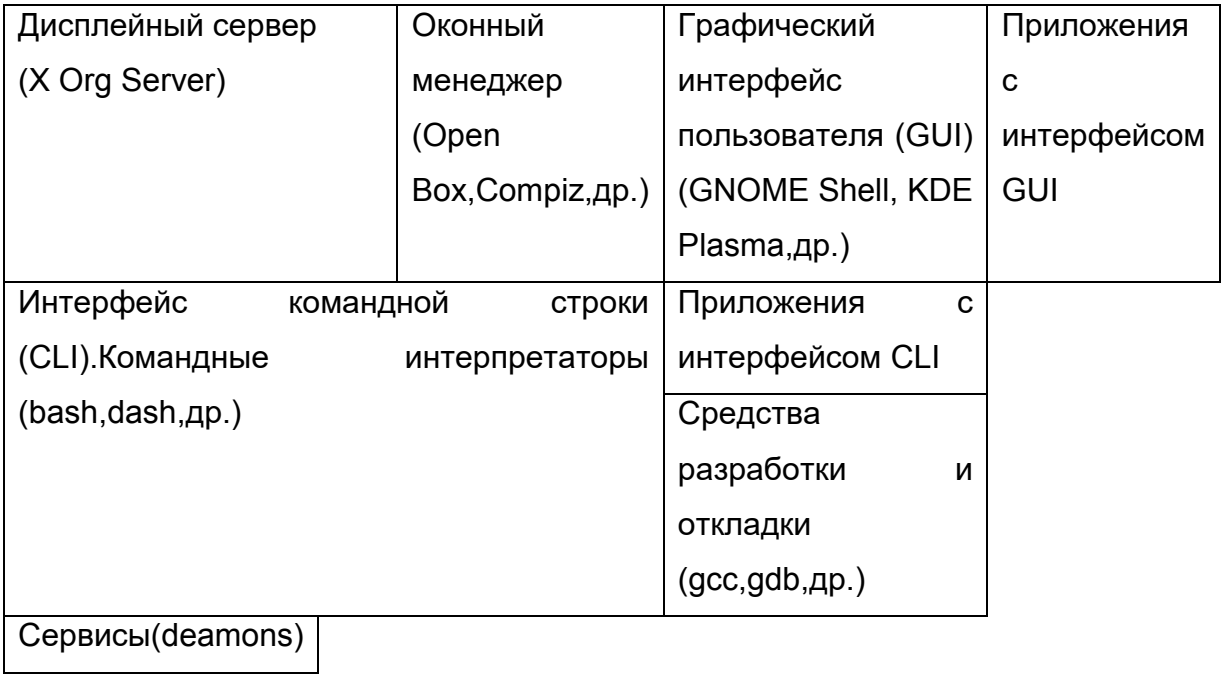
• разработка формальной функциональной спецификации СЗИ;

• верификация соответствия между разработанной формальной функциональной спецификацией СЗИ и моделью политики 6езопасности;

• верификация соответствия между формальной функциональной спецификацией СЗИ, проектом ПО СЗИ и реальной реализацией СЗИ.

В ходе выполнения этих требований разработчики ОССН совместно Институтом системного программирования им. В.П. Иванникова Российской академии наук (ИСП РАН) решают задачи по созданию и практической апробации методики и инструментальных средств верификации МРОСЛ ДП-модели и механизмов СЗИ в ОССН. Для этого ИСП РАН разработан комплекс инструментальных средств *AstraVer Toolset* для дедуктивной верификации СЗИ, реализованных в ОССН [33]. Этот комплекс базируется на проектах свободного ПО *Rodin Platform*

(инструментальное средство дедуктивной верификации моделей на формальном языке *Event*-B), Why3 (платформа дедуктивной верификации Си-программ с использованием инструмента *Frama-C*, основанного на языках спецификаций *ACSL u WhyML)* [19, 93, 94, 110, 114].



1.3.2. Архитектура ОССН

Архитектурной основой ОССН является проект *Debian*/Linux [34, 41] — ассоциация разработчиков свободного ПО, основой которой являются . ОС семейства *GNU/Linux* на базе ядра *Linux Kernel* [142] . Дистрибутив ОССН включён в список производных

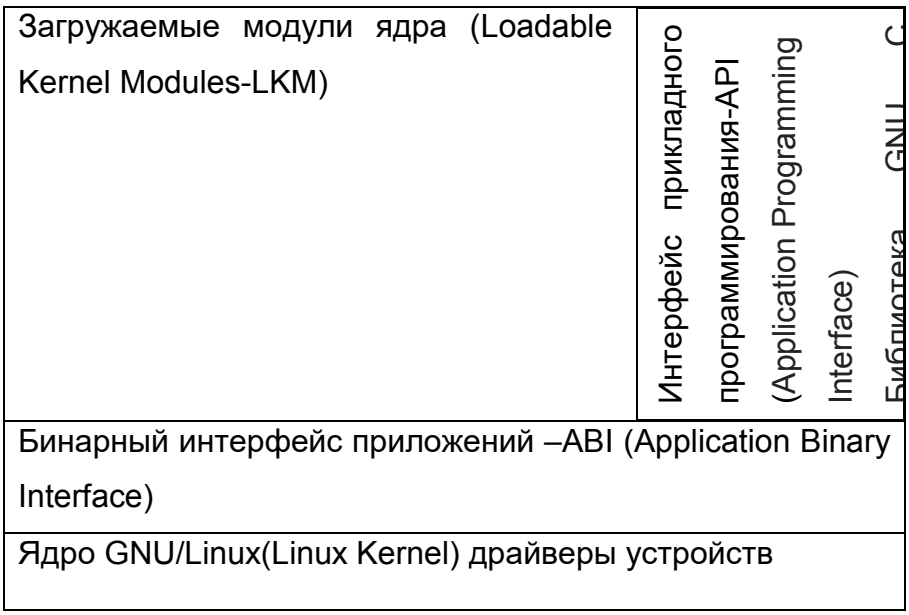


Рис. 1.2. Обобщённая архитектура ОС на базе проекта GNU/Linux

дистрибутивов *Debian* (ветка *Russian Debian derivatives*)[109]. В общем случае это означает, что сообщество разработчиков *Debian* признает дистрибутив ОССН как удовлетворяющий следующим критериям:

• разработчики дистрибутива активно взаимодействуют с проектом *Debian*;

• дистрибутив активно сопровождается;

• в команде разработчиков дистрибутива имеется хотя бы один участник проекта *Debian*;

• разработчики дистрибутива вступили в программу сбора сведений о производных дистрибутивах и добавили файл sources.list на страницу сведений о своём производном дистрибутиве;

• производный дистрибутив имеет какую-либо особенность или ориентацию;

• производный дистрибутив относится к известным (авторитетным) дистрибутивам.

Таким образом, в целом архитектура ОССН соответствует архитектурным решениям *GNU/Linux* (рис. 1.2).

При этом особенности реализации ОССН соответствуют особенностями реализации ОС проекта *Debian GNU/Linux*, в частности:

• поддержка последних версий стандартов в рамках проектов *Linux Standard Base (LSB)* [124] и *Filesystem Hierarchy Standard (FHS)*[113];

•наличие более одиннадцати официальных переносов (портов) на различные процессорные архитектуры;

• наличие системы управления пакетами программ *APT*(*Advanced Packaging Tool*) с жесткой политикой по отношению к разрабатываемому ПО, поддержкой разветвлённой сети репозиториев и стандарта механизма выбора предпочтительного ПО среди нескольких вариантов (*alternatives*);

• большое число (более 50 тысяч) пакетов совместимого прикладного ПО;

• наличие системы отслеживания ошибок *BTS* (*Bug Tracking System*), обеспечивающей высокое качество кода драйверов, системных сервисов и поддерживающей высокую стабильность функционирования производных дистрибутивов на базе *Debian GNU/Linux*

Являясь производным дистрибутивом *Debian GNU/Linux*, ОССН официально поддерживает переносы на следующие архитектуры:

• amd464 — процессоры *Intel* и *AMD*с микропроцессорной архитектурой x86-64;

•armel,armhf,AArch64— 32- и 64-разрядные процессорные ядра разработки фаблесс компании *ARM Limited;*

• s390x — 64-разрядное пространство пользователя для мэйнфреймов *IBM System z;*

• MIPS (*mips* и mipsel) — 32-разрядное процессорное ядро разработки фаблесс компании *MIPS Technologies, Inc*. Поддержка реализации процессорного ядра *P5600* архитектуры *MIPS32* Release 5 в системе на чипе *BE-T1000* (*Baikal-T1*) фаблесс комцании *Baikal Electronics* и систем на чипе *1890ВМ8Я* «КОМДИВ64А-М» и *1907ВМ028* «КОМДИВ64-КНИ» разработки ФГУ ФНЦ «НИИСИ РАН » [2];

• *POWER* — процессорная архитектура разработки компании *IBM* (консорциум *openPOWER).*Поддержка реализации суперскалярных процессоров *IMB POWER*8 (*Turismo SCM)* реализована в высокопроизводительных серверных платформах *YARDO VESNIN* компании *YARDO* [90].

Вне рамок официальных переносов *Debian GNU/Linux* версия 1.6 ОССН поддерживает процессорную архитектуру «Эльбрус», реализованную в процессорах Эльбрус-8С, Эльбрус-1С + , Эльбрус-4С компании МЦСТ [29].

Существующие релизы дистрибутива ОССН базируются на указанных переносах *Debian GNU/Linux* и собственных сборках для архитектуры «Эльбрус». Их обозначения отличаются редакцией дистрибутива и номером версии. Редакция дистрибутива определяет специфику дистрибутива: поддерживаемый перенос (аппаратная платформа) и область применения. Номер версии — две или три цифры определяют версию дистрибутива ОССН.

В установленном дистрибутиве ОССН обозначение релиза указано в файле /*etc/astra_version*.. Формат записи релиза:

EDITION Release (Codename),где:

•*EDITION*— редакция релиза дистрибутива (для ОССН имеет значение «SE»);

• *Release* — цифровое кодирование текущей версии релиза ОССН в формате VUY, где V — первая цифра релиза, связанная с его именем (*Codename*), U — вторая цифра, отражающая текущую версию релиза, Y— третья цифра, отражающая номер обновления в пределах текущей версии релиза (если таких обновлений не было, номер обновления отсутствует);

• (*Codename*) — имя релиза на латинице (связано с редакцией ООСН и первой цифрой релиза). В частности, для ОССН в качестве имен релизов используются названия городов-героев России.

Также информацию о текущей версии релиза и имени релиза можно получить в консольном режиме с помощью команды отображения информации о дистрибутиве *lsb_releasea*. Пример вывода этой команды представлен на рис. 1.3.

Существующие в настоящее время дистрибутивы и их именование для релизов ОССН версии 1.6 представлены в табл. 1.2, для релизов ОССН версии 1.4 и 1.5 представлены в табл. 1.3.

В дальнейшем (если это специально не оговорено по тексту) рассматривается релиз «Смоленск» ОССН версии 1.6. Он основан на релизе *Debian GNU/Linux* 9.4 (кодовое имя *Stretch*). Это означает что в качестве пакетной базы этого релиза ОССН используется репозиторий *Debian GNU/Linux* 9.4. Фактически, это задаёт базовую архитектуру релиза, которая дополняется:

•регулярно выпускаемыми оперативными обновлениями безопасности версий пакетов, в функциональности которых были обнаружены те или иные уязвимости ;

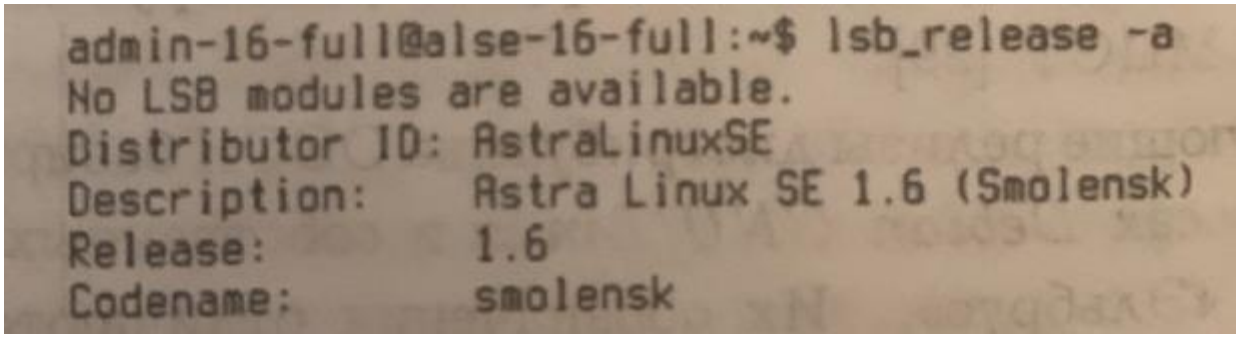


Рис. 1.3. Вывод информации о текущей версии релиза и имени релиза

Таблица 1.2

Релизы ОССН версии 1.6

Имя релиза	Обозначение релиза	Архитектура	СВТ,для которых предназначен релиз
Смоленск	SE 1.6 (smolensk)	amd64 (x86-64)	Персональные компьютеры (в том числе и планшетные) и серверы на базе процессоров *Inte*l и *AMD*
Новороссийск	SE 16 (novorosstysk)	ARM	
Севастополь	SE 6.1.1 (sevastopol)	MIPS32 Release 5	Мобильные терминалы (планшетные персональные компьютеры) и встраиваемые системы
Севастополь	SE 6.1.2 (sevastopol)	MIPS архи-тектура KOMDIV64	Персональные компьютеры и серверы на базе процессоров BE-T1000 (Baikal-T1)
Ленинград	SE 1.6 (leningrad)	Эльбрус	Персональные компьютеры и серверы на базе процессоров 18908ВМ8Я «КОМДИВ64-М»1907ВМ028 «КОМДИВ64-КНИ»

Имя релиза	Обозначение релиза	Архитектура	СВТ,для которых предназначен релиз
Новороссийск Мурманск Тула	SE 4.0.1 (novorosstysk) пакетная база релиза SE 1.5 SE 3.1 (murmansk) пакетная база релиза SE 1.4 SE 1.4 (tula)	ARM S390x ----------	Мобильные терминалы (планшетные персональные компьютеры) и встраивае-мые системы Мейнфреймы *IBM System z* (z9, z10, z12) Маршрутизаторы, шлюзы, межсетевые экраны
			Персональные компьютеры и серверы на базе процессоров Эльбрус-8С, Эльбрус-1С + , Эльбрус-4С

Таблица 1.3

Релизы ОССН версий 1.4 и 1.5

• программами и сервисами с открытым исходным колом, разрабатываемыми в рамках концепции свободного ПО;

•программами и сервисами, являющимися коммерческими продуктами;

• программами и сервисами, являющимися собственной разработкой производителя АО «НПО РусБИТех».

Детализация базовой архитектуры релиза относительно обобщенной архитектуры ОС на базе проекта *GNU/Linux* представлена на рисунке 1.4

Таким образом, архитектурно ОССН базируется на двух вариантах модульного ядра проекта *GNU/ Linux* версии 4.15.3-1:

1.*Kernel Generic*— вариант ядра общего назначения, обеспечивающий функциональность ОССН, такую, как:

• диспетчеризация аппаратных ресурсов компьютера между выполняющимися приложениями и поддержка вытесняющей многозадачности (многопоточности)

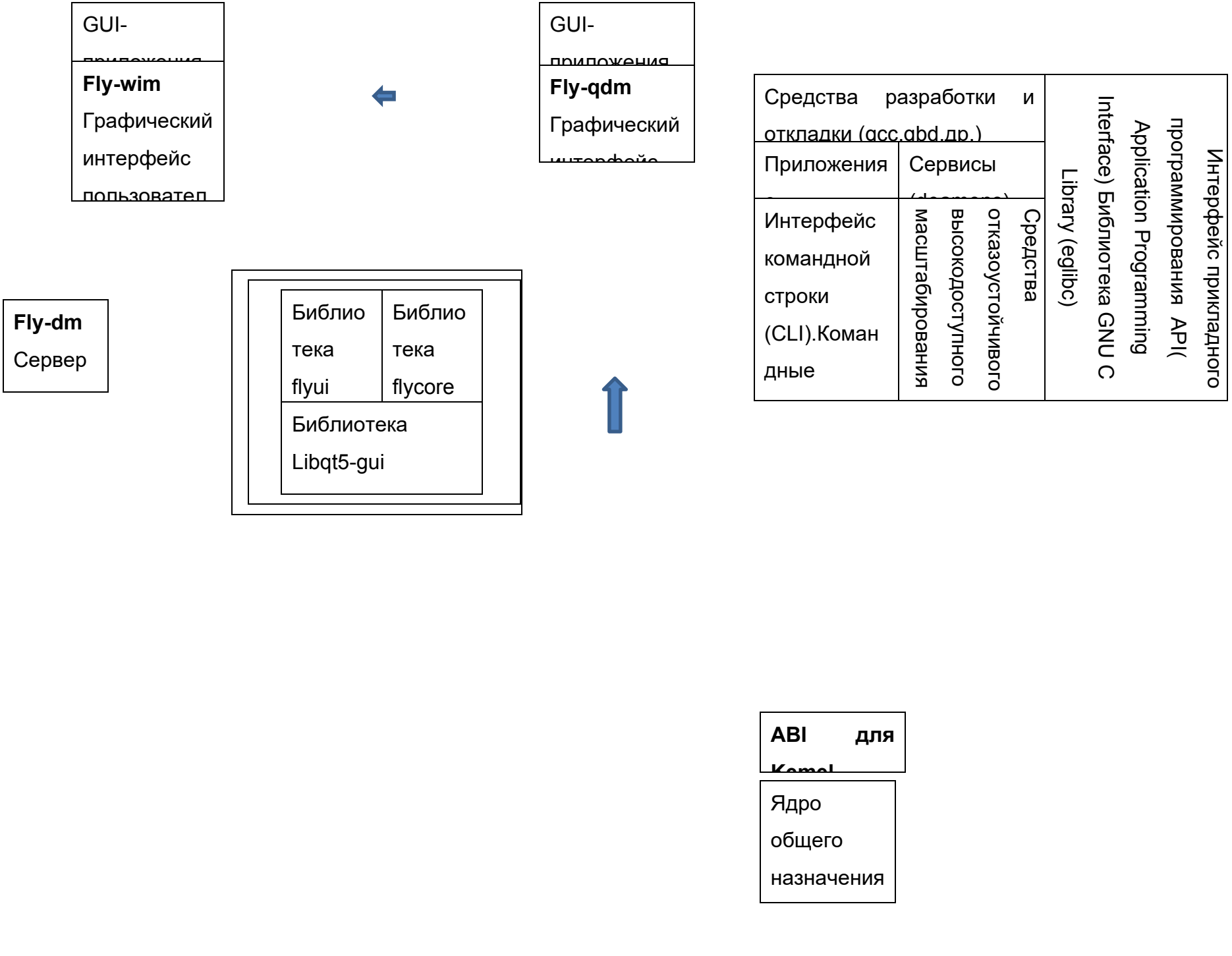
• предоставление механизмов межпроцессного взаимодействия:

• поддержка механизма виртуальной памяти;

• поддержка стека драйверов запоминающих устройств / *storage device*),

• поддержка виртуального коммутатора файловых систем *Linux* (*Linux Virtual File System Switch*);

• поддержка стека сетевых протоколов *TCP/IP* версий 4 и 6.


GUI-
Fly-wim
Графический интерфейс пользовател
GUI-
Fly-qdm
Графический
Fly-dm
Сервер
Библио тека flyui
Библио тека flycore
Библиотека Libqt5-gui
Средства разработки и откладки (gcc,gbd,др.)
Приложения
Сервисы
Интерфейс командной строки (CLI).Коман дные
Средства отказоустойчивого высокодоступного масштабирования
Интерфейс прикладного программирования API(Application Programming Interface) Библиотека GNU C Library (eglibc)
ABI для
Ядро общего назначения

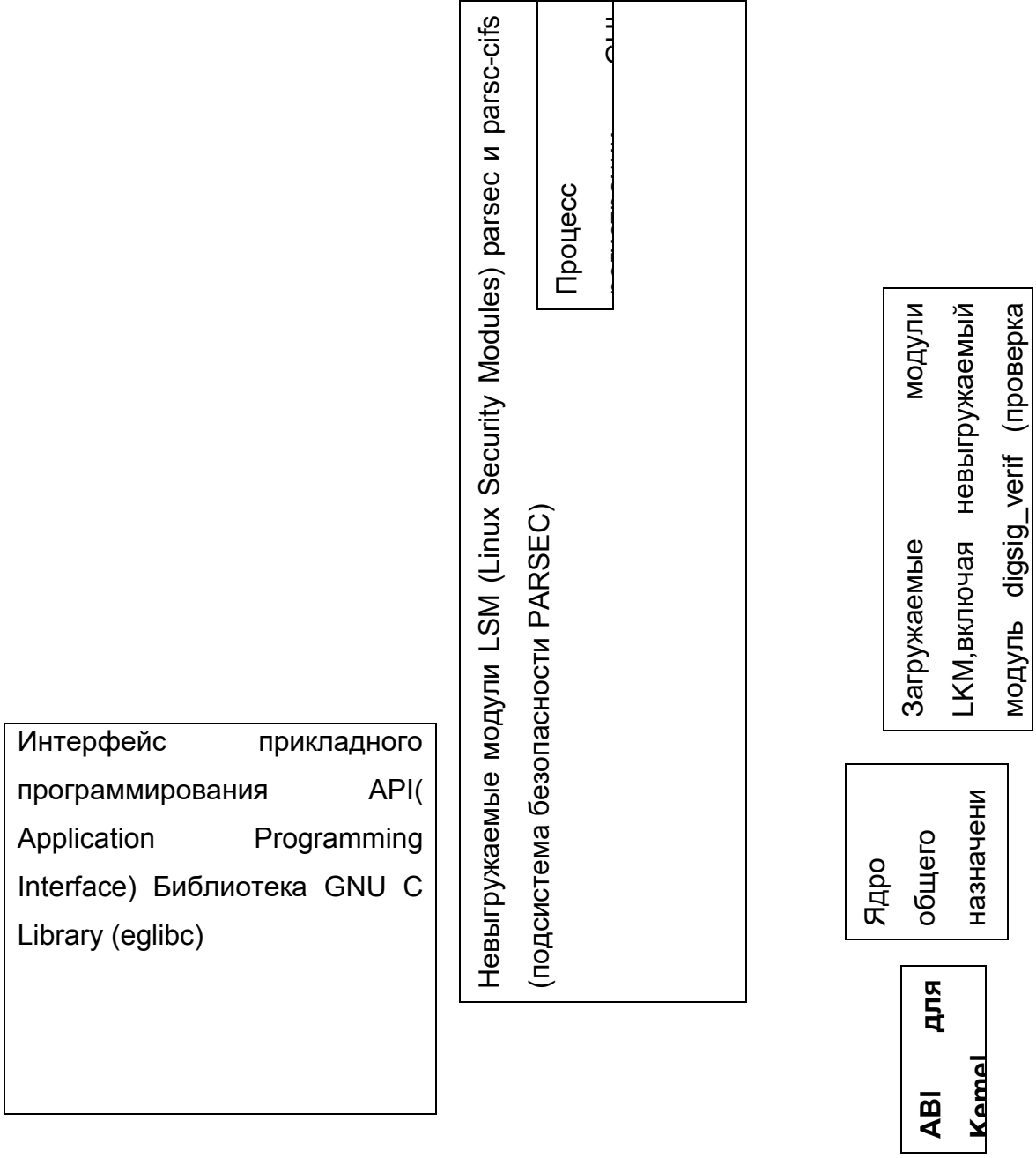


Рис.1.4Обобщенная архитектура ОССН

2. Kernel Hardened — усиленный (защищённый) вариант ядра ОССН, обеспечивающий дополнительно с базовой функциональностью расширенные функции, связанные с защитой ядра, такие, например, как:

•ряд функций модуля PaX (например, *UDEREF* и kernexec), применяемых ранее в ОССН версий 1.4 и 1.5, устанавливающие правила доступа прикладных программ к адресному пространству памяти, что предотвращает несанкционированное выполнение кода, связанного с уязвимостями разыменования указателей или повреждения памяти (*memory corruption*);

• функции, например, такие, как PIE (*Position Independent Executables*) и SSP (*Stack Smashing Protector*), предотвращающие переполнения стека, например, из-за неправильного

использования массивов переменной длины (*Very Large Array*), а также эксплуатации уязвимостей, использующих ошибки глубокой вложенности (*deep nesting*) и рекурсивных функций;

• функции, используемые ядром при копировании данных в/из пространства памяти прикладных программ (*userspace*). Их использование гарантирует, что копии, передаваемые в/из кучи (*heap*) и стека объекта, не превышают его размер (это предотвращает изменение объектов ядра или утечку данных из них);

• функции, предотвращающие внедрение вредоносного кода в процессы путём перехвата начального потока управления.

В качестве параметров конфигурирования исходного кода ядра релиза для получения варианта *Kernel Hardened* релиз ОССН использует рекомендации проекта KSPP (*Kernel Self Protection Project*) [112]

В общем случае ядро *Kernel Hardened* ориентировано на создание высокозащищенных, стабильных серверных платформ, обеспечивающих повышенную функциональную живучесть. Реализация

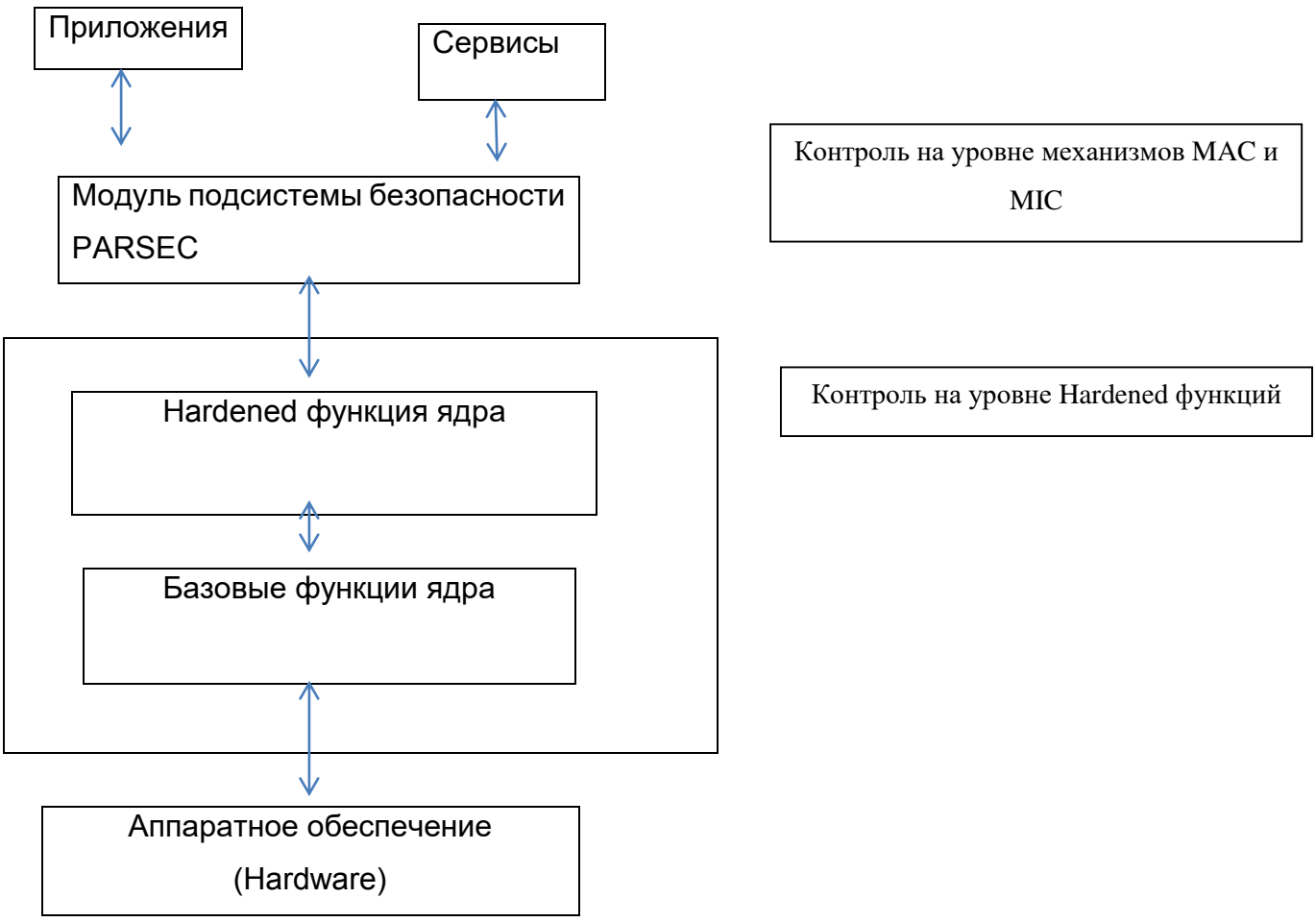


Рис. 1.5. Место функций Hardened Kernel в системе многоуровневой защиты ОССН

перечисленных функций в Kernel Hardened является дополнительным уровнем защиты, наряду с мандатными управлением доступом (МАС) и контролем целостности (МТС), а также, в перспективе, ролевым управлением доступом (*И*ВАС), реализованными в подсистеме безопасности РАЯ5ЕС на основе МРОСЛ АП-модели. Подобный многоуровневый подход усложняет нарушителю эксплуатацию потенциальных уязвимостей системы. В обобщённом виде он представлен на рис. 1.5.

Бинарная совместимость прикладных программ и сервисов как с ядром *Kernel Generic*, так и с ядром Kernel Hardened обеспечивается соответствующими уровнями ABI (*Application Binary Interface)*.

В качестве интерфейса прикладного ПО — API (*Application Programming Interface*) используется библиотека *Embedded GNU C Library (EGLIBC*), принятая в качестве стандарта API по умолчанию в проекте *Debian GNU/Linux*[141].

Функциональность ядер *Kernel Generic* и *Kernel Hardened* расширяется совокупностью загружаемых модулей ядра — LKM (*Loadable Kernel Modules*), ряд из которых является невыгружаемыми.

К ним в первую очередь, относятся:

1) модуль *digsig_verifko*, реализующий функции проверки цифровой подписи бинарных файлов формата *ELF* [63], запускаемых в ходе инициализации системы и пользовательских сеансов;

2) модули, относящиеся к стандарту *LSM (Linux Security Modules*) [145] и реализующие функциональность подсистемы безопасности *PARSEC*. К ним относятся:

• модуль *parsec.ko*— базовые функции подсистемы безопасности *PARSEC*;

• модуль *parsec-cifs.ko* —расширение виртуального коммутатора файловых систем *Linux VFS*, поддерживающего *SNIA* вариант спецификации протокола *CIFS* для организации защищённого удалённого файлового обмена [72]. В качестве основы модуля *parsec-cifs.ko* используется модуль *fs-cifs.co*, доработанный разработчиками ОССН с целью поддержки мандатных меток для протокольных объектов *CIFS*.

Подсистема безопасности *PARSEC* расширяет стандартную для проекта *GNU/Linux* систему привилегий, предназначенную для передачи учётным записям пользователей прав выполнения определённых действий администратора (пользователя *root*), поддержкой описанной в главе 2 иерархической МРОСЛ ДП-модели.

В общем случае она реализована в соответствии с рекомендациями разработки *LSM Module Policy Engine* и функционально представляет собой реализацию монитора безопасности, принимающего решение о доступе процессов, выполняющихся в *userspace* , к затребованным ими объектам режима *systemspace* . Особенностью реализации такого.

модуля *LSM* является использование совокупности механизмов перехвата (*hooks*), которые перехватывают и анализируют аргументы запросов процессов и, в соответствии с установленными правилами доступа МРОСЛ ДП-модели, разрешают или запрещают их. Обобщённое представление процесса функционирования подсистемы безопасности *PARSEC* даётся на рис. 1.6.

Подсистема Р *PARSEC* поддерживает следующие расширенные привилегии:

• посылать сигналы процессам, игнорируя дискреционные и мандатные правила управления доступом;

• право посылать мандатную метку и устанавливать другие привилегии;

• право менять мандатные метки файлов;

• право управлять политикой аудита:

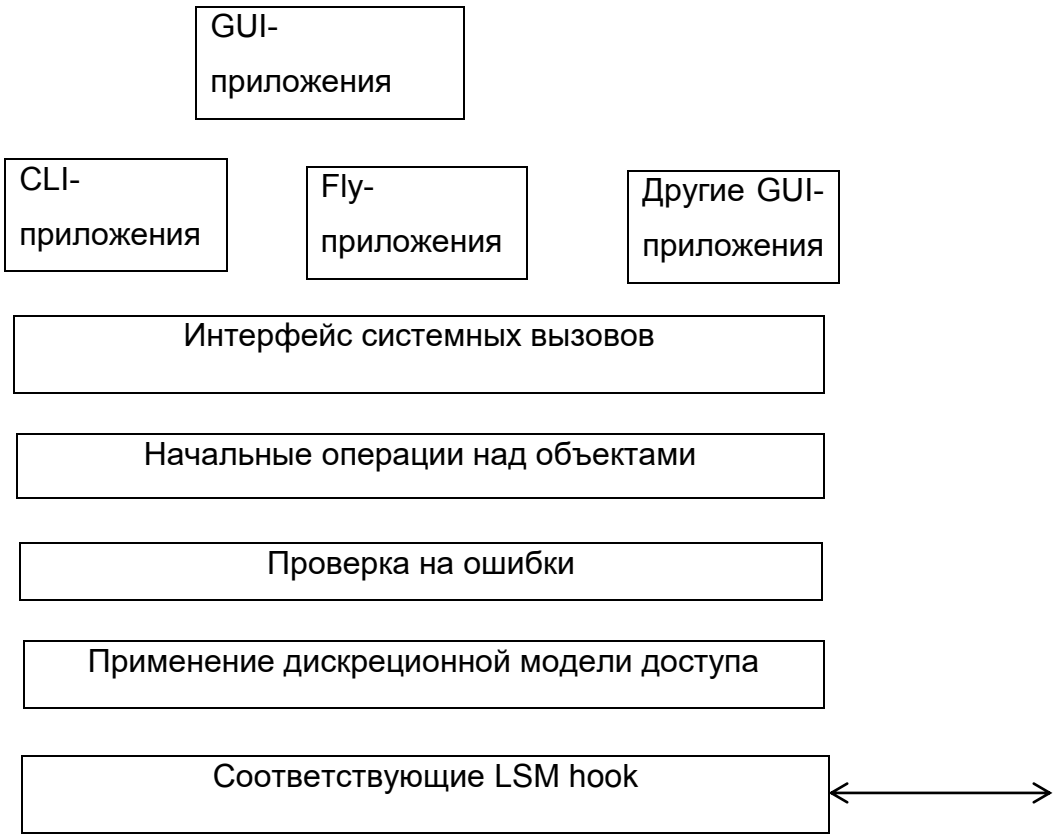
• право игнорировать мандатную политику при чтении и поиске файлов (исключая функцию записи);

• право создавать привилегированный сокет и менять его мандатную метку;

• право изменять время доступа к файлу;

• право игнорировать мандатную политику по уровням и категориям;

• право устанавливать привилегии на файлы;



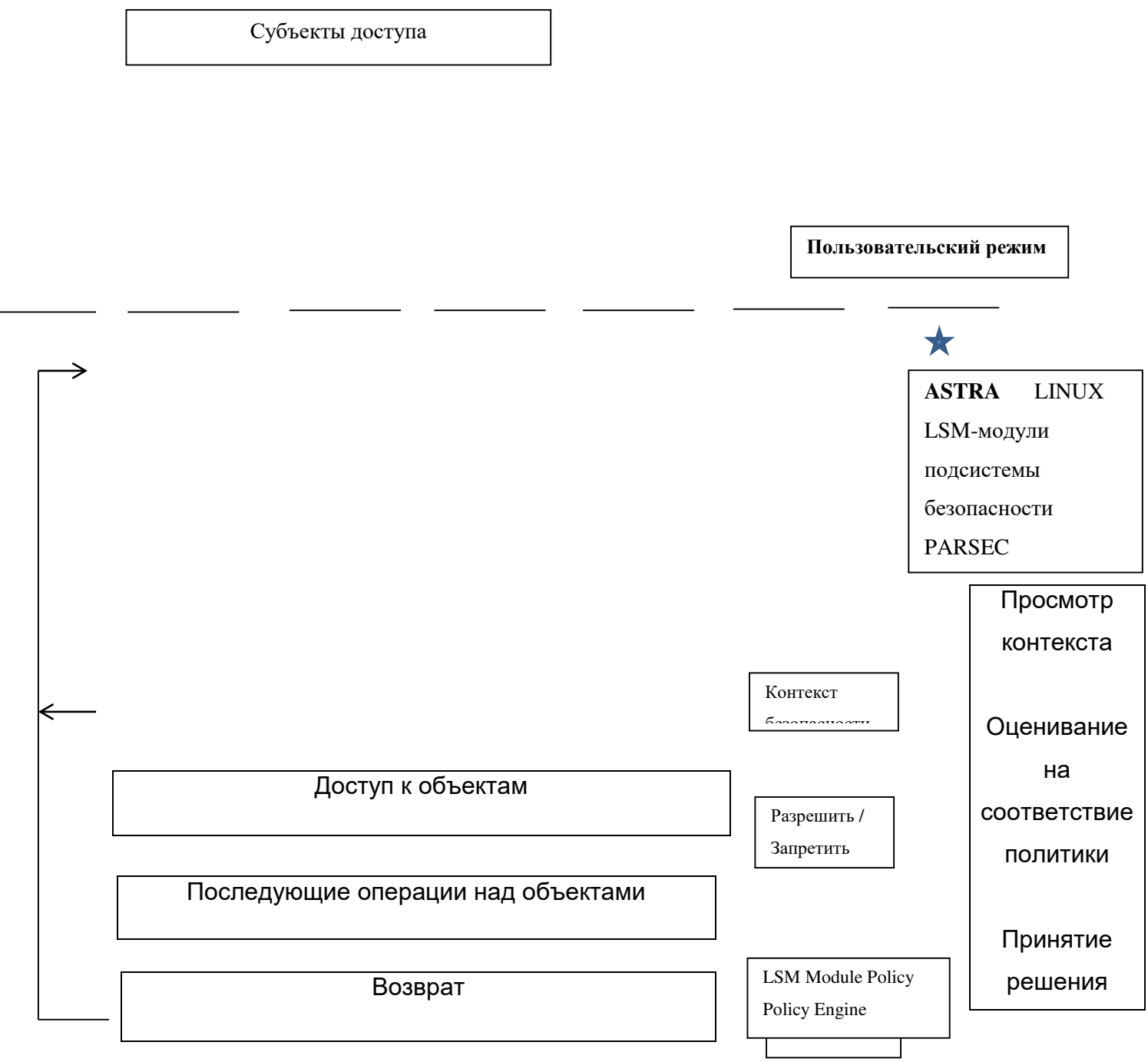

Субъекты доступа
Пользовательский режим
ASTRA LINUX
LSM-модули подсистемы безопасности PARSEC
Просмотр контекста
Оценивание на соответствие политики
Принятие решения
Контекст
Доступ к объектам
Разрешить / Запретить
Последующие операции над объектами
Возврат
LSM Module Policy
Policy Engine

Рис. 1.6. Схема функционирования подсистемы безопасности *PARSEC*

• право устанавливать любой непротиворечивый набор привилегий для выбранного процесса;

•право менять мандатную точку сетевого соединения.

Организация пользовательских сессий в ОССН реализуется традиционно для проекта *GNU/Linux* как консольном (интерфейс командной строки — *CLI Command Line Interface*), так и графическом (графический интерфейс пользователя — *CLI Command Line Interface*) режимах.

Консольный режим пользовательских сессий может быть организован в среде следующих командных интерпретаторов (оболочек):

• *bash* (*Bourne again shell*) — GNU версия POSIX совместимого интерпретатора *sh*;

•*dash (Debian Almquist shell*) — GNU версия POSIX совместимого интерпретатора аз. Функционально совместим с интерпретатором *bash*, и, за счёт отсутствия ряда функций, является более производительным (например, при обработке *shell*-сценариев).

Графический режим пользовательских сессий организован с помощью защищённой графической подсистемы *Fly.* Это обобщённое обозначение следующих функциональных компонентов GUI:

• *fly-wm(Fly Window Manager*) —менеджер окон , именуемый в Руководстве пользователя ОССН как «Рабочий стол *Fly*» и реализующий традиционную для менеджеров окон функциональность:

- поддержку множества рабочих столов (*Desktops*) и переключение между ними;
- доступ к запускаемым приложениям и функциям управления сессией с помощью меню «Пуск»;
- управление окнами GUI-приложений;
- управление переключением фокуса работы с GUI-приложениями на основе панели задач;
- управление блокировкой пользовательской сессии;

- управление оповещениями о событиях на основе области уведомлений (notification area);

•*fly-dm* и *fly-qdm (Fly Display Manager)*- дисплейный менеджер графической подсистемы *Fly* (представлен серверным *fly-dm* и клиентским компонентами). Реализует функции *GUI login*:

- запуск локальной или удалённой сессии учётной записи пользователя;
- процесс входа (идентификация и аутентификация) пользователя в систему.

Основой защищённой графической подсистемы *Fly* является графический сервер X.*Org Server* —реализация *X Window System* с открытым исходным кодом. Базовые библиотеки *Fly*, предоставляющие GUI-приложениям функции формирования требуемых визуальных компонентов GUI, базируются как на кроссплатформенной платформе *Qt5* [128], так и на функциональных решениях разработчиков ОССН:

• *flyui* — высокоуровневая библиотека компонентов, основанная на библиотеке *libqt5-gui* платформы *Qt5;*

• *kf5* — высокоуровневые библиотеки компонентов платформы *KDE Framework* (версия 5), также базирующиеся на библиотеке *ltbgt5-gui*;

• *flycor*e— библиотека, не зависящая от библиотек платформы *Qt5*, выполняющая сервисные функции (взаимодействие с *fly-wm*, управление абсолютными путями к служебным файлам рабочих столов, доступ к коллекциям иконок, обработка конфигурационных файлов GUI-приложений), являющаяся функциональным решением разработчика ОССН.

Благодаря правилам для создания приложений ОССН, включаемых в состав рабочего стола *Fly* или взаимодействующих с ним, вне зависимости от представленных выше высокоуровневых библиотек компонентов, появляется возможность добиться:

• единообразия внешнего вида GUI-приложений, их внутренней структуры и поведения;

• унификации сборки и установки GUI-приложений,

• простой модификации и сопровождения GUI-приложений.

Функциональность GUI компонентов *fly-dm* и *fly-qdm* расширяется функциями библиотеки *libparsec-mac-qt5-1*, предоставляющей доступ к функциям подсистемы безопасности *PARSEC* и содержащей набор классов для работы с мандатными атрибутами. На основе этих функций:

• процесс *GUI login* (приложения fly-dm и fly-qdm) получает возможность формирования диалога выбора атрибутов мандатных управления доступом и контроля целостности учётной записи пользователя при организации ее GUI-сессии;

• менеджер окон (приложение *fly-wm*) получает возможность создания диалога для установки атрибутов мандатных управление доступом и контроля целостности объектов

файловой систе-мы, а также графической визуализации мандатного контекста окон GUI-приложений;

• графический сервер X.*Org Server* получает возможность определения привилегий *XOrg* клиента (GUi-приложения) и передачи их с использованием модифицированного X-протокола менеджеру окон *fly-wm*, который и выполняет привилегированные операции в ходе запуска GUI-приложений с различными мандатными контекстами. При этом на рабочем столе *Fl*y выполняется отображение:

•мандатного контекста пользовательской сессии в области уведомлений (*notification area*);

• мандатного уровня каждого окна;

• мандатного уровня всех С(/1-приложений, размещённых на рабочем столе *Fly*;

•мандатного уровня окна для локально или удалённо запущена приложений (цветовое обрамление (рамка) окна приложения).

В табл. 1.4 представлены версии ядра *GNU/Linux*, базовых библиотек, приложений и средств разработки дистрибутива ОССН.

Кроме базовых компонентов, библиотек и средств разработки в состав дистрибутива ОССН входит общее ПО, обеспечивающее поддержку следующих функций:

Таблица 1.4

Релизы базовых компонентов, библиотек и средств разработки дистрибутива ОССН

Компоненты	Номер версии
Ядро *GNU/Linux (Linux Kernel)*	4.15.3
Системная библиотека *eglibc (Embedded glibc*)	2.24
Компилятор *gcc*	6.3.10
Отладчик *gdb*	7.12
Набор программ сборки объектных файлов *binutils*	2.28
Утилита автоматизации процесса компиляции *make*	4.1
Реализация протокола *LDAP openldap*	2.4.44
Криптографический пакет *OpenSSl*	1.10f-3
Командный интерпретатор *bash*	4.4
Система *X.Org*	1.19.6
	5.10.1

Библиотека GUI-фрейворка *Qt 5.0 libgt5* Менеджер окон *fly-wm*	2.13.11

• обработка документальной информации (текстовые, табличные редакторы и средства создания презентационных материалов и доступа к реляционным базам данных);

• сканирование, печать и передача документальной информации;

• доступ к информации, хранящейся в реляционных базах данных, включая поддержку ПО «1С» и ПО для работы с географическими объектами *PostGIS*;

• доступ к информации через сервер гипертекстовой обработки

данных (HTTP-сервер и клиент);

• обмен сообщениями электронной почты (SMTP/IMAP-серверы и клиент);

• работа с графикой и мультимедиа (звук, видео).

Наименование и версии перечисленных видов общего ПО представлены в табл. 1.5.

ОССН обеспечивает поддержку развёртывания, конфигурирования и эксплуатации развитых, масштабируемых, высокодоступных, отказоустойчивых сетевых инфраструктур, которые базируются на технологиях:

• централизованного управления сетевыми ресурсами и политиками безопасности на основе доменной архитектуры (*Domain networking*);

• виртуализации вычислительных и сетевых ресурсов (*Virtualization technology*);

• распределенных и/или параллельных вычислений на основе кластеров (*Cluster computing*) с целью: обеспечения высокой доступности сетевых сервисов (*High availability*) и/или повышенной производительности вычислительной системы (High-performance computing)

Таблица 1.5

Наименование и версии общего ПО дистрибутива ОССН

Наименование общего ПО	Версия общего ПО
Средства обработки документальной информации Офисный пакет *LibreOffice*	5.2.7
Средства сканирования ,печати и передачи документальной информации Комплекс программ *Hewlett-Packards Linux Imaging and Printing*	3.17.10 2.2.1
Средства доступа к информации ,хранящейся в реляционных базах данных СУБД *PostgreSQL*	9.2.14 и 9.6
Средства доступа к информации через сервер гипертекстовой обработки данных Web-сервер Apache2 Web-браузер Firefox	2.4.25 54.0.2 66.033.59.117

Web-браузер Chromium	
Средства обмена сообщениями электронной почты Сервер электронной почты Exim4 Сервер электронной поты Dovecot Клиент электронной почты Thunderbird	4.89 2.2.33 52.6.0
Средства мнговенного обмена сообщениями (instant messaging) XMPP клиент Psi+ Средства для работы с графикой и мультимедиа Редактор растовой графики GIMP Редактор векторной графики Inkspace Редактор трехмерной компьютерной графики Blender Набор звуковых драйверов и микшер ALSA Аудиоредактор AUDACITY Медиапроигрыватель VLC	1.3.425 2.8.18 0.92.1r15371 2.78 1.1.3 2.1.2 3.0.0

При проектировании сетевых инфраструктур указанные технологии могут применяться как по отдельности, так и в совокупности, что обеспечивает высокую гибкость их конфигурирования и модификации.

Особенностью использования указанных технологий в рамках ОССН является возможность формирования защищённых вариантов сетевых инфраструктур, использующих единую политику безопасности, базирующуюся на механизмах подсистемы безопасности *PARSEC*.

Взаимосвязь представленных технологий в рамках создания защищённых распределенных вычислительных инфраструктур и централизованного управления их ресурсами на основе единой

политики безопасности, поддерживаемой *PARSEC*, показана на рис.1.7

Базовой технологией являются сервисы формирования доменной

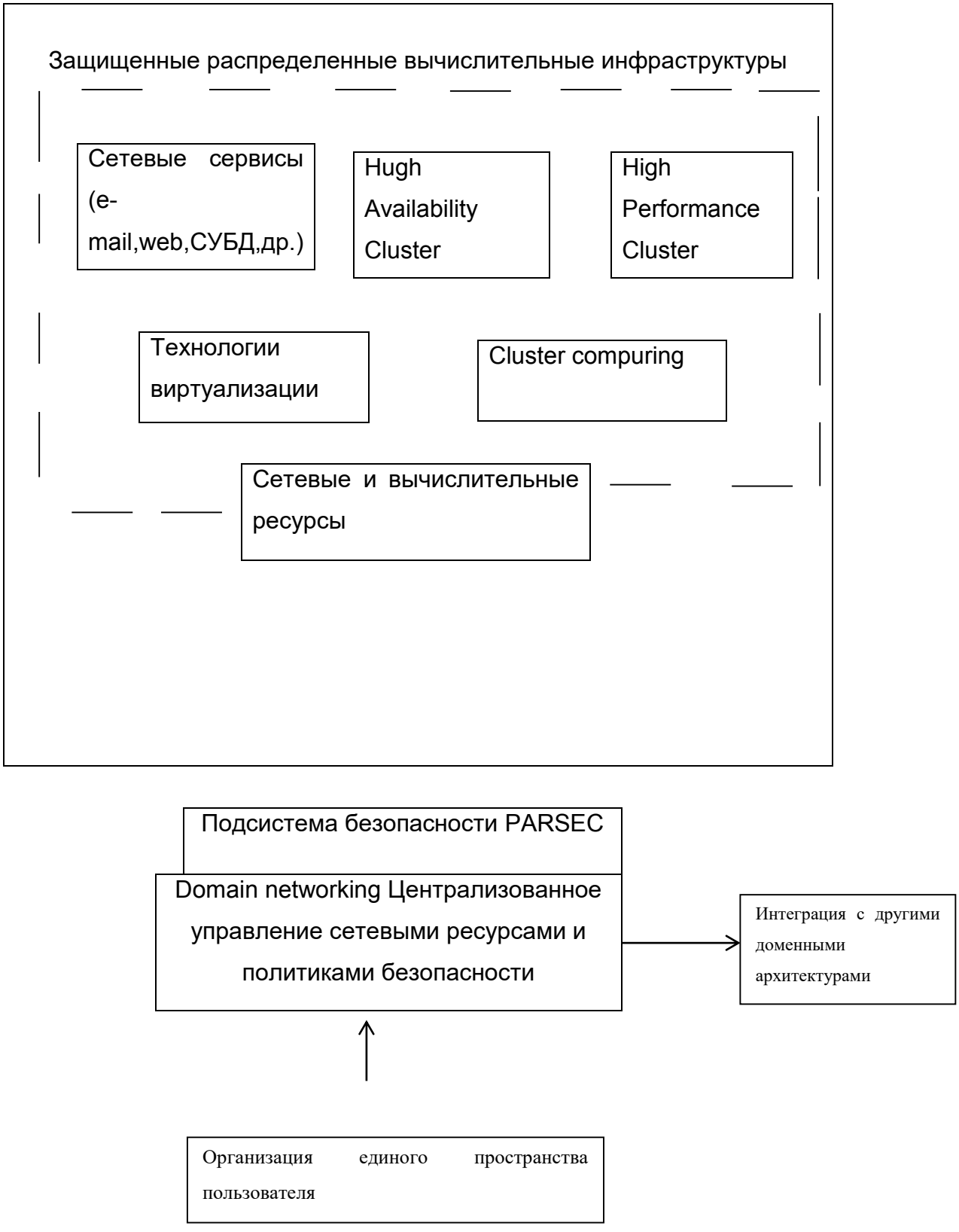


Рис. 1.7. Взаимосвязь технологий поддержки защищённых распределенных вычислительных и сетевых инфраструктур в ОССН

сетевой инфраструктуры. ОССН поддерживает несколько технологий её формирования, имеющих разное функциональное назначение:

• *Astra Linux Domain (ALD*) — традиционная для релизов ОССН доменная архитектура. Обеспечивает формирование защищённого единого пространства пользователей (ЕПП) с защищенной щённой авторизацией на контроллере домена и поддержкой политики безопасности хранимых в домене данных и потоков информации на основе функциональных возможностей подсистемы безопасности *PARSEC*. *И*нфраструктура *ALD* обеспечивает решение задач резервирования данных контроллера домена ALD и установления доверительных отношений между основными и резервным контроллерами. Она является функциональным решением разработчика ОССН и более подробно рассматривается в главе 3.

•*FreeIPA* (Free Identity, Policy *and Audit*) — доменная инфраструктура, формируемая в рамках проекта с открытым исходным кодом *FreeAPI*, доработанная разработчиком ОССН с целью поддержки политики безопасности хранимых в домене данных и потоков информация на основе функциональных возможностей подсистемы безопасности *PARSEC* [68]. Базовым отличием доменной архитектуры *FreeAPI* от архитектуры *ALD* является возможность её масштабирования при создании как доменной инфраструктуры с несколькими контроллерами домена, так и интеграции с уже существующими защищенными доменными инфраструктурами на базе *FreeAPI*, путём установления внутри-и междоменных доверительных отношений на основе функций *Forest Trust. FreeAPI* полностью поддерживает стандарты *Forest Trust* описанные в [96], и обеспечивает как формирование доверительных отношений внутри доменного леса, так и доверительных отношений между лесами доменов с поддержкой сквозной (через лес) аутентификации и авторизации (*Cross-Forest Authentication*). Более подробно возможности технологии *FreeAPI* в ОССН рассматриваются в главе 3.

• *Samba DC (Samba 4 DomainController*) — контроллер домена на основе пакета программ *Samba* (версии 4.x). *Я*вляется реализацией доменной инфраструктуры *Microsoft Active Directory Domain Services* для *Windows Server 2003/2008* [95] на базе ОССН с возможностью подключения клиентов домена на основе пользовательских и серверных сборок ОС семейства *Microsoft Windows.Позволяет* создавать объекты групповой политики (*GPO-Group Policy Objects*) для всех клиентов домена.

Одним из достоинств *Samba DC* является возможность использования в качестве инструментов управления доменными службами пакета инструментов *Microsoft Remote Server Administration Tools* (RSAT) [131], что делает её эффективным средством в случае необходимости выполнения миграции с доменной инфраструктуры *Microsoft Active Directory* на платформу ОССН.

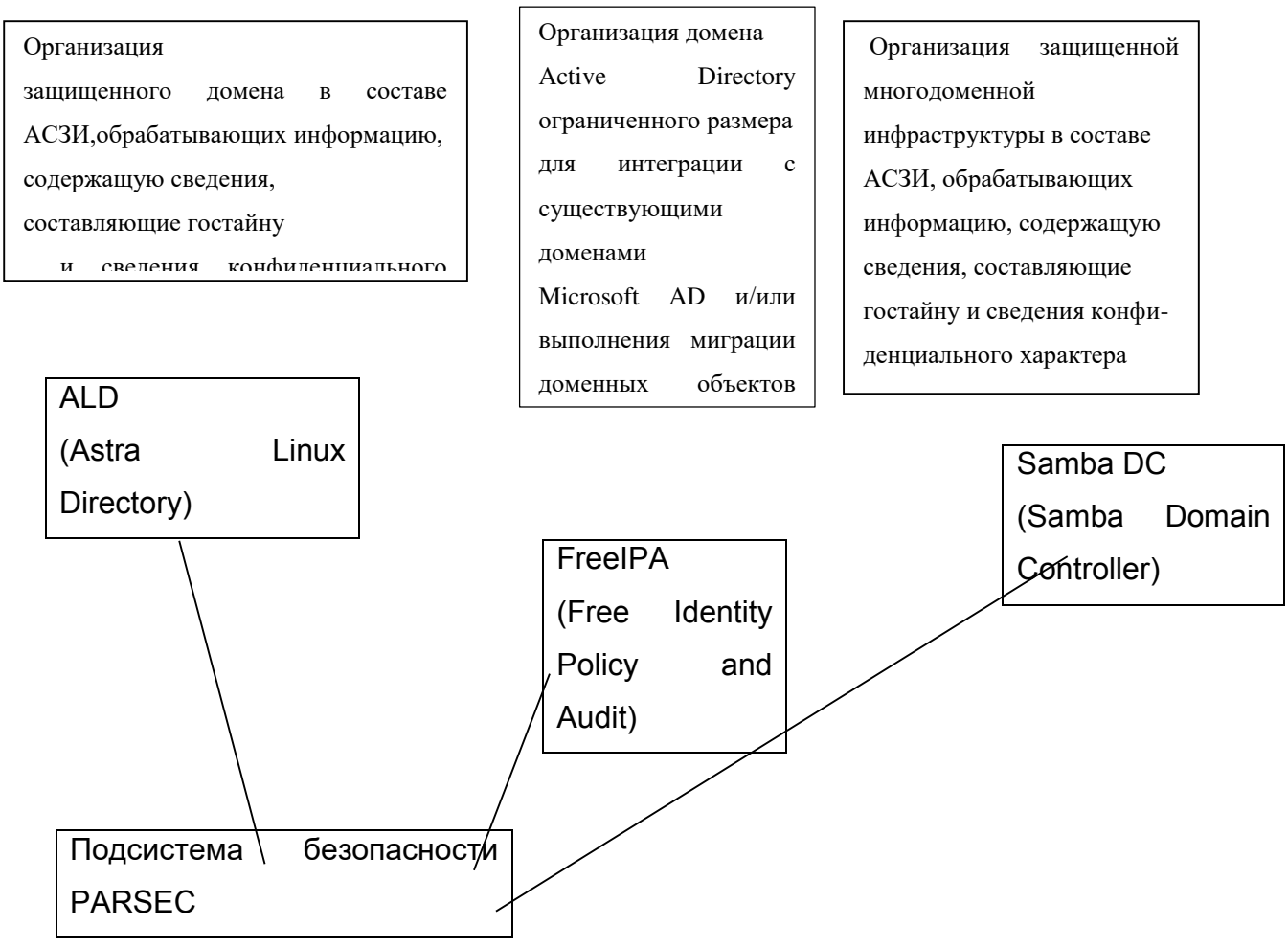
Технологии *Samba DC* присущи недостатки, ограничивающие функциональность её использования. К основным из них относятся:

• ограничения на число записей об объектах в каталоге *Active Directory*, связанные с 32-разрядной архитектурой *Trivial DataBase* (базы данных, по умолчанию используемой сервисами *Samba*), а также на размер файлов *tbd* в 4 гигабайта, что позволяет создавать не более десяти тысяч записей;

• отсутствие поддержки взаимодействия с контроллерами домена на базе *ОС Windows Server 2012* и *ОС Windows 2012 R2;*

• отсутствие поддержки *MIT Kerberos* и функции *SID Filtering*, что не позволяет организовать защищённую аутентификацию клиентов домена.

подробно возможности и ограничения технологии *Samba DC* в ОССН рассматриваются в главе 3.



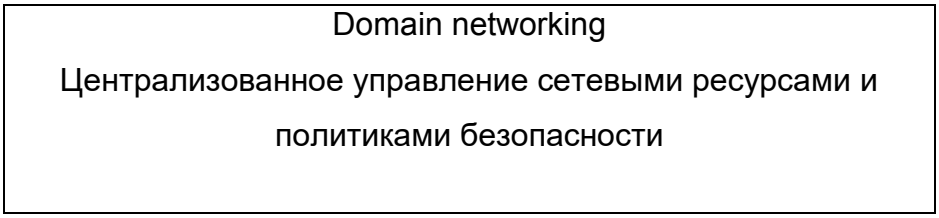


Рис. 1.8. Функциональность применения технологий доменной архитектуры на базе ОССН

Перечисленные технологии доменной инфраструктуры базируются на *OpenLDAP* — открытой реализации протокола *LDAP* [56]. Для формирования ключей шифрования, сертификатов открытых ключей и выполнения шифрования данных *SSL/TSL*, соединений в состав дистрибутива ОССН входит криптографический пакет *Open-SSL* [143].

В обобщённом виде функциональность применения технологий *ALD, FreeIPA и Samba DC* в составе распределенных инфраструктур на базе ОССН представлена на рис. 1.8.

Ещё одной особенностью рассматриваемого релиза ОССН версии 1.6 является включение в его состав технологий виртуализации, обеспечивающих возможность формирования локальных и распределенных виртуальных инфраструктур. Их использование существенно повышает эффективность применения АСЗИ с точки зрения масштабируемости, высокой доступности и отказоустойчивости сетевых и вычислительных ресурсов.

Основой подсистемы виртуализации в ОССН является технологический стек, состоящий из следующих уровней.

1. Доступ к аппаратному обеспечению физического хоста (в терминологии стека — узла *Node*). Этот уровень в ОССН представлен программным *обеспечением KVM (Kernel-based Virtual Machine)*[99],обеспечивающим поддержку технологий аппаратной виртуализации *Intel TV (Virtualization Technology*) и *AMD SVM (Secure Virtual Machine*) [135]. Реализован в виде модулей ядра *kvmko (*базовый сервис виртуализации),*kvmintelko,kvmamdko*(процессорно-специфические сервисы виртуализации)

2.Эмуляция аппаратного обеспечения для гостевых ОС (в терминологии стека — доменов). Представлен модификацией гипервизора *QEMU*, представляющим собой компонент пользовательского режима *KVM*, который использует интерфейс уровня доступа к аппаратному обеспечению узла для настройки адресного пространства гостевых доменов и эмуляций видеоадаптеров и устройств ввода-выводы.

3.Управления виртуализацией.Представлен набором инструментов *libvirt* включающим:

•библиотеку API виртуализации на базе проекта с открытым исходным кодом [122];

•сервис (демон) *libvirtd*;

•CLI-интерфейс пользователя *virsh.*

Взаимодействие уровня управления виртуализацией и гипервизора *QEM*U осуществляет с использованием протокола *QMP(QEMU machine Protocol).*

Указанные уровни стека виртуализации ОССН могут использоваться как для локального создания и управления, так и для защищенного удалённого управления

гостевыми доменами. В последнем случае в качестве протоколов удалённого доступа к рабочему столу гостевого домена могут применяться:

•*RFB(Remote Frame Buffer)* системы *VNC*

•*RDP(Remote Desktop Protocol)*

•*SPICE(Simple Protocol for Independent Computing Environments)*

Сервис *libvirt*d поддерживает следующие технологии защищенной идентификации и аутентификации для доступа к ресурсам гостевых доменов:

•набор средств предоставления сервиса аутентификации *SASL (Simple Authentication and Security Layer),*которые в ОССН реализован с поддержкой сетевого протокола аутентификации *Kerberos* и более подробно рассматривается в главе 3;

•система аутентификации на уровне сервиса *SS*H,использующая алгоритмы электронной подписи *RSA* и *DSA*

•протокол защиты транспортного уровня *TLS(Transport Layer Sec*urity),базируется на спецификации протокола SSL 3.0.

<table>
<tr><td colspan="2"></td><td>Гостевая ОС (Домен1)</td></tr>
<tr><td colspan="2"></td><td>Гостевой агент QEMU</td><td>Гипервизор QEMU</td></tr>
<tr><td>Сервис libvirtd</td><td>Управляющее приложение virsh</td><td>Драйвер гипервизора QEMU</td></tr>
<tr><td colspan="2">Libvirt API</td><td>Интерфейс ABI</td></tr>
<tr><td colspan="2">Хостовая ОС(домен0)-ОССН</td><td>Модуль kvm.ko</td></tr>
<tr><td colspan="3">Физический хост-узел (Node)</td></tr>
</table>

Рис. 1.9. Локальный стек виртуализации на базе ОССН

В обобщённом виде схемы локального и удалённого управления стеком виртуализации ОССН представлены на рис. 1.9 и 1.10.

Повышение эффективности использования рассмотренной выше технологии виртуализации реализуется в ОССН включением средств организации высокомасштабируемых *(HS — High Scalability*) отказоустойчивых кластерных систем, в том

числе поддерживающих режим высокой доступности (*HA — High Availability*). Основой этих кластерных систем являются следующие проекты свободного ПО, модифицированные разработчиком ОССН с целью поддержку функций подсистемы безопасности *PARSEC:*

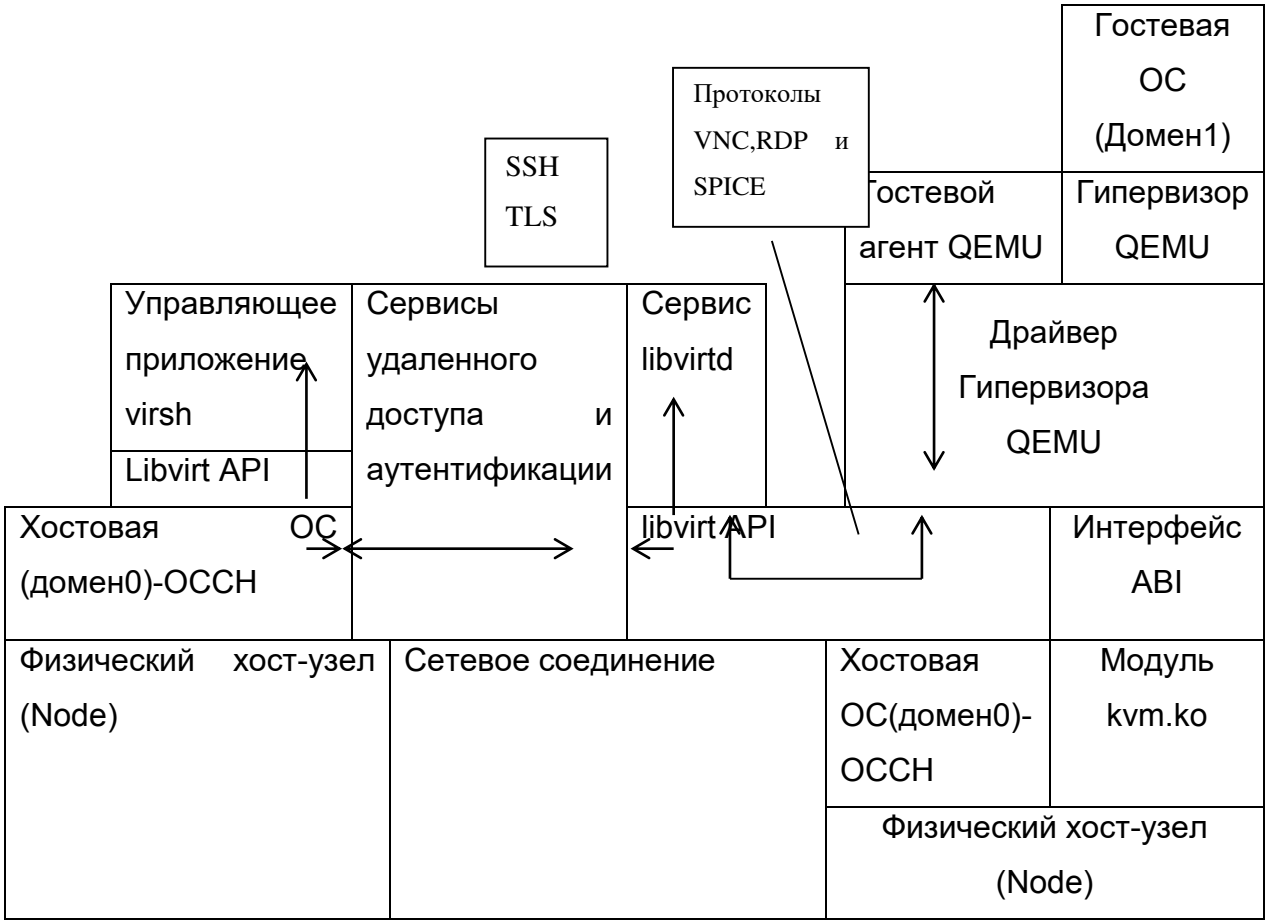


Рис. 1.10. Схема удаленного управления стеком виртуализация на базе ОССН

Pacemaker- продукт сообщества разработчиков *ClusterLabs* [44]. Он является менеджером ресурсов кластера высокой доступности и представлен модулями ПО, выполняющимися на узлах кластера и реализующими функцию минимизации времени простоя требуемых сервисов (в терминах технологии — ресурсов). В рамках этой функции *Pacemaker* решает следующие задачи:

• обнаружение и восстановление после сбоев на уровне узлов кластера и выполняющихся сервисов,

• обеспечение целостности данных путём блокирования отказавших узлов кластера;

• поддержка разнообразных стандартов интерфейса ресурсов (реализация управления ресурсами на уровнях сценария на большинстве доступных скриптовых языков);

• поддержка (опциональная задача) распределенного общего хранилища данных;

• поддержка разнообразных схем резервирования (active/passive,N+1 и тд.);

• поддержка автоматически реплицируемой конфигурации, которая может быть обновлена с любого узла кластера;

•поддержка внутрикластерных отношений между сервисами;

• поддержка расширенных типов сервисов, таких как клоны ресурсов (сервисы, которые должны быть активны на нескольких узлах кластера), ресурсы с состоянием (клоны, которые могут работать в одном из двух режимов) и контейнерные сервисы.

Corosync- (Corosync Cluster Engine) — продукт сообщества разработчкиво Kernelorg,реализующий коммуникационные возможности кластерной системы [43]. Функциями *Corosync* являются:

• отслеживание состояния сервисов, функционирующих в рамках кластера;

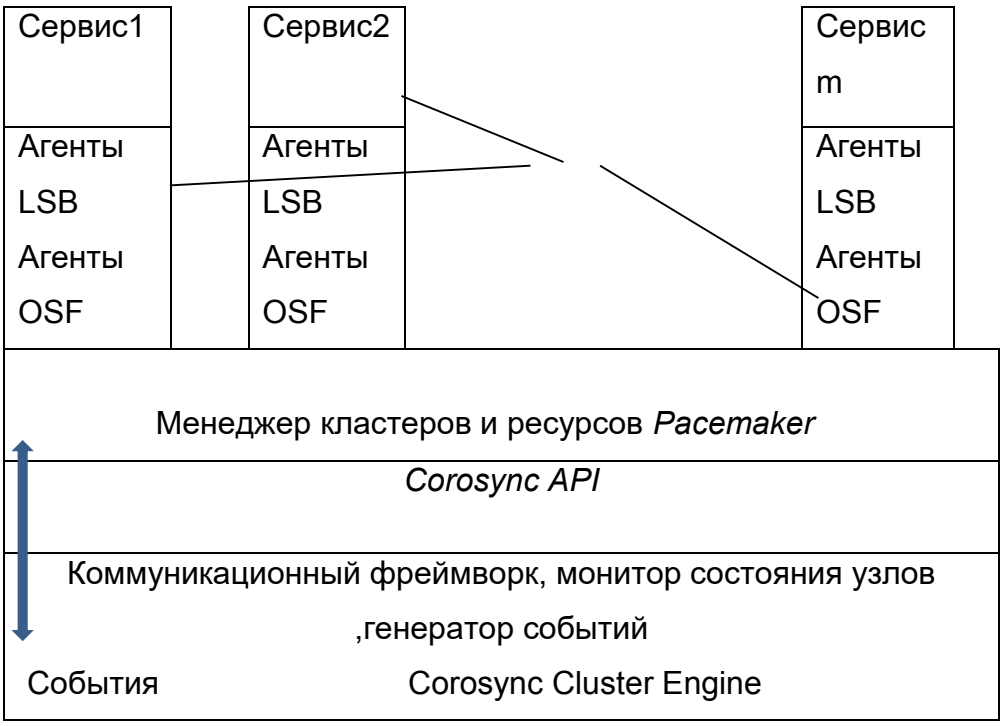
• оповещение сервисов о смене активного узла кластера;

• отправка одинаковых сообщений сервисам на всех узлах кластера;

• предоставление администратору кластера доступа к базе данных с конфигурацией и статистикой, а также отправка уведомлений о ее изменениях.

В обобщённом виде схема отказоустойчивого кластера на основе продуктов *Pacemaker* и *Corosync* представлена на рис. 1.11.

Сервисы, относительно которых организуется отказоустойчивый кластер, фактически являются кластеронезависимым уровнем. Их важной особенностью является возможность управления (старт,



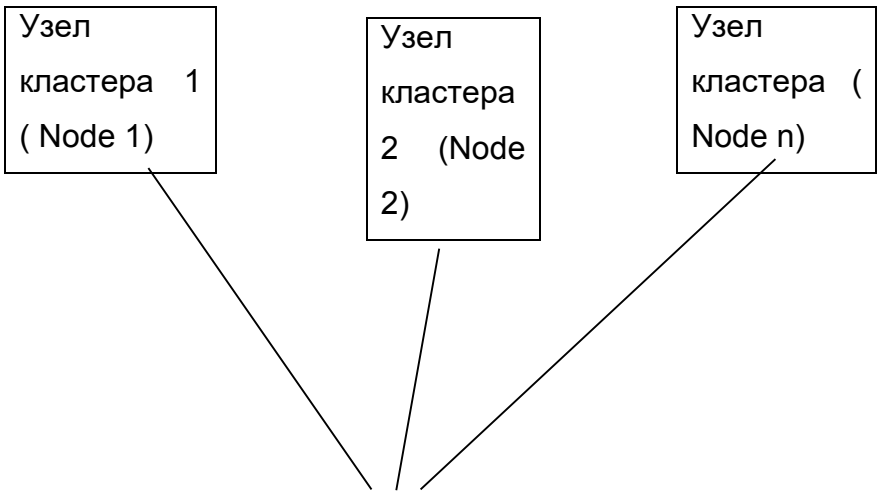


Рис.1.11. Схема организации отказоустойчивого кластера на основе продуктов Pacemaker и Corosync

остановка, перезапуск, мониторинг состояния) с помощью сценариев, именуемых агентами сервисов. В зависимости от спецификации, применяемой для формирования агентов, они делятся:

• на агенты *LSB* ресурсов — сценарии, поддерживающие *Linux Standart Base Core Specification* (размещаются в стандартном для этой спецификации каталоге инициализации */etc/init.d*);

• на агенты OCF ресурсов — сценарии, поддерживающие спецификацию Open Connectivity Foundation , (размещаются в стандартном для этой спецификации каталоге инициализации */ usr/lib/ocf/resource.d/provider*).

Совокупность сервиса и управляющего им агента (агентов) с точки зрения менеджера кластеров *Pacemaker* является ресурсом кластера. Вне зависимости от применяемой спецификации агентов ресурсы кластера можно объединить в группу ресурсов, под которым понимается логическое объединение (список) ресурсов, функционирующих на одном узле кластера, которые инициализируются и останавливаются в определённом порядке. Таким образом, при сбое или отказе одного из ресурса входящих в группу, эта группа “перемещается” (инициализируется) на другой узел кластера.

В рамках предоставленной выше инфраструктуры отказоустойчивого кластера в ОССН включены средства поддержки высокой доступности *HA* за счет введения избыточности к точке входа в тот или иной сервис устраняющие тем самым единую точку отказа (*SPOF, Single Point Of Failure).* Средства поддержки режима высокой доступности позволяют сервисам автоматически перезапускать или перенаправлять задачи на другой узел кластера (ведомый узел) в случае сбоя на ведущем узле, поддерживающем данный сервис. Таким образом, в составе кластера должен быть компонент мониторинга возникающих на ведущем узле ошибок и сбоев и перенаправления задач с ведущего узла на ведомый узел и обратно в случае восстановления функциональности ведущего узла.

Таким компонентом в ОССН является *keepalived*— демон поддержки мониторинга сервисов кластера и автоматического перенаправления трафика запросов/откликов ведомому узлу кластера в случае, если ведущий узел не отвечает. То есть *keepalived* выступает в роли монитора сбоев и балансировщика нагрузки (*load balancer*) в кластере с избыточностью (ведущим и ведомым узлами одинаковой функциональности) — *HA*-кластере.

Основой функционирования демона *keepalived* является плавающий (виртуальный) IP-аадрес (*Floating IP, VIP — Virtual IP*) —публичный статический IP-адрес, присвоенный одному из узлов кластера. Он не подменяет основной публичный IP-адрес узла, а является дополнительным адресом, с помощью которого можно получить доступ к узлу, за которым этот адрес закреплён. Фактически VIP выполняет функцию шлюза, с помощью которого учётные записи пользователей взаимодействуют с сервисом, развёрнутым в HA-кластере. В обобщённом виде схема HA-кластера, основанного на использовании *keepalived*, представлена на рис. 1.12.

Функционирование сервисов в рамках отказоустойчивой, высокодоступной кластерной инфраструктуры, в большинстве случаев, требует развёртывания соответствующей системы хранения данных. К ней,также как и к механизмам управления сервисами на уровне кластера, предъявляются требования по масштабируемости, надёжности хранения и высокой доступности данных.

С целью создания подобной системы хранения данных ОССН одержат средства формирования распределенной файловой системы на основе проекта с открытым исходным кодом *Ceph*[69], который базируется на программно-определяемой объектно-ориентированной файловой системе, создаваемой поверх существующей системы хранения данных кластерной инфраструктуры. Основными функциями *Ceph* являются:

• распределенное в рамках узлов кластера хранение данных пользователей и сервисов, прозрачное с точки зрения доступа к ним;

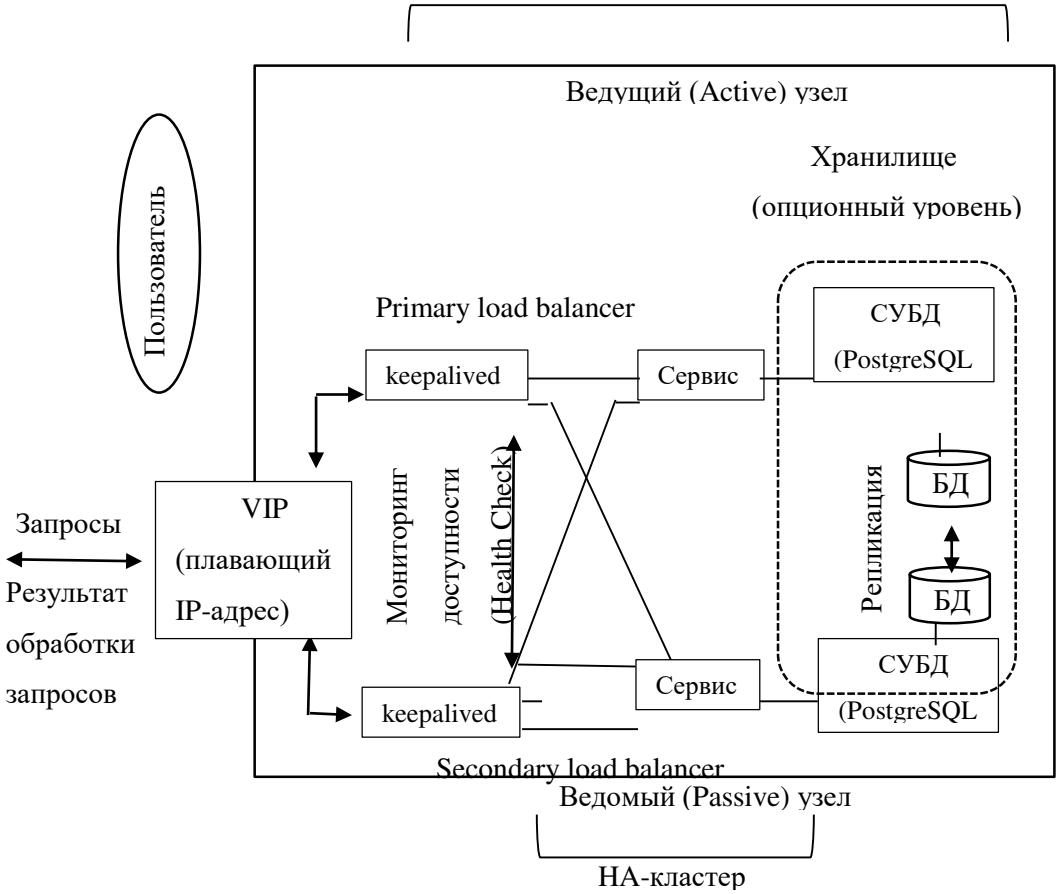


Рис. 1.12. Схема организации HA-кластера на основе сервиса keepalived

- репликация данных между локальными системами хранения данных узлов кластера с целью повышения отказоустойчивости системы хранения и обеспечения высокой доступности данных;

- распределение нагрузки между локальными системами хранения данных узлов кластера с целью обеспечения высокой доступности данных.

Как объектно-ориентированная система *Ceph* определяет единицу хранения данных как объект, который на нижележащих уровнях системы хранения узлов кластера отображается в традиционный стек системы хранения, основанный на таких логических единицах хранения, как дисковые разделы *(partitions)* и кластеры *(clusters)* файловых систем различных типов. Подобный подход позволяет абстрагировать алгоритмы функционирования *Ceph* от реальной организации систем хранения на отдельных узлах кластера, обеспечивая поддержку гетерогенных с точки зрения локальных файловых систем узлов.

Логической единицей, определяющей хранилище данных в *Ceph,* является объектное устройство хранения *OSD (Object Storage Device).* Функции *OSD* включают:

- хранение данных, представляемых объектами *Ceph]*
- обработку пользовательских запросов на доступ и манипулирование данными;
- взаимодействие с другими *OSD* в рамках решения задач доступа к данным, распределенных между несколькими *OSD,* их репликации и восстановления, в случае сбоя и отказа *OSD.*

OSD представлен:

- физическим устройством хранения данных, подключённым к узлу кластера;
- процессом-демоном, управляющим доступом к этим данным.

Физическим устройством, чаще всего, является диск, но может использоваться и HA/D-массив или сетевое хранилище *(NAS)* на базе протокола *iSCSI.* Пространство устройства хранения при этом разделяется на две части: раздел журнала (функция журналирования транзакций с объектами является аналогичной таковой в традиционных файловых системах с журналированием) и раздел хранения данных, в котором размещаются объекты.

Следующим логическим уровнем *Ceph* является группа хранилищ *(PG — Placement Group)* — совокупность взаимосвязанных, с точки зрения задач репликации данных, *OSD.* Число *OSD* в группе хранилищ определяется параметром «фактор репликации». При этом вне зависимости от значения этого параметра один из *OSD* является первичным при размещении хранимых объектов, а остальные — OSD-репликами, в которых размещаются реплицируемые копии хранимых на первичном *OSD* объекты. Сбой или отказ любого | *OSD* приводит к исключению его из всех групп хранилищ, в которых он был зарегистрирован. При этом система управления Ceph выберет новый работоспособный *OSD* и благодаря наличию реплик объектов в других *OSD* группах хранилищ восстановит на нем состав хранимых объектов. При этом любой *OSD* может одновременно принадлежать нескольких группам хранилищ, являясь для одних их них первичным *OSD*> а для других OSD-репликой.

Более высоким логическим уровнем *Ceph* является пул *(Pool),* представленный совокупностью групп хранилищ, объединённых в соответствии с типом хранимых данных (например, образы виртуальных машин облачной инфраструктуры или данные, поддерживаемые сервисами хостинга). Параметрами, определяющими организацию конкретного пула, являются:

- фактор репликации (параметр, аналогичный такому же для от дельной группы хранилищ);
- минимальное число доступных («живых») реплик объекта, не обходимое для получения клиентом *Ceph* подтверждения успешной записи объекта в инфраструктуре *Ceph;*
- политика репликации объектов в пуле, определяющая инфраструктурный уровень репликации (например, в рамках отдельных *OSD* нескольких серверных стоек одного центра обработки данных, нескольких таких центров).

Благодаря подобной многоуровневой логической структуре с гибко настраиваемым фактором репликации объектов логика хранимых в объектах *Ceph* данных будет максимально соответствовать требованиям инфраструктур отказоустойчивых и высокодоступных кластеров, рассмотренных выше.

Система управления *Ceph* представлена множеством мониторов, реализуемых соответствующими процессами-демонами. Монитор — логический узел *Ceph,* хранящий карту доступных в данный момент *OSD (CRUSH-карту).* Эта карта формируется на основе логической (центры обработки данных, залы, ряды, стойки, узлы) и физической (диски и/или другие устройства хранения данных) структур кластера, в рамках которого развёртывается *Ceph.* При этом карта не содержит конкретные пути к объектам, хранимым в *OSD.* Клиент *Ceph* формирует запросы к монитору с целью получения карты доступных *OSD,* а доступ к объектам конкретного *OSD* клиент выполняет самостоятельно с использованием протокола *CRUSH (Controlled Replicated Under Scalable Hashing),* однозначно идентифицирующего каждый хранимый объект на основе хеша его имени.

В зависимости от масштаба хранилища на базе *Ceph* поддерживается один или более мониторов, распределяющих между собой нагрузку по обработке запросов клиентов. Согласованность карты доступных *OSD* поддерживается протоколом достижения консенсуса, для работоспособности которого требуется нечётное число мониторов. При невыполнении этого условия хранилище на базе *Ceph* блокируется во избежание рассогласования реплик *CRUSH-карт.* Мониторы могут выполняться на тех же узлах кластера, которые поддерживают процессы-демоны *OSD,* или же для их выполнения могут использоваться выделенные узлы кластера.

Клиенты *Ceph* могут не использовать логику организации хранилища на основе объектов, а получать доступ к данным на основе традиционных запросов к файловым системам. С целью поддержки такой возможности *Ceph* реализует уровень файловой системы *CephFS (Ceph File System),* для обработки запросов в котором используются *MDS (Meta Data Server)* — узлы *Ceph,* поддерживающие метаданные уровня *CephFS.* Драйвер файловой системы *CephFS* реализуется ядром ОССН, что позволяет монтировать ее, например, с помощью механизма *FUSE (Filesystem in USErspace).*

В обобщённом виде структура распределенного хранилища данных на основе проекта *Ceph* представлена на рис. 1.13.

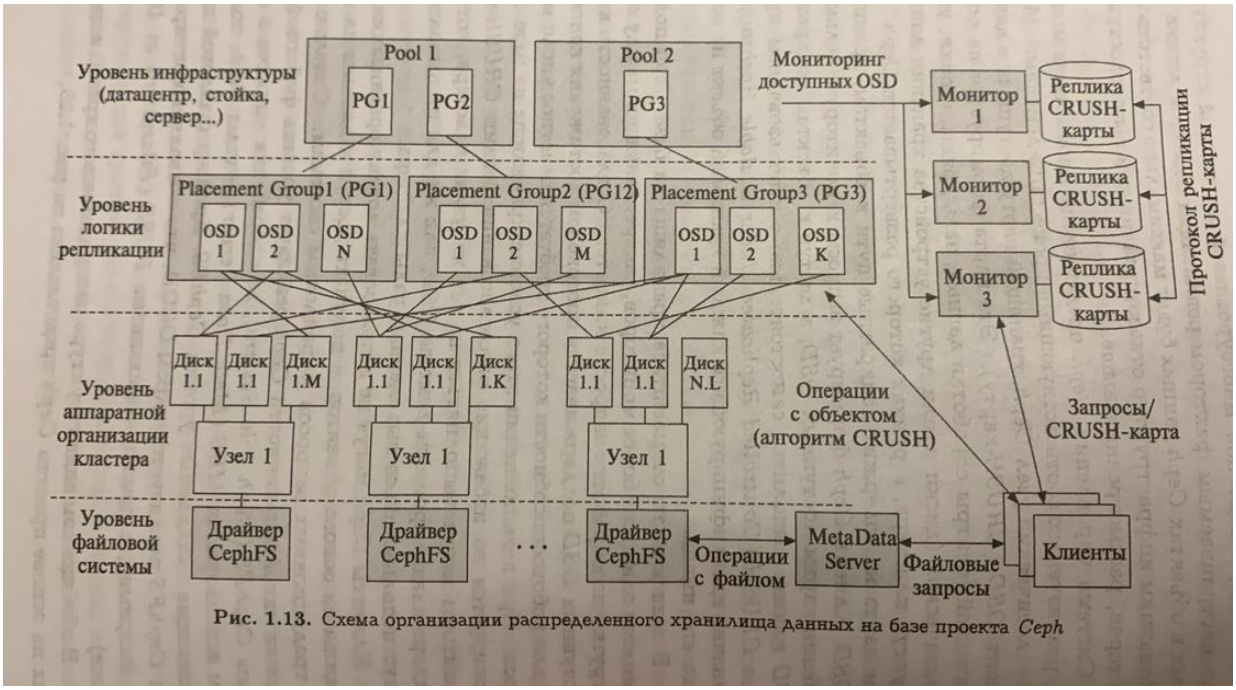


Рис. 1.13. Схема организации распределенного хранилища данных на базе проекта *Ceph*

Для администрирования распределенной системы хранения данных на базе проекта *Ceph* в составе дистрибутива ОССН имеется CLZ-утилита *cephdeploy,* устанавливаемая на одном из узлов- мониторов или на отдельно выделенном узле. Функциональность утилиты *cephdeploy* позволяет конфигурировать уровни инфраструктуры и логики репликации, а также обращаться к мониторам *Ceph* для получения актуальной информации об активности *OSD,* входящих в состав *Ceph.* При этом необходимо учитывать, что ОССН не поддерживает развёртывание компонентов *Ceph* при активном режиме мандатного контроля целостности на узлах кластера.

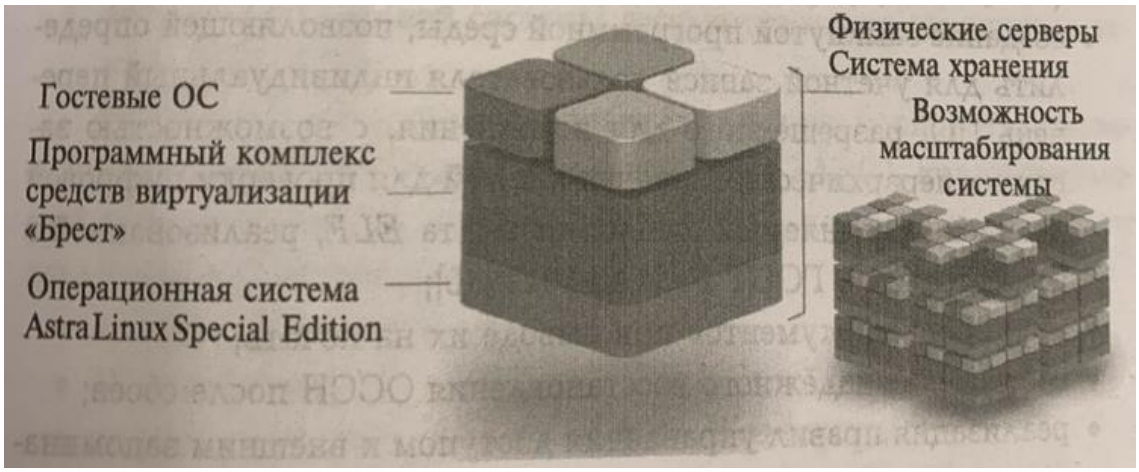
В рамках рассмотренных выше средств развёртывания масштабируемых отказоустойчивых высокодоступных инфраструктур ОССН может использоваться в качестве платформы программного комплекса средств виртуализации (ПК СВ) «Брест», который содержит:

- средства управления средой виртуализации и кластерами, сформированными рассмотренными выше средствами, входящими в состав ОССН;
- средства мониторинга состояния кластеров и развёрнутой инфраструктуры виртуальных машин и распределенных хранилищ данных.

При этом ПК СВ «Брест» выступает в роли платформы конфигурирования и эксплуатации средств, включённых в состав ОССН. Структурно взаимодействие компонентов ПК СВ «Брест» и ОССН представлено на рис. 1.14.

Таким образом, ключевой особенностью дистрибутива ОССН является то, что в его состав входят механизмы защиты, обеспечивающие реализацию следующих функций:

Рис.1.14.Организация масштабируемой инфраструктуры на основе ОССН и ПК СВ “Врест”



- аутентификация пользователей с использованием инфраструктуры *PAM (Pluggable Authentication Modules)* [123] локально или в рамках ЕПП, а также двухфакторная аутентификация на основе цифровой подписи и инфраструктуры открытых ключей, поддерживаемых внешним носителем аутентификационной информации «Рутокен» [7];
- идентификация пользователей с использованием модульного окружения *NSS (Name Service Switch)* локально или в рамках ЕПП [80];

- дискреционное управление доступом процессов к ресурсу поддержкой стандартов *Minimal ACL* и *Extended ACL* [127];
- мандатные контроль целостности и управление доступом процессов к ресурсам на основе МРОСЛ ДП-модели, реализуемое на уровнях механизма межпроцессного взаимодействия, включая файловые системы *proc* и *tmpfs,* стек протоколов *TCP/IP (IPv4),* на уровне виртуальной файловой системы *(VFS)* ив файловых системах семейства *EXTFS (Ext2, Ext3, Ext4);*
- изоляция адресных пространств процессов;
- регистрация (протоколирование) и аудит событий, реализованные в виде централизованной системы с функцией оповещения администратора безопасности о попытках несанкционированного доступа;
- очистка оперативной памяти и освобождаемых областей данных на запоминающих устройствах с файловыми системами *Ext2, Ext3, Ext4*;
- регламентный контроль целостности сущностей файловой системы, в том числе неизменности исполняемых файлов и соответствия дистрибутива ОССН на основе библиотеки *lxbgost,* в которой реализован алгоритм хеширования ГОСТ Р 34.11-201- («Стрибог») [11];
- создание замкнутой программной среды, позволяющей определить для учётной записи пользователя индивидуальный перечень ПО, разрешённого для исполнения, с возможностью загрузки иерархических цепочек ключей для проверки цифровой подписи исполняемых файлов формата *ELF,* реализованной в соответствии с ГОСТ Р 34.10-2012 [10];
- маркировка документов при выводе их на печать;
- обеспечение надёжного восстановления ОССН после сбоев;
- реализация правил управления доступом к внешним запоминающим устройствам;
- обеспечение доступа к реляционным базам данных;
- обеспечение доступа к информации через сервер гипертекстовой обработки данных;
- обеспечение обмена сообщениями электронной почты.

Основные из перечисленных механизмов защиты детально рассмотрены в главе 3, а также в лабораторном практикуме, приведённом в главе 4 настоящего учебного пособия.

1.3.3. Области применения ОССН

Рассмотренные архитектурные особенности ОССН, включающие поддерживаемые сервисы, средства создания масштабируемых отказоустойчивых высокодоступных инфраструктур, в том числе и с применением средств виртуализации, а также

интегрированный в состав ОССН комплекс механизмов защиты определяют основные области её применения:

- автоматизированные системы в защищённом исполнении;
- программно-технические комплексы и комплексы средств автоматизации;
- корпоративные сети;
- территориально-распределенные автоматизированные системы.

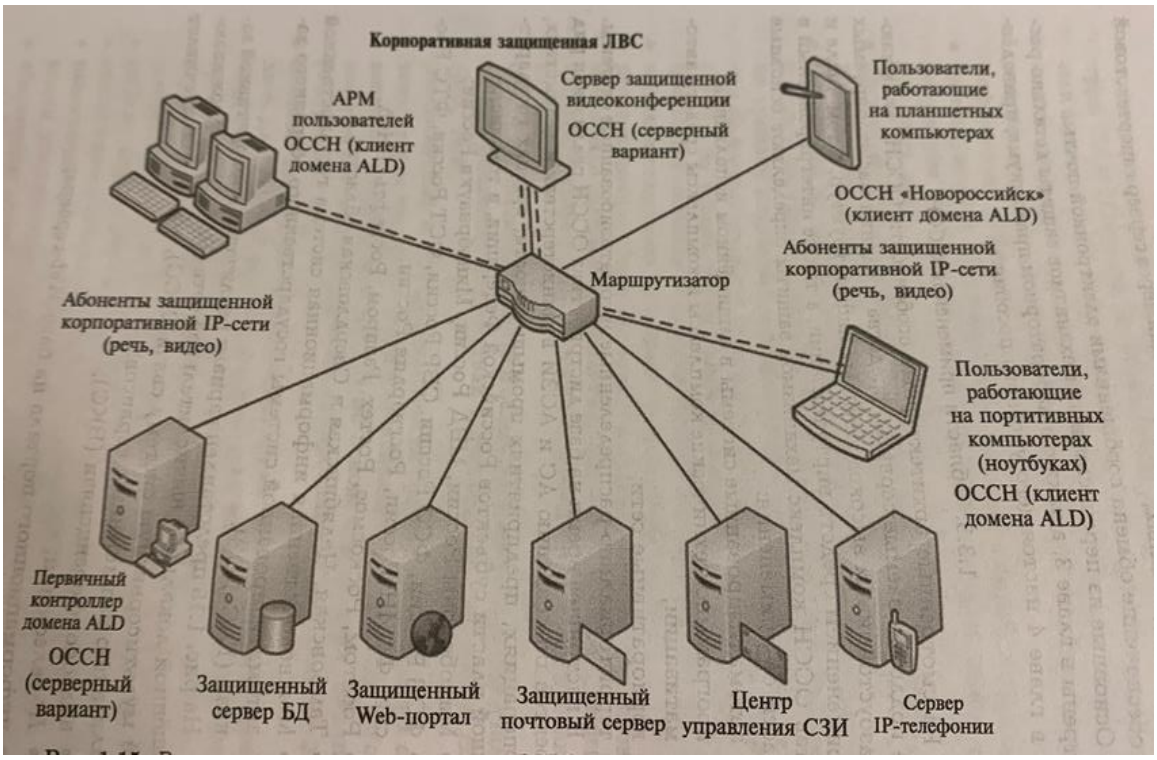
В настоящее время на базе дистрибутива ОССН реализован ряд проектов по созданию АС и АСЗИ в министерствах и ведомствах, корпорациях, предприятиях промышленности, органах государственной власти субъектов Российской Федерации, в том числе:

- Минобороны России, МВД России, Минобрнауки России;
- ФСБ России, ФСО России, СВР России, ФСТ России, ФТС России, ФСИН России, Росгвардия России;
- Росатом, Роскосмос, Ростех, Газпром, Роснефть, РЖД;
- Тамбовская, Челябинская и Свердловская области;
- Межведомственная информационная система государственной автоматизированной системы государственного оборонного заказа (ГАС ГОЗ).

На рис. 1.15 представлен вариант реализации корпоративной защищённой локальной вычислительной сети (ЗЛВС), поддерживающей мультисервисную систему связи (МСС), которая обеспечивает реализацию защищённых сервисов:

- видеоконференцсвязи (ВКС);
- IP-телефонии;
- информационного портала на базе Web-сервера,
- сервера баз данных;
- почтового сервера.

Рис.15. Вариант реализации корпоративной ЗЛВС для мультисервисной системы связи на базе ОССН



Указанные виды сервисов создаются на базе серверных платформ, которые функционируют в серверном варианте развертывания ОССН.

В качестве клиентских компонент такой ЗЛВС выступают:

- стационарные и мобильные компьютеры с процессорной архитектурой *Intel* X86-64, которые функционируют в клиентском варианте установки ОССН релиза «Смоленск»;
- планшетные компьютеры с процессорной архитектурой *ARM* функционирующие в клиентском варианте установки ОССН релиза «Новороссийск».

Для абонентов корпоративной ЗЛВС организуется ЕПП на базе доменных инфраструктур *ALD* или *FreeIPA* с выделенным контроллером домена, функционирующим в серверном варианте развёртывания ОССН, например, релиза «Мурманск».

Возможности ОССН позволяют организовать многоуровневые масштабируемые высоконадежные системы обработки информации обеспечивающие поддержку как контура, в котором циркулирует открытая информация, так и контура, в котором обрабатывается информация различного уровня конфиденциальности, включая информацию с грифом секретности до «совершенно секретно» включительно. Пример организации подобной двухконтурной системы обработки информации представлен на рис. 1.16.

Таким образом, комплект релизов ОССН обеспечивает возможность создания как АСЗИ на различных программно-аппаратных платформах, так и различных конфигураций защищенных компьютерных сетей, включающих, в том числе облачные технологии и механизмы защищенного удаленного доступа.

Также ОССН все более активно используется в образовательном процессе по направлению «Информационная безопасность», так как её механизмы защиты являются

удобной лабораторной базой для изучения теории и практики по этому направлению, например мандатных управления доступом и контроля целостности. Для этого АО «НПО РусБИТех» осуществляет безвозмездную передачу ОССН осуществляющим подготовку специалистов по защите информации образовательным организациям страны.

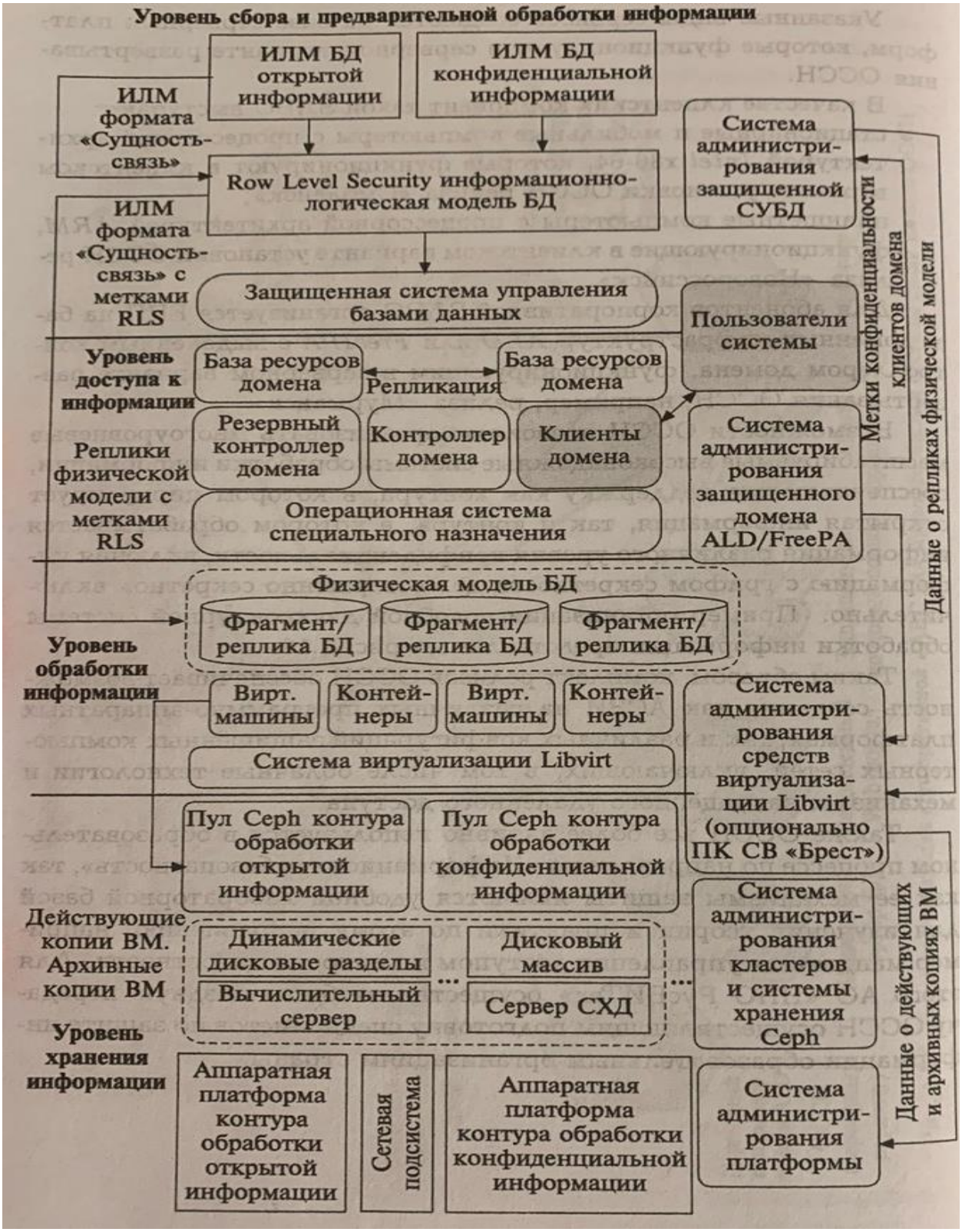


Рис.1.16. Вариант организации двухконтурной масштабируемой высокодоступной системы информации на базе ОССН

1.4. Основы пользовательской работы и администрирования в ОССН

При включении питания компьютера с установленной на нем $_0$ССН после этапа процедуры *POST* системы *BIOS/EFI* отображаем экран начального загрузчика *GNU GRUB 2 bootloader* (рис. 1.17). По умолчанию ОССН может быть загружена в двух основных режимах:

- *Hardened* — загрузка *Hardened*-ОССН, особенности которого рассмотрены в предыдущем разделе;
- *Generic* — загрузка базового ядра ОССН.

В режимах *Hardened* и *Generic* можно выбрать загрузку *recovery mode* (восстановления), в котором выполняется сценарий с проверкой ошибок доступа к корневой файловой системе или остановки процесса загрузки из-за ошибок в каком-либо сервисе. При этом режим *recovery mode* предоставляет администратору ОССН (в данном случае пользователю от имени учётной записи *root)* интерфейс *CLI* для запуска утилит диагностики и восстановления. Следует помнить, что в случае, когда учётная запись пользователя *root* заблоки-

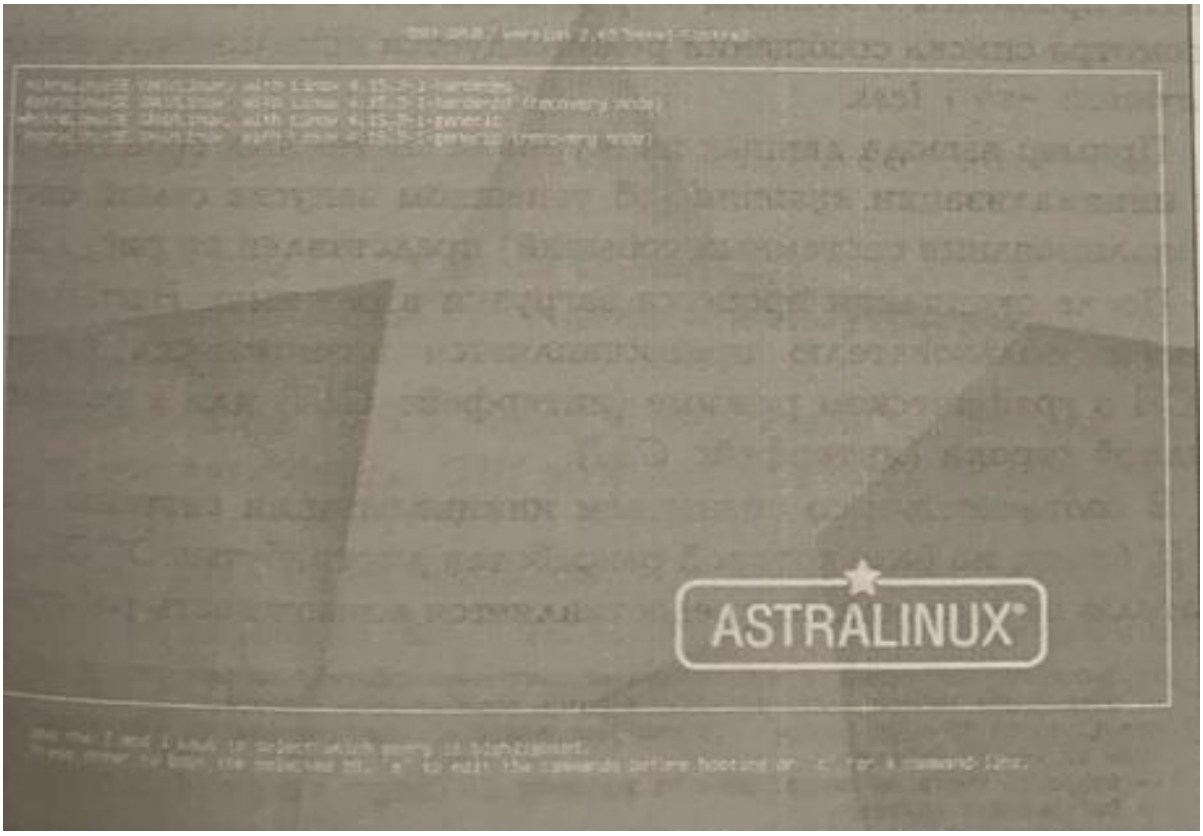


Рис. 1.17. Экран загрузчика *GNU GRUB 2*

You are in recue mode. After logging in, type “journalctl –xd” to view system logs, “systemctl reboot” to reboot, “systemctl default” or ^D to boot into default mode.

Cannot open access to console, the root account is locked. See sulogin(8) man page for more details.

Press Enter to continue.

Рис. 1.18. Предупреждение сценария загрузки *recovery mode* о невозможности доступа к консоли суперпользователя, при блокировке учетной записи *root*

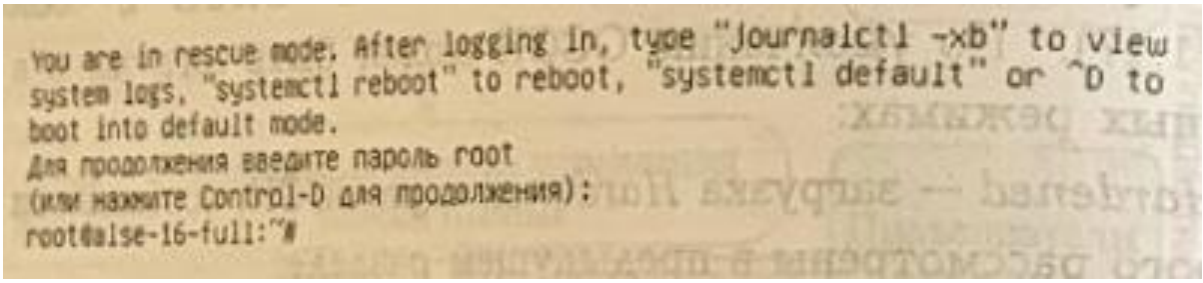


Рис. 1.19. Консоль суперпользователя при загрузке в режиме *recovery mode*

-рована (в ОССН версии 1.6 это сделано по умолчанию), доступ к консоли *recovery mode* невозможен и загрузка ОССН продолжится в штатном режиме (рис.1.18). В случае разблокированной учётной записи *root* суперпользователь получает доступ к CLI интерфейсу *recovery mode* (рис.1.19.)

Ввод команды journalctl с опциямиxb позволяет выполнить вывод данных из журнала системных событий системы инициализации systemd, специфичных для выбранного режима загрузки. Анализ этих данных даёт возможность суперпользователю проанализировать проблемы с этапами загрузки ОССН. С целью поэкранного просмотра списка сообщений рекомендуется использовать конвейер journalctl –xb |less.

Пример вывода данных из журнала системных событий системы инициализации systemd (об успешном запуске самой системы журналирования системных событий) представлен на рис. 1.20.

После окончания процесса загрузки в режимах Hardened или Generic пользователю предоставляется возможность работы с ОССН в графическом режиме (интерфейс GUI) или в режиме командной строки (интерфейс GLI).

В соответствии со сценарием инициализации системы Debian GNU/Linux, на базе которой разработан дистрибутив ОССН, пользователю по умолчанию предоставляется возможность работы:

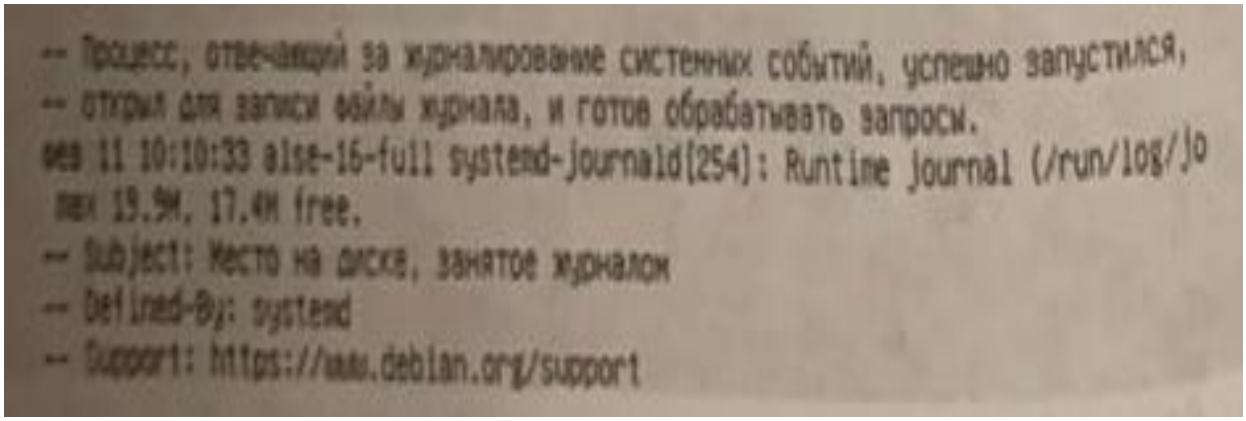
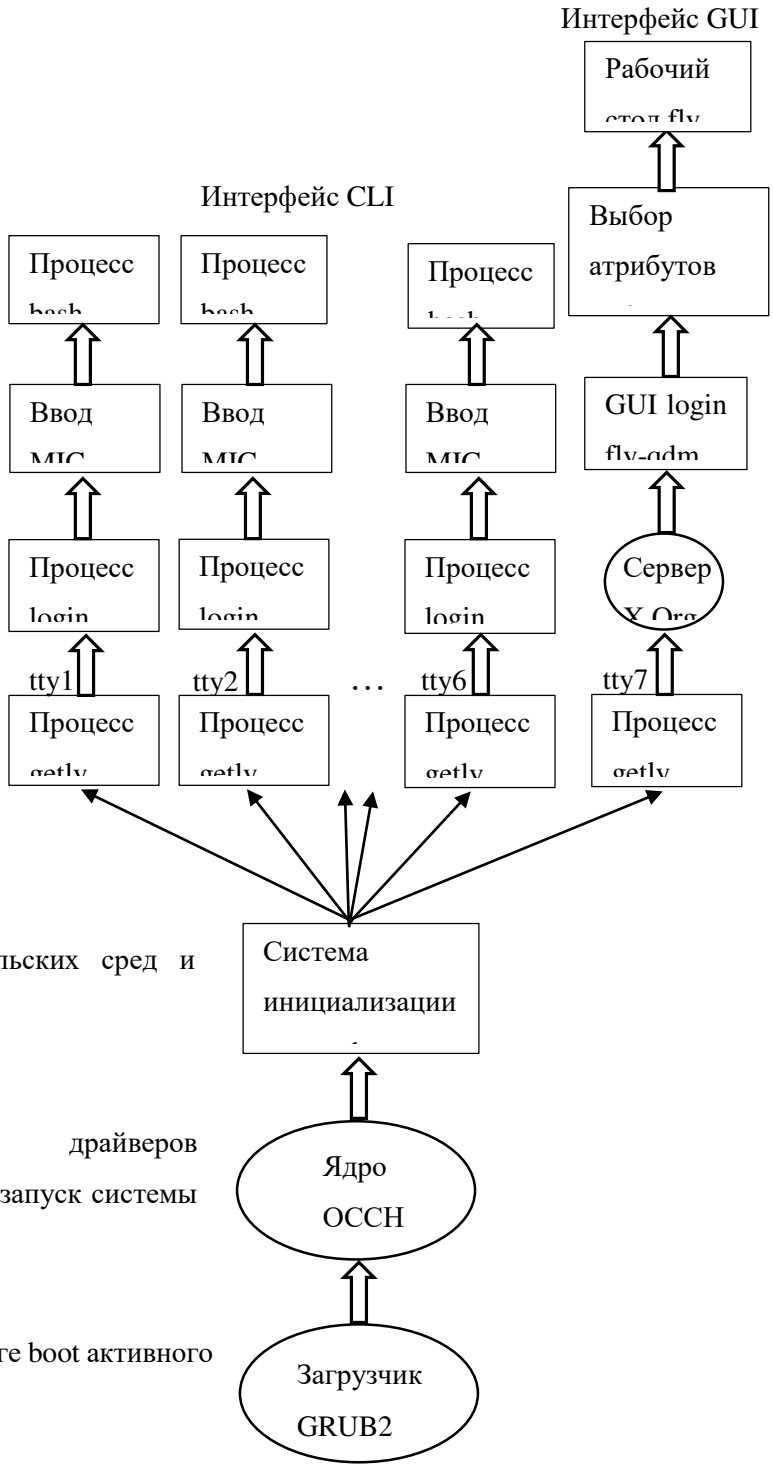


Рис. 1.20. Пример ввода данных из журнала системных событий системы инициализации system



Доступ к символьному экрану и

Доступ к графическому экрану, клавиатуре и мыши и/или сенсорному

уск пользовательских сред и анных сервисов

ициализация драйверов ройств, поиск и запуск системы

ск ядра в каталоге boot активного дела

Рис. 1.21. Последовательность загрузки *CLI-* и *GUI*-интерфейсов

- с интерфейсом *CLI* в шести терминалах — консоли ttyl-tt^6 (в качестве оболочки по умолчанию используется командный интерпретатор *basK)*
- с интерфейсом *GUI* в седьмом терминале — консоль *ttyl* (в качестве интерфейса *GUI* используется защищённый рабочий стол *Fly).*

В общем виде последовательность загрузки *CLI- u* GUI-интерфейсов представлена на рис. 1.21. Переключение между терминалами *tty1-tty7* осуществляется комбинацией клавиш *Ctrl*+ *Alt*+F_N, где N — номер консоли (устройства *tty),* в котором пользователь будет регистрировать свою сессию.

В консолях tty1-tty6 при входе пользователя в систему (процесс login) реализован диалог ввода значения мандатного уровня

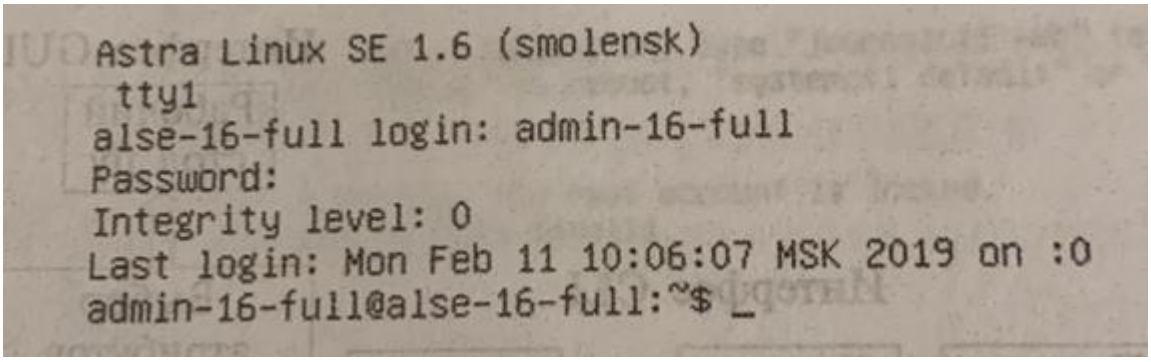


Рис. 1.22. CLI-интерфейс пользователя в консоли tty1

целостности. По умолчанию допускается ввод значений “Низкий” (числовое значение 0) или «Высокий» (числовое значение 63). Вид терминала на примере консоли *tty1* представлен на рис. 1.22.

В консоли *tty7* при входе пользователя также реализован диалог выбора атрибутов безопасности (в случае отсутствия настроек мандатного управления доступом в диалоге возможен только ввод уровня целостности):

- уровня доступа;
- неиерархических категорий;
- уровня целостности;

Для уровня этих атрибутов загружается GUI-интерфейс представленный начальным экраном графического входа в систему (процесс fly-qdm). Его управляющие элементы показаны на рис.1.23.

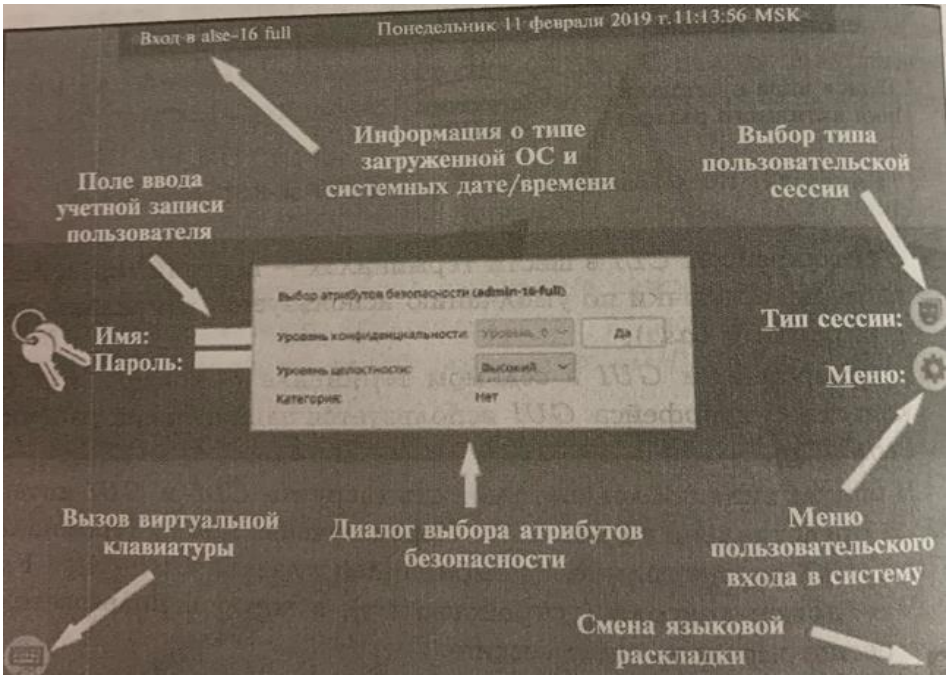


Рис. 1.23. Экран графического входа в систему

Рассмотрим элементы экрана входа в систему “Тип сессии” и “Меню” подробнее.

«**Тип сессии**» элемент управления обеспечивает пользователю выбор среды пользовательской сессии ОССН. При этом X предоставляется возможность переключиться между следующими режимами сессии:

- *безопасный* (графическая сессия с использованием графической версии терминала — утилиты *fly-term)*;
- *десктоп* (графическая сессия с использованием защищённой графической подсистемы *Fly)* — применяется по умолчанию;
- *мобильный* (используется GW-интерфейс рабочего стола *Fly* и набор приложений, адаптированные для работы с экранами мобильных устройств);
- *планшетный* (вариант сессии десктоп, адаптированный для работы с сенсорными экранами).

Безопасный режим предназначен для работы в ОССН с интерфейсом *CLI* (реализуемым после запуска графического сервера *X.Org* утилитой *fly-term)* с использованием консольных команд. Часто применяемые консольные команды в утилите *fly-term* заданы в виде выпадающего списка. Выбранная из него команда переносится в командную строку, после чего достаточно ввести её аргументы и (при необходимости) опции. Также этот список можно изменить с помощью соответствующего элемента меню «Настройка» утилиты *fly-term.* Выход из безопасной сессии выполняется с помощью элемента «Выход» меню «Файл» утилиты *fly-term* или командой *ent* в командной строке. При этом в случае, если открыты две или более вкладки виртуальных терминалов потребуется подтверждение завершения сессии.

Мобильный режим предназначен для использования ОССН на мобильных устройствах (планшеты, смартфоны), имеющих поддерживаемую ОССН аппаратную конфигурацию. Элементы управления рабочего стола *Fly* адаптированы под особенности подсистемы ввода-вывода этих устройств (сенсорный экран с многопальцевым управлением и использованием жестов). В мобильном режиме используется специально адаптированный под этот режим набор приложений (рис. 1.24). Не рекомендуется выбирать этот режим при эксплуатации ОССН на обычных компьютерах.

Планшетный режим реализует функциональность режима дектоп. Элементы GW-интерфейса в этом режиме адаптированы (Укрупнены) для пальцевого управления ими с использованием

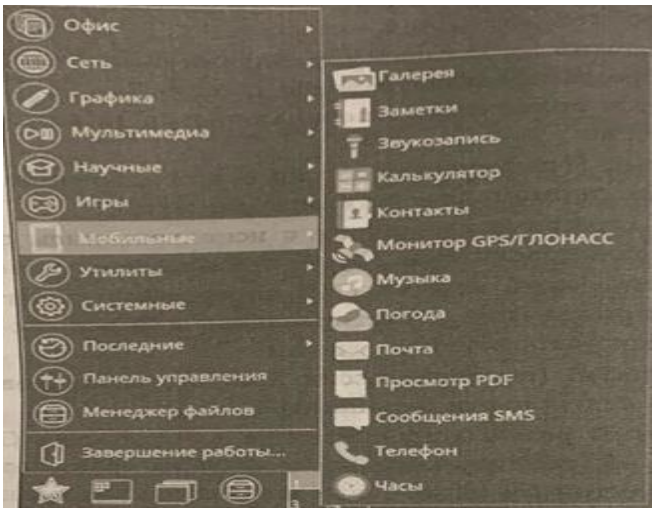


Рис. 1.24. Состав приложений, адаптированных для работы в мобильном режиме

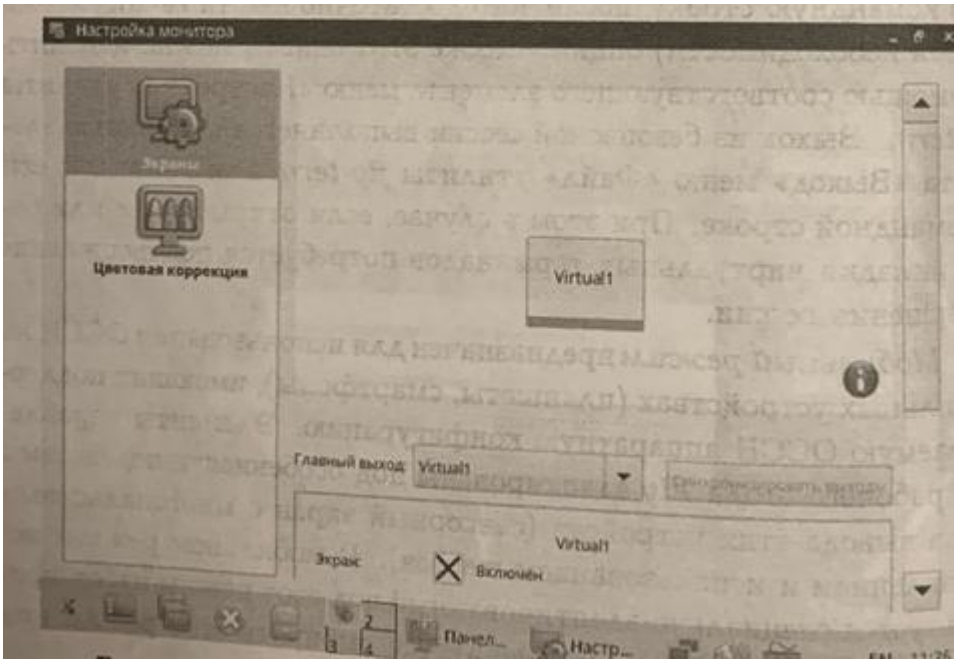


Рис. 1.25. Пользовательский интерфейс планшетного режима

«**Меню**»» — элемент управления, позволяющий реализовать следующие возможности:

- *сменить пользователя* (комбинация клавиш *Alt+I*) — отображает список открытых пользовательских сессий с указанием: пользовательских учётных записей, номеров виртуальных терминалов *(vt1-vt7),* на которых они выполняются и типов пользовательских сессий (например, TTY);
- *перезапуск X сервера* (комбинация клавиш *Alt+E)* — позволяет передать текущему сеансу *X.Org* сервера сигнал принудительного перезапуска;
- *виртуальная клавиатура* (комбинация клавиш *Alt + K)* — выводит на экран регистрации виртуальную клавиатуру. Применяется в случае невозможности использования аппаратной клавиатуры на традиционных рабочих станциях или же при установке ОССН на устройствах с сенсорным экраном;
- *консольный вход* (комбинация клавиш *Alt+N)* — позволяет работать с интерфейсом *CLI* в терминале консоли *tty1.* При выборе этого пункта меню пользователю предварительно отображается окно с сообщением о том, что переключение в консольный режим приведёт к невозможности работы в графическом режиме, который станет возможным снова через 10 секунд после окончания последнего успешного консольного входа или через 40 секунд, если ни один консольный вход не будет осуществлён;
- *выключение* (комбинация клавиш *Alt+S)* — выводит на экран окно завершения работы с ОССН. Используя это окно, пользователь может произвести выключение или перезагрузку ОССН без выполнения входа какого-либо пользователя, а также спланировать выключение/перезагрузку по истечении заданного интервала времени (режим планирования доступен только суперпользователю или пользователю с правами суперпользователя и требует ввода его пароля).

Рассмотренные настройки графического входа в систему заданы по умолчанию. Однако администратор ОССН имеет возможность изменить их с использованием графической утилиты *fly-admin- dm,* запуск которой можно осуществить двумя способами:

- с применением элемента «Вход в систему» меню «Настройки» главного меню защищённой графической подсистемы *Fly* (рис. 1.26);
- через ввод команды *fly-admin-dm* в командной строке утилиты *fly-term* (рис. 1.27).

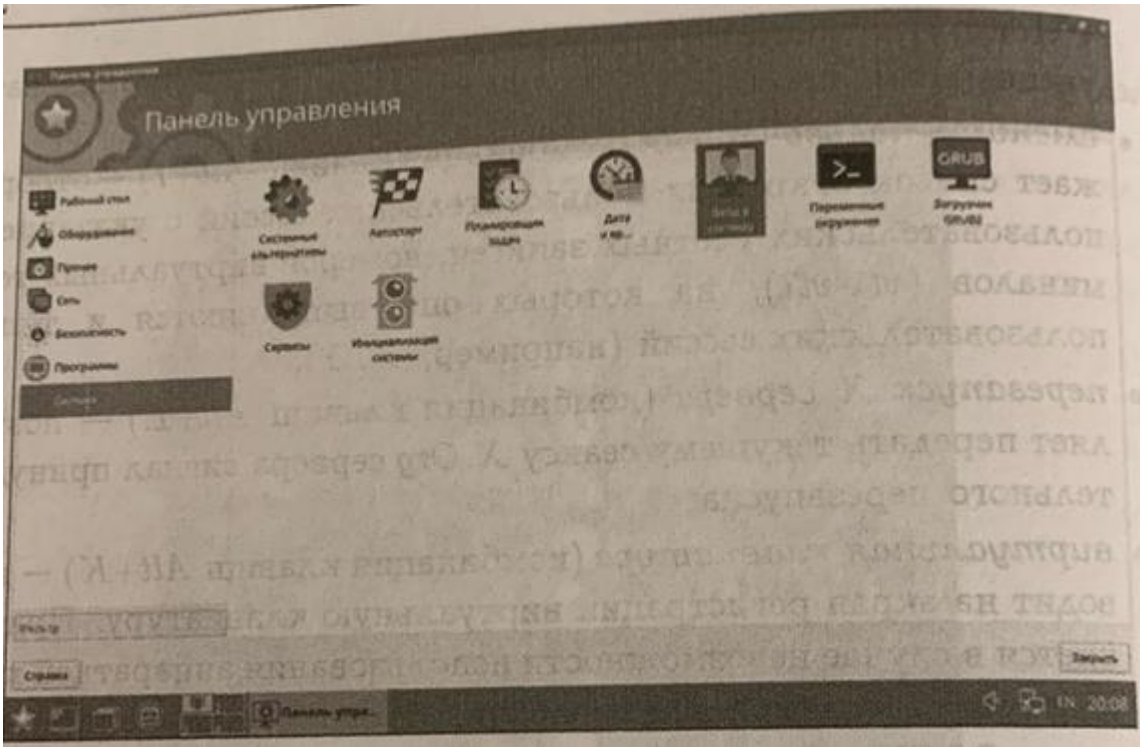


Рис. 1.26. Элемент “Вход в систему” меню “Настройки”

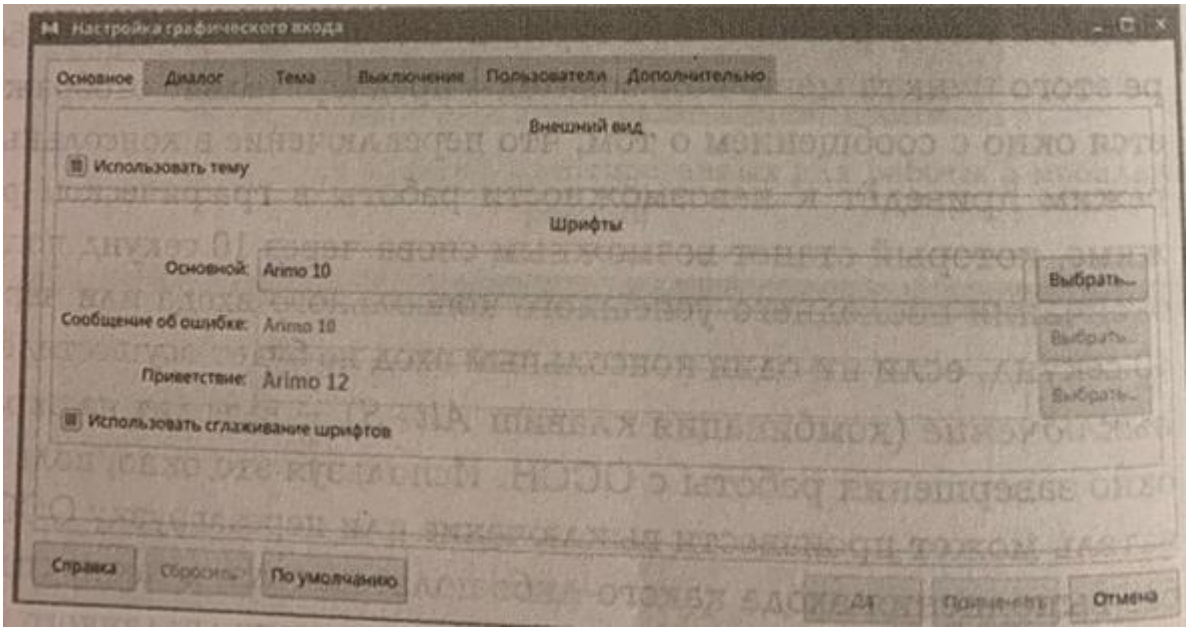


Рис. 1.27. Интерфейс графической утилиты *fly-admin-dm*

Первый три вкладки меню fly-admin-dm (“Основное” ,”Диалог” и ”Тема”) предназначены для оформления окна графического входа в систему с использованием соответствующих тем. Вкладки “Выключение”, “Пользователи” и “Дополнительно” предназначены для конфигурирования работы интерфейса графического входа в систему.

Для выхода пользователя из сеанса работы и/или завершения работы ОССН используется утилита “Выход и выключение” (графическая утилита fly-shutdown-dialog), вызываемая из “Системные” меню “Завершение работы” главного пользовательского меню , интерфейс которой приведен на рис. 1.28.

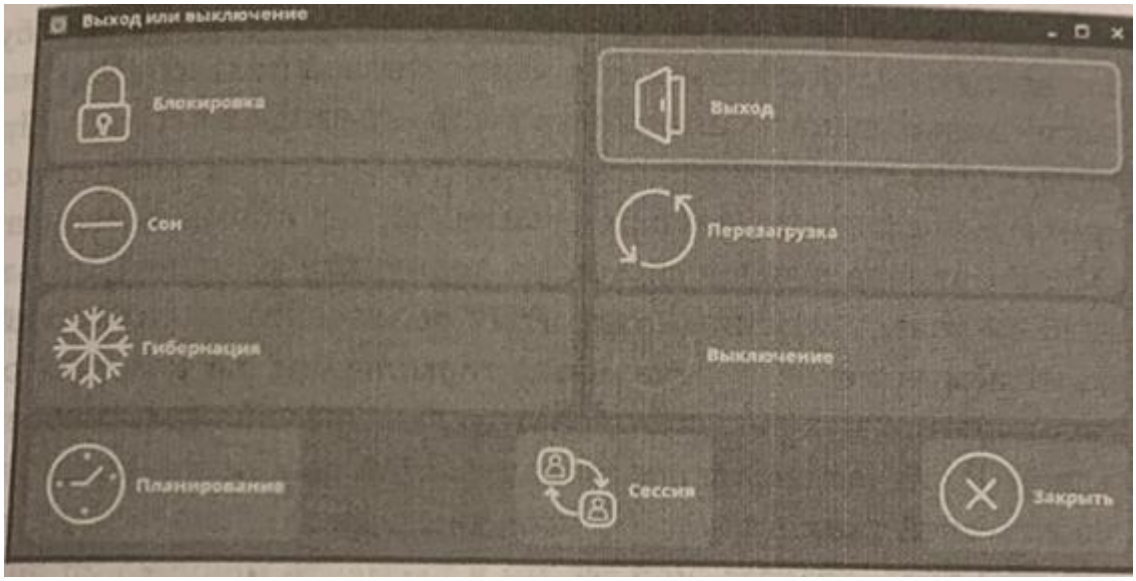


Рис. 1.28. Интерфейс графической утилиты выхода из сессии и/или завершения работы ОССН

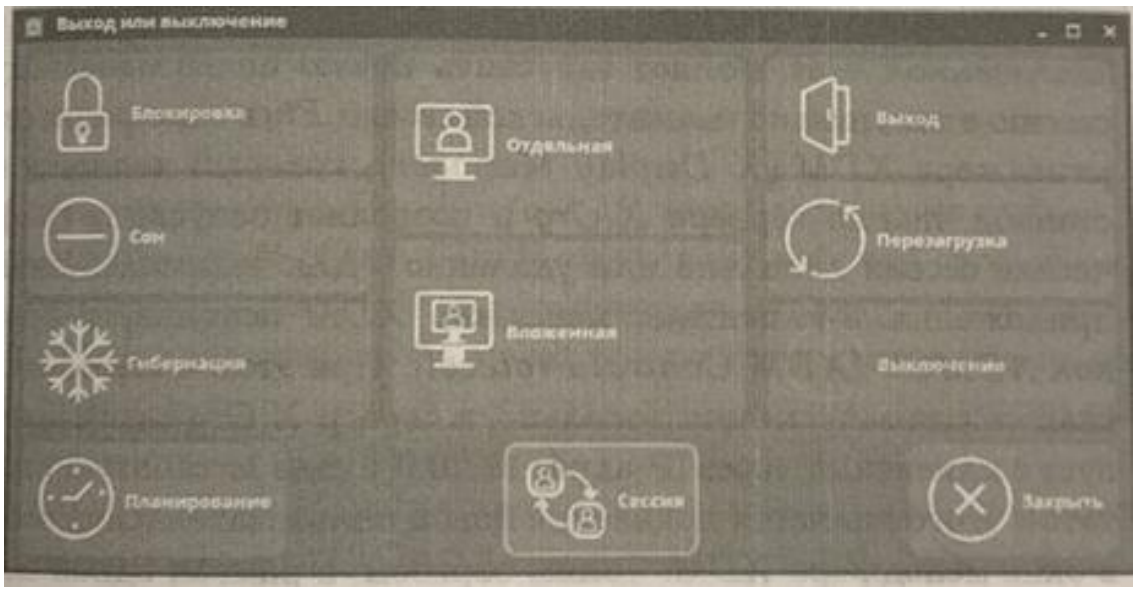


Рис. 1.29. Интерфейс диалога выбор типа сессии

Управляющие кнопки интерфейса сгруппированы по трём позициям (рис. 1.29):

- прерывание работы системы — кнопки «Блокировка», «Сон», «Гибернация»;
- выход из пользовательской сессии или выключение (перезагрузка) ОССН — кнопки «Выход», «Перезагрузка», «Выключение»;
- планирование выполнения перечисленных функций или смена типа сессии пользователя — кнопки «Планирование», «Сессия», «Закрыть».
- Интерфейс диалога смены типа сессии включает следующие
- От*дельная* — позволяет запустить пользовательскую сессию в новом виртуальном терминале. При этом сессия текущего пользователя останется открытой, и в дальнейшем к ней можно будет вернуться в любой момент. Запуск новой пользовательской сессии возвращает пользователя к экрану входа в систему. При

вводе данных учетной записи нового пользователя она будет открыта в новом виртуальном терминале. При этом в элементе “Меню” экрана входа в систему в подпункте “Сменить пользователя” появятся записи открытых пользовательских сессий с указанием номеров виртуальных терминалов, на которых они выполняются и типов пользовательских сессий (рассмотрены выше). В случае смены пользовательской сессии в защищенной графической подсистеме Fly при уже выполняющейся другой (других) пользовательской сессии в элементе меню “Тип новой графической сессии” появится дополнительный подпункт, указывающий, какой пользователь, на каком терминале и какого типа открыл эту сессию.

- *Вложенная* позволяет запустить новую пользовательскую, сессию в текущей пользовательской сессии *Fly в отельной окне* менеджера *XDM (X Duplay Manager),* который является составной частью сервера *X.Org* и позволяет запускать графические сессии локально или удалённо. Для взаимодействия с приложениями-клиентами менеджер *XDM* использует протокол *XDMCP (XDM Control Protocol).* При этом пользовательская сессия запускается локально, и сервер *X.Org* взаимодействует с клиентами через IP-адрес 127.0.0.1 узла *localhost. Резуль*татом этого является появление новой пользовательской сессии в окне менеджера *XDM.* Таким образом, в рамках одной пользовательской сессии *Fly* может функционировать другая пользовательская сессия *Fly.*

1.4.2.Основные приёмы работы с защищённой графической подсистемой *Fly*

В графической сессии по умолчанию используется защищённая графическая подсистема *Fly,* при функционировании которой применяются:

- сервер *X.Org* реализация сервера *X Window System* с открытым исходным кодом;
- рабочий стол *Fly,* который, в свою очередь, состоит из менеджера окон *Fly Window Manager* (утилита *fly-wm)* и набора графических утилит (*fly*-утил) для пользователей и администраторов с графическим интерфейсом *GUI.*

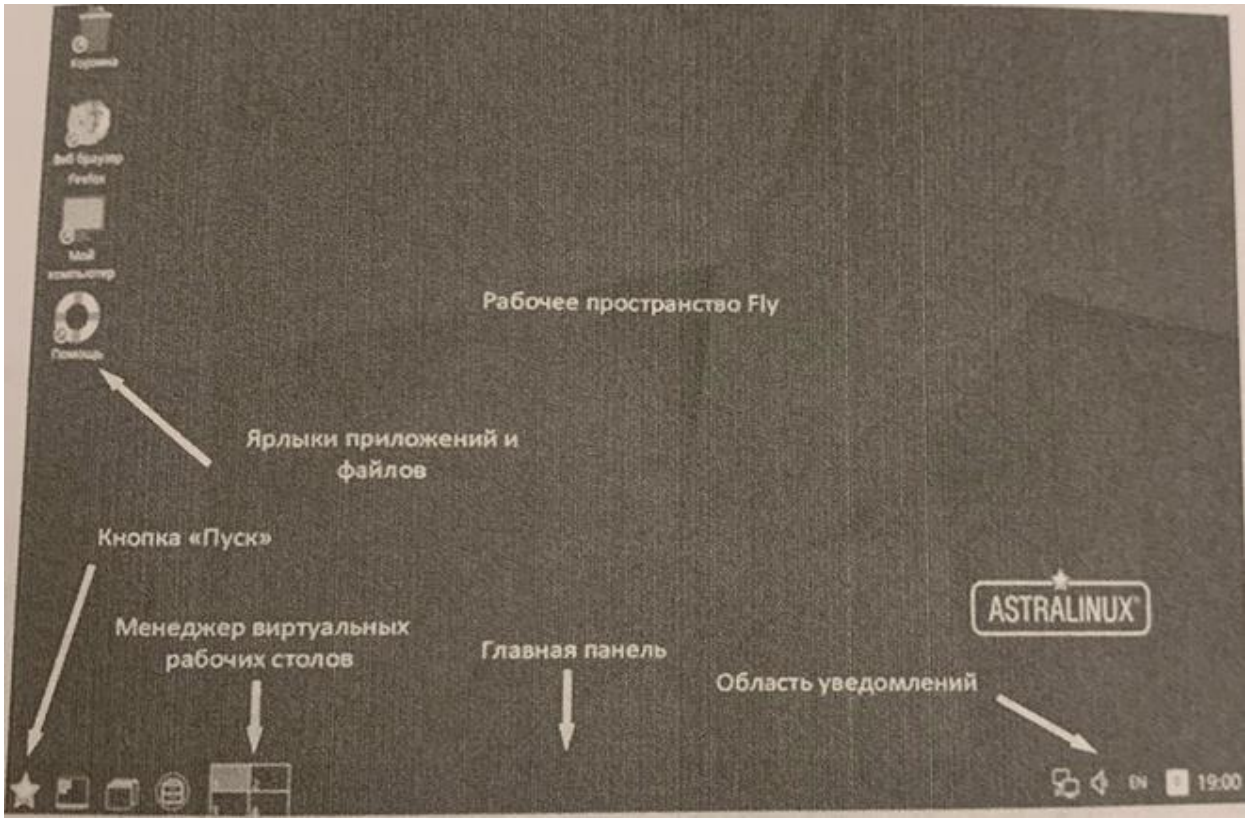


Рис. 1.30. Элементы интерфейса рабочего стола fly

Менеджер окон *Fly Window Manager* организует работу графической оконной среды ОССН, загружает рабочий стол *Fly* и его окружение (заданный набор графических утилит). Интегрированный в среду рабочего стола менеджер рабочих столов обеспечивает поддержку одновременной работы с несколькими рабочими столами (по умолчанию, с четырьмя).

Рабочий стол *Fly* загружается после регистрации пользователя в графической сессии. Он содержит пространство рабочего стола с фоновым изображением, панель задач и элементы интерфейса учетной записи пользователя (рис. 1.30).

Главное пользовательское меню (вызывается при нажатии на экране кнопки «Пуск») состоит из каскадно-выпадающих списков доступных пользователю приложений и функции (рис.1.30.) По своей функциональности оно максимально приближено к главному пользовательскому меню ОС семейства *Mzcrosof Windows.*

Каждый пользователь ОССН имеет возможность индивидуально настроить рабочий стол *Fly* (внешний вид, расположение элементов, особенности работы с клавиатурой и мышью). Часть таких настроек жестко определяется администратором и недоступна непривилегированному пользователю.

Доступные пользователю настройки могут быть заданы с использованием графической утилиты «Панель управления» (*fly-admin-center*) вызываемой последовательно из основного меню

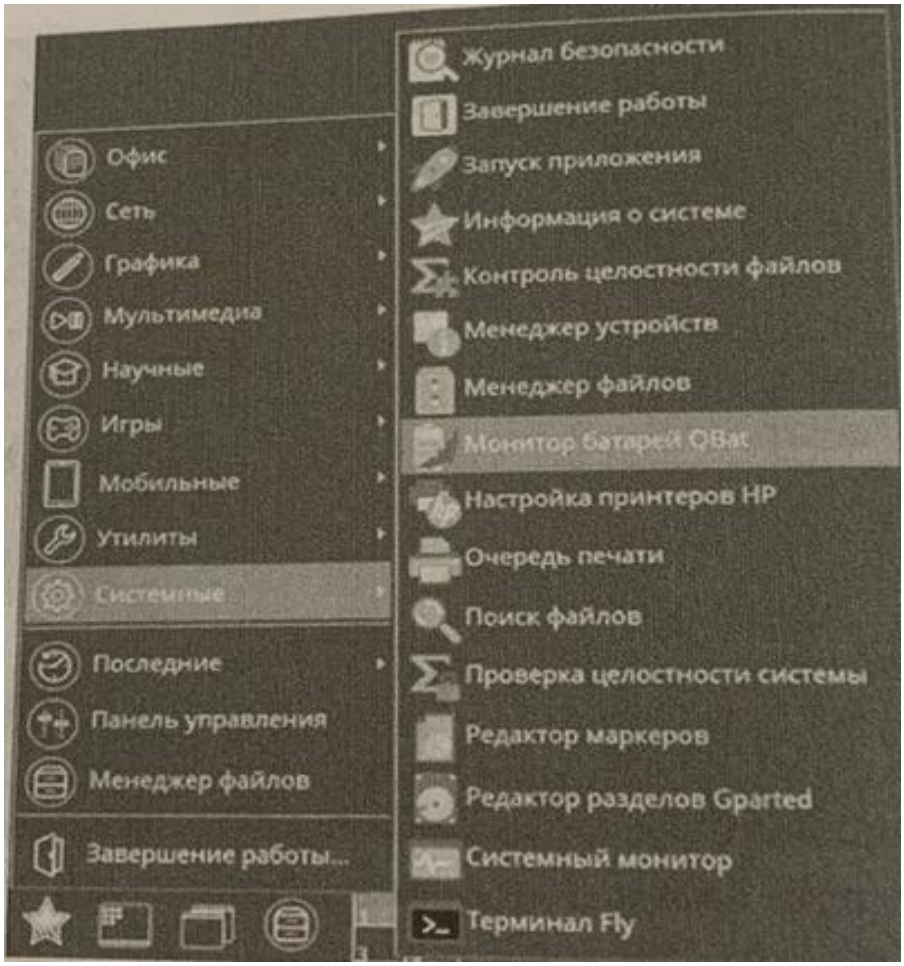


Рис. 1.31. Пример каскадно-выпадающего списка главного пользовательского меню (кнопка ”Пуск”)

«Пуск» — «Панель управления» — «Рабочий стол». Эта утилита позволяет централизовано использовать некоторые административные и пользовательские приложения рабочего стола *Fly,* которые для удобства разделены на несколько категорий. Например, категория «Рабочий стол» объединяет графические утилиты, большинство из которых могут быть применены пользователем для настройки своего индивидуального рабочего стола.

Важной особенностью защищённой графической подсистемы *Fly* является интегрированная поддержка механизмов защиты ОССН. По умолчанию пользователю доступны следующие ее функции:

- изменение пароля своей учётной записи с помощью консольной команды *passwd* или графической утилиты *«Политика безопасности»;*
- изменение владельца или группы, владеющей объектом файловой системы (сущности в рамках МРОСЛ ДП-модели), созданным пользователем, с помощью консольных команд *chown* и *chgrp*, файловых менеджеров *Midniqht Commander* и *fly-fm*;
- изменение прав доступа в рамках модели *minimal ACL* (дискреционное управление доступом) к сущности файловой системы

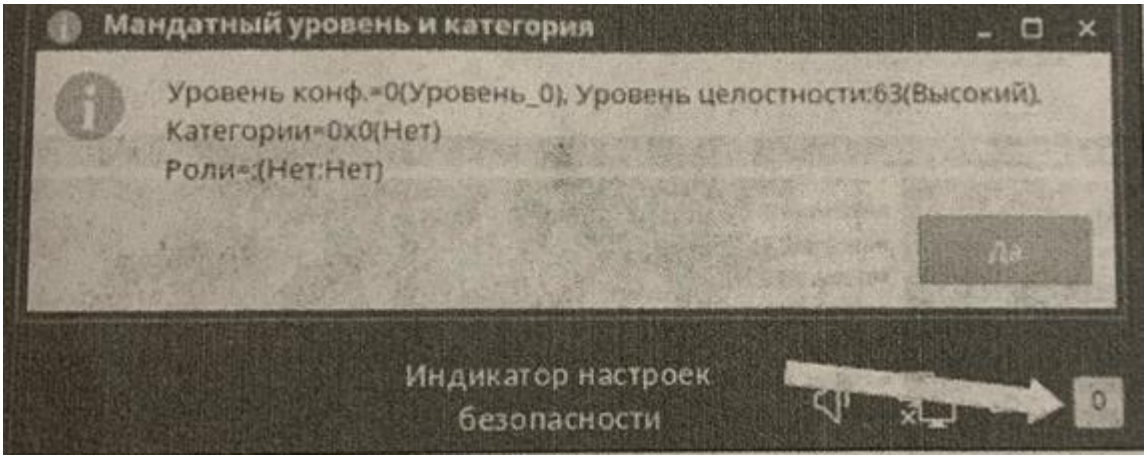


Рис. 1.32. Вид индикатора настроек безопасности пользовательской сессии

- созданной пользователем, с помощью консольной команды *chmod,* файловых менеджеров *Midnight Commander* и *fiy-fm;*
- установка уровней доступа (включая неиерархические категории) и целостности при создании новой пользовательской сессии (субъект-сессии в рамках МРОСЛ ДП-модели);
- получение для текущей пользовательской сессии информации о её уровнях доступа и целостности с помощью консольной команды *pdp-id* или визуально в области уведомлений на панели задач.

Таким образом, для учётной записи пользователя, для которой администратором ОССН установлены несколько допустимых мандатных уровней доступа и целостности, на экране графического входа в систему (см. рис. 1.23) отобразится меню установки их конкретных значений. В дальнейшем выбранные уровни отображаются в виде индикатора в области уведомлений на панели задач (рис. 1.32).

Для большей наглядности при обработке информации конкретным оконным приложением с *GUI*-интерфейсом значение его уровня доступа дублируется цветовым кодированием:

- уровень 0 — голубой;
- уровень 1 — желтый;
- уровень 2 — оранжевый;
- уровень 3 — темно-розовый;
- уровень 4 — красный;
- уровень 5 — коричневый;
- уровень б — пурпурный;
- уровень 7 — темно-фиолетовый.

Цветовое кодирование применяется как для индикации значений уровня доступа в области уведомлений, так и к рамке, обрамляющей окно приложения. На рис. 1.33 и 1.34

показан пример цветового кодирования окна файлового менеджера *fly-fm* для пользовательской сессии со значением уровня доступа равным 1 (желтое цветовое кодирование).

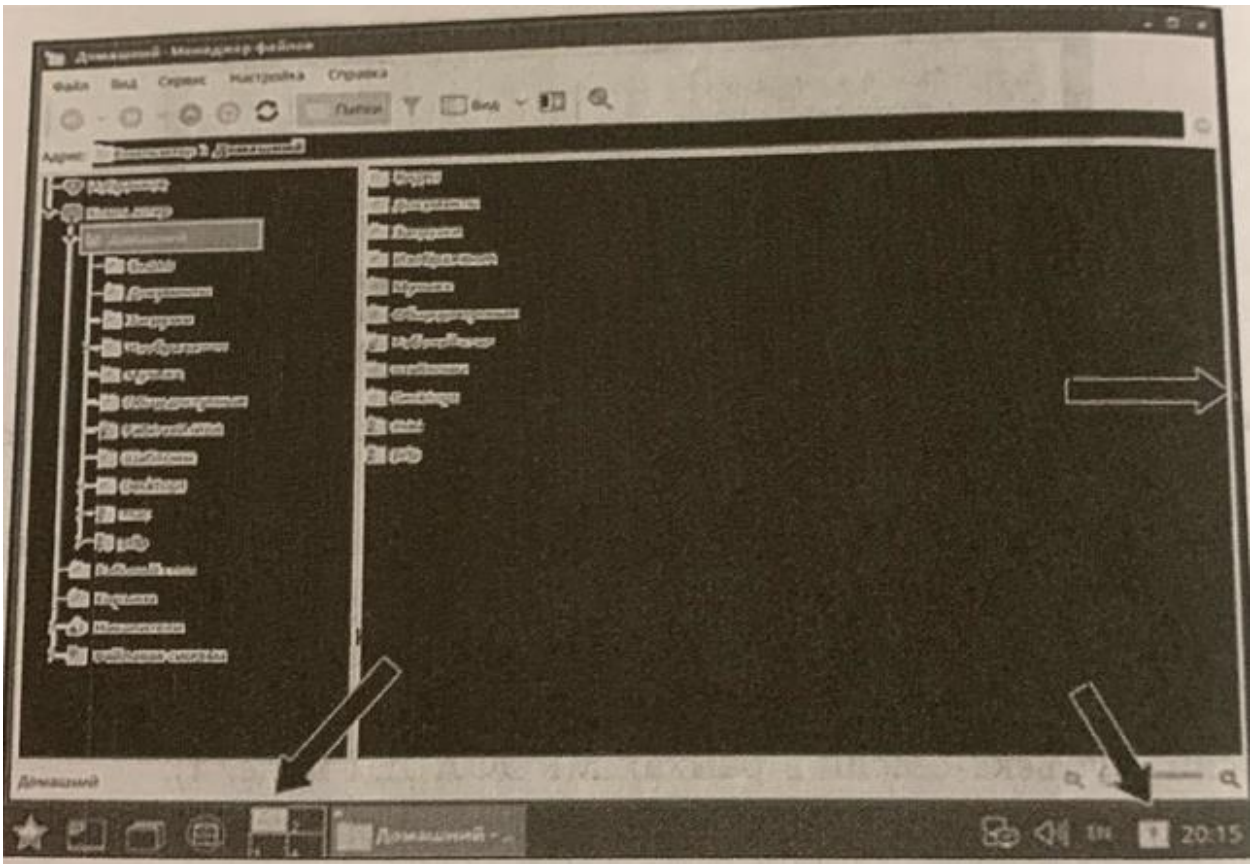


Рис. 1.33. Пример цветового кодирования значения уровня доступа субъект-сессии, соответствующей оконному приложению

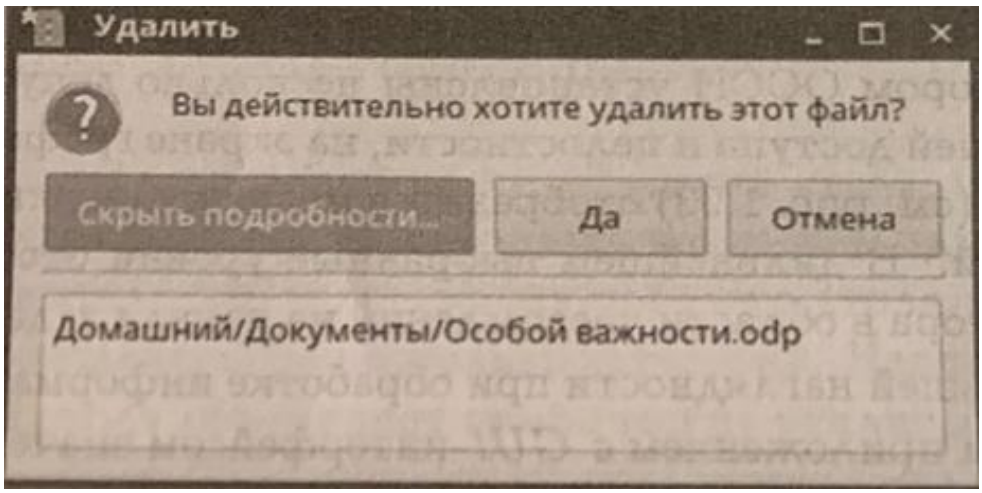


Рис. 1.34. Пример цветового кодирования значения мандатного уровня модального окна файловой операции

Как правило, в реальных ОССН этому значению соответствует уровень доступа “Для служебного пользования”.

Детальное описание теоретических основ реализуемого в ОССН на основе МРОСЛ ДП-модели мандатных управления доступов и контроля целостности будет приведено в главе 2, а порядок их стройки и администрирования рассмотрен в главе 3.

1.4.3. Основные задачи администрирования ОССН

В общем случае процесс администрирования ОС на базе проекта *Debian GNU/Linux* (в том числе ОССН) включает решение следующих основных задач:

- администрирование учётных записей пользователей и групп с целью организации многопользовательской работы ОССН и реализации управлении доступом процессов (субъект-сессий, функционирующих от имени учётных записей пользователя) к файлам, каталогам и другим объектам доступа ОССН (сущностям в рамках МРОСЛ ДП-модели);
- администрирование процессов с целью оптимизации распределения между ними ресурсов аппаратного обеспечения;
- администрирование запоминающих и периферийных устройств с целью управления доступным их пространством и динамически подключаемыми устройствами.

В случае включения устройства с ОССН в сетевую среду и его функционирования в составе доменной инфраструктуры, перечень задач дополняется администрированием сети и доменов, рассмотренным в п. 1.3 и более подробно в главе 3.

Администрирование учётных записей пользователей и групп. Отправной точкой реализации управления доступом в ОССН являлось унаследованное от ОС проекта *GNU/ Linux* дискреционное управление доступом *(DAC — Discretionary Access Control).* Поэтому, несмотря на то что в настоящее время в ОССН используются мандатные управление доступом и контроль целостности, а в перспективе ролевое управление доступом, целесообразно рассмотреть основные элементы дискреционного управления доступом, не утратившие своей актуальности в современных релизах и версиях ОССН.

Дискреционное управление доступом в ОС проекта *GNU/Linux* базируется на понятии владения (использовании права доступа владения) файлом, каталогом, процессом (сущностями и субъект-сессиями). Так, в файловых системах семейства *EXTFS,* в частности которая по умолчанию используется в ОССН, с каждым файлом или каталогом связана учётная запись пользователя — их владельца *(owner).* Процесс, функционирующий от имени такой учетной записи-владельца сущности, имеет право изменять дискреционные права доступа к ней, например назначать их учетным других пользователей ОССН на основе стандарта *POSIX ACL [127].*

Для оптимизации и облегчения администрирования дискреционного управления доступом в случаях, когда к одним и тем же файлам или каталогам требуется установить одинаковые права доступа более чем для одной учётной записи пользователя, в ОССН

применяются группы учётных записей пользователей. В результате для файлов и каталогов владельцем (обладателем к ним правом доступа владения) может быть задана группа. При этом для них остаются владельцами и соответствующие учетные записи пользователей. В перспективе при реализации в ОССН ролевого управления доступом вместо учетных записей пользователей и групп владельцами будут задаваться роли или административные роли.

Таким образом, при управлении доступом в ОССН, в том числе дискреционным, администрируют следующие его элементы:

- учётные записи пользователей *(account);*
- группы (логические объединения учётных записей пользователей с равными дискреционными правами доступа);
- права доступа к файлам, каталогам, другим объектам доступа (сущностям и субъект-сессиям);
- режимы доступа, обеспечивающие возможность учёта ряда особенностей управления доступом.

Утилиты администрирования учётных записей пользователей и групп реализованы в пакете *Shadow Suite* [137]. Кроме них для этого используется ряд системных файлов, которые конфигурируются администратором системы. Такие файлы в рамках МРОСЛ ДП-модели рассматриваются как сущности, параметрически ассоциированные с учётными записями пользователей.

Основой для задания учётных записей пользователей и групп являются их идентификаторы:

- *uid (User ID)* — число, уникально идентифицирующее учётную запись пользователя в ОССН;
- *gid (Group ID)* — число, уникально идентифицирующее группу пользователей ОССН.

Эти идентификаторы хранятся в следующих конфигурационных файлах, которые используются специальным процессом реализующим вход пользователя в ОССН — его идентификацию и аутентификацию:

- */etc/passwd* —файл с данными об учётных записях пользователей;
- */etc/group* — файл с данными об учётных записях групп пользователей.

Учётная запись каждого пользователя представляет одну строку в файле /etc/ *passwd.* Таким образом, регистрация новой учётной записи пользователя в системе, фактически, является добавлением соответствующей строки в этот файл (рис. 1.35). Разделителем полей данных в строке учётной записи является символ «:». В результате у администратора есть возможность фильтрации требуемых дынных об учетной записи пользователя (например, с помощью утилит *grep* или *cut*).

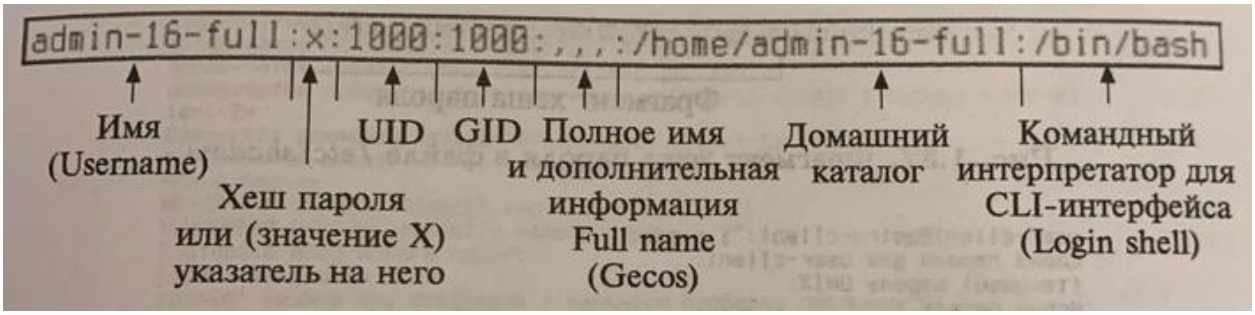


Рис. 1.35. Пример строки, соответствующей учетной записи пользователя admin-16-full в файле /etc/passwd

В состав строки, соответствующей учётной записи пользователя, входят поля:

- *username* — уникальное символьное имя, присваиваемое каждой учётной записи пользователя (при выборе имени могут использоваться буквы и цифры, а также знак подчёркивания и точка);
- *uid* — уникальный идентификатор учётной записи пользователя (представлен числом);
- *gid* — уникальный идентификатор первичной группы, в которую входит учётная запись пользователя (представлен числом);
- *хеш пароля* — может храниться либо непосредственно в рассматриваемой строке (рис. 1.36), либо, если вместо пароля задан символ <x», то он хранится в системном файле */etc/shadow,* доступном для чтения и записи только суперпользователю или пользователю с правами суперпользователя (рис. 1.37);
- *full name* — полное имя учётной записи пользователя (например, пользователь с *username* равным *stdor* может иметь полное имя *«Ivan Sidorov»,* а также в этом же элементе через запятую могут перечисляться дополнительные сведения о пользователе: домашний и рабочий номера телефонов, адрес и т. д.);
- *home directory* — домашний каталог учётной записи пользователя, находящийся в каталоге */home*\
- *login shell* — командный интерпретатор *[shell),* который запускается при входе пользователя в ОССН, по умолчанию это интерпретатор *bash (/btn/bash).*

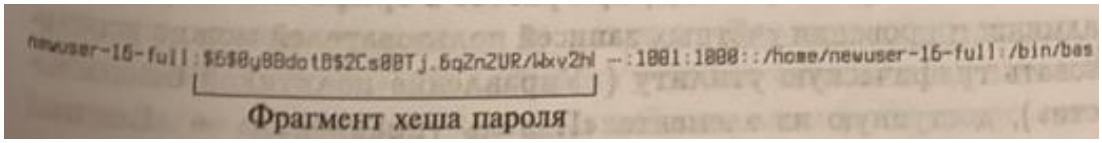


Рис. 1.36. Фрагмент хеша пароля в файле */etc/passwd*

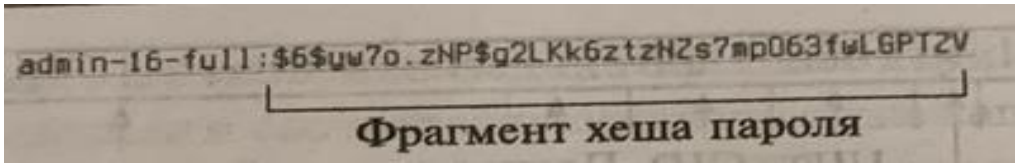


Рис. 1.37. Фрагмент хеша пароля в файле */etc/shadow*

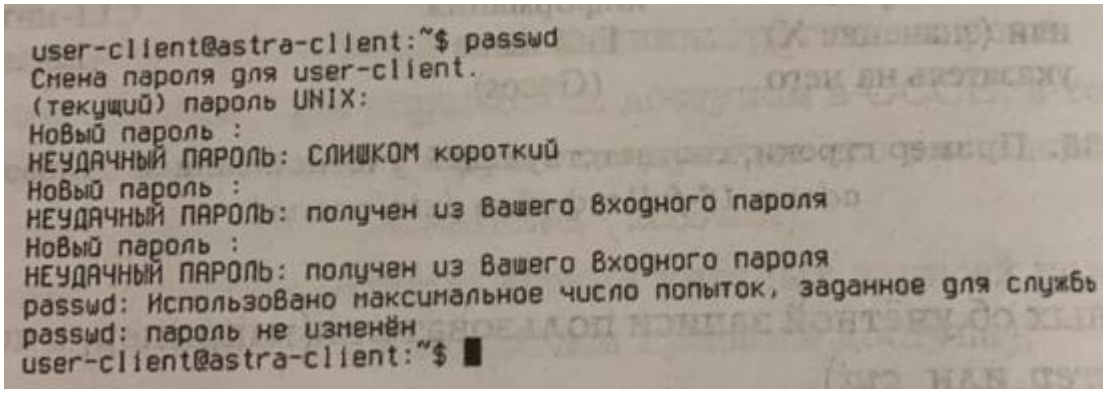


Рис. 1.38. Диалог команды *passwd* при смене пароля пользователя

Создание, удаление и изменение учетных записей пользователей осуществляется администратором (в рамках МРОСЛ ДП-модели ему соответствует доверенная субъект-сессия, обладающая соответствующими текущими специальными административными ролями) с использованием следующих приемов:

- непосредственное редактирование файла */etc/passwd*;
- применение утилит из пакетов *passwd* и *adduser*;
- активизация из меню “Пользователь” графической утилиты *fly-admin-smc* (“Управление политикой безопасности”).

В ОССН получение идентификатора учётной записи пользователя *(uid),* а также других связанных с ним наборов параметров, возможно с помощью команд *whoami, id, who.* Для изменения пароля учётной записи пользователя используется команда *passwd,* которая проверяет заданные правила формирования пароля, например его минимальную длину или же то, что он создан на основе предыдущего пароля (рис. 1.38).

Для корректного администрирования учётной записи пользователя (в том числе с добавлением при её создании в домашний каталог необходимых конфигурационных файлов) администратор может использовать команды *adduser, useradd* (создание), *usermod* (модификация параметров) и *userdel* (удаление). В рамках МРОСЛ ДП-модели этим командам соответствуют де-юре правила вида *create_user, set_user_labels u delete_user*. Пример использования команды *adduser* приведен на рис. 1.39.

Как было указано выше, при работе в графической среде, для администрирования учетных записей пользователей можно использовать графическую утилиту (“Управление политикой безопасности”), доступную из элемента “Панель управления” – “Безопасность” главного пользовательского меню (рис. 1.40).

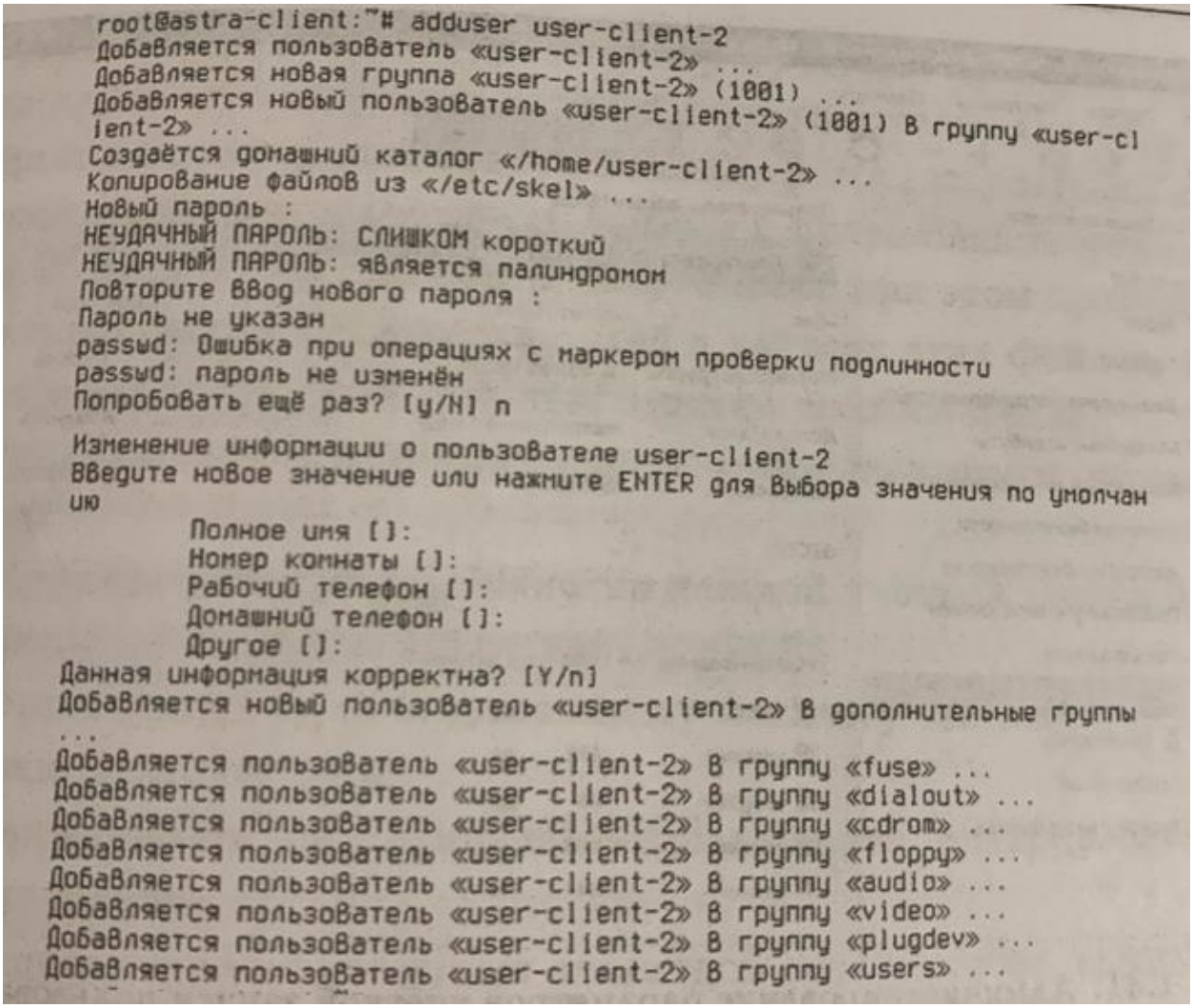


Рис. 1.39. Пример использования команды *adduser*

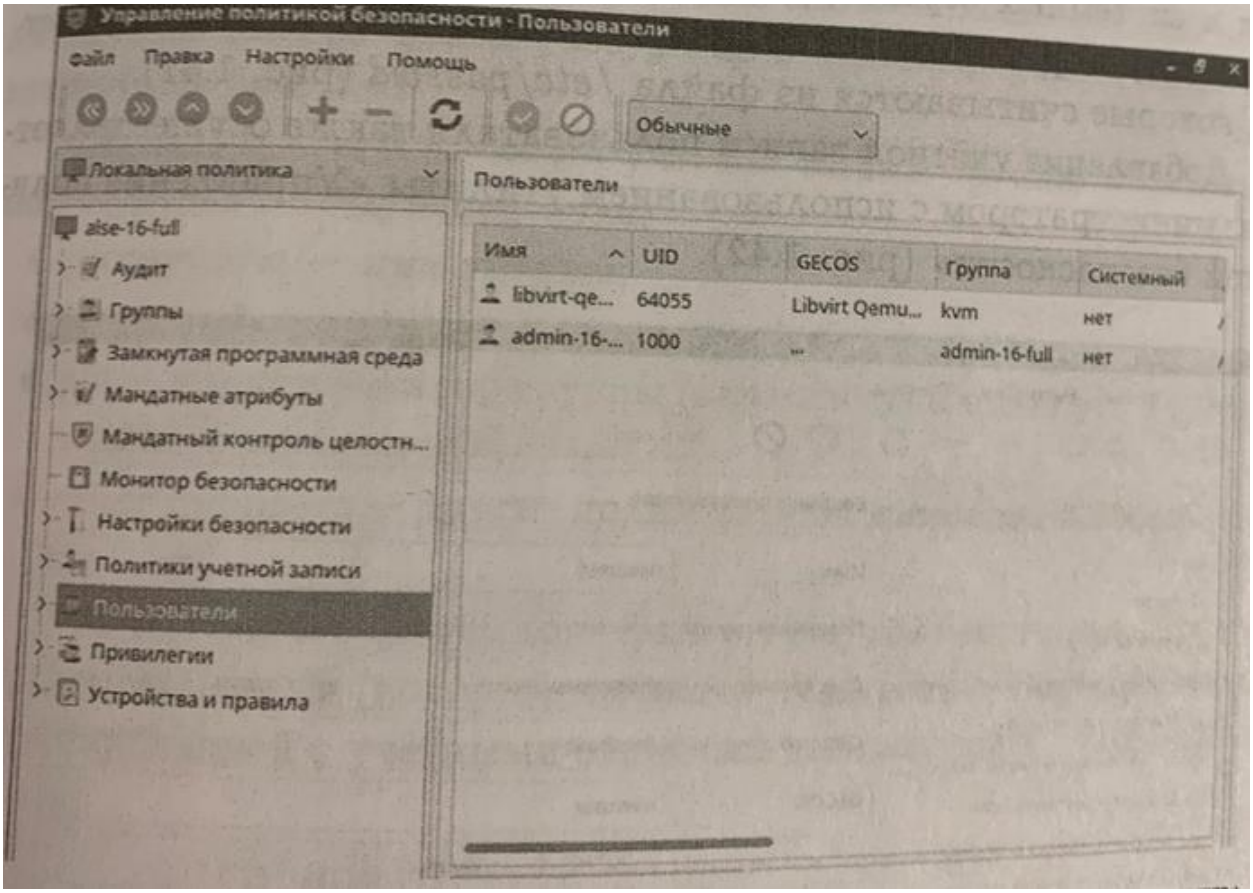


Рис. 1.40. Главное окно утилиты “Управление политикой безопасности”

Учётные записи пользователей, зарегистрированных в ОССН локально, администрируются из раздела «Пользователи» вкладки «Локальная политика». При этом все учётные записи пользователей разделены на два класса: *Обычные и Системные* (в этот класс входят встроенные учётные записи, а также учётные записи прикладных и системных сервисов).

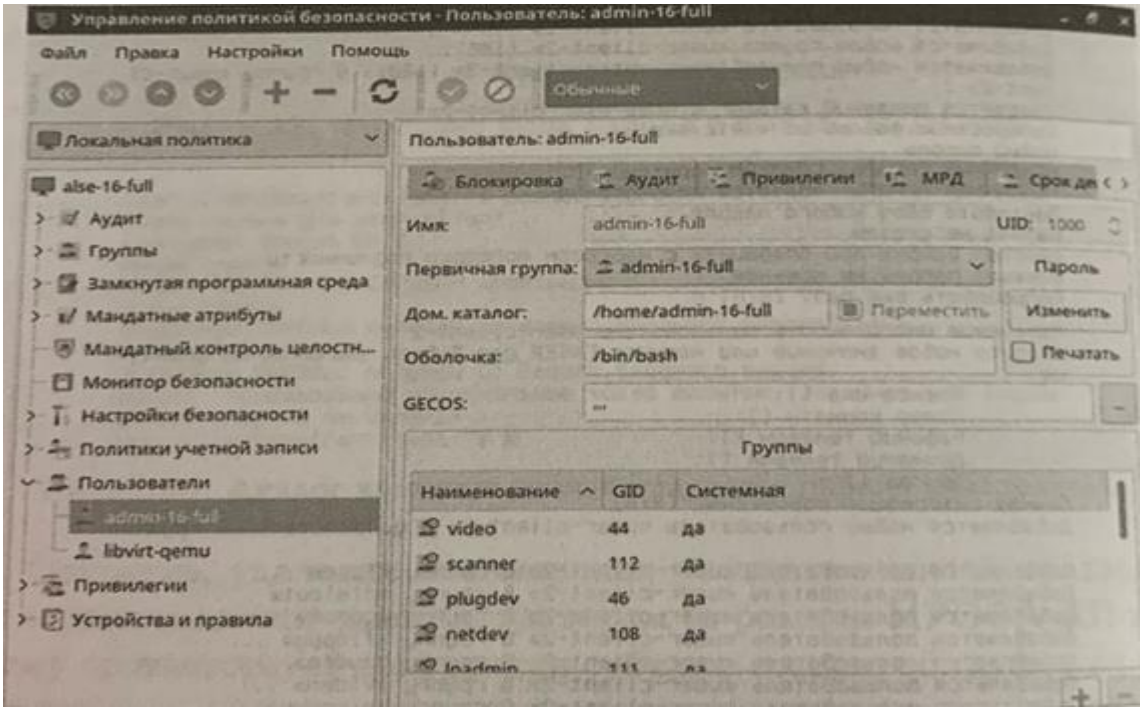


Рис. 1.41. Администрирование параметров учетной записи пользователя

Выбрав какую-либо учетную запись администратор ОССН может посмотреть и изменить ее параметры, которые считываются из файла /etc/passwd (рис. 1.41).

Добавление учетной записи пользователя так же осуществляется администратором с использованием утилиты “Управление политикой безопасности” (рис. 1.42).

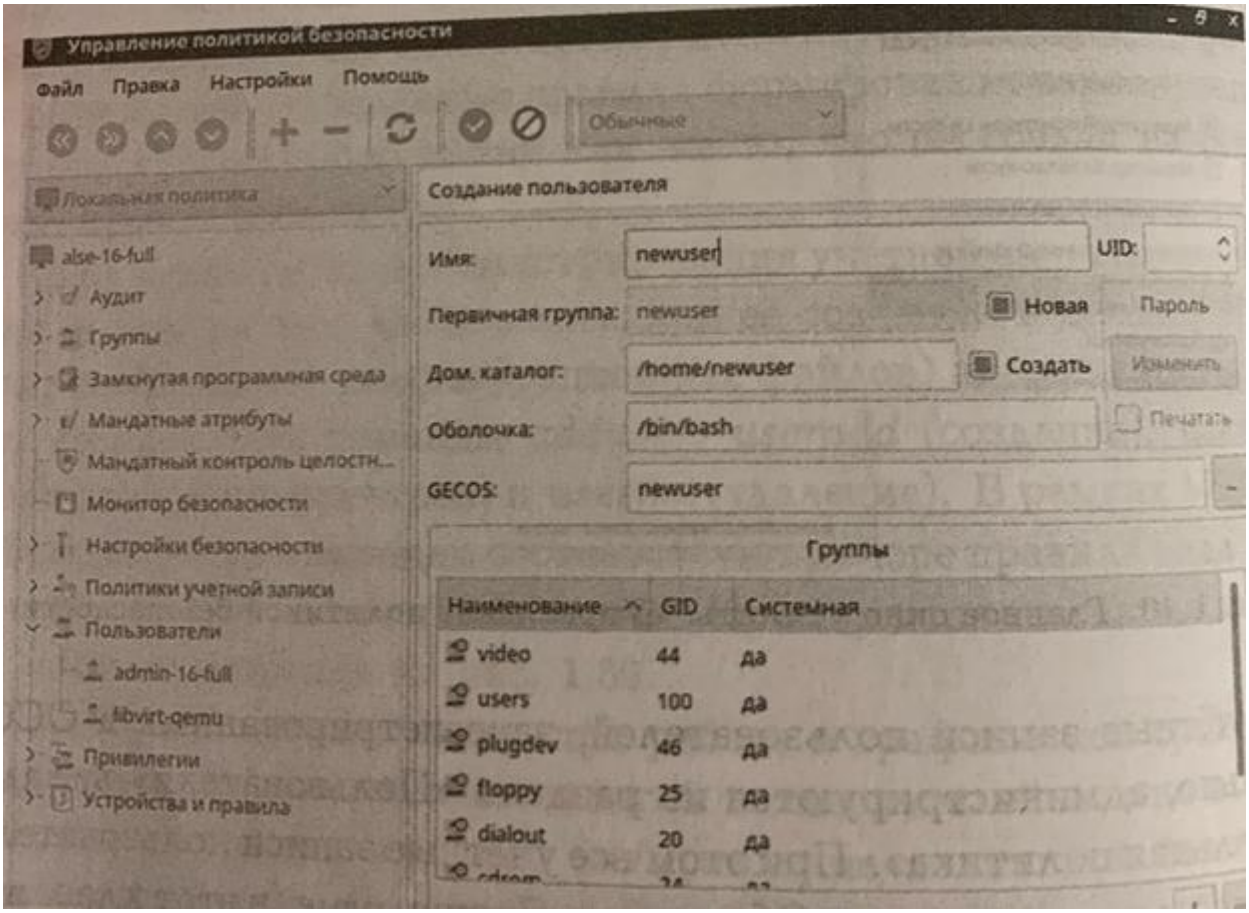


Рис. 1.42. Интерфейс меню “Создание пользователя”

Для безопасности ОССН важна регулярная проверка корректности параметров учетных записей пользователей, для чего администратор может использовать команду *pwck* (от *password check*), которая проверяет целостность данных и правильность формата записи в файлах */etc/passwd* и */etc/shadow*, при этом проверяются:

- корректность количества полей в записях файлов;
- уникальность имен учетных записей пользователей;
- уникальность идентификаторов учетных записей пользователей;
- корректность принадлежности каждой учетной записи пользователя ее первичной группе
- корректность пути к домашнему каталогу каждой учетной записи пользователя
- корректность сведений об используемом интерпретаторе по умолчанию.

Исправление выявленных командой *pwck* ошибок администратор может осуществить с помощью команды *usermod*.

Данные о каждой группе учетных записей пользователей находится в системном файле */etc/group* в строке формата, представленного на рис. 1.43.

Строка, соответствующая учетной записи группы пользователей, содержит следующие поля:

- *groupname*- имя группы;
- *password*- пароль, установленный для группы;
- *gid*- индетефикатор группы(аналогичный полю gid в учетной записи пользователя);
- *other members*- другие пользователи, входящие в состов группы;

Как и в случае системного файла */etc/passwd*, право на запись в файл */etc/group* (который является сущностью, параметрически ассоциированной с учетными записями доверенных и недоверенных пользователей)

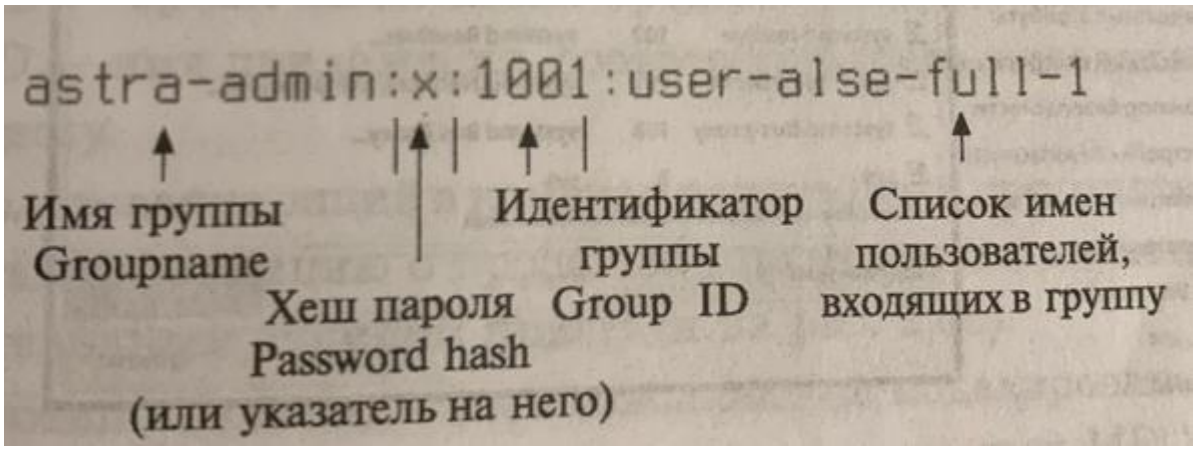


Рис. 1.43. Пример строки, соответствующей учетной записи группы *admin-16-full* в файле */etc/group*

имеет только администратор, остальные пользователи могут только просматривать его содержимое.

Пользователь может получить данные о своей принадлежности к группам с помощью команд *id* и *groups*. Администратор может создавать, удалять или изменять параметры групп учётный записей пользователей:

- непосредственно редактируя файл */etc/group*;
- редактируя файлы */etc/group* и */etc/gshadow* (назначение последнего аналогично назначению файла */etc/shadow*) командой *gpasswd;*
- используя команды *groupadd* и *groupdel*;
- изменяя параметры групп учетных записей пользователей командой groupmod;
- использую графическую утилиту “Управление политикой безопасности” (рис. 1.44.).

Для проверки корректности параметров групп учетных записей пользователей используется аналогичная команде *pwck* команда *grpck*, которая анализирует данные файлов */etc/group* и */etc/passwd*. При этом проверяются:

- корректность количества полей в записях этих файлов;

- уникальность имен групп;

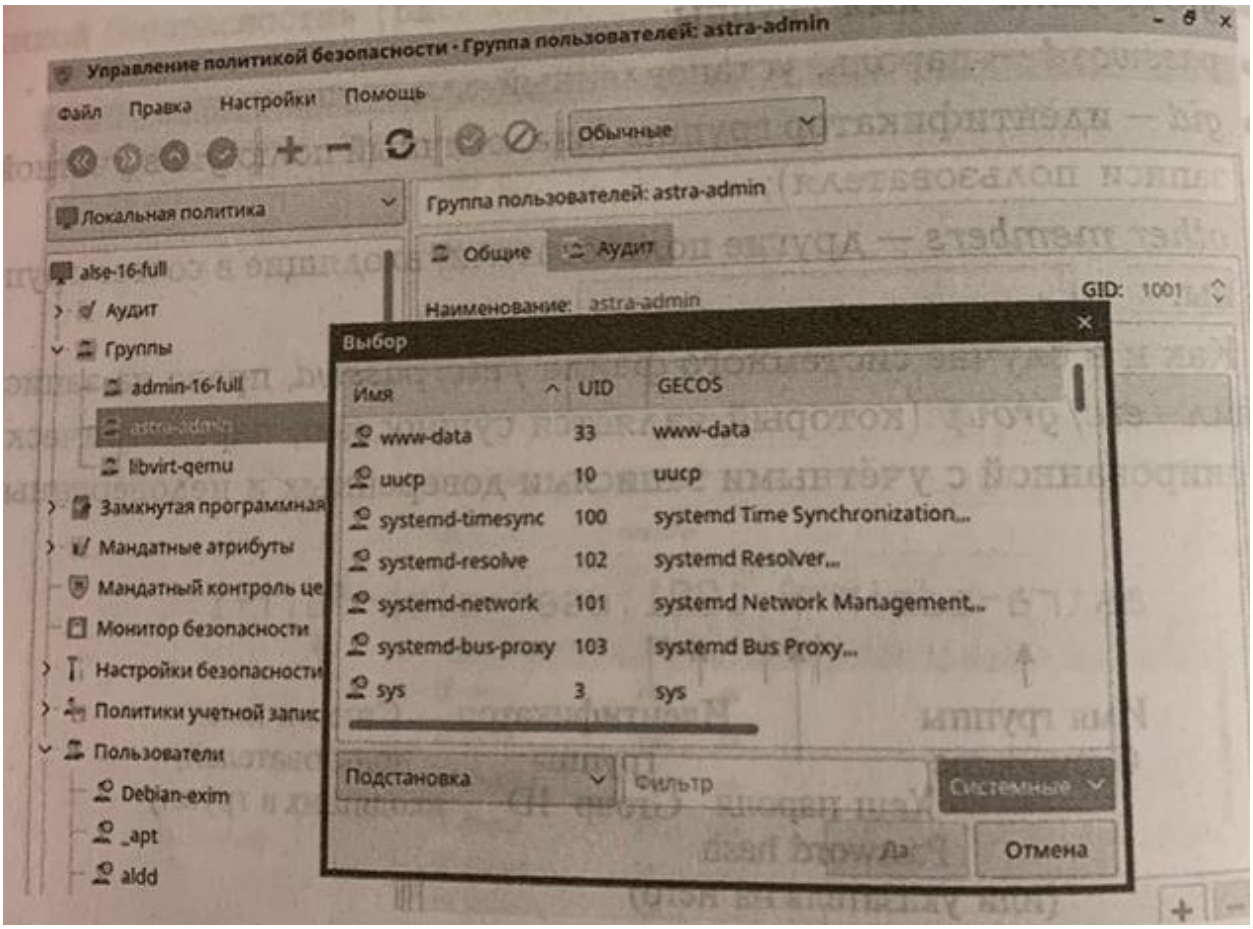


Рис. 1.44. Администрирование групп учетных записей пользователей с использованием утилиты “Управление политикой безопасности”

- уникальность идентификаторов групп;
- корректность состава учётных записей пользователей и администраторов каждой группы.

Исправление выявленных командой *grpck* ошибок администратор может осуществить с помощью команды *groupmod.*

Так как в перспективе на смену группам в ОССН придут роли, запрещающие роли или административные роли, то в рамках МРОСЛ ДП-модели командам администрирования групп учётных записей пользователей соответствуют де-юре правила вида: *grant_admin_rights, remove_admin_rights, add_negative_role, remove_negative_role, create-role, delete.role create _hard_link_role delete_hard_linк_role, rename-role, get_role_attr set_role_labels.*

Администрирование процессов. Администратору ОССН требуется контролировать и управлять выполнением совокупности процессов (доверенных или недоверенных субъект-сессий в рамках МРОСЛ ДП-модели), которые были запущены пользователями в рамках своих сеансов, а также ядром ОССН, системными и прикладными сервисами.

Основной командой для мониторинга состояния процессов в ОССН является команда *ps.* Запущенная без опций и аргументов, она выводит в терминал список процессов, выполняемых на активном терминале *(tty)* или псевдотерминале *(pts)* (рис. 1.45). При этом для каждого процесса указываются:

- *PID* — идентификатор процесса, который присваивается при его старте из числа свободных *PID* в диапазоне значений от О до 2^{32} 1;
- *TTY* — терминал *(tty)* или псевдотерминал (pts), в котором выполняется процесс;
- *TIME* — время выполнения процесса процессором;
- *CMD* — имя приложения, соответствующего выполняющемуся процессу.

*И*спользование опций в команде *ps* позволяет просмотреть более Детальную информацию о процессах (пример вывода команды *ps* с дополнительными опциями приведён на рис. 1.46).

Прототипом команды ps, позволяющим моделировать получение данных о процессах (субъект-сессиях), в рамках МРОСЛ ДП- модели является де-юре правило вида *get_subject_attr*.

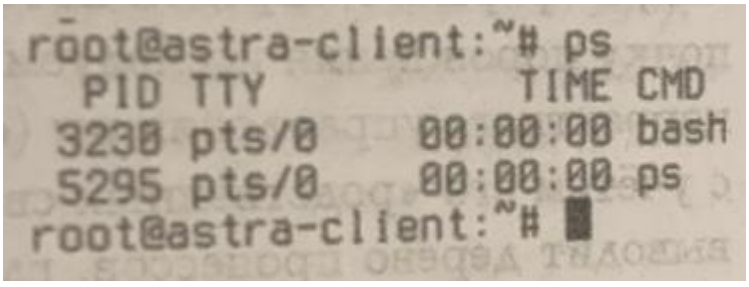


Рис. 1.45. Пример ввода команды *ps* без параметров

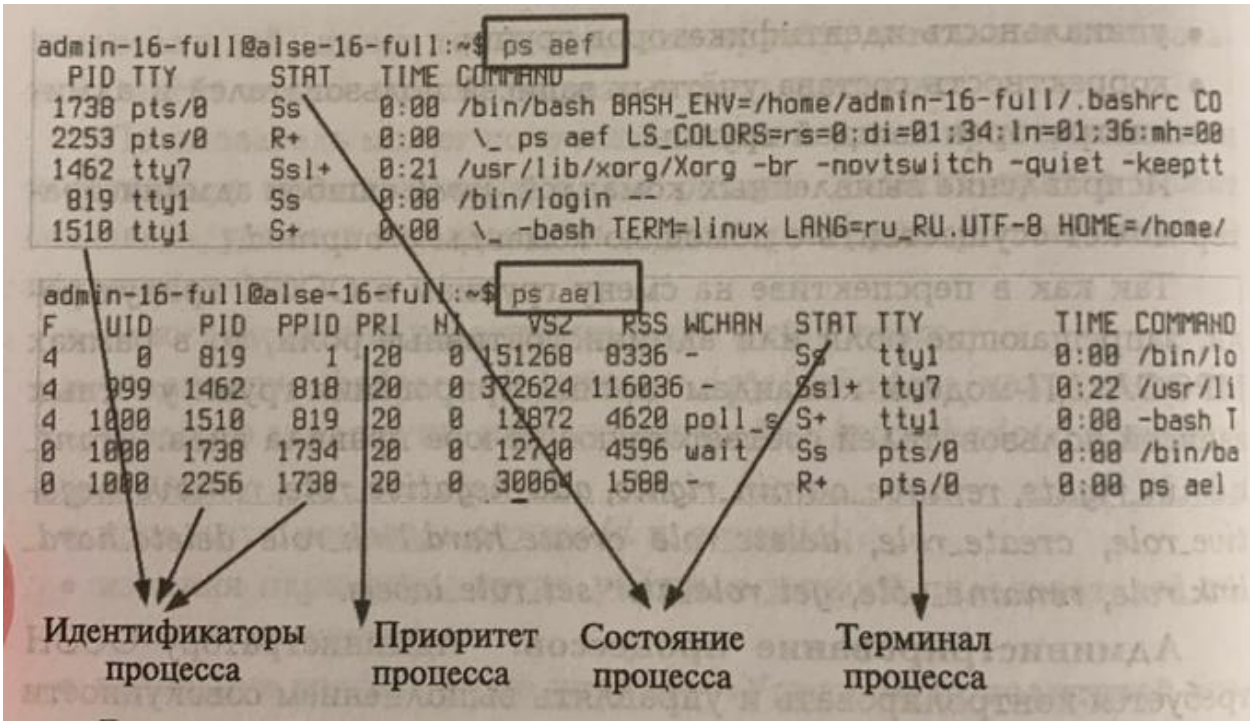


Рис. 1.46. Пример вывода команды *ps* с дополнительными опциями

Команда *pstree* дополняет команду *ps,* с ее помощью можно отобразить дерево процессов, функционирующих на всех терминалах, *в* формате “предок-потомок”. Это позволяет проследить цепочку порождения исследуемого процесса и получить возможность корректного управления им (особенно это касается его завершения) с учётом его «родственных связей». Без параметров команда *pstree* выводит дерево процессов, где каждый процесс идентифицируется именем соответствующего ему приложения (рис. 1.47).

Команды *ps* и *pstree* позволяют получить сведения о запущенных процессах, отследить их текущее, состояние и взаимосвязи друг с другом. Для того чтобы получать данные о процессах, собранные за некоторый интервал времени, используется команда *top.* Принцип её работы основан на периодическом анализе обновлений таблицы процессов ОССН, что позволяет отследить изменение их параметров (рис. 1.48).

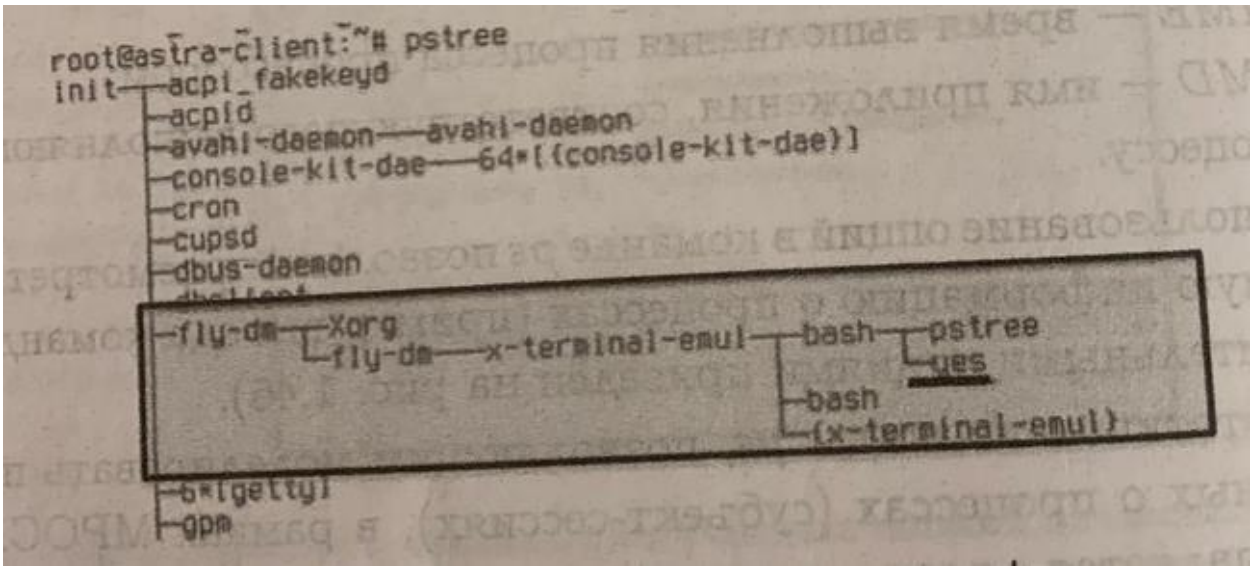


Рис. 1.47. Пример ввода команды *pstree*

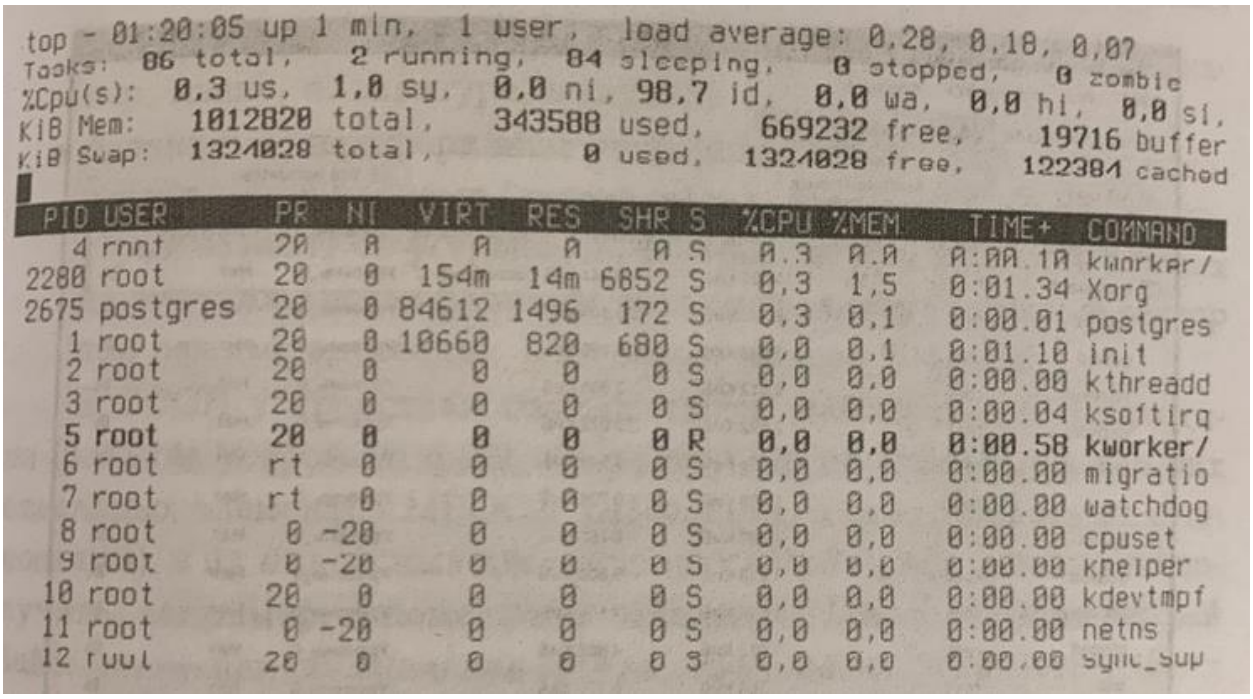


Рис. 1.48. Пример ввода команды *top*

Информация, выводимая командой top, содержит общие данные о процессах и ресурсах ОССН:

- число пользовательских сеансов (user);
- усреднённая загруженность ресурсов системы *(load average)-,*
- общее число процессов *(total task)* и число процессов, находящихся в разных состояниях *(running, sleeping, stopped* и *zom- bte);* ' ;ь г- ... ;
- загруженность процессора *(%Cpu);*
- распределение оперативной памяти между процессами *(KiB Mem);*
- используемые области подкачки страниц *(KiB Swap).*

Кроме того, команда *top* выводит в терминал все данные, выводимые командой *ps* в режиме детализированного вывода.

При работе в защищённой графической подсистеме *Fly* информацию, аналогичную выводимой командой *top,* можно получить с использованием графической утилиты «Системный монитор» меню «Системные» главного меню (рис. 1.49).

Дополнительно графическая утилита «Системный монитор» может осуществлять управление процессами, при этом доступны следующие функции:

- остановки/возобновления работы процессов;
- прерывания (для допускающих прерывания процессов);
- завершения процессов.

В ОССН администратор может управлять запущенными процессами, передавая им сигналы с использованием команды *kill* (в Рамках МРОСЛ ДП-модели ей соответствует де-юре правило вида *delete_session*).

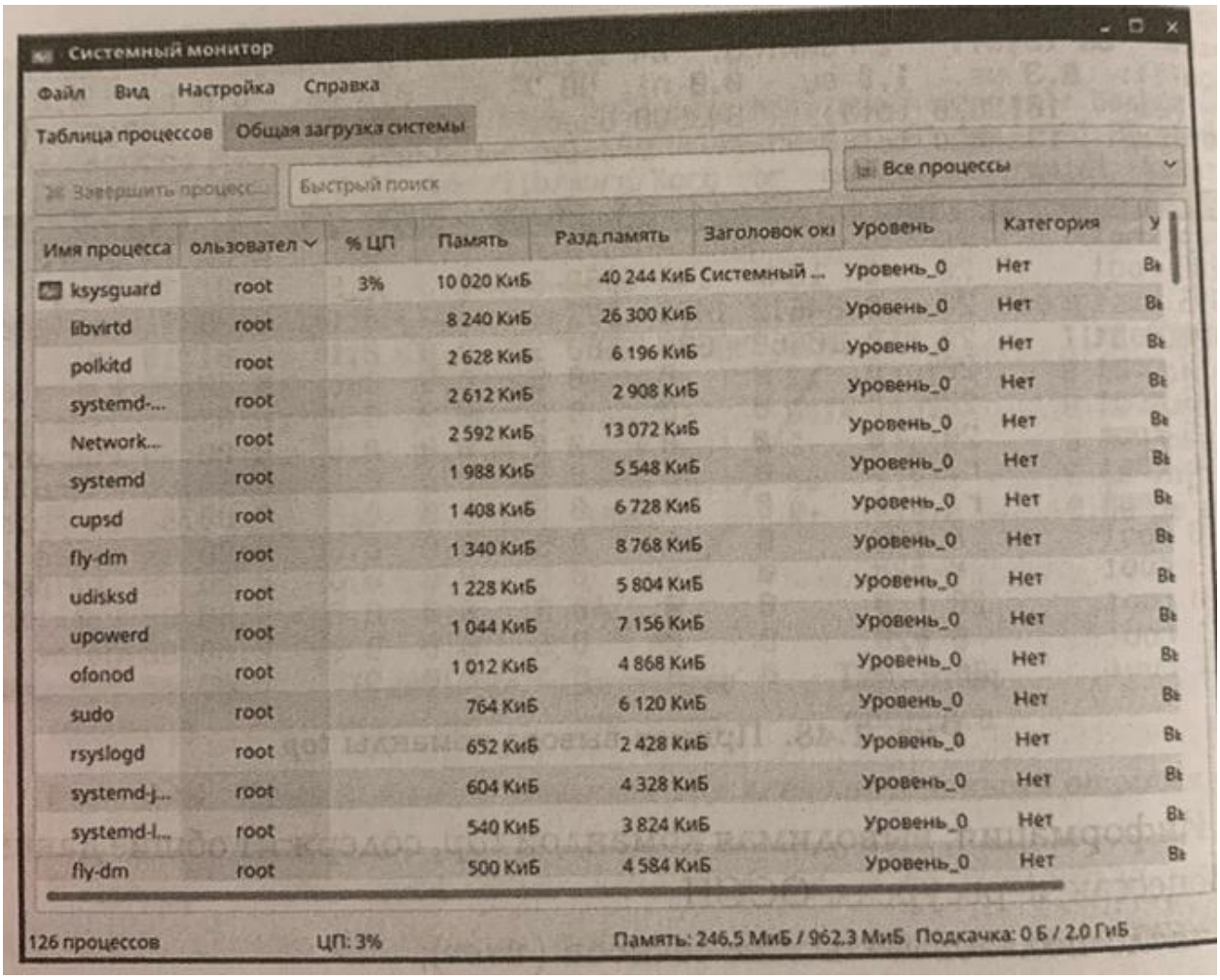


Рис. 1.49. Пример вывода графической утилиты “Системный монитор”

Процесс, получив сигнал, может отреагировать на него либо по умолчанию, заданному ядром ОССН, либо использовать собственную функцию обработки сигнала. В основном сигналы команды *kill* предназначены для реализации различных режимов прерывания, завершения, приостановления или возобновления работы процессов. Команда *killall* является расширением команды *kill u* отличается от неё тем, что посылает сигналы всем процессам запущенным при выполнении одного приложения.

Также администратор и пользователи ОССН имеют возможность влиять (например, с помощью команды nice) на использование выполняющимися процессами ресурса времени центрального процессора путем изменения их приоритета, указывающего ядру ОССН, где расположить процесс в общей очереди готовых к выполнению на центральном процессоре процессов.

Администрирование устройств (на примере запоминающих устройств).Для упрощения администрирования устройств, несмотря на большое разнообразия

соответствующего им периферийного оборудования и запоминающих устройств, в ОССН заданы всего два их типа:

- символьные- любые периферийные и запоминающие устройства, обмен данными с которыми ведется посимвольно (побайтно). К таким устройствам относятся, например, принтер, сканер, мышь, клавиатура, монитор;
- блочные – периферийные устройства, обмен данными с которыми ведется блоками (последовательностями байт), размер которых зависит от устройства. Например, при хранении данных на жёстких дисках блоками являются сектора (каждый сектор при режиме адресации *LBA* имеет размер 2048 байт).

В ОССН устройствам соответствуют файлы специального типа (задание устройств файлами позволило не определять для них цельных элементов МРОСЛ ДП-модели, а представлять их сущностями), в их индексных дескрипторах (inode), которые можно получить, например, с помощью команды *lsl,* используется первый байт:”c” — как обозначение файла символьного устройства, “b” — как обозначение файла блочного устройства.

Кроме того, в *inode* файлов устройств размещается два параметра, используемых ядром ОССН для задания драйверов устройства:

- *major number* — указывает подсистеме ввода-вывода ядра ОССН на драйвер класса устройств (например, всех жёстких дисков или всех сетевых карт);
- *minor number* — указывает подсистеме ввода-вывода ядра ОССН на драйвер конкретного устройства.

На рис. 1.50 показан пример вывода команды *lsl* для файлов устройств */dev/sda* (файл, соответствующий интерфейсу контроллера *SCSI)* и */dev/ttyl* (файл, соответствующий первому виртуальному терминалу). При этом первый файл является блочным устройством (через интерфейс *SCSI* данные передаются блоками — секторами), а второй файл символьным устройством (виртуальная консоль обрабатывает входные данные побайтно).

Очевидно, что для драйверов файловой системы ОССН и подсистемы безопасности *PARSEC* файлы устройств представляют собой обычные файловые объекты, к которым применимы как стандартные вызовы по работе с файлами, так и функции назначения Уровней конфиденциальности или целостности.

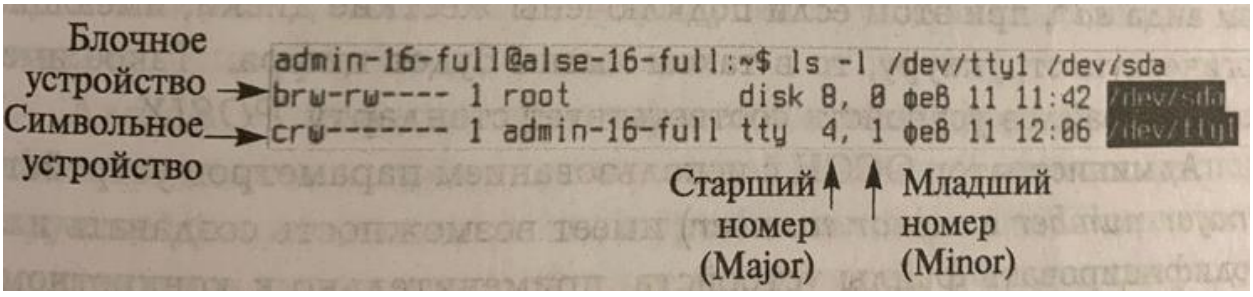


Рис. 1.50. Пример ввода *inode* файлов устройств */dev/sda* и */dev/tty1*

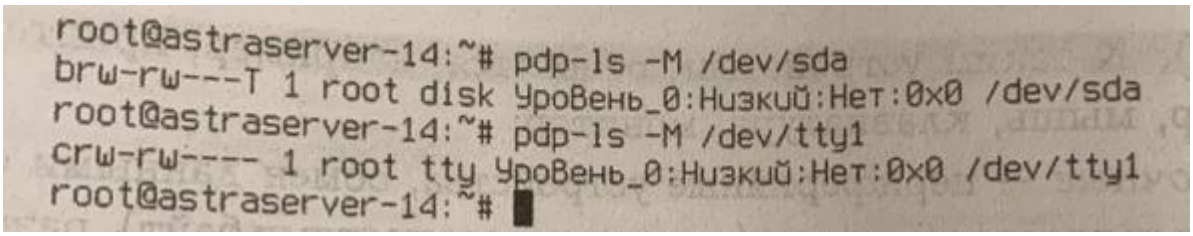


Рис 1.51. Пример использование команды pdp-ls-M для вывода уровней конфиденциальности файлов устройств /dev/sda и /dev/ttyl

На рис. 1.51 представлен пример вывода команды pdp-ls-M (ее функциональность и синтаксис будут подробно рассмотрены в главе 3), которая является аналогом команды *ls c* расширенными функциями отображения меток конфиденциальности файлов и каталогов. При этом для файлов символьных (файл */dev/ttyl)* и блочных (файл */dev/sda)* устройств могут быть назначены уровни конфиденциальности и целостности, а также заданы неиерархические категории. Благодаря такому использованию контекста безопасности для файлов устройств можно задать правила мандатного управления доступом и контроля целостности, как и для других сущностей ОССН.

Именование файлов запоминающих устройств в большинстве случаев стандартизовано для всех версий ОС проекта *GNU/Linux.* Важной особенностью ядра ОССН является то, что запоминающие устройства с интерфейсами *ATA (IDE), SATA (eSATA)* и USB подключаются к нему не напрямую, а опосредованно, через драйвер запоминающих устройств с интерфейсом SCSI (рис. 1.52). Связано это с тем, что реализация системы команд интерфейса *SCSI* существует как поверх интерфейсов *ATA/SATA* (называется *ATAPI— ATA Packet Interface),* так и поверх протокола USB (называется *MSD- Mass Storage Device).* Они позволяют подключать в ОССН любые *ATA, SATA* и USB запоминающие устройства, не разрабатывая для них собственного протокола обмена, а используя имеющийся в системе драйвер интерфейса *SCSI.* Такая модификация стека драйверов запоминающих устройств была реализована в рамках проекта *FLOSS (Free/Libre Open Source Software)* [139] и получила название объединённая подсистема *ATA-SCSI.* Согласно этому подходу устройствам в каталоге / *dev* соответствуют файлы устройств с именем вида *sd ,* при этом если подключены жёсткие диски, имеющие логическую структуру, то в таком имени будет цифра. Такое именование файлов устройств соответствует стандарту *POSIX.*

Администратор ОССН с использованием параметров устройств (major number и minor number) имеет возможность создавать или модифицировать файлы устройств, применительно к конкретному перечню устройств, имеющихся на компьютере.

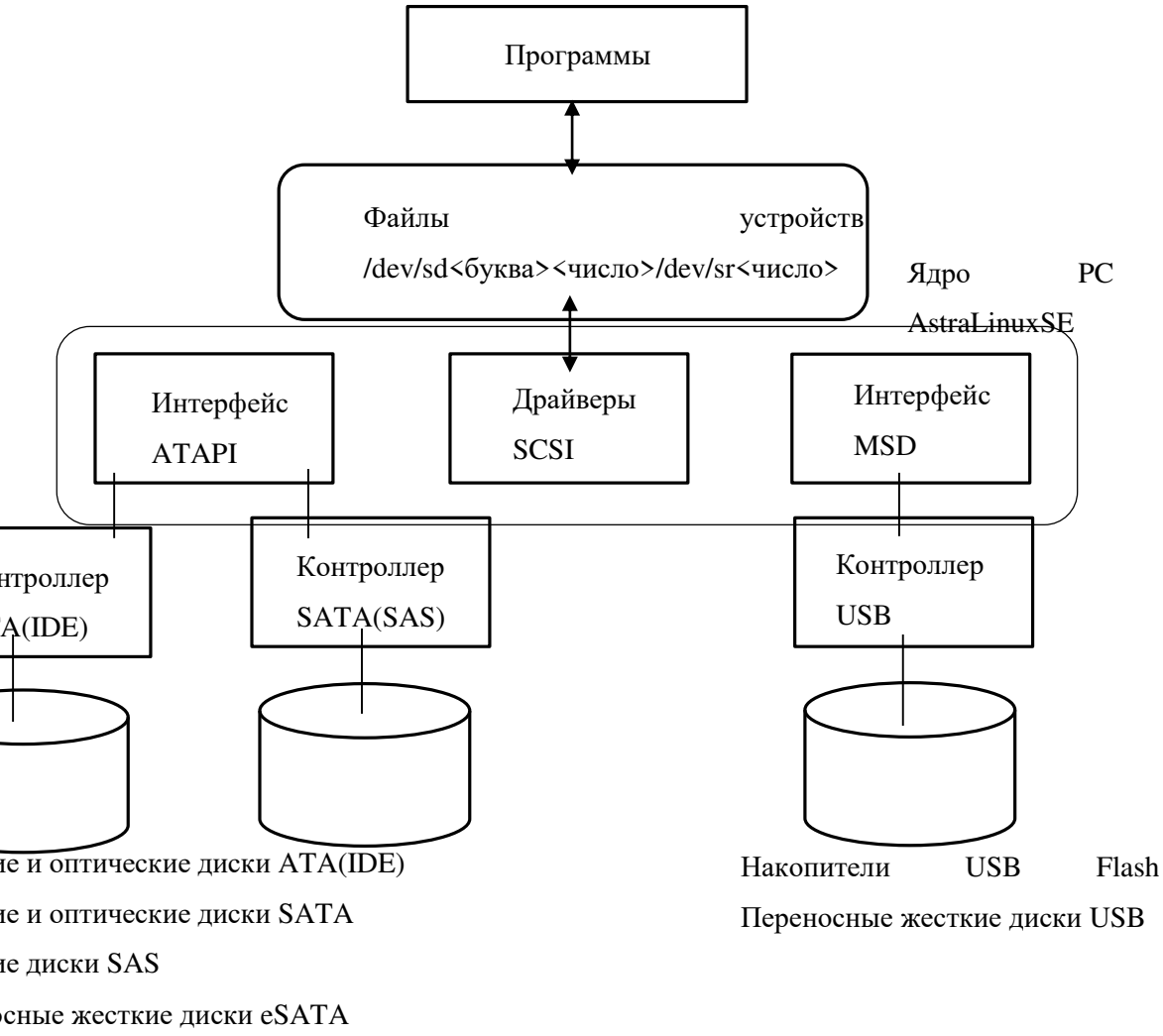


Рис. 1.52. Схема реализации стека драйверов запоминающих устройств на основе объединенной подсистемы ATA-SCSI

Однако подобный подход является не совсем удобным в современных условиях, когда разнообразие типов и число конкретных реализаций устройств существенно возросло. Поэтому в версиях ОС проекта *GNU/Linux,* начиная с ядра 2.6.т, включая ОССН, начиная с версии 1.4 и выше, применяется система динамического именования запоминающих устройств *udev,* которая использует идентификационную информацию непосредственно самих устройств, а не их абстрактные параметры *(major number* и *minor number).* К такой идентификационной информации относится: серийный номер устройства, его положение в SATA-интерфейсе или канале *PATA*- шины. Сочетание этой идентификационной информации устройств (и соответственно, их дисковых разделов) является уникальным, что позволяет динамически формировать уникальные имена для каждого устройства.

Для получения этой идентификационной информации система обращается к *sysfs* — виртуальной файловой системе, экспортирующей на пользовательский уровень из ядра ОССН данные о имеющихся в системе устройствах и драйверах. Эти данные могут быть получены администратором с помощью графической утилиты fly-admin-device-manager(элемент “Менеджер устройств” меню “Системные” главного меню) (рис. 1.53).

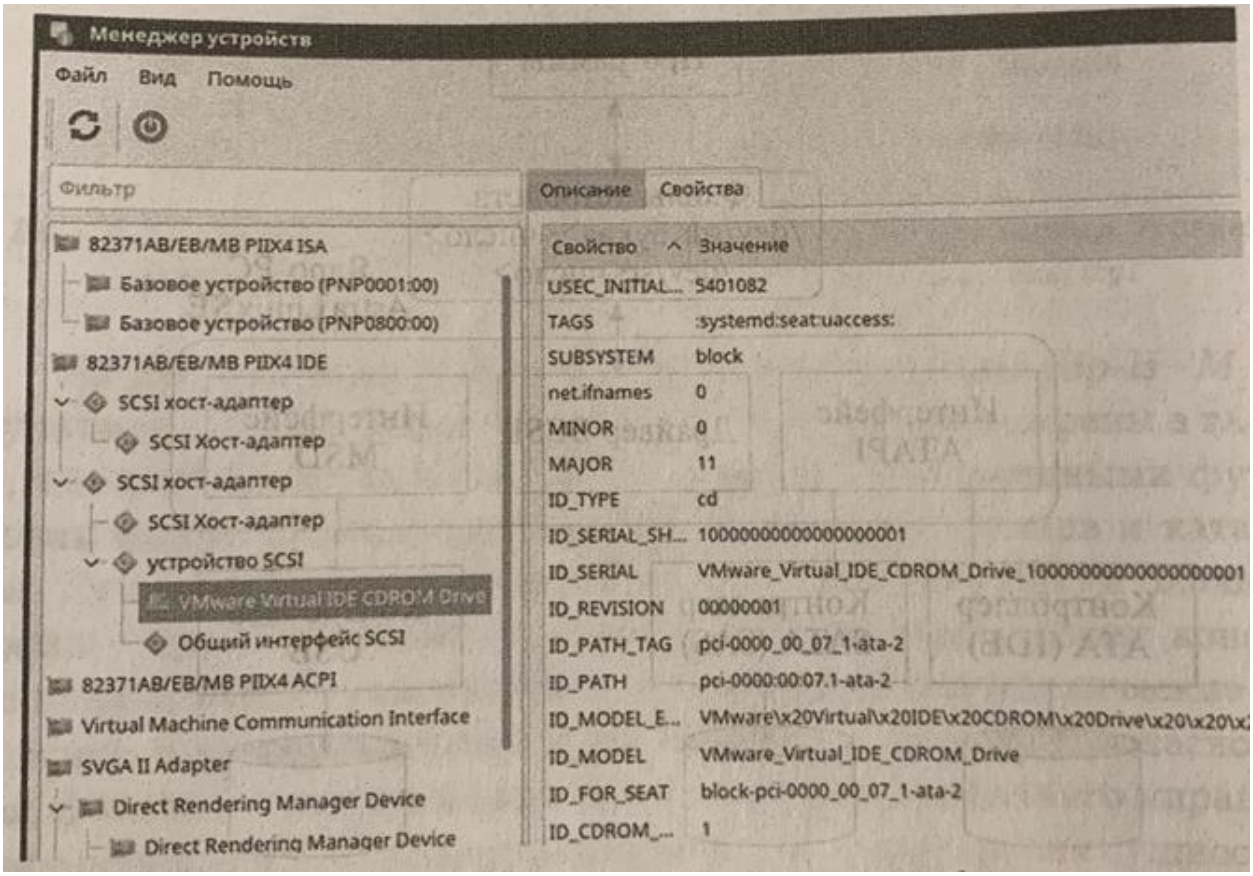


Рис. 1.53. Пример идентификационной информации о блочном устройстве *CDROM/DVDROM*, отображаемой утилитой “Менеджер устройств”

Система *udev* обеспечивает все необходимые средства для динамического создания и удаления файлов устройств и символических ссылок в каталоге / *dev,* её управление осуществляется командой *udevadm.* При этом система *udev* позволяет администратору

ОССН изменять порядок её работы, например, путём разработки собственных сценариев интерпретатора *bash.* Благодаря *udev* в каталоге */dev* находятся файлы только тех устройств, которые в настоящий момент подключены к системе. Если устройство отключается от системы, то файл устройства, связанный с ним, удаляется.

Функциональные возможности системы *udev* реализуются демоном (системным процессом, работающим в фоновом режиме без непосредственного взаимодействия с пользователем) *udevd,* порядок старта которого в ОССН указан в сценарии */ etc/init.d/udev* (рис. 1.54). Дополнительные параметры для этого демона могут быть указаны в конфигурационном файле /*etc/udev/udev. conf.*

Каждому устройству в ОССН демон *udevd* присваивает уникальный идентификатор UUID *(Universally Unique Identifier).* Его использование ядром ОССН (драйверами устройств) позволяет сделать независимым обработку обращений к устройству от текущих параметров его подключения. Это, например, удобно для переносных устройств вида eSATA жестких дисков, USB-дисков, интерфейс подключения которых может меняться.

```
#!/bin/sh -e
### BEGIN INIT INFO
# Provides:          udev
# Required-Start:    mountkernfs
# Required-Stop:
# Default-Start:     S
# Default-Stop:
# Short-Description: Start systemd-udevd, populate /dev and load drivers.
### END INIT INFO

# we need to unmount /dev/pts/ and remount it later over the devtmpfs
unmount_devpts() {
  if mountpoint -q /dev/pts/; then
    umount -n -l /dev/pts/
  fi
  if mountpoint -q /dev/shm/; then
    umount -n -l /dev/shm/
  fi
:
```

Рис. 1.54. Пример сценария инициализации демона *udevd*

```
admin-16-full@else-16-full:/etc/init.d$ sudo blkid
/dev/sda1: UUID="3fa47601-b38a-4de6-95df-118d8d522523" TYPE="ext4" PARTUUID=
"3702e3c8-01"
/dev/sda5: UUID="de58dafb-7bc9-4951-be01-571fd2fb5c7e" TYPE="swap" PARTUUID=
"3702e3c8-05"
```

Рис. 1.55. Пример ввода команды *blkid*

Для получения используемых системой *udev* идентификаторов и параметров блочных устройств администратор может использовать команду *bikid,* которая без параметров выведет данные файла /*etc/blkid.tab* (рис. 1.55), обновляющегося при каждой загрузке ОССН.

В ходе динамического именования устройств система *udev* также может выполнять ряд дополнительных действий, которые могут быть описаны в виде правил системы *udev (udev rules),* хранящихся в файлах с расширением *.rules* в каталоге */etc/udev/rules.d.* При этом одному устройству может соответствовать больше одного правила, что позволяет задавать для каждого устройства разные альтернативные имена, а также определять разные дополнительные действия, ассоциированные с этими именами. Система *udev,* обнаружив соответствующий устройству файл правил, будет продолжать просматривать каталог */ etc/udev/rules, d* в поисках других файлов правил, в которых указан *UUID* этого устройства.

Благодаря возможности автоматического формирования файлов правил системой *udev* подсистема безопасности *PARSEC* в ОССН реализует следующие дополнительные функции по работе с устройством:

- регистрация устройств в локальной базе учета (в случае автономной рабочей станции) или базе учета, хранящейся в базе учета контроллера домена *ALD* (в случае клиента домена *ALD*);
- управление доступом к зарегистрированным устройствам на основе политики безопасности, основанной на уровне их конфиденциальности и целостности.

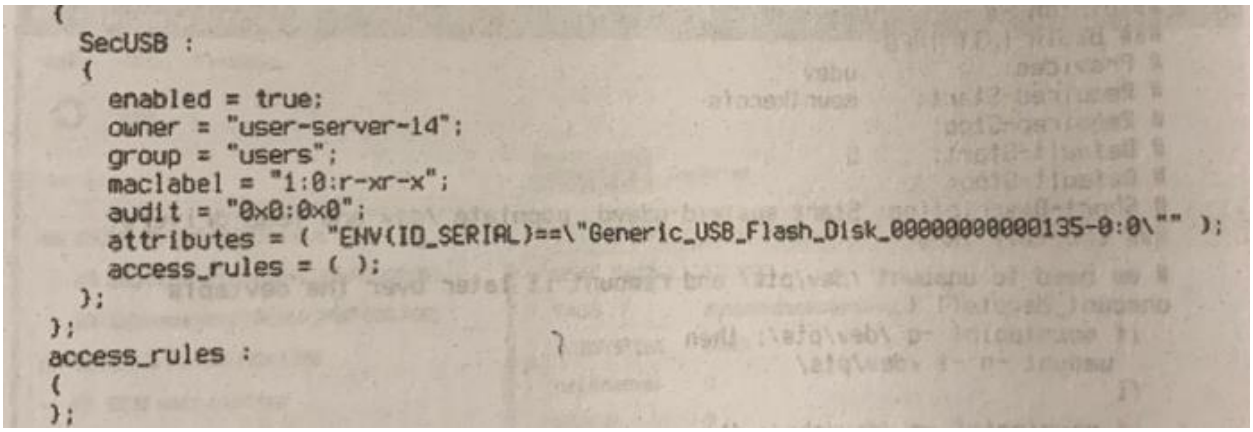


Рис. 1.56. Пример формата секции база учета устройств подсистемы безопасности *PARSEC* (на примере регистрации USB-носителя)

В случае локально* регистрации устройств база учёта создаётся в конфигурационном файле *etc/ parsec/ devices, cfg.* Для каждого зарегистрированных устройств формируется отдельная секция ограниченная блоком вида “{...}” (рис. 1.56).

Наряду с идентификационными данными устройства соответствующая ему секция содержит данные о дискреционных правах доступа к нему, а также о его мандатных уровнях конфиденциальности и целостности. Задание последних выполняется с использованием графической утилиты «Управление политикой безопасности» *fy-adrmn-smc)* в разделе «Устройства и правила» (рис. 1.57).

На основе данных из базы учёта устройств подсистема безопасности *PARSEC* автоматически генерирует файлы правил системы иаеъ, соответствующие учтённым устройствам, в состав которых народу с базовыми соответствиями *{matches)* и действиями *{actions)* добавлено соответствие *ENV{ID_SERIAL},* включающее действия *OWNER, GROUP* и *MACL ABEL,* связанные с управлением доступом к устройству. Пример формата такого правила следующий:

ENV{ID_SERIAL}==”Generic_USB_Flash_Disk_00000000000135-0:0”,
OWNER=”user-server-14”, GROUP=”user” MACLABEL=”1:0r-xr-x”

Поскольку в процессе учёта устройства его идентификатор UUID ассоциируется с идентификационными данными учётной записи пользователя, для блочных устройств, требующих выполнения операции монтирования имеющихся на них файловых систем, в ОССН используется модификация стандартной утилиты mount. допускающей выполнение монтирования не только процессам от имени учетной записи администратора системы, но и от имени непривилегированного пользователя, зарегистрированного в текущем сеансе.

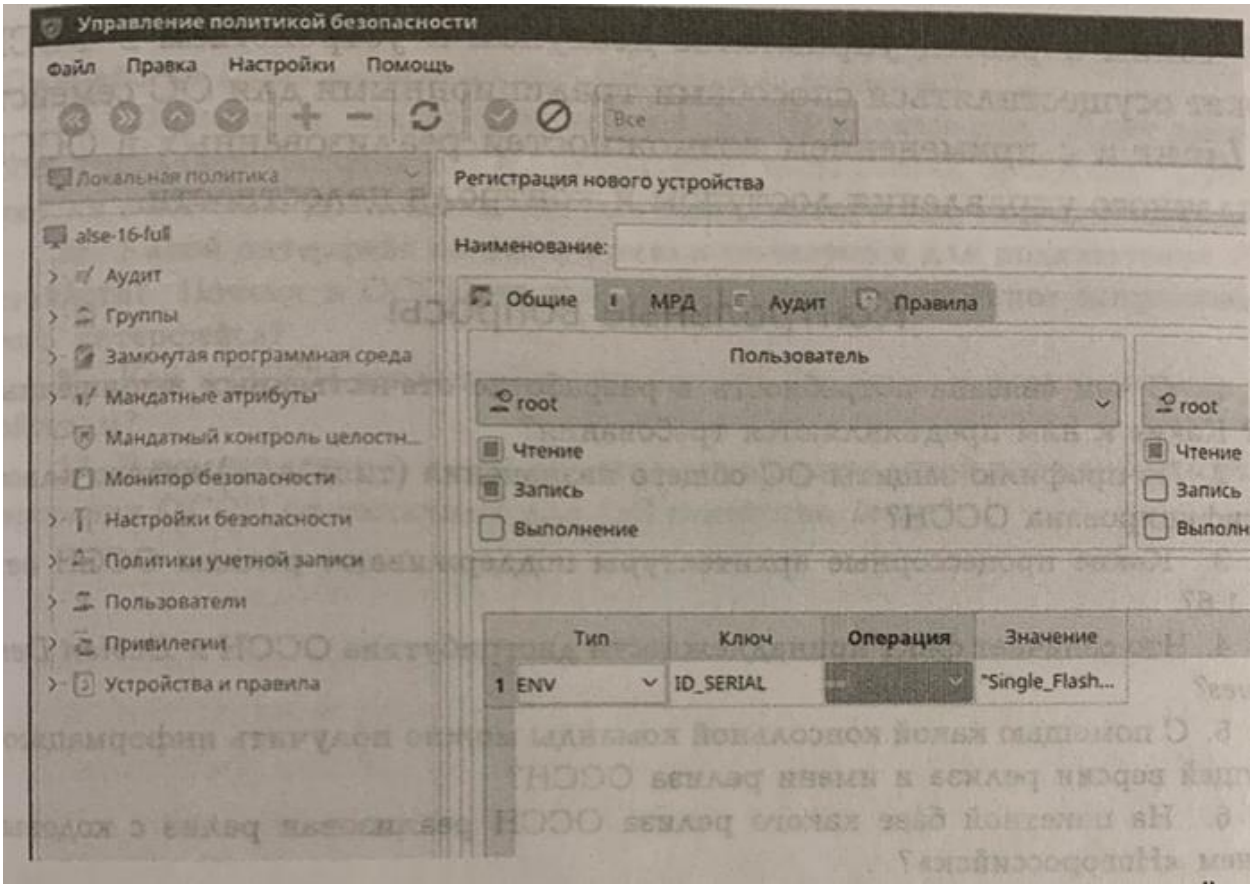


Рис. 1.57. Раздел управления доступом к зарегистрированному устройству

При этом точкой монтирования файловой системы будет соответствующий каталог в каталоге */ run/user/ %uid%/media* данного пользователя, для которой будут применяться соответствующие уровни конфиденциальности и целостности, заданные ключом *maclabel* в секции, соответствующей устройству в базе учёта (в файле *devices, cfg).*

Для обеспечения такого способа монтирования файловых систем в конфигурационном файле */etc/fstab* включены соответствующие записи с ключами *owner* и *group* для следующих видов блочных устройств:

- учтённые съёмные устройства для локально зарегистрированных учётных записей пользователей;
- учтённые съёмные устройства для учётных записей пользователей домена *ALD;*
- файловые системы FA732, *NTFS* и *EXTFS,* расположенные в разделах учтённых несъемных блочных устройств;
- Файловые системы *UDF* и ISO9660, расположенные на учтённых оптических дисках.

Примером записи в файле */etc/fstab* для учтённых съёмных устройств локально зарегистрированных пользователей является следующая запись:
/dev/s/home/*/media/*auto owner, group, noauto, noexec, iocharset=utf8, defaults 0 0.*

Таким образом, управление доступом к устройствам в ОССН может осуществляться способами традиционными для ОС семейства Linux и с применением возможностей реализованных в ОССН мандатного управления доступом и контроля целостности.

Контрольные вопросы

1. С чем связана потребность в разработке отечественных защищенных ОС? Какие к ним предъявляются требования?
2. По профилю защиты ОС общего назначения (типа А) какого класса сертифицирована ОССН?
3. Какие процессорные архитектуры поддерживают релизы ОССН версии 1.6?
4. Что означает факт принадлежности дистрибутива ОССН к Debian Derivatives?
5. С помощью какой консольной команды можно получить информацию о текущей версии релиза и имени релиза ОССН?
6. На пакетной базе какого релиза ОССН реализован релиз с кодовым именем “Новоросийск”?
7. Какова область применения ОССН?
8. Как реализована в ОССН подсистема безопасности *PARSEC,* и какие функции она обеспечивает?
9. Какие дополнительные защитные функции реализованы в усиленном *(Hardened)* варианте ядра ОССН?
10. В чем состоят основные функциональные отличия между технологиями формирования доменной архитектуры, поддерживаемыми ОССН?
11. Какие модули ядра в составе ОССН релиза 1.6 относятся к невыгружаемым?
12. В каком терминале (устройство *tty*)* в ОССН по умолчанию реализуется доступ к защищённому рабочему столу *Fly?*
13. Какая библиотека подсистемы защищённого рабочего стола *Fly* не зависит от библиотек платформы *Qt5* и является решением разработчика ОССН?
14. 14 Какую функцию в составе подсистемы защищённого рабочего стола Fly реализует библиотека *Ubparsec-mac-qt5-l?*
15. Какие компоненты входят в состав технологического стека виртуализации, реализованного в ОССН релиза 1.6?
16. Какие протоколы доступа к удалённому рабочему столу гостевого домена используются в стеке виртуализации ОССН релиза 1.6?
17. На основе каких компонентов реализован отказоустойчивый кластер ОССН релиза 1.6?
18. Каково назначение сервиса keepalived для организации кластера на базе ОССН релиза 1.6?
19. Каково назначение файловой системы Ceph в состав ОССН релиза 1.6?

20. В чем разница между вложенной и отдельной пользовательскими сессиями?
21. Какие права имеет администратор по управлению учетными записями пользователей и групп пользователей?
22. С какой целью для учетной записи пользователя ОССН определяются первичная и вторичная группы, и с какими правилами МРОСЛ ДП-модели можно ассоциировать их администрирование?
23. Чему в МОРСЛ ДП-модели соответствует наследственное в ОССН от ОС семейства Linux дискреционное управление доступом?
24. С помощью какого механизма в ОССН пользователь имеет возможность управлять состояниями и приоритетом выполнения процессов? Существуют ли в МОРСЛ ДП-модели правила, соответствующие такому механизму?
25. Какой интерфейс по умолчанию используется для подключения USB-устройств? Почему в ОССН не реализовано непосредственное использование этого интерфейса?
26. Как в ОСН реализовано управление доступом к подключаемым устройствам?
27. В чем сходство, а в чем отличия пользовательской работы и администрирования ОССН от типичных для ОС семейства Linux?

2 Мандатная сущностно-ролевая ДП-модель управления доступом и информационными потоками в операционных системах семейства Linux

2.1 Подход к формированию модели в ее иерархическом представлении

Традиционно механизм управления доступом рассматривается как основа обеспечения безопасности компьютерных систем, в особенности защищённых ОС с мандатным управлением доступом. Для научного исследования свойств этого механизма за более чем 40 лет построены десятки теоретических моделей [17, 105]. Однако большинство отечественных разработчиков защищённых ОС используют (как правило, только формально) для реализации механизма управления доступом устаревшие, неадекватные условиям функционирования современных ОС модели (например, классическую модель Белла-ЛаПадулы [17, 103], ролевые модели семейства *RBAC* [17, 133, 134]), что не позволяет повысить доверие к безопасности этого механизма.

При этом для современных защищённых ОС актуальна разработка формальной модели, обеспечивающей возможность сочетания востребованного в них мандатного управления доступом с перспективным ролевым управлением доступом. «Механическое» объединение этих двух видов управления доступом без учёта специфики каждого из них может негативно отразиться на безопасности полученного решения, в том числе при обеспечении защиты от запрещённых информационных потоков по памяти или по времени. Например, запрещённые информационные потоки по времени от сущностей с высоким уровнем конфиденциальности к сущностям с низким уровнем конфиденциальности (“аналогичные рассмотренным в [15]) назначения и субъектами с использованием назначения и отзыва в согласованные моменты времени прав доступа ролей.

Изложенный в [17, 134] подход к созданию на основе *RBAC* мандатного ролевого управления доступом не может быть эффективно реализован в реальных защищенных ОС, так как для противодействия запрещенным информационным потокам по памяти он предполагает возможность разделения всех ролей по уровням конфиденциальности сущностей, права доступа к которым они предоставляют, и по видам доступа к ним роли-«на чтение» и роли-«на запись», с применением соответствующих ограничений на функции текущих ролей сессий (субъектов) и прав доступа ролей. При этом не рассматриваются механизмы защиты от запрещённых информационных потоков по времени и невозможно (без использования в дополнение ролевому дискреционного управления доступом) задание индивидуальных прав доступа пользователей.

Рассматриваемая в пособии мандатная сущностно-ролевая ДП- модель управления доступом и информационными потоками в операционных системах семейства Linux (МРОСЛ

ДП-модель) является результатом применения апробированных на классических моделях [17, 105] и семействе ДП-моделей [14, 17] подходов к теоретическому анализу безопасности управления доступом и информационными потоками и адаптации моделей к условиям функционирования современных компьютерных систем. Модель впервые [16, 20, 76] обеспечивает сочетание мандатного управления доступом, мандатного контроля целостности и ролевого управления доступом, позволяющее через представление ролей сущностями-контейнерами (элементами виртуальной файловой системы ОССН) обеспечить возможность противодействия созданию нарушителем с использованием параметров ролей запрещённых информационных потоков по времени «сверху-вниз» (от сущностей с высоким уровнем конфиденциальности к сущностям с низким уровнем конфиденциальности). При этом нацеленность модели на реализацию в ОССН позволила сфокусировать её разработку на формировании комплексного научно обоснованного технического решения [18], заключается в следующем:

- включение в модель элементов типичных для ОС семейства Linux, в особенности для механизмов защиты ОССН;
- строгое доказательство существенных для практического применения ОССН условий её безопасности в первую очередь безопасности мандатного и ролевого управления доступа и мандатного контроля целостности (в том числе безопасности информационных потоков по памяти и по времени),
- обеспечение возможности с использованием методик верификации кода программ с применением инструментов дедуктивного доказательства *Rodin Platform (Event-B), Frama-C (Why3)* [19, 93,94,110,114] осуществить вывод и формальное обоснование корректности самой модели, а также корректности ее реализации непосредственно в программном коде ОССН.

Ряд элементов модели, например ролевое управление доступом, еще не используются в ОССН. Эти элементы включены в модель для создания условий для разработки будущих версий ОССН на основе изначально прошедших через теоретический анализ технических решений.

Несмотря на достаточно детальную проработку в МРОСЛ ДП-модели порядка функционирования механизма мандатного и ролевого управления доступом и мандатного контроля целостности ОССН, модель продолжает своё развитие, становясь, с одной стороны, «ближе» к ОССН, с другой стороны, позволяя строго теоретически обосновывать в рамках модели новые проверяемые на практике условия безопасности ОССН. Таким образом, представленное в настоящем пособии описание модели имеет ряд существенных отличий от изложенного в [17] и в первом (втором) издании настоящего учебного пособия [4].

Так, в [17, 4] описание МРОСЛ ДП-модели являлось «монолитным», т. е. элементы модели давались в порядке, принятом в классических моделях, который удобен при относительно небольшом объёме описания модели. В начале такого описания приводились элементы состояний задаваемой в рамках модели абстрактной системы, далее описывались требования к реализации в системе мандатного и ролевого управления доступом и мандатного контроля целостности, затем давались правила преобразования состояний системы, и s заключение формулировались и обосновывались условия безопасности системы. В результате модель в её «монолитном» описании стало все труднее корректировать, так как каждое изменение в каком- либо её элементе требовало учёта во всех других связанных с ним элементах модели, а также проверки справедливости большинства обоснованных в модели утверждений. Кроме того, из-за большого объёма и «монолитности» модели, невозможности в таком виде её поэтапной реализации затруднялось использование модели разработчиками ОССН, а также создание на её основе новых моделей (например, модели гипервизора или СУБД).

Все перечисленное стало причиной переработки МРОСЛ ДП- модели в её иерархическое представление. Первоначально при его разработке в модель (по сравнению с её «монолитным» представлением) не добавлялись новые элементы, а основной целью являлось формирование следующих четырех упорядоченных уровней [21]:

- первый уровень (базовый) модель системы ролевого управления доступом;
- второй уровень модель системы ролевого управления доступом и мандатного контроля целостности
- третий уровень модель системы ролевого управления доступом, мандатного контроля целостности и мандатного управления доступом только с информационными потоками по памяти
- четвертый уровень модель системы ролевого управления доступом, мандатного контроля целостности и мандатного управления доступом с информационными потоками по памяти и по времени.

Каждый нижний уровень иерархического представления модели соответствует абстрактной системе, элементы которой не зависят от новых элементов, принадлежащих более высокому уровню модели, который, в свою очередь, наследует, а при необходимости корректирует или дополняет элементы нижнего уровня. Такой подход позволяет постепенно усложнять формулировки определений и утверждений модели по мере включения в неё соответствующих очередному рассматриваемому уровню элементов.

Следующим шагом стало включение в иерархическое представление МРОСА ДП-модели новых уровней, содержащих элементы, до этого не использованные в «монолитном» представлении.

Во-первых, практика использования МРОСА ДП-модели при реализации ОССН показала, что в ряде случаев применение ролей, обладающих только правами доступа к сущностям, разрешающим субъектам получение соответствующих доступов к ним, является недостаточным и создаёт неудобство при администрировании ОССН. В (79) описываются ситуации, когда, например, только одну учётную запись пользователя системы необходимо лишить права доступа, предоставляемого ей через роль, которой обладают все учётные записи пользователей системы, при этом все другие права доступа этой роли должны быть оставлены без изменений. В связи с этим в иерархическое представление модели был добавлен уровень запрещающих ролей [23]. Запрещающие роли, в отличие от «обычных» ролей, содержат права доступа к сущностям или субъектам, которые не разрешают, а наоборот, запрещают получение соответствующих доступов. Этот уровень основан на уровне ролевого управления доступом модели, на нем не реализовано мандатное управление доступом и мандатный контроль целостности, а для задания запрещающих ролей используется описанный еще в ролевых моделях семейства *RBAC* механизм ограничений *(constraint).*

Во-вторых, изначально в МРОСЛ ДП-модели для задания мандатного контроля целостности применялись всего два уровня целостности: высокий (i_high) и низкий (i_low). Этого было вполне достаточно для разделения всех элементов ОССН на системные (доверенные компоненты ОССН, обеспечивающие функционирование процессов её ядра и системных процессов, в том числе механизмов защиты и администрирования), обладающие высоким уровнем целостности, и пользовательские (недоверенные компоненты ОССН, выполняющие функции процессов непривилегированных пользователей, в том числе нарушителей), обладающие низким уровнем целостности. Именно в таком виде мандатный контроль целостности реализован в версиях 1.4 и 1.5 ОССН. Но при использовании в ОССН технологий виртуализации (гипервизора), например, на основе программного комплекса средств виртуализации «Брест» [53], а также при реализации сетевой доменной архитектуры, требуются дополнительные уровни целостности. Например, одним из возможных решений здесь может быть использование трёх уровней целостности: высокого, соответствующего системному для основной ОССН, среднего, соответствующего системному для ОССН, запущенной в среде виртуализации (виртуализированной), и низкому — пользовательскому для основной и виртуализированной ОССН. Кроме того, в перспективе уровней целостности может потребоваться больше, в том числе для реализации невырожденной (состоящей из более чем двух уровней, где не каждый уровень сравним с каждым) решётки уровней целостности. В связи с изложенным был разработан уровень мандатного контроля целостности с невырожденной решёткой уровней целостности [22].

В-третьих, наиболее интересным стало формирование уровня ролевого управления доступом с запрещающими ролями и мандатного контроля целостности с невырожденной решёткой уровней целостности МРОСЛ ДП-модели, основанном не на одном, а впервые на двух предшествующих уровнях. В результате было показано, что структура иерархического представления модели может развиваться не только «древовидно». Она позволяет после разработки уровня, содержащие существенные новые элементы, далее дополнить этими элементами существующие «верхние» уровни модели, ориентированные на мандатное управление доступом с контролем информационных потомков по памяти и по времени. Именно это было осуществлено далее при создании уровней ролевого управления доступом с запрещающими ролями, мандатного контроля целостности с невырожденной решеткой уровней целостности и мандатного управления доступом с информационными потоками по памяти, по памяти и по времени.

В-четвёртых, было осуществлено формирование уровней предназначеных для моделирования единого механизма управления доступом ОССН и СУБД *PostgreSQL*|24|, аналогично включающих уровни ролевого управления доступом, мандатного контроля целостности, мандатного управления доступом с информационными потоками по памяти и по времени.

Однако такое ветвление и усложнение структуры иерархического представления МРОСЛ ДП-модели, включающее более 10 уровней, в том числе ориентированных на СУБД *PostgreSQL*, показало, что без оптимизации этой структуры дальнейшее развитие модели будет затруднено. Причём такая оптимизация фактически следовала из практики внедрения модели в ОССН. Например, с инженерной точки зрения добавление запрещающих ролей при реализации в ОССН ролевого управления доступом не составляет особой трудности, а после начала применения в ОССН версии 1.6 невырожденной решётки уровней целостности наличие в модели отдельного уровня всего для двух уровней целостности утратило смысл.

В связи с этим ряд уровней МРОСЛ ДП-модели были объединены, в результате их структура стала соответствовать изначально использованной, состоящей из перечисленных ранее четырёх уровней (рис. 2.1). При этом «параллельно» в этой структуре сформированы уровни для СУБД *PostgreSQL* и рассматривается возможность добавления уровней для моделирования механизма управления доступом гипервизора.

Поскольку полное описание МРОСЛ ДП-модели имеет значительный объем, в настоящем учебном пособии детально приводится только уровень ролевого управления доступом (первый уровень) этой математической модели (в отличие от (33) этот уровень включает запрещающие роли). Однако используемая при этом последовательность изложения содержания уровня полностью соответствует применяемым для всех

последующих уровней иерархического представления модели, которые в силу большого объема их описание приводится фрагментарно.

Структура описания уровня ролевого управления доступом иерархического представления МРОСЛ ДП-модели состоит из четырех частей, следующих далее. Сначала определяются структуры данных, составляющие модель, в первую очередь необходимые для описания элементов состояний рассматриваемой в рамках модели абстрактной системы. Затем приводятся ограничения на связи между элементами разных видов

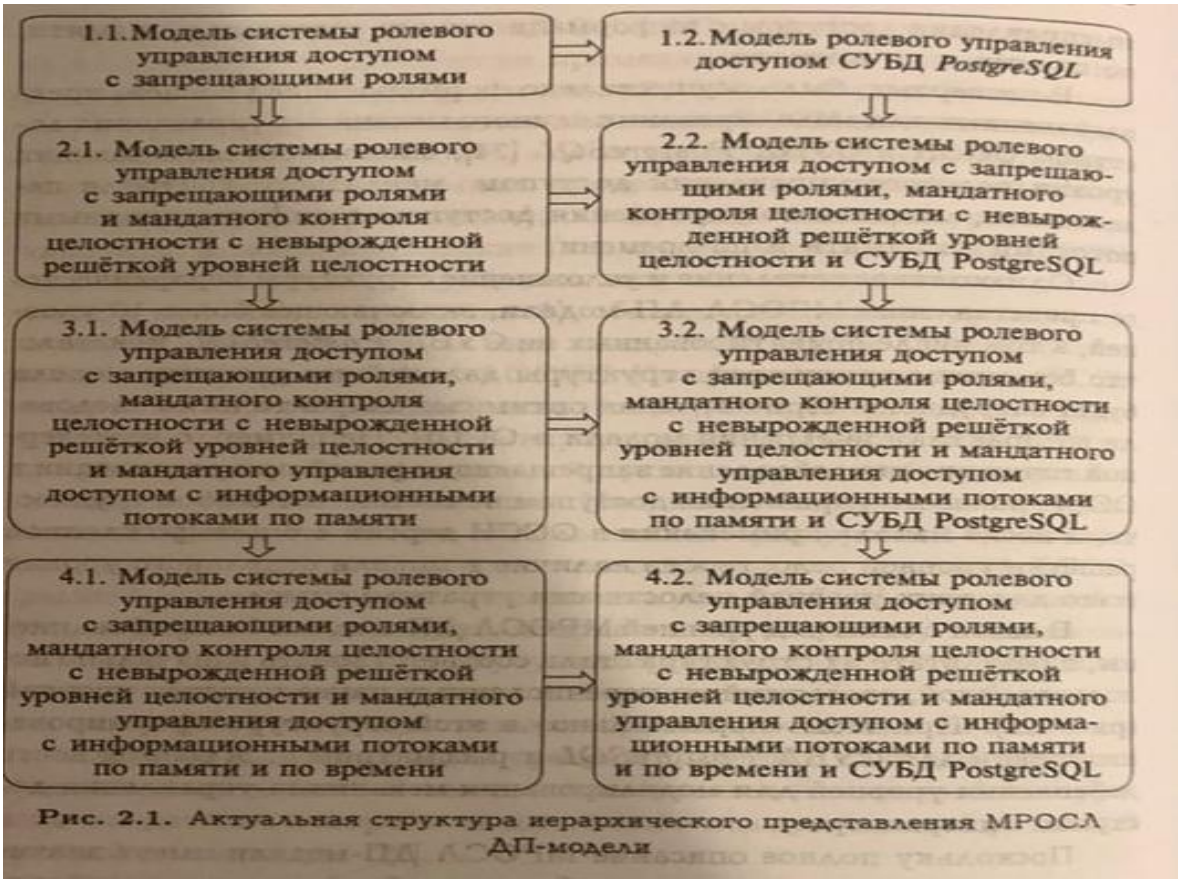


Рис. 2.1. Актуальная структура иерархического представления МРОСЛ ДП-модели

регламентируются условия корректности состояний системы, а также переходов системы из состояния в состояние. Далее описываются правила перехода системы из состояния в состояние, которые соответствуют выполнению тех или иных действий в механизме управления доступом ОССН, инициированные непривилегированными пользователями администраторами безопасности, а также функционирующими от их имени процессами В заключение формулируется утверждение о корректности правил перехода системы из состояния в состояние, т.е. выполнения при их применении условий корректности состояний системы и самих переходов из состояния в состояние. После описания уровня ролево-[4] управления доступом приводятся описания основных элементов последующих трёх уровней. При этом даются пояснения возможных способов или особенностей их реализации в ОССН.

2.2 Уровень ролевого управления доступом

2.2.1 Элементы состояния системы

Рассмотрим уровень ролевого управления доступом иерархического представления МРОСА ДП-модели. Сделаем следующее предположение (предложения в рамках формальной модели описывают на неформальном языке её базовые свойства, что обеспечивает её лучшее понимание; они формулируются по ходу описания модели, реализуются формально в элементах модели и используются в доказательствах утверждений и теорем).

Предположение 2.1. В рамках МРОСЛ ДП-модели пользователям соответствуют их учётные записи.

Дадим следующие определения и используем обозначения:

U — конечное непустое множество учётных записей пользователей;

S — конечное непустое множество субъект-сессий учётных записей пользователей (всех «активных» компонент защищённой ОССН);

user: *S*> *U* — функция принадлежности субъект-сессии учётной записи пользователя, задающая для каждой субъект-сессии учётную запись пользователя, от имени которой она активизирована;

E = O U *C* — конечное непустое множество сущностей (всех «пассивных» компонент защищённой ОССН, к которым назначаются права доступа), где *O* — множество объектов (например, файлов), *C* — множество контейнеров (например, каталогов) и *O* & *C =ø* .

Также определим:

NAMES — множество допустимых имён сущностей, ролей, запрещающих и административных ролей;

entity-name: C x *E*> 2^{NAMES} Функция имён сущностей в составе сущностей-контейнеров. При этом для любых контейнеров c, *c'* ϵ *C,* по определению выполняются условия:

- |*entity _name(c,* c<)| ≤ 1;
- если *c ≠ c'* и существует c" ϵ C такой, что *entity_name*(c,c")≠Ø, то *entity _name(c, c'")* = Ø
- существует единственная сущность- «корневой контейнер» *ROOT* ϵ *C* такая, что *entity_name(c, ROOT) =* Ø и, *если c≠ ROOT,* то существует единственная последовательность контейнеров c_1=c,c_2…,c_n= ROOT ϵ C такая, что n≥2 и *entity_name*(c_i,c_i-1)≠Ø, где 1 < i ≤ n.

Определение 2.1. *И*ерархией сущности называется заданное на множество сущностей E бинарное отношение " ≤ ", удовлетворяющее условию: для двух сущностей e,e' ϵ E выполняются отношения e≤e', когда либо e=e', либо существует последовательность сущностей e1=e,e_2,…,e_n ≠ e' ϵ E такая, что n≥2 и *entity_name*(e_i,e_i-1)≠Ø, где 1<i≤n. В случае, когда для двух сущностей e_1,e_2 ϵ E выполняется условие e_1≤e_2 и e_1≠e_2, будем говорить, что сущность e_1 содержится в сущности-контейнере e_2, и будем использовать обозначение e_1<e_2. Определим функцию иерархии сущностей H_E: E-> 2^E, где для e ϵ E выполняется H_E(e)={e' ϵ E: *entity_name*(e,e') ≠Ø}.

Определение 2.2. Иерархией субъект-сессий называется заданное на множестве S отношение частичного порядка «<», удовлетворяющее условию: если для субъект-сессии s ϵ S существую субъект-сессии s_1, s_2 ϵ *S* такие, что s ≤ s_2, s ≤s_1, то s_1≤ s_2 ила s_2≤ s_1. В случае, когда для двух субъект-сессий s_1, s_2 ϵ *S* выполняются условия s_1≤ s_2 и s_1≠ s_2, будем говорить, что субъект- сессия s_1 является потомком s_2, и будем использовать обозначение s_1< s_2. Определим H_s*: S> 2*^{s} — функцию иерархии субъект-сессий (сопоставляющую каждой субъект-сессии s ϵ *S* множество субъект-сессий *Hs(s)* ϵ *S,* непосредственно в ней содержащихся удовлетворяющую условиям:

- если субъект-сессия s_1 ϵ $H_s(s_2)$, то s_1 < s_2 и не существует субъект-сессии s ϵ *S* такой, что s_1 < s < s_2;
- для любых субъект-сессий s_1 s_2 ϵ S, s_1 ≠ s_2, выполняется равенство $H_s(s_1)$ & $H_s\{s_2)$ = Ø.

В рамках МРОСЛ ДП-модели иерархия сущностей задана не «абстрактной» функцией H_E, а реализуемой явно в ОССН функцией имён сущностей в составе сущностей-контейнеров *entity_name* что позволит в дальнейшем описать правила предоставления содержимого контейнеров (например, списков файлов и каталогов, выдаваемых в ОССН по команде ls) *с* учётом параметров мандатного управления доступом. При этом учтено наличие механизма создания “жестких” ссылок (*hard link)* в файловой системе ОССН, обеспечивающего возможность размещения сущностей-объектов одновременно в нескольких сущностях-контейнерах, в том числе несколько “жестких” ссылок на одну сущность-объект в составе одной сущности-контейнера. Также исключается возможность наличия у сущности-контейнера двух имен (двух “жестких” ссылок) в составе другой сущности-контейнера, а также описаны свойства корневой сущности-контейнера ROOT, как правило, соответствующей в ОССН корневому каталогу “/”.

По сравнению с предыдущими ДП-моделями определено, что множество субъектов-сессий не входит во множество сущностей. Это связно с тем, что в ОССН функции управления доступом к субъект- сессиям (процессам) и сущностям (файлам, каталогам) реализуются раздельно. Таким образом, иерархия на множестве субъект-сессий определяется независимо от иерархии сущностей и при реализации в ОССН может быть задана двумя способами. Первый способ, когда существует явная связь между родительскими субъект-сессиями и их потомками (например, когда при завершении работы родительской субъект-сессии завершают работу её субъект-сессии потомки, подчинённые родительской в иерархии). Второй способ, когда порождённая субъект-сессией другая субъект-сессия функционирует независимо, в этом случае подчинение её в иерархии родительской субъект-сессии нецелесообразно.

Используем следующие обозначения:

R — множество ролей;

NR — множество запрещающих ролей, при этом по определению R & NR = Ø;

AR — множество административных ролей, при этом по определению AR & NR = AR & R = Ø (административные роли — особый вид ролей, предназначенный для изменения множеств прав доступа ролей и запрещающих ролей, авторизации на роли, а также выполнения функций по администрированию системы);

SAR *c* AR — множество специальных административных ролей, которые не могут создаваться, удаляться, переименовываться, менять свои параметры в процессе функционирования системы;

R_r = {reed$_r$, write$_r$, execute$_r$, own$_r$) множество видов прав доступа;

R_a = {read$_a$, write$_a$} множество видов доступа;

P*c* (E *x* R_r) U (S *x* {own$_r$} множество прав доступа к сущностям и субъект-сессиям;

AP *c* (R *U* NR *U* AR) *x* R_r множество административных прав доступа к ролям, запрещающим ролям или административным ролям;

A *c* S *x* E **x** R_a- множество доступов субъект-сессий к сущностям;

AA *c* S x (R *U* NR*U* AR) *x* R_a- множество доступов субъект-сессий к ролям, запрещающим ролям или административным ролям;

PA: R U NR U AR → 2^P функция прав доступа к сущности ролей запрещающих ролей и административных ролей, при этом для каждого права доступа p ∈ P существует роль *r* ∈ R *U* NR *U* AR такая, что выполняется условие *p* ∈ PA(г);

APA: AR → 2^{AP} — функция административных прав доступа к ролям, запрещающим ролям и административным ролям административных ролей, при этом для каждого административного права доступа *ap* ∈ *AP* существует административная роль *ar* ∈ AR такая, что выполняется условие *ap* ∈ *APA(ar):*

Shared_container:. C U *R* U *NR* U *AR* —> *{true, false}* — функция разделяемых контейнеров такая, что сущность-контейнер, роль, запрещающая роль или административная роль *c* ∈ *C* U *R* U *NR* U *AR* является разделяемой, когда *shared_container(c) = true,* в противном случае *shared_container(c) = false;*

V: E U R U NR U AR → {(a_i,… ,a_n): a_1 ∈ {0,l}, 1 ≤ i ≤ n_1 n ≥ 1} U *{Ø}* — функция значений сущностей, ролей, запрещающих ролей и административных ролей (как аналогов сущностей-контейнеров), задающая возможность для каждой из них принимать значение любой (в том числе пустой) конечной последовательности битов;

role-name: (R U NR U AR) × (R *U NR.UAR)* → *NAMES* — функция имён ролей, запрещающих ролей и административных ролей в составе роли, запрещающей роли или административной роли (как контейнера). При этом для любых ролей, запрещающих ролей или административных ролей *r, r'* ∈ R U *NR* и *AR,* по определению выполняются условия:

- |*role.name(r, r')*| ≤ 1;

- если *role.name*(r, r') ≠ 0, то либо r, r' ∈ R, либо r, r' ∈ NR. либо r, r' ∈ AR;
- для роли, запрещающей роли или административной роли r'' ∈ R *U* NR *U* AR справедливо *role-name*{r, r") = *role-name*{r',r");
- если существует последовательность ролей, запрещающих ролей или административных ролей r_1 =r,r_2,…,r_n=r'∈ R *U* NR *U* AR такая, что n ≥ 2 и *role.name*{r_i,r_{i-1})≠Ø, где 1< i ≤ n, то не существует последовательности ролей, запрещающих ролей или административных ролей r'$_1$=r',r'$_2$,…,r'$_m$ = r ∈R *U* NR *U* AR такой, что m≥2 и *role.name*(r'$_i$, r'$_{i-1}$) ≠Ø, где 1< i ≤ m:

Поскольку традиционно в ОССН рассматриваются три вида прав доступа к сущностям: на чтение, на запись и на выполнение (или на использование контейнера каталога)а так же при дискреционном управлении доступом к сущностям и субъект-сессиям учитывается наличие у каждой из них уникального владельца, имебщего право передавать права доступа к ним другим учётным записям пользователей, то в рамках МРОСЛ ДП-модели будем использовать виды прав доступа read,., write$_r$, execute$_r$, own$_r$, и виды доступов read$_a$,, и wtite$_a$

Определение 2.3. *И*ерархией ролей, запрещающих ролей или административных ролей называется заданное на множестве ролей R, *NR* или *AR* соответственно бинарное отношение «≤», удовлетворяющее условию: для ролей, запрещающих ролей или административных ролей *r, r'* ∈ R U *NR U AR* выполняется отношение r ≤ r', когда либо r = r', либо существует последовательность ролей, запрещающих ролей или административных ролей r_1=r,r_2, ..., r_n = r' ∈ R *U NR U AR* такая, что n ≥ 2 и *role_name*(r_i,r_{i-1})≠Ø, где 1 < *i* ≤ *п.* В случае, когда для двух ролей, запрещающих ролей или административных ролей r_1, r_2 ∈ *R* U *N R* U *AR* выполняются условия $r_1 \le r_2$ и $r_1 \ne r_2$, будем говорить, что роль, запрещающая роль или административная роль r_1 содержится в роли, запрещающей роли или административной роли r_2, и будем использовать обозначение $r_1 < r_2$. Определим функцию иерархии ролей, запрещающих ролей или административных ролей H_R:*R* U *NR* U *AR* → 2^R U 2^{NR} U 2^{AR}, где для *r* ∈ *R U NR U AR* выполняется H_R *(r) = {r,' ∈ R U NR U AR*| *role_name(*r_1 r') ≠Ø}.

В отличие от других формальных моделей управления доступом, впервые предлагается считать роли, запрещающие или административные роли аналогом сущностей-контейнеров, к которым субъект-сессии могут иметь (через административные роли) права доступа и получать доступы. При этом в рамках МРОСЛ ДП- модели иерархии ролей, запрещающих ролей и административных ролей заданы (по аналогии с иерархией сущностей) не функцией H_R а функцией имён ролей в составе ролей-контейнеров *role_name*. Таким образом, право доступа own$_r$, — владелец роли, read$_r$ – право получать роль как текущую, просматривать её параметры, write$_r$, право изменять множество прав доступа роли, execute$_r$, — право обращаться к ролям, подчинённым данной роли в иерархии ролей (по умолчанию предполагается, что такое право доступа к ролям имеется всегда); доступ read$_a$

получение субъект—сессией роли как текущей, доступа $write_a$- изменение прав доступа роли или состава ролей подчиненных ей в иерархии.

Имеющиеся в ОССН привилегии целесообразно задать административными или “обычными” ролями, это обеспечит целостность механизма управления доступом в ОССН, и, кроме того, позволит в дальнейшем присваивать ролям-привилегиям уровни конфиденциальности и уровни целостности.

В результате закладывается основа единого механизма мандатного управления доступом и мандатного контроля цнлостности для доступов к сущностям, получения в качестве текущих и администрирования ролей субъект-сессиями, с возможностью противодействия В дальнейшем запрещённым информационным потокам по памяти или по времени.

Кроме того, для сущностей-контейнеров, ролей, запрещающих ролей и административных ролей задана функция разделяемых контейнеров (помечаемых в ОССН атрибутом «t») и функция их значений. При реализации иерархий ролей в ОССН по аналогии с файловой системой можно использовать виртуальную структур, сущностей-ролей, отличающуюся от структур для файлов наличием «жёстких» ссылок не только на роли-«объекты» (роли, которым в иерархии не подчинена ни одна другая роль), но и на роли-«контейнеры» (роли, которым в иерархии подчинена хотя бы одна роль).

В моделях семейства RBAC и других ролевых ДП-моделях предполагалось, что функция авторизованных ролей учётных записей пользователей UA обладает свойством «наследования» подчинённых ролей, т. е. выполняется следующее условие: для учетной записи пользователя $u \in U$, если роли r, r’, $\in R$ такие, что $r \in UA(u)$ и $r' \leq r$, то выполняется условие $r' \in UA(u)$. Это обеспечивается «наследованием» административной ролью права доступ- $read_r$ от данной роли ко всем подчинённым ей ролям в иерархии. При этом права доступа $wrte_т$ и own_r таким свойством не обладают, что может позволить гибко задавать «диапазоны» администрируемых ролей по аналогии с моделью ролевого администрирования ARBAC {17,134}.

Ранее в моделях семейства RBAC и ролевых ДП-моделях для задания текущих ролей использовалась функция roles, а для администрирования ролей функции вида can_assign, can_revoke и can_manage_rights, которые потребовали бы отдельной реализации в защищенной ОССН. В МРОСЛ ДП-модели для этого используется контролируемые системой мандатного управления доступом (на последующих уровнях модели) доступы субъект-сессий к ролям (задаются множество AA) и административные права доступа административных ролей (задаются функцией APA). При этом для обеспечения большего быстродействия ОССН при проверке прав доступа субъект-сессий к сущностям целесообразно для каждой

субъект-сессии хранить списки (по аналогии со списком привилегий) текущих ролей, к которым она имеет соответственно, доступы $read_a$ или $write_a$.

Для обеспечения применения запрещающих ролей {79} по аналогии с моделью RBAC {17,134} зададим механизм ограничений (Constraints) необходимого обладания запрещающей ролью с помощью следующей функции:

$constraint_{NR}$: R U AR → 2^{NR} функция необходимого обладания запрещающей ролью, задающая для каждой роли или административной роли множество запрещающих ролей, которые должна иметь субъект-сессия как текущие в случае обладания ею как текущей соответствующей ролью или административной ролью

Определение 2.4. Пусть определены множества учётных записей пользователей *U* , сущностей E, субъект-сессий *S,* прав доступа к сущностям *P,* доступов субъект-сессий к сущностям A, доступов субъект-сессий к ролям, запрещающим ролям и административным ролям *AA,* функции административных прав доступа к ролям административных ролей *APA,* прав доступа ролей *PA,* принадлежности субъект-сессий учётным записям пользователей *user,* иерархии ролей H_R, иерархии сущностей H_e, иерархии субъект- сессий H_S. Определим G = (APA, PA, user, A, AA, H_R,H_E,H_S, $constraint_{NR}$) — состояние системы.

Данное определение необходимо для теоретического описания состояний защищённой ОССН в рамках МРОСЛ ДП-модели и не требует непосредственной реализации в ОССН. Используем обозначения:

Σ(G*,OP)-система, при этом:

- G*- множество всех возможных состояний;
- OP- множество правил преобразования состояний, заданных в табл. 2.1 (см. разд. 2.2.3);
- Субъект-сессии не могут иметь друг к другу никаких доступов: для субъект-сессий s,s'∈ S выполняется {(s,s',a_a): a_a ∈ R_a}& A=Ø

Условие 2 (административные права доступа и иерархия ролей, запрещающих ролей или административных ролей):

- все роли, запрещающие роли и административные роли являются "разделяемыми контейнерами": для каждой роли, запрещающей роли или административной роли r ∈ R U NR U AR справедливо равенство *shared_container(r)*=true;
- у каждой административной роли есть права доступа *$ezecute_r$* ко всем ролям, запрещающим ролям и административным ролям: для каждой административной роли *ar* ∈ *AR,* роли, запрещающей роли или административной роли r ∈ *R.* *JNR* *J AR* выполняется условие (r, *$execute_r$) ∈ APA(ar).*

Условие 3 (доступы и права доступа, право доступа владения, администрирование параметров, прав доступа сущностей, ролей, запрещающих ролей, административных ролей);

- к ролям, запрещающим ролям, административным ролям и сущностям субъект-сессии могут иметь любые виды доступа из множества R_a,
- роли, запрещающие роли и административные роли могут иметь к сущностям любые права доступа из множества R_r, только административные роли могут иметь к ролям, запрещающим ролям или административным ролям административные права доступа из множества R_r;
- для управления доступом к сущности субъект-сессия должна иметь к ней через соответствующую текущую роль или административную роль право доступа владения и не должна иметь текущую запрещающую роль, обладающую правом доступа владения к этой сущности;
- для каждой сущности или субъект-сессии, если существует, то единственная роль или административная роль, обладающая к ней правом доступа владения, при этом для изменения роли или административной роли, обладающей правом доступа владения к сущности или субъект-сессии, необходимо наличие у субъект-сессии доступа на чтение к административной роли *entities_admin_role* или *subjects_admin_role* соответственно: для каждой сущности e ∈ E выполняется условие $|\{r \in R \cup AR:(e, own_r) \in PA(r)\}|\leq 1$, для каждой субъект-сессии s ∈ S выполняется условие $|\{r \in R \cup AR: (s, own_r) \in PA(r)\}|\leq 1$, где entities_admin_role, subjects_admin_role ∈ SAR;
- для каждой роли существует единственная административная роль roles_admin_role, обладающая к ней правом доступа владения, для каждой запрещающей роли существует единственная административная роль negative_roles_admin_role, обладающая к ней правом доступа владения, для каждой административной роли существует единственная административная роль admin_roles_role, обладающая к ней правом доступа владения: для каждой роли r ∈ R выполняется условие $\{ar' \in AR:(nr,own_r) \in APA(ar')\}=\{admin\_roles\_admin\_role\}$.
- для управления доступом к роли, запрещающей роли или административной роли субъект-сессия должна иметь доступ на чтение к административной роли *roles_admin_role, negative_roles_ admin_role* или *admin_roles-admin_role* соответственно.

Условие 4 (доступ к сущностям в иерархии сущностей). Для получения субъект-сессией любого доступа к сущности, создания «жёсткой» ссылки на неё, получения или изменения параметров, прав доступа к ней, активизации из неё субъект-сессии требуется существование последовательности непосредственно вложенных друг в друга сущностей-контейнеров, начинающейся с некоторой сущности-«корневой контейнер» (например,

корневой контейнер «/» в ОССН) и заканчивающейся сущностью-контейнером, в состав которой непосредственно входит сама сущность, и наличие у субъект-сессии текущих ролей или административных ролей, обладающих в совокупности правами доступа $execute_r$ ко всем сущностям-контейнерам этой последовательности, а также отсутствие у субъект-сессии запрещающей роли, обладающей правом доступа $execute_r$ хотя бы к одной сущности-контейнеру этой последовательности: для состояний системы G и G', правила преобразования состояний системы op ∈ OP таких, что $G\text{-}_{op}G'$, если субъект-сессия s ∈ S сущность e ∈ E, (s, e, a_a) ∉ A и (s,e,a_a) ∈ A', где $a_a \in R_a$, то существует контейнер c ∈ C и *execute_container(s, c, e) =true*

Условие 5 (создание, переименование или удаление сущности, роли, запрещающей роли или административной роли, или "жесткой» ссылки на неё, получение её параметров):

- для создания, переименования или удаления сущности, роли, запрещающей роли или административной роли или "жесткой" ссылки на нее в сущности-контейнере, роли, запрещающей роли или административной роли соответственно, субъект-сессии необходимо иметь к последней доступ на запись и текущую роль или административную роль, обладающую к последней правом доступа на выполнение $execute_r$;
- при создании, переименовании или удалении сущности или «жесткой» ссылки на нее в сущности-контейнере субъект-сессия не должна иметь текущую запрещающую роль, обладающую правом доступа на выполнение к этой сущности-контейнеру;
- для изменения прав доступа роли, запрещающей роли или административной роли субъект-сессии необходимо иметь к ней доступ на запись, за исключением случаев, когда либо административным ролям назначаются административные права доступа *read,* или *execute,* к ролям, запрещающим ролям или административным ролям при изменении их иерархии, либо удаляются сущности, субъект-сессии, роли, запрещающие роли или административные роли, и, соответственно, удаляются имеющиеся к ним у ролей, запрещающих ролей или административных ролей права доступа;
- для переименования, удаления сущности или «жёсткой» ссылки на сущность *e* в сущности-контейнере c (e ∈ HE (c)), помеченной как разделяемая *(shared_container(c) = true),* требуется наличие у субъект-сессии доступа на чтение к роли или административной роли, обладающей правом доступа владения *own,* к сущности e, и отсутствие у субъект-сессии, текущей запрещав щей роли, обладающей этим правом доступа к этой сущности;
- сущности-контейнеры при создании помечаются как неразделяемые. Для изменения метки разделяемости сущности-контейнера субъект-сессии необходимо либо обладать текущей ролью или административной ролью, имеющей право доступа владения к этой сущности-контейнеру, не обладать текущей запрещающей ролью, имеющей право доступа

владения к этой сущности-контейнеру, либо обладать административным доступом на чтение к административной роли *entities_admin_role*;

- для получения субъект-сессии данных о ролях, запрещающих ролях или административных ролях, обладающих правами доступа к сущности, требуется либо наличие у субъект-сессии роли или административной роли, обладающей правом доступа владения own_r к этой сущности, отсутствие текущей запрещающей роли, обладающей правом доступа владения к этой сущности, либо доступа на чтение к административной роли *entities_admin_role*;
- для получения субъект-сессии данных о ролях, запрещающих ролях или административных ролях, обладающих правом доступа владения к другой субъект-сессии или имеющихся у этой субъект-сессии текущих ролях, запрещающих ролях или административных ролях, требуется либо наличие у первой субъект-сессии доступа к роли или административной роли, обладающей правом доступа владения own_r *ко* второй субъект-сессии, отсутствие текущей запрещающей роли, обладающей правом доступа владения ко второй субъект-сессии, либо доступа на чтение к административной роли *subjects.admin.role;*
- для создания, переименования, удаления, получения параметров роли, запрещающей роли или административной роли, «жёсткой» ссылки на неё, числа «жёстких» ссылок к ней, множества административных ролей, обладающих к ней правами доступа, требуется наличие у субъект-сессии доступа на чтение к административной роли *roles.admin.role, negatıve.roles.ad- min.role* или *admin.roles.admin.role* соответственно:

Условие 6 (администрирование учетных записей пользователей):

- для создания или удаления учётной записи пользователя требуется наличие у субъект-сессии доступа на чтение к административным ролям *users.admin.role, negatıve.roles.admin.role*$\in$ *SAR,* доступов на чтение, а при создании и на запись, к административным ролям *roles.admin.role* и *admin.roles.admin.ro1e,* при этом множество субъект-сессий, функционирующих от имени данной учётной записи пользователя, должно быть пустым;
- для получения субъект-сессией параметров учётной записи пользователя она либо должна функционировать от её имени, и о иметь текущую административную роль *user_.admın_role.*

Условие 7 (создание и удаление субъект-сессий).

- Субъект-сессия может активизировать из сущности новую субъект-сессию от имени некоторой учётной записи пользователя только при наличии к сущности права доступа на выполнение у хотя бы одной из ролей, к которой активизирующая субъект-сессия имеет доступ на чтение, и отсутствии у нее текущей запрещающей роли, обладающей правом доступа на выполнение к этой сущности;

- Субъект-сессия может удалить субъект-сессию, только обладая текущей ролью или административной ролью, имеющей к ней право доступа владения, и не обладая текущей запрещающей ролью, имеющей право доступа владения ко второй субъект-сессии.

Условие 8 (вид метки):

- для каждой сущности, роли, запрещающей роли или административной роли задаётся вид её метки: прямая или косвенная, который не изменяется в процессе функционирования системы задаётся функция *direct E \J Pu NR(J AP → {true*, *false*}, где для *e ∈ E* U *R* U *NR* и *AP,* если *direct{e) = true,* то метка прямая, иначе косвенная;
- метка каждой роли, запрещающей роли или административной роли является прямой: для *г ∈ P* U *NP* U *AP* верно равенство *director) = true;*
- если у некоторой сущности-контейнера метка косвенная, то у всех сущностей ниже её в иерархии она также косвенная: если для сущности-контейнера *c ∈ C* выполняется *dırect(e·) false,* то для всех сущностей e ∈ *E* таких, что *e* ≤ c, выполняется *direct(e) = false;*
- для каждой сущности с косвенной меткой существует единственная старшая её в иерархии сущность-контейнер с прямой меткой, для которой все сущности, находящиеся ниже её в иерархии имеют косвенные метки, а права доступа всех ролей, запрещающих ролей или административных ролей к ним равны правам доступа к этой сущности-контейнеру: для каждой сущности e ∈ E такой, что direct(e)=false, существует единственная сущность-контейнер c ∈ C такая, что direct(c)=true, e<c и для любой сущности e’ ∈ E такой, что e’<c, верно direct(e’)=false и выполняется $(e',a_r) \in PA(r)$ тогда и только тогда , когда $(c, a_r) \in PA(r)$;
- в сущностях-контейнерах с косвенной сеткой нельзя создавать сущности с прямой меткой или “жесткие” ссылки на них. В сущности-контейнере с прямой меткой могут создаваться сущности или “жесткие” ссылки на них с метками одного вида. “Жесткая” ссылка на сущность-объект с косвенной меткой может создаваться в сущности-контейнере, подчиненной в иерархии той же самой единственной сущности-контейнеру, которой подчинена в иерархии сама сущность-объект.

Условие 9 (индивидуальная административная и индивидуальная роли учетной записи пользователя, общая роль):

- для каждой учётной записи пользователя *u ∈ U* задаётся индивидуальная административная роль *u_admin* ∈ AR, не находящаяся в иерархии других ролей, при этом задано множество всех индивидуальных административных ролей *U_ADMIN = {u_admin. u ∈ U}.* Множество других ролей учётной записи пользователя *u* задаётся с использованием административных прав доступа этой административной роли, определяемых функцией *APA.* На траекториях функционирования системы у этой административной роли не изменяется имя;

- для каждой учётной записи пользователя задаётся индивидуальная роль, не находящаяся в иерархии других ролей, административными правами доступа на чтение, запись и выполнение к которой обладает её индивидуальная административная роль: для каждой учётной записи пользователя $u \in U$ задаётся роль u_c ∈ R такая, что (u_c, a_r) ∈ $APA(u\_admin)$, где $\alpha_r \in \{read_r, write_r, execute_r\}$, при этом задано множество всех индивидуальных ролей $U.ROLES = \{u\_c: u \in U\}$;
- задаётся общая роль $common.role \in R$, не находящаяся в иерархии других ролей, административными правами доступа на чтение, запись и выполнение к которой обладают все индивидуальные административные роли всех учётных записей пользователей, и задано множество $COMMON\text{-}ROLES = \{common.role\}$: для каждой учётной записи пользователя $u \in U$ выполняется $(common\text{-}role, a_r) \in APA(u\text{-}admin)$, где $\alpha_r \in \{read_r, write_r, execute_r\}$;
- административные роли *roles-admin_role* и *admin.roles-Ci.d- min.role* не используются для нарушения правил назначения административных прав доступа к индивидуальным административным ролям, индивидуальным ролям учётных записей пользователей и общим ролям.

Условие 10 (администрирование функции необходимого обладания запрещающей ролью, получение её значения, доступы и права доступа запрещающих ролей):

- для специальных административных ролей из множества *SAR* не задаются запрещающие роли: для $ar \in SAR$ верно $constraint_{NR}(ar) = \emptyset$
- Для изменения значения функции необходимого обладания запрещающей ролью $constraint_{NR}$ субъект-сессия должна иметь административный доступ на чтение к административной роли *negative_roles_admin_role,* а также в зависимости от того, является параметр функции ролью или административной ролью, субъект-сессия должна иметь административный доступ на чтение к административной роли *roles_admin_role* или *admin_roles_admin_role* соответственно;
- При добавлении для некоторой роли или административной роли запрещающих ролей ни одна субъект-сессия не должна иметь к такой роли или административной роли административных доступов;
- для получения субъект-сессией запрещающих ролей, заданных с помощью функции $constraint_{NR}$ для роли или административной роли, или, наоборот, ролей или административных ролей, для которых задана запрещающая роль, субъект-сессия должна иметь административный доступ на чтение к административной роли *negative_roles_admin_role*;
- в случае, когда для некоторой роли или административной роли задано непустое множество запрещающих ролей, то административный доступ на чтение субъект-сессии к такой роли или административной роли предоставляется только при получении ею

административного доступа на чтение ко всем соответствующим запрещающим ролям: если существуют субъект-сессия s ∈ S роль или административная роль r ∈ R U AR, запрещающая роль nr ∈ NR такие, что nr ∈ $constraint_{NR}$ (r), (s, r, $read_a$) ∈ AA, то (s, nr, $read_a$) ∈ AA.

Условие 11 (доступы субъект-сессии к индивидуальным ролям и общим ролям, запрещающим ролям, соответствующим индивидуальным административным ролям, индивидуальным ролям или общим ролям):

- при создании каждой субъект-сессии она получает доступ на чтение к индивидуальной административной роли и доступ на чтение и запись к индивидуальной роли ее учетной записи пользователя и общей роли: для субъект-сессии s ∈ S выполняются условия (*s' user(s)_admin, $read_a$), (s, user(s)_c, $read_a$), (s, common_role, $read_a$), (s, user(s)_c, $write_a$), (s, common_role, $write_a$) ∈ AA;*
- при создании каждой субъект-сессии индивидуальная роль ее учетной записи пользователя получает право доступа владения к этой субъект-сессии: для субъект-сессии s ∈ S выполняется условие (s, own_r) ∈ PA(user(s)_c);
- если для индивидуальной административной роли или индивидуальной роли учетной записи пользователя заданы запрещающие роли, то индивидуальная административная роль должна обладать административными правами доступа на чтение ко всем таким запрещающим ролям, и при создании от имени этой учетной записи пользователя субъект-сессии она получает доступ на чтение ко всем соответствующим запрещающим ролям: если для учетной записи пользователя u∈U выполняется nr ∈ $constraint_{NR}$(u_admin) U $constraint_{NR}$(u_c), (nr,$read_r$) ∈ APA(u_admin); если для субъект-сессии s∈S верно nr ∈ $constraint_{NR}$(user(s)_admin)U $constraint_{NR}$(user(s)_c);
- если для общей роли заданы запрещающие роли, то индивидуальная административная роль каждой учётной записи пользователя должна обладать административными правами доступа на чтение ко всем таким запрещающим ролям, и при создании любой субъект-сессии она получает доступ на чтение ко всем соответствующим запрещающим ролям: если выполняется nr ∈ $constraint_{NR}$(common_role), то для всехучётных записей пользователей u ∈ *U* верно (nr, *$read_r$) ∈ APA(u_admin),* и для всех субъект-сессий s ∈ *S* таких, что (s, *common _role, read,.) ∈ ∈ A A,* верно (s, nr, *$read_a$) ∈ AA;*
- при удалении субъект-сессии удаляются все её административные доступы к индивидуальной, индивидуальной административной и общей ролям.

Рассмотрим особенности реализации в ОССН условий предположения 2.2.

В условии 1, в отличие от других ДП-моделей, предполагается, что субъект-сессии не могут иметь доступов друг к другу, что соответствует условиям функционирования ОССН. При реализации условия потребуется уточнить, что предоставляет право доступа владения

к субъект-сессии, например даёт ли такое право возможность её отладки или возможность выполнить от её имени какое-либо действие в системе.

Условие 2 обеспечивает потенциальную возможность получения доступа к роли вне зависимости от ее положения в иерархии ролей. Кроме того, оно не позволяет субъект-сессии, не обладающей соответствующими административными ролями, изменить иерархию ролей. Получая доступ на запись к некоторой роли (дающий возможность изменять множество ее прав доступа), такая субъективная-сессия не может удалить роли, подчиненные данной роли в иерархии, так как она помечена как “разделяемый контейнер”.

Условие 3 задает общий порядок получения доступов и назначения прав доступа ролей, запрещающих ролей или административных ролей, ролей-“владельцев” к сущностям, субъект-сессиям, ролям, запрещающим ролям или административным ролям, при этом вводятся специальные административные роли *entities_admin_role, subjects_admin_role, roles_admın_role, negatıue_roles_admın_role* и *admin_roles_admın_role,* используемые для администрирования сущностей, субъект-сессий, ролей, запрещающих ролей или административных ролей соответственно.

Условия 4 и 5 задают порядок получения доступа к сущностям, их создания, переименования или удаления, получения параметров типичный для ОС семейства *Linux,* при этом эти условия основаны на ролевом управлении доступом и использовании функции *execute_container,* при определении которой учитывается требование отсутствия даже одной текущей запрещающей роли, обладающей правом доступа на выполнение к сущности-контейнеру, через которую соответствующая субъект-сессия делает попытку получить какой-либо доступ к сущности. Кроме того, в условиях раскрывается содержание ограничений, которые накладываются правами доступа («запрещающими» правами доступа) запрещающих ролей при попытке осуществления субъект-сессиями соответствующих доступов к сущностям.

В условии 6 указываются роли, требуемые для администрирования учётных записей пользователей, что в предшествующих ДП-моделях не допускалось. Типичные для ОССН условия создания или удаления субъект-сессий заданы в условии 7, в которых учтены «запрещающие» права доступа запрещающих ролей.

Для обеспечения возможности задания в ОССН прав доступа к сущностям, находящимся в архивах или на внешних устройствах (файловая система которых часто не позволяет хранить права доступа или другие параметры механизма управления доступом, например уровни конфиденциальности или целостности, в соответствии с требованиями последующих уровней МРОСЛ ДП-модели). в условии 8 используются косвенные метки сущностей, с помощью которых указывается, что права доступа ролей, запрещающих ролей или административных ролей к сущности (в дальнейшем уровни конфиденциальности или

целостности) наследуется от сущности-контейнера, являющейся “точкой монтирования” к файловой системе архива или внешнего устройства. Так как файловая система ОССН не позволяет создавать “жесткие” ссылки на сущности между файловыми системами различных устройств, то в соответствии с условием 2 такая “точка монтирования” определяется однозначно для каждой сущности с косвенной меткой, а следовательно, однозначно задаются права доступа к ней.

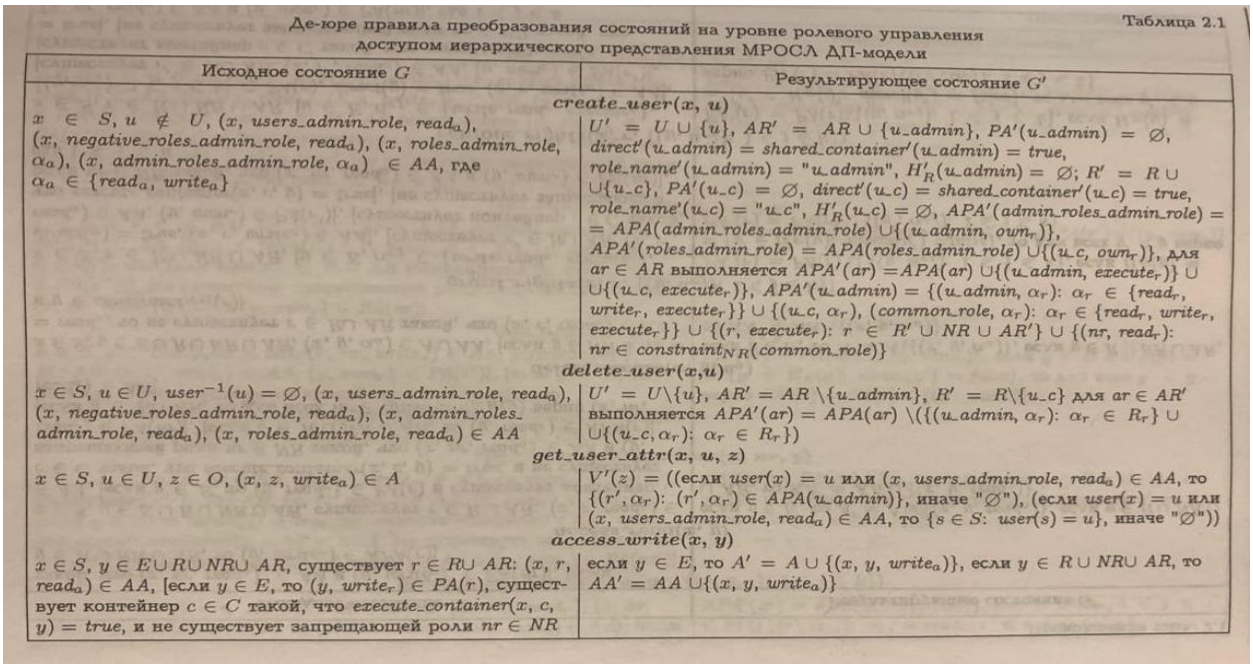
В условии 9, в отличие от моделей семейства RBAC и других ролевых ДП-моделей, вместо функций UA и AUA для задания авторизированных ролей и административных ролей учетных записей пользователей используются права доступа их индивидуальных административных ролей, задаваемых функцией APA. Такой подход позволяет реализовать ролевое управление доступом назначен прав доступа учётным записям пользователей только через роли. Также достигается большая совместимость со штатными механизмами управления доступом ОССН. Для обеспечения возможности функционирования в ОССН субъект-сессии (процессу) необходимо предоставить хотя бы одну индивидуальную роль, обладающую ила имеющую возможность получения прав доступа к сущностям (особенно при их создании), «принадлежащим» только учётной записи пользователя, от имени которой функционирует субъект-сессия. Например, такие сущности могут располагаться в «домашнем» каталоге *(/home/user}.* Кроме того, для предоставления прав доступа ролей, которые в дискреционных ОС семейства *Linux* задаются «для всех остальных» учётных записей пользователей, аналогично индивидуальным ролям учётных записей пользователей используется общая роль. На уровне ролевого управления доступом модели она единственная, на последующих уровнях модели их может быть несколько, поэтому для общих ролей задано соответствующее множество *COMMON-ROLES.*Таким образом, с использованием индивидуальных административных и индивидуальных ролей учётных записей пользователей и общей роли легко выразить традиционный для ОС семейства *Linux* подход к реализации дискреционного управления доступом.

Условие 10 определяет порядок администрирования функции необходимого обладания запрещающей ролью $constraint_{NR}$ а также получения значений этой функции. При этом указывается на отсутствие запрещающих ролей, соответствующих специальным административным ролям, так как обратное маловероятно потребуется в ОССН и может сильно затруднить её администрирование. Также описывается порядок использования функции $constraint_{NR}$, заключающийся в обязательном получении субъект-сессий запрещающей роли как текущей в случае, когда эта субъект-сессия запрещающей роли как текущей в случае, когда эта субъект-сессия получает как текущую соответствующую роль или административную роль.

Выполнение условия 11 обеспечивает возможность получения субъект-сессий заданной запрещающей роли так текущей, в случае если эта роль соответствует индивидуальной роли или индивидуальной административной роли учетной записи пользователя этой субъект-сессии. Правом доступа на чтение к такой запрещающей роли должна обладать индивидуальная административная роль учетной записи пользователя. В этом случае при создании субъект-сессии она сразу получает соответствующую запрещающую роль. Аналогично в случае, когда запрещающая роль задана для общей роли, то правом доступа на чтение к запрещающей роли должны обладать все учётные записи пользователей. Таким образом, условие обеспечивает согласованность задания значений функции необходимого обладания запрещающей ролью $constraint_{NR}$ с правами доступа индивидуальных административных ролей учётных записей пользователей. Кроме того, условие не указывает, какой административной роли должно быть дано право доступа на чтение к запрещающей роли, заданной для какой-то роли или административной роли. Это означает, что для того, чтобы получить эту роль или административную роль как текущую, субъект-сессии необходимо иметь право доступа на чтение к запрещающей роли через индивидуальную административную роль учётной записи пользователя субъект-сессии или через любую другую административную роль, которую как текущую должна предварительно получить субъект сессия. В противном случае субъект-сессия не сможет получить роль или административную роль как текущую. Также отметим, что в условии описан конкретный способ задания запрещающей роли для индивидуальной административной роли или индивидуальной роли некоторой учётной записи пользователя, делающий такую запрещающую роль текущей для всех субъект-сессий функционирующих от имени этой учётной записи пользователя. Возможны другие более «гибкие» способы применения запрещающих ролей. Например, в некоторых случаях в ОССН требуется ограничить права доступа субъект-сессии (процесса), активизированного из конкретной сущности (исполняемого файла). Тогда роли, которая обладает правом доступа на выполнение $execute_r$ к исполняемому файлу, можно с помощью функции $constraint_{NR}$ назначить соответствующую запрещающую роль. После чего родительский процесс должен сначала получить как текущую эту роль вместе с запрещающей ролью, активизировать из исполняемого файла новый процесс, который “наследует” от родительского и запрещающую роль.

Таким образом, включение в МРОСЛ ДП-модель ролевого управления доступом с запрещающими ролями в перспективе реализации его в ОССН обеспечит гибкость такого управления доступом, которая, как правило, достигается только при использовании атрибутивного управления доступом (Attribute Based Access Control- ABAC) {117}. При этом в отличии от ABAC в рамках МРОСЛ ДП-модели осуществляется строгое теоретическое

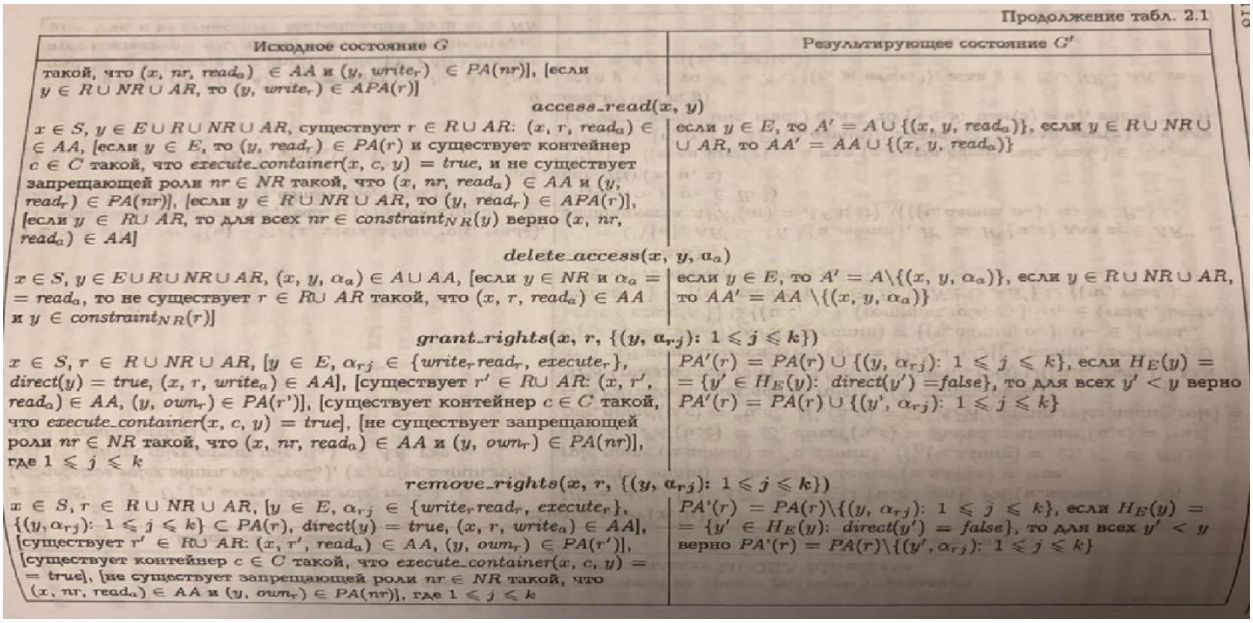
доказательство условий безопасности моделируемого механизма управления доступом ОССН.



Де-юре правила преобразования состояний на уровне ролевого управления доступом иерархического представления МРОСЛ ДП-модели

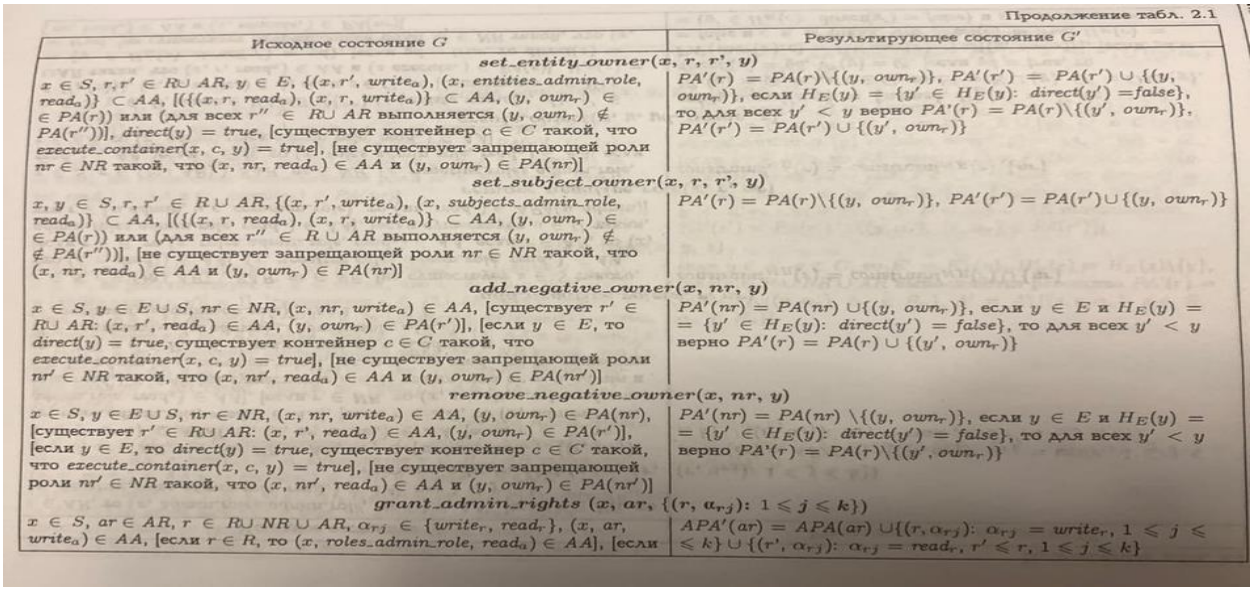
Таблица 2.1

Исходное состояние G	Результирующее состояние G'
$create\_user(x, u)$	
$x \in S, u \notin U, (x, users\_admin\_role, read_a), (x, negative\_roles\_admin\_role, read_a), (x, roles\_admin\_role, \alpha_a), (x, admin\_roles\_admin\_role, \alpha_a) \in AA$, где $\alpha_a \in \{read_a, write_a\}$	$U' = U \cup \{u\}$, $AR' = AR \cup \{u\_admin\}$, $PA'(u\_admin) = \varnothing$, $direct'(u\_admin) = shared\_container'(u\_admin) = true$, $role\_name'(u\_admin) =$ "u_admin", $H'_R(u\_admin) = \varnothing$; $R' = R \cup \cup\{u\_c\}$, $PA'(u\_c) = \varnothing$, $direct'(u\_c) = shared\_container'(u\_c) = true$, $role\_name'(u\_c) =$ "u_c", $H'_R(u\_c) = \varnothing$, $APA'(admin\_roles\_admin\_role) = = APA(admin\_roles\_admin\_role) \cup\{(u\_admin, own_r)\}$, $APA'(roles\_admin\_role) = APA(roles\_admin\_role) \cup\{(u\_c, own_r)\}$, для $ar \in AR$ выполняется $APA'(ar) = APA(ar) \cup\{(u\_admin, execute_r)\} \cup \cup\{(u\_c, execute_r)\}$, $APA'(u\_admin) = \{(u\_admin, \alpha_r): \alpha_r \in \{read_r, write_r, execute_r\}\} \cup \{(u\_c, \alpha_r), (common\_role, \alpha_r): \alpha_r \in \{read_r, write_r, execute_r\}\} \cup \{(r, execute_r): r \in R' \cup NR \cup AR'\} \cup \{(nr, read_r): nr \in constraint_{NR}(common\_role)\}$
$delete\_user(x,u)$	
$x \in S, u \in U, user^{-1}(u) = \varnothing, (x, users\_admin\_role, read_a), (x, negative\_roles\_admin\_role, read_a), (x, admin\_roles\_admin\_role, read_a), (x, roles\_admin\_role, read_a) \in AA$	$U' = U \setminus \{u\}$, $AR' = AR \setminus \{u\_admin\}$, $R' = R \setminus \{u\_c\}$ для $ar \in AR'$ выполняется $APA'(ar) = APA(ar) \setminus (\{(u\_admin, \alpha_r): \alpha_r \in R_r\} \cup \cup\{(u\_c, \alpha_r): \alpha_r \in R_r\})$
$get\_user\_attr(x, u, z)$	
$x \in S, u \in U, z \in O, (x, z, write_a) \in A$	$V'(z) =$ ((если $user(x) = u$ или $(x, users\_admin\_role, read_a) \in AA$, то $\{(r', \alpha_r): (r', \alpha_r) \in APA(u\_admin)\}$, иначе "$\varnothing$"), (если $user(x) = u$ или $(x, users\_admin\_role, read_a) \in AA$, то $\{s \in S: user(s) = u\}$, иначе "$\varnothing$"))
$access\_write(x, y)$	
$x \in S, y \in E \cup R \cup NR \cup AR$, существует $r \in R \cup AR$: $(x, r, read_a) \in AA$, [если $y \in E$, то $(y, write_r) \in PA(r)$, существует контейнер $c \in C$ такой, что $execute\_container(x, c, y) = true$, и не существует запрещающей роли $nr \in NR$	если $y \in E$, то $A' = A \cup \{(x, y, write_a)\}$, если $y \in R \cup NR \cup AR$, то $AA' = AA \cup \{(x, y, write_a)\}$



Продолжение табл. 2.1

Исходное состояние G	Результирующее состояние G'
такой, что $(x, nr, read_a) \in AA$ и $(y, write_r) \in PA(nr)$], [если $y \in R \cup NR \cup AR$, то $(y, write_r) \in APA(r)$]	
$access\_read(x, y)$	
$x \in S, y \in E \cup R \cup NR \cup AR$, существует $r \in R \cup AR$: $(x, r, read_a) \in AA$, [если $y \in E$, то $(y, read_r) \in PA(r)$ и существует контейнер $c \in C$ такой, что $execute\_container(x, c, y) = true$, и не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(y, read_r) \in PA(nr)$], [если $y \in R \cup NR \cup AR$, то $(y, read_r) \in APA(r)$], [если $y \in R \cup AR$, то для всех $nr \in constraint_{NR}(y)$ верно $(x, nr, read_a) \in AA$]	если $y \in E$, то $A' = A \cup \{(x, y, read_a)\}$, если $y \in R \cup NR \cup \cup AR$, то $AA' = AA \cup \{(x, y, read_a)\}$
$delete\_access(x, y, \alpha_a)$	
$x \in S, y \in E \cup R \cup NR \cup AR, (x, y, \alpha_a) \in A \cup AA$, [если $y \in NR$ и $\alpha_a = read_a$, то не существует $r \in R \cup AR$ такой, что $(x, r, read_a) \in AA$ и $y \in constraint_{NR}(r)$]	если $y \in E$, то $A' = A \setminus \{(x, y, \alpha_a)\}$, если $y \in R \cup NR \cup AR$, то $AA' = AA \setminus \{(x, y, \alpha_a)\}$
$grant\_rights(x, r, \{(y, \alpha_{rj}): 1 \leqslant j \leqslant k\})$	
$x \in S, r \in R \cup NR \cup AR$, [$y \in E, \alpha_{rj} \in \{write_r read_r, execute_r\}$, $direct(y) = true$, $(x, r, write_a) \in AA$], [существует $r' \in R \cup AR$: $(x, r', read_a) \in AA$, $(y, own_r) \in PA(r')$], [существует контейнер $c \in C$ такой, что $execute\_container(x, c, y) = true$], [не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(y, own_r) \in PA(nr)$], где $1 \leqslant j \leqslant k$	$PA'(r) = PA(r) \cup \{(y, \alpha_{rj}): 1 \leqslant j \leqslant k\}$, если $H_E(y) = = \{y' \in H_E(y): direct(y') = false\}$, то для всех $y' < y$ верно $PA'(r) = PA(r) \cup \{(y', \alpha_{rj}): 1 \leqslant j \leqslant k\}$
$remove\_rights(x, r, \{(y, \alpha_{rj}): 1 \leqslant j \leqslant k\})$	
$x \in S, r \in R \cup NR \cup AR$, [$y \in E, \alpha_{rj} \in \{write_r read_r, execute_r\}$, $\{(y, \alpha_{rj}): 1 \leqslant j \leqslant k\} \subset PA(r)$, $direct(y) = true$, $(x, r, write_a) \in AA$], [существует $r' \in R \cup AR$: $(x, r', read_a) \in AA$, $(y, own_r) \in PA(r')$], [существует контейнер $c \in C$ такой, что $execute\_container(x, c, y) = = true$], [не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(y, own_r) \in PA(nr)$], где $1 \leqslant j \leqslant k$	$PA'(r) = PA(r) \setminus \{(y, \alpha_{rj}): 1 \leqslant j \leqslant k\}$, если $H_E(y) = = \{y' \in H_E(y): direct(y') = false\}$, то для всех $y' < y$ верно $PA'(r) = PA(r) \setminus \{(y', \alpha_{rj}): 1 \leqslant j \leqslant k\}$



Продолжение табл. 2.1

Исходное состояние G	Результирующее состояние G'
$set\_entity\_owner(x, r, r', y)$	
$x \in S, r, r' \in R \cup AR, y \in E, \{(x, r', write_a), (x, entities\_admin\_role, read_a)\} \subset AA, [(\{(x, r, read_a), (x, r, write_a)\} \subset AA, (y, own_r) \in PA(r))$ или (для всех $r'' \in R \cup AR$ выполняется $(y, own_r) \notin PA(r''))]$, $direct(y) = true$, [существует контейнер $c \in C$ такой, что $execute\_container(x, c, y) = true$], [не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(y, own_r) \in PA(nr)$]	$PA'(r) = PA(r) \setminus \{(y, own_r)\}$, $PA'(r') = PA(r') \cup \{(y, own_r)\}$, если $H_E(y) = \{y' \in H_E(y)$: $direct(y') = false\}$, то для всех $y' < y$ верно $PA'(r) = PA(r) \setminus \{(y', own_r)\}$, $PA'(r') = PA(r') \cup \{(y', own_r)\}$
$set\_subject\_owner(x, r, r', y)$	
$x, y \in S, r, r' \in R \cup AR, \{(x, r', write_a), (x, subjects\_admin\_role, read_a)\} \subset AA$, [$\{(x, r, read_a), (x, r, write_a)\} \subset AA$, $(y, own_r) \in PA(r))$ или (для всех $r'' \in R \cup AR$ выполняется $(y, own_r) \notin PA(r''))]$, [не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(y, own_r) \in PA(nr)$]	$PA'(r) = PA(r) \setminus \{(y, own_r)\}$, $PA'(r') = PA(r') \cup \{(y, own_r)\}$
$add\_negative\_owner(x, nr, y)$	
$x \in S, y \in E \cup S, nr \in NR, (x, nr, write_a) \in AA$, [существует $r' \in R \cup AR$: $(x, r', read_a) \in AA$, $(y, own_r) \in PA(r')]$, [если $y \in E$, то $direct(y) = true$, существует контейнер $c \in C$ такой, что $execute\_container(x, c, y) = true$], [не существует запрещающей роли $nr' \in NR$ такой, что $(x, nr', read_a) \in AA$ и $(y, own_r) \in PA(nr')$]	$PA'(nr) = PA(nr) \cup \{(y, own_r)\}$, если $y \in E$ и $H_E(y) = \{y' \in H_E(y)$: $direct(y') = false\}$, то для всех $y' < y$ верно $PA'(r) = PA(r) \cup \{(y', own_r)\}$
$remove\_negative\_owner(x, nr, y)$	
$x \in S, y \in E \cup S, nr \in NR, (x, nr, write_a) \in AA, (y, own_r) \in PA(nr)$, [существует $r' \in R \cup AR$: $(x, r', read_a) \in AA$, $(y, own_r) \in PA(r')]$, [если $y \in E$, то $direct(y) = true$, существует контейнер $c \in C$ такой, что $execute\_container(x, c, y) = true$], [не существует запрещающей роли $nr' \in NR$ такой, что $(x, nr', read_a) \in AA$ и $(y, own_r) \in PA(nr')]$	$PA'(nr) = PA(nr) \setminus \{(y, own_r)\}$, если $y \in E$ и $H_E(y) = \{y' \in H_E(y)$: $direct(y') = false\}$, то для всех $y' < y$ верно $PA'(r) = PA(r) \setminus \{(y', own_r)\}$
$grant\_admin\_rights(x, ar, \{(r, \alpha_{rj})$: $1 \leqslant j \leqslant k\})$	
$x \in S, ar \in AR, r \in R \cup NR \cup AR, \alpha_{rj} \in \{write_r, read_r\}, (x, ar, write_a) \in AA$, [если $r \in R$, то $(x, roles\_admin\_role, read_a) \in AA]$, [если	$APA'(ar) = APA(ar) \cup \{(r, \alpha_{rj})$: $\alpha_{rj} = write_r, 1 \leqslant j \leqslant k\} \cup \{(r', \alpha_{rj})$: $\alpha_{rj} = read_r, r' \leqslant r, 1 \leqslant j \leqslant k\}$

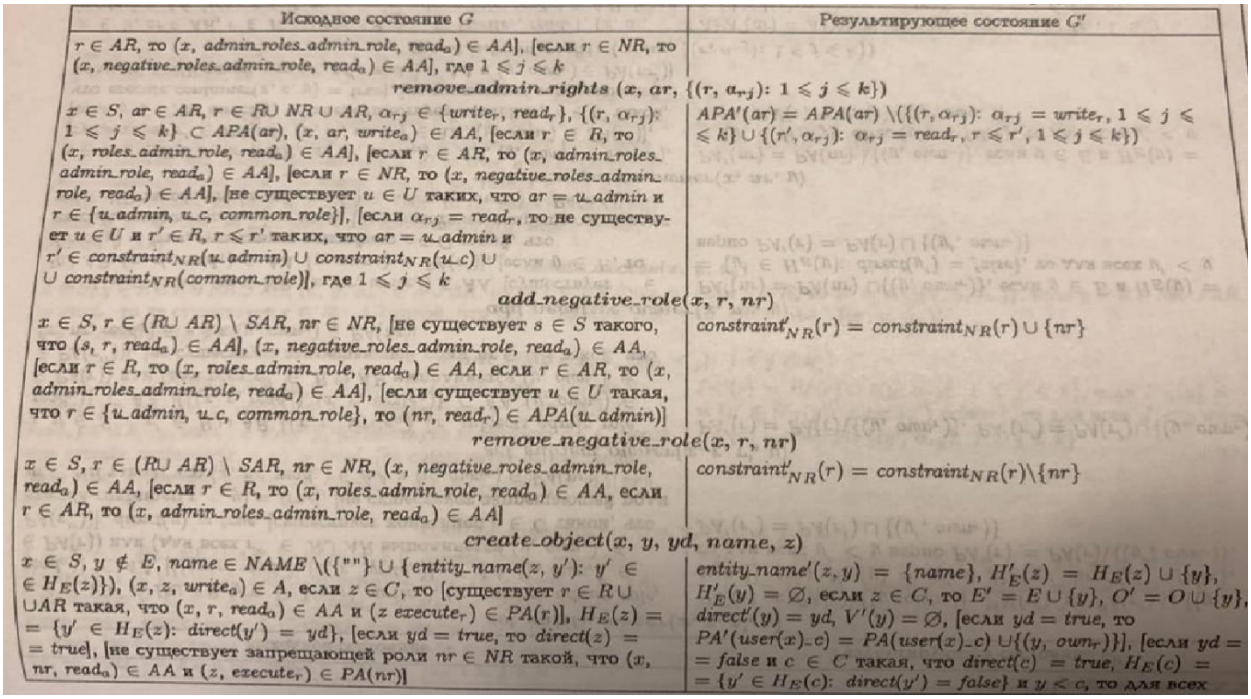
2.2.3. Де-юре правила преобразования состояний

В рамках МРОСА ДП-модели используются правила преобразования состояний из множества *OP,* которые по аналогии с моделью *Take-Grant* классифицированы на *де-юре правила* — правила, которые требуют реализации в ОССН, т. е. приводящие к «реальным» изменениям её параметров: изменению множеств прав до- F ступа ролей, получению доступов субъект-сессий к сущностям ила ролям и т.д.; и *де-факто правила* — правила, которые не требуют реализации в ОССН, так как используются в модели для отражения факта получения субъект-сессией де-факто владения субъект- сессиями или факта реализации информационного потока по памяти или по времени. В связи с этим на уровне ролевого управления доступом иерархического представления МРОСА ДП-модели задаются только де-юре правила преобразования состояний. При этом в результатах их применения не указываются неизменяющиеся элементы состояний системы.

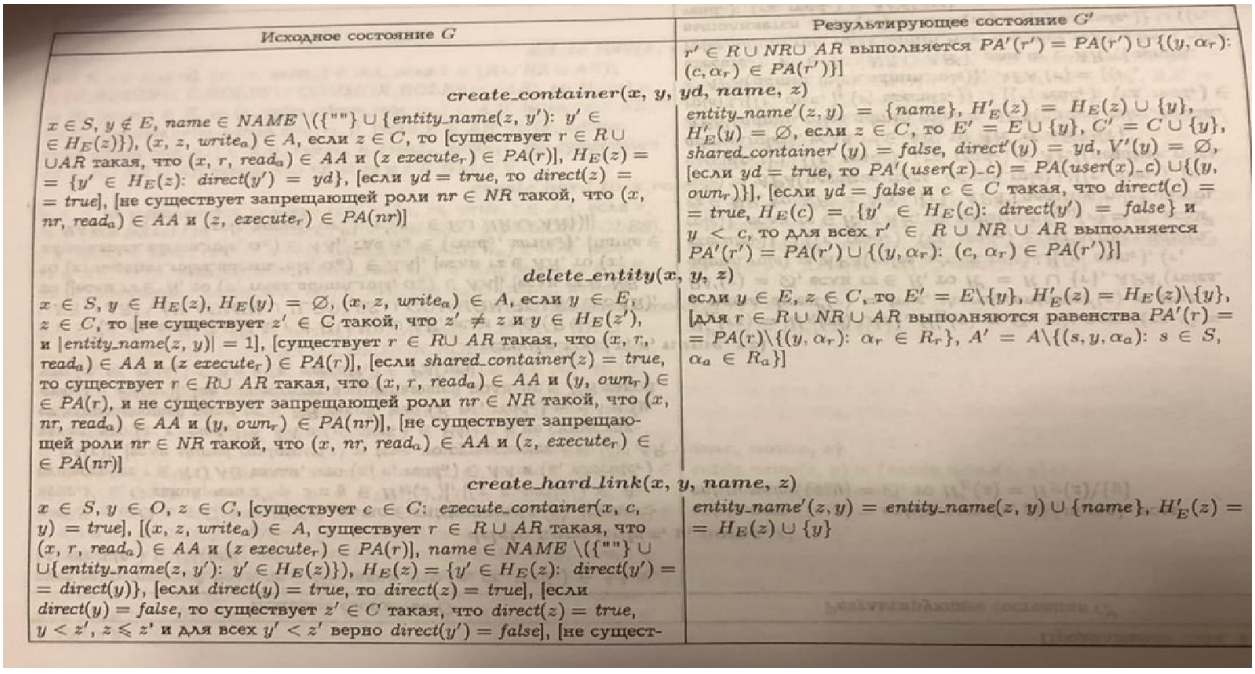
В рамках уровня ролевого управления доступом МРОСА ДП- модели заданы 35 де-юре правил преобразования состояний, условия и результаты применения которых соответствуют предположениям модели. Эти правила предназначены для формального описания (спецификации) следующих основных функций механизма управления доступом ОССН:

- создание, удаление, переименование, получение или изменение параметров учётных записей пользователей, ролей, запрещающих ролей, административных ролей, сущностей или “жестких” ссылок на них, субъект-сессий;
- получение доступов субъект-сессий к сущностям, ролям, запрещающим ролям или административным ролям;
- изменение прав доступа ролей, запрещающих ролей или административных ролей к сущностям, субъект-сессиям, ролям, запрещающим ролям или административным ролям;

- изменение иерархии сущностей, ролей, запрещающих ролей или административных ролей;



Исходное состояние G	Результирующее состояние G'
$r \in AR$, то $(x, admin\_roles\_admin\_role, read_a) \in AA$], [если $r \in NR$, то $(x, negative\_roles\_admin\_role, read_a) \in AA$], где $1 \leqslant j \leqslant k$	
remove_admin_rights $(x, ar, \{(r, \alpha_{rj}): 1 \leqslant j \leqslant k\})$	
$x \in S$, $ar \in AR$, $r \in R \cup NR \cup AR$, $\alpha_{rj} \in \{write_r, read_r\}$, $\{(r, \alpha_{rj}): 1 \leqslant j \leqslant k\} \subset APA(ar)$, $(x, ar, write_a) \in AA$, [если $r \in R$, то $(x, roles\_admin\_role, read_a) \in AA$], [если $r \in AR$, то $(x, admin\_roles\_admin\_role, read_a) \in AA$], [если $r \in NR$, то $(x, negative\_roles\_admin\_role, read_a) \in AA$], [не существует $u \in U$ таких, что $ar = u\_admin$ и $r \in \{u\_admin, u\_c, common\_role\}$], [если $\alpha_{rj} = read_r$, то не существует $u \in U$ и $r' \in R$, $r \leqslant r'$ таких, что $ar = u\_admin$ и $r' \in constraint_{NR}(u\_admin) \cup constraint_{NR}(u\_c) \cup constraint_{NR}(common\_role)$], где $1 \leqslant j \leqslant k$	$APA'(ar) = APA(ar) \setminus (\{(r, \alpha_{rj}): \alpha_{rj} = write_r, 1 \leqslant j \leqslant k\} \cup \{(r', \alpha_{rj}): \alpha_{rj} = read_r, r \leqslant r', 1 \leqslant j \leqslant k\})$
add_negative_role (x, r, nr)	
$x \in S$, $r \in (R \cup AR) \setminus SAR$, $nr \in NR$, [не существует $s \in S$ такого, что $(s, r, read_a) \in AA$], $(x, negative\_roles\_admin\_role, read_a) \in AA$, [если $r \in R$, то $(x, roles\_admin\_role, read_a) \in AA$, если $r \in AR$, то $(x, admin\_roles\_admin\_role, read_a) \in AA$], [если существует $u \in U$ такая, что $r \in \{u\_admin, u\_c, common\_role\}$, то $(nr, read_r) \in APA(u\_admin)$]	$constraint'_{NR}(r) = constraint_{NR}(r) \cup \{nr\}$
remove_negative_role (x, r, nr)	
$x \in S$, $r \in (R \cup AR) \setminus SAR$, $nr \in NR$, $(x, negative\_roles\_admin\_role, read_a) \in AA$, [если $r \in R$, то $(x, roles\_admin\_role, read_a) \in AA$, если $r \in AR$, то $(x, admin\_roles\_admin\_role, read_a) \in AA$]	$constraint'_{NR}(r) = constraint_{NR}(r) \setminus \{nr\}$
create_object $(x, y, yd, name, z)$	
$x \in S$, $y \notin E$, $name \in NAME \setminus (\{""\} \cup \{entity\_name(z, y'): y' \in H_E(z)\})$, $(x, z, write_a) \in A$, если $z \in C$, то [существует $r \in R \cup AR$ такая, что $(x, r, read_a) \in AA$ и $(z\ execute_r) \in PA(r)$], $H_E(z) = \{y' \in H_E(z): direct(y') = yd\}$, [если $yd = true$, то $direct(z) = true$], [не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(z, execute_r) \in PA(nr)$]	$entity\_name'(z, y) = \{name\}$, $H'_E(z) = H_E(z) \cup \{y\}$, $H'_E(y) = \varnothing$, если $z \in C$, то $E' = E \cup \{y\}$, $O' = O \cup \{y\}$, $direct'(y) = yd$, $V'(y) = \varnothing$, [если $yd = true$, то $PA'(user(x)\_c) = PA(user(x)\_c) \cup \{(y, own_r)\}$], [если $yd = false$ и $c \in C$ такая, что $direct(c) = true$, $H_E(c) = \{y' \in H_E(c): direct(y') = false\}$ и $y < c$, то для всех



Исходное состояние G	Результирующее состояние G'
	$r' \in R \cup NR \cup AR$ выполняется $PA'(r') = PA(r') \cup \{(y, \alpha_r): (c, \alpha_r) \in PA(r')\}$]
create_container $(x, y, yd, name, z)$	
$x \in S$, $y \notin E$, $name \in NAME \setminus (\{""\} \cup \{entity\_name(z, y'): y' \in H_E(z)\})$, $(x, z, write_a) \in A$, если $z \in C$, то [существует $r \in R \cup AR$ такая, что $(x, r, read_a) \in AA$ и $(z\ execute_r) \in PA(r)$], $H_E(z) = \{y' \in H_E(z): direct(y') = yd\}$, [если $yd = true$, то $direct(z) = true$], [не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(z, execute_r) \in PA(nr)$]	$entity\_name'(z, y) = \{name\}$, $H'_E(z) = H_E(z) \cup \{y\}$, $H'_E(y) = \varnothing$, если $z \in C$, то $E' = E \cup \{y\}$, $C' = C \cup \{y\}$, $shared\_container'(y) = false$, $direct'(y) = yd$, $V'(y) = \varnothing$, [если $yd = true$, то $PA'(user(x)\_c) = PA(user(x)\_c) \cup \{(y, own_r)\}$], [если $yd = false$ и $c \in C$ такая, что $direct(c) = true$, $H_E(c) = \{y' \in H_E(c): direct(y') = false\}$ и $y < c$, то для всех $r' \in R \cup NR \cup AR$ выполняется $PA'(r') = PA(r') \cup \{(y, \alpha_r): (c, \alpha_r) \in PA(r')\}$]
delete_entity (x, y, z)	
$x \in S$, $y \in H_E(z)$, $H_E(y) = \varnothing$, $(x, z, write_a) \in A$, если $y \in E$, $z \in C$, то [не существует $z' \in C$ такой, что $z' \neq z$ и $y \in H_E(z')$, и $\lvert entity\_name(z, y)\rvert = 1$], [существует $r \in R \cup AR$ такая, что $(x, r, read_a) \in AA$ и $(z\ execute_r) \in PA(r)$], [если $shared\_container(z) = true$, то существует $r \in R \cup AR$ такая, что $(x, r, read_a) \in AA$ и $(y, own_r) \in PA(r)$, и не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(y, own_r) \in PA(nr)$], [не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(z, execute_r) \in PA(nr)$]	если $y \in E$, $z \in C$, то $E' = E \setminus \{y\}$, $H'_E(z) = H_E(z) \setminus \{y\}$, [для $r \in R \cup NR \cup AR$ выполняются равенства $PA'(r) = PA(r) \setminus \{(y, \alpha_r): \alpha_r \in R_r\}$, $A' = A \setminus \{(s, y, \alpha_a): s \in S, \alpha_a \in R_a\}$]
create_hard_link $(x, y, name, z)$	
$x \in S$, $y \in O$, $z \in C$, [существует $c \in C$: $execute\_container(x, c, y) = true$], [$(x, z, write_a) \in A$, существует $r \in R \cup AR$ такая, что $(x, r, read_a) \in AA$ и $(z\ execute_r) \in PA(r)$], $name \in NAME \setminus (\{""\} \cup \{entity\_name(z, y'): y' \in H_E(z)\})$, $H_E(z) = \{y' \in H_E(z): direct(y') = direct(y)\}$, [если $direct(y) = true$, то $direct(z) = true$], [если $direct(y) = false$, то существует $z' \in C$ такая, что $direct(z) = true$, $y < z'$, $z \leqslant z'$ и для всех $y' < z'$ верно $direct(y') = false$], [не сущест-	$entity\_name'(z, y) = entity\_name(z, y) \cup \{name\}$, $H'_E(z) = H_E(z) \cup \{y\}$

Продолжение табл. 2.1

Исходное состояние G	Результирующее состояние G'
вует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(z, execute_r) \in PA(nr)$]	
delete_hard_link$(x, y, name, z)$	
$x \in S$, $y \in O$, $z \in C$, $y \in H_E(z)$, $name \in entity\_name(z, y)$, [существует $z' \in C$ такой, что $z' \neq z$ и $y \in H_E(z')$], $[(x, z, write_a) \in A$, существует $r \in R \cup AR$ такая, что $(x, r, read_a) \in AA$ и $(z, execute_r) \in PA(r)]$, [если $shared\_container(z) = true$, то существует $r \in R \cup AR$ такая, что $(x, r, read_a) \in AA$ и $(y, own_r) \in PA(r)$, и не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(y, own_r) \in PA(nr)]$, [не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(z, execute_r) \in PA(nr)]$	$entity\_name'(z, y) = entity\_name(z, y) \setminus \{name\}$, если $entity\_name'(z, y) = \varnothing$, то $H'_E(z) = H_E(z) \setminus \{y\}$
create_role$(x, r, name, rz)$	
$x \in S$, $(x, rz, write_a) \in AA$, $r \notin R \cup NR \cup AR$, если $rz \in (R \cup UNR \cup AR) \setminus (U\_ADMIN \cup U\_ROLES \cup COMMON\_ROLES \cup SAR)$, то [[если $rz \in R$, то $(x, roles\_admin\_role, \alpha_a) \in AA$], [если $rz \in NR$, то $(x, negative\_roles\_admin\_role, \alpha_a) \in AA$], [если $rz \in AR$, то $(x, admin\_roles\_admin\_role, \alpha_a) \in AA$], где $\alpha_a \in \{read_a, write_a\}$, [$name \in NAME \setminus (\{""\} \cup \{role\_name(r', r''): r', r'' \in R \cup NR \cup AR\})$]]	$H'_R(rz) = H_R(rz) \cup \{r\}$, $H'_R(r) = \varnothing$, $role\_name'(rz, r) = name$, $direct'(r) = true$, $shared\_container'(r) = true$, $PA'(r) = \varnothing$, если $rz \in R$, то $R' = R \cup \{r\}$, $APA'(roles\_admin\_role) = APA(roles\_admin\_role) \cup \{(r, own_r), (r, execute_r)\} \cup \{(r, read_r): (rz, read_r) \in APA(roles\_admin\_role)\}$, если $rz \in R \cup AR$, то $constraint'_{NR}(r) = \varnothing$, если $rz \in NR$, то $NR' = NR \cup \{r\}$, $APA'(negative\_roles\_admin\_role) = APA(negative\_roles\_ admin\_role) \cup \{(r, own_r), (r, execute_r)\} \cup \{(r, read_r): (rz, read_r) \in APA(negative\_roles\_admin\_role)\}$, если $rz \in AR$, то $AR' = AR \cup \{r\}$, $APA'(admin\_roles\_admin\_role) = APA(admin\_roles\_admin\_role) \cup \{(r, own_r), (r, execute_r)\} \cup \{(r, read_r): (rz, read_r) \in APA(admin\_roles\_admin\_role)\}$, $APA'(r) = \{(r', execute_r): r' \in R \cup NR \cup AR'\}$, для $ar \in AR \setminus \{admin\_roles\_admin\_role\ roles\_admin\_role, negative\_roles\_admin\_role\}$ выполняется $APA'(ar) = APA(ar) \cup \{(r, execute_r)\} \cup \{(r, read_r): (rz, read_r) \in APA(ar)\}$

Продолжение табл. 2.1

Исходное состояние G	Результирующее состояние G'
delete_role(x, r, rz)	
$x \in S$, $r \in H_R(rz)$, $(x, rz, write_a) \in AA$, если $r \in (R \cup NR \cup AR) \setminus (U\_ADMIN \cup U\_ROLES \cup COMMON\_ROLES \cup SAR)$, то $H_R(r) = \varnothing$, [если $r \in R$, то $(x, roles\_admin\_role, \alpha_a) \in AA$], [если $rz \in NR$, то $(x, negative\_roles\_admin\_role, \alpha_a)$], [если $r \in AR$, то $(x, admin\_roles\_admin\_role, \alpha_a) \in AA$], где $\alpha_a \in \{read_a, write_a\}$, [не существует $rz' \in R \cup NR \cup AR$ такой, что $rz' \neq rz$ и $r \in H_R(rz')$]	$H'_R(rz) = H_R(rz) \setminus \{r\}$, для $ar \in AR'$ выполняются равенства $APA'(ar) = APA(ar) \setminus \{(r, \alpha_r): \alpha_r \in R_r\}$, $AA' = AA \setminus \{(s, r, \alpha_a): s \in S, \alpha_a \in R_a\}$, если $r \in R' \cup UNR' \cup AR'$, то $R' = R \setminus \{r\}$, $AR' = AR \setminus \{r\}$, $NR' = NR \setminus \{r\}$, для всех $r' \in R' \cup AR'$ верно $constraint'_{NR}(r') = constraint'_{NR}(r') \setminus \{r\}$
create_hard_link_role(x, r, rz)	
$x \in S$, не выполняется условие $rz \leqslant r$, $(x, rz, write_a) \in AA$, если $r, rz \in (R \cup NR \cup AR) \setminus (U\_ADMIN \cup U\_ROLES \cup COMMON\_ROLES)$, $r \notin SAR$, то [если $rz \in R$, то $(x, roles\_admin\_role, \alpha_a) \in AA$], [если $rz \in NR$, то $(x, negative\_roles\_admin\_role, \alpha_a) \in AA$], [если $rz \in AR$, то $(x, admin\_roles\_admin\_role, \alpha_a) \in AA$], где $\alpha_a \in \{read_a, write_a\}$	$H'_R(rz) = H_R(rz) \cup \{r\}$, если $r, rz \in R \cup NR \cup AR$, то $role\_name'(rz, r) = role\_name(rz', r)$ и $r \in H_R(rz')$, для $ar \in AR$ таких, что $(rz, read_r) \in APA(ar)$, выполняется $APA'(ar) = APA(ar) \cup \{(r', read_r): r' \leqslant r\}$
delete_hard_link_role(x, r, rz)	
$x \in S$, $r \in H_R(rz)$, $(x, rz, write_a) \in AA$, если $r \in (R \cup NR \cup AR) \setminus (U\_ADMIN \cup U\_ROLES \cup COMMON\_ROLES \cup SAR)$, то [если $r \in R$, то $(x, roles\_admin\_role, \alpha_a) \in AA$], [если $rz \in NR$, то $(x, negative\_roles\_admin\_role, \alpha_a) \in AA$], [если $r \in AR$, то $(x, admin\_roles\_admin\_role, \alpha_a) \in AA$], где $\alpha_a \in \{read_a, write_a\}$, [существует $rz' \in R \cup NR \cup AR$ такая, что $rz' \neq rz$ и $r \in H_R(rz')$]	$H'_R(rz) = H_R(rz) \setminus \{r\}$
rename_entity$(x, y, old\_name, name, z)$	
$x \in S$, $y \in H_E(z)$, $old\_name \in entity\_name(z, y)$, $name \in NAME \setminus (\{""\} \cup \{entity\_name(z, y'): y' \in H_E(z)\})$, $(x, z, write_a) \in A$, если $y \in E$, $z \in C$, то [существует $r \in R \cup AR$ такая, что $(x, r, read_a) \in AA$ и $(z, execute_r) \in PA(r)$], [если $shared\_container(z) = true$, то существует $r \in R \cup AR$ такая, что $(x, r, read_a) \in AA$ и $(y, own_r) \in PA(r)$, и не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(y, own_r) \in PA(nr)$], [не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(z, execute_r) \in PA(nr)$]	$entity\_name(z, y) = (entity\_name(z, y) \cup \cup \{name\}) \setminus \{old\_name\}$

Продолжение табл. 2.1

Исходное состояние G	Результирующее состояние G'
rename_role$(x, ry, name)$	
$x \in S$, если $ry \in (R \cup NR \cup AR) \setminus (U\_ADMIN \cup U\_ROLES \cup COMMON\_ROLES \cup SAR)$, то $name \in NAME \setminus (\{""\} \cup \{role\_name(r', r''): r', r'' \in R \cup NR \cup AR\})$, [если $ry \in R$, то $(x, roles\_admin\_role, read_a) \in AA$], [если $ry \in NR$, то $(x, negative\_roles\_ admin\_role, read_a) \in AA$], [если $ry \in AR$, то $(x, admin\_roles\_admin\_role, read_a) \in AA$], [для $rz \in R \cup NR \cup AR$: $ry \in H_R(rz)$, выполняется $(x, rz, write_a) \in AA$]	для $rz \in R \cup NR \cup AR$ таких, что $ry \in H_R(rz)$, выполняется $role\_name'(rz, ry) = name$
read_container(x, y, z)	
$x \in S$, $z \in O$, $(x, z, write_a) \in A$, если $y \in C \cup R \cup NR \cup AR$, то существует $r \in R \cup AR$: $(x, r, read_a) \in AA$, [если $y \in C$, то $[(y, read_r) \in PA(r)$, существует $r' \in R \cup AR$: $(x, r', read_a) \in AA$, $(y, execute_r) \in PA(r')$, и существует контейнер $c \in C$: $execute\_container(x, c, y) = true]$, и [не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и либо $(y, read_r) \in PA(nr)$, либо $(y, execute_r) \in PA(nr)]]$, [если $y \in R \cup NR \cup AR$, то $(y, read_r) \in APA(r)$]	если $y \in C$, то $V'(z) = \{entity\_name(y, e): e \in H_E(y)\}$, если $y \in R \cup NR \cup AR$, то $V'(z) = \{role\_name(y, r): r \in H_R(y)\}$
get_entity_attr(x, y, z)	
$x \in S$, $y \in E$, $z \in O$, [существует контейнер $c \in C$ такой, что $execute\_container(x, c, y) = true]$, $(x, z, write_a) \in A$	$V'(z) = (direct(y)$, (если $y \in C$, то $shared\_container(y)$, иначе "$false$"), (если $y \in O$, то $|\{e \in C: y \in H_E(e)\}|$, иначе "1"), (если [существует $r \in R \cup AR$: $(x, r, read_a) \in AA$, $(y, own_r) \in PA(r)$, и не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(y, own_r) \in PA(nr)]$ или $[(x, entities\_admin\_role, read_a) \in AA]$, то $\{(r', \alpha_r): r' \in R \cup NR \cup AR, (y, \alpha_r) \in PA(r')\}$, иначе "$\varnothing$"))
get_subject_attr(x, y, z)	
$x \in S$, $y \in S$, $z \in O$, $(x, z, write_a) \in A$	$V'(z) = (user(s)$, (если [существует $r \in R \cup AR$: $(x, r, read_a) \in AA$, $(y, own_r) \in PA(r)$, и не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(y, own_r) \in PA(nr)]$ или $[(x, subjects\_admin\_role, read_a) \in AA]$, то $\{(r', own_r): r' \in R \cup NR \cup AR, (y, own_r) \in PA(r')\}$,

Продолжение табл. 2.1

Исходное состояние G	Результирующее состояние G'
	иначе "$\varnothing$"), (если [существует $r \in R \cup AR$: $(x, r, read_a) \in AA$, $(y, own_r) \in PA(r)$, и не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(y, own_r) \in PA(nr)$] или [$(x, subjects\_admin\_role, read_a) \in AA$], то $\{(r', \alpha_a)$: $(yr', \alpha_a) \in AA\}$, иначе "$\varnothing$"))
get_role_attr(x, y, z)	
$x \in S$, $y \in R \cup NR \cup AR$, $z \in O$, $(x, z, write_a) \in A$	$V'(z) = (direct(y), shared\_container(y)$, (если [$y \in R$ и $(x, roles\_admin\_role, read_a) \in AA$], или [$y \in AR$ и $(x, admin\_roles\_admin\_role, read_a) \in AA$], или [$y \in NR$ и $(x, negative\_roles\_admin\_role, read_a) \in AA$], то $\|\{r \in R \cup NR \cup AR$: $y \in H_R(r)\}\|$, иначе "1"), (если [$y \in R$ и $(x, roles\_admin\_role, read_a) \in AA$] или [$y \in AR$ и $(x, admin\_roles\_admin\_role, read_a) \in AA$], или [$y \in NR$ и $(x, negative\_roles\_admin\_role, read_a) \in AA$], то $\{(r, \alpha_r)$: $r \in AR$, $(y, \alpha_r) \in APA(r)\}$, иначе "$\varnothing$"), (если $(x, negative\_roles\_admin\_role, read_a) \in AA$, то если $y \in NR$, то $\{r \in R \cup AR$: $y \in constraint_{NR}(r)\}$, иначе если $y \in R \cup AR$, то $constraint_{NR}(y)$, иначе "$\varnothing$"))
set_container_attr(x, y, t)	
$x \in S$, $y \in C$, $t \in \{true, false\}$, [[либо существует $r \in R \cup AR$: $(x, r, read_a) \in AA$, $(y, own_r) \in PA(r)$, и не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(y, own_r) \in PA(nr)$], либо $(x, entities\_admin\_role, read_a) \in AA$], [существует контейнер $c \in C$ такой, что $execute\_container(x, c, y) = true$]	$shared\_container'(y) = t$

Окончание табл. 2.1

Исходное состояние G	Результирующее состояние G'
create_first_subject(x, u, y, z)	
$x \in S$, $u \in U$, $y \in E$, $z \notin S$, [существует $r \in R \cup AR$ такая, что $(x, r, read_a) \in AA$ и $(y, execute_r) \in PA(r)$, и не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(y, execute_r) \in PA(nr)$], [существует контейнер $c \in C$ такой, что $execute\_container(x, c, y) = true$], [для всех $nr \in constraint_{NR}(u\_admin) \cup constraint_{NR}(u\_c) \cup constraint_{NR}(common\_role)$ верно $(nr, read_r) \in APA(u\_admin)$]	$S' = S \cup \{z\}$, $user'(z) = u$, $H'_S(z) = \varnothing$, $PA'(u\_c) = PA(u\_c) \cup \{(z, own_r)\}$, $AA' = AA \cup \{(z, u\_admin, read_a), (z, u\_c, write_a), (z, common\_role, write_a), (z, u\_c, read_a), (z, common\_role, read_a)\} \cup \cup \{(z, nr, read_a)$: $nr \in constraint_{NR}(u\_admin) \cup \cup constraint_{NR}(u\_c) \cup constraint_{NR}(common\_role)\}$
create_subject(x, y, z)	
$x \in S$, $y \in E$, $z \notin S$, [существует $r \in R \cup AR$ такая, что $(x, r, read_a) \in AA$ и $(y, execute_r) \in PA(r)$, и не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(y, execute_r) \in PA(nr)$], [существует контейнер $c \in C$ такой, что $execute\_container(x, c, y) = true$], [для всех $nr \in constraint_{NR}(user(x)\_admin) \cup constraint_{NR}(user(x)\_c) \cup constraint_{NR}(common\_role)$ верно $(nr, read_r) \in APA(user(x)\_admin)$]	$S' = S \cup \{z\}$, $user'(z) = user(x)$, $H'_S(x) = H_S(x) \cup \{z\}$, $H'_S(z) = \varnothing$, $PA'(user(x)\_c) = PA(user(x)\_c) \cup \{(z, own_r)\}$, $AA' = AA \cup \{(z, user(x)\_admin, read_a), (z, user(x)\_c, write_a), (z, common\_role, write_a), (z, user(x)\_c, read_a), (z, common\_role, read_a)\} \cup \{(z, nr, read_a)$: $nr \in constraint_{NR}(user(x)\_admin) \cup \cup constraint_{NR}(user(x)\_c) \cup constraint_{NR}(common\_role)\}$
delete_subject(x, z)	
$x, z \in S$, $H_S(z) = \varnothing$, [существует $r \in R \cup AR$: $(x, r, read_a) \in AA$, $(z, own_r) \in PA(r)$, и не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(z, own_r) \in PA(nr)$]	$S' = S \setminus \{z\}$, $A' = A \setminus \{(z, e, \alpha_a)$: $e \in E$, $\alpha_a \in R_a\}$, для $r \in R \cup NR \cup AR$ выполняется $PA'(r) = PA(r) \setminus \{(z, own_r)\}$, для $z' \in S$ такой, что $z \in H_S(z')$, справедливо равенство $H'_S(z') = H_S(z') \setminus \{z\}$, $AA' = AA \setminus \{(z, r, \alpha_a)$: $r \in R \cup NR \cup AR$, $\alpha_a \in R_a\}$

- управление множествами запрещающих ролей для ролей и административных ролей.

Приведённые правила в основном соответствуют применяемым *ОС* семейства *Linux* подходам к реализации механизма дискреционного управления доступом, некоторые параметры которого выражены с помощью ролей. Рассмотрим подробнее условия и результаты применения де-юре правил преобразования состояний.

Де-юре правила вида *create_user(x, u), delete_user(x, u)* и *get_user_attr(x, u, г)* позволяют субъект-сессии *x* создать, удалить или получить параметры учётной записи пользователя *u.* Во всех случаях, кроме получения параметров, требуется наличие у субъект-сессии *x* доступов на чтение к специальным административным ролям *users_admin_role* и *negative_roles_admin_role.* При этом в последующем состоянии иерархия ролей модифицируется (создаются или удаляются индивидуальная роль и индивидуальная административная роль, назначаются или удаляются административные права доступа к ним) в соответствии с условиями предположения 2.2. Для создания или удаления требуется наличие у субъект-сессии *x* доступов на чтение, а в случае создания и на запись, к

специальным административным ролям *roles_admin_role* и *admin_roles_admin_ role,* так как именно эти роли получают права доступа владения к создаваемым в соответствии с предположением 2.2 при применении правил индивидуальным административным ролям и индивидуальным ролям учётной записи пользователя *u.* Также при создании индивидуальной административной роли новой учётной записи пользователя даются административные права доступа на чтение ко всем запрещающим ролям, заданным для общей роли. В случае удаления учётной записи пользователя требуется, чтобы в этот момент времени от её имени в системе не функционировала ни одна субъект-сессия. При получении параметров при условии, что *x* Функционирует от имени *u* или обладает текущим административным доступом на чтение к роли *users_admi_.role,* в сущность z (к которой субъект-сессия *x* должна иметь доступ на запись) записываются данные о ролях и правах доступа к ним, которыми обладает индивидуальная административная роль *u,* и записываются данные о субъект-сессиях (в ОССН идентификаторы субъект-сессий), функционирующих от имени и (в этом случае полные данные об учётной записи пользователя предоставляются либо субъект-сессии, функционирующей от её имени, либо субъект-сессии, которая может администрировать эту учётную запись).

Де-юре правила вида *access_read(x*, y) и *access_write(χ, y)* позволяют субъект-сессии x получить к сущности, роли, запрещающей роли или административной роли y соответствующий доступ или административный доступ. Для этого при получении административного доступа требуется наличие у x текущей административной роли r, содержащей соответствующее административное право доступа к y, а при получении доступа наличие текущей роли или административной роли r, содержащей соответствующее административное право доступа к y, а при получении доступа наличие текущей роли или административной роли r, содержащей соответствующее право доступа к сущности y, и отсутствие у x текущей запрещающей роли обладающей этим правом доступа к y. При этом требуется, чтобы доступ к y был предоставлен с учетом прав доступа текущих ролей, запрещающих ролей или административных ролей субъект-сессии x к сущности-контейнерам или ролям-контейнерам, содержащим y. Кроме того, если используется де-юре правило вида *access_read(χ , y) для* получения субъект-сессией *x* административного доступа на чтение к роли или административной роли *y,* то требуется наличие у субъект-сессии текущих доступов на чтение ко всем запрещающим ролям, заданным функцией $constraints_{NR}$ для *y.* Де-юре правило *delete_access{x, y,* a_a*)* позволяет субъект-сессии *x,* обладающей доступом a_a к сущности или административным доступом к роли, запрещающей роли или административной роли *y,* удалить этот доступ, при этом не допускается удаление у *x* административного доступа на чтение к запрещающей роли *y,* когда эта субъект-сессия

имеет текущую роль или административную роль, для которой запрещающей является роль *y.*

Де-юре правила вида *grant_rights(x, r,* {(y, a_{rj})·. 1 ≤ *j* ≤ k}) и *rernouex_rights(x, r,* {(y, αŋ): 1 ≤ *j* ≤ k}) позволяют субъект-сессии *x* добавить или удалить, соответственно, права доступа к сущности *y* из множества прав доступа (за исключением права доступа владения) роли или административной роли г. Возможность с использованием правил одновременного изменения нескольких прав доступа к сущностям с косвенной меткой осуществляется одновременно с изменением прав доступа к единственной существующей по условию 8 предположения 2.2 соответствующей сущности-контейнеру. Для применения правил необходимо наличие у субъекта-сессии x доступа на запись к роли r и наличие текущей роли, обладающей правом доступа владения к сущности y, а также отсутствие у x текущей запрещающей роли, обладающей правом доступа владения к y. При этом требуется, чтобы x могла получить доступ к сущности y с учетом прав доступа к сущностям-контейнерам, содержащим y.

Де-юре правила вида set_entity_owner(x,r,r',y) и set_subject_owner(x,r,r',y) позволяют субъект-сессии x либо изменить, либо задать единственную роль-“владелец” (имеющую право доступа владения) к сущности или субъект-сессии y соответственно, с роли или административной роли r на роль или административную роль r'. Для этого субъект-сессии *x* необходимо иметь административные доступы на чтение и запись к r и на запись к *r'* (чтобы иметь возможность менять права доступа данных ролей), а также иметь административный доступ на чтение соответственно, либо к административной роли *entities_admin_role,* либо к *subjects_admin_role.* При этом у субъект-сессии *x* должна отсутствовать текущая запрещающая роль, обладающая правом доступа владения к y. Непосредственно изменять роль-«владельца» разрешено только к сущностям с прямой меткой. Изменение роли-«владельца» к сущности косвенной меткой осуществляется одновременно с изменением роли-«владельца» к единственной существующей по условию 8 предположения 2.2 соответствующей сущности-контейнеру. Когда изменяется роль-«владелец» сущности *y* требуется, чтобы x могла получить доступ к сущности *y c* учётом прав доступа к сущностям-контейнерам, содержащим *y.*

Де-юре правила вида add_negatiue_owner(x, nr,y), remove_negative_owner(x, nr, y) являются аналогичными правилам вида grant_тights(x, г, {(y, a_{rj})' 1 ≤ j ≤ k}), remoue_nights(x, г, {(y, α_{rj}): 1 ≤j≤ k}), set_entıty_owner(x, г, r', y) и set_subject_owner(x, г, r', y). Они позволяют субъект-сессии x, обладающей ролью или административной ролью, имеющей право доступа владения к сущности или субъект-сессии y, а также не обладающей запрещающей ролью, имеющей это право доступа к y, добавить или удалить у запрещающей роли nr (к которой субъект-сессия x должна иметь административный доступ на запись) право доступа владения

к y. При этом запрещающая роль не становится ролью-«владельцем» или ролью ''антивладельцем'' сущности или субъект-сессии y, допускается, что запрещающих ролей, обладающих правом доступа владения к сущности или субъект-сессии может быть несколько. С этим связано неиспользование для этого правила вида *set_entity_owner(x,r,r',y) и set_subject_owner(x,r,r',y).*

Де-юре правила вида grant_admin_rights(x, ar, {(r, a_{rj}): 1≤ j ≤ k}) и remove_admin_rights(x,ar,{(r, a_{rj}): 1≤ j ≤ k}) позволяют субъект-сессии x добавить или удалить соответственно, права доступа на чтение или запись к роли, запрещающей роли или административной роли r из множества прав доступа административной роли ar, к которой x должна иметь административный доступ на запись. Для изменения прав доступа к роли r требуется наличие у x текущей административной роли roles_admin_role, если r является у x текущей административной роли roles_admin_role, а если административной ролью, то административной роли negative_roles_admin_role. При удалении административных прав доступа не должны нарушаться заданные предположении 2.2 требования к правам доступа индивидуальных административных ролей, индивидуальных ролей учетных записей пользователей и общей роли, а также если удаляется право доступа на чтение к запрещающей роли, то она или какая-либо подчиненная ей роль не должны быть заданы для индивидуальной административной роли, индивидуальной роли некоторой учетной записи пользователя или общей роли, когда административная роль ar является индивидуальной ролью этой учетной записи пользователя.

Де-юре правила вида add_negative_role(x,r,nr) и remove_negative_role(x,r,nr) позволяют субъект-сессии x добавить или удалить соответственно, для роли или административной роли r запрещающую роль nr. Для этого субъекта-сессия x должна обладать текущей специальной административной ролью negative_roles_admin_role, а также соответственно либо специальной административной ролью roles_admin_role, либо admin_roles_admin_role. При этом нельзя задавать запрещающие роли для любых специальных административных ролей из множества SAR, и для упрощения администрирования системы при добавлении запрещающей роли для роли или административной роли r ни одна субъект-сессия не должна иметь к ней административных доступов на чтение.

Де-юре правила вида create_object(x,y,yd,name,z), create_container(x,y,yd,name,z) и delete_entity(x,y,z) позволяют субъект-сессии x создать или удалить сущность-объект или сущность-контейнер y, входящую в состав сущности-контейнера z, к которой субъект-сессия x должна иметь доступ на запись и обладать текущей ролью или административной ролью, имеющей к z право доступа на выполнение $execute_r$, и одновременно не обладать текущей запрещающей ролью, имеющей к z это право доступа. В соответствии со спецификой

файловой системы ОССН нельзя создавать одноименные сущности в одной сущности-контейнере или удалять непустые сущности-контейнеры (удаление таких сущностей-контейнеров реализуется в ОССН рекурсивно, с использованием удаления сущностей-объектов и пустых сущностей-контейнеров), а при удалении сущности объекта должно быть проверено отсутствие на него других “жестких” ссылок.

Де-юре правила вида *create_hard_link(x, y, name, z)* и *delete_hard_link(x, y, name, z)* позволяют субъект-сессии *x* создать или удалить соответственно, в составе сущности-контейнера *z* (к которой субъект-сессия *x* должна иметь доступ на запись и обладать текущей ролью или административной ролью, имеющей к *z* право доступа на выполнение $execute_r$, и одновременно не обладать текущей запрещающей ролью, имеющей к *z* это право доступа) «жёсткую» ссылку на сущность-объект y. При создании «жёсткой» ссылки на сущность y учитываются права доступа к сущностям-контейнерам, ее содержащим. Так как в ОС семейства *Linux* возможно создание в одной сущности-контейнере нескольких «жёстких» ссылок (с разными именами) на одну сущность-объект. то в условиях применения правила *create_hard.link* не требуется отсутствие сущности-объекта y в составе сущности-контейнера z, а в правиле *delete_hard_link* указывают имя «жёсткой» ссылки, с которым она будет удалена. Поскольку «жёсткие» ссылки создаются только на объекты, то не требуются проверки на появление циклов в иерархии сущностей. При создании «жёсткой» ссылки учитывается вид её метки, в результате, если у сущности y метка прямая, то она прямая у сущности-контейнера z, а если у сущности y метка косвенная, то “жесткая” ссылка на нее может быть осуществлена только когда сущность контейнера z подчинена в иерархии единственной удовлетворяющей условию 8 предположения 2.2 сущности-контейнеру с прямой меткой старшей в иерархии x и y. При удалении проверяется, действительно ли удаляется “жесткая” ссылка (т.е. есть еще другие ссылки на нее), и учитывается, осуществляется ли удаление “жесткой” ссылки в разделяемой сущности-объекта и “жесткой” ссылки на него реализуется, как правило, одной функцией. В то же время, так как формально результаты таких удалений существенно отличаются, то для удобства в рамках модели заданы два правила.

Де-юре правила вида *rename_entity(x,y, old_name, name, z) u name_role(x, ry, name*) позволяют субъект-сессии x переименовать сущность *y,* входящую в состав сущности-контейнера z , к которой субъект-сессия *x* должна иметь доступ на запись и обладать текущей ролью или административной ролью, имеющей к ней право доступа на выполнение $execute_r$, и одновременно не обладать текущей запрещающей ролью, имеющей к *z* это право доступа, или соответственно переименовать роль, запрещающую роль или административную роль *ry* во всех ролях-контейнерах, в которые она входит (так как каждая роль, запрещающая роль или административная роль имеет в отличие от сущностей уникальное имя), и x которым

субъект-сессия *x* должна иметь административные добуду на запись. Поскольку на сущность-объект может быть несколько «жёстких» ссылок в одной сущности-контейнере и, следовательно, несколько имён, то в правило добавлен параметр, указывающей старое имя сущности, подлежащее замене. Для ролей такой параметр не требуется, так как роль должна иметь в системе уникальное имя. Правило переименования роли не использовалось в предыдущих ДП-моделях и включено в МРОСЛ ДП-модель с учётом реализованного в ней представления ролей как аналогов сущностей-контейнеров. В связи с этим для переименования роли, запрещающей роли или административной роли требуется наличие у субъект-сессии доступа на чтение к административной роли roles_admin_role*,* *negative_roles_admin_role* или *admin_role_.admin_role* соответственно, при этом нельзя переименовывать роли, являющиеся индивидуальными ролями и индивидуальными административными ролями учётных записей пользователей, а также общей Ролью и специальными административными ролями из множества SAR. При переименовании сущности учитывается, осуществляется ли одно в разделяемой сущности-контейнере z или нет.

Де-юре правило вида *read_container(x, y, z)* позволяет субъекту-сессии x «считать» в сущность-объект *z* (например, в ОССН в сущности “Рабочий стол”), к которой она должна иметь доступ на запись, содержимое (имена входящих в неё непосредственно сущностей или ролей) сущности-контейнера, роли, запрещающей роли или административной роли y, к которой x должна иметь через соответствующую роль или административную роль права доступа на чтение и выполнение, с учетом прав доступа к сущностям-контейнерам, содержащим y, а так же не иметь текущих запрещающих ролей, обладающих к y правами доступа на чтение или на выполнение.

Де-юре правило вида *get_entity_attr(x,y,z), get_subject_attr(x,y,z) u get_role_attr(x,y,z),* позволяют субъект-сессии x “считать” в сущности-объекта z, к которой она должна иметь доступ на запись, атрибуты сущности, субъекты-сессии, роли, запрещающие роли или административные роли y соответственно. В случае, когда уявляется сущностью, требуется, чтобы субъект-сессия x могла получить к ней доступ с учетом прав доступа x к сущности-контейнерам, содержащим y. А в тех случаях, когда для предоставления параметров сущности или субъект-сессии x текущей роли, обладающей к y правом доступа владения, дополнительно требуется отсутствие у x текущей запрещающей роли, обладающей к y правом доступа владения.

Для сущности *y* выдаются следующие атрибуты:

- вид метки;
- для сущности-контейнера является ли она разделяемой;
- для сущности-объекта число «жёстких» ссылок на неё;

- роли, запрещающие роли или административные роли и имеющиеся у них права доступа к сущности (в случае, когда *x* имеет административный доступ на чтение либо к роли-«владельцу» сущности *y,* либо к административной роли *entities-admin,role)- Для* субъект-сессии *y* выдаются следующие атрибуты:
- учётная запись пользователя, от имени которой она функционирует;
- роль-“владелец” субъект-сессии (в случае, когда x имеет административный доступ на чтение либо к роли-“владельцу” субъект-сессии y, либо к административной роли *subjects_admin_role*), а также запрещающие роли, имеющие права доступа владения к субъект-сессии;
- текущие административные доступы субъект-сессии к ролям, запрещающим ролям или административным ролям (в случае, когда x имеет административный доступ на чтение либо к роли-“владельцу” субъект-сессии y, либо к административной роли *subjects_admin_role*);
- вид метки (всегда *true);*
- является ли она разделяемой (всегда true, выдается для общности формата данных с аналогичными данными для сущностей);
- число «жёстких» ссылок на роль, запрещающую роль или административную роль (в случае, когда x имеет административный доступ на чтение административной роли *roles_admin_role, negatıve_roles_admin_role* или *admin_roles_admin_role* соответственно);
- административные роли и имеющиеся у них права доступа к роди, запрещающей роли или административной роли *y* (в случае когда *x* имеет административный доступ на чтение к административной роли *roles_admin_role, negatiue_roles_admin_role* или *admin_roles_admin_role,* соответственно);
- значение функции $constraint_{NR}$ — для роли и административной роли, заданные для неё запрещающие роли, для запрещающей роли — роли или административные роли, для которых она таковой является (в случае, когда x имеет административный доступ на чтение к административной роли *negative_roles_admin_role).*

Де-юре правило вида *set_container_attr(x, y,* t) позволяет субъект-сессии x задать сущности-контейнеру *y* является ли она разделяемой или нет. При этом субъект-сессия *x* должна иметь либо административный доступ на чтение к роли-«владельцу» контейнера *y* и не иметь текущей запрещающей роли, обладающей правом доступа владения к *y,* либо иметь административный доступ на чтение к административной роли *entıties_admin_role,* а также требуется, чтобы доступ к *y* мог быть предоставлен x с учётом её прав доступа к сущностям-контейнерам, содержащим *y.*

Де-юре правило вида *creat_.first_subject(x, u, y, z)* позволяет субъект-сессии x с использованием сущности *y* и учётной записи пользователя *u* создать от имени *u* новую субъект-сессию z. Для этого требуется наличие у субъект-сессии x доступа на чтение к роли

обладающей правом доступа на выполнение к сущности *y* (к которой субъект-сессия *x* может получить доступ с учётом прав доступа к сущностям-контейнерам, содержащим сущность *y),* и отсутствием у x текущей запрещающей роли, обладающей правом доступа выполнение к *y.* После создания субъект-сессии z она (в отличии от предшествующих ДП-моделей) получает доступы на запись и чтение к индивидуальной роли *u_c* и общей роли *common_role,* и доступ на чтение к индивидуальной административной роли *u_admin.* При этом индивидуальная роль *u_c* получает право доступа владе ния к субъект-сессии *z*.Кроме того, для того чтобы создаваемая субъект-сессия *z* могла получить в качестве текущих все запрещающие роли для индивидуальной административной роли, индивидуальной роли её учетной записи пользователя и для общей роли, проверяется наличие права доступа на чтение ко всем таким запрещающим ролям у индивидуальной административной роли учетной записи пользователя субъект-сессии *z* получает административный доступ на чтение ко всем этим запрещающим ролям.

Дю-юре правила вида *create_subject(x,y,z)* и *delete_subject(x,z)*позволяют субъект-сессии x создать или удалить, соответственно, субъект-сессию *z*. При создании требуется наличие у субъект-сессии *x* доступа на чтение к роли, обладающей правом доступа на выполнения к сущности *y* (к которой субъект-сессия *x* может получить доступ с учетом прав доступа к сущностям-контейнерам, содержащим сущность *y*), и отсутствие *x y* текущей запрещающей роли, обладающей правом доступа на выполнение к *y*. После создания субъект-сессии *z* она получает доступы на запись и чтение к индивидуальной роли *user(x)_c* и общей роли *common_role*, и доступа на чтение к индивидуальной административной роли *user(x)_admin*, соответствующим её учётной записи пользователя (у субъект-сессий *x* и *y* ,общая учётная запись пользователя), и непосредственно подчиняется в иерархии субъект-сессии *x*. При этом индивидуальная роль *user(x)_c* получает право доступа владения к субъект-сессии *z*. Аналогично предыдущему де-юре правилу проверяется наличие права доступа на чтение ко всем: соответствующим запрещающим ролям у индивидуальной административной роли учётной записи пользователя субъект-сессии *x*. После применения правил созданная субъект-сессии *z* получает административный доступ на чтение ко всем этим запрещающим ролям. При удалении субъект-сессии z требуется, чтобы ее в иерархии не подчинялись другие субъект-сессии (в противном случае удаление надо начинать с них), и требуется наличие у субъект-сессии *x* доступа на чтение x роли-«владельцу» (обладающей правом доступа владения) субъект-сессии *z*, а также отсутствие у *x* текущей запрещающей роли, обладающей правом доступа владения к *z.*

С целью дальнейшего формального обоснования свойств систем, построенных в рамках МРОСЛ ДП-модели, целесообразно доказать, что условия и результаты правил преобразования состояний систем заданы корректно, то есть соответствуют предположению

2.2. При этом с учетом особенностей задания этого предположения требуется доказать, что его условия выполняются как в состояниях на траекториях функционирования системы, так и при переходах между состояниями.

Утверждения 2.1. Пусть Go-начальное состояние системы *Σ(G',OP,Go)*, удовлетворяющее условиям предположения 2.2. Тогда для любой траектории *Go-op1 G1-op2 Gn,где N* ≥1,в состоянии *Gn* выполняются условия предложения 2.2,а также переход *Gn-1 opn Gn* удовлетворяет условиям этого предположения.

Доказательство этого утверждения осуществляется аналогично доказательству соответствующего утверждения для уровня ролевого доступом, изложенному в [33].

2.2.4. Подходы к моделированию ролевого управления доступом в СУБД *PostgreSQL*

СУБД P*ostgreSQL* является основной из применяемых на базе ОССН, механизмы управления доступа которой поэтапно интегрируется с соответствующими механизмами ОССН, что в перспективе создает предпосылки для противодействия запрещённым информационным . потокам по памяти и по времени, возникающим при взаимодействии компонент СУБД и ОССН между собой. По этой причине в рамках иерархического представления МРОСЛ ДП-модели разработаны уровни для СУБД P*ostgreSQL* [24], «опирающиеся» на соответствующие уровни для ОССН (см. рис. 2.1).

Вместе с тем основанный на стандарте *SQL* механизм управления доступом в СУБД, имеет ряд существенных отличий от аналогичного механизма ОССН, он гораздо сложнее поддаётся модификации. Поэтому при его моделировании в отличие от [70, 71, 91] не учитывается многие детали его реализации, так как это затруднит разработку подходов по противодействию запрещённым информационным потокам по времени или потребует использования для их применения существенных ресурсов СУБД, что может негативно сказаться на ее производительности.

В результате на уровне ролевого управления доступом СУБД P*ostgreSQL* МРОСЛ ДП-модели использованы:

- множества: ролей СУБД, административных привилегий СУБД *(SUPERUSER,CREATEROLE,CREATEDB,LOGIN,REPLECATION,INHERIT)*,административных ролей СУБД, специальных ролей СУБД, общих ролей СУБД, элементов СУБД, элементов-объектов СУБД(каталоги по событию, расширения, сопоставления, домены, конфигурации, словари, парсеры, шаблоны, функции, последовательности, строки, ограничения, индексы, правила, триггеры, триггерные функции, репликации),элементов-контейнеров СУБД (кластеры, базы данных, схем , таблицы , столбцы, представления , табличные пространства), видов привилегий СУБД (*SELECT, INSERT, PDATE, DELETE, TRUNCATE, REFERENCE, TRIGGER, USAGE, CREATE, CONNECT, TEMPORARY, TEMP, EXECUTE, OWN),* сущностей СУБД (на этом уровне модели сущностями являются элементы СУБД от ее

схем и далее выше по иерархии, например, базы данных, кластеры), привилегий к элементам СУБД;

• функции: административных привилегий ролей СУБД ролей, входа субъект-сессий в СУБД, наследования привилегий ролей к элементам СУБД, управления подчинённостью ролей в иерархии, административных прав доступа административных ролей ОССН и СУБД к ролям СУБД, привилегий к элементам СУБД ролей СУБД, соответствия административных привилегий и видов привилегий к элементам СУБД правам доступа, эффективных прав доступа ролей СУБД.

Кроме того, переопределены:

• множество доступов субъект-сессий к ролям, запрещающим ролям, административным ролям или ролям СУБД;

• функции: имён сущностей и элементов СУБД, имён ролей, запрещающих ролей, административных ролей, ролей СУБД, доступа субъект-сессий к сущностям или элементам СУБД в контейнерах;

• иерархия сущностей и элементов СУБД;

• иерархия ролей, запрещающих ролей, административных ролей и ролей СУБД;

• состояние системы.

Сформулированы несколько десятков условий, которым должно удовлетворять единое ролевое управление доступом в СУБД и ОССН, в том числе определяющих порядок:

• использования административных прав доступа, привилегий ролей и специальных административных ролей СУБД;

• создания, удаления, изменения иерархии ролей СУБД;

• администрирования иерархии сущностей и элементов СУБД;

• управления доступом к сущностям СУБД;

• использование административных доступов субъект-сессий к ролям СУБД.

При этом по аналогии с уровнями модели для ОССН описание условий, касающихся порядка реализации в СУБД мандатного контроля целостности и мандатного управления доступом, осуществлено на последующих соответствующих уровнях модели.

На уровне ролевого управления доступом СУБД P*ostgreSQL* иерархического представления МРОСЛ ДП-модели по сравнению с уровнем ролевого управления доступом для ОССН не было задано новых де-юре правил преобразования состояний, их осталось 35, хотя большая их часть была существенно модифицирована путем добавления в правила новых параметров, условий и результатов применения , учитывающих специфику управления доступом в СУБД.В итоге сформулировано и обосновано утверждение (аналогичное утверждению 2.1) о соответствии (корректности) правил преобразования состояний системы

заданным на этом уровне модели условиям, которым должно удовлетворять ролевое управление доступом.

Таким образом, удалось разработать и апробировать подходы х моделированию ролевого управления доступом в СУБА, P*ostgreSQL*, совместимые с использованными при формировании иерархического представления МРОСЛ ДП-модели, изначально предназначавшегося для моделирования безопасности мандатного и ролевого управления доступом, мандатного контроля целостности в ОССН. Это создаёт предпосылки для дальнейшего использования этих видов управления доступом при развитии модели управления доступом в СУБД, её верификации с использованием инструментальных средств, практической реализации единого механизма управления тупом в ОССН и СУВА, а также сертификации СУБД P*ostgreSQL* на соответствии требованиям высоких уровней доверия [83].

2.3. Уровень мандатного контроля целостности

2.3.1. Элементы состояния системы

Рассмотрим основные элементы, вводимые на уровне мандатного контроля целостности МРОСЛ ДП-модели. Используем следующее предположение.

Предположение 2.3. Каждая учётная запись пользователя является учётной записью либо доверенного, либо недоверенного пользователя. Каждая субъект-сессия является либо доверенной,либо недоверенной.От имени учетных записей недоверенных пользователей могут функционировать только недовренные субъект-сессии.

Здесь и далее использован классический для теории моделирования безопасности управления доступом подход, заключающийся в следующем. Как правило, предполагается, что проверка безопасности системы должна осуществляться после того, как доверенные (привилегированные, административные субъект-сессии, являющиеся частью под безопасности, чью функциональность можно верифицировать) субъект-сессий выполнили свои задачи по изменению параметров функционирования системы, в том числе по передаче прав доступа или созданию новых доверенных субъект-сессий. При этом требуется удостоверяться, что дальнейшее функционирование системы под воздействием недоверенных (от имени учётных записей непривилегированных пользователей, часто рассматриваемых как нарушители) субъект-сессий будет безопасным, т.е. не приведёт к несанкционированной передаче прав доступа или созданию

запрещённых информационных потоков. Однако для более полного описания требований к реализации управления доступом в ОССН в модель будут включены элементы, используемые для администрирования ОССН доверенными субъект-сессиями.

Используем следующие обозначения:

Rf={write m}-множество видов информационных потоков (при обеспечении целостности рассматриваются только информационные потоки по памяти);

F<(E u S) x (E u S) x Rf-множество информационных потоков;

(LI, ⩽)-решетка уровней целостности данных, при этом заданы минимальной *i_low=+LI* и максимальный *i_high=+LI* элементы решетки соответственно;

(*iu*,*ic*,*ir,is)* ∈ I- четвёрка функций уровней целостности, при этом:

iu:U-LI функция, задающая для каждой учётной записи пользователя её уровень целостности — максимальный разрешённый уровень целостности субъект-сессий, функционирующих от ее имени*;*

ie:U-LI функция, задающая уровень целостности для каждой сущности*;*

ir:R u NR u AR – LI-функция, задающая для каждой роли, запрещающей роли или административной её уровень целостности;

is: S-LI- функция, задающая для каждой субъект-сессий ее текущий уровень целостности;

I-множество всех четверок функций заданного вида;

CCRI:E u R u NR u AR-{true,false}-функция, задающая способ доступа к сущностям внутри контейнеров, ролям, запрещающим или административным ролям в иерархии ролей (с учетом их мандатных уровней целостностей). Если сущность *e* ∈ *C* является контейнером и доступом сущностям, содержащимся внутри контейнера e, разрешен без учета уровня целостности контейнера *E*, то по определению выполняется равенство *CCRI(e) = false*, в противном случае выполняется равенство *CCRI(e) = true.* При этом по определению для каждой сущности объекта o ∈ *O* выполняется равенство *CCRI(o) = true*. При этом для ролей, запрещающих ролей административных ролей *r* ∈ *R U NR U AR* выполняется равенство *CCRI(r) = false.*

Реализация мандатного контроля целостности целесообразно в современных защищённых ОС. Такой механизмы *(MIC – Mandatory Integrity Control u UAC — User Account Control)* в ОС семейства *Microsoft Windows Vista/7/2008 [108]* позволил существенно повысить их защищенность. С его применением все учётные записи пользователей, субъект-сессии, сущности и роли чётко разделяются на два непересекающихся множества — влияющих на целостность и безопасность ОССН (соответствующих, например, уровню целостности *i_high*) или нет (соответствующих, например, уровню целостности *i_low*). Преимущества мандатного контроля целостности аналогичны преимуществам мандатного управления доступом в сравнении

с дискреционным управлением доступом, т.е. обеспечивается большая ясность правил разделения компонент системы на критичные и некритичные с точки зрения её целостности, тем самым снижается риски ошибок.

В мандатном управлении доступом решётка многоуровневой безопасности содержит достаточно много элементов (за счёт наличия неиерархических категорий), при этом требуется противодействие запрещенным информационным потокам по времени. Таким

образом, для обеспечения большей надёжности мандатное управление доступом целесообразно в дальнейшем реализовывать в модели раздельно от ролевого управления доступом. Иначе потребовалось бы задание сложной ролевой иерархии с разделением ролей, как минимум , на «роли на чтение» и «роли на запись» (например, как в мандатной модели *RBAC* [17, 134]). При этом пользователям будет достаточно сложно выбирать необходимый набор ролей и корректно их администрировать. Для мандатного контроля целостности достаточно меньшего числа уровней и его задание с помощью ролей может оказаться более понятным для пользователей и администраторов ОССН, кроме того, требуемый для пользователя уровень целостности (уровень целостности субъект-сессий, функционирующих от имени его учетной записи) непосредственно зависит от выполняемой пользователем функции (роли) в защищенной ОССН. При этом в большинстве случаев пользователи будут тать в ОССН, используя только низкий (минимальный) уровень целостности (вне зависимости от текущего уровня доступа манда,, управления доступом), а высокий уровень целостности (или уровень выше минимального) будет необходим только при выполнении административных или системных функций. Мандатные атрибуты целостности *CCHI* используются аналогично мандатным атрибутам конфиденциальности сущностей-контейнеров *CCHR* (*Container Clearance Required*), наследованным из модели систем военных сообщений (СВС) [17, 121]. В качестве источников и приёмников информационных потоков по памяти на уровне мандатного контроля целостности рассматриваются только сущности и субъект-сессии. так как роли, запрещающие роли или административные (что будет учтено в перспективе) могут использоваться только для созданы информационных потоков по времени.

Для отражения этих изменений на текущем уровне модели в функции *execute_container* дополняется одно из её условий:

Условие 3. Существует $ri \in R$ *u AR* такая, что *(s,r1,read)* $\in \in AA$ *u (ci execute)* $\in PA$ *(ri),*верно *ie(ei)* $\leqslant$ *is(s)* или *CCRI (ei)= false*, где $1 \leqslant I < n$.

Определение 2.5. Пусть определены множество информационных потоков *F* и функции уровней целостности (*iu, ie, ir, is)* и мандатных атрибутов целостности CCHR. Переопределим *G = = (APA, PA, user, (iu, ie, ir, is), CCRI, A, AA, F, Hr, He,Hs, constraint nr)*— состояние системы.

Предположение 2.4. В рамках МРОСЛ ДП-модели новь» значения для функции задаются только при создании соответствующих новых субъект-сессий. В начальном состоянии *Go* любой системы *Σ (G* , OP, Go)* для любых субъект-сессии $s \in So$ и сущности $e \in Eo$,если $(s,e,a) \in Ao$,где $a \in Ra$,то существеует контейнер $c \in Co$ и справедливо равенство *execute_container0(s,c,e)=true.*

В предложении требуется, чтобы в начальном состоянии доступы субъект-сессии к сущностям соответствовали мандатному контролю целостности. Кроме того, по сравнению

с предшествующим ДП-модели в рамках МРОСЛ ДП-модели допускается администрирование существенных параметров системы , влияющих на ее безопасность (множество учетных записей пользователей, ролей, запрещающих ролей или административных ролей, значений уровней целостности четных записей пользователей и сущностей). Это целесообразно для строгого задания соответствующих правил преобразования состояний системы, ориентированных на реализацию в ОССН. Однако для сохранения преемственности с классическим подходом в дальнейшем планируется показать целесообразность рассмотрения при анализе безопасности системы только таких траекторий её функционирования, на которых данные «административные» правила не будут использоваться.

2.3.2. Функционально или параметрически ассоциированные сущности

Сформулируем предположения, позволяющие учесть наличие в ОССН сущностей, функционально или параметрически ассоциированных с учётными записями пользователей, субъект-сессиями или ролями.

Предположение 2.5. Для каждой учётной записи пользователя задаётся множество сущностей, параметрически с ней ассоциированных, получение доступов на чтение или на запись к которым субъект-сессией необходимо для того, чтобы создать субъект-сессию от имени данной учётной записи пользователя. Множество сущностей, параметрически ассоциированных с учётной записью пользователя не изменяется в ‘процессе функционирования системы.

Примерами параметрически ассоциированных сущностей с учётными записями пользователей являются файлы паролей, *cookies* или ключей и т.д., т.е. «знание» чего (получение к ним доступа на чтение) или «подмена чего (получение к ним доступа на запись) позволяет недоверенной субъект-сессии создать в системе субъект-сессию от имени данной учётной записи пользователя. При реализации ОССН необходимо иметь возможность для каждой учётной записи пользователя чётко описывать множество всех параметрически ассоциированных с нею сущностей, т.е. может потребоваться некоторая методика, возможно, автоматизированная в ПО, осуществляющем формирование множества параметрически ассоциированных сущностей для каждой учётной записи. пользователя.

Предположение 2.6. Параметрически ассоциированными сущностями с субъект-сессией являются сущности, которые содержат данные, позволяющие захватить управление (контроль) над нею. Множество сущностей, параметрически ассоциированных с субъект-сессией, задается в момент ее создания только в зависимости от сущности, из которой создается данная субъект-сессия, и учетной записи пользователя, от имени которой другая субъект-сессия создает данную субъект-сессию.

Примерами параметрически ассоциированных сущностей с субъект-сессии являются файлами *cookies.* «Знание» таких сущностей (получение от них информационного потока по памяти) позволяет недоваренной субъект-сессии «выдать» себя за другую субъект-сессию (получить над нею контроль). В ОССН наличие у субъект-сессий параметрически ассоциированных сущностей достаточно редкое явление. Как правило, такие сущности есть у приложений, осуществляющих удалённый доступ к чему-либо, или у криптографических приложений, которым нужны соответствующие ключи. При реализации ОССН необходимо иметь возможность (соответствующую методику, ПО) для каждой субъект-сессии чётко описывать множество всех параметрически ассоциированных с нею сущностей.

Предположение 2.7. Функционально ассоциированными с субъект-сессией являются сущности, от которых зависит преобразование данных (функциональность), реализуемое субъект-сессией. Только информационный поток по памяти к сущности, функционально ассоциированной с субъект-сессией, приводит к изменению преобразования данных, реализуемого этой субъект-сессией, и позволяет захватить над нею управление (контроль). Множество сущностей, функционально ассоциированных с субъект-сессией, задается пря создании субъект-сессии. При создании новой субъект-сессий ,множество функционально ассоциированных с ней сущностей задаётся только в зависимости от сущности, из которой создаётся данная субъект-сессия, и учётной записи пользователя, от имени которой другая субъект-сессия создает данную субъектную-сессию.

Примерами функционально ассоциированных сущностей являются исполняемые файлы, файлы динамических библиотек, конфигурационные файлы, сегменты памяти, в которых размещается исполняемый код, стек и т.д. Таким образом, запись в эти сущности может позволить изменить функциональность субъект-сессии, поставить ее “под контроль” другой субъект-сессии. Описание всех функционально ассоциированных с субъект-сессией сущностей является задачей более сложной, чем описание параметрически ассоциированных сущностей, так как в ОССН достаточно трудно для каждой субъект-сессии в отдельности описывать в точности все, от чего зависит ее функциональность. Поэтому при реализации ОССН необходимо иметь возможность (соответствующим методику, ПО) для каждой субъект-сессии, возможно, с некоторой избыточностью описывать множество ее функционально ассоциированных сущностей, а любой компромисс здесь(пример, невключение сущности в соответствующее множество) должен быть обоснован как часть политики безопасности системы.

Предположение 2.8. Для каждой роли, запрещающей роли или административной роли задается (возможно, пустое) множество сущностей, параметрически с ней ассоциированных. При этом для получения или удаления субъект-сессией доступа к роли кроме наличия к данной роли соответствующего права доступа, этой субъект-сессией необходимо получить

доступ на чтение или на запись ко всем сущностям, параметрически ассоциированным с данной ролью. Множество сущностей , параметрически ассоциированных хотя бы с одной ролью, запрещающей ролью или административной ролью, не изменяется в процессе функционирования системы.

Примерами параметрически ассоциированных сущностей с ролями являются файлы паролей, файлов ключей. Лишь у небольшого числа ролей могут быть такие сущности, их наличие позволяет обеспечить ,дополнительную защиту при получении субъект-сессией доступа к роли, например, потребовав дополнительно повторно ввести пароль. При реализации ОССН необходимо иметь возможность (соответствующую методику, ПО) для каждой роли чётко описывать множество всех параметрически ассоциированных с нею сущностей. Поскольку все роли субъект-сессии получают через индивидуальные административные роли их учётных записей пользователей, то требуется совпадение соответствующих множеств параметрически ассоциированных сущностей с индивидуальными административными ролями и учётными записями пользователей.

С учетом сделанных предположений и замечаний используем следующие обозначения:

|*u*| $\in$ *E* – множество сущностей, параметрически ассоциированных с учетной записью пользователя u ∈ U (здесь используется предположение 2.5);

UE = {e $\in$ *]u[: u* $\in$ *U}* – множество сущностей, каждая из которых параметрически ассоциирована хотя бы с одной учетной записью пользователя;

|s| ∈ E – множество сущностей, параметрически ассоциированных с субъект-сессией *s* $\in$ *S*;

Fp: U x E – 2^E – функция , задающая множества сущностей, параметрически ассоциированных с субъект-сессией при ее создании из сущности от имени учетной записи пользователя. При этом если из сущности *e* $\in$ *E* от имени учетной записи пользователя *u* $\in$ *U* невозможно создать новую субъект-сессию, то по определению справедливо равенство *fp(u,e) = Ø;*

[s] c E- множество сущностей, функционально ассоциированных с субъект-сессией *s;*

fa:U x E- 2e-функция задающие от множества сущностей, функционально ассоциированных с субъект-сессией при ее создании от имени учётной записи пользователя из сущности.При этом если из сущности *e* $\in$ *E* от имени учётной записи пользователя *u* $\in$ *U* невозможно создать новую субъект-сессию, то по определению справедливо равенство *fa(u,e)=0*. Кроме того ,по определению выполняется условие: для каждой субъект-сессии s $\in$ *S* существует единственная сущность e $\in$ *E* такая, что справедливы равенства *fp(user(s)es)=]s[u fa(user(s)es)=[s]* (здесь используется предложение предположение 2.7)

]r[c E- множество сущности параметрически ассоциированных с ролью, запрещающей ролью или административной ролью *r*$\in$ *R u NR u AR (*здесь используется предположение 2.8.)

RE={e ∈]r[: r ∈ R u NR u AR}-множество сущностей, параметрически ассоциированных хотя бы с одной ролью, запрещающей ролью или административной ролью.

2.3.3. Условия корректности мандатного контроля
целостности для состояний системы

Для задания мандатного контроля целостности для элементов состояний системы используем следующее предположение.

Предположение 2.9. В системе на уровне мандатного контроля целостности МРОСЛ ДП-модели выполняются следующие условия.

Условие 1 (текущий уровень целостности субъект-сессии):

• учётные записи доверенных пользователей имеют высокий уровень целостности *i_high*, а недоверенных пользователей — меньший *i_high* (например*, i_low*), при этом используем следующие обозначения: LU = {u ∈ U: iu(u) = i_high}— множество учётных записей доверенных пользователей, Nu = {u ∈ U: iu(u) << i_high} — множество учётных записей недоверенных пользователей;

• текущий уровень целостности субъект-сессии не превосходит уровня целостности учётной записи пользователя, от имени которой она функционирует: для субъект-сессии s ∈ S верно неравенство is(s) <<iu(user(s)), при этом используем следующие обозначения:Ls={s ∈ *S:is(s)=i_high}*-множество доверенных субъект-сессий,Ns={s ∈ *S:is(s)< i_high}*-множество надоверенных субъект-сессий, по определению выполняется следующее условие: если для субъекта-сессии s ∈ *S верно is(s)=i_high,то user(s) ∈ Lu* (высокий уровень целостности могут иметь только субъект- сессии функционирующие от имени учетной записи доверенных пользователей) ;

• текущий уровень целостности субъекта и не превосходит текущего уровня целостности субъект-сессии, которой она подчинена в иерархии: для субъект-сессии s, *s′ Є S*, если *s ≤ s′,* то *is*(s) ≤ *is(s’).*

Условия 2 (уровень целостности сущности, вид метки) :

• уровень целостности сущности, входящей в состав сущности-контейнера, не превосходит уровня целостности сущности-контейнера: для сущностей *e,e’ Є E,* если *e ≤ e’, mo ie(e) ≤ ie(e’);*

• мандатный атрибут целостности CCRI каждой сущности с косвенной меткой равен *true*, ее уровень целостности равен уровню целостности соответствующей единственной по условию 8 предложения 2.2 сущности-контейнера: для каждой сущности e Є E такой, что *direct(e) = false*, существует единственная сущность-контейнер *c Є C* такая, что *direct(c) = true*, e меньше c и для любой сущности *e’ Є E* такой, что e’ меньше c, выполняется *ie(e’) = ie(c) u CRRI(e’) = true.*

Условия 3 (уровень целостности роли, запрещающий роли или административной роли):

• уровень целостности роли , запрещающей роли или административной роли не превосходит уровней целостности ролей, которым она подчинена в иерархия ролей: для ролей *r, r' Є R U NR U AR, если r ≤ r', то ir(r) ≤ ir(r');*

• все административные роли из множества *SAR* обладают высоким уровнем целостности: для административной роли *ar Є SAR* справедливо равенство *ir(ar) = i_high*.

Условие 4 (уровни целостности и вид метки параметрически ассоциированных сущностей):

• уровни целостность сущностей, параметрически ассоциированных с учетной записью пользователя , совпадает с её уровнем целостности , и метка таких сущностей всегда прямая: для e *Є UE* верно *direct(e) = true*, для сущностей *e Є]u[,* где *u Є U*,справедливо равенство *ie(e) = iu(u);*

• уровни целостности сущностей, параметрически ассоциированных с ролью или административной ролью, совпадают с её уровнем целостности, и метка таких сущностей всегда прямая: для *e Є RE* верно *direct(e) = true*, для сущностей *e Є]r[*, где *r Є R U AR*, справедливо равенство *ie(e) = ir(r);*

• у запрещающих ролей множество параметричеки ассоциированных сущностей пусто: для запрещающей роли *nr ∈ NR* верно *]nr[= 0*

•сущности параметричеки ассоциированных с субъект-сессиями имеют прямую метку: для каждых сущностей e,e' *∈ E,*учетной записи пользователя *u ∈ U,* субъект-сессии *s ∈ S* таких, что *e ∈ fp(u,e')* или *e ∈]s[,*верно *direct(e)=true*

Условие 5 (создание или удаление сущности, «жесткой» ссылки на неё, получение доступа к ней, изменение прав доступа ролей, запрещающих ролей или административных ролей к ней).

• субъект-сессия может создать, удалить, переименовать сущность, «жёсткую* ссылку на неё или изменить права доступа ролей, запрещающих ролей или административных ролей к ней только, если эта сущность обладает уровнем целостности не выше текущего уровня целостности субъект-сессии;

• любой доступ субъект-сессии к сущности, создание "жёсткой" ссылки на неё, получение или изменение параметров, прав доступа к ней, активизация из неё субъект-сессии должны осуществляться при выполнении условий доступа внутрь сущностей-контейнеров с учётом мандатных атрибутов целостности *CCRI*:для субъект-сессии s *∈ S* и сущности *e ∈ E,*если *(s,c,e) ∈ A,*то существует контейнер *c ∈ C u execute_container(s,c,e)=true,*где a *∈ Ra;*

•субъект-сессии может быть предоставлен доступ на запись к сущности только в случае, когда её уровень целостности не выше текущего уровня целостности субъект-сессии: для субъект-сессии s *∈ S* и сущности *e ∈ E*, если (s, e, write,) ∈ A, то ie(e) << i(s);

• задано множество сущностей-объектов E_HOLE с O, которое не изменяется на траекториях функционирования системы, доступы на запись или чтение к объектам из него не могут быть

использованы для реализации информационных потоков по памяти (в них нельзя хранить данные).

Условие 6 (создание или удаление роли, запрещающей роли административной роли, получение доступа к ней, изменение прав доступа административных ролей к ней, изменение иерархий ролей, администрирование функции необходимого обладания запрещающей ролью):

• субъект-сессия может создать, удалить, переименовать, получит административный доступ к роли, запрещающей роли, административной роли, «жёсткой" ссылке на нее (изменить иерархию ролей, запрещающих или административных ролей) или изменить права доступа административных ролей к ней только , если эта роль , запрещающая роль или административная роль обладает уровнем целостности не выше текущего уровня целостности субъект-сессии : для субъект-сессии s ∈ S, роли, запрещающей роли или административной роли *r ∈ R U NR U AR*, если *(s, r, aa) ∈ AA, mo ir(r) ≤ is(s), где aa ∈ Ra;*

•для изменения значения функции необходимого обладания запрещающей ролью *constraint NR* субъект-сессия должна иметь уровень целостности не ниже уровней целостности той роли или административный роли , для каждой меняется значение этой функции , а также запрещающей роли, которая добавляется или удаляется из множества значений этой функции;

•если для некоторой роли или административной роли задана запрещающая роль , то уровни целостности этих ролей должны быть равны: для роли или административной роли *r ∈ R U AR*, если запрещающая роль *nr ∈ constraint NR(r), mo ir® = ir(nr).*

Условия 7 (создание , удаление субъект-сессии, изменение её роли-«владельца»):

• текущий уровень целостности активизируемой субъект-сессии не превосходит уровня целостности используемый для активизации сущности и уровня целостности учетной записи пользователя , от имени которой она функционирует , при этом должны быть учтены параметрически ассоциированные с ней сущности (должно быть выполнено предположение 2.5);

• субъект-сессия может изменить роль или административную роль, обладающую правом доступа владение к некоторой субъект сессии, или удалить субъект-сессию только с текущим уровнем целостности, не превосходящим текущего уровня целостности первой субъект-сессии;

• субъект-сессия может назначить или удалить права доступа владение запрещающей роли к некоторой субъект-сессии только с текущим уровнем целостности , не превосходящим текущего уровня целостности первой субъект-сессии.

Условие 8 (специальная сущность для доступа к элементам системы с высоким уровнем целостности). Задана специальная сущность-объект *i_entity ∈ E_HOLE*, обладающая высоким

уровнем целостности *ie(i_entity)* = *i_high* и прямой меткой *direct(i_entity)* = *true*, получение доступа на запись к сущности *i_entity* субъект-сессией (или кооперирующей с ней другой субъект- сессией) является необходимым, когда она осуществляет следующие действия:

• создаёт, удаляет, изменяет уровень целостности учетной записи пользователя

• создает субъект-сессию с уровнем целостности выше *i_low*;

• получает любой доступ к роли, запрещающей роли или административной роля с уровнем целостности выше *i_low*;

• изменяет значение функции необходимого обладания запрещающей ролью *constaint nr* для роли или административной роли с уровнем целостности выше *i_low*;

• включает или исключает из числа запрещающих ролей для роли или административной роли запрещающую роль с уровнем целостности выше *i_low* :

• получает доступ на запись к сущности с уровнем целостности выше *i_low* за исключением самой сущности *i_entity*;

• создаёт, удаляет, переименовывает, изменяет параметры сущности, роли, административной роли или «жёсткой» ссылки на неё с уровнем целостности выше *i_low*;

• изменяет содержимое сущности-контейнера, роли, запрещающей роли или административной роли с уровнем целостности повыше *i_low*;

• изменяет права доступа ролей , запрещающих ролей или административных ролей к сущностям или субъект сессиям с уровнем целостности выше *i_low*;

•изменяет административные права доступа административных ролей к ролям , запрещающим ролям или административным от ролям с уровнем целостности выше *i_low*;

• изменяет уровень целостности сущности .

Условие 9 (индивидуальные административные роли, индивидуальные роли учётной записи пользователя и общие роли) :

•для каждой учётной записи пользователя u $\in$ *U* для к уровня целостности , не превосходящего её уровня цел задаются индивидуальные административные роли , не авторизована только эта учётная запись , множество авторизованных ролей учётной записи пользователя с использованием административных прав доступа административных ролей , определяемых функцией APA, на траекториях функционирования системы у этих административных ролей не изменяются имена и уровни целостности, каждая такая индивидуальная административная роль обладает правами доступа на чтение, запись и выполнение ко всем индивидуальным административным ролям данной учетной записи пользователя с меньшим уровнем целостности ,при этом переопределено множество *U_ADMIN:*для каждых учетной записи пользователя *u* $\in$ *U* и уровня целостности *li*<< *iu(u)* задается административная роль u_admin_li $\in$ *AR* такая, что

*ir(u_c_li)=li, (U_admin_li',ar) ∈ APA(u_admin_li),*где *li'<<li,ar ∈{read,write,execute},U_ADMIN={u_admin_li: u ∈ U,li<<iu(u)};*

• на множестве индивидуальных административных ролей учтённой записи пользователя задаётся иерархия, которая изменяется только при создании, удалении и изменении уровней целостности соответствующих учетных записей пользователей и является независимой от иерархии других административных ролей: для каждых учётной записи пользователя u *∈ U* и её индивидуальных административных ролей *u_admin_li, u_admin_li', ∈ AR* справедливо *u_admin_li<<u_admin_li'* тогда и только тогда, когда *li<<li';*

• определённая для каждой учётной записи пользователя u *∈ U* индивидуальная административная роль *u_admin_i_low*.считается тождественной заданной в условии 9 предположения 2.2 индивидуальной административной роли *u_admin*, при этом переопределено множество *U_ADMIN:верно u_admin=u_admin_i_low,U_ADMIN={u_admin_li;* u *∈ U,li<<iu(u)};*

• для каждой учетной записи пользователя u *∈ U* для каждого уровня целостности, не превосходящего её уровня целостности, задаются индивидуальные роли, административными правами доступа на чтение, запись и выполнение к которым обладают ее индивидуальные административные роли с соответствующим уровнем целостности, на траекториях функционирования систем у этих административных ролей не изменяются имена и уровни целостности, при этом переопределено множество *U_ROLES:*для каждых учетной записи пользователя u *∈ U* и уровня целостности *li<<iu(u)* задается роль *u_c_li ∈ R* такая,что *ir(u_c_li)=li,(u_c_li,ar) ∈ APA (u_admin_lo),* где *ar ∈ {read,write,execute}, U_ROLES={u_c_li u ∈ U,li<<iu(u)};*

•на множестве индивидуальных ролей учётной записи пользователя задается иерархия , которая изменяется только при создании, удалении и изменении уровней целостности соответствующих учетных записей пользователей и является независимой от иерархии других ролей: для каждых учетной записи пользователей изменения уровней целостности пользователя u *∈ U* и ее индивидуальных административных ролей *u_c_li , u_c_li' ∈ R* справедливо *u_c_li<<u_c_li'* тогда и только тогда , когда *li<<li';*

•определённая для каждой учётной записи пользователя u *∈ U* индивидуальная роль *u_c_ (i _low)* считается тождественной данной в условии 9 предположения 2 . 2 индивидуальной роли, при этом переопределено множество *U_ROLES*; верно *u_c=u_c_(i_low)* , *U_ROLES = { u_c_li; u ∈ U , li<<iu(u*).

• для всех уровней целостности задаются общие роли , административными правами доступа на чтение , запись и выполнение к которым обладают индивидуальные административные роли всех учётных записей пользователей в соответствии с их уровнями целостности , при этом переопределено множество *COM-MON _ ROLES* : для каждого уровня

целостности li $\in LI$ задана общая роль *common_li* , при этом ir *(common_li*) = *li* , и для каждой учётной записи пользователя $u \in U$ верно (*common_li,ar*) $\in APA$ *(u_admin_li)* ,*где li << iu(u),ar,* $\in$ *{ read , writer , execute } . COMMON_ROLES = { common_li:li* $\in$ *LI } ;*

• на множестве общих ролей задаётся иерархия , независимая от иерархии других ролей , которая не изменяется на траекториях функционирования системы : для общих ролей *common_li* , *common_li'* $\in$ R справедливо с *common_li<< common_li'* тогда и только тогда , когда *li < li'*;

• общая роль *common_*(*i_low*) считается тождественной заданной в условии 9 предположения 2.2 общей роли common_role,при этом переопределено множество *COMMON_ROLES*:верно *common_role = common(i_low*) ,*COMMON_ROLES* = { *common_li* $\in$ *LI* } ;

•для каждой индивидуальной административной роли, индивидуальной роли учётной записи пользователя, общей роли специальной административной роли множество сущность параметрически с ней ассоциированных, не изменяется в процессе функционирования системы ;

•административные роли *roles_admin_role* и *admin_roles_admin_role* не используются для нарушения заданных иерарх вил назначения административных прав доступа к инициальным административным ролям , индивидуальные учетных записей пользователей и общим ролям.

Условие 10 (права доступа ролей , запрещающих ролей административных ролей):

•в начальном состоянии системы только учетные записи доверенных пользователей могут обладать административным ролями, имеющими права доступа к административным ролям, позволяющими получить права доступа владения к какой либо роли запрещающей роли или административной роли : в начальном состоянии системы для каждых учетной записи пользователя $u \in U$,роли или административной роли r $\in R\ U\ NR\ U$ AR, административной роли

u_admin_li $\in AR$,где *li* $\in LI$, если существует последовательность административных ролей *ar1… arn,* где *N>>1*,таких,что *ar1=u_admin_li,(r,own)* $\in$ *APA(arn),(ari+1,read)* $\in$ *APA (ar),*где *i=1…,N-1* то *u*-учетная запись доверенного пользователя;

• роль, запрещающие роль или административное роль может содержать право доступа на владение или запись к сущности с уровнем целостности не выше ,чем у роли запрещающие роли или административной роли: для роли или административной роли $r \in R\ U\ NR\ U\ AR$ и сущность e $\in E$,если (e,ar) $\in PA(r)$,*mo ie(e)<<ir(r),*где *ar* $\in$ *{own,write};*

•административная роль может содержать административное права доступа на владение, запись, или чтение к роли ,запрещающие роли или административные роли с уровнем целостности не выше, чем у первой административной роли: для административной роли ar $\in AR$ и роли, запрещающей роли или административной роли ar $\in Ar$ и роли, запрещающей

или административной роли *r ∈ R U NR U AR,* если *(r,a) ∈ APA (ar),* то *ir(r)<<ir(ar),*где *ar ∈ {own,write,read};*

•роль, запрещающие роль или административное роль может содержать право доступа на владение к субъект-сессии с текущим уровнем целостности не выше, чем уровень целостности роли запрещающие роли или административной роли для: роли запрещающие роли или административной роли r *∈ R U NR U AR* и субъект-сессии *s ∈ S, если (s,own) ∈ PA(r),*то *is(s)<<ir(r).*

Условие 11 (доступ субъект-сессии к индивидуальным административным, индивидуальным, общим ролям и их запрещающим ролям):

• при создании каждой субъект-сессии она получает доступ на чтение к индивидуальным административным ролям, к индивидуальным ролям ее учетной записи пользователя и общим ролям, уровень целостности которых не превосходит текущего уровня целостности субъект-сессии s *∈ S* выполняются условия *(s,user(s)_admin_li,read),(s,user(s)_c_li,read) ∈ (s,common_li,read) ∈ AA,*где *li<<is(s);*

• при создании каждой субъект сессии она получает доступ на запись к единственным таким индивидуальной роли и общей роли, уровня целостности которых равны соответственно, текущему уровню целостности субъект сессии , и дуальная роль получает право доступа владения к созданной субъект-сессии: для субъект сессии s*∈* S выполняются условия (*s,user (s)_c_is,(s), write) , (s ,common_is,(s), write) ∈ AA,(s,own) ∈ PA (user(s)_c_is,(s));*

• если для учётной записи пользователя индивидуальной административной роли или индивидуальной роли с некое уровнем целостности заданы запрещающие роли , то индивидуальная административная роль с соответствующим уровне , целостности этой учётной записи пользователя должна обладать административными правами доступа на чтение ко всем так , запрещающим ролям , и при создании от имени этой учётной за . писи пользователя субъект сессии она получает доступ на чтение ко всем соответствующим запрещающим ролям с соответствующим уровнем целостности : если для учётной записи пользователя u *∈* U выполняется *nr ∈ constraint nr (u _ admin _ li) U constraint NR (u _ c _ li) , mo (nr , read) ∈ APA (u_admin_li)*

• если для общей роли с некоторым уровнем целостности заданы запрещающие роли , то индивидуальная административная у роль каждой учётной записи пользователя , соответствующая

этому уровню целостности , должна обладать административными правами доступа на чтение ко всем таким запрещающие ролям , и при создании субъект сессии она получает доступ на чтение ко всем соответствующим запрещающим ролям с соответствующим уровнем целостности : если выполняется nr *∈ constraint nr* (*common_li*) , то для всех учётных записей пользователей *u ∈ U* верно *(nr , read) ∈ APA (u _ admin _ li)* , и для субъекта-

сессии s∈ S верно *(s , nr , read) ∈ AA , где li<< is (x) ;*
• при удалении субъект сессии удаляются все её доступ индивидуальным административным , индивидуальным и общим ролям .
- Условие 12 (администрирование уровней целости, параметров , имён сущностей , ролей и административных ролей):
• изменение уровня целостности , мандатного атрибута целостностиCCRI роли, запрещающей роли или административной роли (со значения *false*) запрещено ;
• сущности-контейнера создаются с мандатным атрибутом целостности CCRI равным *true;*
•для изменения мандатного атрибута целостности CCRI сущности-контейнера субъект сессии необходимо обладать либо теку ей ролью или административной ролью , имеющей право доступа владения к этой сущности контейнеру , либо доступом на чтение к административной роли *entities _ admin role* ;
•для изменения субъект сессией уровня целостности сущности к ей не должно быть доступов , и требуется наличие у субъект сессии текущей административной роли *entities _ admin role*.
Условие 13 (администрирование учётных записей пользователей) :
• для изменения уровня целостности учётной записи пользователя требуется наличие у субъект сессии текущих административных ролей *user_ admin _ role , roles _ admin _ role , negative_roles _ admin role u admin_roles_admin_role* из множества SAR , при этом множество субъект-сессий, функционирующих от имени данной учётной записи пользователя , должно быть пустым ;
• субъект-сессия может создать, удалить или изменить уровень целостности учётной записи пользователя с уровнем целостности не выше текущего уровня целостности этой субъект-сессии.
Рассмотрим особенности реализации в ОССН условий предположения 2.9. Условия 1 3 определяют порядок задания уровней целостности сущностей , ролей , запрещающих ролей и административных ролей (по аналогии с сущностями) с учётом их иерархии , уровней целостности учётных записей пользователей и субъект сессий с учётом их иерархии . Кроме того , в условии 1 определяется порядок задания уровней целостности учётных записей доверенных и недоверенных пользователей , что также обеспечивает чёткую классификацию компонентов ОССН на критичные и некритичные для обеспечения ее без опасности.. При этом недоверенными считаются все учётные записи пользователей , уровень целостности которых меньше *i_high* .Уровни целостности сущностей с косвенной меткой задаются аналогично правам доступа к ним.
Так как в рамках уровня мандатного контроля целостности МРОСЛ ДП- модели рассматриваются сущности , параметрически ассоциированные с

учётными записями пользователей, ролями запрещающим ролями или административными ролями , (которых позволяет либо создать субъект сессию о соответствующей учётной записи пользователя , либо полу ответствующей роли) , то по условию 4 их уровень жен совпадать с уровнем целостности учётной записи или уровнем целостности роли (административной роли) соответсвенно (при этом у запрещающих ролей не должно бы параметрически с ними ассоциированных , что связано необходимостью обеспечения возможности получения запрещающих ролей как текущих всеми субъект сессиями , когда для их ролей административных ролей указаны запрещающие роли , а также запрещающие роли не расширяют права доступа субъекта наоборот , сокращают их) . В ОССН маловероятна ситуация таких сущностей будет косвенная метка , в связи с этим треба чтобы она всегда была прямая . Описание таких сущностей в ОССН может быть легко реализовано . В то же время аналогичное описание сущностей , параметрически или функционально ассоциированных с субъект сессиями , труднореализуемо на практике . Выявление таких сущностей является самостоятельной задачей при разработеке защищённой ОССН . В связи с этим постулирование в предположениях модели аналогичных требований для сущностей , параметрически или функционально ассоциированных с субъект сессиями , нецелесообразно . Вместе с тем соответствует условиям функционирования ОССН предположение о том , что только сущность с прямой меткой может являться параметрически ассоциированной с субъект сессией.

Условия 5 , 6 и 7 задают для субъект сессий порядок создания , удаления , переименования , получения доступа к суши субъект сессиям , ролям , запрещающим ролям или административным ролям , администрирования функции необходимого запрещающей ролью *constraintNR* с учётом мандатного к целостности . Так как для него не представляют угроза информационные потоки по времени , то требования предъявляются доступам к сущностям на запись Для обеспечения условий реализации в ОССН некоторых типовых ресурсов во множество включены специальные объекты дырки » , которые не использованы для создания информационных потоков по памяти (например , “слушающие “ сокеты) . При доступе к ролям, запрещающим ролям или административным ролям требует доступ на чтение , так как он соответствует авторизации на текущую роль субъект сессией , и , следовательно , ее уровень целостности должен быть не ниже чем уровень целостности роли. При этом должно быть предусмотрено два способа создания субъект-сессий. Первый, когда субъект-сессия создается с “разрывом” связи с создающей субъект-сессией как первая субъект-сессия-корень иерархии (например , по командам *sudo или sumac*).В этом случае возможно создание субъект-сессии с уровнем целостности не выше уровня целостности учетной записи пользователей, от имени которой она создается. Второй способ, когда субъект-сессия создается как потомка активизирующей субъект-сессии. В этом случае ее уровень

целостности должен совпадать с уровнем целостности создающей субъект-сессии. Порядок создания сущностей или “жестких” ссылок на них в зависимости от вида их меток следует из условия 2.

Условие 8 задаёт в рамках модели требования к реализации механизма защиты, аналогичного механизму UAC в ОС семейство *Microsoft Windows*. Данный механизм предполагает что для осуществления ряда действий с использованием высокого уровня целостности требуется реализация доступа на запись к специальной сущности. В ОС семейства *Microsoft Windows* такому доступу соответствует возможность подтвердить (нажать кнопку на защищённой системном рабочем столе, доступ к которому запрещён для пользовательского ПО) необходимость выполнения действий с высоким уровнем целостности. Это условие требует, чтобы доступ запись субъект-сессией (или кооперирующей с ней другой субъект-сессией)к специальной сущности-объекту i_entity реализовывался всегда осуществляются изменения уровней целостности или иные любые действия с учётными записями пользователей, ролями, запрещающими : ролями, административными ролями, сущностями, субъект-сессиями с уровнями целостности выше минимального-*i_low*.

Применение по условию 9 соответствующих уровням целостности индивидуальных административных и индивидуальных ролей учетных записей пользователей и общих ролей для задания их авторизованных ролей <прав доступа к сущностям направлен на обеспечение приоритетности и монолитности мандатного контроля целостности и в первую очередь в дальнейшем мандатного управления доступом как при исправлении доступом к сущностям так и при использовании или администрировании ролей.

Выполнение условия 10 позволяет предотвратить несанкционированное администрирование и изменение иерархии ролей недоверенными субъект-сессиями. Такие субъект-сессии изначально (в начальном состоянии системы) лишаются возможности получить (через соответствующую административную роль) право доступа владения к любой роли. При этом одной из задач по обеспечению безопасности системы является предотвращение возможности перехода систем в состоянии в котором не будет выполняться часть условия 9,заданная для начального состояния системы. В соответствии с условием 9 для каждой роли, запрещающей роли или административной роли разрешается иметь права доступа на владение и запись только сущностям, ролям, запрещающим ролям или административным ролями с уровнем целостности не выше уровня целостности первой роли. Для соответствия условию 5,которое разрешает получение субъект-сессии доступа на чтение к роли. Таким образом, в соответствии с условиями 5-9 в модели задаются три «рубежа защиты» мандатного контроля целостности: первый — контроль уровней целостности при непосредственно предоставлении субъект-сессиям доступов к сущностям, ролям,

запрещающим ролям или административным ролям; второй — необходимость субъект-сессии при осуществлении действий на высоком уровне целостности получать доступ на запись к специальной, особо защищаемой ОССН сущности; третий — назначение каждой роли, запрещающей роли и административной роли прав доступа на владение и запись (и в ряде случаев на чтение) в соответствии с её уровнем целостности.

Условие 11 задаёт порядок получения доступа на чтение (авторизации) на индивидуальные административные роли учётной записи пользователя функционирующей от её имени создаваемой субъект-сессией; также доступов на чтение и запись на индивидуальные роли этой учётной записи пользователя и общие роли в зависимости от уровня целостности субъект-сессии. Кроме того, определяется порядок получения субъект-сессией как текущих запрещающих ролей, если они заданы с помощью функции *constainr nr* для текущих индивидуальных ролей его учетной записи пользователя и общих ролей. При этом требуется предоставление его индивидуальным административным ролям необходимых административных прав доступа на чтение к запрещающим ролям с учетом их уровней целостности.

Условие 12 предназначено для защиты от несанкционированно изменения мандатных атрибутов целостности CCRI,уровней целостности ,имен ролей запрещающих ролей административных ролей и сущностей. При этом изменение уровней целостности ролей, запрещающих ролей или административных ролей запрещено ,так как выполнение условия 9 при изменении такого уровня целостности всех сущностей к которым данная роль имеет права доступа на владение или запись, что в ОССН практически нереализуемо. Порядок изменения условия 2 и условия 8 предположения 2.2.

Условие 13 задает порядок администрирования (создания, удаления) учетных записей пользователей, получения их параметров с учетом мандатного контроля целостности, для получения их параметров с учетом мандатного контроля целостности, для чего требуется специальные административные роли *user_admin_role,roles_admin role,negative_roles_admin_role u admin_roles_admin_role.*

2.3.4. Фактическое владение и де-факто правила преобразования состояний

Ранее при разработке ДП-моделей для отражения ситуации, когда одна субъект-сессия получает контроль над другой субъект-сессией использовался доступ владения owna.В то же время в ОССН такой доступ, как правило явно не задаётся. Например, когда субъект-сессия осуществляет откладку другой субъект-сессии или когда первая субъект-сессия через переполнение буфера памяти получила над ней контроль. В связи с этим используем обозначение, дадим определение и сделаем предположения:

De_facto_own:S-2s- функция де-факто владения субъект-сессиями.

Определение 2.6. Будем говорить, что субъект-сессия *s* де-факто владеет субъект-сессией *s* когда выполняется условие s' $\in$ *de_facto_own(s).*При этом по определению всегда s $\in$ *de_facto)own(s).*

Предположение 2.10. Если субъект-сессия *s* реализовала информационный поток памяти от себя к сущности, функционально ассоциированной с другой субъект-сессией *s'* или субъект-сессия реализовала информационный поток по памяти от себя к *s'* и к себе от всех сущностей параметрически ассоциированных с другой субъект-сессией *s'* то субъект-сессия *s* получает де-факто владение субъект-сессией *s' (s'* $\in$ *de_facto)own(s)).*

Предположение 2 . 11 . Если субъект сессия s де-факто владеет субъект сессией s ' , то с учётом наличия у субъект-сессии s' текущих запрещающих ролей субъект сессия s получает возможности :

• использовать доступы субъект сессии s ' к сущностях ,административным ролям ;

• инициировать от имени субъект сессии s ' выполнение правил преобразования состояний системы ;

• получать де факто владение субъект сессиями , которые де-факто владеет субъект сессия s ' ;

• использовать информационные потоки , в реализации которых участвует субъект сессия s ' ;

• удалить субъект сессию s'.

Используем обозначения :

de_facto_accesses:— функция де факто доступов субъект сессий , при этом по определению в каждом состоянии системы G для каждой субъект сессии s $\in$ S верно равенство.

Определение 2 . 7 . Доступы из множества *de facto _ accesses* (s) будем называть де факто (фактическими) доступами субъект сессии s.

Фактическое владение , фактические доступы не требуют реализации в ОССН и используются для теоретического анализа её без опасности , а также для упрощения записи де факто правил преобразования состояний системы . При этом де факто владение субъект сессией другой субъект сессией позволяет первой из них управлять второй и пользоваться её возможностями (в том числе от ее имени правами доступа её текущих ролей или административных ролей). Однако , если у второй субъект сессии имеются запрещающие роли , обладающие некоторыми правами доступа , то воспользоваться соответствующими правами доступа для получения доступа к сущностям или субъект сессиям от имени второй субъект сессии можно .

С учётом добавления в модель по сравнению с её уровнем ролевого управления доступом мандатного контроля целостности рожденной решеткой уровней целостности де юре правила преобразования состоянии дополняются соответствующими условия результатами их

применения . Всего на уровне мандатного контроля целостности МРОСЛ ДП-модели заданы 38 де-юре правила преобразования состояний, из которых три правила вида *set_user_labels,set_entity_labels u set_role_labels* отсутствовали на уровне ролевого управления доступом так предназначены для изменения уровней целостного учётных записей пользователи или сущности а так же сущностей параметрически ассоциированных с учетными записями пользователей или ролями. При этои для каждого правила в табл. 2.2. в первой строке укажем его параметры условия и результаты применения наследованные с уровня ролевого управления доступом без изменения соответствующие уровню мандатного контроля целостности МРОСЛ ДП-модели.

Так как получение доступа на чтение с использованием де-юре правила вида access_read(x,x'y) к роли, запрещающей роли или

Примеры де-юре правила преобразования состояний МРОСЛ ДП-модели уровня мандатного контроля целостности

Таблица 2.2

Параметры	Исходное состояние G	Результирующее состояние G'
	$access\_read(x, x', y)$	
x, y	$x \in S, y \in E \cup R \cup NR \cup AR$, существует $r \in R \cup AR$: $(x, r, read_a) \in AA$, [если $y \in E$, то $(y, read_r) \in PA(r)$ и существует контейнер $c \in C$ такой, что $execute\_container(x, c, y) = true$, и не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(y, read_r) \in PA(nr)$], [если $y \in R \cup NR \cup AR$, то $(y, read_r) \in APA(r)$], [если $y \in R \cup AR$, то для всех $nr \in constraint_{NR}(y)$ верно $(x, nr, read_a) \in AA$]	если $y \in E$, то $A' = A \cup \{(x, y, read_a)\}$, если $y \in R \cup NR \cup AR$, то $AA' = AA \cup \cup \{(x, y, read_a)\}$
x'	$x' \in S$, [если $y \in R \cup NR \cup AR$, то $i_r(y) \leqslant i_s(x)$, для $e \in]y[$ либо $(x, e, read_a) \in A$, либо $(x, e, write_a) \in A$], [если $y \in R \cup NR \cup AR$ и $i_r(y) > i\_low$, то $(x', i\_entity, write_a) \in A$]	—
	$set\_entity\_labels(x, x', y, yi)$	
x, x', y, yi	$x, x' \in S, y \in E, y \notin UE \cup RE \cup \{i\_entity\}$, $direct(y) = true$, $(x, entities\_admin\_role, read_a) \in AA$, $\{(s, y, \alpha_a): s \in S, \alpha_a \in R_a\} \cap A = \varnothing$, $\max(i_e(y), yi) \leqslant i_s(x)$, [$yi \leqslant i_e(z)$ для всех $z \in E$ таких, что $y \in H_E(z)$], [$i_e(z) \leqslant yi$ для всех $z \in E$ таких, что $z \in H_E(y)$], [для всех $r \in R \cup NR \cup AR$ если $(y, \alpha_r) \in PA(r)$, то $yi \leqslant i_r(r)$, где $\alpha_r \in \{own_r, write_r\}$], [если $\max(i_e(y), yi) > i\_low$, то $(x', i\_entity, write_a) \in A$]	$i'_e(y) = yi$, если $H_E(y) = \{y' \in H_E(y): direct(y') = false\}$, то $\{(s, y', \alpha_a): s \in S, y' < y, \alpha_a \in R_a\} \cap A = \varnothing$ и для $y' < y$ верно $i'_e(y') = yi$

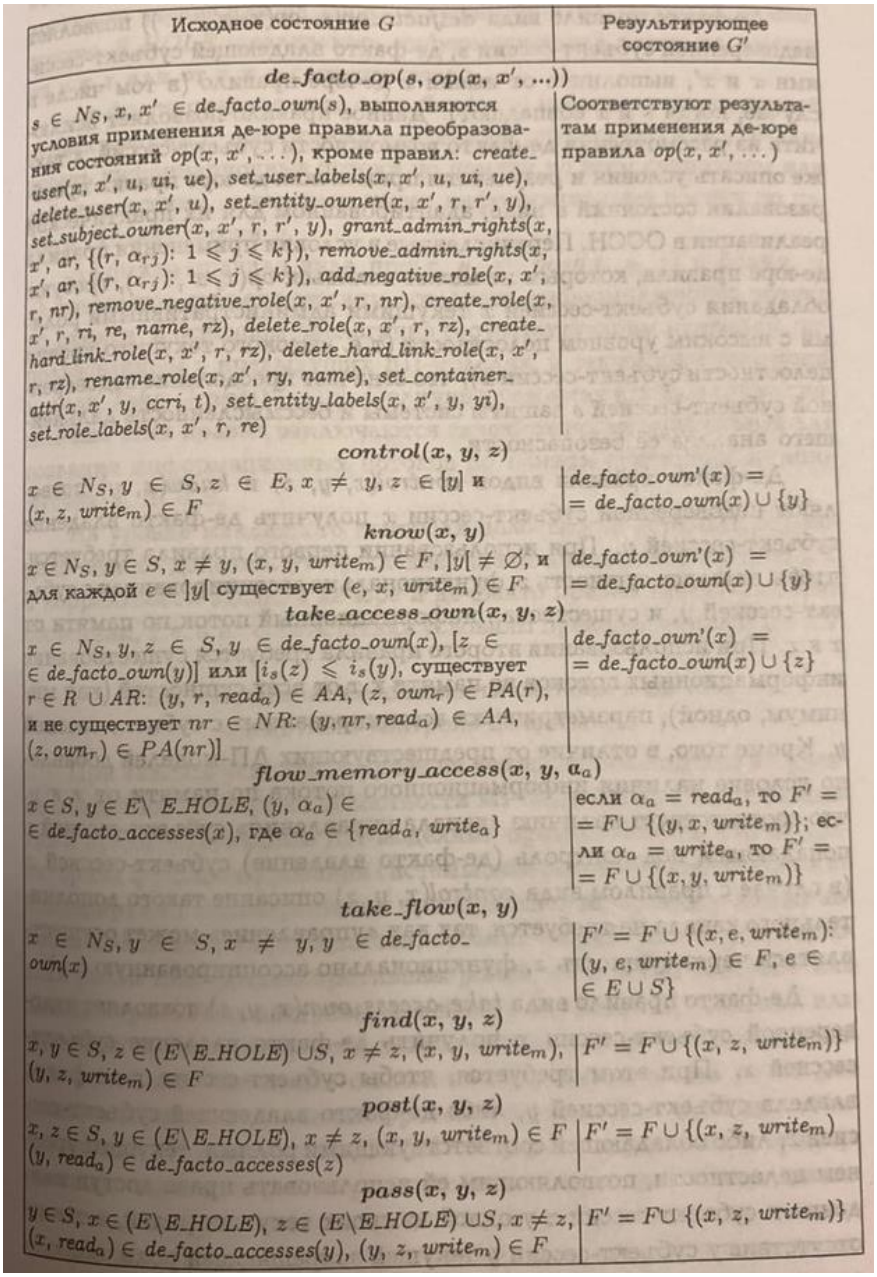
административной роли означает получение её как тек этом случае требуется , чтобы уровень целостности роли *y* не превосходил текущего уровня целостности субъект – сессии *x* обладала доступами на чтение или на запись ко всем сущностям параметрически ассоциированным с *y* , а когда уровень целости у выше минимального *i_low*, то субъект сессия *x '* имела доступ на запись к сущности *i_entity* .

Де юре правило вида *set_entity _ labels* (*x , x , y , yi*) до субъект сессии *x* задать сущности *y* с прямой меткой (не являющиейся параметрически ассоциированной ни с одной ролью или записью пользователей , и к которой на момент применения по ни одна субъект сессия системы не имеет доступов) её новый урок целостности , для чего требуется наличие у *x* административного ступа на чтение к административной роли *entities _ admin _ role* . При этом текущие и устанавливаемые уровни целостности сущности и не должны превосходить текущего уровня целостности субъект сессии *x* . Кроме того , устанавливаемый уровень целостности сущности у не должен превосходить уровней целостности всех сущностей контейнеров , в которых непосредственно находится у , и , наоборот , должен быть не ниже уровней целостности всех сущностей , непосредственно входящих в у , а также не должен превосходить уровней целостности всех ролей , запрещающих ролей или административных ролей , обладающих к ней правами доступа на владение или на запись . Когда у всех сущностей , подчинённых сущности у , метки косвенные , и к ним не имеют доступов субъект сессии , то их уровни целостности устанавливаются равными новому значению уровня целостности сущности у . Если либо текущий , либо устанавливаемый уровень целостности сущности у является уровнем целостности больше минимального *i_low* , то требуется наличие у кооперирующей субъект сессии доступа на запись к сущности *i_entity .*

Де факто правила преобразования состояний (табл . 2 . 3) вводятся на уровне мандатного контроля целостности МРОСЛ ДП модели так как на её уровне ролевого управления доступом не рассматриваются информационные потоки и фактическое владение .

На текущем уровне модели заданы 9 де факто правил преобразования состоянии , условия и результаты применения которая ответствуют предположениям модели . Эти правила не требуют реализации в ОССН (их применение не приводит к изменения параметров) , а предназначены для теоретического анализа ее безопасности в рамках МРОСЛ ДП модели . Некоторые де факто е могут инициироваться только недоверенными субъект-сессиями, которыми являются все субъект-сессии с уровнем целостности меньшим максимального i_high.

Де-факто правила преобразования состояний увроня мандатного контроля целостности МРОСЛ ДП-модели



Исходное состояние G	Результирующее состояние G'
$de\_facto\_op(s, op(x, x', ...))$	
$s \in N_S$, $x, x' \in de\_facto\_own(s)$, выполняются условия применения де-юре правила преобразования состояний $op(x, x', \dots)$, кроме правил: *create_user*(x, x', u, ui, ue), *set_user_labels*(x, x', u, ui, ue), *delete_user*(x, x', u), *set_entity_owner*(x, x', r, r', y), *set_subject_owner*(x, x', r, r', y), *grant_admin_rights*$(x, x', ar, \{(r, \alpha_{rj}): 1 \leqslant j \leqslant k\})$, *remove_admin_rights*$(x, x', ar, \{(r, \alpha_{rj}): 1 \leqslant j \leqslant k\})$, *add_negative_role*$(x, x', r, nr)$, *remove_negative_role*$(x, x', r, nr)$, *create_role*$(x, x', r, ri, re, name, rz)$, *delete_role*$(x, x', r, rz)$, *create_hard_link_role*(x, x', r, rz), *delete_hard_link_role*(x, x', r, rz), *rename_role*$(x, x', ry, name)$, *set_container_attr*$(x, x', y, ccri, t)$, *set_entity_labels*(x, x', y, yi), *set_role_labels*(x, x', r, re)	Соответствуют результатам применения де-юре правила $op(x, x', \dots)$
$control(x, y, z)$	
$x \in N_S, y \in S, z \in E, x \neq y, z \in [y]$ и $(x, z, write_m) \in F$	$de\_facto\_own'(x) = de\_facto\_own(x) \cup \{y\}$
$know(x, y)$	
$x \in N_S, y \in S, x \neq y, (x, y, write_m) \in F$, $]y[\neq \varnothing$, и для каждой $e \in]y[$ существует $(e, x, write_m) \in F$	$de\_facto\_own'(x) = de\_facto\_own(x) \cup \{y\}$
$take\_access\_own(x, y, z)$	
$x \in N_S, y, z \in S, y \in de\_facto\_own(x)$, $[z \in de\_facto\_own(y)]$ или $[i_s(z) \leqslant i_s(y)$, существует $r \in R \cup AR$: $(y, r, read_a) \in AA$, $(z, own_r) \in PA(r)$, и не существует $nr \in NR$: $(y, nr, read_a) \in AA$, $(z, own_r) \in PA(nr)]$	$de\_facto\_own'(x) = de\_facto\_own(x) \cup \{z\}$
$flow\_memory\_access(x, y, \alpha_a)$	
$x \in S, y \in E \setminus E\_HOLE$, $(y, \alpha_a) \in de\_facto\_accesses(x)$, где $\alpha_a \in \{read_a, write_a\}$	если $\alpha_a = read_a$, то $F' = F \cup \{(y, x, write_m)\}$; если $\alpha_a = write_a$, то $F' = F \cup \{(x, y, write_m)\}$
$take\_flow(x, y)$	
$x \in N_S, y \in S, x \neq y, y \in de\_facto\_own(x)$	$F' = F \cup \{(x, e, write_m): (y, e, write_m) \in F, e \in E \cup S\}$
$find(x, y, z)$	
$x, y \in S, z \in (E \setminus E\_HOLE) \cup S, x \neq z, (x, y, write_m), (y, z, write_m) \in F$	$F' = F \cup \{(x, z, write_m)\}$
$post(x, y, z)$	
$x, z \in S, y \in (E \setminus E\_HOLE), x \neq z, (x, y, write_m) \in F$ $(y, read_a) \in de\_facto\_accesses(z)$	$F' = F \cup \{(x, z, write_m)$
$pass(x, y, z)$	
$y \in S, x \in (E \setminus E\_HOLE), z \in (E \setminus E\_HOLE) \cup S, x \neq z,$ $(x, read_a) \in de\_facto\_accesses(y), (y, z, write_m) \in F$	$F' = F \cup \{(x, z, write_m)\}$

Таблица 2.3

Де факто правило вида *de_facto op (s , op(x,x'…))*позволяет недоверенной субъект сессии S , де факто владеющей субъект-сессиями x и x' , выполнить от имени x где де- юре правило. Данное правило позволяет исключить из рассмотрения де факто возможности субъект-сессии а также описать условия и результаты применения де юре правил преобразования состояний в виде , адаптированном реализации в ОССН . Перечисленные в условии при де юре правила , которым не может являться орт обладания субъект сессией с текущими административные с высоким уровнем целостности (т . е . высокого те целостности субъект

сессии x) , что означает преодоление : ной субъект сессией в защиты системы и бессмысленность дальнейшего анализа её безопасности.

Де факто правила видов *control (x , y , z)* и *know (x , y)* позволяют недоверенной субъект сессии x получить де факто владение субъект сессией y . При использовании первого правила требуется . чтобы нашлась сущность 2 , функционально ассоциированная с субъект сессией y , и существовал информационный поток по памяти y . При использовании второго правила требуется существование информационных потоков по памяти от x к z т от всех сущностей (как минимум , одной) , параметрически ассоциированных с субъект сессией y . Кроме того , в отличие от предшествующих ДП моделей добавлено условие наличия информационного потока по памяти от что соответствует наличию « канала управления » субъект попадающей под контроль (де факто владение) субъект (в случае с правилом вида *control (x , y , z)* описание так тельного канала не требуется , так как « управление » может осуществляться через сущность *z* , функционально ассоциированную с *y*).

Де-факто правило вида take_access_own(x,y,z) позволяет недоверенной субъект-сессии *x* получить де-факто владения субъект-сессией *z.*При этой требуется чтобы субъект-сессия *x* де-факто владела субъект-сессией *z* либо обладающей соответствующими ролями текущими уровнем целостности позволяющим ей использовать право доступа владения к субъект-сессии *z* у одной из своих текущих ролей, а также отсутствие *y* субъектов-сессии *y* текущей запрещающей роли, обладающей этим правом доступа к *z.*

Де-факто правило вида *flow_memory_access(x,y,a)* позволяет субъект-сессии x имеющей де-факто доступ на чтение или на запись к сущности *y*, не входящей во множество *E_HOLE*,реализовать информационный поток по памяти от *y к x или от x к y* соответственно.

Де-факто правило вида *take_flow(x,y)* позволяет недоверенной субъект-сессии x,фактически владеющей субъект-сессией *y*,реализовать информационные потоки по памяти ко всем сущностям или субъект-сессиям, к которым имеет информационные потоки по памяти субъект-сессия *y*.

Де-факто правила вида *find(x,y,z),post(x,y,z) u pass (x,y,z)* позволяют субъект-сессиям создавать информационные потоки по памяти, в основном аналогично соответствующим правилам, использованным в предыдущих ДП-моделях, а также в расширенной модели *Take-Grant[17,116*]. Отличия в условиях и результатах применения этих правил заключаются невозможности применения для создания информационных потоков по памяти сущностей из множества *E_HOLE*.

На уровне мандатного контроля целостности МРОСА ДП-модели можно доказать утверждение, аналогичное утверждению 2.1, о корректности задания де-юре и де-факто правил преобразования состоянии, т. е. о их соответствии условиям предположений 2.2 и 2.9.

2.3.5. Нарушение безопасности системы в смысле мандатного контроля целостности

При анализе условий нарушения безопасности системы на уровне мандатного контроля целостности МРОСА ДП-модели по аналогии с предшествующими ДП-моделями предполагается, что на траекториях функционирования системы доверенные субъект-сессии не кооперируют с недоверенными при администрировании учётных записей пользователей, сущностей, субъект-сессий, ролей, запрещающих ролей или административных ролей, получении доступов или административных доступов, изменении параметров, создании или Удалении сущностей, ролей, запрещающих ролей, административных ролей или «жёстких» ссылок на них, запрещающих ролей для Ролей или административных ролей, создании или удалении субъект-сессий. Кроме того, в ОССН создаются сущности (например, сокеты), через которые осуществляется безопасный обмен данными между доверенными и недоверенные субъект-сессиями, т. е интерфейсы взаимодействия через такие сущности тщательно проверяются (верифицируется и минимизируется код ПО, реализующего такие интерфейсы ,реализуются процедуры фильтрации передаваемых данных),и обосновывается невозможность их использования для получения недоверенным субъект-сессиям. В связи с этим дадим следующие определения.

Определение 2.8. Субъект-сессию *y,* обладающую уровне целостности выше минимального (*i_low<is(y))*, назовем функционально корректной относительно субъект-сессии *y* , обладающей уровнем целостности не выше уровня целостности первой субъект- сессии (*is(y')<<is(y)).* и сущности или субъект-сессии *e,* когда субъект-сессия y не реализует информационный поток по памяти от сущности или субъект-сессии e к некоторой сущности c , функционально ассоциированной с субъект-сессией *y'*. Субъект-сессию y назовём абсолютно функционально корректной относительно субъект-сессии *y'* и сущности или субъект-сессии e, когда субъект-сессия y не реализует информационный поток по памяти от сущности или субъект- сессии *e* к некоторой сущности *e'*, функционально ассоциированной с субъект-сессией *y'*. При этом используем обозначения:

f_correct{ s ∈ S |i_low<is(s))- функция, задающая для каждой субъект-сессии, обладающей уровнем целостности выше минимального, множество пар вида субъект-сессия и сущность или субъект-сессия, относительно которых она функционально корректна;

af_correct S- функция, задающая для каждой субъект-сессии множество пар вида субъект-сессия и сущность или субъект-сессия, относительно которых она абсолютно функционально корректна.

Определение 2.9. Субъект-сессию y,обладающую уровнем целостности выше минимального (*i_low<is(y))*, назовём параметрически корректной относительно субъект-сессии *y',* обладающей уровнем целостности не выше уровня целостности первой субъект- сессии

(is(y')<<is(y)) ,и сущности или субъект-сессии *e*, когда субъект-сессия у не реализует информационный поток по памяти к сущности или субъект-сессии е от некоторой сущности *e',* параметрически ассоциированной с субъект-сессией *y'*. Субъект-сессию у назовем абсолютно параметрически корректной относительно субъект-сессии у и сущности или субъект-сессии е, когда субъект-сессия *у* не реализует информационный поток по памяти к сущности ил» субъект-сессии е от некоторой сущности *e',* параметрически ассоциированной с субъект-сессией *y*'. При этом используем обозначения:

p_correct{s ∈ *S* |*i_low<is(s)}*-функция, задающая для каждой доверенной субъект-сессии множество пар вида доверенная субъект-сессия и сущность или субъект-сессия, относительно которых она праматрически корректна.

Ap_correct:S-функция, задающая для каждой субъект-сессии множество пар вида субъект-сессия и сущность или субъект-сессия, относительно косоротых она абсолютно параметрически корректна.

Определение 2 . 10 . Назовём траекторию *Go G1…Gn* системы Σ *(G' , OP , Go*) , где N>>0 , на которой довременные субъект сессии не инициируют выполнение де юре правил преобразования состояний , траекторией без кооперации доверенных и недоверенных субъект сессий . Таким образом , по определению в системе (G " , OP , G0) на траекториях без кооперации доверенных и недоверенных субъект сессий для передачи прав доступа доверенные субъект сессии могут инициировать выполнение только де факто правил *flow _ memory _ access* (*x , y , a*) , *find(x , y , z) , post (x , y , я) и pass (x , y , z) .*

По определению 2 . 10 на траекториях без кооперации доверенных и недоверенных субъект сессий могут инициировать выполнение любых де юре или де факто правил все недоверенные субъект сессии из множества Ns (имеющие уровень целостности меньше максимального *i_high*) .

На уровне мандатного контроля целостности МРОСЛ ДП модели целесообразно определить соответствующий смысл нарушения безопасности системы и дать определение безопасного состояния системы , так как в дальнейшем с учётом включения в последующие уровни модели мандатного управления доступом будут добавлены другие смыслы нарушения безопасности системы . При этом , поскольку после нарушения безопасности системы в одном из состоянии некоторой траектории дальнейший её анализ не имеет смысла , целесообразно рассматривать траектории , нарушение безопасности системы на которых возможно только в их конечном состоял также учесть , что захват контроля субъект сессией с меньшим уровнем целостности над субъект сессией не обязательно доверенной имеющей больший уровень целостности , следует рассматривать нарушение безопасности в смысле мандатного контроля целостности . Дадим определения.

Определение 2 . 11 . Начальное состояние Go системы Σ (G', OP , Go) назовём безопасным , когда оно удовлетворяет следую условиям .

Условие 1. Для каждых субъект-сессий x,y $\in$ *S0* таких, что *y* $\in$ *de_facto_own(x),верно iso(y)<<iso(x).*

Условие 2.Для каждых недоваренной субъект-сессии x $\in$ *Nso,*субъект-сессии *y* $\in$ *So u* сущности *e* $\in$ *E* таких, что либо *(e* $\in$ *[y] u (x,e,write)* $\in$ *Fo),*либо *(e* $\in$ *]y[* и либо *(e,x,write)* $\in$ *Fo,либо (x,e,read)* $\in$ *Fo,),*либо *(x,e,read)* $\in$ *A0),*верно неравенство *iso(y)<<iso(x).*

Условие 3. Для каждой доверенной субъект-сессии y $\in$ *Lso* верно условие *(y,i_entity,write)* $\in$ *Ao.*

Условие 4. Для каждого информационного потока (x,y,write) $\in$ *Fo* справедливо *ixo(x)>>iyo(y),*где *ixo* и *iyo-* соответствующие функции *ico* или *iso.*

Определение 2.12. Пусть Go-безопасное начальное состояние системы Σ (G', OP , Go) и существует траектория без кооперации доверенных и недоверенных субъект-сессией *Go G1…Gn,*где N>>1.Будем говорить что в состоянии Gn произошло нарушение безопасности системы в смысле мандатного контроля целостности когда существуют недоверенная субъект-сессия x$\in$ *Nsn* и субъект-сессия y $\in$ *de_facto_ownx(x)* такие что не верно неравенство *is(y)<<is(x).*

Проанализируем условия определений 2.11 и 2.12. Выполнение заданных в определении 2.11 требований к начальному состоянию (кроме условий предположений 2.2 и 2.9) обеспечивает отсутствие в нем недоверенных субъект-сессий либо фактически владеющих субъект-сессиями, обладающими более высоким текущим уровнем целостности, либо имеющих возможность получения контроля над такими субъект-сессиями с использованием сущностей, параметрически или функционально с ними ассоциированными. Также в начальном состоянии предполагается отсутствие запрещённых информационных потоков по памяти от сущностей или субъект-сессией с низким уровнем целостности к сущностям или субъект-сессиям с более высоким или несравнимым уровнем целостности. В результате в начальном состоянии не выполняется условие определения 2.12-не происходит нарушение безопасности системы в смысле мандатного контроля целостности. Кроме того в этом состоянии отсутствуют доверенные субъект-сессии имеющие доступ на запись к специальной сущности i_entity,заданной в условии 8 предположения 2.9,что не позволяет недовереным субъект-сессиям сразу реализовывать попытку повышения своего уровня целостности. Однако допускается что в начальном состоянии доверенные субъект-сессии де-факто владеют другими субъект-сессиями.

Как правило, в ОССН захват контроля (получение фактического владения) над процессами (субъект-сессиями) других учетных записей пользователей не является для нарушителя самоцелью. При реализации в ней мандатного управления доступом целью субъект-сессия

нарушителя, вероятно, будет создание соответствующих запрещённых информационных потоков «сверху вниз». Для чего эта субъект-сессия будет либо сразу пытаться реализовать данные запрещённые информационные потоки, либо сначала, «взломав» защиту' ОССН, получить фактическое владение субъект-сессией с 6олее высоким уровнем целостности или даже доверенной субъект-сессией (осуществить нарушение безопасности в смысле мандатного контроля целостности и, как правило, полный захват контроля над системой) и реализовать запрещённые информационные потоки с её использованием. При этом доверенные субъект-сессии не будут кооперировать с недоверенными, т. е. не будут реализовывать де- юре правила и ограничатся использованием только перечисленных в определении 2.10 четырёх де-факто правил с учётом корректности субъект-сессий в смыслах определений 2.8 и 2.9.

Достаточные условия безопасности системы в смысле мандатного контроля целостности (в соответствии с определением 2.12) при безопасном её начальном состоянии обосновываются в следующей теореме.

Теорема 2.1.Пусть Go- безопасное начальное состояние системы *Σ (G', OP , Go).* Пусть на всех траекториях системы без кооперации доверенных или недоверенных субъект-сессий *G0,G1,Gn,где N>>0*,и в каждом состоянии *Gn* для каждой субъект-сессии s $\in S$ и сущности e $\in E$ выполняются следующие условия.

Тогда на этих траекториях система *Σ (G', OP , Go)*безопасна смысле мандатного контроля целостности (в смысле определения 2.12).

Условия теоремы требуют от ОССН функционально и параметрически корректной реализации субъект-сессий и задания соответствующих уровней целостности функционально или параметрически ассоциированных с ними сущностей. В ней сформулированы дополнительные требования к де-юре правилам преобразования состояний системы, создающим субъект-сессии (при этом задаются сущности, параметрически или функционально с ними ассоциированные), создающим сущности, изменяющим права доступа к ним или их параметры, так как эти сущности могут оказаться функционально или параметрически ассоциированными с субъект-сессиями.

Говоря о разработке уровня мандатного контроля целостности иерархического представления МРОСЛ ДП-модели для СУБД *PostgreSQL*(см. рис. 2.1), целесообразно отметить следующее. Во- первых, потребовалось учесть существенные особенности реализации этого механизма управления доступом в СУБД. Например, чтобы не снижать производительность СУБД было предложено назначать уровни целостности элементам СУБД до схем включительно, т. е. не назначать их таблицам, функциям, триггерам. Во-вторых, определить значимые для реализации мандатного контроля целостности требования к использованию функций как элементов СУБД, включив в модель режимы их выполнения,

задаваемые в СУБД параметрами *SECURITY DEFINER* и SECURITY INVOKER (в модели с помощью функции *db_security*), Так как, очевидно, функция СУБД влияет на функциональность активизировавшей её субъект- сессии, то содержащая функцию сущность-схема СУБД была определена функционально ассоциированной с такой субъект-сессией, а уровень целостности этой сущности-схемы задан не ниже текущего уровня целостности субъект-сессии. При этом для режима выполнения функции СУБД с правами роли СУБД являющейся ее владельцем уровень целостности этой роли СУБД определён строго равным уровню целостности сущности-схемы содержащей функцию.

2.4 Уровни мандатного управления доступом

2.4.1 Элементы состоянии системы

Мандатное управление доступом с контролем информационных потоков по памяти и по времени требует достаточно большего описание соответствующих им уровней МРОСЛ ДП-модели что выходит за рамки настоящего учебного пособия. Поэтому рассмотрим только наиболее существенные из них элементы. Используем обозначения:

(LC,<<)- решетка многоуровневой безопасности уровней конфиденциальности (декартово произведение линейной шкалы уровней конфиденциальности данных и множества всех подмножеств конечного множества иерархических категорий данных).При этом заданы максимальный *c_high=+Lc* и минимальный *c_low=+LC* элементы решетки соответственно.

(fu,fe,fr,fs) $\in$ *FC*-четверка функций уровней конфиденциальности при этом:

Fu:U-LC-функция задающая для каждой учетной записи пользователя ее уровень доступа-максимальный разрешенный уровень субъект-сессии функционирующий от ее имени;

Fs:S-LC-функция задающая для каждой субъект-сессии ее текущий уровень доступа;

Fe:E-LC-функция задающая роль конфиденциальности для каждой сущности;

FC-множество всех четверок функций заданного вида;

CCR:E U R U NR U AR-{true,false}-функция задающая способ доступа к сущностям внутри контейнеров или ролям в иерархии ролей .Если сущность e $\in$ *C* является контейнером e,разрешен без учета уровня конфиденциальности контейнера e,то по определению выполняется равенство *CCR(e)=false*,в противном случае выполняется равенство *CCR(e)=true*.При этом по определению для каждой сущности объекта o $\in$ *O* выполняется равенство *CCR(o)=true*,для ролей, запрещающих ролей или административных ролей r $\in$ *R U NR U AR* всегда выполняется равенство *CCR(r)=false*.

Также во множество специальных административных ролей SAR добавлена обладаемая высоким уровнем целостности роль *downgrade_admin_role*,позволяющая доверенным субъект-сессиям нарушать требования мандатного управления доступом.

Мандатные атрибуты конфиденциальности сущностей-контейнеров CCR целесообразно реализовать в ОССН с целью обеспечения большей гибкости мандатного управления

доступом. Мандатные атрибуты конфиденциальности CCR для ролей использовав для единообразия с сущностями, так как при реализации ролевого управления доступом предполагается, что наличие возможное получения субъект-сессией некоторой роли всегда означает возможность получения этой субъект-сессией всех подчинённых данной роли ролей.

На уровне мандатного управления доступом с информационными потоками по памяти МРОСЛ ДП-модели в функции *execute_container* дополняется одно из её условии:

Условие 3. Существует ri $\in R\ U\ AR,$такая что *(s,r,read)* $\in AA$ и *(e,execute)* $\in PA(ri),$верно *[i(ei)*<< или *CCRI(ei)=false],[fe(ei)<<fs(s) или CCR(ei)=false],*где *1 <<i<<n.*

Определение 2.13.Пусть определены функции уровней конфиденциальности *(fu,fe,fr,fs*) и мандатных атрибутов конфиденциальности *CCR,(iu,ie,ir,is),CCRI,A,AA,F,Hr,He,Hs,constraint nr)*-состояние системы.

Rf={write,write}-множество видов информационных потоков (по памяти и по времени);

E_HOLE=RW_HOLE U W_HOLE-множество специальных объектов-дырок, где *RW_HOLE*-объекты дырки первого вида, которые по определению не могут быть использованы для создания информационных потоков,*W_HOLE*-объекты-дырки второго вида, которые по определению не могут быть использованы для создания информационных потоков по памяти,*RW_HOLE U W_HOLE=0;*

Хотя мандатное управление доступом в МРОСЛ ДП-модели по дуется без применения ролей, задание уровней конфиденциальности ролей необходимо для предотвращения использования прав доступа ролей для создания запрещённых информационных потопов по времени [15, 17]. Например, добавляя или удаляя права доступа к сущностям у некоторой роли, т. е. «записывая» в неё некоторые данные, кооперирующие субъект-сессии, находящиеся на различных уровнях доступа, могут осуществлять передачу данных из сущностей с высоким уровнем конфиденциальности к сущностям с низким уровнем конфиденциальности. В связи с этим дано следующее определение.

Определение 2.14. Доверенную субъект-сессию назовём корректной относительно информационных потоков по времени, когда она не участвует в их реализации. При этом используем обозначения:

LFs C LS- множество доверенных субъект-сессий корректных относительно информационных потоков по времени, *NFs c LS* — множество доверенных субъект-сессий некорректных относительно информационных потоков по времени, при этом по определению справедливы равенства *LF U NFs=;LFs U NFs=Ls.*

Всегда предполагается, что недоваренные субъект-сессии могут участвовать в реализации информационных потоков по времени (являются относительно них некорректными).

Обеспечить в ОССН корректность доверенных субъект-сессий относительно информационных потоков по времени не всегда возможно. Например, если доверенная субъект-сессия (коммуникационный процесс) всегда осуществляет передачу данных из одной сущности (входящего сокета) в другую (исходящий сокет), то она будет принимать участие в реализации информационных потоков по времени с использованием этих сущностей.

2.4.2. Условия корректности мандатного управления доступом и правила преобразования состояний

В системе на уровне мандатного управления доступом с информационными потоками по памяти МРОСЛ ДП-модели сформулированы условия, определяющие порядок:

•задания текущего уровня доступа субъект-сессии;

•задания уровня конфиденциальности сущности, учёта при этом вида её метки;

•задания уровня конфиденциальности роли, запрещающей роли или административной роли;

• задания уровней конфиденциальности параметрически ассоциированных с учетной записью пользователя или ролью сущностей.

• создания или удаление сущности, жёсткой ссылки на нее, получение доступа к ней изменения прав доступа ролей, запрещающих ролей или административных ролей к ней.

• создание или удаления роли ,запрещающие роли или административной роли ,получение доступа к ней, изменение прав доступа административных ролей к ней, изменение иерархии ролей администрирование функций необходимого обладания запрещающей ролью;

•создания, удаления субъект-сессии, изменения ее роли-владельца;

•задания уровней конфиденциальности мандатного атрибута конфиденциальности CCR сущностей ,ролей, запрещающих ролей и административных ролей;

В системе на уровне мандатного управления доступом с информационными потоками по времени МРОСЛ ДП-модели дополнительно сформулированы и скорректированы условия, определяющие порядок:

• доступа к специальным сущностям-объектам из множества *RW_HOLE;*

• создания или удаления сущности, «жёсткой» ссылки на неё, изменения прав доступа ролей, запрещающих ролей или административных ролей к ней, переименования сущности-контейнера;

•администрирования роли, запрещающей роли или административной роли, получения доступа к ней, использования ролей для создания информационных потоков, администрируют функции необходимого обладания запрещающей ролью, лучения её значения;

• задания индивидуальных административных ролей, индивидуальных ролей четной записи пользователя и общих ролей, их запрещающих ролей;

•получение доступов субъект-сессии к индивидуальным ролям и общим ролям их запрещающих ролям;

•получение параметров сущностей, субъект-сессий, ролей, запрещающих ролей или административных ролей;

•администрирование учётных записи пользователь ,получения их параметров;

•нарушение правил мандатным управления доступом с использованием специальной административной роли *downgrade_admin_role.*

Пример иерархии индивидуальных ролей учетных записей пользователей, заданных на уровне мандатного управления доступом с информационным потоками по времени МРОСЛ ДП-модели, для случая, когда LC={0,1}x2(решетка уровней конфиденциальности их двух элементов линейной шкалы и двух неиерархических категорий, задаваемых маской из двух битов),приведёт на рис. 2.2.

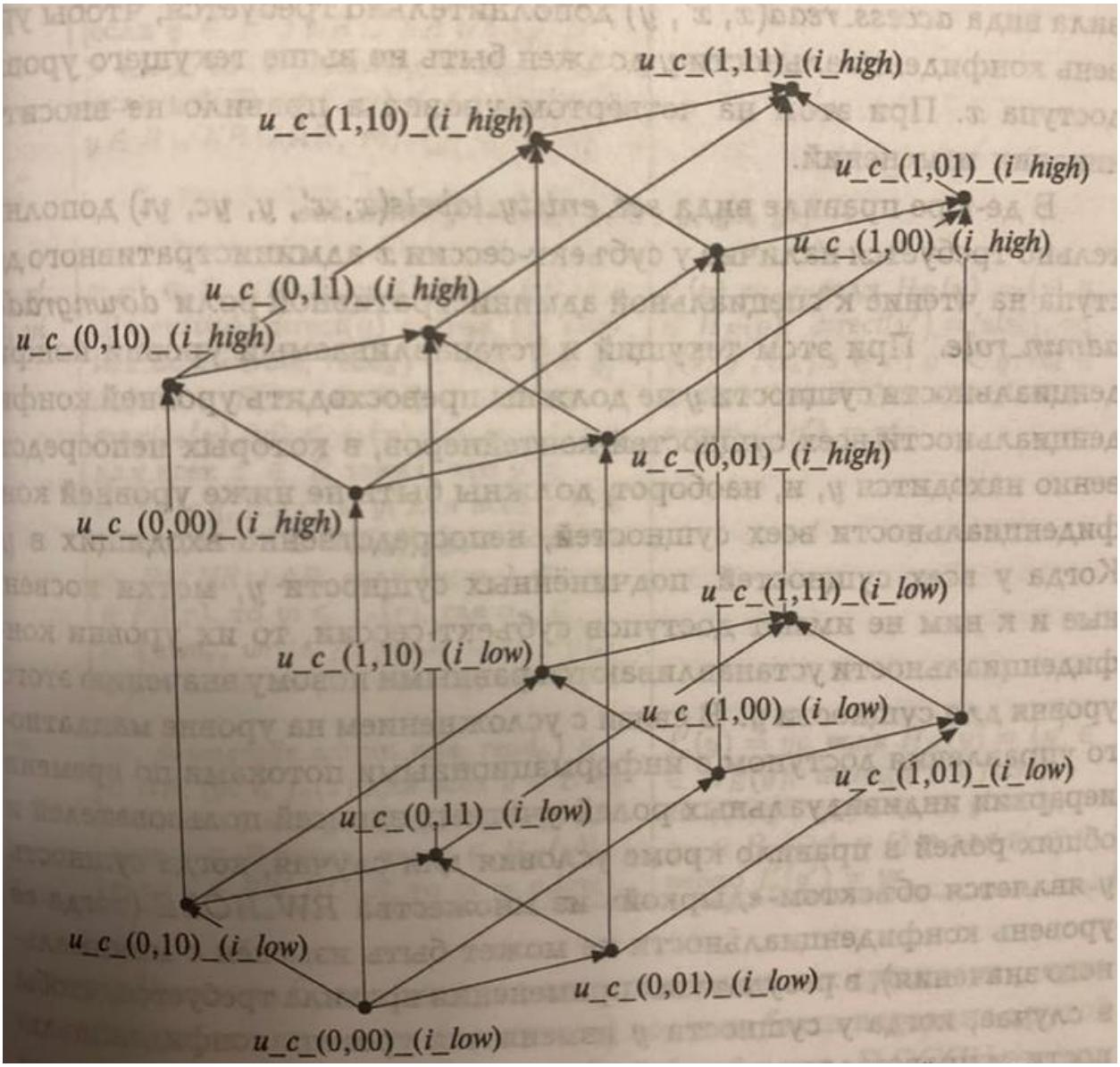
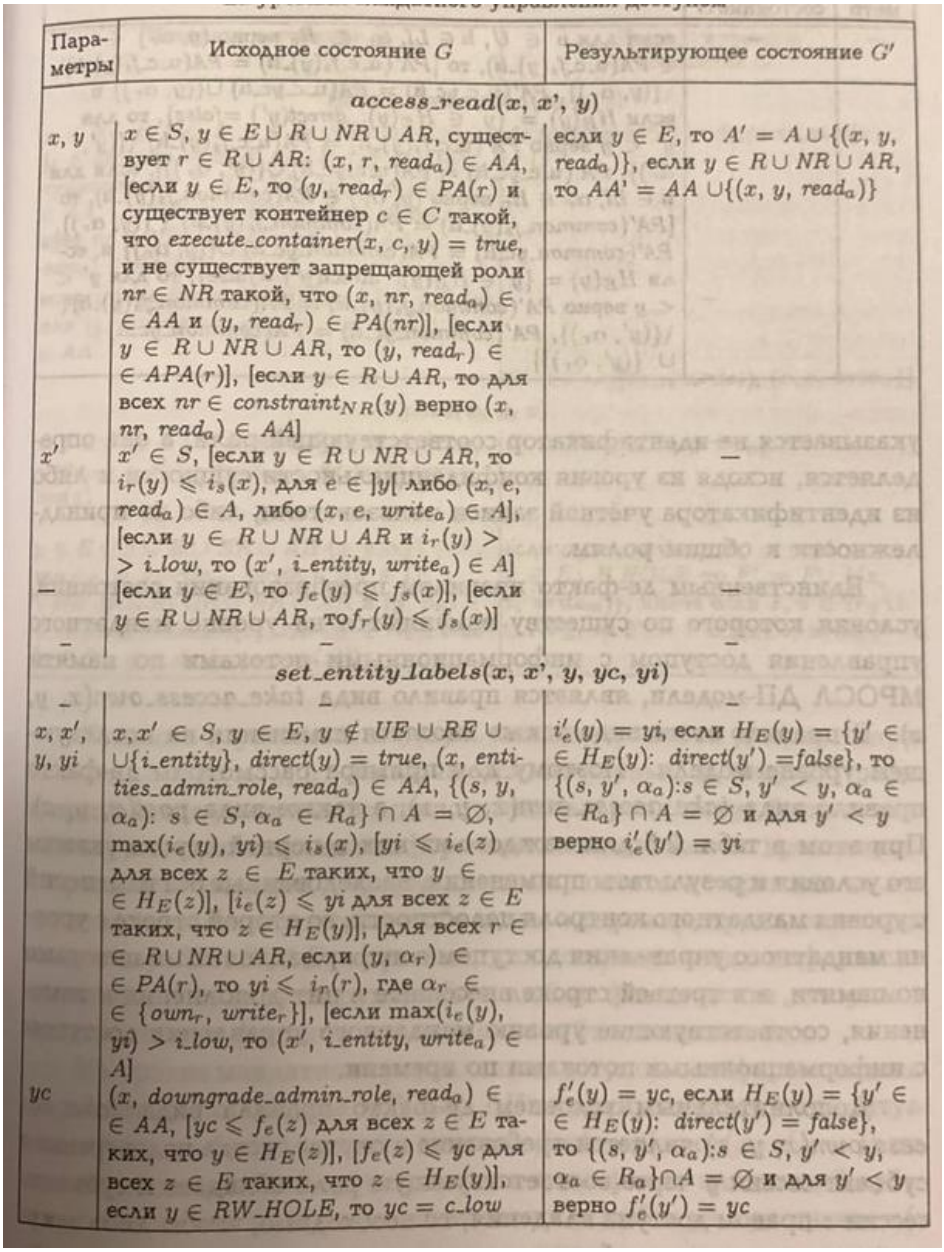


Рис.2.2. Пример иерархии индивидуальных ролей учетной записи пользователя

На уровнях мандатного управления доступом МРОСЛ ДП-модели не задаются новые де-юре правила преобразования состояний (их, как на предыдущем уровне, 38). Вместе с тем условия и результаты их применения существенно дополнены. Для примера рас. смотрим те же, что и на уровне мандатного контроля целостности де-юре правила видов *acces_read(x,x,y) u set_entity_labels(x,x',y,yi)*.При этом для каждого правила в табл. 2.4 в первой строке укажем его параметры, условия и результаты применения, наследованные без изменений с уровня ролевого управления доступом, во второй строке — с уровня мандатного контроля целостности, в третьей строке — добавленные на уровне мандатного управления доступом с информационными потоками по памяти, а в четвёртой строке — внесённые в них дополнения и изменения, соответствующие уровню мандатного управления доступом с информационными потоками по времени.

При получении доступа на чтение с использованием де-юре правила вида *acces_read(x,x',y)* дополнительно требуется, чтобы уровень конфиденциальности *y* должен быть не выше текущего уровня доступа *x*. При этом на четвёртом уровне в правило не вносится никаких изменений.

В де-юре правиле вида *set_entity_labels(x, x', y, yc, yi*) дополнительно требуется наличие у субъект-сессии *x* административного доступа на чтение к специальной административной роли *downgrade,admin_role*. При этом текущий и устанавливаемый уровни конфи-денциальности сущности у не должны превосходить уровней конфиденциальности всех сущностей-контейнеров, в которых непосредственно находится у, и, наоборот, должны быть не ниже уровней конфиденциальности всех сущностей, непосредственно входящих в у Когда у всех сущностей, подчинённых сущности у, метки косвенные и к ним не имеют доступов субъект-сессии, то их уровни конфиденциальности устанавливаются равными новому значению этого уровня для сущности у. В связи с усложнением на уровне мандатного управления доступом с информационными потоками по временя иерархии индивидуальных ролей учётных записей пользователей и общих ролей в правило кроме условия для случая, когда сущность *y* является объектом-«дыркой» из множества *RW_HOLE* (тогда ее уровень конфиденциальности не может быть изменён с минимального значения), в результатах применения правила требуется, чтобы в случае, когда у сущности у изменяется уровень конфиденциальности, и правом доступа к ней обладает какая-либо индивидуальная роль учётной записи пользователя или общая роль, то это право доступа(а также права доступа всех подчиненных сущности у сущностей, если все их метки косвенные)должно быть передано другой соответствующей индивидуальной или общей роли. В ОССН это может быть реализовано автоматически в случае, когда для сущности



Параметры	Исходное состояние G	Результирующее состояние G'
	***access_read*(x, x', y)**	
x, y	$x \in S$, $y \in E \cup R \cup NR \cup AR$, существует $r \in R \cup AR$: $(x, r, read_a) \in AA$, [если $y \in E$, то $(y, read_r) \in PA(r)$ и существует контейнер $c \in C$ такой, что $execute\_container(x, c, y) = true$, и не существует запрещающей роли $nr \in NR$ такой, что $(x, nr, read_a) \in AA$ и $(y, read_r) \in PA(nr)$], [если $y \in R \cup NR \cup AR$, то $(y, read_r) \in APA(r)$], [если $y \in R \cup AR$, то для всех $nr \in constraint_{NR}(y)$ верно $(x, nr, read_a) \in AA$]	если $y \in E$, то $A' = A \cup \{(x, y, read_a)\}$, если $y \in R \cup NR \cup AR$, то $AA' = AA \cup \{(x, y, read_a)\}$
x'	$x' \in S$, [если $y \in R \cup NR \cup AR$, то $i_r(y) \leqslant i_s(x)$, для $e \in]y[$ либо $(x, e, read_a) \in A$, либо $(x, e, write_a) \in A$], [если $y \in R \cup NR \cup AR$ и $i_r(y) > i\_low$, то $(x', i\_entity, write_a) \in A$]	—
—	[если $y \in E$, то $f_e(y) \leqslant f_s(x)$], [если $y \in R \cup NR \cup AR$, то $f_r(y) \leqslant f_s(x)$]	—
–	–	–
	***set_entity_labels*(x, x', y, yc, yi)**	
–	–	–
x, x', y, yi	$x, x' \in S$, $y \in E$, $y \notin UE \cup RE \cup \cup \{i\_entity\}$, $direct(y) = true$, $(x, entities\_admin\_role, read_a) \in AA$, $\{(s, y, \alpha_a): s \in S, \alpha_a \in R_a\} \cap A = \varnothing$, $\max(i_e(y), yi) \leqslant i_s(x)$, $[yi \leqslant i_e(z)$ для всех $z \in E$ таких, что $y \in H_E(z)]$, $[i_e(z) \leqslant yi$ для всех $z \in E$ таких, что $z \in H_E(y)]$, [для всех $r \in R \cup NR \cup AR$, если $(y, \alpha_r) \in PA(r)$, то $yi \leqslant i_r(r)$, где $\alpha_r \in \{own_r, write_r\}$], [если $\max(i_e(y), yi) > i\_low$, то $(x', i\_entity, write_a) \in A$]	$i'_e(y) = yi$, если $H_E(y) = \{y' \in H_E(y)$: $direct(y') = false\}$, то $\{(s, y', \alpha_a): s \in S, y' < y, \alpha_a \in R_a\} \cap A = \varnothing$ и для $y' < y$ верно $i'_e(y') = yi$
yc	$(x, downgrade\_admin\_role, read_a) \in AA$, $[yc \leqslant f_e(z)$ для всех $z \in E$ таких, что $y \in H_E(z)]$, $[f_e(z) \leqslant yc$ для всех $z \in E$ таких, что $z \in H_E(y)]$, если $y \in RW\_HOLE$, то $yc = c\_low$	$f'_e(y) = yc$, если $H_E(y) = \{y' \in H_E(y)$: $direct(y') = false\}$, то $\{(s, y', \alpha_a): s \in S, y' < y, \alpha_a \in R_a\} \cap A = \varnothing$ и для $y' < y$ верно $f'_e(y') = yc$

Пример де-юре правил преобразования состояний МРОСЛ ДП-модели на уровнях мандатного управления доступом

Таблица 2.4

Параметр	Исходное состояние G	Результирующее состояние G'
—	—	если для $u \in U$, $li \in LI$, $\alpha_r \in R_r$ верно $(y, \alpha_r) \in PA(u\_c\_f_s(y)\_li)$, то $[PA'(u\_c\_f_s(y)\_li) = PA(u\_c\_f_s(y)\_li) \setminus \{(y, \alpha_r)\}$, $PA'(u\_c\_yc\_li) = PA(u\_c\_yc\_li) \cup \{(y, \alpha_r)\}$ и, если $H_E(y) = \{y' \in H_E(y)$: $direct(y') = false\}$, то для $y' < y$ верно $PA'(u\_c\_f_s(y)\_li) = PA(u\_c\_f_s(y)\_li) \setminus \{(y', \alpha_r)\}$, $PA'(u\_c\_yc\_li) = PA(u\_c\_yc\_li) \cup \{(y', \alpha_r)\}]$, если для $li \in LI$, $\alpha_r \in R_r$ верно $(y, \alpha_r) \in PA(common\_f_s(y)\_li)$, то $[PA'(common\_f_s(y)\_li) = PA(common\_f_s(y)\_li) \setminus \{(y, \alpha_r)\}$, $PA'(common\_yc\_li) = PA(common\_yc\_li) \cup \{(y, \alpha_r)\}$ и, если $H_E(y) = \{y' \in H_E(y)$: $direct(y') = false\}$, то для $y' < y$ верно $PA'(common\_f_s(y)\_li) = PA(common\_f_s(y)\_li) \setminus \{(y', \alpha_r)\}$, $PA'(common\_yc\_li) = PA(common\_yc\_li) \cup \{(y', \alpha_r)\}]$

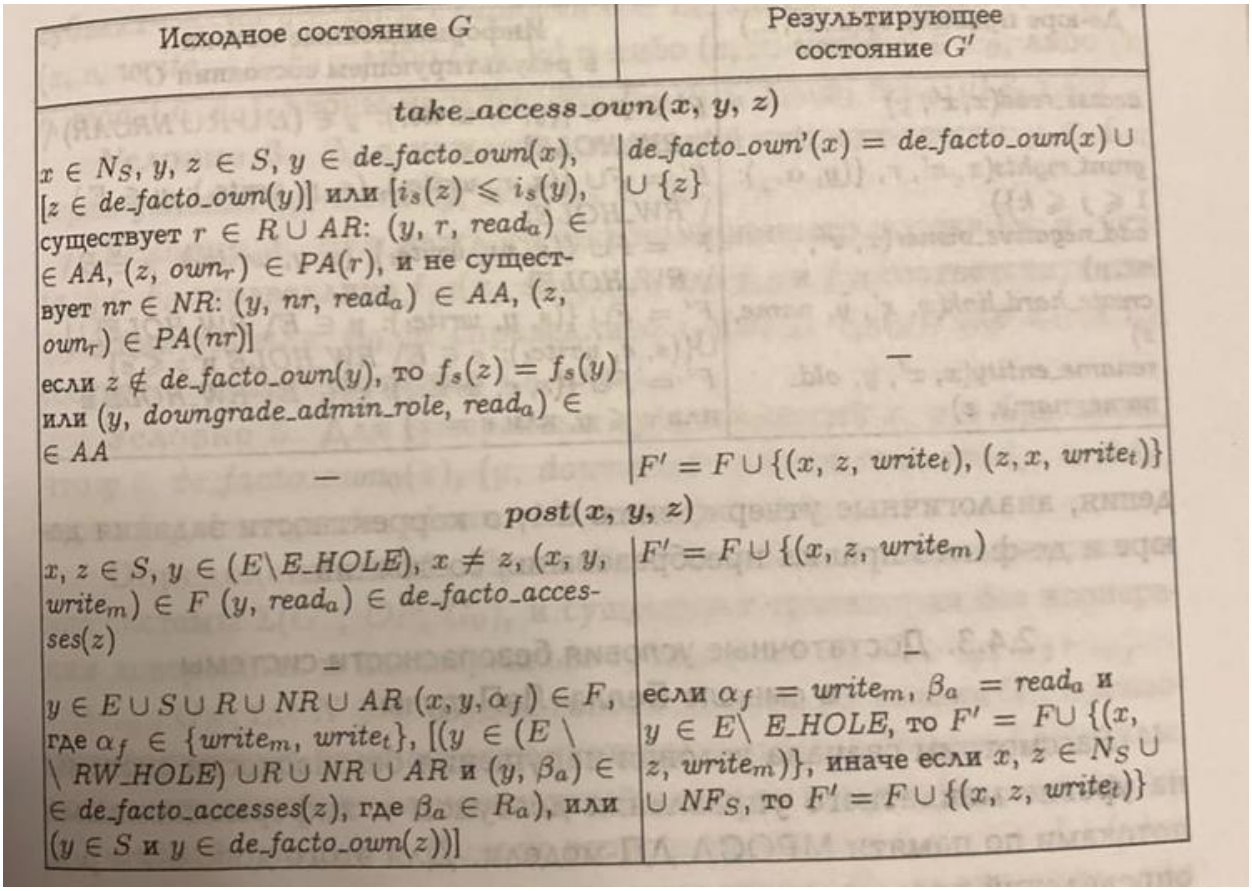
Окончание **табл. 2.4**

указывается не идентификатор соответствующей роли, а она определяется, исходя из уровня конфиденциальности сущности и либо из идентификатора учётной записи пользователя, либо из принадлежности к общим ролям.

Единственным де-факто правилом преобразования состояний, условия которого по существу изменяются на уровне мандатного V управления доступом с информационными потоками по памяти МРОСЛ ДП-модели, является правило вида *take_access_own*(*x, y, z*). В правило этого вида также вносятся изменения на последующем уровне модели. Поэтому для примера рассмотрим де-факто правило вида *take_access_own*(*x, y, z*), а также вида *post(x,y,z)*. При этом в табл. 2.5 для каждого из них в первой строке укажем его условия и результаты применения, наследованные без изменении с уровня мандатного контроля целостности, во второй строке с уровня мандатного управления доступом с информационными потоками по памяти, а в третьей строке внесённые в них дополнения и изменения, соответствующие уровню мандатного управления доступом с информационными потоками по времени.

Дополнительным условием де-факто правила вида *take_access_own*(*x, y, z*)является требование в случае, когда недоверенная субъект-сессия у через соответствующую роль обладает к субъект- сессии г правом доступа владения, то необходимо, чтобы либо текущий уровень доступа у был равен текущему уровню доступа г, либо наличие у субъект-сессии у административного доступа на чтение к административной роли *downgrade_admin_role*. А на

последующем указывается, что создается информационных потоки по времени между субъект-сессиями *x* и *z* .



Исходное состояние G	Результирующее состояние G'
$take\_access\_own(x, y, z)$	
$x \in N_S$, $y, z \in S$, $y \in de\_facto\_own(x)$, [$z \in de\_facto\_own(y)$] или [$i_s(z) \leqslant i_s(y)$, существует $r \in R \cup AR$: $(y, r, read_a) \in AA$, $(z, own_r) \in PA(r)$, и не существует $nr \in NR$: $(y, nr, read_a) \in AA$, $(z, own_r) \in PA(nr)$] если $z \notin de\_facto\_own(y)$, то $f_s(z) = f_s(y)$ или $(y, downgrade\_admin\_role, read_a) \in AA$	$de\_facto\_own'(x) = de\_facto\_own(x) \cup \cup \{z\}$
—	$F' = F \cup \{(x, z, write_t), (z, x, write_t)\}$
$post(x, y, z)$	
$x, z \in S$, $y \in (E \setminus E\_HOLE)$, $x \neq z$, $(x, y, write_m) \in F$ $(y, read_a) \in de\_facto\_accesses(z)$	$F' = F \cup \{(x, z, write_m)$
—	—
$y \in E \cup S \cup R \cup NR \cup AR$ $(x, y, \alpha_f) \in F$, где $\alpha_f \in \{write_m, write_t\}$, [$(y \in (E \setminus \setminus RW\_HOLE) \cup R \cup NR \cup AR$ и $(y, \beta_a) \in \in de\_facto\_accesses(z)$, где $\beta_a \in R_a$), или $(y \in S$ и $y \in de\_facto\_own(z))$]	если $\alpha_f = write_m$, $\beta_a = read_a$ и $y \in E \setminus E\_HOLE$, то $F' = F \cup \{(x, z, write_m)\}$, иначе если $x, z \in N_S \cup \cup NF_S$, то $F' = F \cup \{(x, z, write_t)\}$

Примеры де-факто правил преобразования состояний МРОСЛ ДП-модели на уровнях мандатного управления доступом

Таблица 2.5

Отличия в условиях и результатах применения де-факто правила вида past(x,y,z) заключается, во-первых, в том, что роли, запрещающие роли или административные роли могут быть источниками или приемниками информационных потоков по времени, во-вторых, для создания информационных потоков не могут использоваться доступы объектам-дыркам принадлежащим множеству RW_HOLE,в-третьих для создания таких информационных потоков могут примениться любые фактические доступы к сущностям или фактическое владение субъект-сессиями.

Так же, как на предыдущих уровнях, на уровнях мандатного управления доступом МРОСЛ ДП-модели можно доказать утверждение, аналогично утверждению 2.1,о корректности задания де-юре и де-факто правил преобразования состояний.

Де-юре правило $op(x, x', \ldots)$	Информационные потоки в результирующем состоянии G'
$access\_read(x, x', y)$	$F' = F \cup \{(y, s, write_t)$: $y \in (E \cup R \cup NR \cup AR) \setminus RW\_HOLE\}$
$grant\_rights(x, x', r, \{(y, \alpha_{rj})$: $1 \leqslant j \leqslant k\})$	$F' = F \cup \{(s, r, write_t), (s, y, write_t)$: $y \in E \setminus RW\_HOLE\}$
$add\_negative\_owner(x, x', nr, y)$	$F' = F \cup \{(s, nr, write_t), (s, y, write_t)$: $y \in E \setminus RW\_HOLE\}$
$create\_hard\_link(x, x', y, name, z)$	$F' = F \cup \{(s, y, write_t)$: $y \in E \setminus RW\_HOLE\} \cup \cup \{(s, e, write_t)$: $e \in E \setminus RW\_HOLE$ и $z \leqslant e\}$
$rename\_entity(x, x', y, old\_name, name, z)$	$F' = F \cup \{(s, e, write_t)$: $e \in E \setminus RW\_HOLE$ и или $e \leqslant y$, или $e = z\}$

Примеры информационных потоков по времени при применении де-факто правил *de_facto_op(s,op(x,x',…)*

Таблица 2.6

2.4.3 Достаточные условия безопасности системы в смысле Белла-ЛаПадулы

Рассмотрим сначала условия нарушения безопасности системы на уровне мандатного управления доступом с информационными потоками по памяти МРОСЛ ДП-модели .Для этого заполним ряд определение предыдущего уровня и определим нарушение безопасности системы во втором смысле(в смысле Белла-ЛаПадулы).

Определение 2.15 Назовём траекторию системы на которой доверенные субъекты-сессии не инициирует выполнение де-юре правил преобразование состояние, доверенные субъекты-сессии , обладающий текущей административной роли ,не инициирует выполнение де-юра и де-факто правил, траектории и без кооперации доверенных и не доверенных субъект-сессий. Таким образом по определению в системе на траекториях без кооперации доверенных и недоверенных субъектов цессии для передачи прав доступа доверенный субъект-сессии, не обладающие текущей административной ролью, могут инициировать выполнение только де-факто правил.

Определение 2.16. Начальное состояние Go системы (G',OP,Go)назовем безопасным, когда оно удовлетворяет следующим условиям.

Условие 1. Для каждых субъект-сессий x,y *Σ So* таких что y *Σ de_facto_own(x),*верны неравенства fs(y)=fs(x) и iso(y)<<iso(x).

Условие 2.Для каждых не доверенной субъект-сессии x $\in$ E0,таких что либо (e$\in$ *[y] u (x,e,write)*

Условие 3. Для каждой доверенной субъект-сессии *y* $\in$ *Lso* верное условие *(y,i_entity,write) A0.*

Определение 2.17. Пусть Go безопасное начальное состояние системы *Σ (G', OP , Go).*, и существует траектория без кооперации доверенных и недоверенных субъект-сессий

G0,G1..Gn,где N>>1. Будем говорить, что в состоянии Gn произошло нарушение безопасности системы в смысле Белла-ЛаПадулы, когда существует информационный поток по памяти (x, y, write) € Fn такой, что x, y € En и не верно неравенство fn(x)<<fn(y) и это условие не выполняется в состояниях Gi, траектории, где 0 <<i< N. Назовём систему *Σ (G', OP , Go)*., безопасной в смысле Белла-ЛаПадулы, когда в ней невозможно соответствующее нарушение безопасности.

Проанализируем условия определения 2.16 и 2.17. Выполнение заданных в первом определении дополнительных требований к начальному состоянию обеспечивает отсутствие в нем недоверенных субъект-сессий, фактически владеющих субъект-сессиями с использованием получения контроля над такими субъект-сессиями с использованием сущностей,параметрически или функционально с ними ассоциированными, либо имеющих возможность с использованием специальной административной роли downgrade_admin_role нарушать правила мандатного управления доступом. Также в начальном состоянии не должно быть запрещенных информационных потоков по памяти от сущностей или субъект-сессиями с низким уровнем конфиденциальности или текущим уровнем доступа. Отсутствие таких потоков по памяти является основополагающими требованием к ОССН , реализующей мандатное управление доступом (соответствие этому требованию часто называют ее безопасно^ в смысле Белла-ЛаПадулы). Появление таких потоков является, нарушением безопасности в смысле определения 2.17, при ли, запрещающие роли и административные роли в соответствии с результатами применения де-факто правил преобразования состояний не могут являться источниками или приемниками информационного потоков по памяти. В результате безопасное по определению 2.16 начальное состоянии системы является безопасным в смысле Белла-ЛаПадулы (в нем не происходит нарушение безопасности в указанном смысле).

Можно доказать теорему о том, что при безопасном начальной состоянии системы достаточным условием безопасности системы в смысле Белла-ЛаПадулы (не происходит нарушения безопасности в смысле определения 2.17) является контроль над сущностями, функционально или параметрически ассоциированными с субъект-сессиями, а также безопасность системы в смысле мандатного контроля целостности (не происходит нарушения безопасности в смысле определения 2.12), достаточные условия которой, обоснованные в теореме 2.1, остаются верными на рассматриваемом уровне модели.

Теорема 2.2 (**базовая теорема безопасности — БТБ-ДП**). Пусть Go — безопасное начальное состояние системы *Σ (G', OP , Go*).Пусть на всех траекториях системы без кооперации доверенных или недоверенных субъект-сессий G',Op,Go, где N>> 0, и в каждом состоянии Gn для каждой субъект-сессии s *∈ Sn* и сущности e *∈* En выполняются следующие условия.

Условие 1. Если e $\in$ *[s],* то выполняются условия *i(s)<<i(e), u (f(s)=f(e).*

Условие 2. Если e $\in$ *[s],*то верно *fn(s)=fn(e)=in(s)<<in(e)* и для каждой роли или административной роли *r* $\in$ *Rn U Ar.*

Условие 3. Для всех субъект-сессий s $\in$ *Sn* таких что *i_low<in(s),*выполняется равенства *f_correct (s)=Sn x (En U Sn)* (это условие является более строгим чем аналогичное условие в теореме 2.1, так как в нем для противодействия запрещенным информационном потокам по памяти требуется функциональная корректность субъект-сессии с текущим уровнем целостности выше минимального относительно всех.)

Условие 4. Для всех субъект-сессий s $\in$ *Sn* выполняется равенства *{s'∈ Sn:fn(s')=fn(s)}.*

Тогда на этих траекториях система *Σ (G', OP , Go*) безопасна в смысле мандатного контроля целостности и Белла-ЛаПадулы (не происходит нарушения безопасности системы в смыслах определений 2.12. и 2.17).

Условия теоремы 2.2 требуют от ОССН функционально и пара метрически корректной (абсолютно корректной) реализации субъект-сессий и задания соответствующих уровней конфиденциальности и целостности функционально или параметрически ассоциированных с ними сущностей. Если, например, субъект-сессия, имеющая высокий текущий уровень доступа, некорректно обрабатывает данные («заражается») в сущностях с низким уровнем конфиденциальности, и это приводит к получению фактического владения над нею субъект-сессией с низким текущим уровнем доступа, то система защиты ОССН не всегда сможет этому воспрепятствовать. Кроме того, условие 4 требует реализации в ОССН изоляции (невозможности создания информационных потоков по памяти с использованием функционально или параметрически ассоциированных сущностей) субъект-сессий (процессов) друг от друга, в том числе когда они обладают одинаковыми уровнями доступа. Таким образом, условия теоремы указывают на необходимость повышения качества разработки прикладного ПО для ОССН. В условиях 1, 2 также сформулированы дополнительные требования к де-юре правилам преобразования состояний системы, создающим субъект-сессии (при этом задаются сущности, параметрически или функционально с ними ассоциированные), создающим сущности, изменяющим права доступа к ним или их параметры, так как эти сущности могут оказаться функционально или параметрически ассоциированными с субъект сессиями.

2.4.4.Достаточные условия безопасности системы в смысле информационных потоков по времени

На уровне мандатного управления доступом с информационными потоками по времени МРОСЛ ДП-модели в определении безопасного состояния дополняется только одно из условий.

Определение 2.18. Начальное состояние Go системы *Σ (G', OP , Go*) назовем безопасным, когда оно удовлетворяет условиям 1-3,5 определения 2.16 и выполняется следующее условие.

Условие 4. Для каждого информационного потока (x,y,a) *∈ Fo,*где,*a ∈ {write m,write t}.*

Определим нарушение безопасности системы в третьем смысле(в смысле контроля информационных потоков по времени)

Определение 2.19. Пусть Go безопасное начальное состояние системы *Σ (G', OP , Go*),и существует траектория без кооперации доверенных и недоваренных субъект-сессий G',Op,Go, где N>> 1.Будем говорить, что в состоянии произошло нарушение безопасности системы в смысле контроля информационных потоков по времени, когда в нём существует информационный поток по времени (x,y,write) € Fn такой, что x, y € En и не верно неравенство fn(x)<<fn(y) и это условие не выполняется в состояниях G1 траектории, где 0 <<i< N. Назовём систему *Σ (G', OP , Go*), безопасной в смысле контроля информационных потоков л по времени, когда в ней невозможно соответствующее нарушение безопасности.

Это определение описывает случай реализации запрещённого информационного потока «сверху вниз» по времени между сущностями. Хотя роли или административные роли могут являться источниками или приёмниками информационных потоков по времени, для нарушения безопасности требуется создание именно информационного потока по времени между сущностями. Это объясняется тем, что в ОССН источником конфиденциальных данных являются только сущности и для вывода таких данных используются также сущности (например, файлы, устройства вывода).

Можно доказать теорему о том, что при безопасности системы в смысле мандатного контроля целостности и Белла-ЛаПадулы (в смыслах определений 2.12 и 2.17) и при безопасном её начальном состоянии (в смысле определения 2.18) достаточным условием ее безопасности в смысле контроля информационных потоков по времени (в смысле определения 2.19) является отсутствие в системе сущностей из множества W_HOLE, которые возможно использовать л • создания информационных потоков по времени, а также сущностей-контейнеров с мандатными атрибутами конфиденциальности и целостности равными true, в состав которых входят сущности с меньшим уровнем конфиденциальности.

Теорема 2.3. Пусть Go-безопасное начало состояние системы *Σ (G', OP , Go*),(в смысле определения 2.18.)Пусть на траектории системы без кооперации доверенных или недоверенных субъект-сессии G0,G1..Gn,где N>>1,каждое состояние Gi безопасно в смыслах определений 2.19.Тогда в состоянии Gn не происходит нарушения безопасности системы в смысле определения 2.1.9,когда выполняется следующие условия.

Условие 1.Не существует субъект-сессии x *∈ Nsm U NF* и сущности *y ∈ W_HOLE* таких что *(x,y,write) ∈Am.*

Условие 2. Не существует сущности-контейнера с $\in C$ и сущности $e \in Em$ таких что $CCRm(c)=CCRIm(c)=true$,*где* $1<<M<<N$.

Из теоремы следует, что для предотвращения запрещённых информационных потоков по времени «сверху-вниз» в ОССН достаточно обеспечения её безопасности в смыслах определений 2.12 и 2.17, исключения создания (особенно при установке или администрировании ОССН) сущностей-контейнеров с мандатными атрибутами конфиденциальности и целостности равными 1гие, в состав которых входят сущности с меньшим уровнем конфиденциальности, а также либо полный запрет на использование сущностей из множества *W_HOLE (*задание *W_HOLE=0*), либо такую реализацию этих сущностей, когда их нельзя будет применять для создания информационных потоков времени.

При разработке уровней мандатного управления доступом с контролем информационных потоков по памяти и по времени иерархического представления МРОСЛ ДП-модели для СУБД *PostgreSQL*(рис. 2.1) использовался подход, аналогичный применённому на соответствующем уровне мандатного контроля целостности СУБД *PostgreSQL* а именно, уровни конфиденциальности назначались элементам СУБД до схем включительно. Кроме того, для предотвращения запрещённых информационных потоков по времени, во-первых, потребовался запрет на использования механизма передачи роли СУБД права на дальнейшую передачу привилегии (использованием права *WITH GRANT OPTION),* во-вторых, отсутствие функций СУБД, содержащихся в сущностях-схем. имеющих высокий уровень целостности и уровень конфиденциальности выше минимального.

В заключение главы отметим, что научные подходы по переводу описания иерархического представления МРОСЛ ДП-модели с математического языка на формальный язык *Event-B*,дедуктивной верификации модели и верификации ее реализации посредственно в программном коде ОССН, кратко излагавшийся в предыдущем издании учебного пособия [4],теперь подробно рассмотрены в монографии [33].

Контрольные вопросы

1.Из каких уровней состоит иерархическое представление МРОСЛ ДП-модели, ориентированное на моделирование маринизма управления доступом в ОССН?

2.Как и почему менялась структура иерархическое представление МРОСЛ ДП-модели по мере ее разработки?

3.Как типичные для ОС семейства *Linux* с дискреционным управлением доступа три группы прав доступа (права владельца, группы владельца ,для всех остальных) к сущностям могут быть выкрадены с использованием ролей, заданных в рамках МРОСЛ ДП-модели?

4.Для моделирования каких важных для практики случаев в МРОСЛ ДП-модели заданы прямые и косвенные метки сущностей-контейнеров?

5. какие функции интерфейса механизма управления доступом ОССН, основанном на Linux Security Module (LSM), соответствуют де-юре правилам МРОСЛ ДП-модели?

6. Какие учетные записи пользователей или субъект-сессии (процессы) ООСН являются доверенными или недоверенными?

7. Какие сущности ООСН могут рассматриваться как функционально и параметрически ассоциированные с учетными записями пользователей или субъект-сессиями процессами?

8. Что в ООСН целесообразно считать фактическим владением недоверенной субъект-сессией доверенной субъект-сессии?

9. Для чего в МРОСЛ ДП-модели используются специальные административные роли из множества SAR? Какая из этих ролей самая «опасная»? Какие им могут быть найдены аналоги из реализованных в ООСН привилегий?

10. В чем отличие де-юре и де-факто правил преобразования состояний?

11. Какие ситуации в ООСН можно считать соответствующими каждому из де-факто правил?

12. Для чего используется специальная сущность i_entity?

13. Почему допускается несоответствия между уровнем конфедициальности и уровнями конфиденциальности сущностей , к которым она обладает правами доступа, и, наоборот, такое несоответствие не разрешается для уровни целостности?

14. Как зависит порядок доступа к сущности-контейнеру от значений её атрибута конфиденциальности CCR и целостности CCRI? Конфиденциальности и целостности ,соответствующим уровням этой учетной записи пользователя?

15. Какие параметры файловой системы ООСН могут быть потенциально использованы для создания запрещённых информационных потоков по времени «сверху-вниз»?

16. Какие сущности ООСН могут быть включены во множества RW_HOLE или W_HOLE?

17. Почему в МРОСЛ ДП-модели для сущностей , параметрически ассоциированных с учетной записью пользователя , требуется равенство их уровней

18.Обеспечение безопасности в каком из смыслов, указанных в определениях 2.12,2.17 и 2.19, для ОССН является основной ее безопасности в целом?

19.Как можно доработать условия и результаты применения де-юре и де-факто правил МРОСЛ ДП-модели, чтобы в них были учтены условия теорем 2.1,2.2 и 2.3?

20. Какие подходы используются при моделировании управления доступом в СУБД *PostgreSQL?*

21.Для чего МРОСЛ ДП-модель разрабатывается как единая модель механизма управления доступом для ОССН и СУБД *PostgreSQL?*

3 Управление безопасностью ОССН

3.1. Мандатное управление доступом

3.1.1. Проблемы реализации мандатного управления доступом в операционных системах.

В главе 1 были рассмотрены основные способы администрирования в ОССН, унаследованные от традиционного для ОС семейства *Linux* механизма дискреционного управления доступом. Как известно, данный механизм позволяет явно разрешать или запрещать те или иные доступы субъектов к сущностям, но не позволяет управлять информационными потоками, содержащими информацию различных уровней конфиденциальности. Например, после того как данные из некоторой сущности прочитаны процессом (субъект-сессией), он потенциально может записать их в любую другую сущность, доступную ему на запись, и механизм дискреционного управления доступом не сможет этому воспрепятствовать. Аналогично, если полномочия учётной записи пользователя, от имени которой выполняется процесс, позволяют воспользоваться электронной почтой, то прочитанные данные могут быть направлены на любой адрес сети Интернет. Таким образом, отсутствие при использовании механизма дискреционного управления доступом чётких понятных всем пользователям ОССН правил обработки конфиденциальной информации может являться причиной для её утечки.

По этой причине базовым при реализации защиты в ОССН стал построенный на основе уровней мандатного управления доступом с информационными потоками по памяти и по времени иерархического представления МРОСЛ ДП-модели механизм мандатного управления доступом. Несмотря на ряд неоспоримых преимуществ этого механизма, его эффективное применение в ОССН возможно только при понимании особенностей настройки и знании типичных ошибок реализации мандатного управления доступом в современных многопользовательских многозадачных распределенных ОС. Среди многих пользователей таких ОС распространено убеждение, что ман-

датное управление доступом (к тому же реализованное на основе устаревшей модели Белла-ЛаПадулы [17, 103]) — своеобразная «серебряная пуля», позволяющая избавиться от утечек конфиденциальных данных раз и навсегда. Это неверно, мандатное управление доступом не создаёт абсолютно непреодолимой защиты, существует целый ряд стандартных приёмов его обхода, рассмотрим некоторые из них.

Нарушитель может попытаться найти в ОС сущность, в которую возможна запись данных, но не входящую в область действия мандатного управления доступом. Например, такая сущность может оказаться в каталоге временных файлов /*tmp*, быть одним из файлов аудита (logфайлом) или по ошибке стать частью какого-либо файла настроек ОС (по аналогии с файлом записей реестра ОС семейства *Microsoft Windows).*

Нарушитель может постараться найти возможность для реализации информационного потока по памяти, не контролируемого или некорректно контролируемого механизмом защиты ОС. Наиболее часто такие информационные потоки обнаруживаются в графических подсистемах современных ОС, средствах отладки приложений, низкоуровневых средствах взаимодействия процессов.

Потенциально возможна реализация нарушителем попыток создать информационный поток по времени с применением совместного доступа контролируемых им разноуровневых субъект-сессий (процессов) к одной сущности или использованием параметров ОС, общесистемной статистики [15]. Несмотря на кажущуюся простоту, такие информационные потоки заблокировать крайне трудно.

Практически все реализации мандатного управления доступом поддерживают специальную привилегию, позволяющую уполномоченной субъект-сессии нарушать правила мандатного управления доступом (в МРОСЛ ДП-модели такая ситуация соответствуют наличию у субъектсессии текущей специальной административной роли *(downgrade_admin_role).* Эта привилегия должна использоваться только доверенными субъект-сессиями, но иногда нарушителю удаётся найти уязвимость, позволяющую получить её недоверенной субъект-сессией. Например, в большинстве ОС(отличных от ОССН), реализующих концепцию суперпользователя *(root),* захват нарушителем его полномочий автоматически влечёт за собой преодоление всей системы мандатного управления доступом.

Реализация перечисленных приёмов для каждой конкретной ОС требует нахождения и последующей эксплуатации её концептуальных или алгоритмических уязвимостей.

Разработка системы

мандатного управления доступом, не содержащей уязвимостей и не позволяющей реализовать ни один из перечисленных или аналогичных подходов, возможна лишь теоретически. Например, на практике можно либо значительно затруднить создание нарушителем рассмотренных запрещённых информационных потоков, либо сделать их пропускнуюспособность очень низкой. Поэтому кратко рассмотрим наиболее существенные проблемы, стоящие перед разработчиками системы мандатного управления доступом.

Двунаправленные информационные потоки. В формальныхмоделях безопасности мандатного управления доступом [17, 105] часто предполагают, что контролируемые информационные потоки являются однонаправленными. Если, например, субъект-сессия читаетсодержимое некоторой сущности, то данные передаются только в одну сторону — от сущности к субъекту, но не наоборот. В рассмотренных в рамках МРОСЛ ДП-модели де-факто правилах вида fl*owmemory_access(x, y*, a_a*), flow-time_access(x, y)* и *de_facto_op(s, op(x, x',* . ..)) (когда его параметром *op(x, $',* .. .) является де-юре правило вида *access_read(x, x', y))* доступ или право доступа на чтение могут использоваться только для создания информационных потоков по памяти или по времени от сущности к субъект-сессии. На практике это условие верно, если сущность является, например, файлом локальной файловой системы ОССН, но может нарушаться при обращении к сетевым файлам и некоторым другим сущностям.

Низкоуровневая техническая реализация доступа на чтение может предполагать, что запрос на чтение передаётся от субъекта к сущности по тому же самому каналу связи, что и ответ на него. Тогда разрешённому информационному потоку на чтение от сущности с низким уровнем конфиденциальности к субъекту с высоким уровнем доступа предшествует запрещённый информационный поток в обратном направлении. Заметим, что данное противоречие не являетсяотличительной особенностью программной реализации мандатного управления доступом, оно проявляется и в безмашинном, бумажном документообороте (всякий раз, когда, например, осуществляется обращение к источнику неконфиденциальнойинформации, сам факт такого обращения может рассматриваться как утечка информации).

Рассмотрим пример. Пусть процесс с высоким уровнем доступа обращается на чтение к файлу, имеющему низкий уровень конфиденциальности и расположенному на удалённом сервере, доступ к которому осуществляется посредством протокола *TCP.* Для иници-

атора запроса этот запрос не противоречит правилам мандатного управления доступом и должен быть выполнен. Но низкоуровневая программная реализация запроса в ядре ОС предполагает отправку серверу пакета, содержащего запрос на установку TCP-соединения, и поскольку этот пакет исходит от процесса с высоким уровнем доступа, он тоже получает высокий уровень конфиденциальности. Если процесссервер, которому адресован запрос, имеет низкий уровень доступа, он не сможет прочитать данный пакет и соединение не будет установлено. Если же процесс-сервер имеет высокий уровень доступа, то соединение будет успешно установлено, но все данные, передаваемые процессомсервером, тоже будут иметь высокий уровень конфиденциальности.

В современном мире разработка сложного ПО ведётся с использованием механизмов инкапсуляции, когда высокоуровневый про- граммный код прикладного или системного ПО абстрагируется от особенностей реализации функций низкоуровневым программным кодом. В рамках данной парадигмы прикладное ПО не должно учитывать, на каком носителе, локальном или сетевом, размещается читаемый им файл. Реализуемый в ПО алгоритм должен работать одинаково и для локального диска, и для удалённого сервера. Мандатное управление доступом в общем случае нарушает эту парадигму. В результате разработчикам мандатного управления доступом приходится либо перерабатывать и адаптировать значительную часть прикладного ПО, либо мириться с тем, что некоторые запрещённые информационные потоки в ряде случаев всё-таки могут быть реализованы.

Проблемы совместимости с прикладным ПО. Часто сложное системное или прикладное ПО требует особой доработки для обеспечения корректного функционирования в ОС, реализующей мандатное управление доступом. Все ПО, входящее в дистрибутив ОССН, подвергается такой доработке, однако в случае установки в ОССН дополнительных программных пакетов *Linux* могут возникать ошибки. Например, если некоторый текстовый редактор для своей работы требует одновременного доступа на чтение и запись к нескольким его конфигурационным файлам (словарей, стилей, настроек графического интерфейса и т.д.), то его работа с документами различных уровней конфиденциальности может привести к многочисленным ошибкам и сделать невозможным использование)того редактора.

Присвоение уровней конфиденциальности системным и служебным сущностям. В рамках МРОСЛ ДП-модели рассмат-

ривались примеры особенных системных сущностей ОССН (например, файлов /d*ev/null* или */dev/zero),* к которым доступ на чтение и запись должен предоставляться субъект-сессиям (процессам), имеющим различные уровни доступа. В противном случае, большая часть пользовательского и системного ПО ОССН утратило бы свою работоспособность. В связи с этим пришлось описать особый порядок доступа к этим сущностям, что усложнило как саму модель, так и её реализацию непосредственно в ОССН.

Таким образом, выявление таких сущностей в ОССН, поиск путей их безопасного использования является самостоятельной сложной задачей, стоящей перед разработчиками ОССН.

Асинхронный ввод-вывод. В ядре современных ОС (например, ОС семейства *Microsoft Windows)* большинство операций ввода-вывода выполняются асинхронно, при этом низкоуровневые компоненты ОС, как правило, не владеют информацией о том, в контексте какого запроса выполняется та или иная операция ввода- вывода. Часто затруднительно даже определить, каким субъектом доступа инициирован тот или иной фрагмент запроса. Если ОС поддерживает только дискреционное управление доступом, это несущественно — права доступа субъекта проверяются до начала выполнения запроса, а в ходе выполнения запроса они изменяться не могут. Но если в ОС реализовано мандатное управление доступом, состоявшийся доступ субъекта к конфиденциальной информации может потребовать отменить одну или несколько асинхронных операций ввода-вывода, выполняющихся в данный момент.

В ОС семейства *Linux,* а значит, и в ОССН, асинхронный ввод-вывод применяется ограниченно, и используемые для этого программные интерфейсы сравнительно просты. Их адаптация к правилам мандатного управления доступом не создаёт серьёзных проблем.

3.1.2. Реализация мандатного управления доступом в ОССН

Как уже отмечалось, основной задачей, решаемой мандатным управлением доступом в современных защищённых ОС, является не полное блокирование каналов утечки конфиденциальной информации, а существенное затруднение их реализации нарушителем. Даже если в ОССН имеются уязвимости, позволяющие в определённых условиях нарушать правила мандатного управления доступом, само наличие этих правил позволяет заметно повысить защищённость ОССН, поскольку:

- многократно снижается риск случайной утечки конфиденциальных данных в результате ошибочных действий пользователя ОССН. Большинство известных подходов к преодолению мандатного управления доступом предполагают выполнение нарушителем сложной последовательности неочевидных действий, которые нельзя выполнить случайно;
- нарушитель, преднамеренно преодолевающий систему мандатного управления доступом, должен затратить значительные усилия на исследование системы, поиск уязвимостей, разработку методов их эксплуатации. В результате вредоносное ПО, разрабатываемое для реализации утечки конфиденциальных данных в популярных ОС, в условиях мандатного управления доступом ОССН практически неработоспособно;
- сложные действия нарушителя, направленные на обход мандатного управления доступом ОССН, должны выявляться её развитой системой аудита или блокироваться организационными мерами защиты.

Для реализации сложных моделей управления доступом в большинстве ОС семейства *Linux* применяется пакет *Security Enhanced Linux (SELinux*) [136]. В ОССН данный пакет не используется по следующим причинам:

- внесение большого объёма программного кода пакета *SELinux* в ОССН предполагает существенный объем работ по его верификациии проверке корректности функционирования, при этом для каждой новой версии пакета *SELinux* эти работы должны повторяться заново;
- средства формального описания политик безопасности, реализуемые в пакете *SELinux,* крайне неудобны для теоретического исследования и описания практически значимых детализированныхполитик безопасности. Не вызывает сомнений, что мандатнуюполитику безопасности, базирующуюся на устаревшей модели Белла-ЛаПадулы, можно построить средствами пакета *SELinux,* но практическая реализация данной задачи сталкивается с серьёзными затруднениями. Говоря неформально «все знают человека, который знает человека, который видел хорошую мандатную политику для пакета *SELinux,* но саму эту политику никто не видел»;
- пакет *SELinux* не имеет встроенных средств для реализации в защищаемой ОС мандатного контроля целостности. Эти средства могут быть созданы с использованием его низкоуровневых механизмов, но практическое решение данной задачи столкнёт-

ся с серьёзными трудностями как на этапе разработки ПО, так и на этапе научного обоснования корректности его функционирования;

• большинство прикладного и системного ПО, предназначенного для выполнения в ОС семейства *Linux,* требуют особой под- стройки для совместимости с пакетом *SELinux* (за исключением вырожденных случаев, когда средствами пакета *SELinux* реализуются тривиальные политики безопасности).

Реализация мандатного управления доступом в ОССН основана на подсистеме безопасности *PARSEC,* самостоятельно разработанной АО «НПОРусБИТех», и включающей соответствующие программный интерфейс и модуль расширения ядра ОССН, поддерживающий виртуальную файловую систему */parsecfs* и набор системных вызовов, позволяющих уполномоченным пользователям управлять политикой безопасности ОССН. Помимо мандатного управления доступом, подсистема безопасности *PARSEC* также реализует мандатный контроль целостности и дополнительные функции аудита. На рис. 1 показано место подсистемы безопасности *PARSEC* в архитектуре ОССН и порядок её взаимодействия с другими компонентами ОССН. Модуль *PARSEC* устанавливает в ядре ОССН собственные обработчики контролируемых информационных потоков, которые получают управление всякий раз, когда необходимо принять решение, следует ли разрешить или запретить то или иное обращение субъект-сессии к сущности. Эти обработчики функционируют автономно, непосредственное взаимодействие модуля *PARSEC с* другими компонентами

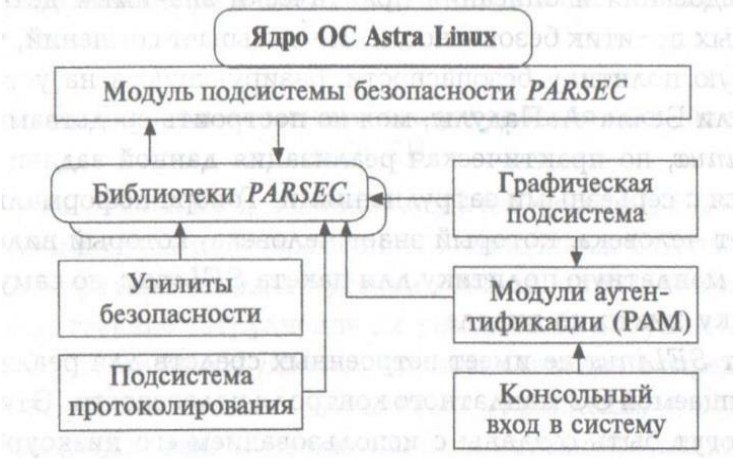


Рисунок 2. Архитектура подсистемы защиты ОССН

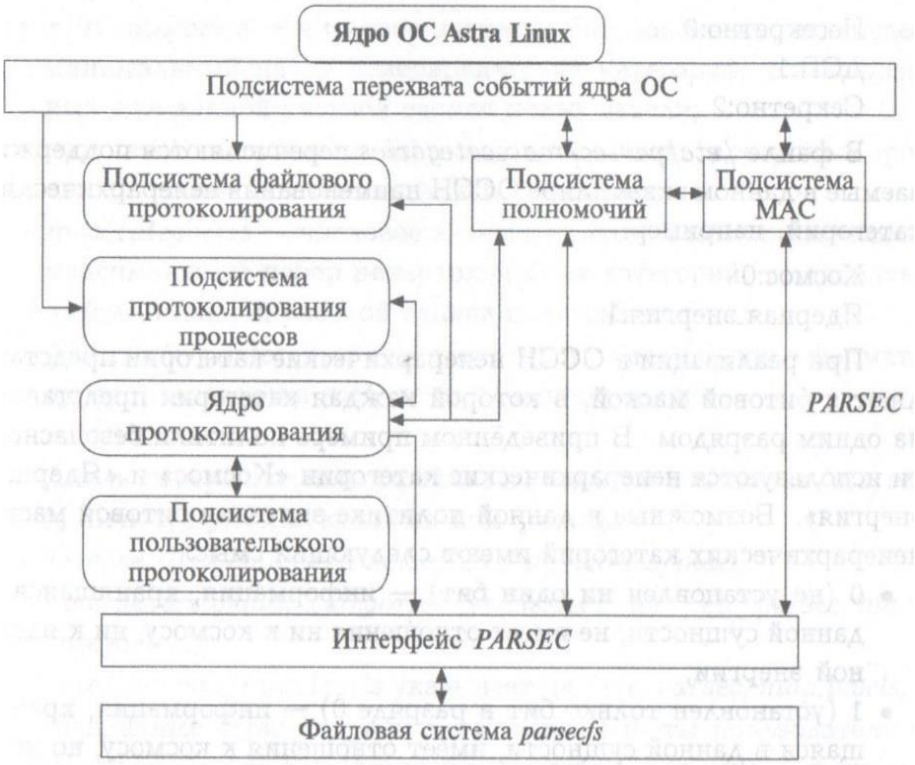


Рисунок 3.Архитектура модуля PARSEC

вующей политике безопасности или модифицирует эту политику. Рис 2 иллюстрирует внутреннюю архитектуру модуля *PARSEC.*

Прикладное и системное ПО реализуют взаимодействие с модулем *PARSEC* (в части касающейся мандатного управления доступом) через шесть системных вызовов:

- *sys_chlbl* — назначает сущности с указанным именем указанную мандатную метку;
- *sys_fchlbl* — назначает сущности с указанным дескриптором указанную мандатную метку;
- *sys_statlbl* — получает мандатную метку сущности с указанным именем;
- *sys_fstatlbl* — получает мандатную метку сущности с указанным дескриптором;
- *sys_setlbl* — назначает выполняющемуся процессу с указанным идентификатором (*PID)* указанную мандатную метку;
- *sys_getlbl* — получает мандатную метку выполняющегося процесса (субъекта) с указанным *PID.*

Файл */etc/parsec/ mac_levels* содержит перечисление поддерживаемых в данном экземпляре ОССН наименований мандатных уровней, например:

Несекретно: 0

ДСП:1

Секретно:2

В файле /*etc/parsec/mac_categories* перечисляются поддерживаемые в данном экземпляре ОССН наименования неиерархических категорий, например:

Космос: 0

Ядерная_энергия: 1

При реализации в ОССН неиерархические категории представляются битовой маской, в которой каждая категория представле на одним разрядом.

В приведённом примере политики безопасности используются неиерархические категории «Космос» и «Ядерная энергия». Возможные в данной политике значения битовой маски неиерархических категорий имеют следующий смысл:

- 0 (не установлен ни один бит) — информация, хранящаяся в данной сущности, не имеет отношения ни к космосу, ни к ядерной энергии;
- 1 (установлен только бит в разряде 0) — информация, хранящаяся в данной сущности, имеет отношения к космосу, но не к ядерной энергии;
- 2 (установлен только бит в разряде 1) — информация, хранящаяся в данной сущности, имеет отношения к ядерной энергии, но не к космосу;
- 3 (установлены биты в разрядах 0 и 1) — информация, хранящаяся в данной сущности, имеет отношение и к космосу, и к ядерной энергии.

Таким образом, в ОССН непосредственно определяется решёткамногоуровневой безопасности, которая в рамках МРОСЛ ДПмодели обозначена через (*LC*, <=).Мандатные атрибуты, назначенные конкретным учётным записям пользователей, перечисляются в каталоге / *etc/parsec/macdb.* Для каждой учётной записи пользователя, которой указан ненулевойи мандатный уровень или непустой перечень неиерархических категорий,создаётся текстовый файл, имя которого совпадает с *uid* учётной записи пользователя. Файл включает в себя единственную строку вида

<usemame>:<min_level>:<mincategories>:<max_level>:<max_categories>, где:

- *username* — имя учётной записи пользователя;
- *min_level* — минимальный мандатный уровень, доступный процессам от имени учётной записи пользователя;

- *min_categories* — числовое значение битовой маски, задающее минимальный набор неиерархических категорий, установленных для данной учётной записи пользователя;
- *max_level* — максимальный мандатный уровень, доступный процессам от имени учётной записи пользователя;
- *max_categories* — числовое значение битовой маски, задающей максимальный набор неиерархических категорий, установленный для данной учётной записи пользователя.

Файл /*etc/parsec/mac* содержит строку аналогичного формата для суперпользователя *root,* по умолчанию равную: *root:* **0**:**0**:**0:0**

Для всех перечисленных файлов и каталогов в каталоге / *etc/ security* имеются указывающие на них ссылки:

- */etc/security/mac* указывает на */etc/parsec/mac/;*
- /*etc/security/mac_categories* указывает на */etc/parsec/mac_ categories,*
- */etc/security/mac_levels* указывает на */etc/parsec/mac_levels.* Мандатные атрибуты текущего сеанса работы пользователя с

ОССН хранятся в файле /*home/%usemame%/.mac_info* в формате

<level>:<category>, где:

- *<level>* — числовое значение мандатного уровня текущего сеанса;
- *<category>* — числовое значение битовой маски иерархических категорий, назначенных текущему сеансу.

Мандатные метки сущностей ОССН по умолчанию распределяются следующим образом.

Корневому каталогу / назначается наивысший мандатный уровень,поддерживаемый в текущем экземпляре ОССН (по умолчанию 3), в битовой маске неиерархических категорий устанавливаются все биты, устанавливаются атрибуты *CCNR* и *CCNRI.* Данный набор настроек фактически указывает, что в файловой системе ОССН могут храниться любые сущности с любыми мандатными метками, при этом мандатный уровень сущностей не может превосходить максимально возможного значения, назначенного каталогу /. Кроме того, благодаря наличию атрибутов *CCNR* и *CCNRI* процесс с любым уровнем доступа, даже минимальным, может обратиться внутрь корневого каталога /.

Системным каталогам */bin, /boot, /etc, /lib, /lib32, /lib64, /lost +found, /media, /mnt, /opt, /proc, /root, /sbin, /selinux, /srv, /sys, /usr* назначается нулевой мандатный уровень и пустая (нулевая)

.

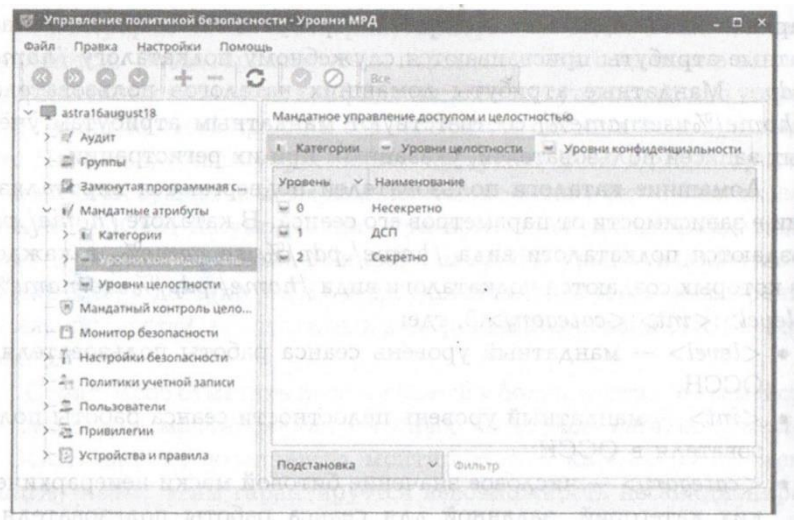


Рисунок 4.Задание уровней доступа и конфиденциальности для ОССН

«Настройки» главного пользовательского меню. Интерфейс этой утилиты интуитивно ясен, при этом большинство её функций доступны только учётным записям пользователей, входящим в группу *astra_admin.*

Используемые в ОССН уровни доступа и конфиденциальности определяются в разделе «Мандатные атрибуты / Уровни» (рис.3).

Каждому уровню соответствует числовое значение, чем оно больше, тем более конфиденциальной считается информация,отнесённая к данному уровню. Администратор ОССН может создавать или удалять уровни, а также редактировать наименование уровня — произвольную текстовую строку, используемую приложениями при отображении сущностей, отнесённым к данному уровню. Наименование уровня не оказывает никакого влияния на порядок обработки информации, отнесённой к нему.

Неиерархические категории определяются в разделе «Мандатные атрибуты / Категории», например, так, как показано на рис. 4.

Назначение учётной записи пользователя уровня доступа и набора неиерархических категорий (задание функции F(u): *U> LC* в МРОСЛ ДПмодели) осуществляется во вкладке «Дополнительные» раздела

«Пользователи и группы / Пользователи / Локальные пользователи» (рис. 3.5).

Назначение элементов окна вполне очевидно. Выбирать ненулевые минимальные значения не рекомендуется.

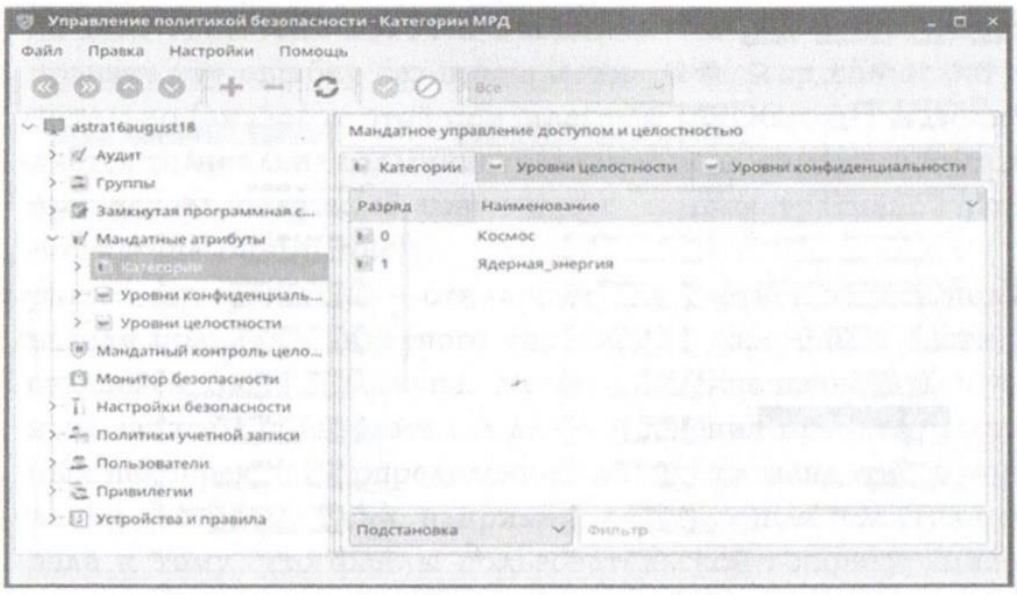


Рисунок 5.Пример задания иерархических категорий

В примере на рис. 5 процессы, выполняющиеся от имени учётной записи пользователя *sec* имеют доступ ко всем сущностям ОССН любого мандатного уровня конфиденциальности, но не имеют доступа к сущностям с неиерархической категорией «Космос». По умолчанию все мандатные атрибуты учётной записи пользователя имеют нулевые значения, что запрещает его процессам всякий доступ к сущностям, содержащим данные, отнесённые к любой неиерархической категории.

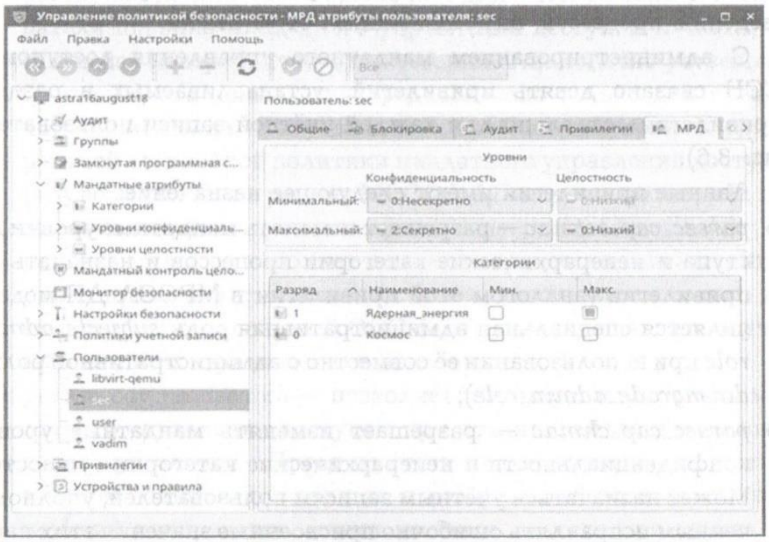


Рисунок 6.Назначение учётной записи пользователя уровня доступа и набора иерархических категорий

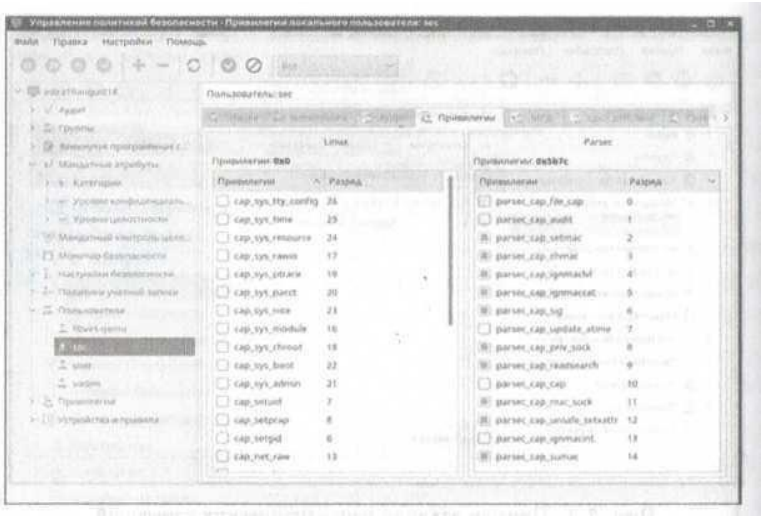


Рисунок 7.Задание привелегий для администрирования мандатного управления доступом

При необходимости параметры мандатного управления доступом можно назначать не только учётным записям «физических» пользователей, но и учётным записям системных пользователей (их часто называют псевдопользователями), предназначенных для обеспечения функционирования ОССН. Делать это без веских причин нерекомендуется, назначение ненулевых уровней доступа учётной записи системного пользователя может привести к непредсказуемому поведению процессов, выполняющихся от её имени.

С администрированием мандатного управления доступом в ОССН связано девять привилегий, устанавливаемых в разделе «Привилегии» отдельно для каждой учётной записи пользователя (рис. 6). Данные привилегии имеют следующее назначение:

- *parsec_cap_setmac* — разрешает изменять мандатные уровни доступа и неиерархические категории процессов и назначать им привилегии (аналогом этой привилегии в МРОСЛ ДП-модели является специальная административная роль *subjects_admin_role* при использовании её совместно с административной ролью *downgrade _admin_role);*
- *parsec_cap_chmac* — разрешает изменять мандатные уровни конфиденциальности и неиерархические категории сущностей. Может назначаться учётным записям пользователей, уполномоченным исправлять ошибочно присвоенные значения этих параметров. Если, например, файл с неконфиденциальным содер-

жимым ошибочно помечен мандатной меткой «секретно», исправить эту ошибку без привилегии *parsec_cap_chmac* затруднительно (аналогом этой привилегии в МРОСЛ ДП-модели является специальная административная роль *entities_admin_role* при использовании её совместно с административной ролью *downgrade _admin_role);*

- *parsec_cap_ignmaclvl* — отключает для учётной записи пользователя правила мандатного управления доступом в части, касающейся уровней доступа. Может временно назначаться учётным записям пользователей для устранения проблем, вызванных некорректным определением политики мандатного управления доступом. Если, например, действующая политика привела к тому, что файлы пользовательской конфигурации получили ненулевые уровни конфиденциальности, устранить эту проблему без временного назначения пользователю данной привилегии затруднительно. Привилегия *parsec_cap_ignmaclvl* предназначена только для временного применения, не следует назначать её никаким учётным записям пользователей на постоянной основе — она фактически отключает мандатное управление доступом для этих учётных записей, тем самым создаётся брешь в безопасности ОССН (возможности, предоставляемые этой привилегией, в МРОСЛ ДПмодели даёт специальная административная роль (*downgrade_admin_role);*
- *parsec_cap_ignmaccat* — отключает для учётной записи пользователя правила мандатного управления доступом в части, касающейся неиерархических категорий. Аналогично *parsec_cap_ignmaclvl,* эта привилегия предназначена только для временного применения при устранении проблем, вызванных некорректной настройкой политики мандатного управления доступом (эти возможности в рамках МРОСЛ ДП-модели также даёт специальная административная роль *downgrade_admin_role);*
- *parsec_cap_sig* — позволяет посылать процессам сигналы, игнорируя правила управления доступом, используется только для системных пользователей ОССН (не должна назначаться учётным записям «физических» пользователей);
- *parsec_cap_readsearch* — позволяет игнорировать правила мандатного управления доступом при чтении файлов и каталогов, но не при записи. Предназначена для использования при резервном копировании данных, к которым пользователь, выполняющий копирование, может не иметь доступа при других обстоятельствах;

- *parsec_cap_mac_sock* — разрешает изменять мандатные уровни конфиденциальности и неиерархические категории сокетов. Используется для обеспечения работоспособности самой ОССН, в связи с этим не следует назначать её никаким учётным записям «физических» пользователей;
- *parsec_cap_unsafe_setxattr* — разрешает изменять мандатные метки сущностей безотносительно мандатных меток каталогов, в которых эти сущности размещаются. Используется только для системных пользователей ОССН (не должна назначаться учётным записям «физических» пользователей);
- *parsec_cap_sumac* — разрешает одновременное выполнение в одном сеансе работы пользователя процессов, имеющих различные мандатные метки (в МРОСЛ ДП-модели такая возможность предусмотрена при использовании де-юре правила вида *create_first_session(x, x', u, y, z, zc, zi)).* Пользователи, не обладающие данной привилегией, не могут пользоваться опцией «Выполнить с другим мандатным уровнем» утилиты «Выполнить команду». Не рекомендуется назначать эту привилегию пользователям без явной необходимости, так как её использование может создать условия для возникновения запрещённых информационных потоков по времени.

Как уже упоминалось в главе 1, пользователь ОССН, начиная сеанс работы с ОССН, указывает мандатные уровни доступа и неиерархические категории для предстоящего сеанса в окне сразу после ввода корректных его имени и пароля (рис. 7).

Элементы окна имеют следующее назначение:

- «Уровень конфиденциальности» — мандатный уровень доступа, заданный по умолчанию для всех субъект-сессий (процессов) начинающегося сеанса.

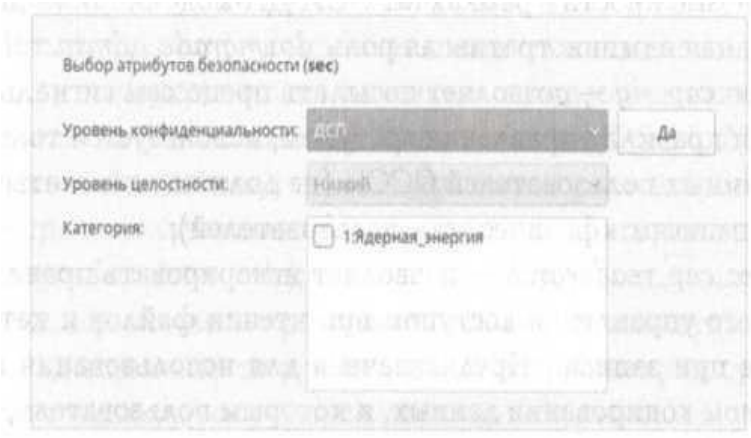


Рисунок 8.Окно ввода параметров мандатного управления доступом при входе пользователя в оссн

· «Уровень целостности» — «Низкий» для непривилегированных (недоверенных) сеансов, «Высокий» для доверенных сеансов, в которых предполагается менять конфигурацию ОССН, устанавливать, конфигурировать или удалять пакеты ПО, и т. д. (подробно администрирование мандатного контроля целостности в ОССН будет рассмотрено в следующем разделе).

· «Категория» — набор неиерархических категорий процессов сеанса.

Данное окно выводится только в том случае, если учётной записи пользователя доступно несколько различных мандатных уровней доступа или неиерархических категорий. Если выбор пользователя предопределён, окно не выводится. Началу сеанса работы пользователя в ОССН в рамках МРОСЛ ДП-модели соответствует применение де-юре правила вида *create_first_session(x, x', u, y, г, zc, zi).*

После входа пользователя в ОССН числовое значение пара- метров мандатного управления доступом для процессов текущего сеанса выводится в правом нижнем углу экрана (области уведомлений) внутри квадрата, цвет которого соответствует этому уровню (рис. 3.8).

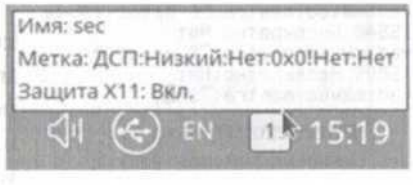


Рисунок 9. Вывод параметров мандатного управления доступом текущего сеанса

Альтернативный способ получить текущие параметры мандат- ного управления доступом для процессов сеанса основан на использовании команды *pdp-id* (рис. 9), которой в МРОСЛ ДП-модели соответствует деюре правило *get_subject_attr(x, y, z).*

Типичным для данной команды является её использование со следующими параметрами:

• *pdp-id* — выдать все параметры мандатного управления доступом и уровень целостности сеанса в текстовом представлении;

```
vadim@astra16august18:~$ pdp-id
Уровень конф.=0(Несекретно), Уровень целостности:63(Высокий), Категории=0x0(Нет)
Роли=:(Нет:Нет)
vadim@astra16august18:~$ pdp-id -l
0
vadim@astra16august18:~$ pdp-id -ln
Несекретно
vadim@astra16august18:~$ pdp-id -c
0x0
vadim@astra16august18:~$ pdp-id -cn
Нет
vadim@astra16august18:~$
```

Рисунок 10.Примеры применения команды pdp-id

- *pdp-idl* — выдать только мандатный уровень доступа в числовом представлении;
- *pdp-idln* — выдать только мандатный уровень доступа в текстовом представлении;
- *pdp-idc* — выдать только битовую маску неиерархических кате горий в числовом представлении;
- *pdp-idcn* — выдать только перечень неиерархических категорий в текстовом представлении.

Ещё один способ получить параметры мандатного управления доступом текущего сеанса — команда *macid,* которая считается устаревшей, и в будущих версиях ОССН её поддержка может быть прекращена.Синтаксис команды *macid* аналогичен синтаксису команды *pdp-id* за исключением того, что команда *macid* не выдаёт данных о текущем уровне целостности.

Получить мандатные атрибуты конкретного процесса позволяет команда *psmac* (рис. 10). Параметром которую необходимо получить. Специальное значение 0 позволяет получить мандатные атрибуты текущего процесса (т. е. самой выполняющейся *команды psmac).*

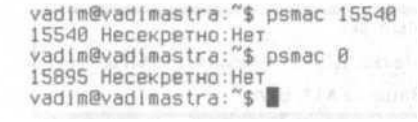


Рисунок 11

При наличии привилегии *parsec_cap_setmac* команда *psmac* позволяет модифицировать мандатные атрибуты процессов.

Допускаются следующие варианты её применения:

- *psmac <pid>* <уровень>:<категория> — назначить указанному процессу указанные мандатные атрибуты;
- *psmacd <pid>* — эквивалентно *psmac <pid>*0:0.

Ручное управление мандатными атрибутами процессов предназначено только для отладки реализованного в ОССН мандатно но для пользователей нецелесообразно.

Штатно пользователь на протяжении всего сеанса работает с данными одного и того же мандатного уровня конфиденциальности и набора неиерархических категорий. Все описываемые далее операции по управлению подсистемой мандатного управления доступом ОССН не предназначены для повседневной работы, они выполняются лишь в редких случаях, когда возникает необходимость «перекатегорировать» те или иные данные, уточнить или модифицировать действующую политику безопасности и т. д. В обычных

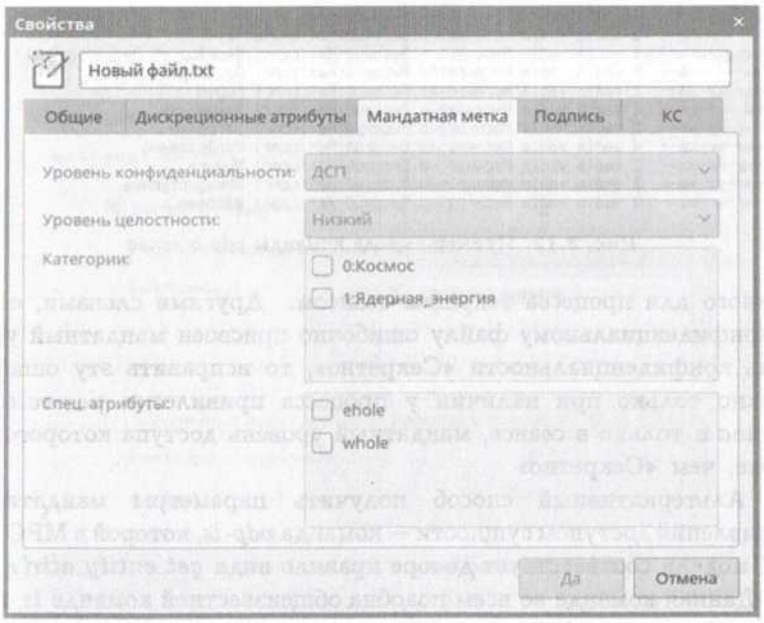


Рисунок 12.Администрирование параметров мандатного управления доступом файла

условиях пользователям нет необходимости явно управлять подсистемой мандатного управления доступом, пользователь может сконцентрироваться на текущей работе, не задумываясь о действующих правилах безопасности.

Просматривать мандатные параметры файлов удобнее всего с помощью графической утилиты «Менеджер файлов» через выбор в контекстном меню для файла раздела «Свойства» и вкладки «Мандатная метка» (рис. 11).

Заметим, что выполнение данной операции считается доступом к файлу на чтение и, следовательно, завершается успешно только в том случае, когда выполняющему его процессу доступ на чтение к файлу не запрещён ни дискреционным, ни мандатным управлением доступом.

Обычно пользователь может только просматривать, но не редактировать параметры мандатного управления доступом файла. Но если учётной записи пользователя предоставлена привилегия *parsec_cap_chmac,* то процесс, выполняющийся от его имени, может как просматривать, так и редактировать параметры мандат- ного управления доступом, не выходя, однако, за пределы мандатного уровня доступа и набора неиерархических категорий, установ-

Рисунок 13. Пример вывода команды pdp-ls-mac

ленного для процесса текущим сеансом. Другими словами, если неконфиденциальному файлу ошибочно присвоен мандатный уровень конфиденциальности «Секретно», то исправить эту ошибку можно только при наличии у процесса привилегии *parsec_cap_chmac* и только в сеансе, мандатный уровень доступа которого не ниже, чем «Секретно».

Альтернативный способ получить параметры мандатного управления доступом сущности — команда *pdp-ls,* которой в МРОСЛ ДП-модели соответствует де-юре правило вида *get_entity_attr(x, y,* z*).* Данная команда во всем подобна общеизвестной команде *ls,* выдающей перечень файлов и подкаталогов заданного каталога, но реализует дополнительный ключ командной строки — *mac.* В этом случае она выдаёт сведения о файлах в формате, похожем на формат вывода команды *lsl* (рис. 12).

Ещё один способ получить мандатные атрибуты сущности — команда *getfmac* (рис. 13).

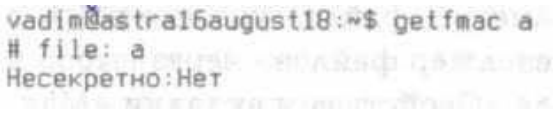


Рисунок 14

• c — не выводить имена обрабатываемых файлов и каталогов; • *s* — пропускать файлы с нулевыми мандатными атрибутами.

Обычно все процессы, запущенные от имени учётной записи пользователя, выполняются с параметрами мандатного управления

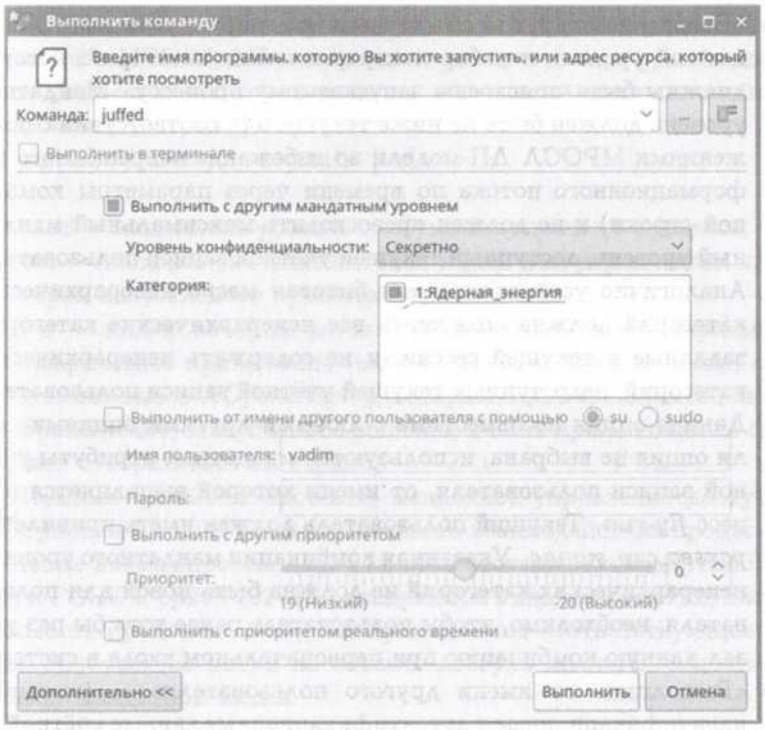


Рисунок 15.Запуск процесса с нестандартными мандатными атрибутами

доступом, заданными в начале сеанса его работы. При необходимости пользователь может запустить один конкретный процесс с другими параметрами мандатного управления доступом, используя графическую утилиту *fly-run,* пользовательский интерфейс которой представлен на рис. 14.

Элементы пользовательского интерфейса имеют следующее назначение:

- «Команда» — имя приложения вместе с командной строкой, которое следует выполнить.
- «Выполнить в терминале» — перед порождением нового процесса создаётся временный терминал, на который отображается стандартный вывод порождаемого процесса. Обычно графические приложения *X Window System* выводят на стандартный вывод диагностические сообщения, которые могут помочь разобраться в причинах происходящего, если приложение функционирует некорректно. Данная опция несовместима с опцией «Выполнить с другим мандатным уровнем».

· «Выполнить с другим мандатным уровнем» — указывается мандатный уровень и набор неиерархических категорий, которые должны быть присвоены запускаемому процессу. Мандатный уровень должен быть не ниже текущего (в соответствии с положениями МРОСЛ ДП-модели во избежание запрещённого информационного потока по времени через параметры командной строки) и не должен превосходить максимальный мандатный уровень, доступный текущей учётной записи пользователя. Аналогично устанавливаемая битовая маска неиерархических категорий должна содержать все неиерархические категории, заданные в текущей сессии, и не содержать неиерархических категорий, недоступных текущей учётной записи пользователя. Данная опция несовместима с любыми другими опциями. Если опция не выбрана, используются мандатные атрибуты учётной записи пользователя, от имени которой выполняется процесс *fly_run.* Текущий пользователь должен иметь привилегию *parsec_cap_sum.ac.* Указанная комбинация мандатного уровня и неиерархических категорий не должна быть новой для пользователя, необходимо, чтобы пользователь ранее хотя бы раз указал данную комбинацию при первоначальном входе в систему.

· «Выполнить от имени другого пользователя» — указывает идентификационные и аутентификационные данные учётной записи пользователя, от имени которой должен быть запущен процесс. Дополнительно нужно выбрать механизм, используемый для запуска процесса от имени другой учётной записи пользователя *(su* или *sudo).* Если опция не выбрана, используется также учётная запись пользователя, от имени которой выполняется процесс *fly-run.*

· «Выполнить с другим приоритетом» — указывает приоритет, назначаемый запускаемому процессу. Данную опцию можно использовать для понижения приоритетов «жадных» программ, неумеренно потребляющих аппаратные ресурсы компьютера. Если опция не выбрана, запускаемому процессу назначается тот же приоритет, что и у процесса *fly-run.*

· «Выполнить с приоритетом реального времени» — указывает, что запускаемый процесс должен быть выполнен с приоритетом реального времени. Данная опция не должна использоваться непривилегированными пользователями. При её выборе выдаётся сообщение, предупреждающее, что запуск процесса в данном режиме может привести к краху ОССН, и требующее дополнительного подтверждения.

Те же самые действия можно выполнить с помощью команды *sumac* (ей в МРОСЛ ДП-модели соответствует де-юре правило вида *create_first_session(x, x', u, y, z, zc, zi*)) со следующим типичным набором параметров: *sumacl <mac>c <cat>* [-x], *где:*

- *mac* — мандатный уровень доступа запускаемого процесса;
- *cat* — числовое значение битовой маски неиерархических категорий запускаемого процесса;
- —*x* — ключ, который должен быть указан, если запускается графическое приложение (если не указать данный ключ, системныйвызов *XOpenDisplay*, который будет выполнять данное приложение, завершится с ошибкой и окно приложения не сможет отобразиться на экране).

Помимо файловой системы, мандатное управление доступом поддерживается в ОССН и для сетевого взаимодействия процессов. В сетевые пакеты протокола IPv4 в соответствии со стандартом RFC 1108 и ГОСТ Р 58256-2018 [13] внедряются мандатные метки, наследуемые от процессов, которым принадлежат соответствующие со- кеты. Отсутствие метки на объекте доступа является синонимом нулевой мандатной метки. Для ряда сетевых сервисов (сервера *LDAP*, *DNS, Kerberos* и т.д.) необходимо обеспечить возможность их работы с клиентами, имеющими разный мандатный контекст безопасности, без внесения изменений в исходные тексты сервиса. Для предоставления данной возможности в ОССН реализован специальный механизм, названный *privsock.*

Управление конфигурацией данного механизма осуществляется путём редактирования конфигурационного файла / *etc/parsec/privsock.conf*

3.1. **Мандатный контроль целостности**

В настоящее время большинство успешных взломов защиты ОС реализуются с применением программных закладок [52] — небольших по объёму кода программ или фрагментов программ, которые скрытно внедряются в атакуемую систему и предоставляют нарушителю скрытный доступ к ресурсам атакуемой ОС, вносят уязвимости в её подсистему безопасности, противодействуют антивирусному ПО, пакетным фильтрам, системам обнаружения атак и т. д.

Компьютерные вирусы и сетевые черви являются частными случаями программных закладок.

Степень уязвимости ОС в отношении программных закладок в основном определяется двумя взаимосвязанными факторами:

· насколько легко программной закладке внедрить свой программный код в критически важные (например, системные) области атакуемой ОС;

· насколько большие полномочия может получить внедрённая в ОС программная закладка в практически значимых ситуациях, Мандатный контроль целостности *(Mandatory Integrity Control — MIC)* [17, 104] в основном предназначен для того, чтобы затруднить программным закладкам внедрение в защищаемую ОС и дальнейшее функционирование в ней [107]. В качестве побочного эффекта нейтрализуется угроза вывода ОС из строя некорректно работающим инсталлятором или деинсталлятором прикладного или системного ПО, ненамеренно повреждающим критически важные программные модули ОС (в англоязычных источниках данную угрозу обычно называют *DLL hell* [111]).Как правило, при реализации в ОС мандатного контроля целостности её сущности разделяются на несколько уровней целостности.

В ОССН, начиная с версии 1.6, реализована решётка иерархических уровней целостности (в МРОСЛ ДП-модели ей соответствует решётка *(LI,* <=)) от 0 до 255, каждый элемент которой является маской из 8 битов. Самыми важными элементами этой решётки, устанавливаемыми по умолчанию, являются уровни целостности «Низкий» (0) и «Высокий» (63). Кроме этих уровней целостности, которые при наличии соответствующих полномочий могут быть непосредственно выбраны пользователем для сеанса его работы, в ОССН также реализованы «промежуточные» уровни целостности для важных для её безопасности компонент (табл. 1).

Таблица 1

Уровни целостности ОССН версии 1.6

Table 1

Значение	Битовая маска	Комментарий
0	00000000	Уровень «Низкий» (*Low*)
1	00000001	Уровень «Сетевые сервисы»
2	00000010	Уровень «Виртуализация»
4	00000100	Уровень «Специальное ПО»
8	00001000	Уровень «Графический сервер»
16	00010000	Свободен, может быть использован для СУБД
32	00100000	Свободен, может быть использован для антивируса
63	00111111	Уровень «Высокий» (*High*)
64	01000000	Свободен, может быть использован для гипервизора
128	10000000	Свободен, может быть использован для контроллера домена

Мандатный контроль целостности, как формально определено в рамках соответствующего уровня иерархического представления МРОСЛ ДП-модели, не допускает модификации сущностей с высоким уровнем целостности недоверенными процессами (субъект-сессиями) с низким уровнем целостности. Обращения процессов ОССН к сущностям, потенциально нарушающие их целостность (операции записи, перемещения или удаления, понимаемые в широком смысле вариантов их реализации), разрешаются только в том случае, когда уровень целостности процесса не уступает уровню целостности сущности, к которой он обращается. Другими словами, модификация низкоцелостным процессом высокоцелостной сущности запрещается независимо от дискреционных или мандатных прав доступа процесса к сущности. Модификация сущностей ОССН, важных с точки зрения её безопасности, разрешается только доверенным процессам, выполняющимся от имени привилегированных учётных записей пользователей с высоким уровнем целостности. Для ограничения возможностей перемещения или удаления критически важных сущностей присваивается соответствующий уровень целостности контейнеру, содержащему такие сущности.

Процесс, выполняющийся на низком уровне целостности, не имеет возможности:

- получать доступ к процессам, выполняющимся на более высоких уровнях целостности, в том числе не может направлять управляющие сообщения их окнам;
- порождать процессы, выполняющиеся от имени другой учётной записи пользователя, с использованием механизмов *su, sudo, suid/sgid*;
- порождать процессы, выполняющиеся на высоком уровне целостности.

В начале сеанса работы с ОССН пользователь выбирает уровень целостности для корневого процесса своей пользовательской сессии. Если для сессии выбран низкий уровень целостности, то все процессы, выполняющиеся в ней, гарантированно выполняются на низком уровне целостности.

Если нарушителю удалось выполнить свой программный код в рамках такой сессии, возможности нарушителя будут сильно ограничены.

Этот код не способен ни обеспечить собственную автоматическую загрузку в следующей сессии того же или другого пользователя, ни эффективно скрывать своё присутствие в ОССН от утилит администрирования и антивирусного ПО. В результате ПО, используемое нарушителем и не проникшее на высокий уровень целост-

ности, как правило, не в состоянии причинить атакованной ОССН сколько-нибудь заметный долговременный ущерб.

Заметим, впрочем, что низкий уровень целостности не ограничивает возможности ПО нарушителя по несанкционированному доступу к пользовательским данным, хранящимся и обрабатывающимся в ОССН. Если нарушитель имеет возможность многократно повторять внедрение программной закладки в атакуемую ОССН (например, эксплуатируя критическую уязвимость в её коде), наличие мандатного контроля целостности не создаёт нарушителю серьёзных затруднений, здесь защита от утечки конфиденциальных данных должна осуществляться с применением мандатного управления доступом.

Большинство пользовательских сеансов должны стартовать на низком уровне целостности. Высокий уровень целостности следует выбирать только в том случае, если пользователь решает задачи администрирования системного ПО, настройки или конфигурирования ОССН в целом. Сеансы с высоким уровнем целостности не должны использоваться чаще, чем это необходимо.

По умолчанию сеанс с высоким уровнем целостности имеет характерную «красную» цветовую гамму (рис. 3.15). Данная цветовая гамма не позволяет пользователю забыть, что он работает в потенциально опасном режиме, требующем повышен-

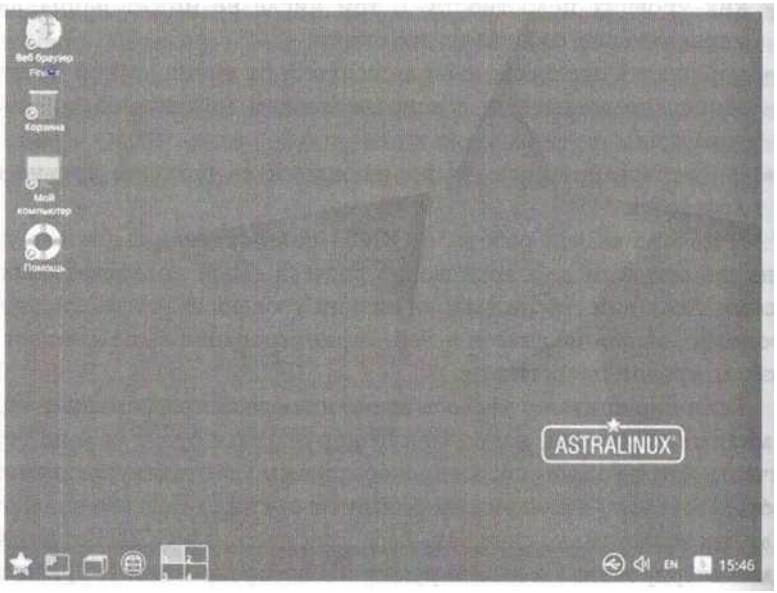


Рисунок 16.Внешний вид сеанса с высоким уровнем целостности

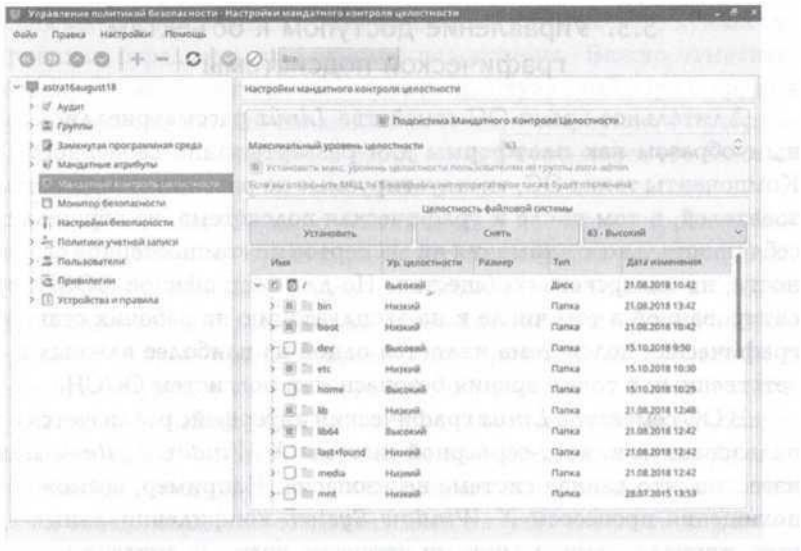


Рисунок 17.Интерфейс утилиты управления подсистемой мандатного контроля целостности

ной внимательности. Кроме того, если пользователь постоянно работает с высоким уровнем целостности, его коллегам становится очевидно, что он пренебрегает правилами безопасности.

Для управления мандатным контролем целостности служит раздел «Мандатный контроль целостности» графической утилиты «Управление политикой безопасности» (рис. 16).

В некоторых случаях, когда администратору безопасности ОССН необходимо выполнить действия по её администрированию, например установить новые пакеты, с помощью этой графической утилиты он может отключить мандатный контроль целостности. Это следует делать на минимально необходимый интервал времени, обеспечив при этом отсутствие в ОССН процессов, функционирующих от имени учётных записей пользователей (как минимум, «физических») с уровнем целостности «Низкий».

В целом корректная настройка мандатного контроля целостности является нетривиальной задачей, требующей от администратора безопасности высокой квалификации и глубокого понимания особенностей функционирования установленного в ОССН системного и прикладного программного обеспечения. Заданная по умолчанию политика назначения сущностям уровней целостности подходит для большинства конфигураций ОССН, менять её без веских причин не рекомендуется.

3.3. **Управление доступом к объектам графической подсистемы**

Длительное время ОС семейства *Linux* рассматривались главным образом как платформы для развёртывания серверного ПО. Компоненты таких ОС, эксплуатируемые на рабочих станциях пользователей, в том числе и графическая подсистема, не обращали на себя пристального внимания ни экспертов по компьютерной безопасности, ни хакерского сообщества. Но для защищённой ОССН, ориентированной в том числе и на эксплуатацию на рабочих станциях, графическая подсистема является одной из наиболее важных и ответственных с точки зрения безопасности подсистем ОССН.

ВОС семейства *Linux* графический интерфейс реализуется с ис- пользованием клиент-серверной системы *X Window System.* Давно известно, что данная система небезопасна. Например, возможность похищения процессом *X Window System* конфиденциальных дан- ных, принадлежащих другому процессу, впервые описана в 1994 г. Р. Браатеном [106]. В 2002 г. Дж. Фишер сформулировал основные угрозы, которые процесс *X Window System* может представлять для других процессов [115]:

- модификация параметров сессии;
- несанкционированное создание/уничтожение окон;
- перехват оконных событий;
- навязывание оконных событий.

Логично предположить, что за прошедшие годы перечисленные угрозы были повсеместно нейтрализованы, но на самом деле это не так. Во всех без исключения (кроме использованной в ОССН) исследованных авторами реализациях *X Window System* некоторые важные проверки, необходимые для управления доступом пользователей к элементам графической подсистемы ОС, реализованы некорректно либо не реализованы вообще [60]. В результате непривилегированный процесснарушитель в ряде случаев может несанкционированно повышать свои полномочия (например, направляя в окна привилегированных процессов последовательности оконных событий, имитирующие действия пользователя), а также похищать конфиденциальные данные из окон других процессов (например, несанкционированно внедряя своё дочернее окно внутрь окна атакуемого процесса). В результате проведённых исследований было обнаружено несколько уязвимостей, каждая из которых может быть устранена добавлением в код *X Window System* проверок соответствующих условий, но для полного их устранения, а также устранения

предпосылок к появлению новых подобных уязвимостей, нужна существенная переделка графической подсистемы. Важно отметить, что обнаруженные уязвимости носят концептуальный характер, для их устранения недостаточно внести точечные изменения в программный код *X Window System*, в исправлении нуждаются его базовые концепции.

Для того чтобы наглядно пояснить одну из многих проблем безопасности *X Window System,* рассмотрим знаменитую атаку *Shatter,* впервые применённую для ОС *Microsoft Windows* в 2002 г. [1138]. В классической реализации данной атаки процесс-нарушитель отправляет в окно привилегированного процесса ОС сообщение о состоянии несуществующего таймера, обработчиком которого якобы является функция, которую нарушитель желает несанкционированно вызвать в контексте привилегированного процесса. Другая известная реализация атаки [140] находит одно из невидимых системных окон, незаметно создававшихся на рабочем столе в ранних версиях ОС *Microsoft Windows XP,* назначает этому окну клавиатурный фокус и имитирует нажатие клавиши F1. Открывается окно справки ОС, при этом обслуживающий его процесс *winhlpZ2.exe* запускается с полномочиями привилегированной учётной записи псевдопользователя *SYSTEM.* Затем в данное окно с помощью имитации перемещения файла из окна в окно (*drag-n-drop*) загружается заранее подготовленный файл справки, содержащий внешнюю ссылку на приложение нарушителя, по которой сразу имитируется активизация. В результате программный код, предоставленный нарушителем, начинает выполняться в контексте привилегированного процесса.

Атака *Shatter* основана на том, что в ОС *Microsoft Windows* 2003 и более ранних версиях окна двух разных процессов, размещённых на одном рабочем столе, могли обмениваться сообщениями фактически без ограничений. В результате во многих практически значимых ситуациях непривилегированные процессы могли несанкционированно повышать свои полномочия, направляя специально подобранные последовательности сообщений окнам привилегированных процессов. Каждая последовательность сообщений, приводящая к несанкционированному повышению полномочий, связана с отдельной уязвимостью, которую чаще всего несложно устранить, но до тех пор, пока сохраняется сама возможность управления окном привилегированного процесса из непривилегированного процесса, говорить о безопасности графической подсистемы не приходится.

Похожие уязвимости были обнаружены и в X *Window System* [60].

В ОС семейства *Microsoft Windows* данная проблема была в основном решена реализацией в ОС *Microsoft Windows Vista* мандатного контроля целостности, дополненного контролем учётных записей (*User Account Control* — U*AC)* [108]. При этом в программный код графической подсистемы ОС были включены несколько сотен дополнительных проверок, учитывающих мандатные уровни целостности процессов и сущностей при реализации самых разнообразных операций в графической подсистеме. Данный подход предъявляет повышенные требования к квалификации и аккуратности реализующих его программистов — достаточно всего лишь одной пропущенной проверки, чтобы в графической подсистеме появилась критическая уязвимость. Особенно трудно реализовать данный под- ход в ситуации, когда базовый функционал графической подсистемы не создан с нуля доверенным разработчиком, а взят из стороннего проекта, скудно документированного, не имеющего централизованной технической поддержки и продолжающего развиваться непонятным и непредсказуемым образом.

В ОССН, начиная с версии 1.4, реализуется другой подход, ос- нованный на изоляции сущностей графической подсистемы, мандатные атрибуты которых отличаются от заданных по умолчанию, в особые сеансы, изолированные с точки зрения графической подсис- темы от основного сеанса работы пользователя с ОССН. В обычном сеансе работы пользователя таких сущностей не создаётся, они могут создаваться только если пользователь, обладающий привилегией *parsec_cap_sumac,* запускает процессы с нестандартными мандатными атрибутами, используя графическую утилиту *fly-run* или консольную утилиту *sumac.* Тогда при старте процесса автоматически открывается новый сеанс (в терминах *X Windows System* — создаётся дисплей), и когда запущенный процесс вызывает системную функцию *XOpenDtsplay,* подключение перенаправляется на этот новый сеанс, а окно приложения приобретает вид, показанный на рис. 17.

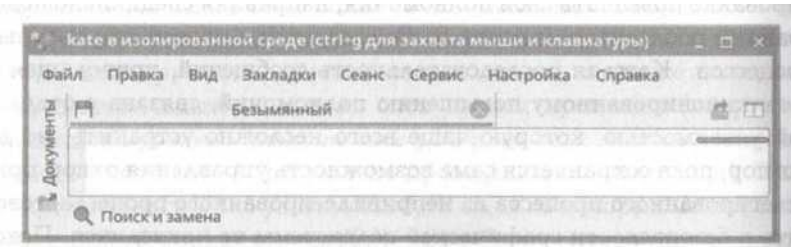


Рисунок 18.Окно приложения выполняющегося в изолированной среде

Окно приложения, выполняющейся в изолированной среде, от- личается от обычных окон следующими особенностями:

· к концу заголовка окна дописана строка «в изолированной сре- де (*ctrl+g* для захвата мыши и клавиатуры)» (заметим, что для большинства приложений захват мыши и клавиатуры осуществляется при необходимости автоматически и не требует нажатия каких-либо особых клавиш);

· цветная рамка, окружающая окно, отличается цветом от размок, окружающих другие окна (за исключением ситуации, когда приложение, выполняющееся в изолированной среде, отличается отдругих приложений только неиерархическими категориями, но не мандатным уровнем);

· каждое приложение, выполняющееся в изолированной среде, имеетсвой собственный буфер обмена (*clipboard*), изолированный от основного буфера обмена, позволяющего пересылать данные между приложениями, при этом передача данных через буфер обмена за пределы изолированной среды невозможна, как и приём данных изза пределов изолированной среды. Для помещения графических приложений в изолированную среду используется утилита *Xephyr,* создающая в ОССН полнофункциональный X-сервер и проецирующая его графический вывод в одно из окон, функционирующих на основном X-сервере. Для графических приложений основного X-сервера окно изолированной среды выглядит как монолитный графический образ. Получить внутреннюю структуру данного окна, используя *XQueryTree* и другие подобные системные вызовы основного X-сервера, невозможно. При запуске приложения в изолированной среде последовательно выполняются следующие действия:

· в основном сеансе работы пользователя, обслуживаемом основ- ным X-сервером ОССН, создаётся новое окно;

· создаётся X-сервер *Xephyr*, его графический вывод перенаправляется в окно, созданное на предыдущем шаге;

· запускается новый экземпляр оконного менеджера *fly-wm,* его подключение к X-серверу (*XOpenDisplay*) перенаправляется на Xсервер *Xephyr* (это необходимо для корректного отображе- ния заголовка окна приложения, корректной работы некоторых функций графического интерфейса);

· запускается целевое приложение, его подключение к X-серверу перенаправляется на X-сервер *Xephyr.* Выполнение графических приложений в изолированной среде требует 40-50 Мбайт дополнительной оперативной памяти на каж-

дое такое приложение. Для сравнения, процесс *Libre Office,* в кото- ром открыт пустой текстовый документ, расходует 110 Мбайт оперативной памяти, пасьянс *QT* с открытой игрой *Free Cell* — 90 Мбайт, калькулятор *Speed, Crunch* — 30 Мбайт. Для большинства современных компьютеров дополнительный расход памяти на погружение приложения в изолированную графическую среду совершенно незаметен. Однако на устаревших аппаратных платформах, а также при запуске в изолированной среде большого количества графических приложений производительность ОССН может снижаться.

Домашний каталог, соответствующий указанной комбинации мандатного уровня и неиерархических категорий, а также все со- путствующие ссылки внутри папки */home/ .pdp* должны существовать до вызова утилиты *fly-run.* Другими словами, утилита *fly-run* позволяет использовать только те комбинации мандатного уровня и неиерархических категорий, которые не являются новыми для данной учётной записи пользователя. Первоначальная инициализация пользовательской среды, соответствующей определённой комбинации мандатного уровня и неиерархических категорий, корректно реализуется только при обычном входе в систему, описанном в главе 1.

В ОССН версии 1.3 или более ранних изолированная среда вы- полнения графических приложений не поддерживалась. Мандатное управление доступом к сущностям графического интерфейса было реализовано путём внесения в программный код X-сервера дополнительных проверок, не позволяющих процессам реализовывать запрещённые информационные потоки по памяти с использованием графического интерфейса. Начиная с версии 1.4, необходимость в этих проверках в значительной степени утрачена, однако программный код, реализующий мандатное управление доступом, не удалялся из графической подсистемы. В современных версиях ОССН он играет роль дополнительного эшелона защиты.

3.4. Особенности аутентификации

Подобно другим современным ОС семейства *Linux* подсисте- ма аутентификации ОССН построена на основе архитектуры *PAM (Pluggable Authentication Modules).* К стандартному набору модулей *PAM* добавлены четыре дополнительных модуля, реализующих назначение мандатных атрибутов, уровня целостности и специфических привилегий ОССН первому процессу в сеансе работы пользователя с ОССН. Если клиентское приложение *PAM* в ходе аутентификации не обращается к перечисленным модулям, сеанс

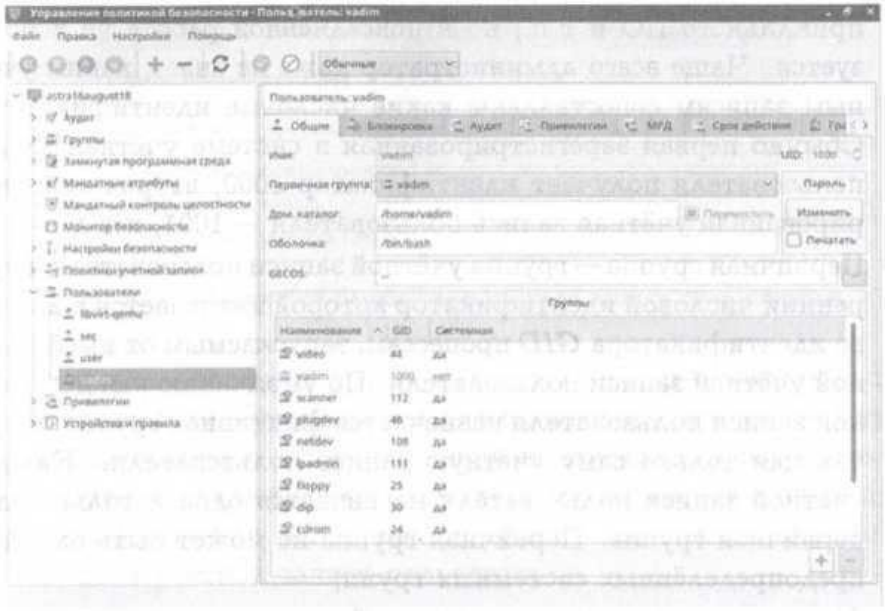


Рисунок 19.Общие настройки учётной записи пользователя

пользователя, аутентифицированного данным приложением, полу- чает низкий уровень целостности и нулевые мандатные атрибуты, не дающие никаких прав доступа к конфиденциальным данным.

В ОССН не предъявляется никаких специальных требований к *PAM*модулям. Любой такой модуль, разработанный для любой ОС семейства *Linux,* должен корректно функционировать в ОССН. В файл */etc/pam.d/common-password* по умолчанию включён *PAM-* модуль *pam_cracklib.so,* запрещающий назначать пользователям простые пароли, нестойкие к подбору.

Администрирование подсистемы аутентификации осуществля- ется с использованием графической утилиты *fly-admin-smc* («Управление политикой безопасности»). Пароль и основные параметры аутентификации для пользователя задаются при регистрации его учётной записи на вкладке «Главные» элемента меню «Пользователи / <имя пользователя>» (рис. 18).

Основные элементы вкладки имеют следующее назначение:

· Имя — имя учётной записи пользователя пользователя (*%username%).*

· *UID* — внутренний числовой идентификатор учётной записи пользователя, назначается при регистрации, в дальнейшем обычно не изменяется. Указывается справочно, так как это может пригодиться при анализе системных журналов, локализации и устранении проблем функционирования системного и

прикладного ПО и т.п., но в повседневной работе не используется. Чаще всего администратор даже не знает, каким учётным записям сопоставлены какие числовые идентификаторы. Обычно первая зарегистрированная в системе учётная запись пользователя получает идентификатор 1000, вторая зарегистрированная учётная запись пользователя — 1001, и т.д.

· Первичная группа — группа учётной записи пользователя, внут- ренний числовой идентификатор которой назначается в качест- ве идентификатора *GID* процессам, запускаемым от имени дан- ной учётной записи пользователя. По умолчанию каждой учёт- ной записи пользователя назначается фиктивная группа, вклю- чающая только саму учётную запись пользователя. Каждой учётной записи пользователя назначается одна и только одна первичная группа. Первичная группа не может быть одной из предопределённых системных групп;

· Дом. каталог — каталог, в котором будут размещаться документы и настройки данной учётной записи пользователя. По умолчанию имя домашнего каталога имеет вид */home/%usemame%;*

· Переместить — указывает, нужно ли при изменении домашнего каталога переместить старое содержимое на новое место. Доступно только если домашний каталог только что изменился и изменения ещё не применены.

· Оболочка — командный интерпретатор, запускаемый в качестве командной оболочки *(shell)* в текстовых терминалах, с которымиработает данный пользователь. По умолчанию используется командный интерпретатор */bin/ bash.* Для учётных записей пользователей, которым запрещено пользоваться командным интерпретатором, данная настройка не имеет смысла.

· Пароль: изменить — позволяет администратору ОССН принудительно назначить учётной записи пользователя новый пароль. Это самый простой способ устранить проблему, если пользователь забыл свой пароль.

· Пароль: печатать — включает отображение учётной карточки пользователя с возможностью её печати.

· *GECOS* — значение элемента *GECOS* учётной записи пользователя.

Данный элемент используется некоторым устаревшим ПО для хранения дополнительной информации о учётной записи пользователя (полное имя, должность, номер комнаты, номер телефона и т.д.), не имеет отношения к обеспечению безопасности. По умолчанию имеет вид «%usemame%».

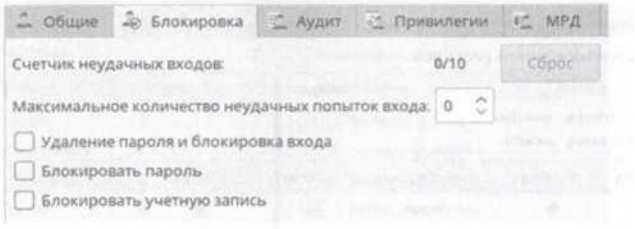


Рисунок 20.Настройка блокировки учётной записи пользователя

· Группы — перечень групп, в которые входит учётная запись пользователя. Заметим, что для предоставления учётной записи пользователя административных полномочий необходимо и достаточно включить её в группу *astra-admin* либо с помощью данной вкладки, либо сделав группу *astraadmin* первичной группой учётной записи пользователя.

Чаще всего все вышеперечисленные свойства учётной записи пользователя однократно задаются при её регистрации и более не изменяются, однако при необходимости администратор ОССН может в любой момент изменить любое её свойство.

Параметры блокировки учётной записи задаются на вкладке «Блокировка», вид которой показан на рис. 19.

Элементы вкладки имеют следующее назначение:

· Счётчик неудачных входов — число неудачных входов в систему, которые пытался совершить пользователь с момента последнего обнуления счётчика неудачных входов либо с момента регистрации учётной записи пользователя, если счётчик ни разу не обнулялся.

· Сброс — кнопка обнуления счётчика неуспешных входов для данной учётной записи пользователя. Доступна, только если счётчик имеет ненулевое значение.

· Максимальное число неудачных попыток входа — максимально допустимое для данной учётной записи пользователя значение счётчика неуспешных входов. При превышении данного значения учётная запись пользователя автоматически блокируется.

· Удаление пароля и блокировка входа — разрешает блокировку входа и удаление пароля при превышении максимального числа неудачных попыток входа.

· Блокировать пароль — включает блокировку пароля;

· Блокировать учётную запись — включает блокировку учётной записи.

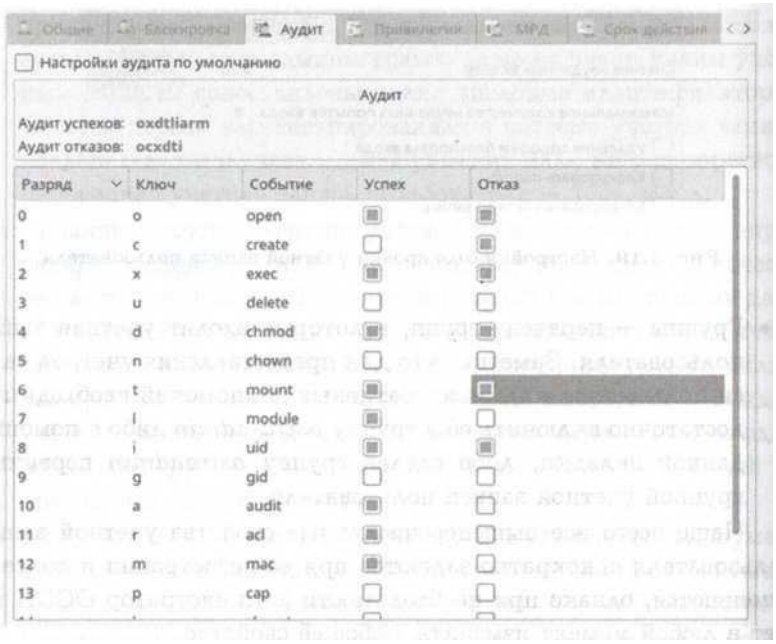


Рисунок 21.Настройка аудита в отношении учётной записи пользователя

Настройки аудита в отношении учётной записи представлены на рис. 20.

Флаг «Настройки аудита по умолчанию» включает для данной учётной записи пользователя настройки, заданные на вкладке «По умолчанию» раздела «Аудит» графической утилиты *fly-admin-smc.* Другие элементы вкладки аналогичны соответствующим элементам вкладки «По умолчанию» раздела «Аудит», они будут рассмотрены ниже.

Вкладка «Привилегии», представленная на рис. 21, перечис- ляет привилегии, назначенные данной учётной записи пользователя. Слева перечислены так называемые привилегии *Linux,* совпадающие с аналогичными привилегиями ОС *Debian Linux,* справа — привилегии *PARSEC,* специфичные для ОССН. Битовые маски внутреннего представления предоставленных привилегий приводятся справочно, эта информация может пригодиться при просмотре журналов аудита и решении некоторых задач администрирования ОССН. В дальнейшем привилегии в соответствии с МРОСЛ ДП-моделью планируется реализовать через роли, запрещающие роли или административные роли.

Рисунок 22.Назначение привелегий учётной записи пользователя

Кратко охарактеризуем привилегии *Linux:*

· *cap-chown* (0) — позволяет объявлять себя владельцем любого файла или каталога;

· *cap_dac-override* (1) — позволяет игнорировать запреты дискреционного управления доступом к файлам и каталогам;

· *cap-dac_read search* (2) — позволяет игнорировать запреты дискреционного управления доступом к файлам и каталогам только в части чтения и поиска информации;

· *cap-fowner* (3) — позволяет получать дополнительные права доступа, как если бы учётная запись пользователя была владельцем всех файлов и каталогов;

· *cap_fsetgid* (4) — позволяет игнорировать запреты на установку битов *SetUID* и *SetGID* в атрибутах дискреционного управления доступом к файлам и каталогам;

· *cap_kill* (5) — позволяет посылать любые сигналы любым процессам;

· *cap_setgid* (6) — позволяет манипулировать атрибутами процесса, указывающими на принадлежность учётной записи пользователя, от имени которой выполняется процесс, той или иной группе;

· *cap_setuid* (7) — позволяет переназначать процессы другим учётным записям пользователей;

· *cap_setpcap* (8) — позволяет назначать и отменять привилегии конкретным процессам;

· *cap_linux_immutable* (9) — разрешает модификацию постоянных файлов и файлов «только для дозаписи» в файловых системах *Ext2* и *Ext3;*

· *cap_net_bind._service* (10) — позволяет привязывать сокеты к IP-портам с номерами, меньшими 1024;

· *cap_net_broadcast* (11) — разрешает широковещательную и многоадресную рассылку сетевых пакетов;

· *cap_net_admin* (12) — позволяет администрировать сетевую под- систему ОС;

· *cap_net_raw* (13) — позволяет осуществлять сетевое взаимодействие на низком уровне, с использованием так называемых «сырых» сокетов;

· *cap_ipc_lock* (14) — позволяет блокировать страницы и секции разделяемой оперативной памяти;

· *cap_ipc_owner* (15) — позволяет получать дополнительные права доступа, как если бы учётная запись пользователя была владельцем всех объектов межпроцессного взаимодействия;

· *cap_sys_module* (16) — позволяет загружать и выгружать модули ядра;

· *cap_sys_rawio* (17) — позволяет получать доступ к портам ввода/вывода посредством системных вызовов *ioperm()* и i*pol();*

· *cap_sys_chroot* (18) — необходима для успешного осуществления системного вызова *chroot();*

· *cap_sys_ptrace* (19) — позволяет осуществлять системный вызов *ptrace()* в отношении любого процесса ОС;

· *cap_sys_pacct* (20) — разрешает настройку попроцессного учета;

· *cap_sys_admin* (21) — позволяет администрировать ОС (проверяется множеством разных системных вызовов, связанных с «обобщённым» администрированием);

· *cap_sys_boot* (22) — позволяет перезагружать ОС системным вызовом *reboot();*

· *cap_sys_nice* (23) — позволяет назначать процессам высокие приоритеты;

· *cap_sys_resource* (24) — позволяет манипулировать квотами распределения аппаратных ресурсов между процессами;

· *cap_sys_time* (25) — позволяет менять системное время; • *cap_sys_tty.config* (26) — необходима для выполнения некоторых действий с текстовым терминалом;

· *cap_mknod* (27) — позволяет осуществлять привилегированные системные вызовы *mknod()*;

· *cap_lease* (28) — позволяет осуществлять блокировку типа «аренда файла».

Привилегии *PARSEC:*

· *parsec_cap_file_cap* (0) — позволяет устанавливать привилегии на файлы;

· *parsec-cap_audit* (1) — позволяет управлять подсистемой аудита;

· *parsec_cap_setmac* (2) — позволяет менять мандатные метки процессов;

· *parsec_cap_chmac* (3) — позволяет менять мандатные метки файлов; • *parsec-cap_ignmaclvl* (4) — позволяет игнорировать правила мандатного управления доступом в части, касающейся мандатных уровней;

· *parsec-cap_ignmaccat* (5) — позволяет игнорировать правила мандатного управления доступом в части, касающейся иерархических категорий;

· *parsec_cap_sig* (6) — позволяет игнорировать правила мандатного управления доступом при отправке сигналов процессам;

· *parsec_cap_update_atime* (7) — позволяет игнорировать правила мандатного управления доступом при установке времени доступа к файлам;

· *parsec_cap_privsock* (8) — позволяет создавать привилегированный сокет и менять его мандатную метку;

· *parsec_cap_readsearch* (9) — позволяет игнорировать правила мандатного управления доступом только в части чтения и поиска информации;

· *parsec_cap_Cap* (10) — позволяет игнорировать правила мандатногоуправления доступом при назначении привилегий процессам;

· *parsec_cap_mac_sock* (11) — позволяет менять мандатную метку сетевого соединения;

· *parces_cap_unsafe_setxattr* (12) — позволяет устанавливать мандатные атрибуты объектов файловой системы без учёта мандатных атрибутов родительского объекта-контейнера;

· *parsec_cap_ignmacint* (13) — позволяет игнорировать правила мандатного контроля целостности;

· *parsec_cap_sumac* (14) — позволяет стартовать процессы с мандатными атрибутами, отличными от заданных для текущего пользовательского сеанса.

По умолчанию никакие учётные записи пользователей не имеющих никаких привилегий. Не следует назначать им привилегии без веских причин, надо понимать, что каждое назначение привилегии создаёт в политике безопасности потенциальную уязвимость.

Для управления привилегиями также могут применяться следующие утилиты командной строки:

· *usercaps* — позволяет просматривать и устанавливать привилегии учётным записям пользователей;

· *execaps* — позволяет устанавливать привилегии запускаемому процессу;

· *pscaps* — позволяет просматривать и устанавливать привилегии выполняющемуся процессу.

Вкладка «МРД», описывающая назначение учётным записям пользователей мандатных атрибутов, уже была рассмотрена в этой главе.

Вкладка «Срок действия», представленная на рис. 22, описывает параметры политики безопасности, связанные со сроками действий пароля и учётной записи пользователя.

Элементы вкладки имеют следующее назначение:

· Минимальное число дней между сменами пароля — описывает минимальный интервал времени между двумя последовательными сменами пароля. Специальное значение 0 (задано по

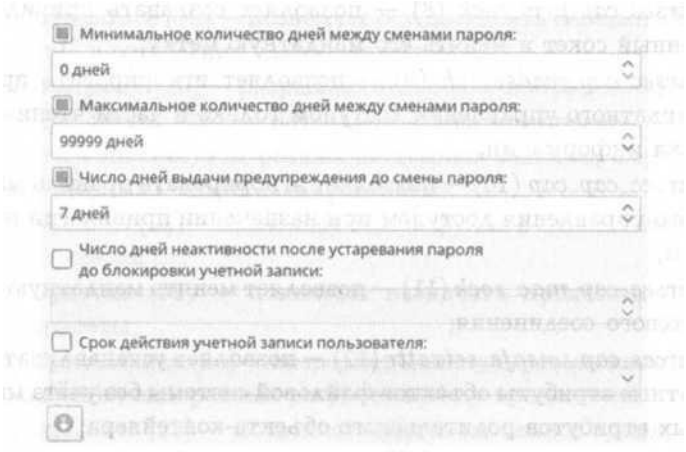


Рисунок 23.Параметры политики безопасности, связанные со сроками действий пароля и учётной записи пользователя

умолчанию) разрешает менять пароль немедленно после предыдущего изменения.

· Максимальное число дней работоспособности пароля — описывает максимальный срок действия пароля, по истечении которого пароль устаревает. По умолчанию задан максимально допустимый для ввода в окне срок 99999 дней, что составляет немногим менее 274 лет и фактически отключает данную политику.

· Число дней выдачи предупреждения до смены пароля — описывает поведение ОССН, когда пароль учётной записи пользователя вот-вот устареет. Если значение данного поля равно *N,* то за *N* дней до устаревания пароля при каждом успешном входе в систему пользователю выдаётся предупреждение, что его пароль скоро устареет.

· Число дней неактивности после устаревания пароля до блокировки учётной записи пользователя — описывает поведение ОССН в случае, если устаревание пароля произошло в период долгой неактивности пользователя, когда пользователь не входил в систему много дней подряд, обычно такие ситуации связаны с отпуском, командировкой или болезнью. Если значение данного поля равно *N,* ОССН разрешает единственный вход по устаревшему паролю в течение *N* дней после устаревания, затем учётная запись блокируется. Специально значение — 1 (установлено по умолчанию) требует блокировать учётную запись пользователя немедленно после устаревания пароля.

· Срок действия учётной записи пользователя — устанавливает дату, по достижении которой учётная запись становится недействительной.

· Зелёная кнопка внизу — импортирует параметры политики из ранее подготовленного шаблона.

Помимо индивидуальных настроек учётных записей, в ОССН действуют три общесистемные политики, связанные с подсистемой аутентификации: политика блокировки, политика паролей и политика создания учётных записей пользователей. Все три политики задаются в разделе «Политики учётных записей» графической утилиты *fly-adminsmc.*

Общесистемная политика блокировки учётных записей пользователей описывается в подразделе «Блокировка», вид соответствующего интерфейса приведён на рис. 23.

Элементы этого интерфейса имеют следующее назначение:

· Индивидуальные настройки — разрешает или запрещает индивидуальное управление максимальным числом неуспешных по-

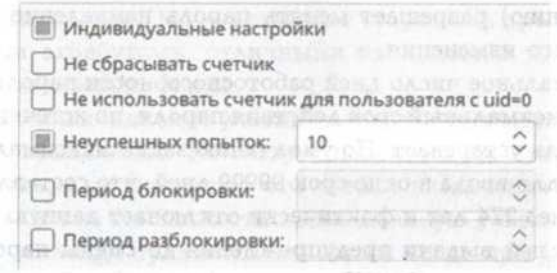


Рисунок 24. Вид интерфейса управления общесистемной политикой блокировки учётных записей пользователей

пыток входа для каждой отдельной учётной записи пользователя.

· Не сбрасывать счётчик — определяет порядок изменения счётчика неуспешных входов: при успешном входе — сбросить (обнулить) либо уменьшить на 1.

· Не использовать счётчик для учётной записи пользователя с *uid* = 0 — запрещает блокировать учётную запись суперпользователя *root.*

· Неуспешных попыток — числовое значение для максимально допустимого количества неуспешных входов.

· Период блокировки — длительность в секундах периода времени, когда запрещается повторно входить в систему после каждой неуспешной попытки входа.

· Период разблокировки — длительность в секундах периода времени, когда запрещается повторно входить в систему после превышения максимально допустимого значения счётчика не успешных попыток входа.

Целесообразно установить период блокировки длительностью порядка несколько десятков секунд, период разблокировки — намного длиннее.

Интерфейс управления общесистемной политикой паролей, за данной в подразделе «Политика паролей», разделяется на три вкладки. Вкладка «Сложность» представлена на рис. 24. Элементы вкладки имеют следующее назначение:

· Проверка имени пользователя — не позволяет использовать в качестве пароля имя учётной записи пользователя или простую функцию от него.

· Проверка *GECOS* — не позволяет использовать в качестве пароля информацию из поля *GECOS* учётной записи пользователя или простую функцию от неё.

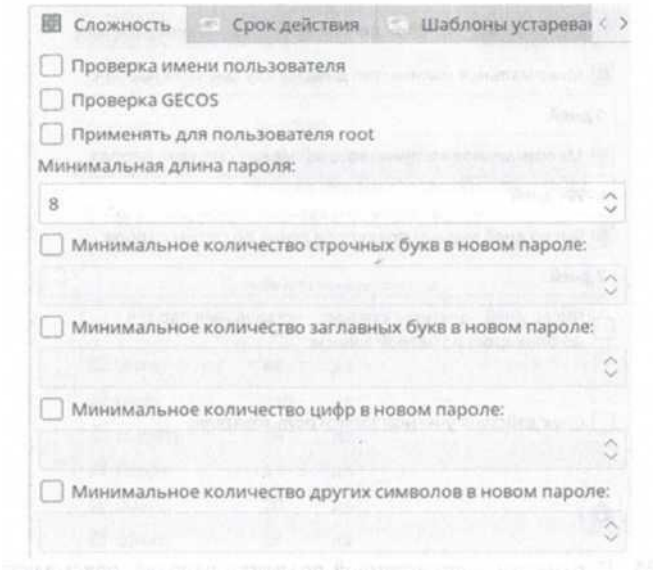


Рисунок 25.Параметры общесистемной политики паролей, описывабщие сложность паролей

· Применять для пользователя *root* — распространяет политику паролей на учётную запись пользователя *root* (по умолчанию никакие ограничения, связанные с безопасностью, на суперпользователя *root* не распространяются).

· Минимальная длина пароля — задаёт минимально допустимую длину пароля в символах.

Минимальное число строчных букв в новом пароле, Минимальное число заглавных букв в новом пароле, Минимальное число цифр в новом пароле, Минимальное число других символов в новом пароле — накладывают ограничения на содержимое пароля, не позволяя применять длинные, но легкоподбираемые пароли наподобие «1234567890» или «ааааааааааааааааааааааааа». Вкладка «Срок действия» описывает параметры политики паролей, имеющие отношение к срокам действия паролей. Вид вклад- ки представлен на рис. 25.

Назначение элементов вкладки вполне очевидно. Зелёная кнопка в нижней части вкладки импортирует параметры политики из шаблона, ранее подготовленного с применением вкладки «Шаблоны устаревания». Раздел, определяющий политику создания учётных записей пользователей, имеет вид, представленный на рис. 26.

Элементы раздела имеют следующее назначение:

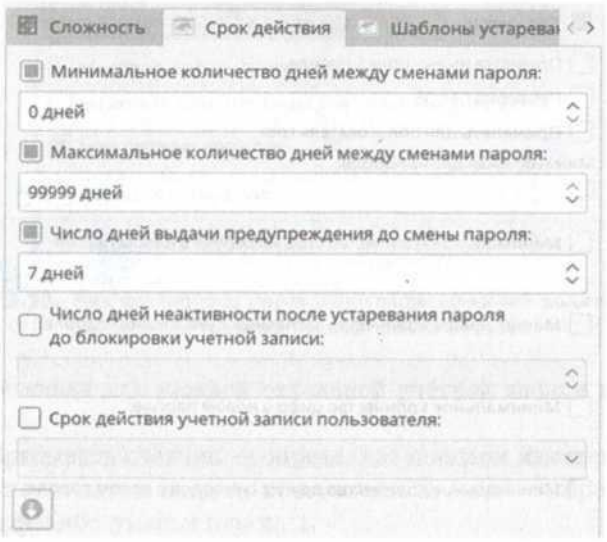


Рисунок 26.Параметры общесистемной политики паролей, описывающие срок действия паролей

· Дом. каталог — описывает каталог, внутри которого размещаются домашние каталоги учётных записей пользователей. По умолчанию это каталог */home,* менять данную настройку не рекомендуется.

· Каталог шаблонов — описывает каталог, содержащий шаблоны, используемые для первоначального заполнения вновь создаваемых домашних каталогов файлами настроек *(profile, bashrc* и т.д.).

· Оболочка — путь к файлу заданного по умолчанию командного интерпретатора.

· Первичная группа — группа, назначаемая по умолчанию первичной группой вновь создаваемым учётным записям пользователей.

Настройка имеет смысл только если не установлено «Создавать новую пользовательскую группу».

· Создавать новую пользовательскую группу — если установлено, для каждой создаваемой учётной записи пользователя автоматически создаётся группа с тем же именем, и она назначается первичной группой данной учётной записи пользователя.

· Добавлять пользователя в дополнительные группы — если установлено, новые учётные записи пользователя автоматически добавляются в перечисленные ниже группы.

· Дополнительные группы — список групп, в которые автомати-

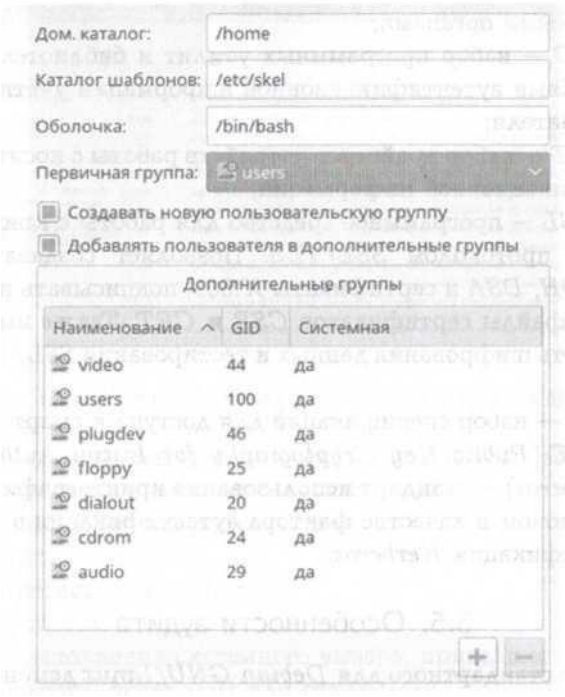


Рисунок 27.Параметры политики создания учётных записей пользователей

чески добавляются вновь создаваемые учётные записи пользователей. Настройка имеет смысл только если установлено «Добавлять пользователя в дополнительные группы».

Помимо вводимых с клавиатуры паролей, в ОССН поддерживаются и другие факторы аутентификации:

· одноразовые пароли (скрэтч-карты); физические устройства или носители, предоставляющие аутентификационную информацию (смарт-карты, USB-токены и др.);

· биометрическая информация.

Для обеспечения двухфакторной аутентификации с помощью внешнего носителя используются следующие средства и технологии:

· *PKCS* (*Public-Key Cryptography Standard)* — группа стандартов криптографии с открытым ключом, в частности стандарты *PKCS*-11,
PKCS-12, *PKCS-15*, относящиеся к работе с криптографическими токенами;

· X.509 — стандарт, определяющий форматы данных и процедуры распределения открытых ключей с помощью сертификатов с цифровыми подписями, которые предоставляются сертифи-

кационными органами;

· *OpenSC* — набор программных утилит и библиотек работы с носителями аутентификационной информации учётной записи пользователя;

· *ОрепСТ* — набор драйверов устройств работы с носителями аутентификационной информации;

· *OpenSSL* — программное средство для работы с криптографическимпротоколом *SSL/ TLS.* Позволяет создавать ключи *RSA, DH, DSA* и сертификаты X.509, подписывать их, формировать файлы сертификатов *CSR* и *CRT.* Также имеется возможность шифрования данных и тестирования *SSL/ TLS* соединений;

· *PC/SC* — набор спецификаций для доступа к смарт-картам;

· *PKINIT (Public Key Cryptography for Initial Authentication in Kerberos)* — стандарт использования криптографии с открытым ключом в качестве фактора аутентификации в протоколе аутентификации *Kerberos.*

3.5. **Особенности аудита**

Помимо стандартного для *Debian GNU/Linux* демона *rsyslogd,* в ОССН также имеется собственная система аудита, реализуемая подсистемой безопасности *PARS EC,* позволяющая более эффективно управлять регистрацией событий, непосредственно связанных с безопасностью ОССН. Архитектура аудита *PARSEC* показана на рис 27.

Рисунок 28.Архитектура аудита PARSEC

Зарегистрированные события записываются в файлы *kernel, mlog* и *user.mlog,* по умолчанию размещаемые в каталоге /*var/log*/

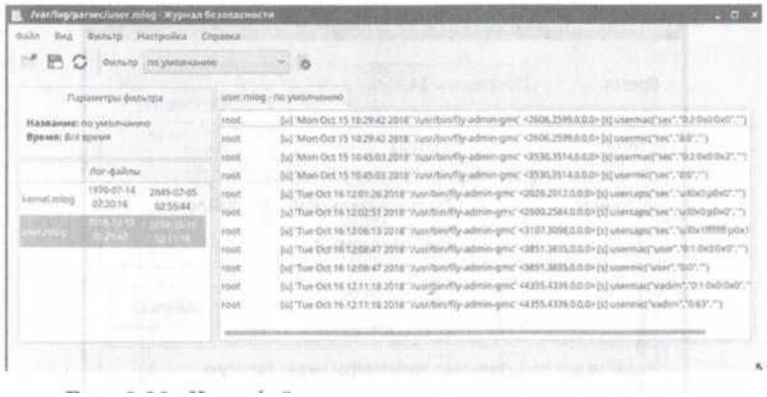


Рисунок 29.Интерфейс утилиты просмотра журналов аудита

parsec. Каждая запись файла соответствует одному зарегистрированному событию и содержит следующие данные:

- имя учётной записи пользователя;
- источник события;
- время регистрации события;
- имя процесса;
- статус выполнения системного вызова, приведшего к регистрации события: успех ([s]) или отказ ([f]);
- имя и параметры системного вызова, приведшего к регистрации события.

Для просмотра зарегистрированных событий аудита используется графическая утилита *fly-admin-view audit*, интерфейс которой показан на рис. 28.

В левой части окна интерфейса перечислены текущие параметры фильтрации отображаемых событий, а также файлы, доступные для просмотра. В правой части окна интерфейса отображаются события, зарегистрированные в выбранном файле аудита и удовлетворяющие текущим параметрам фильтрации. События, связанные с успешными обращениями к данным, отображаются белым цветом, события, связанные с неуспешными обращениями, — красным.

Управление фильтрацией событий аудита осуществляется по команде меню «Фильтр / Изменить» с использованием интерфейса,представленного на рис.29.

Основные элементы интерфейса имеют следующее назначение:

- Заголовок интерфейса содержит имя редактируемого фильтра.
- Время — позволяет задать интервал времени, к которому должны принадлежать отображаемые события аудита. Можно указать в явном виде начальный и конечный моменты, воспользо-

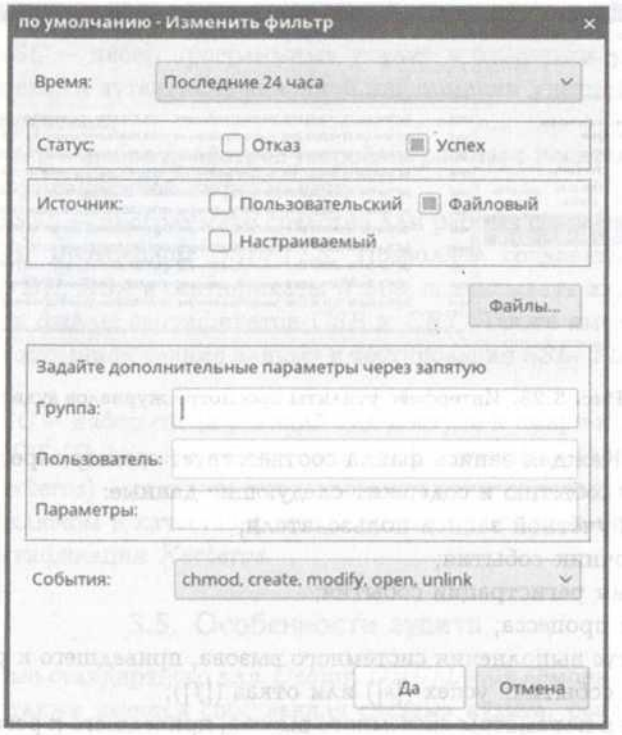


Рисунок 30.Управление фильтрацией событий аудита

ваться одним из предопределённых значений либо отключить фильтрацию событий аудита по времени.

· Статус — позволяет фильтровать отображаемые события аудита по статусу выполнения системного вызова. Если в фильтре выбран только один статус, в окно утилиты выводятся только события аудита с этим статусом, если выбраны или не выбраны оба статуса, фильтрация событий аудита по статусу отключена.

· Источник — позволяет фильтровать отображаемые события аудита по источнику. Если в фильтре не выбран ни один источник, фильтрация по источнику отключена, и в окно утилиты выводятся события аудита с любым источником, так же как когда выбраны все три источника.

· Файлы — позволяет ограничить отображаемые события аудита определённой областью файловой системы ОССН.

· Дополнительные параметры — позволяет фильтровать события аудита, содержащие заданные подстроки в соответствующих полях

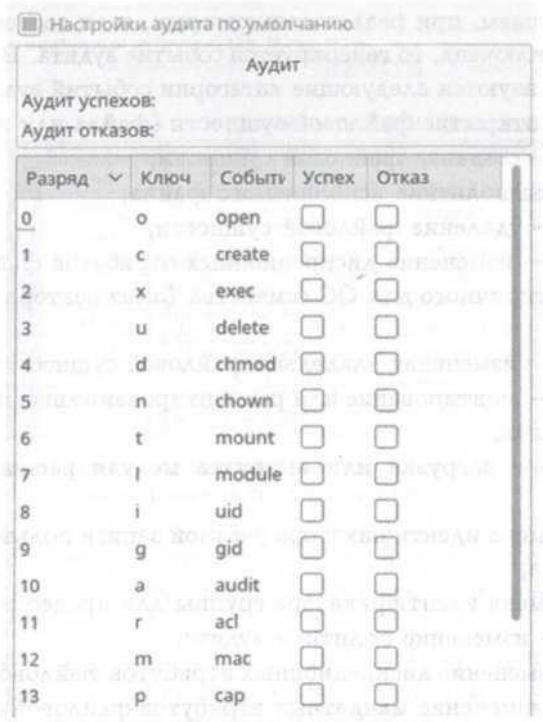


Рисунок 31.Настройка политики аудита

События — позволяет фильтровать отображаемые события аудита по типу системного вызова, с которым оно связано. Политикой аудита удобнее всего управлять с помощью графической утилиты *fly-admin-smc* («Управление политикой безопасности»).

Аналогично политикам паролей в ОССН может одновременно действовать несколько политик аудита. Каждая политика аудита может быть применена к конкретной учётной записи пользователя или к группе учётных записей пользователей, кроме того, существует общесистемная политика, действующая по умолчанию. Создание и редактирование политик аудита осуществляется в разделе «Аудит» (рис. 3.30).

Политика аудита кодируется двумя битовыми масками, в которых каждый бит соответствует категории событий, регистрируемых подсистемой аудита. Одна маска перечисляет успешные события (т. е. события, соответствующие успешным попыткам доступа про- цессов к сущностям), другая — неуспешные события. Каждая категория событий аудита соответствует одному или нескольким сис-

темным вызовам, при реализации которых, если соответствующая категория включена, то генерируется событие аудита. В ОССН версии 1.6 реализуются следующие категории событий аудита:

- *open* — открытие файловой сущности (файла или каталога);
- *create* — создание файловой сущности;
- *exec* — выполнение исполняемого файла;
- *delete* — удаление файловой сущности;
- *chmod* — изменение дискреционных атрибутов файловой сущности (типичного для ОС семейства *Linux* вектора прав доступа);
- *chown* — изменение владельца файловой сущности;
- *mount* — монтирование или размонтирование внешнего носителя данных;
- *module* — загрузка или выгрузка модуля расширения ядра ОССН;
- *uid* — смена идентификатора учётной записи пользователя для процесса;
- *gid* — смена идентификатора группы для процесса;
- *audit* — изменение политики аудита;
- *acl* — изменение дискреционных атрибутов файловой сущности;
- *mac* — изменение мандатных атрибутов файловой сущности;
- *cap* — изменение привилегий учётной записи пользователя или группы;
- *chroot* — смена корня файловой системы ОССН для процесса;
- *rename* — переименование файловой сущности;
- *net* — некоторые сетевые операции.

Каждой учётной записи пользователя и группе ОССН может быть сопоставлена своя политика аудита. Для создания новой политики аудита используются подразделы «Аудит / Группы» или «Аудит / Пользователи» соответственно. Альтернативный способ определить или скорректировать политику аудита для учётной записи пользователя или группы — открыть соответствующую вкладку «Аудит».

По умолчанию каждый экземпляр ОССН реализует собственную политику аудита, хранящуюся на данном компьютере. Кроме того, поддерживается централизованное управление аудитом в масштабе корпоративной сети. В дистрибутив ОССН включены следующие средства централизованного аудита:

- *ossec-hids-server* — сервер аудита, устанавливается на специальновыделенном сервере;

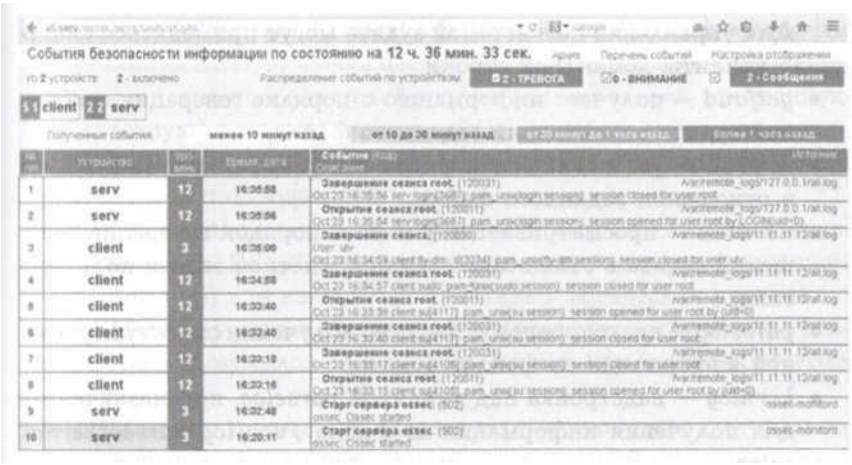


Рисунок 32.Графический интерфейс централизованного управления аудитом просмотр зарегестрированных событий

· *ossec-hids-agent* — агент аудита, устанавливается на каждом компьютере, на котором реализуется централизованное управление аудитом;

· *ossec-web* — графический интерфейс для управления сервером аудита (рис. 31, 32), устанавливается на том же компьютере, что и сервер аудита, требует наличия веб-сервера *Apache*2 с поддержкой *PHP;*

· *ossec-cnt* — средство настройки аудита для бездисковых станций.

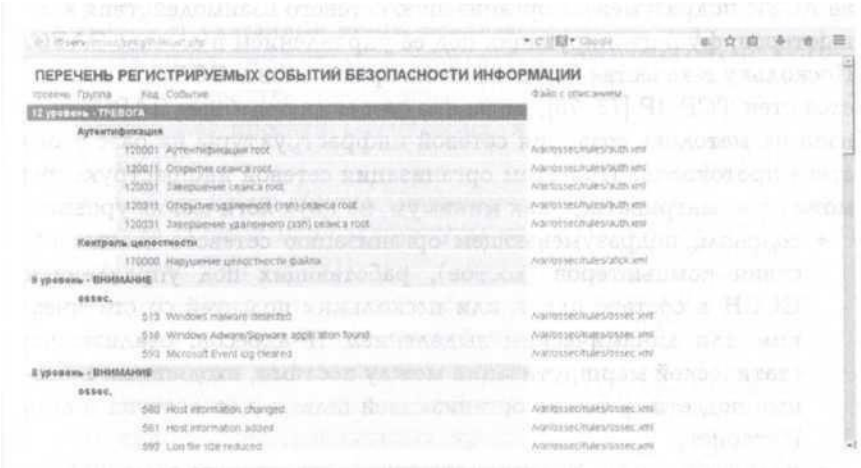


Рисунок 33. Графический интерфейс централизованного управления аудитом управление политикой аудита

Для управления подсистемой аудита могут применяться следующие утилиты командной строки:

· *getfaud* — получает информацию о порядке генерации сообщений аудита в отношении заданного файла;

· *setfaud* — устанавливает порядок генерации сообщений аудита в отношении заданного файла;

· *useraud* — просматривает и изменяет порядок генерации сообщений аудита в отношении заданной учётной записи пользователя; • *parselog* — низкоуровневое средство получения структурированной информации из файлов аудита;

· *kemlog* — надстройка над утилитой *parselog,* предназначенная для получения информации из файла /*var/log/parsec/ kernel, mlog;*

· *userlog* — надстройка над утилитой *parselog,* предназначенная для получения информации из файла / *var/ log/parsec/ user. mlog;*

· *useraud* — просматривает и изменяет порядок генерации сообщений аудита в отношении заданного процесса.

3.6. **Сетевое взаимодействие в ОССН. Организация доменной инфраструктуры**

3.6.1. Логические уровни сетевой инфраструктуры

В главе 1 было отмечено, что использование ОССН в составе АСЗИ подразумевает организацию сетевого взаимодействия компьютеров, функционирующих под её управлением в составе ЗЛВС. Поскольку в качестве базового стека протоколов в ОССН используется стек TCP/IP [73, 75], методика создания подобных ЗЛВС основана на методике создания сетевой инфраструктуры на базе этого стека протоколов. При этом организация сетевой инфраструктуры может рассматриваться, как минимум, на двух логических уровнях:

· *базовом,* подразумевающем организацию сетевого взаимодействия компьютеров (хостов), работающих под управлением ОССН в составе одной или нескольких подсетей со статическим или динамическим выделением IP-адресов, реализацией статической маршрутизации между хостами, входящими в разные подсети, а также организацией шлюза для доступа в сеть Интернет;

· *корпоративном,* подразумевающем организацию доменной сетевой инфраструктуры на базе одного из вариантов службы ка-

талогов *(Directory Service),* обеспечивающей её пользователям механизм ЕПП, включая централизованное управление учётными записями пользователей и групп пользователей, прозрачную (сквозную) аутентификацию в ЕПП с любого хоста, являющегося клиентом домена и централизованное хранение домашних каталогов пользователей.

В этом случае модули *LSM Module Policy Engine* подсистемы безопасности *PARSEC* раздельно реализуют управление доступом *(hooks/decision)* для каждого из указанных логических уровней: LSMмодуль *parsec* обеспечивает управление доступом на уровне стека протоколов TCP/IP (реализован для версии протокола IPv4); LSM-модуль *parsec-cifs* обеспечивает управление доступом на уровне протокола прикладного уровня *CIFS* [72] — диалекта протокола *SMB* [36] (реализован для версии протокола *SMB* 3. 0. Очевидно, что корпоративный уровень сетевой инфраструктуры является оверлейным, то есть развёртывается только поверх базового уровня.

3.6.2. **Формирование базового уровня сетевой инфраструктуры ОССН**

Table 2

arp	Работает с таблицей *ARP* (*ARP* кэшем ядра), используемой протоколом *ARP* для преобразования *IP*-адреса в MAC-адрес
dnsdomainname	Сообщает о доменном имени DNS-системы
domainname	Выдаёт или устанавливает доменное имя *NIS/YP* системы (*Network Information Service/Yellow Pages*) — службы централизованного администрирования промышленных сетей
hostname	Выдаёт или устанавливает имя хоста (сетевого узла)
ifconfig	Является основной утилитой конфигурирования сетевых интерфейсов
ipmaddr	Добавляет, удаляет или показывает широковещательные адреса сетевого интерфейса
iptunnel	Добавляет, удаляет или показывает туннели, используемые в сетевом интерфейсе
mii-tool	Проверяет или устанавливает статус интерфейсного модуля *MII* (*Media Independent Interface* — независимый мультимедийный интерфейс)
nameif	Присваивает интерфейсам имена, используя при этом MAC-адреса
netstat	Получает отчёт о сетевых соединениях, таблицах маршрутизации и статистике работы сетевого интерфейса
nisdomainname	Обладает теми же функциями, что и команда *domainname*
plipconfig	Управляет параметрами протокола *PLIP* — передачи *IP*-пакетов через параллельный порт (подключение компьютеров между собой соединением «точка-точка» через параллельные порты)
rarp	Управляет протоколом *RARP* — удалённое получение бездисковыми станциями *IP*-адреса по MAC-адресу
route	Работает с таблицей маршрутизации протокола *IP*
slattach	Подключает сетевой интерфейс к линии последовательного доступа (для использования терминальных линий при подключении компьютеров между собой соединением «точка-точка»)

Средства управления трафиком (качеством обслуживания) [30]

графические утилиты, встроенные в защищённую графическую подсистему *Fly.*

Далее будут рассматриваться команды из набора *net tools,* поскольку именно он рекомендуется в документации на ОССН в качестве основного средства администрирования базовой сетевой инфраструктуры. В текущий версии ОССН в набор *net tools* входят команды, приведенные в табл.2.

Для создания базовой сетевой инфраструктуры основными из представленных в табл. 3.2 являются команды *ifconfig, netstat, arp* и *route.* Настройка сетевых интерфейсов. Для настройки сетевых интерфейсов используется команда *ifconfig* (от *Interface Configuration*

```
root@astra-server:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:4e:79:f9
          inet addr:10.0.0.2  Bcast:10.255.255.255  Mask:255.0.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:121 errors:0 dropped:0 overruns:0 frame:0
          TX packets:76 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:18708 (18.2 KiB)  TX bytes:13592 (13.2 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:10670 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10670 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1754949 (1.6 MiB)  TX bytes:1754949 (1.6 MiB)
```

Рисунок 34.Пример вывода команды ifconfig

выполняющая следующие функции:

- назначение *IP*-адреса;
- назначение широковещательного адреса и связанной с ним маски подсети;
- включение (*up)* и выключение (*down)* сетевого интерфейса. Обычно команда *ifconfig* применяется во время первоначальной настройки сети, но может использоваться и для внесения изменений в ходе её эксплуатации. Пример вывода этой команды приведён на рис. 34.

Управлять конфигурированием сетевых интерфейсов администратор ОССН может не только с помощью команды *ifconfig,* но и с использованием графической утилиты «Сетевые соединения», вызываемой из главного пользовательского меню через меню «Панель управления» (рис. 35). Она предназначена для управления соста-

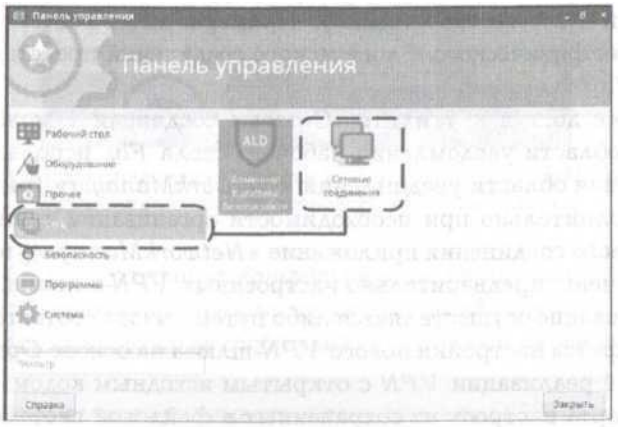


Рисунок 35. Доступ к утилите "Сетевые соединения"в меню "Панели управления"

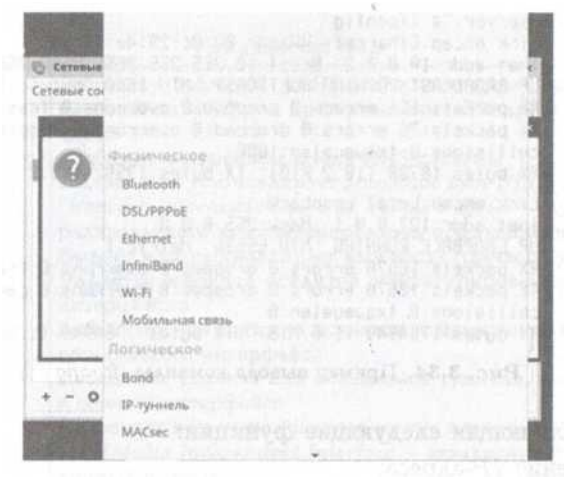


Рисунок 36.Физические и логические сетевые соединения, поддерживаемые утилитой "сетевые соединения"

вом поддерживаемых проводных и беспроводных физических или логических сетевых соединений и конфигурирования их параметров. Перечень поддерживаемых соединений представлен на рис. 36. В зависимости от выбранного типа добавляемого сетевого соединения вызывается окно его конфигурирования.

Для большинства типов сетевых соединений существуют единые параметры, объединенные во вкладки «Основное» (приоритет соединения, доступ к нему авторизованных пользователей, использование для соединения *VPN*-шлюза), «Прокси» (параметры конфигурирования прокси сервера, через который осуществляется сетевое соединение), «Параметры IPv4 и IPv6» (параметры конфигурирования сетевого протокола *IP* версий 4 и 6 соответственно). Примеры интерфейсов конфигурирования физического и логического соединений представлены на рис. 37.

Также доступ к утилите «Сетевые соединения» можно получить из области уведомлений рабочего стола *Fly,* используя меню приложения области уведомлений «*NetworkManager»* (рис. 38).

Дополнительно при необходимости организации туннелирования сетевого соединения приложение « *NetworkManager»* позволяет вызвать меню предварительно настроенных *VPN*-шлюзов. Их конфигурирование осуществляется либо путём вызова соответствующего интерфейса настройки нового *VPN*-шлюза на основе *OpenVPN* — свободной реализации *VPN с* открытым исходным кодом, либо путём экспорта настроек из сохранённых в файловой системе файлов конфигурации *VPN*-шлюзов соответствующих VPN-провайдеров

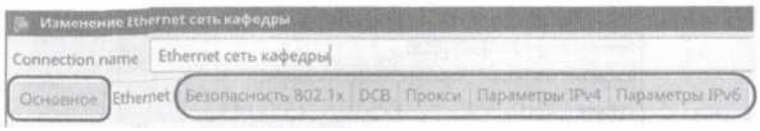


Рисунок 37

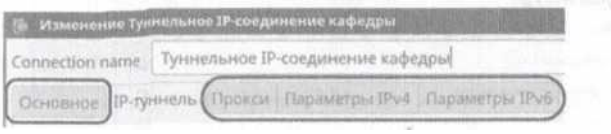


Рисунок 38

Рис. 37. Примеры интерфейсов конфигурирования параметров сетевых соединений: *а* — параметры конфигурирования физического соединения Ethernet; *б* — параметры конфигурирования логического соединения (туннельное IP-соединение)

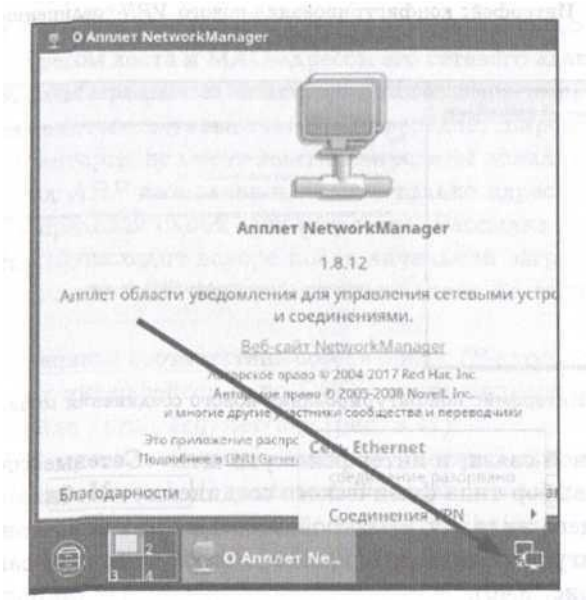


Рисунок 39

Рис. 38. Меню приложения области уведомлений ***«NetworkManager»*** для управления сетевыми устройствами соединениями

(рис. 39). При успешном конфигурировании адресной информации сетевого интерфейса в области уведомлений рабочего стола *Fly* появится соответствующее сообщение об успешном сетевом подключении.

При этом следует учитывать, что поскольку ОССН адаптирована для работы с мобильными устройствами, оборудованными моду-

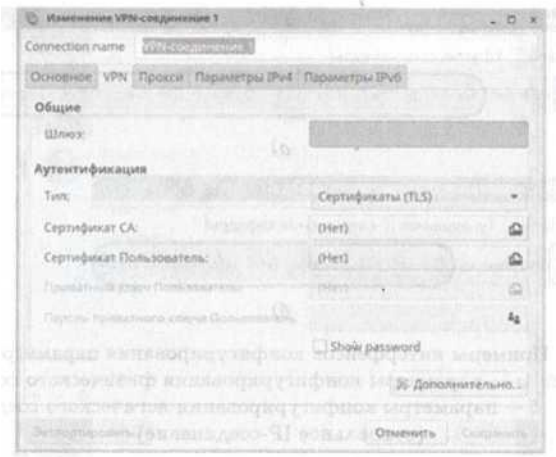


Рисунок 40.Интерфейс конфигурирования нового VPN-соединения на основе реализации OpenVPN

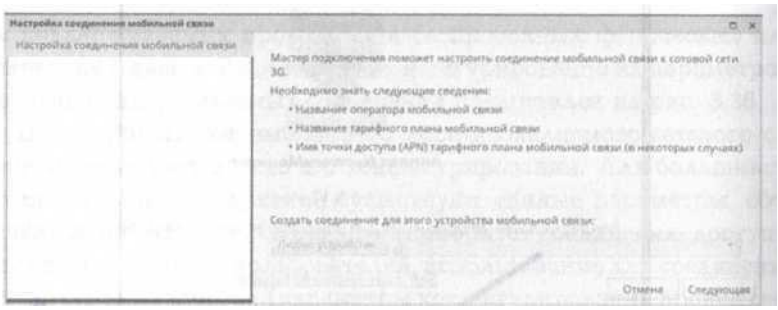


Рисунок 41. Интерфейс конфигурирования нового соединения мобильной связи

лями сотовой связи, в интерфейсе утилиты «Сетевые соединения» возможен выбор типа физического соединения «Мобильная связь», вызывающего вкладку «Настройка соединения мобильной связи» для конфигурирования интерфейса сотовой связи выбранного провайдера (рис. 40).

Проверка и настройка базового уровня сетевой инфра- структуры. Команда *netstat* отображает информацию о состоянии сетевого ПО, включая статистику сетевых интерфейсов и данные из таблицы маршрутизации. Наиболее востребованными её функциями являются:

- вывод информации о конфигурации и статистике работы сетевых интерфейсов;
- получение статических данных о сетевых протоколах;
- проверка состояния сетевых соединений;

просмотр таблицы маршрутизации.

Команда *arp* используется для просмотра, добавления и удаления записей в специальной системной таблице ядра ОССН — кэше *ARP,* в которой задано соответствие *IP*-адресов хостов сети аппаратным интерфейсам сетевых адаптеров (HW-адресам). Для интерфейсов сети *Ethernet* в кэше *ARP* устанавливается соответствие между IP-адресом хоста и MAC-адресом его сетевого адаптера. Для опроса хостов на предмет этой информации используется специальный протокол сетевого уровня *ARP.* Он передаёт широковещательные пакеты, которые не могут выйти за пределы локальной сети, то есть протокол *ARP* позволяет находить только адреса хостов, работающих в пределах одной сети (подсети). Рассылка пакетов протокола *ARP* происходит вскоре после начальной загрузки ОССН, поэтому протокол *ARP* практически не влияет на загруженность сети.

Информацию о соответствии собственных *IP*-адресов и HW-адресов сетевых интерфейсов ядро ОССН хоста сохраняет в таблице *ARP* и в файле */proc/self/net/arp* (рис. 41).

В ОССН правила маршрутизации *IP*-пакетов хранятся в системной таблице *Kernel IP routing table* ядра и в случае таблицы *ARP* отображаются в файле */proc/self/net/route* (рис. 42).

В ОС проекта *GNU/Linux* применяются следующие виды маршрутизации:

- *статическая* — когда маршруты задаются явно и хранятся в таблице маршрутизации до момента необходимости их удале- ния;
- *динамическая* — выполняющаяся демонами *routed* или *gated,* которые заполняют и модифицируют таблицу *Kernel IP routing table* на основе сообщений от других хостов сети (динамическая маршрутизация необходима в том случае, если структура сети не является статичной и меняется с течением времени, и в ней один и тот же компьютер может быть доступен по различным

```
root@astra-server:/proc# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.0.0.0        *               255.0.0.0       U     0      0        0 eth0
link-local      *               255.255.0.0     U     1000   0        0 eth0
```

Рисунок 42.Пример таблицы Kernel IP routing table командой route

\

```
# Uncomment the next two lines to enable Spoof protection (reverse-path filter)
# Turn on Source Address Verification in all interfaces to
# prevent some spoofing attacks
#net.ipv4.conf.default.rp_filter=1
#net.ipv4.conf.all.rp_filter=1

# Uncomment the next line to enable TCP/IP SYN cookies
# See http://lwn.net/Articles/277146/
# Note: This may impact IPv6 TCP sessions too
#net.ipv4.tcp_syncookies=1

# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1

# Uncomment the next line to enable packet forwarding for IPv6
#  Enabling this option disables Stateless Address Autoconfiguration
#  based on Router Advertisements for this host
#net.ipv6.conf.all.forwarding=1
```

Рисунок 43. Строка включения функции "IP forwarding" для протокола IPv4

интерфейсам, например, через разные адаптеры *Ethernet* или беспроводный интерфейс).

Команда *route* определяет только статические маршруты, записываемые в таблицу *Kernel IP routing table.* Пример вывода этой команды приведён на рис. 43.

Поскольку в состав дистрибутива ОССН не включены демоны *routed* и *gated,* динамическая маршрутизация в ней не реализуется.

В случае использования на хосте двух и более сетевых интерфейсов (например, когда хост применяется в качестве маршрутизатора или шлюза в другую подсеть) требуется указать ядру ОССН на необходимость включения функции *«IP forwarding»,* которая позволяет перенаправлять IP-пакеты с одного сетевого интерфейса на другой. Это осуществляется путём вызова команды *sysctl.* Для того чтобы функция *«IP forwarding»* применялась каждый раз при загрузке/перезагрузке ОССН, необходимо задать соответствующий параметр в конфигурационном файле */etc/sysctl.conf (*рис.44).

3.6.3. Формирование корпоративного уровня сетевой инфраструктуры ОССН. Единое пространство *пользователей*

В разд. 1.3.3 представлен вариант реализации корпоративной ЗЛВС для мультисервисной системы связи на базе ОССН. При этом рабочие станции пользователей ЗЛВС являются клиентами домен-

ной сетевой инфраструктуры, координируемой первичным контрол- лером домена *(Primary Domain Controller — PDC)* под управлением ОССН. Также было отмечено, что текущий релиз ОССН поддерживает как собственный вариант защищённой доменной сетевой инфраструктуры — *Astra Linux Directory (ALD*), отличающейся от технологии *Active Directory (AD)* [95], которая базируется на службе *Microsoft Directory Service,* так и вариант доменной сетевой инфраструктуры на базе проекта F*reeIPA*, обеспечивающего совместимость с доменной инфраструктурой *Active Directory.*

В основе указанных доменных сетевых инфраструктур лежат различные реализации следующих протоколов:

- *OpenLDAP* — реализация протокола прикладного уровня *LDAP (Lightweight Directory Access Protocol)* [56, 74] с открытым исходным кодом, обеспечивающего механизм *«I&A» (Identification and Authentication),* а также поиск, добавление, изменение и удаление записей в единый каталог сетевых объектов;
- *Samba* — реализация протокола прикладного уровня *SMB/ CIFS (Server Message Block/ Common Internet File System)* [36, 72] с открытым исходным кодом, обеспечивающего удалённый доступ к сетевым ресурсам (файлам, принтерам), а также реализацию механизма *IPC (Inter-Process Communication)* для удалённого выполнения приложений;
- *Kerberos* — протокол взаимной аутентификации хостов передустановлением соединения между ними, реализующий механизм единого входа *(Single Sign-On — SSO)* [77].

Благодаря поддержке указанных протоколов в рамках корпора- тивного уровня сетевой инфраструктуры ОССН зарегистрированные пользователи получают возможность:

- централизованного хранения данных своих учётных записей и информации о их пользовательском окружении;
- монтирования для учётных записей пользователей их домашних каталогов, расположенных на *PDC,* в состав локальной файловой системы хоста;
- сквозной аутентификации на хостах, входящих с состав доменной сетевой инфраструктуры.

Совокупность указанных возможностей обеспечивает формирование для пользователя хоста на основе ОССН, включённого в до- менную сетевую инфраструктуру, его ВПП.

В общем виде схема организации ЕПП показана на рис. 45, при этом её логическими составляющими являются:

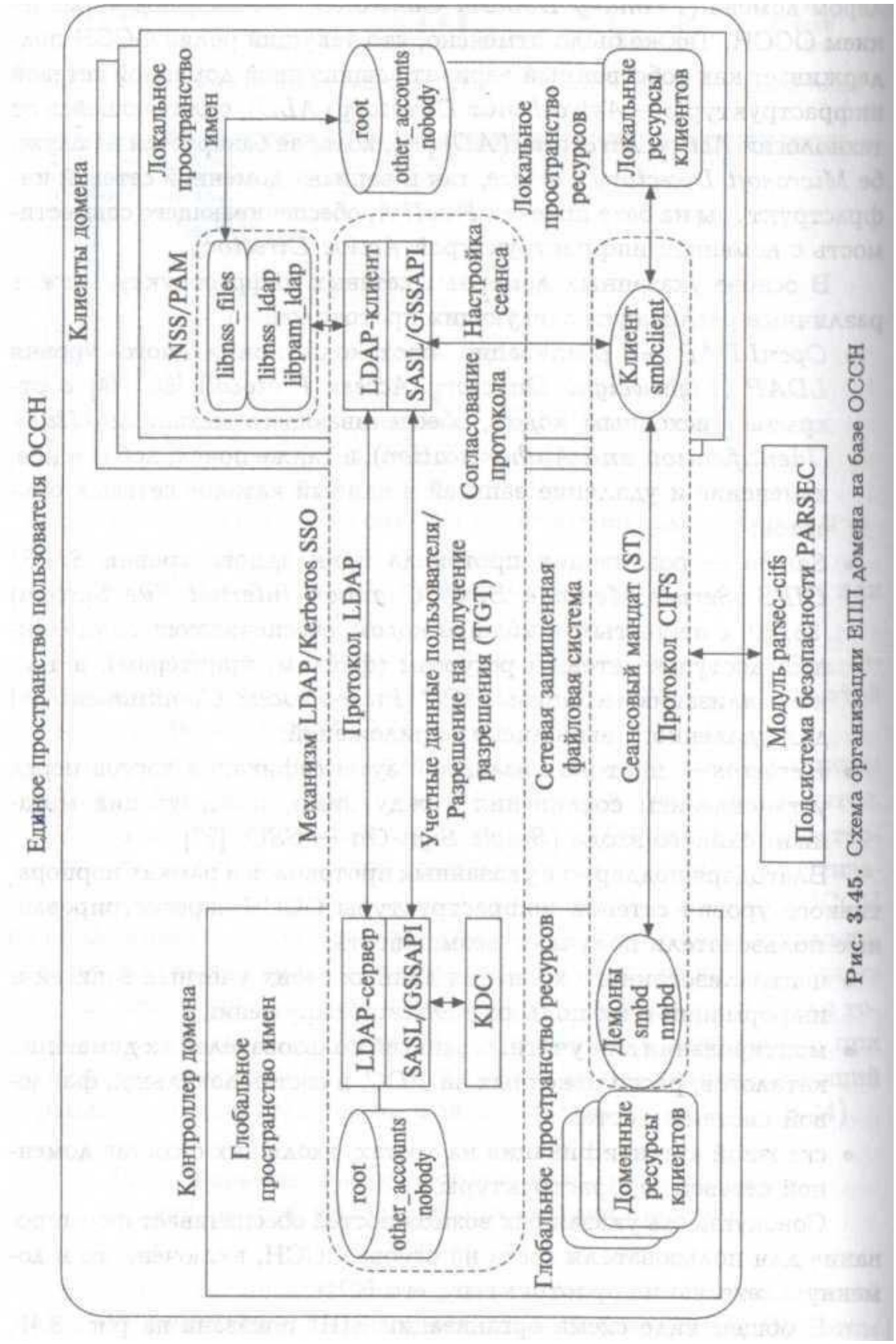


Рис. 3.45. Схема организации ЕПП домена на базе ОССН

Рисунок 44

- множество клиентов домена;
- контроллер (контроллеры) домена.

В рамках клиента домена ЕПП реализует:

- локальное пространство имён — локальные базы данных учётных записей пользователей, имеющих возможность регистрации на клиенте домена;
- локальное пространство ресурсов — локальные файловые сущностии устройства, доступ к которым процессы от имени зарегистрированных учётных записей пользователей имеют в соответствии с политикой безопасности, определённой на клиенте домена.

В рамках контроллера домена ЕПП реализует:

- глобальное пространство имён — базу данных учётных записей пользователей, имеющих возможность регистрации в доменной сетевой инфраструктуре;
- глобальное пространство ресурсов — удалённые файловые сущности и устройства, располагаемые на контроллере домена и/или специально сконфигурированных клиентах домена — файловых серверах, доступ к которым процессы от имени зарегистрированных учётных записей пользователей имеют в соответствии с политикой безопасности, определённой в доменной сетевой инфраструктуре.

Реализацию политики безопасности в рамках доменной сетевой инфраструктуры выполняет LSM-модуль *parsec-cifs* подсистемы безопасности *PARSEC*, который инициализируется на контроллере и клиентах домена в процессе инсталляции на них ОССН с функционалом контроллера и клиента домена, соответственно. По своим функциональным возможностям модуль *parsec-cifs* идентичен модулю *parsec*, функционирующему в случае применения ОССН на компьютерах, не входящих в состав доменной сетевой инфраструктуры.

Единое пространство имён доменной сетевой инфраструктуры формируется из локального и глобального пространств имён, расположенных соответственно на клиенте и контроллере домена. При этом в пределах единого пространства имён пользователь домена получает возможность как локальной регистрации на конкретном клиенте домена, так и централизованной регистрации на контроллере домена. В зависимости от вида регистрации процессы от имени учётной записи пользователя получают доступ к локальным ресурсам или ресурсам узлов домена в соответствии с локальной политикой безопасности или политикой безопасности домена.

В ОССН локальное и глобальное пространства имён (систем учётных записей пользователей и групп пользователей) функционируют параллельно. Для их различения выполнено разграничение диапазонов *UID:* значения *UID* меньшие 2500 относятся к локальному пространству имён, а значения *UID* больше либо равные 2500 — к глобальному пространству имён.

Механизм *«I&A»* в пределах локального пространства имён на клиенте домена под управлением ОССН реализуется с использованием архитектуры *PAM,* особенности использования которой рассмотрены в предыдущих параграфах главы. Реализация этого механизма в рамках глобального пространства имён в домене на базе ОССН основана на инфраструктуре *LDAP* (хранение и администрирование учётных записей пользователей домена), функционирующей совместно с протоколом *Kerberos.*

База данных учётных записей глобального пространства имён реализована в виде «Дерева директорий информации» (*Directory Information Tree — DIT).* Подобная модель хранения данных основана на записях, содержащих наборы атрибутов, различающиеся по «Отличительному имени» (*Distinguished Name — DN),* которые используются для обеспечения однозначности при обращении к записи.

Каждый из атрибутов записи относится к конкретному типу и имеет одно или несколько значений. Типы обычно представлены строковыми переменными.

DIT глобального пространства имён контроллера домена на базе ОССН разделено на две *DN-ветви: ou = People* и ou = *Group,* которые свою очередь имеют дочерние *DN*-ветви, имеющие тип cn и содержащие атрибуты учётных записей самих пользователей домена, групп пользователей домена и их теневых паролей. В общем *DIT* ресурсе контроллера домена указанные DN-ветви включены в *DN*- ветви *dc = engine u dc = local.* В графическом виде общая структура *DIT* контроллера домена на базе ОССН показано на рис. 46.С учётом МРОСЛ ДП-модели атрибуты *DIT*-дерева дополняются DN-ветвями, определяющими информационные сервисы для мандатных уровней конфиденциальности и целостности. Эти дополнительные атрибуты находятся в файле */etc/parsec/mldap.conf* (рис. 47).

В зависимости от необходимости использования локальной или глобальной реализации механизма *«I&A»* на клиенте домена производится их динамическое переключение, основанное на технологии *NSS (Name Service Switch)* [80]. Эта технология создаёт модульное окружение для управления пользовательскими учётными записями,

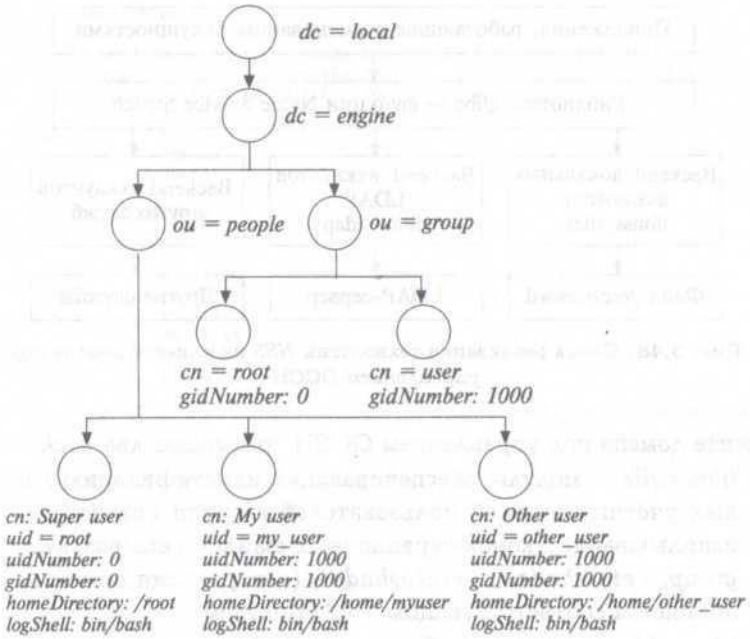


Рисунок 45. Структура DIT-дерева, реализующего глобальное пространство пользователей домена на базе ОССН

```
# LDAP connection timeout (in seconds)
timeout 10

# Kerberos realm
krb_realm ald.domain

# Authentification type (for user MAC attrs). Possible values:
#       simple  - for anonymous bind to LDAP (only for debug)
#       krb     - for Kerberos authentication.
# Bind to MAC level and MAC category databases is always anonymous.
# Examples:
# auth_type simple
# auth_type krb
auth_type krb

# Base for mandate information services
nss_base_mac            ou=users,dc=ald_domain
nss_base_mic            ou=users,dc=ald_domain
nss_base_capabilities   ou=users,dc=ald_domain
nss_base_audit          ou=audit-policies,ou=ald-config,dc=ald_domain
nss_base_mac_levels     ou=mac-levels,ou=mac-services,ou=service-configs,dc=ald_domain
nss_base_mac_categories ou=mac-categories,ou=mac-services,ou=service-configs,dc=ald_domain
```

Рисунок 46. Дополнительные атрибуты DIT-дерева, определяющие информационные сервисы для реализации МРОСЛ ДП-модели

реализованное в виде набора загружаемых библиотек *(backends).* При этом базовые системные вызовы при применении технологии *NSS* реализованы в библиотеке *glibc,* которая в зависимости от кон- фигурации *NSS* вызывает те или иные *backend* (рис. 48).Функции библиотеки *eglibc,* реализующие технологию *NSS* на

Рисунок 47.Схема реализации технологии NSS на клиенте домена под управлением ОССН

клиенте домена под управлением ОССН, вызывают два *backend:*

· *libnss_file* — модуль, обеспечивающий идентификацию локальныхучётных записей пользователей и групп пользователей с использованием конфигурационных файлов */etc/passwd, /etc/ group, / etc/ shadow, / etc/gshadow* (авторизация при этом выполняется соответствующим *PA*M-модулем);

· *libnss_ldap* — модуль, обеспечивающий идентификацию учётных записей пользователей и групп пользователей домена с использованием соответствующих DN-ветвей *DIT*-дерева на контроллере домена (авторизация при этом может выполняться как с использованием *PAM*-модуля *libpam_ldap* — связка *NSS/ PAM,* так и с использованием сквозной аутентификации по протоколу *Kerberos* — метод аутентификации по умолчанию). Конфигурация *NSS* задаётся в файле /*etc/nsswitch.conf.* Таким образом, при выполнении процессов, требующих обращения к именованным сущностям, соответствующие вызовы функций библиотеки *glibc* будут обращаться к функциям *backend,* указанным в файле /*etc/nsswitch.conf.* Пример файла /*etc/nsswitch. conf* клиента домена на базе ОССН приведён на рис. 3.49. Кроме имён и идентификаторов учётных записей пользователей и групп пользователей технология *NSS* позволяет определять имена и идентификаторы протоколов, номера портов сервисов, JP- адреса и имена хостов, а также другие данные. В частности, применительно к МРОСЛ ДП-модели технология *NSS* позволяет определять источники данных для задания мандатных уровней конфиденциальности и целостности, для этого используется конфигурационный файл *etc/parsec/mswitch.conf* подсистемы безопасности *PARSEC* (рис. 50).

```
# /etc/nsswitch.conf
#
# Example configuration of GNU Name Service Switch functionality.
# If you have the `glibc-doc-reference' and `info' packages installed, try:
# `info libc "Name Service Switch"' for information about this file.

passwd:         compat
group:          compat
shadow:         compat

hosts:          files dns
networks:       files

protocols:      db files
services:       db files
ethers:         db files
rpc:            db files

netgroup:       nis
```

Рисунок 48.Пример конфигурационного файла NSS на клиенте домена

```
# The text names of mandate levels (like "not secret", "secret",
# "top secret" etc.). The default source is
# /etc/parsec/mac_levels file.
mac_levels: files
mac_levels_charset: utf-8

# The text names of mandate categories (like "tanks", "planes",
# "submarines" etc.). The default source is
# /etc/parsec/mac_categories file.
mac_categories: files
mac_categories_charset: utf-8

# The mandate integrity level allow to user. The default source
# is entries in /etc/parsec/micdb directory.
mic: files
mic_charset: utf-8
```

Рисунок 49. Конфигурационный файл для определения источников данных на клиенте домена

Для осуществления сквозной аутентификации пользователей домена на базе ОССН применяется протокол *Kerberos.* В рамках доменной инфраструктуры, основанной на применении *OpenLDAP* (представлена демоном *slapd)* и протокола *Kerberos,* механизм единого входа реализуется с использованием технологии *SASL (Simple Authentication and Security Layer)* [81]. В частности, для версии *MIT Kerberos* 5, используемой в ОССН, в рамках технологии *SASL* применён механизм *GSSAPI (The Kerberos Version 5 Generic Security Service Application Program Interface Mechanism*) [42]. Таким образом, модель механизма единого входа в ОССН именуется *LDAP based on Kerberos (SASL/GSSAPI).*

Конфигурационным файлом механизма *GSSAPI* является файл */etc/gssapi_mech.conf* в котором определяется функция инициализации механизма при вызове библиотеки *Kerberos* 5 *GSSAPI* (рис. 47). При этом общим конфигурационным файлом реализа-

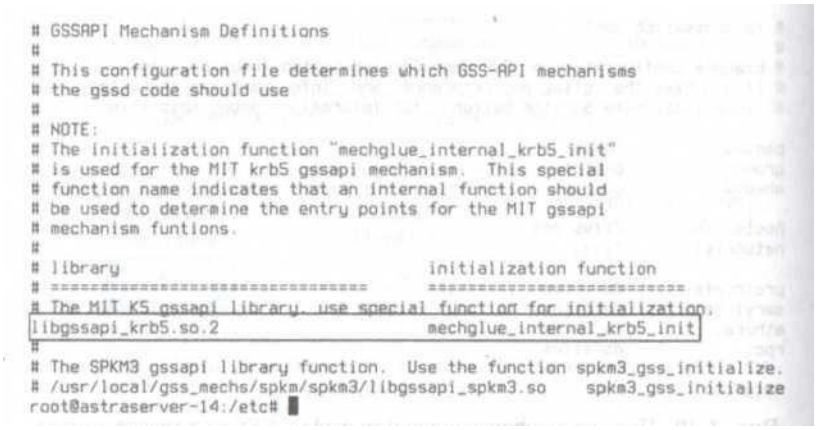


Рисунок 50. Конфигурационный файл механизма Kerberos 5 GSSAPI

ции протокола *MIT Kerberos* 5 является файл */etc/krd5.conf,* конфигурация *KDC* задаётся в файле /*etc/krb5kdc/kdc.conf.*

Организация механизма *«I&A»* в пределах глобального пространства имён ЕПП, реализуемого в ОССН, обеспечивает возможность пользователям доменной сетевой инфраструктуры на базе ОССН получать доступ к глобальному пространству ресурсов, поддержка которого возложена на модифицированную реализацию пакета программ *Samba* [57], именуемую как сетевая защищённая файловая система (СЗФС). СЗФС может функционировать как в составе контроллера домена (выполняющего при этом дополнительные функции файлсервера), так и в составе клиентов домена, реализующих функции файлсервера для предоставления доступа к собственным разделяемым ресурсам. Поскольку СЗФС является модификацией пакета программ *Samba*, она состоит из следующих компонент:

- серверного набора программ: демоны *smbd* и *nmbd;*
- клиентского приложения: *smbclient*;
- набора утилит администрирования: *testparam* и *smbstatus*;
- конфигурационных файлов: */etc/samba/smb.conf u /etc/smb- netfs. conf.*

Дополнительно для администрирования СЗФС имеется графическая утилита «Общие папки *(Samba)» (fly-admin-samba),* расположенная в меню «Настройки» главного пользовательского меню.

Базовыми компонентами СЗФС являются демоны *smbd* и *nmbd,* а также клиентское приложение *smbclient.* Демон *smbd* реализует функции сервиса сетевой печати и разделения файлов для клиентских приложений *smbclient,* функционирующих в рамках доменной сетевой инфраструктуры на базе ОССН, а также клиентских прило-

жений, функционирующих под управлением ОС семейства *Microsoft Windows.* Конфигурация демона *smbd* в ОССН задаётся в файле */etc/samba/smb.conf.* Демон *nmbd* по умолчанию реализует функции сервиса имён протокола *NetBIOS,* а также может использоваться для запроса других сервисных служб имён.

3.6.4. Служба *ALD*. Администрирование доменной сетевой инфраструктуры ОССН

Рассмотренные компоненты доменной сетевой инфраструктуры ОССН требуют конфигурирования в процессе её установки на хос- ты, реализующие контроллеры и клиенты домена, а также адми- нистрирования в процессе эксплуатации домена. Для этого в ОССН имеется служба *ALD,* схема которой показана на рис. 3.52.

Таким образом, служба *ALD* состоит из следующих компонент. *Базовые компоненты,* представленные пакетами программ конфигурирования:

- контроллера *ALD;*
- клиента *ALD;*
- хоста, выполняющего функции узла администрирования *ALD* (может располагаться на одном контроллере *ALD* или одном из клиентов *ALD*);
- клиента *ALD,* реализующего функции файл-сервера (может также располагаться на контроллере *ALD).*

Так как ряд функций базовых компонентов являются обяза- тельными для различных ролей хостов домена *ALD,* указанные па- кеты собираются в кумулятивные пакеты. Например, кумулятивный пакет контроллера домена *ALD — ald-server-dc* содержит в своём составе пакеты клиента и администратора *ALD.*

Расширения службы ALD, предназначенные для обеспечения большей функциональности базовых компонентов службы *ALD* за счёт подключения дополнительных функций, например, следующих:

- организация и администрирование резервных контроллеров до- мена *ALD* и установление междоменных отношений доверия;
- централизованное хранение настроек аудита и ОССН в целом;
- конфигурирование подсистемы управления доступом к подклю- чаемым носителям.

Наименование пакета расширения отражает его назначение, на- пример:

- *ald-client-...* — расширение клиентской части домена *ALD*;
- *ald-admin-...* — расширение администрирования домена *ALD;*

Рисунок 51

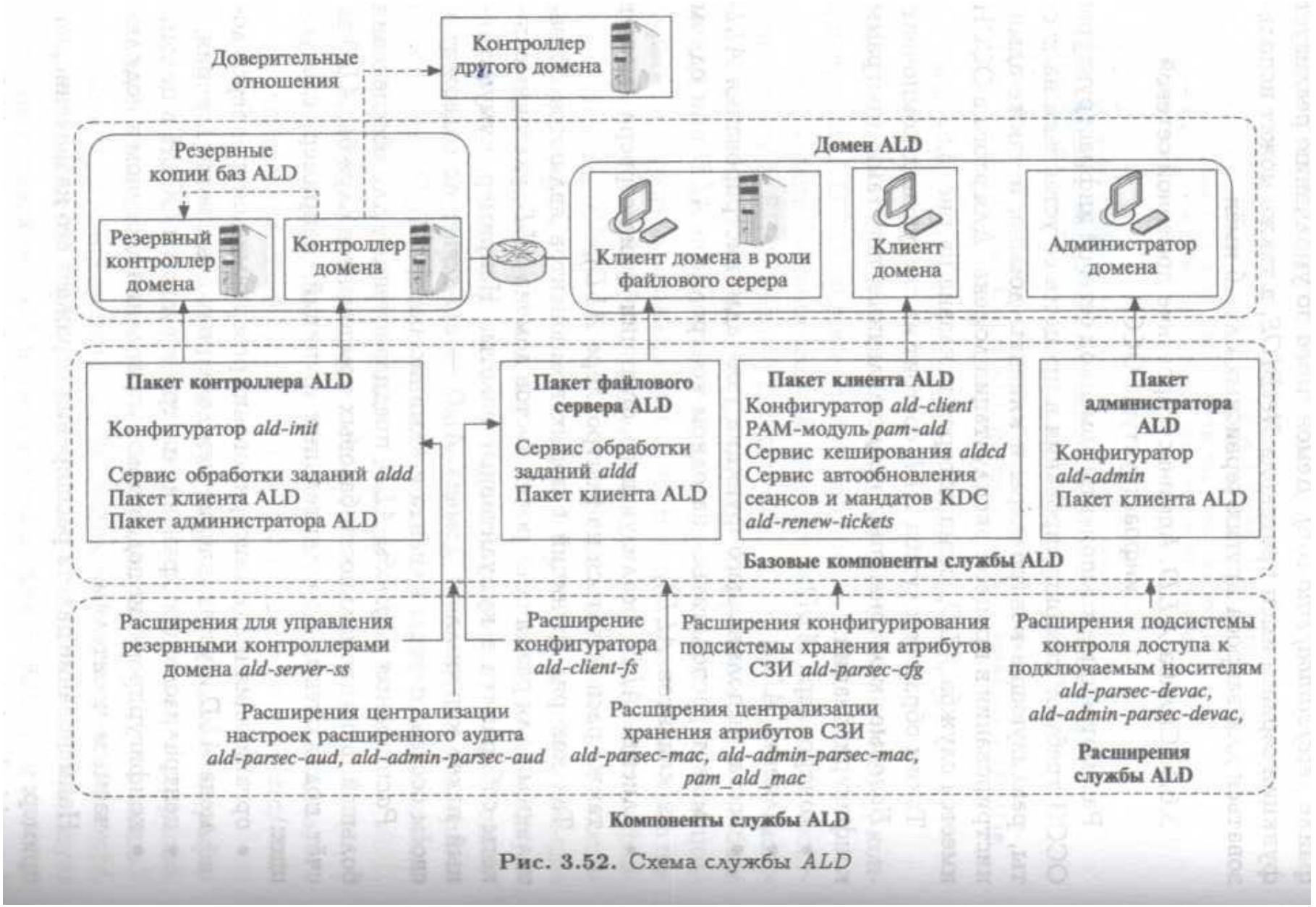


Рис. 3.52. Схема службы *ALD*

- *ald-server-...* — расширение для организации хранения настроек на контроллере домена *ALD.*

Администрирование домена *ALD* выполняется пользователями, обладающими соответствующими полномочиями. В зависимости от назначенных привилегий администраторов домена *ALD* можно раз- делить на следующие группы:

- *корневой администратор* (имя *admin/admin)* — основной администратор домена *ALD,* обладающий всеми полномочиями по управлению доменом;
- *администраторы* — пользователи с привилегией *admin,* обладающие полномочиями по управлению конфигурацией домена и учётными записями пользователей домена;
- *ограниченные администраторы* — пользователи с привилегиями *hosts-add* или *all-hosts-add,* обладающие полномочиями по добавлению хостов в состав домена;
- *пользователи утилит администрирования* — пользователи с привилегией *adm-user,* обладающие полномочиями по запуску утилит администрирования.

Базовый администратор домена *ALD,* используя набор программ базовых компонентов службы *ALD,* а также их расширений и графическую утилиту «Управление политикой безопасности» (*fly- admin-smc)* рабочего стола *Fly,* может выполнять следующие операции:

- создание нового домена;
- резервирование/восстановление конфигурации домена;
- контроль целостности конфигурации домена;
- добавление/удаление хостов в состав хостов домена;
- управление учётными записями пользователей домена;
- управление учётными записями сетевых служб домена;
- управление параметрами ОССН, в первую очередь соответст- вующими МРОСЛ ДП-модели.

В общем случае методику конфигурирования доменной сете- вой инфраструктуры на базе ОССН с использованием службы *ALD* можно разделить на следующие этапы.

1. Настройка сетевого соединения на контроллере домена и хостах рабочих станций пользователей.
2. Настройка именования сервера и клиентов *ALD.*
3. Конфигурирование и запуск сервера *ALD* на хосте, реализующем функции контроллера домена.
4. Запуск клиентов *ALD* на хостах рабочих станций.

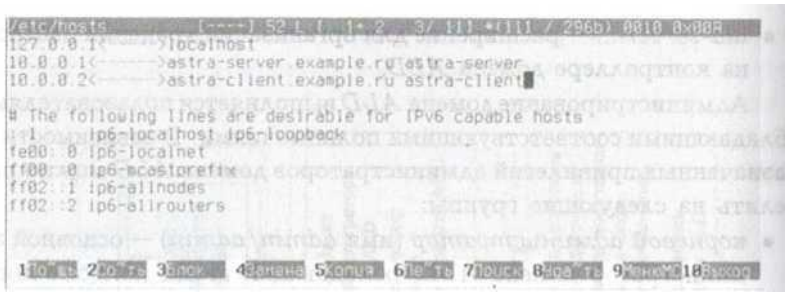


Рисунок 52. Пример конфигурирования файла /etc/hosts/ на хосте, реализующем функции контроллера домена

На первом этапе учитывается, что способ сетевого соединения контроллера и клиентов домена *ALD* зависит от типа адресации, используемой в сетевом сегменте. Статическая адресация целесообразна при небольшом числе хостов, входящих в состав домена *ALD.* При этом сетевой интерфейс каждого хоста, входящего в домен *ALD,* конфигурируется индивидуально. Динамическая адресация целесообразна при значительном числе хостов, входящих в состав домена *ALD* или их большой территориальной удалённости.

В этом случае на контроллере *ALD* (или на другом специально выделенном хосте) конфигурируется и запускается сервер динамической сетевой адресации *DHCP (Dynamic Host Configuration Protocol).*

На втором этапе для функционирования домена *ALD* обеспечивается настройка сетевых имён таким образом, чтобы сетевое имя хоста разрешалось, в первую очередь, как полное имя (например, *astraserver.example.ru).* При этом команда *hostname* должна возвращать короткое сетевое имя хоста (например, *astra-server).*

При небольшом количестве хостов, входящих в домен *ALD,* такую настройку, как правило, выполняют вручную, конфигурируя файл /*etc/hosts* (рис. 53). В случае большого количества хостов в составе домена *ALD* на контроллере домена (или на специально выделенном хосте) конфигурируют службу доменных имён *DNS (Domain Name Service).*

На третьем этапе конфигурируют серверную и клиентскую части домена *ALD,* реализованные в виде одного демона *aldd:*

- путём редактирования конфигурационного файла */etc/aid/ ald.conf* (рис. 54);
- запуская или повторно инициализируя демон *aldd* с помощью команд *ald-init* (на контроллере *ALD)* и *ald-client* (на клиенте *ALD)* (рис. 55);

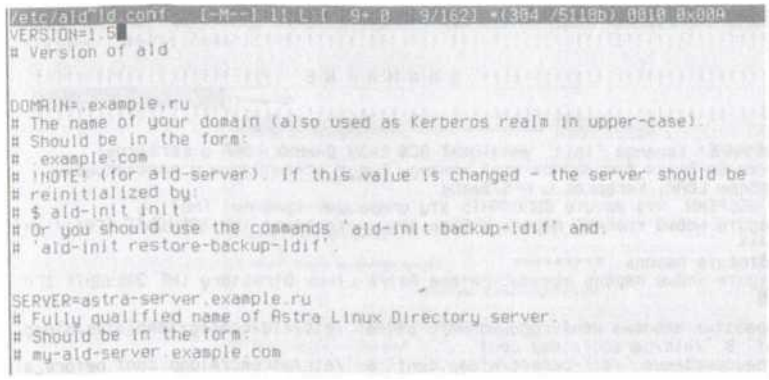


Рисунок 53.Пример конфигурирования файла /etc/aid/aid.conf на хосте, реализующем функци контроллера домена

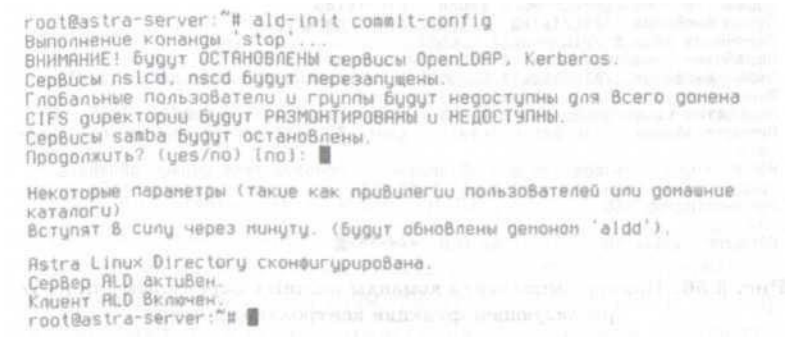


Рисунок 54. Пример выполнения команды aid-init с параметром commit-config на хосте, реализующем функции контроллера домена

- конфигурируя параметры пароля базового администратора *ALD* для реализации сквозной аутентификации на контроллере *ALD* (рис. 56).

Эти действия осуществляются путём редактирования параметров файла */etc/aid/ald.conf* (при этом в большинстве случаев для всех этих параметров используются значения по умолчанию).

Обязательной модификации подлежит параметр *SERVER,* в качестве значения которого указывается выбранное на втором этапе имя сервера(в примере на рис. 54 это *astra-server.example.ru).*

На четвёртом этапе базовый администратор домена с использованием команды *ald-admin* или графической утилиты «Доменная политика безопасности», расположенной в разделе «Сеть» меню «Панель управления» (рис. 57), создаёт или редактирует учётные записи пользователей домена, управляет компьютерами домена, а также политиками безопасности на активном сервере *ALD.*

Рисунок 55. Пример выполнения команды ald-init с параметром init на хосте, реализующем функции контроллера домена

После настройки и запуска сервера *ALD* и клиентов домена администратор *ALD* может выполнять следующие функции:

- создание и/или администрирование учётных записей пользователей *ALD*;
- создание и/или администрирование групп пользователей *ALD;*
- добавление и удаление хостов рабочих станций в/из состава домена;
- резервирование/восстановление учётной информации баз данных домена;
- конфигурирование привилегий и параметров безопасности ОССН для учётных записей пользователей и групп пользователей домена;
- конфигурирование политик паролей *Kerberos*;
- администрирование доступа к съёмным устройствам;
- администрирование учётных записей сетевых служб (сервисов) домена;
- контроль целостности (аудит) конфигурации домена.

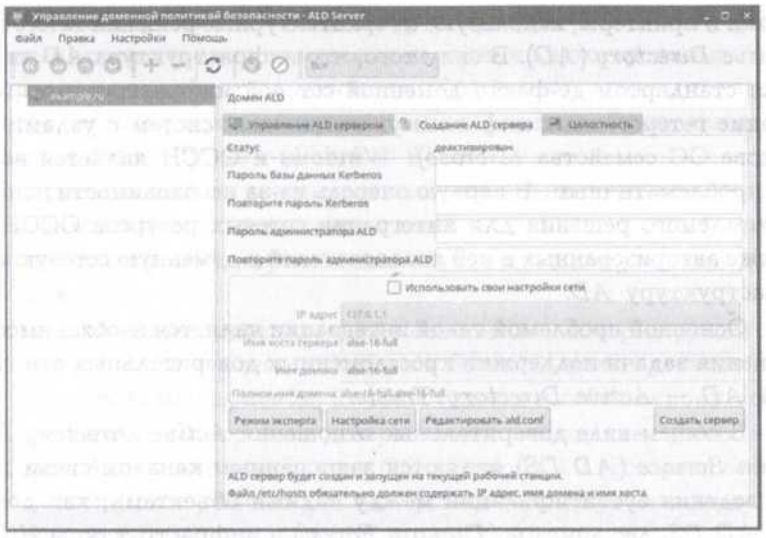


Рисунок 56.Интерфейс конфигурирования сервера ALD для домена "example.ru"

В условиях необходимости интеграции клиентов и сервисов доменной сетевой инфраструктуры ОССН с доменной инфраструктурой, основанной на технологии *Active Directory*, сформированное в рамках технологии *FreeIPA* пространство пользователей расширяется за счет доступа к объектами и сервисам доменной инфраструктуры основанной на технологии *Active Directory,* включая возможности доверительных отношений *AD Trusts* [96].

3.6.5. Служба *FreeIPA.* Формирование гетерогенной доменной сетевой инфраструктуры

В общем случае решение на основе доменной сетевой инфраструктуры *ALD* является эффективным в рамках инфокоммуникационных систем, узлы которых целиком базируются на ОССН. К таким системам, в частности, следует относить различного типа ЗЛВС, а также АСЗИ, ориентированные на обработку информации с различным уровнем конфиденциальности.

Между тем существует широкий сегмент инфокоммуникационных систем, в которых доля узлов, базирующихся на решениях ОС семейства *Microsoft Windows,* является достаточно высокой. В таком сегменте в качестве способа централизованного управления идентификационными и аутентификационными данными пользователей, а также учёта доступных сетевых ресурсов, таких как общие

палки и принтеры, используются архитектурные решения *Microsoft Active Directory (AD).* В силу того, что инфраструктура *AD* является стандартом де-факто доменной сетевой инфраструктуры, со- здание гетерогенных инфокоммуникационных систем с узлами на основе ОС семейства *Microsoft Windows* и ОССН является весьма проблематичным. В первую очередь из-за необходимости поиска приемлемого решения для интеграции сетевых ресурсов ОССН, а также авторизованных в ней пользователей в доменную сетевую инфраструктуру *AD.*

Основной проблемой такой интеграции является необходимость решения задачи поддержки кроссдоменных доверительных отношений *AD — Active Directory Trusts.*

В общем виде доверительные отношения *Active Directory Domain Service (AD DS)* являются защищённым каналом связи для проведения аутентификации между такими объектами, как домены *AD DS,* лес доменов *(Domain Forest)* и инфраструктуры *UNIX Realms*, реализованной на базе *Realmd (Realm Discovery)* [45] — *D-Bus* сервиса, позволяющего выполнять настройку аутентификации и членства в домене. В целом, наличие доверительных отношений позволяет организовать междоменный доступ к ресурсам, пользователям, группам и узлам инфокоммуникационных систем.

Виды доверительных отношений *Active Directory Trusts* представлены на рис. 46.

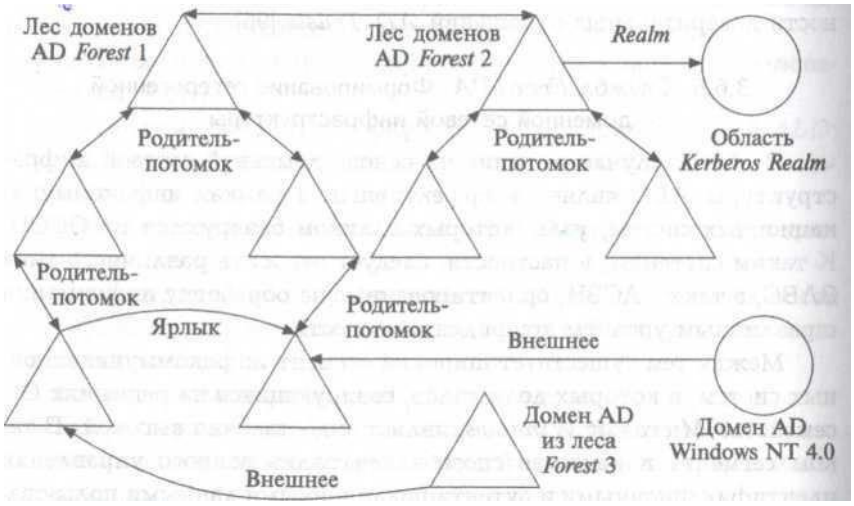


Рисунок 57.Виды доверительных отношений AD Trusts

Таким образом, в рамках инфраструктуры *AD DS* внутрифирменные доверительные отношения являются двухсторонними отношениями типа «родитель-потомок». Кроме того, домены в рамках одного леса могут устанавливать односторонние доверительные отношения типа «Ярлык» (*Shortcut Trusts).*

Кроссдоменные доверительные отношения (*Cross-Domain Trusts)* относятся к следующим типам:

- двухсторонние доверительные отношения между лесами доменов *AD]*
- односторонние доверительные отношения, которые домен *AD* (на рис. 46 — домен *AD* из леса *Forest* 3) может установить с произвольным доменом, не относящимся к собственному лесу;
- внешнее одностороннее доверительное отношение, которое домен *AD,* созданный на базе ОС *Windows* 4.0 (устаревшее технологическое решение), может установить с произвольным доменом *AD* произвольного леса доменов;
- внешнее одностороннее отношение *Realm*, которое лес доменов *AD* (на рис. 46 — лес *Forest* 2) может установить с доменом на базе инфраструктуры *UNIX Realms.*

В общем случае доверительные отношения *AD* являются двухэтапным процессом. На первом этапе происходит установление од- ного из перечисленных выше типов доверительных отношений. На втором этапе, в рамках установленного типа доверительных отно- шения выполняется предоставление соответствующих разрешений на доступ к ресурсам *AD.*

Если между доменами *AD* 1 и 2 установлены доверительные отношения, то пользователи домена 1 могут аутентифицироваться на контроллере домена 2 с использованием специальной учётной записи, расположенной на контроллере домена 2. Двунаправленность таких отношений означает, что на контроллере домена 1 поддерживаются специальные учётные записи пользователей домена 2. Такие специальные учётные записи создаются с использованием вызовов протокола *MS-RPC.*

В рамках решения задачи предоставления разрешений на доступ к ресурсам для установленных доверительных отношений меж- ду доменами или лесами доменов *AD* действует *принцип тран-зитивности,* согласно которому, если доверительные отношения установлены между доменом 1 и доменом 2, а также между доменом 2 и доменом 3, то считается, что между доменами 1 и 3 также установлены доверительные отношения.

Такие возможности достигаются за счёт того, что пользователи и группы пользователей инфраструктуры *AD DS* идентифицируются едиными идентификаторами безопасности *(Secure ID — SID),* являющимися уникальными для каждого домена *AD. И*дентификаторы *SID* используются для управления доступом к ресурсам домена *AD* в рамках сеансов протокола *SMB.* Проверка *SID* позволяет получить доступ к списку *ACL,* связанному с запрашиваемым ресурсом или *A*PI-функциями запрашиваемого сетевого сервиса. Организация доверительных отношений между двумя доменами *AD* фактически означает делегирование функции преобразования имени пользователя в его *SID* и наоборот контроллеру домена *AD,* который владеет сетевым ресурсом, к которому пользователь осуществляет доступ. Такое единообразное управление доступом на основе единого пространства идентификаторов *SID* позволяет любой протокол аутентификации, например, аутентификация с использованием серве- ра *LDAP* или аутентификация *Kerberos.* В этом случае важным является, чтобы в рамках этих протоколов реализовывался поиск требуемого идентификатора *SID.*

Как было рассмотрено в разд. 1.3, в целях интеграции сетевых ресурсов ОССН, а также организации членства пользователей, авторизованных в ОССН, в доменной сетевой инфраструктуре *AD DS,* в релизе 1.6 ОССН реализована поддержка доменной инфраструктуры *FreeIPA.* Эта инфраструктура разрабатывалась для организации поддержки членства клиентских узлов на базе ОС семейства *Microsoft Windows* в домене *IPA.* Начиная с версии 3.0 *(IPAv3),* доменная инфраструктура *FreeIPA* поддерживает кроссдоменные доверительные отношения *Cross-Domain Trusts.*

Такая поддержка является возможной, поскольку инфраструктуры *AD DS* и *FreeIPA* можно рассматривать как *Kerberos Realms.* Это означает, что в домене *AD* можно настроить кроссдоменные доверительные отношения для *MIT Kerberos Realm,* а именно *Kerberos* 5, являющейся основой сервиса аутентификации *FreeIPA.*

Поскольку при организации кроссдоменных доверительных отношений между доменами *AD DS* и *MIT Kerberos Realm* фактически отсутствует контроллер домена *AD* в *MIT Kerberos* реализован механизм управления идентификаторами *SID* на основе их сопоставления с локальными учётными записями пользователей.

В рамках сеанса *SMB,* поддерживаемого инфраструктурой *AD DS,* пространство имён специальных учётных записей для реализации доверительных отношений является совместимым как с серви-

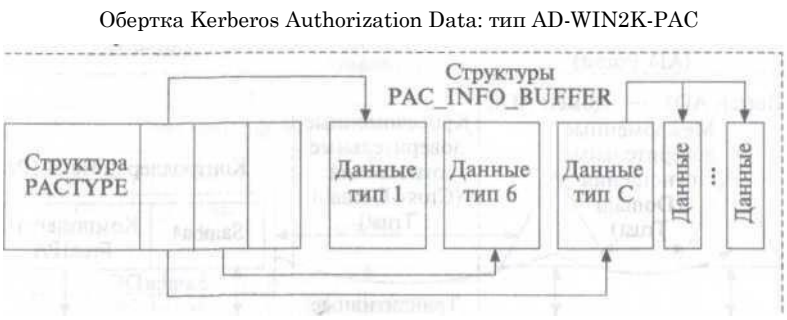


Рисунок 58.Формат упаковки структуры MS-PAC в формат Kerberos Ticket

сом идентификации контроллеров *AD,* так и с сервисом *Kerberos KDC.*

Совместимость достигается за счёт того, что имена пользователей *MIT Kerberos Realm* дополняются именем домена *DNS.* То есть, контроллер *AD* домена *AD*1 будет именовать себя для контрол- лера *MIT Kerberos AD2* как *ADl$©AD2.DOMAIN,* а контроллер *MIT Kerberos AD2* будет сопоставлять это имя с локальной учётной записью *AD1.*

Протокольный блок данных авторизации *MS-PAC,* используемый в инфраструктуре *AD DS* для поддержки пространства имён специальных учётных записей, в случае взаимодействия с *MIT Kerberos Realm* использует специальные обёртки, для упаковки его в формат билета *Kerberos (Kerberos Ticket).* Формат такой упаковки представлен на рис. 3.59.

В обобщённом виде возможности интеграции доменной инфраструктуры *FreeIPA* и инфраструктуры на базе *AD DS* представлены на рис. 3.60.

Таким образом, основой доменной инфраструктуры *IPA,* как и в случае доменной сетевой инфраструктуры *ALD,* является *Samba* — пакет программ кроссплатформенной поддержки доступа к сетевым ресурсам с использованием протокола *SMB/CIFS.* При этом, начиная с версии *Samba* 4, в них была добавлена роль контроллера домена *Samba Domain Controller (Samba DC),* реализующая частичную функциональность сервиса *AD DS,* а именно:

- сервис аутентификации на базе *Kerberos* 5;
- сервис каталогов, поддерживающий совместимость с протоколом *LDAP* и модель репликации *Active Directory Replication Model,*
- сервис управления групповыми политиками на основе объектов групповой политики (*Group Policy Object — GPO;*
- *DNS*-сервер на основе реализации *BIND.*

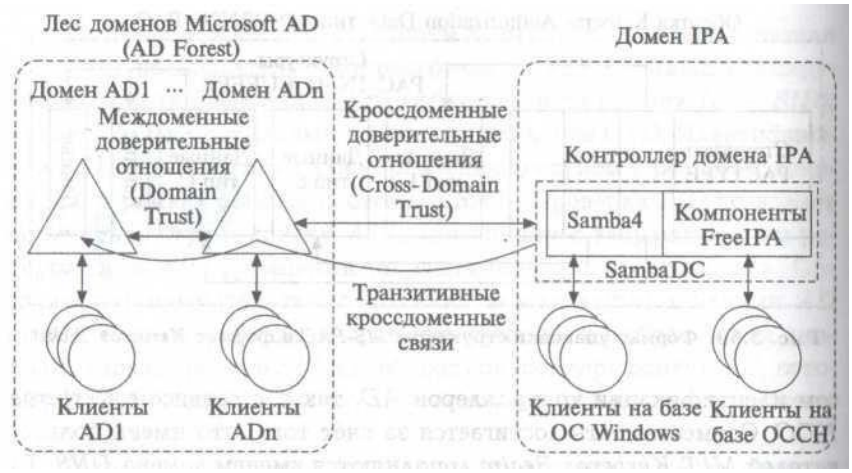


Рисунок 59. Структура интеграции доменных сетевых инфраструктур AD DS и IPA

Возможности контроллера *Samba DC* дополняются компонентами *IPA,* обеспечивающими поддержку отношений *Cross-Domain Trusts.* На рис.61 представлена структура контроллера домена *IPA* (ядра *FreeIPA)* и связи между её элементами.

Таким образом, базовыми компонентами *FreeIPA* являются:

1. Демон *empd* (*End Point Mapper — EPM)* — сервис прямого подключения узлов на базе ОС семейства *Microsoft Windows* к службам *MS-RPC.*
2. Сервисы *Netlogon, LSA* и *SAMR,* обеспечивающие в совокупности организацию контроллеру домена инфраструктуры *AD DS* доверительных отношений с контроллером домена *IPA.* Они также реализуют управление данными специальных учётных записей пользователей в каталоге *LDAP,* в частности следующие функции;

· *Netlogon* — сервис организации защищённого канала кроссдоменного взаимодействия для выполнения сквозной аутентификации как на этапе создания доверительных отношений, так и на этапе кроссдоменного взаимодействия;

· подсистема локальной авторизации *LSA* (*Local Security Authority)* — реализует алгоритмы служб перевода имён *name2sid* и *sid2name,* реализующих трансляцию идентификаторов *SID* и локальных имён *MIT Kerberos Realm*;

· сервис *SAMR* (*Security Account Manager Remote),* используемый для удалённого управления специальными учётными записями попротоколу *MS-RPC*;

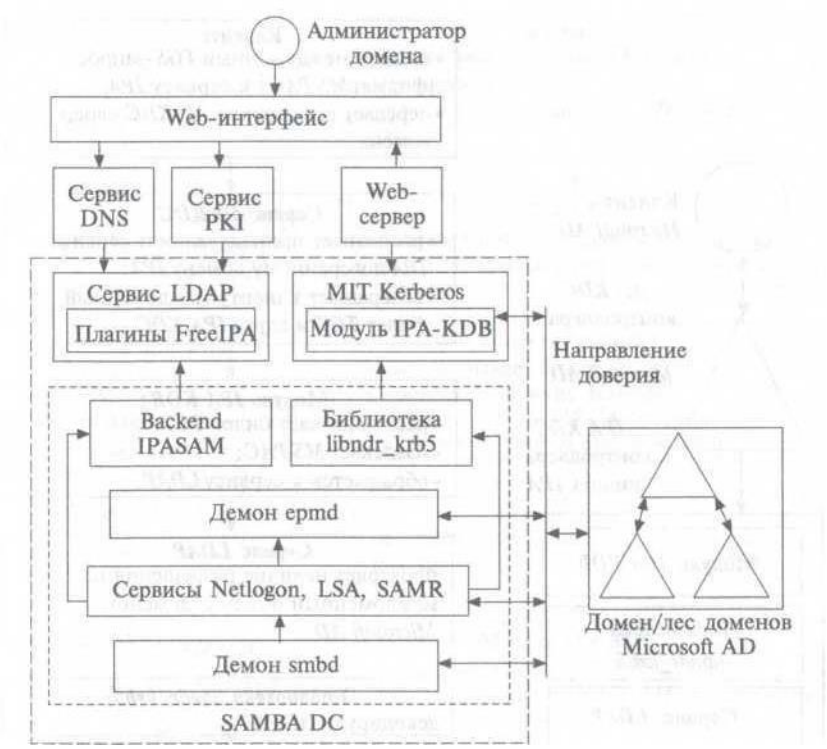


Рисунок 60.Структура ядра FreeIPA

библиотека *libndr_krb5* реализует алгоритмы упаковки и распаковка структур авторизации, в частности структуры *MS-PAC,* в билетах *Kerberos Tickets.* Администратор домена *IPA* выполняет управление ядром *FreeIPA с* использованием Web-интерфейса, разворачиваемого на базе штатного Web-сервера, например *apache*

На рис. 62 и 63 представлены этапы работы перечисленных компонентов ядра *FreeIPA* при организации доступа клиента домена *AD* к ресурсам сервиса, зарегистрированным в домене *IPA,* и клиента домена *FreeIPA* к ресурсам, зарегистрированным в домене *AD* (предполагается, что кроссдоменные доверительные отношения уже установлены).

Поскольку в рамках доменных инфраструктур ОССН *ALD* и *FreeIPA* сквозная аутентификация реализуется сервисом *Kerberos KDC,* то политика безопасности организуемая LSM-модулем *parsec- cifs* в рамках доменной инфраструктуры *ALD* распространяется и на доменную инфраструктуру *FreeIPA,* что позволяет организовать

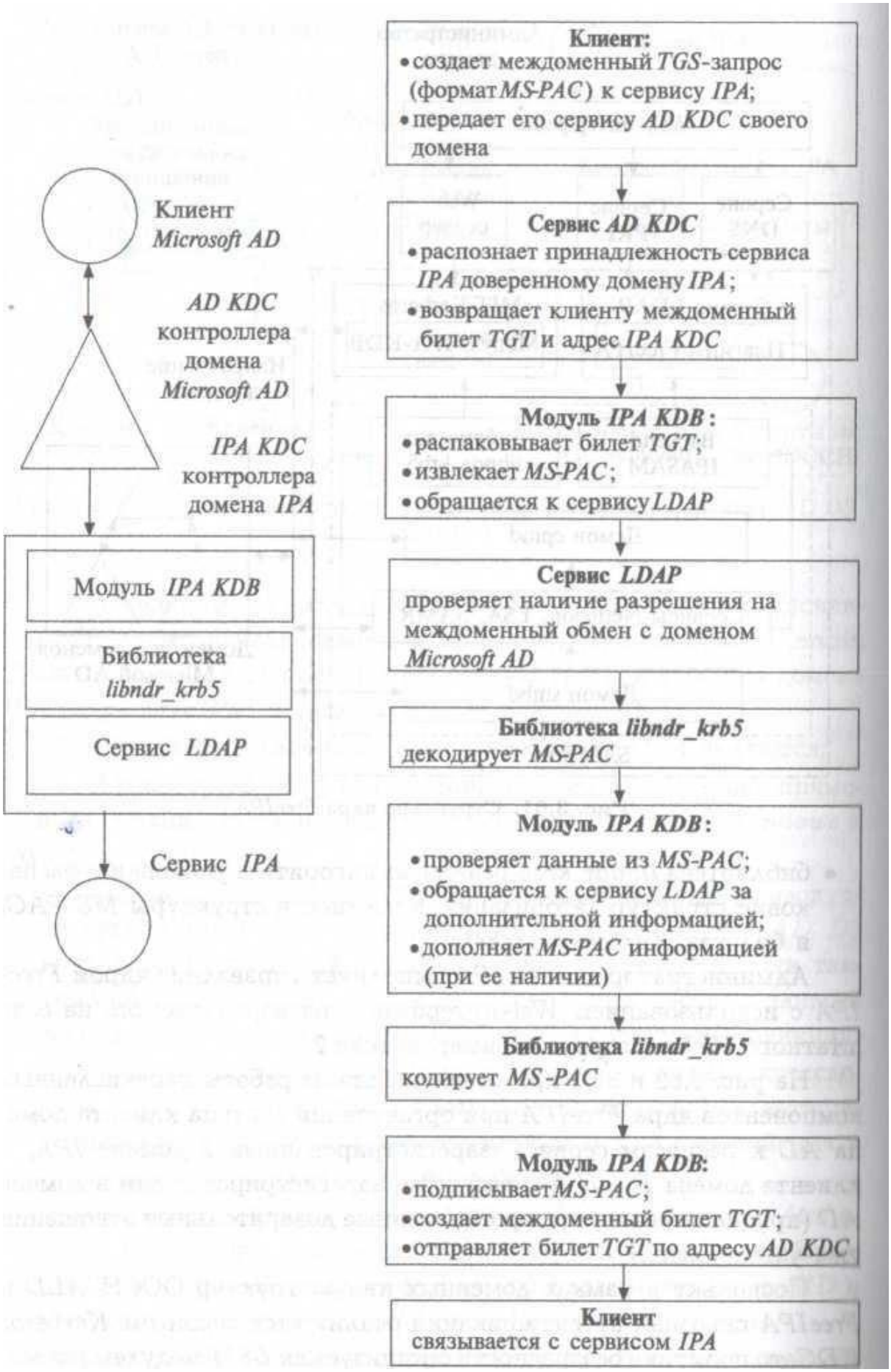


Рисунок 61.Этапы процесса кроссдоменного доступа клиента домена AD к ресурсам сервиса домена IPA

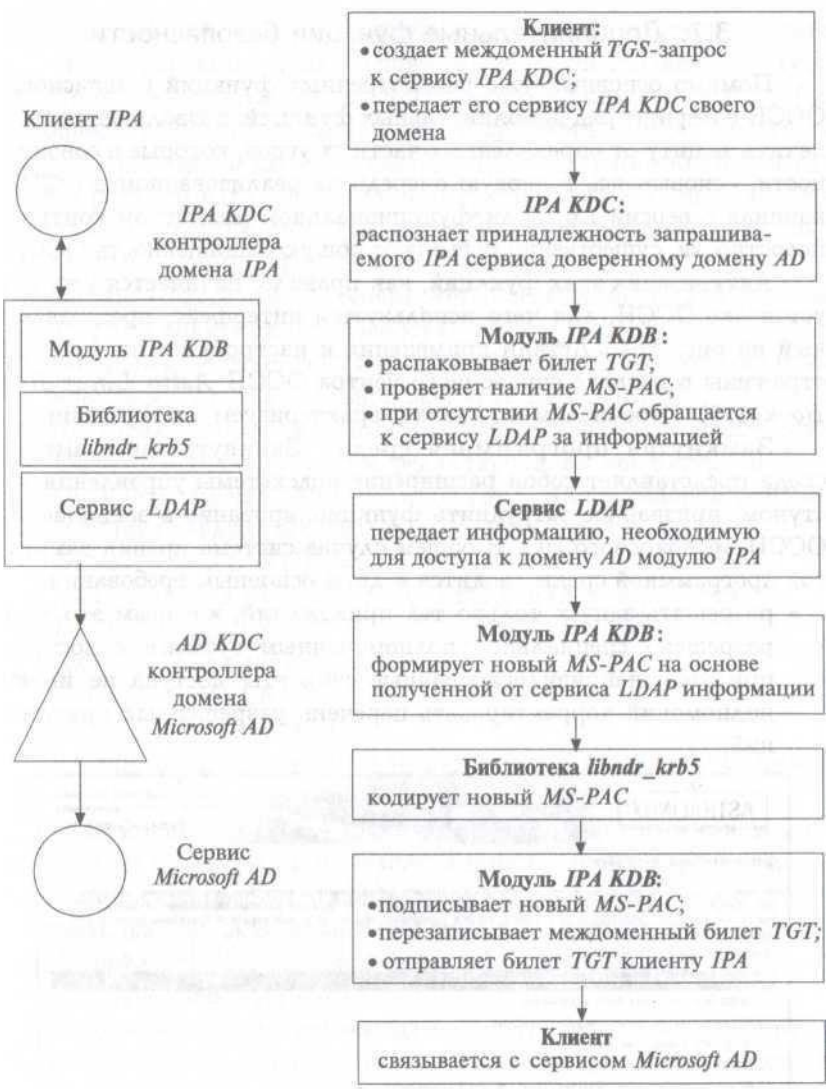


Рисунок 62.Этапы процесса кроссдоменного доступа клиента домена IPA к ресурсам сервиса домена AD

масштабируемую гетерогенную доменную инфраструктуру *AD FreeIPA,* поддерживающую единую политику безопасности, как на этапе миграции — поэтапного перехода от доменной инфраструктуры *AD* к инфраструктуре *FreeIPA,* так и на этапе эксплуатации инфокоммуникационных систем, требующих поддержки гетерогенности платформ ОС семейства *Microsoft Windows* и ОССН.

3.7. Дополнительные функции безопасности

Помимо основных уже рассмотренных функций безопаснасности ОССН содержит ряд дополнительных функций, позволяющих обеспечить защиту от определённых частных угроз, которые в совокупности, основываясь в первую очередь на реализованном в ОССН, начиная с версии 1.6, полнофункциональном мандатном контроле целостности, существенно повышают общую защищенность ОССН

Активизация этих функций, как правило, начинается уже при установке ОССН, для чего используется интерфейс, представленный на рис. 64. Детали применения и настройки этих функций отражены в разделе справочного центра ОССН *Astra Linux Red• Book* [102]. Перечислим и кратко охарактеризуем эти функции.

Замкнутая программная среда. Замкнутая программная среда представляет собой расширение подсистемы управления доступом, призванное затруднить функционирование в защищаемой ОССН вредоносного ПО. В общем случае система правил замкнутой программной среды сводится к двум основным требованиям:

- разрешать запуск только тех приложений, которым это явно разрешеноспециально уполномоченным субъектом доступа, при этом непривилегированные субъекты доступа не имеют полномочий корректировать перечень разрешённых приложений;

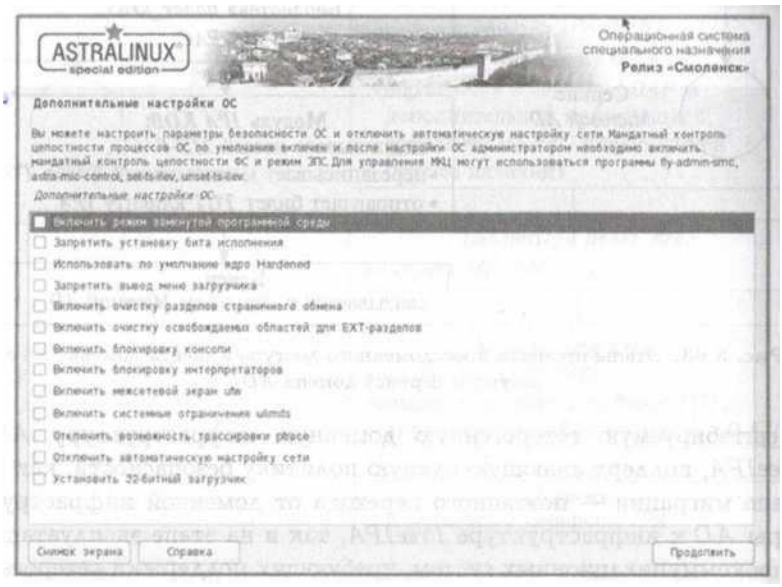


Рисунок 63. Интерфейс активизации дополнительных функций защиты при установке ОССН

- ограничивать доступ субъектов к сущностям, учитывая, в том числе посредством какого приложения осуществляется доступ. Первое из перечисленных требований реализовано в ОССН спомощью цифровой подписи, которой снабжены все исполняемые файлы, входящие в дистрибутив ОССН.

Поддерживаются три варианта реакции ядра ОССН на попытку загрузки для выполнения исполняемого файла, не имеющего корректной цифровой подписи:

- загрузка запрещается (штатный режим);
- загрузка разрешается, генерируется сообщение аудита (режим отладки политики безопасности);
- загрузка разрешается (режим разработчика ПО, не должен применяться в случае реальной эксплуатации ОССН).

Настройка механизма автоматического контроля целостности файлов осуществляется путём редактирования конфигурационного файла /*etc/digsig/digsig_initramfs.conf.* Для установки режима разработчика следует вставить в этот файл строку: *DIGSIG_LOAD_KEYS*= 0

Для установки режима отладки политики безопасности: *DIGSIG_LOAD_KEYS*= 1

Для установки штатного режима:

DIGSIG_LOAD_KEYS= 1 *DIGSIG_ENFORCE*= 1

Ключи цифровой подписи загружаются в каталог /*etc/digsig/ keys.*

Помимо исполняемых файлов, контроль целостности может также устанавливаться и на файлы данных. В этом случае целостность данных проверяется при каждом открытии поставленного на контроль файла. Для включения этой возможности следует вставить в файл /*etc/digsig/digsig_initramfs.conf* строку:

DIGSIG_USE_XATTR=1

Шаблоны имён поставленных на контроль файлов перечисляются в файле /*etc/digsig/xattr_control.* Ключи, используемые для проверки цифровых подписей поставленных на контроль файлов, хранятся в каталоге */etc/digsig/xattr_keys.* Текущее состояние системы контроля целостности хранится в файле /*etc/digsig/enforce.*

Кроме того, проверку цифровых подписей можно выполнять с помощью утилит командной строки *gostsum* и *gostsum_from.deb.*

Реализован механизм, обеспечивающий автоматическую проверку цифровых подписей всех загружаемых для выполнения исполняемых файлов. Фактически данный механизм реализует частный случай первого правила изолированной программной среды, при

Рисунок 64. Управление параметрами замкнутой прогаммной среды

этом множество разрешённых для выполнения приложений определено как множество приложений, файлы которых имеют корректную цифровую подпись их доверенного разработчика.

Управление всеми вышеперечисленными параметрами удобнее всего осуществлять с помощью раздела «Замкнутая программная среда» графической утилиты *fly-admin-smc,* представленного на рис.65.

Для проверки цифровой подписи конкретного файла в ручном режиме можно воспользоваться вкладкой «Подпись» окна «Свойства файла» утилиты «Менеджер файлов», как показано на рис. 3.60,

Запрет установки бита исполнения и блокировка интер- претаторов. Обычно в ОС семейства *Linux* процесс-владелец файла имеет полный, ничем не ограниченный доступ к атрибутам защиты этого файла. Любой пользователь может присвоить любому своему файлу любые атрибуты защиты. В частности, процесс-владелец файла может произвольно манипулировать правом доступа на вы- полнение *(X) к* файлу, что интерпретируется в ОС семейства *Linux* как признак исполняемости файла. Это позволяет ему устанавливать в домашнем каталоге учётной записи пользователя, от имени которой он функционирует, произвольные приложения, создавать и редактировать скрипты командного интерпретатора *bash* и других интерпретируемых языков программирования. Данная возможность значительно упрощает внедрение и функционирование в ОС вредоносных программ.

Фактически единственный способ ограничить возможности вносить сторонние приложения в ОС семейства *Linux* — разместить домашний каталог учётной записи пользователя на отдельном разделе, смонтированном с параметром noexec. Практическое применение

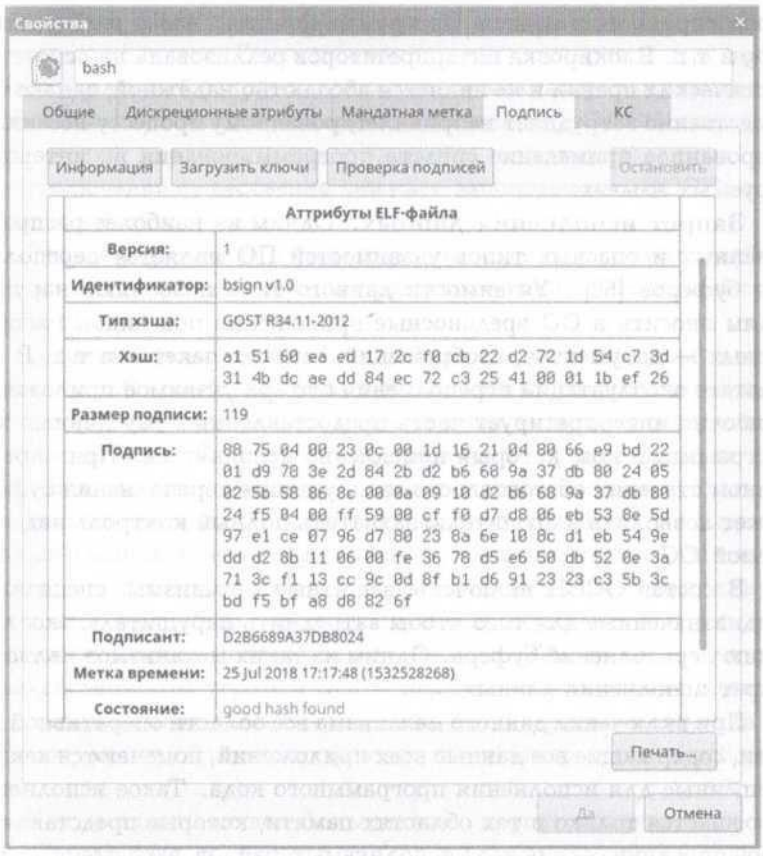


Рисунок 65.Проверка цифровой подписи фалйа

данного способа сопряжено со множеством неудобств, администраторы ОС практически никогда так не делают.

В ОССН, начиная с версии 1.6, поддерживается возможность запретить создание исполняемых файлов процессам непривилегированных учётных записей пользователей, соответственно имеющих уровень целостности «Низкий» (0). В этом случае учётная запись пользователя, процессы которой способны устанавливать младшие биты в байтах векторов доступа к файлам, должна обладать уровнем целостности «Высокий» (63).

Запрет установки бита исполнения может быть дополнен блокировкой интерпретаторов. Если данная функция безопасности включена, то от имени учётных записей пользователей, не входящих в группу *astra-console,* запрещено стартовать процессы интерпрета-

торов скриптовых языков программирования: *bash, perl, python, ruby* и т. п. Блокировка интерпретаторов реализована на основе эвристических правил и не является абсолютно надёжной, однако онасущественно затрудняет непривилегированному процессу несанкционированное применение средств программирования на интерпретируемых языках.

Запрет исполнения данных. Одним из наиболее распространённых и опасных типов уязвимостей ПО является переполнения буферов [58]. Уязвимости данного, типа позволяют нарушителям вносить в ОС вредоносные приложения под видом наборов данных — документов, изображений, сетевых пакетов и т. д. В результате эксплуатации переполнения буфера уязвимое приложение ошибочно интерпретирует часть предоставленных ему данных как программный код, который немедленно исполняется. При определённом стечении обстоятельств эксплуатация переполнения буфера может позволить нарушителю захватить полный контроль над атакуемой ОС. В состав ОССН включены защитные механизмы, специально предназначенные для того чтобы затруднить нарушителю эксплуатацию переполнений буфера. Одним из таких механизмов является запрет исполнения данных.

При включении данного механизма все области оперативной памяти, содержащие все данные всех приложений, помечаются как запрещённые для исполнения программного кода. Такое исполнение разрешается только в тех областях памяти, которые представляют собой проекции секций кода исполняемых файлов, загруженных для выполнения штатным порядком функционирования ОССН. В результате эксплуатация переполнений буферов становится возможна только с применением специальных технологий обхода данной защиты (*ROP, JOP* и т. д.), что серьёзно затрудняет создание эксплойтов для ОССН.

Не все компиляторы совместимы с запретом исполнения данных. Некоторые компиляторы в результате оптимизации програм- много кода или по другим причинам создают приложения, пред- полагающие исполнение программного кода в куче или даже стеке. Иногда, например, компиляторы для ускорения работы приложения преобразуют таблицу указателей на программный код в таблицу команд безусловного перехода на соответствующие адреса, что делает полученное приложение несовместимым с запретом исполнения данных.

Все приложения, входящие в состав дистрибутива ОССН или

включённые в официальный репозиторий, совместимы с запретом исполнения данных. Единственная причина, которая может потребовать отключить запрет исполнения данных, актуальна когда есть необходимость обеспечить корректную работу в ОССН сторонних приложений, несовместимых с запретом исполнения данных. Такое отключение существенно снижает защищенность ОССН, и его следует избегать.

Ручное управление параметрами политики запрета исполнения данных можно осуществлять посредством утилиты командной строки *paxctl.*

Очистка освобождаемых областей на жёстком диске. Обычно в ОС семейства *Linux* освобождаемые области на жёстком диске освобождаются только логически, без затирания содержащей- ся в них данных. Если в них содержались конфиденциальные дан- ные, то после перезагрузки ОС они потенциально могут стать доступны нарушителю. На практике задача выявления конфиденциальных данных в свободных областях жёсткого диска решается не так просто, как может показаться, — искомые данные присутствуют там в виде множества коротких отрезков, собрать из которых полный набор данных крайне затруднительно. Тем не менее для противодействия этой угрозе в ОССН поддерживается функция очистки освобождаемых областей жёсткого диска, при её включении все освобождаемые области затираются немедленно после освобождения.

Как и подавляющее большинство других современных ОС, ОССН реализует концепцию виртуальной памяти, когда редко используемые страницы оперативной памяти временно вытесняются на жёсткий диск. Виртуальная память позволяет в несколько раз увеличить объем доступной приложениям памяти ценой незначи- тельного снижения производительности ОС. Подобно другим освобождаемым областям жёсткого диска, данные страничного обмена (т. е. фрагменты оперативной памяти, временно вытесненные на жёсткий диск) могут содержать конфиденциальные данные, которые потенциально могут стать доступными нарушителю. В ОССН поддерживается функция очистки разделов страничного обмена, она может включаться независимо от функции освобождения других областей жёсткого диска. Включение упомянутых функций незначительно замедляет работу ОССН.

Контроль целостности (неизменности) файлов. Администратор ОССН может в любой момент проверить совпадение ис-

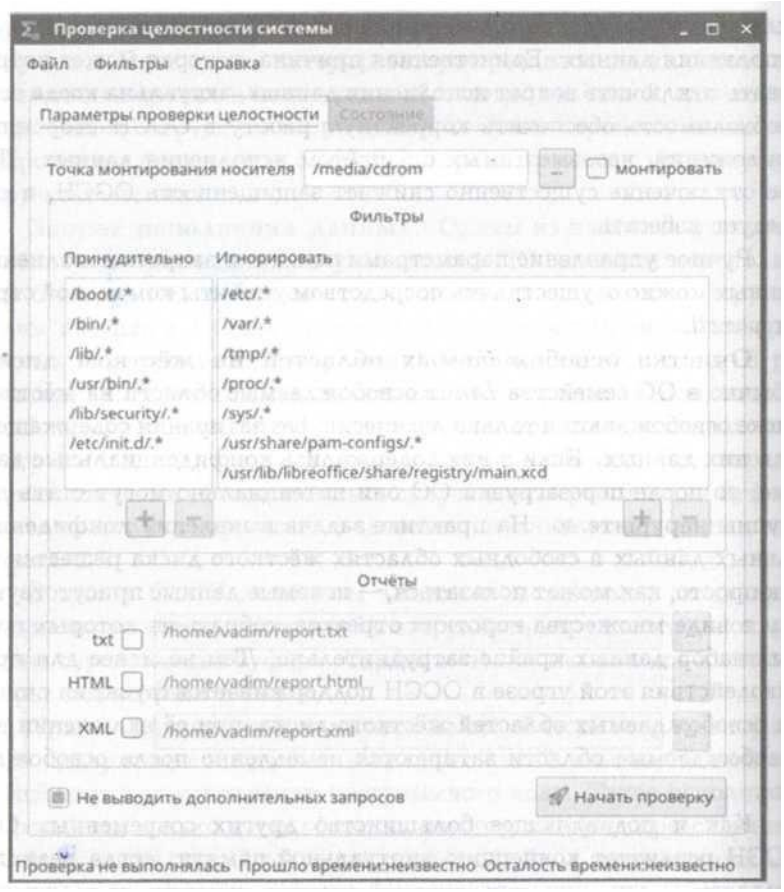


Рисунок 66.Интерфейс утилиты проверки целостности (неизменности) файлов

полняемых файлов ОССН с соответствующими файлами дистрибутива. Для проверки целостности (неизменности) используется графическая утилита *fly-admm-int-check,* интерфейс которой представлен на рис. 3.67.

Элементы интерфейса имеют следующее назначение:

· Точка монтирования носителя — полный путь к носителю с дистрибутивом ОССН.

· Монтировать — указывает, должен ли носитель быть смонтирован автоматически.

· Фильтры / Принудительно — перечень файлов и каталогов, целостность которых проверяется в любом случае, даже если они также содержатся в перечне «Игнорировать».

· Фильтры / Игнорировать — перечень файлов и каталогов, целостность которых не проверяется, кроме случаев, когда они также присутствуют в перечне «Принудительно».

· Отчёты — пути к файлам, в которые будут помещены отчёты о проверке целостности в форматах соответственно, текстовом, *HTML* или *XML.*

· Не выводить дополнительных запросов — блокирует выдачу утилитой дополнительных сообщений наподобие «Вставьте диск».

· Начать проверку — начать проверку немедленно.

Целостность файлов, не включённых ни в список «Принудительно», ни в список «Игнорировать», по умолчанию проверяется.

Поддерживается возможность выполнять периодический регламентный контроль целостности, что обеспечивается специальным набором приложений, запускаемых посредством системного планировщика *cron.* Для управления данным механизмом служит утилита командной строки *afick,* а также конфигурационный файл /etc/ *afick.conf.*

Рекомендуется, как минимум, ежедневно проверять целостность следующих файлов:

· ядро /*boot/vmlinuz-*;*

· модули ядра */lib/modules/*;*

· модули подсистемы аутентификации */lib/security/pam;*

· образы временной файловой системы /*boot/initrd.img-*;*

· конфигурация загрузчика /*boot/grub/menu.lst;*

· конфигурация ядра /*etc/sysctl.conf ;*

· конфигурация процесса *init /etc/inittab;*

· конфигурация графической подсистемы /*etc/ X11 / default- dis- playmanager;*

· конфигурация подсистемы аутентификации */etc/pam.*;*

· конфигурация *NFS /etc/exports;*

· порядок монтирования файловых систем */etc/fstab;*

· скрипты автозапуска демонов /*etc/init.d/** и ссылки на них */etc/rc*;*

· список учётных записей пользователей /*etc/passwd;*

· список групп учётных записей пользователей */etc/group;*

· список безопасных терминалов /*etc/securetty;*

· список зарегистрированных командных интерпретаторов / *etc/ shells;*

· исполняемые файлы из каталогов */bin/*, /sbin/*, /usr/bin/**, */usr/sbin/*.*

Управление доступом к подключаемым устройствам. Практически у всех организаций, предъявляющих повышенные требования к безопасности информации, в состав этих требований входит обеспечение управления доступом к подключаемым устройствам, в первую очередь, USB-дискам. Известно множество случаев, когда бесконтрольное использование подключаемых устройств приводило к внедрению в защищаемые системы вредоносного ПО.

В ОССН поддерживается возможность запрещать подключение незарегистрированных устройств, а зарегистрированным устройствам назначать мандатные метки конфиденциальности и целостности, а также ограничивать дискреционные права доступа к хранящимся на них файлам и каталогам. Управление политикой ограничения доступа к подключаемым устройствам реализуется путём редактирования конфигурационного файла /*etc/parsec/ devices.cfg* либо средствами *ALD.* В качестве клиентской программы удобнее всего использовать раздел «Устройства и правила» графической утилиты *flyadmin-smc.*

Контрольные вопросы

- Как мандатное управление доступом позволяет повысить защищенность ОССН?
- В каких файлах ОССН хранятся поддерживаемые в её конкретном эземпляре мандатные уровни конфиденциальности и неиерархические категории?
- Каким образом в ОССН по умолчанию назначаются мандатные атрибуты файлам и каталогам?
- Какая утилита используется в ОССН для управления политикой без- опасности?
- Как пользователю проще всего узнать мандатные атрибуты текущего сеанса работы с ОССН?
- Как пользователю проще всего узнать мандатные атрибуты файла или каталога?
- Как можно запустить приложение с мандатными атрибутами, отличными от мандатных атрибутов текущего сеанса работы пользователя с ОССН?
- Для чего предназначен мандатный контроль целостности?
- Как проще всего узнать уровень целостности текущего сеанса работы пользователя?
- Какие дополнительные меры применяются в ОССН для усиления защиты графической подсистемы?
- Какие настройки подсистемы аутентификации поддерживает ОССН?
- Как можно просмотреть журналы аудита ОССН?
- Как осуществляется управление политикой аудита ОССН?
- В чём разница в реализации функций управления доступом *LSM*-модулей *parsec* и *parsec-sifs*?
- Для какой версии протокола *IP* подсистема безопасности ОССН *PAR- SEC* обеспечивает раздельную реализацию управления доступом?

- Поддерживает ли ядро ОССН протоколы динамической маршрутизации?
- В какой подкаталог файловой системы */proc* отображается таблица *Kernel IP routing table* ядра ОССН?
- Является ли доменная сетевая инфраструктура ОССН совместимой с инфраструктурой *Microsoft Active Directory Service*?
- Какой протокол используется в сетевой инфраструктуре ОССН для реализации механизма единого входа (550)?
- Из каких логических компонентов состоит ВПП ОССН?
- Каковы особенности модификации дерева директорий информации (DIT-дерева) базы данных учётных записей пользователей глобального пространства имён применительно к реализации МРОСЛ ДП-модели?
- С использованием какой технологии происходит динамическое пере- ключение имён в рамках ЕПП ОССН?
- Какой механизм является основой модульной реализации метода единого входа (550) доменной сетевой инфраструктуры ОССН?
- Какие сервисы (демоны) являются базовыми компонентами СЗФС ОССН?
- Какие пакеты механизма *ALD* в ОССН являются базовыми, а какие реализуют расширенные функции?
- Какие типы доверительных отношений доменной сетевой инфраструктуры *Active*
- *Directory Domain Service* являются внутридоменными, междоменными и кроссдоменными?
- Какая идентификационная информация о пользователе/группе пользователей домена *Active Directory* используется для реализации управления доступом к ресурсам домена и организации междоменных доверительных отношений?
- На основе какого технологического решения инфраструктура *FreeIPA* позволяет организовать кроссдоменные доверительные отношения с инфра- структурой *Active Directory Domain Service*?
- Какой протокольный блок данных инфраструктуры *Active Directory Domain Service* упаковывается в билет *Kerberos Ticket* в процессе авторизации пользователя при кроссдоменных доверительных отношениях?
- Какая версия пакета программ *Samba* используется в инфраструктуре *FreeIPA*?

- Какая служба в составе инфраструктуры *FreeIPA* реализует перевод имён пользователей в их идентификаторы безопасности (*SID*) и обратно?
- Какую функцию в составе инфраструктуры *FreeIPA* реализует библиотека libndr_krb5?
- Какой протокол использует инфраструктура *FreeIPA* для организации удалённого управления кроссдоменными (специальными) учётными записями пользователей и групп?
- Какие дополнительные функции безопасности реализованы в ОССН на основе *Astra Linux Red-Book* и как они администрируются?

4. 1. Лабораторная работа № 1

Работа с учётными записями пользователей и группами

Цель работы

Изучить особенности администрирования локальных учётных записей пользователей и групп в ОССН с использованием командной строки и графического интерфейса.

Время выполнения работы 4 академических часа.

Краткие теоретические сведения

ОССН — многопользовательская ОС, потому учётная запись пользователя — ключевой элемент всей системы управления доступом. Для идентификации учётных записей пользователей и групп в ОССН как во всех ОС семейства *Linux* используются *uid* и *gid* соответственно (подробно порядок работы в ОССН с этими идентификаторами рассмотрен в главе 1). Каждая учётная запись пользователя должна принадлежать как минимум одной группе — первичной группе. При создании учётной записи пользователя командой *adduser* или с использованием графической утилиты *Fly-admin-smc* создаётся группа, имя которой совпадает с системным именем учётной записи пользователя. Данная группа применяется как первичная группа и будет задана идентификатором в учётной записи пользователя, расположенной в файле */etc/passwd.* Учётная запись пользователя может входить более чем в одну группу, тогда имена таких групп (в ОССН данные группы называются вторичными) будут находиться в файле */etc/group.*

Как правило, файлы, владельцами которых являются учётные записи пользователей, хранятся в соответствующих им домашних каталогах, находящихся в каталоге /*home.* При этом во время первого входа в ОССН с заданными уровнем доступа *(Num1),* уровнем целостности *(Num2)* и набором неиерархических категорий (*Num3)* (например, 0x2 – вторая категория) создается уникальный каталог с именем вида */home/.pdp/имя_пользователя/lNum1iNum2cNum3t0xO*,что позволяет распределить по каталогам файлы (в том числе документы) в зависимости от их уровней конфиденциальности и целостности. Доступ субъект-сессий (процессов), функционирующих от имени других учётных записей пользователей, к домашнему каталогу в ОССН версии 1.6 может быть ограничен с использованием как параметров (меток конфиденциальности) мандатного управления доступом, так и дискреционных прав доступа.

При работе от имени учётной записи администратора в ОССН версии 1.6 желательно использовать только высокий уровень целостности (по умолчанию он равен 63) и минимальный уровень конфиденциальности в связи с особенностями монтирования его домашнего каталога.

Данные об учётных записях пользователей и группах хранятся в файлах */etc/passwd* и */etc/group* соответственно, структура которых описана в главе 1. Если в файле */etc/passwd*

для некоторой учётной записи пользователя вместо пароля указан символ «*x*», то свёртка пароля находится в файле */etc/shadow.* Его структура представляет собой список, каждая строка которого состоит из элементов, разделённых символом «:», среди которых: число дней с 1 января 1970 г. до дня последнего изменения пароля, минимальное число дней действия пароля со дня его последнего изменения, максимальное число дней действия пароля, за сколько дней до устаревания пароля ОССН начнёт выдавать предупреждения, число дней со времени обязательной смены пароля до блокировки учётной записи пользователя, день блокировки учётной записи пользователя.

Для администрирования параметров учётных записей пользователей используются следующие команды и утилиты (примеры применения которых рассмотрены в главах 1 и 3):

- *useradd* и *adduser* — команды добавления учётной записи пользователя;
- *passwd* — команда смены пароля учётной записи пользователя;
- *usermod* — команда модификации параметров уже существующей учётной записи;
- *userdel* — команда удаления учётной записи пользователя;
- *gpasswd* — команда управления группами;
- *addgroup* — команда создания группы;
- *delgroup* — команда удаление группы
- *Fly-admin-smc*—графическая утилита, позволяющая решать весь комплекс задач по администрированию учётных записей пользователей и групп, в том числе администрировать параметры мандатного управления доступом и мандатного контроля целостности.

В рамках МРОСЛ ДП-модели учётным записям пользователей соответствует множество *U,* группам — роли из множеств *R* и *AR,* соответствующие перечисленным командам де-юре правила преобразования состояний рассмотрены в главе 2.

При администрировании ОССН необходимо руководствоваться следующими рекомендациями. Если в ней включён мандатный контроль целостности на файловой системе, то для администрирования ОССН требуется его временно отключить, для чего:

- снять мандатный контроль целостности с файловой системы ОССН с помощью графической утилиты *Fly-admin-smc* или командой *unset-fs-ilev,*
- выполнить необходимые действия по администрированию ОССН (настроить ОССН, установить пакеты и т. д.);
- включить мандатный контроль целостности на файловой системе ОССН с помощью графической утилиты *Fly-admin-smc* или командой *set-fs-ilev;*
- настроить метки целостности установленных системных объектов.

Для полного выключения режима мандатного контроля целостности:

- при использовании графического интерфейса с помощью графической утилиты *Fly-*

admin-smc выбрать «Мандатный контроль целостности» и снять отметку «подсистема МКЦ»;

- при использовании терминала *Fly* выполнить команду *astra-mic-control disable.*

Независимо от способа выключения, чтобы изменения вступили в силу необходимо перезагрузить ОССН. Полностью выключать мандатный контроль целостности крайне не рекомендуется, так как многие механизмы защиты связаны с его включённым режимом, а именно: блокировка интерпретаторов, режим блокировки установки бита исполнения — *nochmodx,* блокировка доступа к конфиденциальной информации и т. д.

Используемое методическое и лабораторное обеспечение

- ОССН версии 1.6, в которой создана учётная запись пользователя *user,* с параметрами: максимальный и минимальный уровни доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий», входит в группу администраторов — *astraadmin* (вторичная группа), разрешено выполнение привилегированных команд *(sudo).*
- Документация: «Операционная система специального назначения *«Astra Linux Special Edition».* Руководство администратора. Часть 1», «Операционная система специального назначения *«Astra Linux Special Edition».* Руководство по КСЗ. Часть 1», «Операционная система специального назначения «*Astra Linux Special Edition».* Руководство пользователя».
- Для выполнения работы в течение двух занятий необходимо обеспечить возможность сохранения состояния ОССН за счёт применения технологий виртуализации (создания виртуальных машин с ОССН).

Порядок выполнения работы

1. Начать работу со входа в ОССН в графическом режиме с учётной записью пользователя *user*(уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий»).
2. Запустить терминал *Fly* из меню «Системные».
3. Определить текущую учётную запись пользователя с использованием команды *whoami.*
4. Проверить наличие права доступа на чтение к файлу */etc/passwd* и получить следующие данные, выполнив команды *cat/etc/passwd*или *less*/*etc/passwd:*

- число параметров учётных записей пользователей (для этого дополнительно можно использовать команды *wc* и *sort);*
- текущее число учётных записей пользователей;
- число различных используемых командных интерпретаторов.

5. Вывести строку, соответствующую текущей учётной записи пользователя, из файла */etc/passwd* с использованием команды *cat/etc/passwd*|*grep "^$(whoami):"*, при этом получить

следующие данные:

наличие пароля или свёртки пароля (вывести эти данные командой *cat/etc/passwd|grep"^$(whoami):" |cut -d:-f2):*

root@astra:/home/user# cat/etc/passwd|grep"^$(whoami):" root:x:0:0:root:/root:/bin/bash

root@astra:/home/user# exit

exit

user@astra:~$ cat/etc/passwd|grep "^$(whoami):"

user:x:1000:1000:*,,, :*/home/user:/bin/bash

- группа и идентификатор текущей учётной записи пользователя;
- командный интерпретатор по умолчанию для текущей учётной записи пользователя.

6. найти отличия исполняемых файлов *adduser* и *useradd,* для чего:

- определить расположение файлов *adduser* и *useradd* с использованием команд *sudo which, adduser* и *sudo which useradd*;
- вывести в терминал *Fly* тип обоих файлов командой *file,* определить их принципиальное отличие.

7. Добавить две учётные записи пользователей *user1 u user2*(с соответствующими домашними каталогами) с использованием команд *sudo adduser user1* и *sudo adduser user2.*

8. Проанализировать изменения в ОССН, связанные с добавлением новых учётных записей пользователей, для чего определить:

- домашние каталоги учётных записей пользователей по данным файла */etc/passwd*;
- номер алгоритма свёртки паролей учётных записей пользователей по файлу */etc/shadow с* использованием привилегированного режима или команды *sudo:*

root@astra:/home/user#cat/etc/shadow|grep "^user:"|cut -d$ -f26

- скрипты, которые были перемещены в домашние каталоги учётных записей пользователей из каталога */etc/skel,* при этом сравнить файлы в каталоге */etc/skel* с файлами домашних каталогов учётных записей пользователей с использованием команды *sudo diff –s /etc/skel/home/имя_пользователя|grep*"идентичны";
- новые группы в файле */etc/group,*
- идентификаторы новых учётных записей пользователей и групп в файлах */etc/group и/etc/passwd.*

9. Осуществить попытку создания учётных записей пользователей *user3*, *user4* командами *useradd user3 u useradd user4* без использования команды *sudo,* проанализировать результат. Выполнить те же действия с применением команды *sudo,* после чего определить:

- домашние каталоги учётных записей пользователей по файлу */etc/passwd*;
- были ли созданы домашние каталоги учётных записей пользователей;

- наличие свёрток паролей учётный записей пользователей по файлам */etc/passwd* и */etc/shadow,*
- новые группы в файле */etc/group*
- идентификаторы новых учётных записей пользователей в файле */etc/passwd;*
- командный интерпретатор по умолчанию для созданных учётных записей пользователей:

root0astra:/home/user# tail -1 /etc/passwd|cut -d: -f7 /bin/bash

10. Реализовать попытки задать пароль для учётных записей пользователей *user3* и *user4 с использованием* команд *passwd user3* и *passwd user4* без использования и с использованием команды *sudo,* сравнить результаты. Определить алгоритм свёртки пароля этих учётных записей пользователей по файлу */etc/shadow.*
11. Выполнить дополнительную настройку OCCH для обеспечения возможности входа с учётной записью пользователя *user3,* для чего осуществить следующие действия:
- выполнить вход в OCCH с учётной записью пользователя *user3,* введя его имя и пароль, проанализировать предупреждения, выдаваемые OCCH;
- войти в OCCH с учётной записью пользователя user;
- создать домашний каталог учётной записи пользователя *user3* командами *sudo mkdir /home/user3,* и назначить ему необходимые права доступа: *sudo chown user3:user3 /home/user3, sudo chmod* 750 */home/user3;*
- проверить возможность входа в OCCH с учётной записью пользователя user3;
- войти в OCCH с учётной записью пользователя *user.*
12. Запустить терминал *Fly* в «привилегированном» режиме командой *sudo Fly-term. .*
13. Модифицировать параметры учётных записей пользователей:
- установить домашний каталог учётной записи пользователя user1командой *usermod -d /home/userone user1*;
- установить домашний каталог учётной записи пользователя user2 командой *usermod-m-d /home/usertwo user2*
- проверить содержимое каталога */home* командой l*s –la* и определить отличия в результатах выполнения команды *usermod* на предыдущих этапах.
14. Осуществить последовательные попытки входа в OCCH с учётными записями созданных пользователей user1 и user2, при этом выполнить следующие действия:
- проанализировать причины появления предупреждений при входе в OCCH с учётной записью пользователя user1, сравнив отличия в командах, использованных при модификации параметров учётных записей пользователей *user1* и *user2;*

- вернуть домашний каталог учётной записи пользователя *user1* командой *usermod -d*

/home/user1 user1, рассмотреть результат её выполнения, проверить запись о домашнем каталоге в файле */etc/passwd;*

- повторно установить домашний каталог пользователя *user1* командой *usermod -m -d /home/userone user1,* проверить результат;
- проверить возможность входа в ОССН с учётной записью пользователя *user1,* выйти из ОССН.

15. Войти в ОССН с учётной записью пользователя *user* (Уровень_0, «Высокий»). Запустить графическую утилиту редактирования учётных записей пользователей «Политика безопасности» через меню «Панель управления» главного пользовательского меню.

16. Открыть раздел настройки локальных пользователей, и для созданных учётных записей пользователей *user1, user2, user3, user4* произвольно задать их параметры:

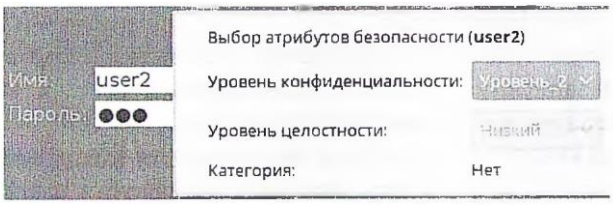
- максимальный и минимальный уровни доступа;
- минимальные и максимальные наборы неиерархических категорий;
- максимальный уровень целостности.

17. Настроить параметры учётной записи пользователя *user2:*

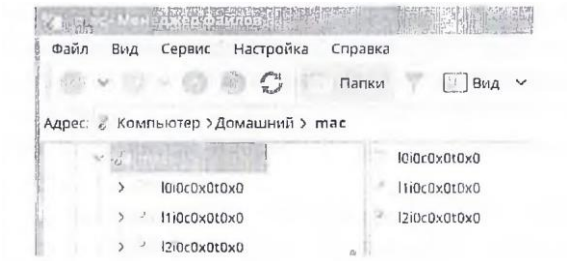
- установить минимальное число дней между сменой пароля —180 дней и до выдачи предупреждения о смене пароля — 5 дней;
- выбрать максимальный уровень — «Уровень_3»;
- проверить возможность задать минимальный или максимальный набор неиерархических категорий.

18. Войти в ОССН с учётной записью пользователя *user2,* выбрав уровень доступа «Уровень_1». Проверить возможность выбора набора неиерархических категорий и уровня целостности. Создать в каталоге «Документы» файл 1*.txt.* Выйти из ОССН.

19. Войти в ОССН с учётной записью пользователя *user2*выбрав уровень доступа «Уровень_2». Создать в каталоге «Документы» файл *2.txt:*



20. Проверить возможность чтения объектов файловой системы ОССН, владельцем которых является учётная запись пользователя *user2*(на текущем уровне доступа «Уровень_2»):



- открыть каталог «Документы» уровне доступа «Уровень_1»(*Компьютер*/*Домашний/mac/l1i0c00x0t0x0/Документы);*
- открыть файл 1*.txt,* проверив возможность его чтение или записи;
- выйти из ОССН.

21. Проверить наличие и возможность чтение объектов файловой системы ОССН, владельцем которых является учётная запись пользователяuser2 на текущем уровне доступа («Уровень _1»):

- войти в ОССН с учётной записью пользователя *user2,* выбрав уровень доступа «Уровень_1»;
- проверить возможность открытия каталога «Документы» для уровня доступа «Уровень_2»(*Компьютер*/*Домашший/mac/l2i0*c0x0t0x0/*Документы*);
- выйти из ОССН.

22. Войти в ОССН с учётной записью пользователе *user.* Запустить графическую утилиту «Политика безопасности». Сравнить списки вторичных групп для учётных записей пользователей *user, user1, user2, user3, user4,* при этом определив:

- учётные записи пользователей, являющиеся администраторами (входящими в группу *astra-admin);*
- учётные записи пользователей, входящие в группу *users.*

23. Создать новую учётную запись пользователя *user10:*

- установить минимальное число дней между сменой пароля — 180 дней и число дней выдачи предупреждения до смены пароля — 5 дней;
- выбрать максимальный уровень доступа — «Уровень_3», минимальный уровень доступа — «Уровень_0», уровень целостности — «Высокий»;
- добавить в список вторичных групп группы *astra-admin* и *lpadmin*;
- проверить возможность входа в ОССН с учётной записью пользователя *user10* с уровнями доступа «Уровень_2» или «Уровень_3» (уровень целостности «Низкий»).

24. Войти в ОССН с учётной записью пользователя *user*10 (уровень доступа — «Уровень_0», уровень целостности «Высокий»), Проверить возможность создания новой учётной записи пользователя *user*11 с использованием графической утилиты *Fly-admin-smc* без использования и с использованием команды *sudo.* Выйти из ОССН.

25. Войти в ОССН с учётной записью пользователя *user1 с* уровнем доступа — «Уровень_0». Осуществить попытки запуска графической утилиты «Политика безопасности» через главное пользовательское меню и запуска её с использованием терминала *Fly* командой *Fly-admin-smc.* Проанализировать результаты и предупреждения ОССН.

26. Осуществить переключение между сеансами различных учётных записей пользователей без выхода из ОССН:

- через меню «Завершение работы» главного пользовательского меню перейти в подменю «Сессия» и далее «Отдельная» и войти в ОССН с учётной записью пользователя *user*(уровень целостности «Высокий»);
- в аналогично вернуться и далее закрыть сеанс от имени учётной записи пользователя user1.

27. С использованием графической утилиты «Политика безопасности» заблокировать пароль учётной записи пользователя *user*1. Проверить изменения файлов */etc/passwd* и */etc/shadow,* осуществив следующие действия:

- в терминале *Fly* выполнить команды *sudo cat/etc/passwd* и *sudo cat/etc/shadow*;
- проверить наличие блокировки учётной записи пользователя по файлу */etc/shadow* (должен быть установлен знак «!» в начале свёртки пароля);
- проверить функционирование блокировки путём осуществления попытки входа в ОССН в отдельном сеансе от имени учётной записи пользователя user1;
- снять блокировку (выполнить удаление пароля и блокировки входа, задать повторно пароль) и проверить возможность входа в ОССН с учётной записью пользователя *user*1.

28. Выполнить удаление учётных записей пользователей:

- удалить учётную запись пользователя *user10* с использованием графической утилиты «Политика безопасности»;

- удалить учётную запись пользователя *user2*командой *sudo deluser user2;*
- удалить учётную запись пользователя *user1*командой *sudo userdel user1;*
- проверить наличие домашних каталогов учётных записей пользователей *user1* и *user2,* после чего с использованием справочной информации по команде *userdel* определить её параметры, позволяющие удалять содержимое домашнего каталога учётной записи пользователя;
- удалить домашние каталоги учётных записей пользователей *user1*и *user2*непосредственно командами *rm –r /home/userone* и *rm -r /home/usertwo*, осуществив попытки удаления без использования и с использованием команды *sudo;*
- проверить наличие домашних каталогов учётных записей пользователей *user1, user2* и

user10 в каталоге */home/.pdp.*

29. Создать новые группу gr*oup3* (с использованием графической утилиты «Политика безопасности») и группу *group4*(командой *sudo addgroup group4,* выполненной в терминале *Fly).*

30. Добавить учётную запись пользователя *user3*во вторичную группу gr*oup3* командой *usermod -a –G group3 user3* и во вторичную группу *group4 c* помощью графической утилиты «Политика безопасности». Проверить включение учётной записи пользователя *user3* в группы *group3* и *group4* путем просмотра содержимого файла */etc/group* командами *cat/etc/group*|*grep*"^*group3*" и *cat/etc/group*| *grep"^group4".*

31. Выполнить удаление учётной записи пользователя *user3* из группы *group3* с использованием графической утилиты «Политика безопасности» и из группы *group4* командой *gpasswd -duser3 group4.*

32. Удалить группу *group3* командой *sudo delgroup group3* в терминале *Fly* и группу gr*oup4* с помощью графической утилиты «Политика безопасности».

33. Изучить порядок хранения параметров мандатного управления доступом и мандатного контроля целостности для учётных записей пользователей. Для этого выполнить следующие действия:

- определить уровни доступа, заданные в ОССН, для этого вывести в терминал *Fly* содержимое файла */etc/parsec/mac_levels;*
- определить неиерархические категории, заданные в ОССН, для этого вывести в терминал *Fly* содержимое файла /*etc/parsec/mac_categories;*
- определить идентификатор учётной записи пользователя *user1* по файлу */etc/passwd*командой *cat/etc/passwd*|*grep*"^user1:"| *cut -d* : *-f3*;
- считать параметры мандатного управления доступом и мандатного контроля целостности для учётной записи пользователя *user1* командой *cat/etc/parsec/macdb/$(cat/etc/passwd\grep*"^user1:"|*cut –d : -f3*) и проверить их соответствие данным, отображаемым в графической утилите «Политика безопасности».

Содержание отчёта о выполненной работе

В отчёте о выполненной работе необходимо указать:

1. Полный перечень использованных команд с кратким описанием их назначения.
2. Примеры выполнения команд, которые были использованы в ходе работы с описанием результатов их выполнения.
3. Описание порядка работы с графической утилитой «Политика безопасности» *(Fly-admin-smc)* при осуществлении следующих действий:

- создание новых учётной записи пользователя или группы;
- удаление существующих учётной записи пользователя или группы;

- добавление учётной записи пользователя в существующую группу;
- добавление учётной записи пользователя в группу администраторов (*astra-admin);*
- изменение параметров мандатного управления доступом и мандатного контроля целостности.

4. Список и назначение системных файлов, связанных с хранением параметров учётных записей пользователей, групп, параметров мандатного управления доступом и мандатного контроля целостности.

Контрольные вопросы

1. Какие имеются особенности создания учётных записей пользователей с использованием команд *adduser, useradd u* графической утилиты «Политика безопасности» *(Fly-admin-smc),* в том числе:
   - какой группе должна принадлежать учётная запись пользователя, чтобы была возможность выполнения команды *adduser?*
   - какими командами создаётся учётная запись пользователя, и какие дополнительные параметры при этом вводятся?
   - какие ограничения накладываются на пароль учётной записи пользователя при его создании?
   - в какие группы автоматически добавляется учётная запись пользователя?
2. Как выполнять привилегированные команды?
3. Создаются ли домашние каталоги учётных записей пользователей при добавлении их с использованием графической утилиты «Политика безопасности»?
4. Создаются ли домашние каталоги учётных записей пользователей при их добавлении с использованием команд *adduser* и *useradd?*
5. Какие минимальный и максимальный уровни доступа задаются по умолчанию для учётных записей пользователей, создаваемых командами *adduser* и *useradd?*
6. Какими способами можно добавить или удалить учётную запись пользователя из группы?
7. Каким образом организовано хранение сущностей файловой системы ОССН, созданных процессами, обладающими различными уровнями доступа?
8. Где и в каком формате хранятся параметры мандатного управления доступом *u* мандатного контроля целостности, заданные в ОССН?
9. Где и в каком формате хранятся параметры мандатного управления доступом и мандатного контроля целостности для учётных записей пользователей?
10. Какой командой задаётся максимальный набор неиерархических категорий для текущей учётной записи пользователя?
11. Каким образом осуществляется переход от текущего сеанса к сеансу,

функционирующему от имени другой учётной записи пользователя?

12. Позволяют ли команды *useradd* и *adduser*задавать параметры мандатного управления доступом и мандатного контроля целостности для создаваемых учётных записей пользователей?

4.2. Лабораторная работа № 2.
Настройка параметров мандатного управления доступом и мандатного контроля целостности

Цель работы

Освоить администрирование основных параметров мандатного управления доступом и мандатного контроля целостности в ОССН с применением графических утилит и консольных команд.

Время выполнения работы 4 академических часа.

Краткие теоретические сведения

В ОССН наряду с традиционной для ОС семейства *Linux* системой дискреционного управления доступом реализована система мандатного управления доступом и мандатного контроля целостности на основе МРОСЛ ДП-модели, рассмотренной в главе 2. С этим связано наличие у сущностей ОССН (файлов, каталогов) мандатных меток конфиденциальности и целостности.

Параметрами мандатного управления доступом (мандатной меткой) являются следующие элементы:

• уровень доступа или конфиденциальности (соответствует уровню конфиденциальности сущности или доступа субъект-сессии);

• набор неиерархических категорий сущности и субъект-сессии;

• уровень целостности сущности и субъект-сессии;

• специальные атрибуты сущности *(CCNR, CCNRI, E.Hole, W-Hole)*

При установке ОССН (по умолчанию) задаются следующие параметры мандатного управления доступом и мандатного контроля целостности:

• 2 непосредственно используемых уровня целостности («Низкий» значение 0, «Высокий» — 63);

• 4 уровня доступа/конфиденциальности («Уровень_0» значение 0, «Уровень_1» — 1, «Уровень_2» — 2, «Уровень_3» — 3);

• неиерархические категории — «Категория_1», «Категория_2».

Мандатное управление доступом процессов (субъект-сессий) к ресурсам (сущностям) основано на реализации соответствующего механизма в ядре ОССН. При этом принятие решения о запрете или разрешении доступа субъект-сессии к сущности осуществляется в соответствии с правилами, описанными в рамках МРОСЛ ДП- модели, и зависит от запрашиваемого вида доступа (чтение, запись, применение права доступа на выполнение) и мандатного контекста (используемых в запросе уровней конфиденциальности, доступа и целостности).

Для администрирования параметров мандатных управления доступом и контроля

целостности применяются следующие команды и графические утилиты (примеры использования которых рассмотрены в главах 1 и 3):

• *pdpl-user*— команда просмотра и изменения допустимых мандатных уровней и неиерархические категорий учётных записей пользователей;

• *pdpl-file* — команда установки параметров мандатного управления доступом на сущность файловой системы;

• *pdp-id* — команда вывода параметров мандатных управления доступом и контроля целостности для текущей сессии;

• *userlev* — команда просмотра и редактирования уровней доступа, заданных в ОССН;

• *usercat* — команда просмотра и редактирования неиерархических категорий учётных записей пользователей в ОССН;

• *usercaps* — команда просмотра и редактирования привилегий учётных: записей пользователей;

• *fly-admin-smc* — графическая утилита, позволяющая решать весь комплекс задач по администрированию учётных записей пользователей и групп, в том числе администрировать параметры мандатных управления доступом и контроля целостности.

Используемое методическое и лабораторное обеспечение

1. ОССН версии 1.6, в которой создана учётная запись пользователя *user,* с параметрами: максимальный и минимальный уровни доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий», входит в группу администраторов — *astra-admin* (вторичная группа), разрешено выполнение привилегированных команд *(sudo).*
2. Документация: «Операционная система специального назначения *«Astra Linux Special Edition».* Руководство администратора. Часть 1», «Операционная система специального назначения *«Astra Linux Special Edition».* Руководство по КСЗ. Часть 1», «Операционная система специального назначения *«Astra Linux Special Edition».* Руководство пользователя».
3. Для выполнения работы в течение двух занятий необходимо обеспечить возможность сохранения состояния ОССН за счёт применения технологий виртуализации (создания виртуальных машин с ОССН).

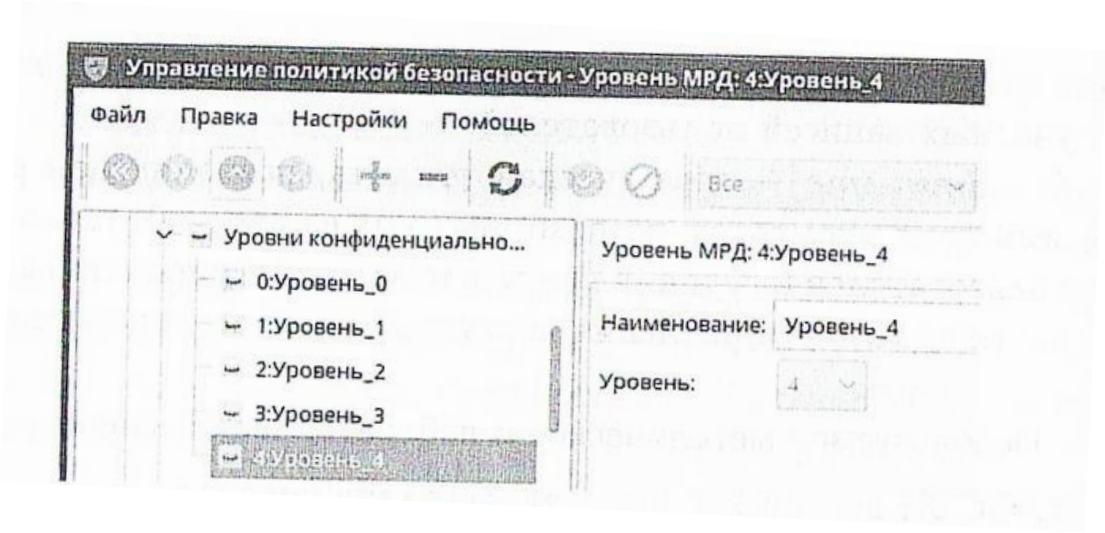
Порядок выполнения работы

1. Начать работу со входа в ОССН в графическом режиме с учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий»).
2. Запустить графическую утилиту редактирования учётных записей пользователей «Политика безопасности» через меню «Панель управления» главного пользовательского меню.

3. Модифицировать параметры мандатного управления доступом, для этого осуществить следующие действия:

• открыть раздел «Мандатные атрибуты», «Уровни конфиденциальности» и выбрать «0»: Уровень_0» и переименовать данный уровень доступа: «Уровень0»;

• выполнить создание уровня доступа с именем «Уровень _4», задав значение равное 4, после чего проверить наличие записи «Уровень_4» в списке «Уровни конфиденциальности»:



• выполнить обратное переименование: «Уровень0» в «Уровень _0».

4. Создать учётную запись пользователя *user*1*,* установив максимальный уровень доступа: «Уровень_4».

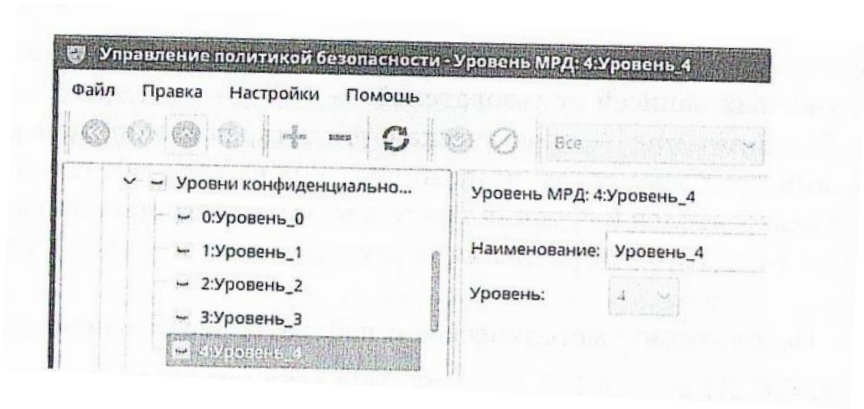
5. Выполнить удаление уровня доступа 4 из раздела «Уровни конфиденциальности» путём выбора в контекстном меню пункта «Удалить».

6. Открыть учётную запись пользователя *user*1 и в вкладке «МРД» в элементе «Максимальный уровень» проверить отсутствие записи имени уровня, при этом, в списке выбора уровня «Уровень_4» также будет отсутствовать.

7. Вывести в терминал *Fly* параметры мандатного управления доступом для учётной записи пользователя *user*1*.* Для этого выполнить следующие действия:

• запустить терминал *Fly* и перейти в каталог / *etc /parsec /macdb;*

• прочитать параметры учётной записи *user*1 командой *sudo grep “^user*1*:”* *:



- определить максимальный уровень доступа учётной записи user1 командой *sudo grep* "user1:" * |*cut -d: -f* 5;
- определить минимальный уровень доступа учётной записи user1 командой *sudo grep* "user1" * |*cut -d: -f* 3 и проверить его соответствие данным, отображаемым в графической утилите «Политика безопасности».

Создать неиерархические категории с использованием графической утилиты «Политика безопасности». Для этого выполнить следующие действия:

- в разделе «Категории» удалить исходные неиерархические категории;
- затем создать новую неиерархическую категорию с именем *«Otdel*1*»,* «Разряд» — 0;
- в разделе «Категории» создать новые неиерархические категории: *«Otdel*2*»* («Разряд» — 1), *«Upravlenie»* («Разряд» — 2).

9. Изменить набор неиерархических категорий с использованием графической утилиты «Политика безопасности», для этого выполнить следующие действия в разделе «Категории»:

• выбрать неиерархическую категорию *«Otdel*1*»* и ввести наименование «Отдел_1»;

• аналогично переименовать неиерархические категории «*Otdel*2*»* и *«Upravlenie»* в «Отдел_2» и «Управление», соответственно;

• проанализировать возможность одновременного изменения элемента «Разряд».

- Изменить мандатный уровень доступа с использованием графической утилиты «Политика безопасности», для этого выполнить следующие действия:

• создать новую группу с именем *«office*1*»* и задать первичную группу учётной записи пользователя *user*1 — *«office*1*»*;

• создать новую учётную запись пользователя *user*0 и установить её первичную группу — *«office*1*»*;

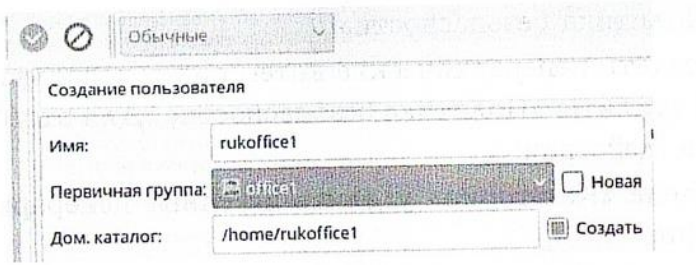
• в вкладке «Дополнительные» осуществить попытку выбора минимального набора неиерархических категорий — «Отдел_2» и проанализировать результат;

• в вкладке «Дополнительные» выбрать максимальный уровень доступа — «Уровень_3», максимальный набор неиерархических категорий — «Отдел_2», после чего задать

минимальный набор неиерархических категорий — «Отдел_2»;

• открыть параметры учётной записи пользователя *user*1 и выбрать максимальный уровень доступа — «Уровень_3», максимальный набор неиерархических категорий — «Отдел_1», минимальный набор неиерархических категорий — «Отдел_1»;

• создать учётную запись пользователя *rukoffice*1 и задать первичную группу: *«office*1*»*:



• в вкладке «Дополнительные» выбрать максимальный уровень: «Уровень_3», максимальный набор категорий: «Отдел_1», «Отдел_2», «Управление».

- Создать общий каталог для работы от имени учётных записей пользователей user1, user2, rukoffice1 в каталоге /home/work. При этом, для работы от имени учётных записей пользователей с наборами неиерархическими категорий равными «Отдел_1», «Отдел_2» и «Управление» выделить отдельные каталоги «*otdel*1», «*otdel*2» и *«upr»* соответственно. При этом обеспечить хранение файлов с различными уровнями конфиденциальности в каталогах с использованием специального атрибута *CCNR,* для чего осуществить следующие действия:

• запустить терминал *Fly* в «привилегированном» режиме командой *sudo fly-term;*

• создать каталог *work* и задать параметры мандатного и дискреционного управления доступом командами:

mkdir /home/work

chown user: *office*1 */home/work*

chmod 750 */home/work*

pdpl-file 3:0: Отдел_1, Отдел_2, Управление: *ccnr /home/work*

• создать каталог для работы от имени учётных записей пользователей с набором неиерархических категорий равным «Отдел_1» и задать параметры мандатного и дискреционного управления доступом командами:

cd /home/work

*mkdir otdel*1

*chown user*1*: office*1 *otdel*1

chmod 770 *otdel*1

pdpl-file 3:0: Отдел_1: *ccnr otdel*1

• создать каталог для работы от имени учётных записей пользователей с набором неиерархических категорий равным «Отдел_2» и задать параметры мандатного и

дискреционного управления доступом командами:

*mkdir otdel*2

*chown user*2*: office*1 *otdel*2

chmod 770 *otdel*2

pdpl-file 3:0: Отдел_2: *ccnr otdel*2

• создать каталог *upr* для работы от имени учётных записей пользователей с набором неиерархических категорий равным «Управление» командами:

mkdir upr

*chown rukoffice*1*: office*1 *upr*

chmod 770 *upr*

pdpl-file 3:0: Управление: *ccnr upr*

• создать вложенные каталоги У1, У2, У3 в каталогах *otde1*1*, otdel*2*, upr* командой:

mkdir {otdel{l,2}, upr} /У {1,2,3}

• установить для каталогов *otdel*1*, otdel*2*, upr* необходимые уровни (см. команды для каталога *upr*):

pdpl-file 1:0: Управление: 0/ home/ work/ upr/У 1

pdpl-file 2:0: Управление: 0/ home/ work/ upr/У 2

pdpl-file 3:0: Управление: 0/ home/ work/ upr/У3

*chown rukoffice*1*: office*1 *upr/У {*1,2,3}

chmod 770 *upr/У {1,2,3}*

- Выполнить последовательные входы в ОССН с учётной записью пользователя *user*1 (неиерархическая категория — «Отдел_1», уровни доступа 1, 2, 3). При работе на уровнях доступа 1, 2 и 3 создать в каталоге /*home/work/otdel*1/уровень*X* файлы с именами 11.*txt,* 12.*txt,* 13.*txt* соответственно, и установить дискреционные права доступа с разрешением на запись и чтение для группы *office*1 в графическом файловом менеджере *Fly (fly-fm).*
- Выполнить последовательные входы в ОССН с учётной записью пользователя *usee2* (неиерархическая категория — «Отдел_2», уровни доступа 1, 2, 3). При работе на мандатных уровнях доступа 1, 2 и 3 создать в каталоге */ home/work/otdel*2*/уровеньX* файлы с именами 21.*txt*, 22.*txt*, 23.*txt* соответственно, и установить дискреционные права доступа с разрешением на запись и чтение для группы *office*1 в файловом менеджере *Fly.12*
- Войти в ОССН с учётной записью пользователя *rukoffice*1 (уровень доступа — 3, неиерархическая категория — «Отдел2») и проверить возможность получения следующих доступов к файлам: доступ на чтение к файлам 21.*txt*, 22.*txt*, 23.t*xt*, доступ на запись к файлу 23.*txt*.
- Войти в ОССН с учётной записью пользователя *rukoffice*1 (уровень доступа — 2, неиерархическая категория — «Отдел_1») и проверить возможность получения следующих

доступов к файлам: доступ на чтение к файлам 11. *txt*, 12. *txt*, доступ на запись к файлу 12.*txt*.

- Войти в ОССН с учётной записью пользователя *rukoffice*1 (уровень доступа — 3, набор неиерархических категорий — «Отдел_1», «Отдел_2», «Управление») и проверить возможность получения доступа на чтение к файлам 11. *txt*, 12.*txt*, 13.*txt*, 21.*txt*, 22.*txt*, 23. *txt*.
- Войти в ОССН с учётной записью пользователя *rukoffice*1 (уровень доступа — 3, неиерархическая категория — «Управление»). Создать файл *u3.txt в* каталоге */home/work/upr/*У3.
- Войти в ОССН с учётной записью пользователя *rukoffice*1 (уровень доступа — 3, набор неиерархических категорий: «Отдел_1», «Отдел_2», «Управление») и проверить возможность получения следующих доступов к файлам: доступ на запись к файлу *u3.txt,* доступ на чтение к файлам *u3.txt,* 11.*txt*, 12.*txt*, 13.*txt*, 21.*txt*, 22.*txt*, 23.*txt*.
- Для доступа к терминалу *Fly* настроить включение учётных записей пользователей user1, *user2, rukoffice*1 во вторичную группу *astra-console.* Это позволит данным учётным записям пользователей запускать терминал *Fly с* использованием комбинации *Win+R.*
- Вывести в терминал *Fly* параметры мандатного управления доступом и мандатного контроля целостности для учётных записей пользователей. Для этого выполнить следующие действия:

• войти в ОССН с учётной записью пользователя *rukoffice*1 (уровень доступа — 2, набор неиерархических категорий: «Отдел_1», «Управление»);

• в терминале *Fly* выполнить команду *pdp-id -a,* проанализировать результат;

• выполнить избирательный вывод параметров мандатного управления доступом (с числовыми значениями) командами *pdp-id -l* и *pdp-id* -с;

• выполнить избирательный вывод параметров мандатного управления доступом (с именами) командами *pdp-id -ln* и *pdp- id -сп.*

- Изменить параметры мандатного управления доступом и мандатного контроля целостности учётной записи пользователя *rukoffice*1*.* Для этого выполнить следующие действия:

• войти в ОССН с учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий») и запустить терминал *Fly* в «привилегированном» режиме командой *sudo fly-term*;

• изменить минимальный и максимальный уровни доступа учётной записи пользователя *rukoffice*1 командой *pdpl-user -l* 0:2 *rukoffice*1*,* а также минимальный и максимальный наборы неиерархических категорий пользователя *rukoffice*1 командой *pdpl- user -c* 0:2 *rukoffice*1;

• обнулить значения уровней доступа и наборов неиерархических категорий в параметрах учётной записи пользователя *rukoffice*1 командой *pdpl-user -z rukoffice*1;

• установить значения уровней доступа и наборов неиерархических категорий в параметрах учётной записи пользователя *rukoffice*1 командой *pdpl-user -l* 1:3 *-c* 0:7 *rukoffice*1;

- Считать параметры мандатного управления доступом и мандатного контроля целостности учётной записи пользователя *rukoffice*1 из файлов настроек. Для этого выполнить следующие действия:

• перейти в каталог / *etc/parsec/ macdb* и считать минимальный и максимальный уровени доступа командами *grep "rukoffice*1*" * | *cut-d : -f* 3 и *grep "rukoffice*1*" * | *cut-d* : *-f* 5 соответственно;

• считать минимальный и максимальный наборы неиерархических категорий командами *grep "rukoffice*1*" * | *cut -d : -f* 4 и *grep "rukoffice*1*" * | *cut-d : -f* 6 соответственно.

- Создать и модифицировать мандатные уровни доступа, осуществив следующие действия:

• вывести в терминал созданные уровни доступа командой *userlev* и сравнить полученные данные с настройками в утилите «Политика безопасности»;

• добавить новый уровень доступа с именем «Уровень_4» (значение 4) командой *userlev* Уровень_4 *--add* 4 и вывести в терминал уровни доступа командой *userlev;*

• выполнить переименование уровня доступа «Уровень_4» в «НовыйУровень» командой *userlev* Уровень_4 — *rename* НовыйУровень;

• добавить возможность работы от имени учётной записи пользователя *rukoffice*1 на уровне доступа 4 командой *pdpl-user -l* 1:4 *rukoffice*1;

• выполнить попытку изменения значения уровня доступа «НовыйУровень» на 3 командой *userlev* НовыйУровень —*modify* 3, проанализировать результат;

• изменить значение уровня доступа «НовыйУровень» на 5 командой *userlev* НовыйУровень —*modify* 5 и вывести в терминал максимальный уровень доступа учётной записи пользователя *rukoffice*1 командой *pdpl-user rukoffice*1*,* проанализировать результат;

• установить максимальный уровень доступа учётной записи пользователя *rukoffice*1 равным 5 командой *pdpl-user -l* 1:5 *rukoffice*1;

• удалить уровень доступа с именем «НовыйУровень» командой *userlev* НовыйУровень *-d* и определить максимальный уровень доступа учётной записи пользователя *rukoffice*1 командой *pdpl- user rukoffice*1*,* проанализировать результат;

• восстановить набор неиерархических категорий и уровней доступа учётной записи пользователя *rukoffice*1 командой *pdpl- user -l* 1:3 -c 0:7 *rukoffice*1*.*

- Создать и модифицировать неиерархические категории:

• в терминале *Fly,* запущенном в «привилегированном» режиме, вывести неиерархические категории командой *usercat*;

• добавить новую неиерархическую категорию командой *usercat otdel*3 —*add* 0x8;

• переименовать неиерархическую категорию *«otdel*3*»* в «Отдел_3» командой *usercat otdel*3 *--rename* Отдел_3;

• осуществить попытку модификации наборов неиерархических категорий учётной записи пользователя *rukoffice*1 командой *pdpl-user -c* 0:15 *rukoffice*1*,* проанализировать результат;

• добавить неиерархическую категорию «Отдел_3» в наборы неиерархических категорий учётной записи пользователя *rukoffice*1 командой *pdpl-user -c 3:F rukoffice*1*,* обратить внимание на то, что неиерархическая категория задаётся в шестнадцатеричном формате;

• осуществить попытку изменения значения неиерархической категории «Отдел_3» на значение 2 командой *usercat* Отдел_3 -- *modify* 2, проанализировать результат;

• изменить значение неиерархической категории «Отдел_3» на 0x10 командой *usercat* Отдел_3 *--modify* 10;

• изменить значение неиерархической категории «Отдел_3» на 0x20 командой *usercat* Отдел_3 *—modify* 0x20, обратить внимание на то, что независимо от указания типа числа по префиксу «0x» (десятичное или шестнадцатеричное) значение неиерархической категории задаётся в шестнадцатеричном формате;

• удалить неиерархическую категорию «Отдел_3» командой *usercat* Отдел_3 *—delete*;

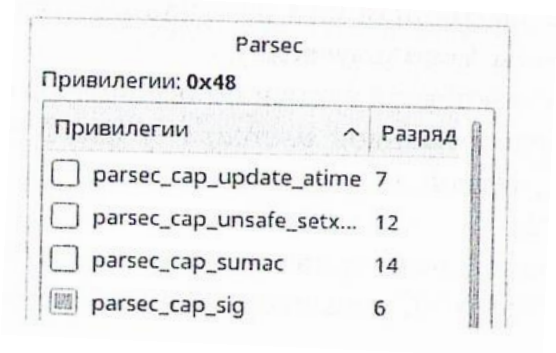
• изменить значение неиерархической категории «Управление» на 0x10 командой *usercat* Управление — *modify* 10, проанализировать результат по данным, выводимым командой *pdpl-user rukoffice1*;

• изменить значение неиерархической категории «Управление» на 4 командой *usercat* Управление *-modify* 4.

- Для настройки привилегий учётных записей пользователей осуществить следующие действия:

• вывести в терминал заданные в ОССН привилегии учётных записей пользователей командой *usercaps,* при работе в терминале *Fly* в «привилегированном» режиме;

• запустить графическую утилиту «Политика безопасности» и открыть настройки учётной записи пользователя user1, в вкладке «Привилегии» установить *Linux*-привилегии *cap_kill, cap_fow- пег* и *PARSEC*-привилегии parsec_cap_chmac, parsec_cap_sig, после чего закончить работу с утилитой:



• вывести привилегии учётной записи пользователя user1 командой *usercaps user*1;

• в графической утилите «Политика безопасности» открыть параметры учётной записи пользователя *user,* в вкладке «Привилегии» выбрать *Linux-привилегии cap-kill, cap_fowner и PARSE*C-привилегии *parsec_cap_chmac, parsec_cap.sig-,*

• запустить терминал *Fly* в «непривилегированном» режиме командой *fly-term u* осуществить попытку запуска команды *usercaps',*

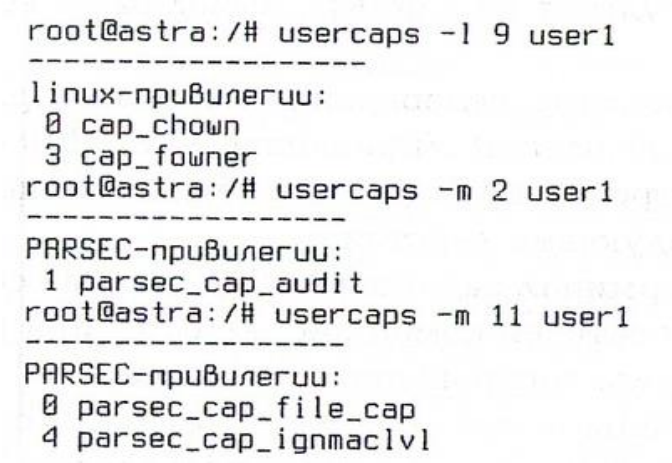
• определить расположение файла *usercaps* командой *which usercaps,* выполненной из «привилегированного» режима, а затем выполнить в «непривилегированном» режиме команду */usr/ sbin/usercaps,* проанализировать результат;

• запустить терминал *Fly* в «привилегированном» режиме командой *sudo fly-term u* выполнить модификацию *Linux*-привилегий и *PARSEC-привилегий* командами:

usercaps -l 9 *user*1*;*

usercaps -m 2 *user*1;

usercaps -m 11 *user*1:



• с использованием графической утилиты «Политика безопасности» определить установленные привилегии и формат параметра модификации привилегий учётных записей пользователей (десятичная, восьмеричная или шестнадцатеричная система счисления при этом используется?);

• установить для учётной записи пользователя *user1* полный список привилегий командой *usercaps -f user1,* затем удалить все привилегии учётной записи пользователя user1 командой *usercaps -z user*1*-,*

• вывести списки *Linux-привилегий и PARSEC-привилегий* командами *usercaps -L* и *usercaps -M* соответственно.

Содержание отчёта по выполненной работе

В отчёте о выполненной работе необходимо указать:

1. Полный перечень использованных команд с кратким описанием их назначения.
2. Примеры выполнения команд, которые были использованы в ходе работы с описанием

результатов их выполнения.

3. Описание порядка работы с графической утилитой «Политика безопасности» при выполнении следующих действий:

• создание, изменение и удаление уровней доступа и неиерархических категорий;

• настройка уровней доступа и неиерархических категорий учётных записей пользователей;

• создание общих каталогов для совместного использования несколькими учётными записями пользователей.

4. Описание порядка работы и команд, использованных при осуществлении следующих действий:

• настройка уровней доступа и неиерархических категорий в ОССН;

• настройка уровней доступа и неиерархических категорий учётных записей пользователей.

5. Описание особенностей функционирования команд при работе в «привилегированном» и «непривилегированном» режимах.

6. Список и назначение системных файлов, связанных с хранением параметров мандатных управления доступом и контроля целостности.

7. Описание команд при настройке привилегий учётных записей пользователей.

Контрольные вопросы

1. Какие уровни доступа и неиерархические категории создаются при установке ОССН?

2. Как настроить минимальный и максимальный уровни доступа учётной записи пользователя с использованием графической утилиты *fly-admin-smc?*

3. Как добавить новые уровни доступа и неиерархические категории в ОССН?

4. Какие имеются особенности удаления и модификации уровней доступа и неиерархических категорий в ОССН?

5. Какие команды используются для создания, модификации и удаления уровней доступа и неиерархических категорий в ОССН?

6. Какие команды используются для настройки привилегий учётных записей пользователей?

7. Как принудительно удалить все привилегии для заданной учётной записи пользователя?

8. Какие существуют особенности настройки привилегий учётных записей пользователей в «непривилегированном» режиме?

4.3. Лабораторная работа № 3.
Организация файловой системы ОССН для работы пользователей в рамках мандатного управления доступом и мандатного контроля целостности

Цель работы:

Изучить особенности настройки мандатного управления доступом и мандатного контроля целостности к каталогам файловой системы ОССН для совместной работы от имени учётных записей пользователей с различными уровнями доступа с документами (файлами) с различными уровнями конфиденциальности.

Время выполнения работы 4 академических часа.

Краткие теоретические сведения

В ОССН мандатное управление доступом интегрировано в файловую систему, за счёт хранения в ней не только дискреционных прав доступа, но и мандатных меток файлов с дополнительными специальными атрибутами, определёнными в рамках МРОСЛ ДП- модели. Параметрами мандатного управления доступом и контроля целостности (мандатными метками) каждой сущности файловой системы являются:

- уровень конфиденциальности;
- набор неиерархических категорий;
- уровень целостности;
- специальные атрибуты *{CCNR, CCNRI, E - Hole, W - Hole).*

При работе с сущностями файловой системы специальные атрибуты отображаются следующим образом: если установлен атрибут *EHole,* то указывается *ehole (whole* для *W - Hole),* для *CCNR ma CCNRI* (данные атрибуты по сравнению заданными в МРОСЛ ДП-моделыо атрибутами *CCR* и *CCRI* в ОССН реализованы инвертированными) — *сспг* и *сспгг* соответственно; если установлены оба атрибута *(CCNR* и *CCNRI),* то указывается *CCNRA*. Специальный атрибут *CCNR* позволяет осуществлять доступ внутрь контейнера (читать содержимое каталога) без учёта уровня конфиденциальности каталога. Аналогично используется атрибут *CCNRI,* только для работы с мандатным уровнем целостности.

Наличие у сущности файловой системы мандатных меток целостности позволяют дополнительно усилить защиту от несанкционированной модификации файлов, влияющих на безопасность ОССН. Это реализуется установкой уровня целостности «Высокий» на критически важных сущностях (файлах) файловой системы, что предотвращает изменение или удаление таких файлов процессами с низким уровнем целостности.

Для администрирования параметров мандатного управления доступом и мандатного контроля целостности файловой системы в лабораторной работе применяются следующие команды и графические утилиты:

- *pdpl-file* — команда модификации параметров мандатного управления доступом сущностей файловой системы;
- *userlev* — команда просмотра и редактирования уровней доступа, заданных в ОССН;
- *fly-admin-smc* — графическая утилита, позволяющая решать весь комплекс задач по администрированию учётных записей пользователей и групп, в том числе администрировать параметры мандатных управления доступом и контроля целостности.

Используемое методическое и лабораторное обеспечение

- ОССН версии 1.6, в которой создана учётная запись пользователя *user,* с параметрами: максимальный и минимальный уровни доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий», входит в группу администраторов — *astra- admin* (вторичная группа), разрешено выполнение привилегированных команд *(sudo).*
- Документация: «Операционная система специального назначения *«Astra Linux Special Edition».* Руководство администратора. Часть 1», «Операционная система специального назначения *«Astra Linux Special Edition».* Руководство по КСЗ. Часть 1», «Операционная система специального назначения *«Astra Linux Special Edition».* Руководство пользователя».
- Для выполнения работы в течение двух занятий необходимо обеспечить возможность сохранения состояния ОССН за счёт применения технологий виртуализации (создания виртуальных машин с ОССН).

Порядок выполнения работы

- Выполнить вход в ОССН в графическом режиме с учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий»).
- Проверить корректность включения в ОССН мандатного контроля целостности. Для этого запустить графическую утилиту «Политика безопасности» через меню «Панель управления», «Безопасность» главного пользовательского меню. Открыть «Монитор безопасности» и убедиться в корректном статусе параметров «Режим Мандатного Контроля Целостности» и «Установленный на ФС высокий уровень Мандатного Контроля Целостности».
- Запустить терминал *Fly* и выполнить следующие команды для создания группы пользователей:

- выполнить попытку создания группы командой *addgroup Office-,*
- выполнить попытку создания группы командой *sudo addgroup*

- *Office-*
- проанализировать полученные ошибки и внести корректировки в команду создания группы;
- создать группу *office* командой *sudo addgroup office.*

- Запустить графическую утилиту «Политика безопасности» через меню «Панель управления», «Безопасность» главного пользовательского меню.

  Разрешить работать учётной записи пользователя *user* до мандатного уровня доступа 3 и с любыми созданными в ОССН неиерархическими категориями.

- Создать следующие учётные записи пользователей:

- учётную запись пользователя *managerl,* задав ей минимальный уровень доступа «Уровень_1», максимальный уровень доступа «Уровень_2» (при этом выявить правильную последовательность задания данных уровней), минимальная и максимальная неиерархическая категория «Категория.!.» (при этом обратить внимание на необходимость задания минимальной неиерархической категории);
- учётную запись пользователя *manager?,* задав ей минимальный уровень доступа «Уровень.1», максимальный уровень доступа «Уровень_2», минимальная и максимальная неиерархическая категория «Категория^»;
- с использованием команды *sudo* установить для учётной записи пользователя *manager?* первичную группу *office,* затем установить для учётной записи пользователя *managerl* первичную группу *office* (при этом использовать команды типа *usermod -д office имя-пользователя).*

- Создать каталог для работы пользователей с различными мандатными уровнями доступа, для этого осуществить следующие действия:

- создать каталог */home/share u* установить на него мандатные атрибуты 2:0:3: *CCNRA* командой *pdpl-file* 2:0:3: *CCNRA /home/ share*\
- разрешить пользователям группы *office* записывать и читать каталог /*home/share: setfacl -mg:office:rwx /home/share-,*
- создать каталоги *otdell* и *otdel2* внутри каталога /*home/share',*
- назначить дискреционные права доступа для каталогов */home/ share/otdell* и */home/share/otdel2: setfacl* -m *g:office:rwx /home/ share/ otdell* и *setfacl -m g:office'.rwx /home/share/otdel2*\
- установить мандатные атрибуты 2:0:1: *CCNRA* и 2:0:2: *CCNRA* для каталогов *otdell u. otdel2* соответственно.

8. Настроить каталоги *otdell* и *otdel2* для работы пользователей с различным мандатным уровнем доступа:

- создать каталоги «ДСП» и «С» внутри каталогов *otdell* и *otdel2* командой: *mkdir-p /home/ share/ otdel{l/2} /*

- на каталог «ДСП» внутри каталога *otdell* назначить следующие метки 1:0:1:0;
- на каталог «С» внутри каталога *otdell* назначить следующие метки 2:0:1:0;
- аналогично создать каталоги в */home/share/otdel2*;
- вывести мандатные метки и дискреционные атрибуты каталога *share* рекурсивно, после чего определить назначение используемых опций команды *pdp-ls*:

user@astra:~$ sudo pdp-ls /home/share/ -MnR
/home/share/:
итого 16
druxruixr-x+m— 4 0 0 2:0:0xl:0x3 otdell dridxriiixr-x+m-- 4 0 0 2:0:0x2:0x3 otdel2
/home/share/otdel1: итого 16
druxruixr-x+m-- 2 0 0 1: 0:0x1:0x0 ДСП drwxruixr-x+m-- 2 0 0 2:0:0xl:0x0 C
/home/share/otdel1/ДСП: итого 0

/home/share/otdel1/C: итого 0
/home/share/otdel2: итого 16
druxriijxr-x+m— 2 0 0 1:0:0x2:0x0 ДСП druxruxr-x+m-- 200 2:0:0x2:0x0 C
/home/share/otde12/ДСП: итого 0
/home/share/otde12/C: итого 0

9. Изучить возможность задания дискреционных атрибутов вложенных каталогов */home/share* с использованием следующих команд:

- сначала очистить все созданные дискреционные атрибуты:
- |root@astra:~H setfacl -b /home/share/otdelfl,21 |root@astra;~H setfacl -b /home/share/otdel{l,21/{flCn,C}
- выполнить команду их создания:

root@astra:~H setfacl -m g:office:ruix /home/share/otdel{1,2>/<ДСП,C> root@astra:~H setfacl -m g:office:rux /home/share/otdel{1,21

10. Модифицировать уровни доступа и неиерархические категории, заданные в ОССН, путём переименования неиерархических категорий в «Отдел1» и «Отдел2», уровней 0-2 в «НС», «ДСП», «С».

11. Выполнить вход в ОССН в графическом режиме с учётной записью пользователя *managerl* (уровень доступа — «ДСП», неиерархические категории — «Отдел1», уровень целостности — «Низкий»):

- в графической утилите *fly.fm* выполнить попытки осуществления доступа и создания файлов *{dsp-manl.txt* и *dsp-doc-odi)* в подкаталогах каталога */home/share-,*
- выполнить изменение файла *dsp-manl.txt',*
- осуществить изменение файла *dsp-doc.odt.*

- Выполнить вход в ОССН в графическом режиме с учётной записью пользователя *managerl* (уровень доступа — «С», неиерархические категории
- «Отдел1», уровень целостности — «Низкий»):
- в файловом менеджере *fly-fm* выполнить попытки осуществления доступа и создания файлов в подкаталогах каталога */home/ share',*
- произвести изменения файла *c-manl.txt-,*
- выполнить попытку изменения файла *dsp-manl.txt.*

12. Выявить особенность функционирования администратора ОССН, в том числе на «Высоком» уровне целостности:

- выполнить вход в ОССН в графическом режиме с учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Низкий»);
- осуществить доступ к каталогу */home/share* и его подкаталогам, определить возможность доступа на чтение администратора ОССН на «Низком» мандатном уровне целостности к сущностям с более высоким уровнем конфиденциальности;
- осуществить аналогичные действия при работе через *sudo* и сравнить результат;
- выполнить выход и вход в ОССН в графическом режиме с учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий»);
- выполнить доступ к каталогу */home/share* и его подкаталогам, определить возможность доступа на чтение при работе от имени учётной записи пользователя *user* на «Высоком» мандатном уровне целостности к сущностям с более высоким уровнем конфиденциальности;
- « осуществить аналогичные действия при работе через *sudo.*
- отделов, которые могут менять мандатные метки созданных файлов:
- создать учётную запись пользователя *ceol* и задать ей минимальный уровень доступа «НС», максимальный уровень доступа «С», минимальная и максимальная неиерархическая категория «Отдел1»;
- установить необходимые привилегии (*cap-chouin, parsec_cap_ chmac)* с использованием команды *usercaps ceol -l* 1 *-m* 8;
- аналогично создать учётную запись пользователя *ceo2* и задать ей минимальный уровень доступа «НС», максимальный уровень доступа «С», минимальная и максимальная неиерархическая категория «Отдел2» и аналогичные привилегии.

15. Установить для учётных записей пользователей ceol и *ceo2:*

- группу по умолчанию *office: usermod-y office ceol && usermod -g office ceo2*\

- установить дополнительные группы для доступа к *fly-term* для учётных записей пользователей ceol, ceo2: *astra-console* (для этого выполнить команду *usermod* с параметрами *-a -G,* группой и именем учётной записи пользователя);
- найти в графической утилите «Политика безопасности» параметр, позволяющий отключить доступ к терминалу *Fly* для пользователей, и активировать данный параметр.

16. Выполнить попытку изменения мандатных меток конфиденциальности файлов:

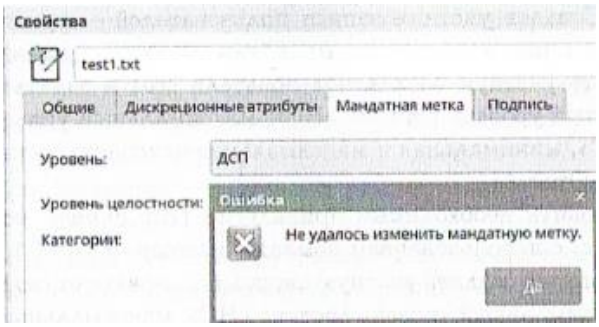
выполнить вход в ОССН в графическом режиме с учётной записью пользователя ceol (уровень доступа — «С», неиерархические категории — «Отдел1», уровень целостности — «Низкий»);

создать в каталоге */home/share/otdell/*С файл *testl.txt-,*

с использованием меню графического файлового менеджера *fly- fm* вывести дискреционные и мандатные атрибуты файла *testl.txt-,*

выполнить попытку установки мандатных атрибутов файла *testl.txt,* при этом необходимо сменить уровень с «С» на «ДСП» (ошибка связана с уровнем конфиденциальности файла *testl.txt)* (см. рисунок на следующей странице);

выполнить попытку копирования файла *testl.txt* в каталог */Kome/share/* oideZl/ДСП (ошибка заключается в том, что уровень доступа текущего процесса, осуществляющего копирование, выше уровня конфиденциальности каталога назначения, т. е. блокируется запись сверху вниз):



- копировать файл *testl.txt* в каталог */home/share/otdell* («С», *CCNRA);*
- сменить мандатные атрибуты файла *testl.txt,* при этом необходимо сменить уровень с «С» на «ДСП».

17. Выполнить вход в ОССН в графическом режиме с учётной записью пользователя *ceol* (уровень доступа — «ДСП», неиерархические категории — «Отдел1», уровень целостности — «Низкий»).

18. Скопировать (или перенести) файл *testl.txt в* каталог */ home/ share/ otdell/*ДСП.

19. При необходимости удаления файла */ home/ share/ testl.txt.* войти в ОССН с учётной записью пользователя ceol («ДСП», «Отдел!»).

20. Выполнить вход в ОССН в графическом режиме с учётной записью пользователя *managerl* (уровень доступа — «С», неиерархические категории — «Отдел1», уровень целостности — «Низкий»):

- в файловом менеджере *fly-fm* выполнить попытки доступа к файлам и создания файлов в подкаталогах «ДСП» и «С» каталога */home/ share/ otdell',*
- выполнить попытку открытия файла *dsp-doc.odt',*
- выявить возможные ошибки работы с файлом.

21. Возможной причиной того, что файл *dsp-doc.odt* не может быть открыт с уровня «С» является особенность работы *Libre Office,* который создаёт в текущем каталоге файл ~.Zocfc. *Файл. о. /\ля* подтверждения этого необходимо войти в ОССН с использованием терминала *Fly с* учётными записями пользователей *managerl* и *managed:*

- выполнить вход в ОССН в графическом режиме с учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий»);
- добавить пользователям *managerl* и *manager^* группу *astra- console командой: usermod-a-G astra-console managerl, user-mod -a -G astra-console manager!* для возможности запуска терминале *Fly*
- выполнить повторный вход в ОССН в графическом режиме с учётной записью пользователя *managerl* (уровень доступа — «С», неиерархические категории — «Отдел1», уровень целостности — «Низкий»);
- запустить *fly-term* с использованием комбинации клавиш *«Astra»* + «Л».

22. С использованием меню графического файлового менеджера *fly-fm* создать «Документ *Libre Office»* в каталоге */home/ share/ otdell/C* файл *c-doc.odt, l.odt.* Открыть файл *l.odt,* перейти в терминале *Fly* в каталог / */home/ share/ otdell/C* и вывести содержимое каталога (обратить внимание на наличие файла .~Zocfc.l.o): managerl@astra:/home/share/otdel1/C$ ls – la итого 28

23. В *LibreOffice* открыть меню «Сервис», «Параметры», *«Lib- reOffi.ce»,* «Расширенные возможности», «Открыть экспертные настройки». Ввести фильтр: *uselocking.*

24. Изменить значение на *false* и сохранить изменения.

25. Ввести фильтр *«lock»* и изменить значения свойств *«UseDo- cumentOOoLockFile»* и *«UseDocumentSystemFileLocking» па false.*

26. Выполнить повторное открытие файла *c-doc.odt* и проверить отсутствие *«lock»*-файла.

27. Выполнить вход в ОССН в графическом режиме с учётной записью пользователя *managerl / manager!* (уровень доступа — «С»/«ДСП», неиерархические категории — «Отдел1»/«Отдел2», уровень целостности — «Низкий») в доступных комбинациях:

- в соответствующем режиме проверить возможность доступа на чтение и запись к документам, созданным *LibreOffice,* с уровнем конфиденциальности, соответствующем уровню доступа учётной записи пользователя;
- проверить необходимые настройки блокировки открываемого документа в настройках *LibreOffice* на текущем уровне;
- проверить возможность осуществления доступа на чтение (и главное проверить возможность открытия) к документам, созданным *LibreOffice,* с уровнем конфиденциальности меньше текущего уровня доступа учётной записи пользователя;
- в конце лабораторной работы вернуть ОССН к начальным настройкам.

Содержание отчёта по выполненной работе

В отчёте о выполненной работе необходимо указать:

- Полный перечень использованных команд с кратким описанием их назначения.
- Примеры выполнения команд, которые были использованы в ходе работы с описанием их результатов.
- Описание порядка работы с графической утилитой при выполнении следующих действий:
  - создание, изменение и удаление уровней доступа и неиерархических категорий;
  - создание и настройка групп пользователей.
- Описание порядка работы и команд, использованных при осуществлении следующих действий:
  - настройка уровней доступа и неиерархических категорий учётных записей пользователей;
  - модификация параметров доступа к сущностям файловой системы ОССН;
  - настройка уровней доступа и неиерархических категорий в ОССН.
- Описание особенностей настройки *Libre Office* для работы с документами в рамках мандатного управления доступом.
- Параметры мандатного управления доступом (метки) для каталогов, предназначенные для совместной работы от имени учётных записей пользователей с различным уровнем доступа.

Контрольные вопросы

- Какая утилита используется для модификации учётных записей пользователей?
- С какими мандатными атрибутами должен войти администратор ОССН для настройки параметров мандатных управления доступом и контроля целостности учётных записей пользователей?
- Какими атрибутами должны обладать каталоги для работы пользователей с различным уровнем доступа?

- Как создать учётные записи пользователей, которые имеют возможность смены мандатных меток конфиденциальности файлов, а также владельцев файлов?
- Как осуществляется доступ к расширенным настройкам *LibreOffice* для реализации совместного доступа к файлам в рамках мандатных управления доступом и контроля целостности?

Содержание отчёта по выполненной работе

В отчёте о выполненной работе необходимо указать:

- Полный перечень использованных команд с кратким описанием их назначения.
- Примеры выполнения команд, которые были использованы в ходе работы с описанием их результатов.
- Описание порядка работы с графической утилитой при выполнении следующих действий:
  - создание, изменение и удаление уровней доступа и неиерархических категорий;
  - создание и настройка групп пользователей.
- Описание порядка работы и команд, использованных при осуществнии следующих действий:
  - настройка уровней доступа и неиерархических категорий учётных записей пользователей;
  - модификация параметров доступа к сущностям файловой системы ОССН;
  - настройка уровней доступа и неиерархических категорий в ОССН.
- Описание особенностей настройки *Libre Office* для работы с документами в рамках мандатного управления доступом.
- Параметры мандатного управления доступом (метки) для каталогов, предназначенные для совместной работы от имени учётных записей пользователей с различным уровнем доступа.

4. 4. Лабораторная работа № 4.
Администрирование ОССН в рамках реализации мандатного контроля целостности

Цель работы

Выявить особенности администрирования основных параметров мандатного управления доступом в ОССН с применением графических утилит и консольных команд при активированном мандатном контроле целостности.

Время выполнения работы 2 академических часа.

Краткие теоретические сведения

В ОССН наряду с традиционной для ОС семейства *Linux* системой дискреционного управления доступом реализована система мандатного управления доступом и мандатного контроля целостности на основе МРОСЛ ДП-модели. С этим связано наличие у сущностей ОССН (файлов, каталогов) мандатных меток конфиденциальности и целостности.

Параметрами мандатного управления доступом и контроля целостности (мандатными метками) являются следующие элементы:

- уровень доступа или конфиденциальности (соответствует уровню конфиденциальности сущности или доступа субъект-сессии);
- набор неиерархических категорий сущности и субъект-сессии;
- уровень целостности сущности и субъект-сессии;
- специальные атрибуты сущности *(CCNR, CCNRI, E_Ho1e, W. Hole).*

Мандатное управление доступом процессов (субъект-сессий) к ресурсам (сущностям) основано на реализации соответствующего механизма в ядре ОССН. При этом решение о запрете или разрешении доступа субъект-сессии к сущности принимается в соответствии с правилами, описанными в рамках МРОСЛ ДП-модели, и зависит от запрашиваемого вида доступа (чтение, запись, применение права доступа на выполнение) и мандатного контекста (используемых в запросе уровней конфиденциальности, доступа и целостности).

Реализация мандатного контроля целостности позволила существенно повысить защищенность ОССН. С его применением все учётные записи пользователей, субъект-сессии, сущности и роли (при установке ОССН для элементов её файловой системы задаётся два мандатных уровня целостности — 0 и 63, хотя их может быть значительно больше, в том числе может использоваться вся решётка уровней целостности 0 до 255) чётко разделяются на два множества (отсюда и два уровня целостности: «Низкий» < «Высокий»), влияющих на целостность и безопасность ОССН или нет. Преимущества мандатного контроля целостности аналогичны преимуществам мандатного управления доступом в сравнении с дискреционным управлением доступом, т. е. обеспечивается большая ясность правил разделения

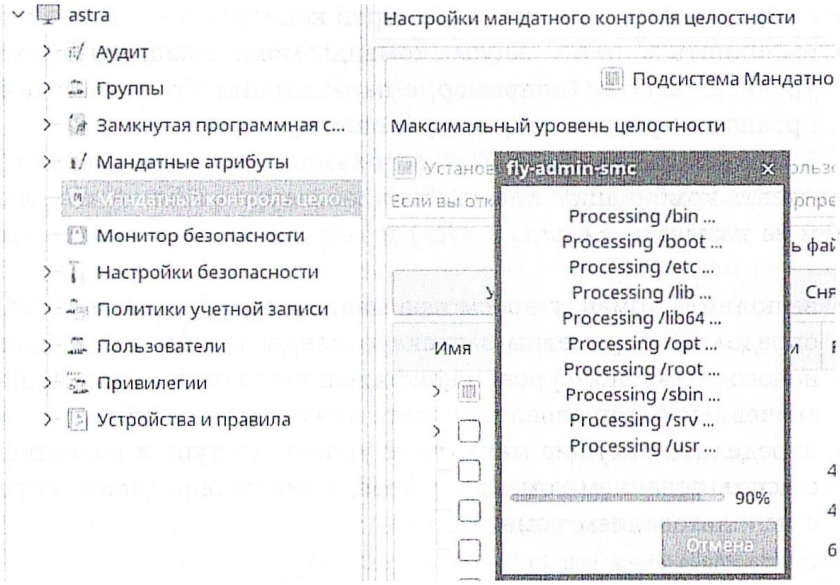
компонентов системы на критичные и некритичные с точки зрения её целостности, тем самым снижается вероятность ошибок.

Используемое методическое и лабораторное обеспечение

- ОССН версии 1. 6, в которой создана учётная запись пользователя *user,* с параметрами: максимальный и минимальный уровни доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий», входит в группу администраторов — *astra- admin* (вторичная группа), разрешено выполнение привилегированных команд (sudo).
- Документация: «Операционная система специального назначения *«Astra Linux Special Edition».* Руководство администратора. Часть 1», «Операционная система специального назначения *«Astra Linux Special Edition».* Руководство по КОЗ. Часть 1», «Операционная система специального назначения *«Astra Linux Special Edition».* Руководство пользователя».

Порядок выполнения работы

- Включить мандатный контроль целостности в файловой системе ОССН. Для этого выполнить вход в ОССН в графическом режиме с учётной записью пользователя *user* (уровень доступа — О, неиерархические категории — нет, уровень целостности — «Высокий»). Запустить графическую утилиту «Политика безопасности» через меню «Панель управления», «Безопасность» главного пользовательского меню. Открыть «Мандатный контроль целостности» и нажать кнопку «Установить». На данном этапе для сущностей файловой системы в соответствии с шаблоном устанавливаются метки мандатного уровня целостности «Высокий» (см. рисунок на следующей странице). Выполнить выход из ОССН.
- Проанализировать необходимость использования входа в систему с уровнем целостности «Высокий» для администрирования ОССН. Для этого вначале выполнить работы по администрированию, используя только уровень целостности «Низкий»:



- начать работу со входа в ОССН в графическом режиме с учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Низкий»);
- запустить графическую утилиту «Политика безопасности» через меню «Панель управления», «Безопасность» главного пользовательского меню.
- Выполнить попытку модификации параметров мандатного управления доступом, для этого осуществить следующие действия:
- открыть раздел «Мандатные атрибуты», «Уровни конфиденциальности» и выбрать «0: Уровень_0» и сделать попытку переименования данного уровня доступа;
- осуществить попытку создания мандатного уровня доступа;
- проанализировать результат и причину невозможности выполнения данной операции.
- Запустить терминал *Fly u* выполнить команду *pdp-id -a* для получения мандатных атрибутов текущего процесса (в данном случае они совпадают с мандатными атрибутами текущей сессии), проанализировать результат:
- выполнить избирательный вывод параметров мандатного управления доступом «с числовыми значениями» командами *pdp-id -I* и *pdp-id* -с;
- выполнить избирательный вывод параметров мандатного управления доступом «с именами» командами *pdp-id -In* и *pdp- id -СП',*
- определить назначение параметров команд;
- выполнить попытку запуска команды модификации мандатных уровней: *userlev* (например, с параметрами Уровень_4 *-а* 4) и проанализировать полученную ошибку.
- Запустить терминал *Fly с* использованием команды *sudo* (использовать комбинацию клавиш на клавиатуре «*Win»* + «У?», далее будем её называть *«Astra»* + «Я») и выполнить следующие команды:
- выполнить команду просмотра мандатных уровней: *userlew,*

• определить параметры запуска команды *userlev* для создания нового мандатного уровня конфиденциальности «Уровень_4» со значением 4 и проанализировать полученную ошибку;

• определить текущие мандатные уровни доступа и целостности с использованием команды *pdp-id,* а также определить группы с использованием команды *id:*

root@astra: /home/userfl pdp-id

Уровень конф. =0(Уровень_О), Уровень целостности: 0(Низкий), Категории=0х0(Нет) Роли=: (Нет: Нет) root@astra: /home/userH id

uid=0(root) gid=0(root) rpynnw=0(root) root@astra: /home/user

Н Й

- Исходя из особенностей работы от имени учётной записи пользователя администратора *(user)* в обычном режиме (без использования *sudo)* и с использованием *sudo,* можно сделать вывод, что для администрирования ОССН необходимо получение некоторых «дополнительных прав» (т. е. наличие группы с идентификатором 0 недостаточно), которые связаны с реализацией мандатного контроля целостности в ОССН. Для изучения особенностей маркировки файлов ОССН метками с высокой целостностью выполнить в запущенном терминале *Fly* следующие команды:

вывести в терминал *Fly* дискреционные права доступа и параметры мандатного управления доступом файлов и каталогов корня файловой системы командой *pdp-ls* -М /;

• определить опцию команды *pdp-ls,* позволяющую осуществлять вывод меток мандатного контроля целостности в числовом формате;

• с использованием найденной опции определить максимальное значение мандатного уровня целостности, заданное для каталога: rootEastra: /И pdp-ls -Мп / I cut -d : -f 2 I sort I grep -v "итого" I tail -1 63

• изучить с использованием команды *man* и справки назначение команд *(pdp-ls, cut, sort, grep, tail)* и использованных опций;

• выполнить команду получения списка имён файлов и каталогов с уровнем целостности «Высокий» для каталога / и /*etc* с использованием конвейера и команды *grep;*

• определить уровень целостности файла /*etc/passwd;*

• выполнить попытки модификации файла / *etc/passwd* путём добавления в него новых записей с использованием текстовых редакторов.

- Выполнить выход и вход в ОССН в графическом режиме с учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий»).
- Запустить графическую утилиту «Политика безопасности» через меню «Панель управления», «Безопасность» главного пользовательского меню.

- Выполнить модификацию параметров мандатного управления доступом. Для этого выполнить следующие действия:

• открыть раздел «Мандатные атрибуты», «Уровни конфиденциальности» и выбрать: «3: Уровень_3» и переименовать данный уровень доступа: «Уровень. Макс»;

• создать уровень доступа с именем «Уровень_4», установив значение равное 4;

• проанализировать результат и причину отсутствия ошибок.

- Выполнить модификацию параметров учётной записи пользователя *user.*

• открыть раздел «Пользователи», *«user»* и выбрать вкладку «МРД»;

• выполнить попытку изменения минимального уровня доступа учётной записи пользователя *user;*

• выполнить изменение максимального уровня доступа учётной записи пользователя *user:* «Уровень_2», категории «Катего- рия. 1»;

• выйти из графической утилиты «Политика безопасности».

- Запустить терминал *Fly* и выполнить следующие команды для определения особенностей работы в консольном режиме на различных мандатных уровнях целостности:

• выполнить команду получения мандатных атрибутов текущей сессии (с использованием команды *pdp-id)* и вывести только текущий уровень целостности: userEastra: pdp-idl auk '{print $3, $4}' УроВень целостности: 63(63),

• выполнить аналогичную команду, но с использованием команды *ps:*

user@astra: sudo pdpl-ps $(ps auxl grep fly-term I grep -v "grep" I auk '{print $2}') I cut —d : -f 2 Высокий

• изучить с использованием *man* и справки назначение применённых команд (awfc, *pdpl-ps, ps).*

- Для определения достаточности уровня целостности «Высокий» при администрировании ОССН выполнить следующие команды:

• вывести содержимое файла / *etc/passwd u* выполнить попытку его модификации: изменить *Shell* «по-умолчанию» для произвольной учётной записи пользователя;

• выполнить попытку выдачи на экран содержимого файла */etc/ shadow.*

- Изучить особенности работы приложений с уровнем целостности «Высокий» для пользователя администратора ОССН без и с использованием команды *sudo:*

• запустить терминал *Fly* с использованием команды *sudo;*

• сравнить результаты вывода команды *pdp-ls* для каталога */etc* для режима *sudo* и без него (использовать опцию отображения мандатных меток);

• вывести содержимое файла / *etc/passwd* и выполнить модификацию: изменить *Shell* «по-умолчанию» для учётной записи пользователя *«user».*

- Проверить результаты проведённой переконфигурации ОССН:

- выполнить вход в ОССН от имени учётной записи пользователя *user* на уровне доступа 2 в графическом и консольном режимах;
- после выполнения соответствующего входа вывести текущие мандатные уровни доступа и целостности с использованием команды *pdp-id.*

- Определить порядок хранения параметров мандатного управления доступом и мандатного контроля целостности в ОССН. Для этого проанализировать содержимое каталога / *etc/parsec:*

- с использованием контекстного поиска выделить файлы, содержащие наименования и значения мандатных уровней целостности и доступа в ОССН (например, файлы содержащие «Высокий»);
- создать произвольную учётную запись пользователя, установить минимальный уровень доступа 1, максимальный — 2, максимальные категории «Категория^» и «Категория^»;
- определить файлы, содержащие параметры мандатного управления доступом учётных записей пользователей (для этого выполнить поиск файлов в каталоге */etc* с именем равным идентификатору пользователя);
- самостоятельно выполнить смену максимального мандатного уровня доступа созданной учётной записи пользователя на «Уровень_4» с использованием редактирования содержимого найденного файла;
- проверить результат изменения с использованием графической утилиты «Политика безопасности»;
- проверить возможность входа в ОССН в графическом режиме с созданной учётной записью пользователя (уровень доступа — 4, неиерархические категории — нет, уровень целостности — «Низкий»);
- определить файл, используемый для хранения параметров максимального мандатного уровня целостности созданной учётной записи пользователя;
- в конце лабораторной работы вернуть ОССН к начальным настройкам (выполнить удаление созданных мандатных уровней доступа и т. п.).

Содержание отчёта по выполненной работе

В отчёте о выполненной работе необходимо указать:

- Полный перечень использованных команд с кратким описанием их назначения.
- Примеры выполнения команд, которые были использованы в ходе работы с описанием результатов их выполнения.
- Описание порядка работы с графической утилитой «Политика безопасности» при выполнении следующих действий:

- создание, изменение и удаление уровней доступа и неиерархических категорий;
- настройка уровней доступа и неиерархических категорий учётных записей пользователей.

- Описание порядка работы и команд, использованных при осуществлении следующих действий:

- настройка уровней доступа и неиерархических категорий учётных записей пользователей;
- настройка уровней доступа и неиерархических категорий в ОССН.

- Описание особенностей функционирования команд при работе на «Высоком» и «Низком» уровнях целостности.
- Список и назначение системных файлов, связанных с хранением параметров мандатного управления доступом и мандатного контроля целостности ОССН и пользователей.

Контрольные вопросы

- Как из *fly-term* запустить графическую утилиту «Политика безопасности»?
- Как с использованием графического интерфейса создать мандатный уровень доступа ОССН?
- Как определить текущие мандатные уровни доступа и целостности сессии пользователя?
- В чем заключаются отличия работы приложений с различным мандатным уровнем целостности для пользователя администратора ОССН без и с использованием команды *sudo?*
- Как выполнить смену максимального уровня доступа заданной учётной записи пользователя с использованием консольных команд?
- Как определить параметры мандатного управления доступом учётных записей пользователей?
- Какие файлы используются для хранения параметров мандатного управления доступом учётных записей пользователей?

4. 5. Лабораторная работа № 5.
Настройка механизмов организации замкнутой программной среды. Контроль целостности КСЗ

Цель работы

Изучить принципы и технологии контроля целостности данных (в том числе комплекса средств защиты — КСЗ), реализованных в ОССН. Освоить умения, необходимые для решения задач подсчёта носителей контрольных сумм файлов и оптических носителей, контроля соответствия дистрибутиву, регламентного контроля целостности и создания замкнутой программной среды.

Время выполнения работы 2 академических часа.

Краткие теоретические сведения

АСЗИ на базе ОССН должны обеспечивать функции как аудита доступа к сущностям файловой системы, так и контроля целостности (integrity) данных и содержимого исполняемых файлов. Подобный контроль позволяет с достаточной уверенностью констатировать факт отсутствия в данных, обрабатываемых системными процессами ОССН, недекларируемых для АСЗИ возможностей.

Для решения задачи контроля целостности в состав КСЗ ОССН включены средства, реализующие частные функции управления целостностью данных:

- вычисления и проверки контрольных сумм файлов и оптических дисков;
- контроля соответствия дистрибутиву;
- регламентного контроля целостности;
- создания замкнутой программной среды.

Базовым методом контроля целостности сущностей файловой системы ОССН является контроль их модификации путём вычисления контрольных сумм.

Контрольная сумма — значение, рассчитанное по набору данных путём применения определённого алгоритма и используемое для проверки целостности данных при их передаче или хранении. Она используется для быстрого сравнения двух наборов данных на эквивалентность: с очень большой вероятностью отличающиеся наборы данных будут иметь разные контрольные суммы.

Алгоритмы вычисления контрольной суммы, как правило, делятся на два вида:

- Алгоритмы общего назначения. К таким алгоритмам, в первую очередь, относится циклический избыточный код (Cyclic Redundancy Check, CRC), реализацией которого являются алгоритмы CRC8, CRC16, CRC32, применяющиеся для проверки целостности цифровых данных при их передаче по каналам связи.

- Криптографические алгоритмы. Эти алгоритмы основаны на процедуре хэширования — преобразования входного массива данных произвольной длины в выходную битовую строку фиксированной длины. К таким алгоритмам относятся, например, семейства алгоритмов MD {Message Digest Algorithm— MD2- MD6), SHA {Secure Hash Algorithm — SHA-1, SHA-2), ГОСТ Р 34. 11 (ГОСТ Р 34. 11-94, снятый с эксплуатации с 1 января 2013 г. , ГОСТ Р 34. 11-2012 «Стрибог») и другие. Областью применения этих алгоритмов является подтверждение целостности и подлинности передаваемых и хранимых данных.

В составе КСЗ ОССН включены следующие средства контроля целостности:

1. Команды, реализующие криптографические алгоритмы:
   - md5sum (реализация алгоритма МОБ);
   - shasum (реализация семейства алгоритмов SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/256 и SHA-512/224).
2. Средства проверки соответствия файловых сущностей ОССН её дистрибутиву:
   - команда gostsum (реализация алгоритма ГОСТ Р 34. 11-94, ГОСТ Р 34. 11-2012 256 и 512 битов);
   - графическая утилита fly-admin-int-check.

Указанные команды и утилиты реализуют статический контроль целостности файловых сущностей ОССН, включающий следующие компоненты:

1.Система мониторинга целостности файлов (FIM — File integrity monitoring) AFICK (Another File Integrity ChecKer), реализующая регламентный (периодический) контроль целостности файловых сущностей ОССН — вариант динамического контроля целостности.

2.Механизм контроля целостности исполняемых файлов и разделяемых библиотек формата ELF при запуске приложений на выполнение. Он реализован в выгружаемом модуле ядра ОССН dig- sig. verif и обеспечивает:

- контроль целостности исполняемых файлов и разделяемых библиотек на основе их контрольных сумм, вычисляемых в соответствии с ГОСТ Р 34. 11-94, ГОСТ Р 34. 11-2012 и электронной подписи, реализованной в соответствии с ГОСТ Р 34. 10-2001 и ГОСТ Р 34. 10-2012. Контрольная сумма и электронная подпись внедрены в файлы формата ELF в процессе сборки ОССН;

внедрение электронной подписи в исполняемые файлы формата ELF, входящие в состав устанавливаемого ПО.

Команды и утилиты статического контроля целостности функционируют в режимах вычисления (compute) и проверки (check) контрольных сумм файловых сущностей ОССН.

Например, команда shasum в режиме вычисления контрольной суммы файловой сущности с именем /root/file с использованием алгоритма SHA-256 имеет следующий синтаксис:

shasum -a 256 /root/file

В режиме проверки контрольных сумм файловых сущностей команды статического контроля целостности вычисляют их для сущностей, полный путь которых указан в текстовом файле с эталонными контрольными суммами, и сравнивают их с эталонными контрольными суммами из этого файла. Результатом их выполнения в режиме проверки контрольных сумм является передача на стандартный вывод строки формата (в случае совпадения контрольных сумм):

полный_путь_к_файловой_сущности: ОК или (в случае их несовпадения):

полный_путъ_к_файловой_сущности: FAILED

Например, команда shasum в режиме проверки контрольных сумм файловых сущностей в каталоге /root/dir1, с использованием алгоритма SHA-256 и при наличии текстового файла с эталонными

контрольными суммами /root/dir1. sha, имеет следующий синтаксис:

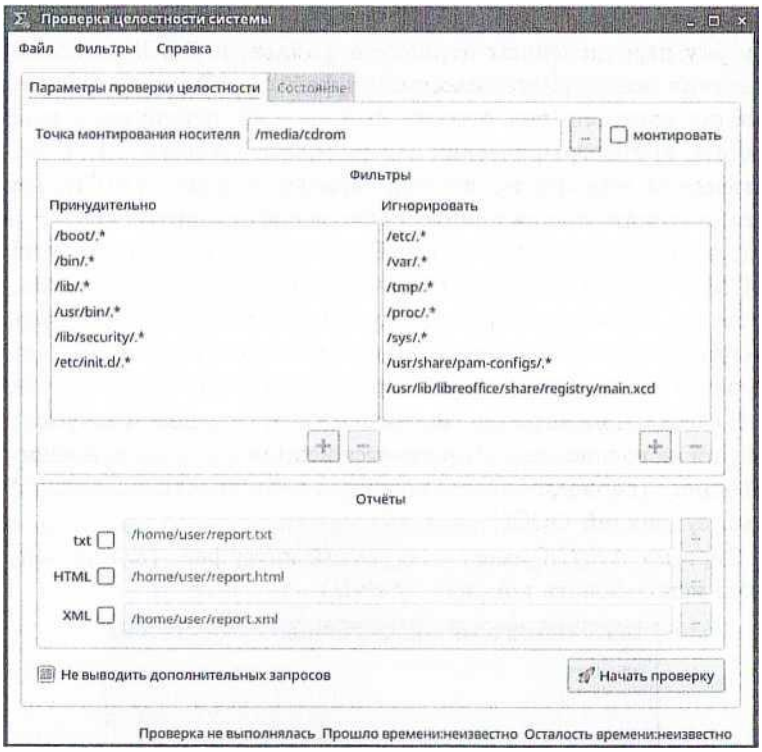
shasum -a 256 —c /root/dirl. sha

Для вычисления контрольных сумм файловых сущностей ОССН с использованием криптографического алгоритма ГОСТ Р 34. 11-2012 256 битов применяется команда gostsum, имеющая следующий синтаксис:

gostsum полный_путъ_к_файловой_сущности

- о полный_путь_к_файловой_сущности_с_контролъными_ суммами

Для проверки соответствия модулей установленной ОССН модулям, входящим в состав её дистрибутива, используется графическая утилита fly-admin-int-check, имеющая следующий интерфейс:



Для выполнения этой проверки в состав дистрибутива ОССН входит файл gostsums. txt, созданный командой gostsum и содержащий список контрольных сумм всех файлов, входящих в пакеты программ дистрибутива. Получаемый в результате проверки отчёт сохраняется в форматах *. txt, *. htm и *. xml.

Проверка соответствия модулей установленной ОССН модулям, входящим в состав её дистрибутива, является вариантом статического контроля целостности и обеспечивает контроль целостности файловых сущностей, копируемых в корневой раздел ОССН на этапе установки, что позволяет убедиться в отсутствии изменений в файловых сущностях модулей, произошедших на этапе их эксплуатации.

Однако такая проверка является неэффективной для файловых сущностей, содержимое которых многократно изменяется в ходе эксплуатации ОССН, например конфигурационных файлов. Кроме того, контроль целостности на основе только контрольной суммы файла не затрагивает проверку таких атрибутов файла, как временные метки (timestamps), дискреционные атрибуты (Minimal ACL и EA ACL), мандатные метки безопасности. Для выполнения расширенного контроля целостности файлов, обеспечивающего проверку перечисленных атрибутов файлов, в ОССН используется система мониторинга целостности файлов AFICK, реализующая функции контроля целостности файлов и их атрибутов с использованием криптографических алгоритмов MD5 и SHA-1. В ОССН применяется модифицированный вариант системы AFICK. дополнительно реализующий криптографический алгоритм ГОСТ Р 34. 11 (для приложения gostsum дополнительно поддерживаются алгоритмы ГОСТ Р 34. 11-94 и ГОСТ Р 34. 11-2012 с длиной ключа 256 или 512 битов), а также контроль мандатных меток и атрибутов подсистемы аудита безопасности! Дополнительно система AFICK имеет возможность настройки правил проверки целостности каталогов.

Благодаря интеграции системы AFICK с сервисом запуска приложений по расписанию cron имеется возможность выполнения регламентного (периодического) контроля целостности заданных файловых сущностей ОССН.

Система AFICK имеет следующий интерфейс (для её запуска можно использовать команду afick-tk):

Конфигурационным файлом системы AFICK является /etc/ afick. conf — текстовый файл, структурированный по секциям.

Секция alias содержит перечень действий (action) контроля целостности, из которых формируются правила контроля каталогов и файловых сущностей:

```
###############
# alias section
###############
it action : a list of item to check :
# md5 : md5 checksum
# shat : shal checksum
# cl : device
# i : inode
# p :    permissions
# n : number of        links
# u : user
# g : group
# s : size
It b : number of blocks # m : intime # c : ctime # a : atime # e : parsec mac # t : parsec aud # post : qost
```

Типовыми действиями является проверка:

- контрольных сумм, полученных заданным криптографическим алгоритмом (md5, sha1);
- inode (i) каталога или файловой сущности, её размера (size) и временных меток (mtime, ctime, atime);
- UID и GID (user, group) каталога или файловой сущности, а также прав доступа к ней (permissions).

Применительно к организации подсистемы безопасности ОССН дополнительно определены три действия:

- e: parsec mac — контроль целостности мандатных меток безопасности файловых сущностей;
- t: parsec aud — контроль целостности данных системы аудита безопасности ОССН;
- gost:gost — контроль целостности файловых сущностей с использованием криптографического алгоритма ГОСТ Р 34. 11.

В результате этих действий в секции alias формируются типовые правила для каталогов (DIR), файлов конфигурации ОССН (ETC) и файлов журналов системы аудита (Logs).

Например, правило для каталогов вида

DIR = p + i + n + u + g

указывает на необходимость выполнения проверки прав доступа, метаданных, количества ссылок и других стандартных атрибутов.

Дополнительно секция action включает правила, специфичные для подсистемы безопасности PARSEC — PARSEConly, PARCEC и GOST. Например, правило PARSEC вида PARSEC = p + d + i + n + u + g + s + b + md5 + m + e + t указывает на необходимость выполнения проверки стандартных атрибутов файловых сущностей с использованием криптографического алгоритма MD5, проверки расширенных атрибутов (меток безопасности и флагов аудита) и списков ACL этих файловых сущностей.

В правиле GOST вида GOST = p + d + i + n + u + g + s + b + gost + m + e + t параметр gost указывает на необходимость выполнения проверки стандартных атрибутов файловых сущностей с использованием криптографического алгоритма ГОСТ Р 34. 11.

В секции files to scan задаются полные пути и правила, применяемые к каталогам и файловым сущностям, для которых выполняется регламентный контроль целостности. Формат записей секции files to scan следующий:

- file action — проверяются каталоги, подкаталоги и файловые сущности с параметром «действия»;
- file — из проверки каталогов и подкаталогов исключается файловая сущность file;
- = directory action — с параметром «действия» проверяется только каталог, и из проверки исключаются подкаталоги.

Например:

- /boot GOST — проверка в каталоге /boot всех подкаталогов и файловых сущностей с помощью правила GOST;
- = /DIR — проверка с помощью правила DIR только корневого каталога, исключая подкаталоги;
- !/root/. bash_history — исключение проверки в каталоге /root файловой сущности . bash-history.

Эталонные значения контрольных сумм и атрибутов файловых сущностей и каталогов хранятся в базе данных системы AFICK в файле с расширением ndbm. Эта база данных создаётся в соответствии с параметрами секции «files to scan» файла / etc/ afick. conf.

Результаты контроля целостности оформляются в виде log-файлов и сохраняются:

- в случае принудительного (инициированного администратором) контроля — в каталоге /var/lib/afick/ archive в log-файлах с форматом имени afick.

YYYYMMDDHHMMSS. В аналогичных log-файлах сохраняются результаты обновления (update) базы данных системы AFICK;

- в случае регламентного (периодического, инициированного сервисом стоп) контроля — в каталоге / var/ log/ afick в log-файлах с форматом имени afick. log. N (где N принимает значения от 1 до 7).

Средство создания замкнутой программной среды в ОССН — невыгружаемый модуль ядра ОССН digsig_verif функционирует в трёх режимах (аналогично применяются режимы проверки подписи в расширенных атрибутах, т. е. не только для ELF-файлов, с использованием параметра DIGSIG_XATTR_MODE):

- штатный режим — исполняемым файловым сущностям формата ELF и разделяемым библиотекам, не имеющим ЭП или имеющим некорректную ЭП, исполнение запрещается (DIGSIG_ELF_MODE = 1);
- режим проверки ЭП в комплексе средств системного ПО — исполняемым файловым сущностям формата ELF и разделяемым библиотекам, не имеющим ЭП или имеющим некорректную ЭП, исполнение разрешается, но при этом выводится сообщение об ошибке проверки ЭП (DIGSIG_ELF_MODE = 2);
- отладочный режим для тестирования комплекса средств системного ПО (установлен по умолчанию) — ЭП исполняемых файловых сущностей формата ELF и разделяемых библиотек не проверяется (DIGSIG_ELF_MODE — 0).

Для выбора одного из указанных выше режимов функционирования модуля digsig_verif необходимо отредактировать конфигурационный файл / etc/ digsig/ digsig_initramfs. conf.
Управление модулем digsig_verif осуществляется через графический интерфейс fly-admin-smc либо через интерфейс файловой системы sysfs с использованием следующих файлов:

- / sys/ digsig/ enforce — в данном файле задаются указанные выше режимы работы;
- /sys/digsig/key — файл загрузки мастер-ключа ЭП;
- /sys/digsig/additional — файл загрузки дополнительных ключей ЭП.

Каждый дополнительный ключ для подписи системного ПО должен быть помещён в каталог /etc/digsig/keys.

Создание дополнительных ключей выполняется с помощью команды gpg (GNU Privacy Guard), модифицированной для использования криптографических алгоритмов ГОСТ Р 34. 11-94 и ГОСТ Р 34. 11-2012.

При администрировании средств контроля целостности данных и средств контроля соответствия дистрибутиву, а также при работе со средствами создания замкнутой программной среды используются следующие команды:

- afick — команда управления параметрами системы контроля целостности файловых сущностей;
- bsign — команда создания и проверки ЭП в файлах формата ELF;
- digsig_initramfs — команда загрузки ключей ЭП и инициализация режима Enforce модуля digsig_verif;
- fly-admin-int-check — графическая утилита администрирования контроля целостности файловых сущностей;
- gpg — команда работы с сертификатами пользователей;
- lsmod — команда получения списка загруженных модулей ядра;
- modinfo — команда получения информации о заданном модуле ядра;
- md5sum, gostsum, shasum — команда вычисления контрольных сумм;
- update-initramfs — команда инициализации начального загрузочного образа ОССН (initrd).

Используемое методическое и лабораторное обеспечение

1. ОССН версии 1. 6, в которой создана ученная запись пользователя user, с параметрами: максимальный и минимальный уровни доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий», входит в группу администраторов — astra-admin (вторичная группа), разрешено выполнение привилегированных команд (sudo).
2. Дистрибутив ОССН.
3. Документация: «Операционная система специального назначения «Astra Linux Special Edition». Руководство администратора. Часть 1», «Операционная система специального назначения «Astra Linux Special Edition». Руководство по КСЗ. Часть 1».
4. Для выполнения этапа работы по п. 43 необходимо наличие дополнительных ключей ЭП (ключи должны быть получены у разработчика и подписаны мастер-ключом «JSC RPA RusBITech (PRIMARY RBT ROOT KEY 2018)»).

Порядок выполнения работы

1. Начать работу со входа в ОССН в графическом режиме с учётной записью пользователя user (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий») и запустить терминал Fly в «привилегированном» режиме командой sudo fly-term.
2. В домашнем каталоге создать подкаталог checksum и скопировать в него все файлы (включая вложенные каталоги) из каталога /etc.
3. Используя алгоритм MD5, вычислить контрольные суммы всех файлов в каталоге /home/user/ checksum и перенаправить результат их вычисления в файл / home/

user / md5 check, а поток с перечнем ошибок в файл /home/user/error. md5, командой md5sum /home/user/checksum/ * > /home/user/md5check 2 > /home/user/ error. md5:

root@astra: /home/user# cat md5check

470c4dc7cb89f4f59f3d40af24438f4c d41d8cd98f00b204e9800998ecf8427e

6daf827d6d70c8e2be08b81338b8586b

/home/user/checksum/adduser. conf /home/user/checksum/adjtime/home/user/checksum/afick. conf

61e28705bd00f4410e182eeeb5469bel 00ff43422e8756204113c5546b00d529

21767f2203d324c988256b07f2634708 37d875e620170c9ed2ff3b0884b23b73

87b895cef45b8090d628a1d9a0f4bfb8 a81b3f1cb197219b815942f4fc7fa94e

4c09213317e4e3dd3c71d74404e503c5

/home/user/checksum/aliases

/home/user/checksum/anacrontab

/home/user/checksum/astra-safepolicy. conf

/home/user/checksum/astra_vers ion /home/user/checksum/bash. bashrc

/home/user/checksum/bash_completion

/home/user/checksum/bindresvport. blacklist

4. Вывести в терминал содержимое файлов /home/ user/md5 check и /home/user/error. md5 цепочкой команд cat /home/user/ md5check; cat /home/user/error. md5 и указать, для каких объектов в каталоге / home/user/ checksum контрольные суммы не были созданы.
5. Используя алгоритм SHA-512/256, вычислить контрольные суммы всех файлов в каталоге /home/ user/ checksum и перенаправить результат вычислений в файл /home/user/sha512256check командой shasum -a 512256/home/user/checksum/ * > /home/user/ sha512256check. Вывести на экран содержимое файла /home/user/ sha512256check командой less / home/ user/ sha512256check.
6. Используя редактор vim, изменить содержимое файла /home /user/checksum/passwd, удалив из него учётную запись суперпользователя (строку root: x: 0: 0: root: /root: /bin/bash).
7. Используя алгоритм MD5, проверить контрольные суммы всех файлов в каталоге /home/user/ checksum и перенаправить результат проверки в файл /home/user/fullcheck командой md5sum -c . /md5check > / home/user/fullcheck.
8. Используя алгоритм SHA512/256, проверить контрольные суммы всех файлов в каталоге /home/user/checksum и перенаправить результат проверки (с добавлением) в файл /home/user/full-check командой shasum -a 512256 -c . /sha512256check >> /home/ user/fullcheck.

9. Найти в файле /home/user/fullcheck строки, указывающие на файлы с нарушением целостности (содержащие слова ПОВРЕЖДЁН и FAILED), вывести в терминал их содержимое и число цепочкой команд grep 'ПОВРЕЖДЁН ' /home/user/fullcheck > /home/ user/tmpcheck; grep ‘FAILED’ /home/user/fullcheck » /home/ user/tmpcheck; wc -l /home/user/tmpcheck; less / home/ user/ tmp- check.
10. Используя алгоритм ГОСТ Р 34. 11-2012 (256 битов), вычислить контрольную сумму файла /home/user/checksum/shadow, перенаправить результат проверки в файл / home/user/gostcheck и вывести в терминал содержимое файла / home/ user/gostcheck цепочкой команд gostsum /home/user/checksum/shadow -o . /gostcheck; less /home/user/gostcheck.
11. Установить оптический диск с дистрибутивом ОССН и, используя алгоритм ГОСТ Р 34. 11-2012 (256 битов), вычислить его контрольную сумму (по умолчанию файл устройства оптического диска /dev/srO) и перенаправить результат вычисления в файл /home/user/isocheck командой gostsum -d /dev/sr0 > /home/user/ isocheck (выполнение команды занимает длительное время).
12. Запустить графическую утилиту fly-admin-int-check и во вкладке «Параметры проверки целостности»:
    - выбрать точку монтирования устройства «Astra Smolensk amd64» (по умолчанию это, чаще всего, каталоги /media/cdrom или /media/ cdrom0) и выполнить монтирование;
    - настроить фильтр проверки целостности в разделе «Принудительно», добавив регулярное выражение, содержащее абсолютный путь ко всем файлам каталога /usr/lib: /usr/lib/*;
    - настроить фильтр проверки целостности в разделе «Игнорировать», удалив регулярное выражение, содержащее абсолютный путь к каталогу /tmp;
    - в разделе « Отчёты» задать только текстовый формат файла отчёта, определив путь размещения файла report. txt в каталоге / home/user/report;
    - изменить содержимое файла / usr/ share/ doc/ libcap2/ copyright командой vim /usr/ share/ doc/libcap2/ copyright, удалив в нем две первые строки:

Upstream-Contact: Andrew G. Morgan <morgan@kernel. org> Source: https: //www. kernel. org/pub/linux/libs/security/linux-privs/libcap2/: d2

    - начать проверку и зафиксировать предполагаемое время проверки, перейти во вкладку «Состояние» и проконтролировать статус проверки, после окончания проверки завершить работу графической утилиты;
    - в файле /home/user/report. txt найти строки, содержащие текст: «Файлы, целостность которых нарушена», «Контр. сумма» и

«/usr/share/doc/libcap2/copyright», сохранить результаты поиска в файл /home/user/report-2 цепочкой команд: grep ‘Файлы, целостность которых нарушена’ /home/user/report. txt -A 4 >/home/user/report-2-, grep ’Контр. сумма’ /home/user/report. txt -A 4 >> /home/user/report-2; grep '/usr/share/doc/libcap2 /copyright’ /home/user/report. txt » /home/user/report-2.

13. Отредактировать секцию directives конфигурационного файла / etc/ afick. conf системы AFICK, отменив проверку выполняющихся приложений: исходный вариант секции directives: running_ files : = yes, отредактированный вариант секции directives: runningfiles : = 0.
14. Отредактировать секцию alias конфигурационного файла / etc/ afick. conf системы AFICK:
    - изменить правило ETC, удалив из него проверку размера файловых сущностей и добавив проверку времени их модификации: исходный вариант правила: ETC = p + d + i + и + g + s + md5, отредактированный вариант правила: ETC = p + d + i + и + д + т + md5;
    - отредактировать правило MyRule, удалив из него проверку для файловых сущностей количества ссылок на них и добавив проверку контроля целостности мандатных меток безопасности, контроля целостности данных системы аудита безопасности и контроля целостности с использованием криптографического алгоритма ГОСТ Р 34. 11-2012 вместо алгоритма MD5: исходный вариант правила: MyRule = p+d+i+n+u+g+s+b+md5+m, отредактированный вариант правила: MyRule = p + d + i + u + g + s + b + gost + т + e + t.
15. Отредактировать секцию file section конфигурационного файла / etc/ afick. conf.
    - заменить для каталога /boot правило проверки GOST на правило проверки PARSEC: исходный вариант: /boot GOST, отредактированный вариант: /boot PARSEC-,
    - добавить для файловой сущности / etc/fstab правило проверки MyRule: отредактированный вариант: /etc/fstab MyRule-,
    - активировать правило проверки по умолчанию для каталога

/lib: исходный вариант: #/lib MyRule, отредактированный вариант: /lib MyRule.

16. Обновить базу данных системы AFICK с учётом выполненных изменений в секции file section командой afick -и.
17. Изменить содержимое файла /etc/fstab, удалив в нем две первые строки.
18. Запустить графическую утилиту «Контроль целостности файлов» (afick-tk) управления AFICK из меню «Системные» главного пользовательского меню и

выполнить принудительную проверку целостности, выбрав действие — сравнение с базой.

19. После завершения контроля целостности:
    - в меню утилиты afick-tk «Файл — история» определить дату и время последнего принудительного контроля целостности;
    - найти в каталоге /var/lib/afick/archive log-файл, соответствующий выполненной принудительной проверке (значение YYYYMMDDHHMMSS в имени log-файла должно совпадать с найденными в предыдущем пункте датой и временем проверки);
    - просмотреть найденный log-файл с помощью команды less и в его секции #detaled changes найти запись о нарушении целостности файловой сущности /etc/fstab (раздел changed file : /etc/fstab);
    - проанализировать найденную запись о нарушении целостности и определить параметры, соответствующие действиям (action) нарушения целостности, и их текущие значения.
20. Запустить терминал Fly в «привилегированном» режиме командой sudo fly-term.
21. Просмотреть загруженные модули ядра ОССН и вывести в терминал данные о невыгружаемом модуле digsig_verif конвейером команд lsmod / grep «digsig_verif». Ответить на вопрос: связан ли модуль digsig_verif с другими загружаемыми (невыгружаемыми) модулями?

root@astra: /etc/digsig# lsmod | grep "digsig_verif“ digsig_verif 491520 0

- Просмотреть информацию о модуле digsig_verif командой modinfo digsig_verif. Определить расположение модуля digsig_verif и информацию о разработчике.
- Выполнить импорт открытых ключей, используемых для проверки ЭП файлов. Для этого выполнить следующие действия:
    - инициализировать каталог / root/ . gnupg при просмотре текущих ключей командой gpg --list-sigs;
    - импортировать открытый мастер-ключ «JSC RPA RusBITech (PRIMARY RBT ROOT KEY 2018)» командой gpg —import / etc/ digsig/primary_key-2018. gpg;
    - импортировать открытые ключи partners_rbt_root_key_2018. gpg и build_system_rbt_root_key_2018. gpg (данный ключи используется для подписи файлов ОССН), командой gpg --import /etc/ digsig/имя_файла_ключа:

root@astra: /etc/digsig# gpg --list-keys
/root/. gnupg/pubring. kbx
---

pub gP256 2018-06-17 [SC]
8066E98D2201D9783E2D842BD2B6689A37DB8024
uid [неизвестно] JSC RPA RusBITech (BUILD-SYSTEM RBT ROOT KEY 2018) <mail@rusbitech. ru>

pub gP256 2018-06-17 [SC]
A12D7B5BAFCE40D87FD41DAFC82D49FC3675B6FA
uid [неизвестно] JSC RPA RusBITech (PARTNERS RBT ROOT KEY 2018) <mail@rusbitech. ru>

pub gP256 2018-06-17 [SC]
8E839E2F389F882A259F47997285E858D8069FF5
uid [неизвестно J JSC RPA RusBITech (PRIMARY RBT ROOT KEY 2018) <mail@rusbitech. ru>

- Вывести текущие ключи командой gpg --list-sigs. Определить идентификатор мастер-ключа «JSC RPA RusBITech (PRIMARY RBT ROOT KEY 2018)». Используется ли он для подписи других загруженных ранее ключей?

root@astra: /etc/digsig#gpg --list-sigs
/root/. gnupg/pubring. Kbx
---

pub gP256 2018-06-17 [SC]
8066E9BD2201D9783E2D842BD2B6689A37DB8024
uid [неизвестно] JSC RPR RusBITech (BUILD-SYSTEM RBT POCT KEY 2018) <mail@rusbitech. ru>
sig 3 D2B6689R37DB8024 2018-06-17 JSC RPR RusBITech (BUILD-SYSTEM RBT ROOT KEY 2018) <mail@rusbltech. ru>
sig 7285E858DB069FF5 2018-06-17 JSC RPR RusBITech (PRIMRRY RBT ROOT KEY 2018) <mail@rusbltech. ru>

pub gP256 2018-06-17 [SC]
R12D785BRFCE40D87FM1DRFC82D49FC3675B6FA
uid [неизвестно] JSC RPR RusBITech (PARTNERS RBT ROOT KEY 2018) <mail@rusbltech. ru>
sig 3 C82D49FC3675B6FA 2018-06-17 JSC RPR RusBITech (PARTNERS RBT ROOT KEY 2018) <mail@rusbitech. ru>

sig 7285EB58DB069FF5 2018-06-17 JSC RPR RusBITech (PRIMARY RBT ROOT KEY 2018) <mail@rusbltech. ru>

pub gP256 2018-06-17 [SC]BE839E2F389F882R259F47997285E858DB069FF5
uid [неизвестно] JSC RPR RusBITech (PRIMRRY RBT ROOT KEY 2018) <mail@rusbitech. ru>
sig 3 7285E858DB069FF5 2018-06-17 JSC RPR RusBITech (PRIMRRY RBT ROOT KEY 2018) <mail@rusbitech. ru>

- Проверить корректность ЭП файла /Ыn/dash командой bsign -w $(which dash'). Определить, каким ключом был подписан данный файл по его идентификатору в строке «signer: ».
- Переписать открытый ключ /etc/ digsig/build_system_rbt- root-key_2018. gpg в каталог /etc/ dig sig/keys командой cp /etc/digsig/buildsystem-rbt_root_key_2018. gpg /etc/digsig/keys.
- Перейти в каталог /etc/digsig и изменить файл digsig_initramfs. conf (значение DIGSIG_ELF_MODE установить равным 1).
- Проверить корректность установки данного параметра путём открытия настройки «Замкнутой программной среды» в «Панели управления».
- Проверить корректность ключа путём его загрузки в модуль digsig_verif командой digsig_initramfs (для поиска этой команды можно использовать команду find).
- Создать дополнительный ключ ЭП командой gpg --fullgenerate-key. В диалоге команды gpg:
    - выбрать пункт 15 «GOST R 34. 10-2012», указать неограниченный срок действия дополнительного ключа ЭП, выбрав значение 0;
    - указать параметры Real Name: rootserver, Email: root@server. test и получить User ID: «rootserver <root@server. test>».
- Вывести текущие ключи командой gpg —list-sigs и определить идентификатор ключа «rootserver <root@server. test>».
- Скопировать файл /bin/dash в каталог /root, указав при этом новое имя файла 1. elf.
- Подписать файл l. elf новым ключом «rootserver <root@ser- ver. test>» командой bsign --sign /root/l. elf.
- Вывести новую подпись файла командой bsign -w / root/1. elf и проверить соответствие идентификатора ключа ЭП в строке «signer. » данным ключа «rootserver <root@server. test>».

- Включить штатный режим проверки ЭП с использованием модуля digsig_verif, установив значение ключа DIGSIG_ELF_MODE = 1 в конфигурационном файле /etc/digsig/digsig_initramfs. conf.
- Активировать настройки командой sudo update-initramfs -и -к all, затем выполнить перезагрузку и повторный вход в ОССН.
- Запустить терминал Fly в «привилегированном» режиме командой sudo fly-term.
- Проверить включение штатного режим функционирования модуля digsig_verif (в файле /sys/ digsig/ elf_mode должно быть установлено значение «1») командой cat /sys/digsig/elf'_mode.
- Выполнить попытку запуска файла /root/1. elf, который был подписан с использованием ключа «rootserver <root@server. test>» (данный ключ не был подписан мастер-ключом «JSC RPA RusBITech (PRIMARY RBT ROOT KEY 2018)»), и проанализировать выводимые ошибки.
- Установить значение ключа DIGSIG_ELF_MODE=0 в конфигурационном файле / etc/ digsig/ digsig_initramfs. conf, активировать настройки командой update-initramfs -и -к all, затем выполнить перезагрузку и повторный вход в ОССН.
- В «привилегированном» режиме терминала Fly выполнить команду /root/1. elf и проанализировать выводимые ошибки.
- Выйти из запущенного интерпретатора «dash» (файл 1. elf) командой exit. Активировать настройки командой update-initramfs -и -к all, затем выполнить перезагрузку и повторный вход в ОССН.
- При наличии дополнительных ключей ЭП (ключи должны быть получены у разработчика и подписаны мастер-ключом «JSC RPA RusBITech (PRIMARY RBT ROOT KEY 2018)») выполнить следующие действия (далее имена файлов ключей sign_public. gpg — открытый ключ, sign_secret. gpg — закрытый ключ):
    - в«привилегированном» режиме терминала Fly выполнить очистку ключей в папке /root/. gnupg командой rm -r /root/. gnupg;
    - импортировать ключи командами gpg--import sign_public. gpg; gpg --import signsecret. gpg;
    - импортировать открытый мастер-ключ командой gpg —import / etc/ digsig/ primary _key_2018. gpg;
    - проверить импорт ключей командой gpg --list-sigs, при этом должно отобразиться два ключа;
    - добавить действующий ключ ЭП в модуль digsig_verif командой cat / etc/ digsig/build_system_rbt_root_key_2018. gpg > / sys/ digsig /keys;

- добавить новый ключ ЭП в модуль digsig_verif командой cat sign_public. gpg > /sys/digsig/keys;
- выполнить команду echo "1" > / sys/ digsig/elf_mode активации режима проверки ЭП файлов;
- скопировать файл /root/1. elf в новый файл командой cp /root/1. elf /root/1 signed. elf;
- выполнить команду /root/1. elf и проанализировать выводимые ошибки (данный файл не должен запускаться по причине использования ЭП ключом «rootserver <root@server. test>»);
- подписать файл 1 signed. elf командой bsign --sign /root/1sign-ed. elf, затем проверить ЭП командой bsign—w /root/1 signed. elf;
- запустить подписанный файл командой /root/1 signed. elf и проверить отсутствие ошибок;
- выполнить перезагрузку и повторный вход в ОССН, запустить терминал Fly в «привилегированном» режиме командой sudo fly-term;
- проверить текущий режим модуля digsig_verif командой cat /sys/digsig/elf_mode (режим должен быть равен 0);
- перейти в каталог /root, осуществить запуск файлов 1signed. elf и 1. elf, затем проанализировать полученные результаты и выводимые ошибки;
- добавить ключи ЭП в модуль digsig_verif командой cat /etc/ digsig/ build_system_rbt. root-. key_2018. gpg > /sys/digsig/keys и cat sign_public. gpg > /sys/ digsig/ keys, осуществить повторный запуск файлов 1 signed. elf, 1. elf; а затем проанализировать полученные результаты.

- Создать ключи и выполнить подпись файла конфигурации:
  - запустить терминал Fly от имени учётной записи пользователя user командой fly-term;
  - скопировать файлы / etc/passwd и / bin/ dash в каталог ~ и сменить владельца на user: user;
  - выполнить команду генерации мастер-ключа для подписи в xattr командой gpg --full-generate-key, выбрать алгоритм (15) и установить имя: xattr-key;
  - выполнить команду генерации ключа для подписи в xattr командой gpg --full-generate-key, выбрать алгоритм (15) и установить имя: xattr-key-sign;
  - выполнить подпись ключа «xattr-key-sign» командой gpg --sign-key "xattr-key-sign" > xattr-key-sign. gpg;
  - экспортировать ключ «xattr-key» командой gpg--export"xattr- key" xattr-key. gpg;

- проверить наличие подписанного ключа «xattr-key-sign» командой gpg --list-sigs (при этом ключ «xattr-key-sign» должен быть подписан ключом «xattr-key»);
- запомнить идентификаторы ключей «xattr-key» и «xattr-key- sign» (8 байтов в шестнадцатеричном формате — 16 символов);
- создать хэш файла ~/passwd и записать его в расширенные атрибуты командой bsign - -hash ~/passwd (обратить внимание, что никаких ключей разблокировки секретного ключа при этом не запрашивается у пользователя);
- создать файл ~/. gnupg/gpg. conf с содержимым: default-key идентификатор _ключа_xattr-key-sign;
- выполнить подпись файла passwd командой bsign —sign ~ /passwd;
- выполнить проверку подписи файла passwd командой bsign -w ~ /passwd;
- скопировать ключи (xattr-key-sign. gpg и xattr-key. gpg) для работы с подписями файлов в каталог /etc/digsig/xattr_keys;
- в графическом файловом менеджере fly-fm перейти в каталог «Домашний» и открыть в контекстном меню «Свойства», «Подпись» файла passwd;
- нажать кнопки «Загрузить ключи» и «Информация», при этом проверить корректность созданного хэш и наличие подписи.

Содержание отчёта по выполненной работе

В отчёте по выполненной работе необходимо указать:

- Полный перечень использованных команд с описанием их назначения.
- Примеры выполнения команд, которые были использованы в ходе работы, с описанием результатов их выполнения.
- Описание порядка работы с графическим интерфейсом при выполнении следующих операций:
  - конфигурирование обязательных для проверки и игнорируемых путей в графической утилите fly-admin-int-check;
  - конфигурирование пути размещения файла отчёта в графической утилите fly-admin-int-check;
  - просмотр текущих правил проверки в графической утилите «Контроль целостности файлов» (afick-tk) системы регламентного контроля целостности AFICK',
  - запуск проверки целостности данных в графической утилите afick-tk.
- Описание порядка работы с командами при выполнении следующих операций:
  - вычисление контрольной суммы файла с использованием команды md5sum;
  - вычисление контрольной суммы файла с использованием команды shasum;

- вычисление контрольной суммы файла с использованием команды gostsum;
- проверка целостности файлов с использованием команды md5sum;
- проверка целостности файлов с различными алгоритмами вычисления контрольной суммы с использованием утилиты shasum.

• Описание особенностей конфигурирования и режимов функционирования модуля digsig_verif.

Контрольные вопросы

1. В чем заключается отличие команд md5sum, shasum и gostsum с точки зрения вычисления контрольной суммы файлов?
2. В каком формате организован вывод команд md5sum, shasum и gostsum при вычислении контрольной суммы файлов?
3. Какая из команд md5sum, shasum или gostsum выполняет только вычисление контрольной суммы файлов и не выполняет проверку их целостности?
4. В какой из команд md5sum, shasum или gostsum возможно изменение алгоритма хэширования?
5. Каково назначение файла gostsum. txt, и где этот файл располагается?
6. Какие псевдонимы (aliases) в конфигурационном файле системы регламентного контроля целостности AFICK соответствуют правилам проверки целостности мандатных меток безопасности файлов и контроля целостности данных системы аудита безопасности?
7. Какие правила в конфигурационном файле системы регламентного контроля целостности AFICK сформированы по умолчанию, а какие являются специфическими для подсистемы безопасности PARSEC?
8. В каких средствах контроля целостности используется библиотека libgost?
9. Как инициализировать базу данных системы регламентного контроля целостности AFICK после внесения изменений в её конфигурационный файл?
10. Каким образом реализована взаимосвязь системы регламентного контроля целостности AFICK и сервиса cron?
11. Каким видом модулей ядра ОССН является модуль digsig_verif?
12. Какой формат файлов ключей ЭП СПО использует модуль digsig. verif?
13. Какая специализированная файловая система применяется для хранения данных о состоянии и функционировании модуля digsig_verif?
14. Какой файл сценария командного интерпретатора bash применяется при добавлении дополнительных ключей ЭП для модуля digsig_verif?

4.6. Лабораторная работа № 6.
Настройка сетевого взаимодействия

Цель работы

Изучить принципы и порядок конфигурирования сетевого взаимодействия узлов АСЗИ на базе ОССН. Освоить навыки решения задач управления сетевыми интерфейсами и протоколами, а также маршрутизации пакетов.

Время выполнения работы 4 академических часа.

Краткие теоретические сведения

Взаимодействие узлов АСЗИ между собой и с сетевым оборудованием в сетях с пакетной коммутацией на базе стека протоколов TCP/IP основано на сетевых службах, реализуемых соответствующими процессами в ОССН. Такие процессы на серверных узлах АСЗИ создают программные интерфейсы (сокеты), связанные с требуемыми сетевыми службами сетевыми портами. Процессы ОССН на клиентских узлах АСЗИ формируют запрос на открытие соответствующего сокета с указанием IP-адреса и сетевого порта серверных узлов АСЗИ. Результатом выполнения такого запроса является установление соединения между серверными и клиентскими узлами ОССН в рамках заданной сетевой службы.

Особенностью этого соединения в ОССН является поддержка передачи по нему данных мандатного контекста сущностей сетевой службы и субъектов доступа к ней в соответствии с ГОСТ Р 58256-2018, при этом монитором обращений выступает подсистема безопасности *PARSEC.* В случаях, когда сетевой сервис не обрабатывает данные мандатного контекста, однако осуществляет соединение с процессами ОССН, работающими в различных мандатных контекстах, применяется механизм *privsock.*

Конфигурация сетевых интерфейсов, настройка адресной информации и статической маршрутизации пакетов реализуются на-борами команд *Net tools* (пакет *net-tools,* в том числе команда *mii-tool*) и *Iproute*2 (пакет *iproute*2), графической утилитой *fly-admin-device-manager.* Для статического конфигурирования параметров сетевых интерфейсов также используется конфигурационный файл/ *etc/network/interfaces,* а для управления запуском приложений перед инициализацией и закрытием сетевых интерфейсов — наборы сценариев, расположенные в каталогах: */etc/network/if-up.d, /etc/network/if-down.d, /etc/if-pre-up.* d и /*etc/if-post-down.d.*

В пакеты *iproute2, net-tools* входят следующие команды:

ip — команда для просмотра параметров и конфигурирования сетевых интерфейсов, сетевых адресов, таблиц маршрутизации, правил маршрутизации;

- *arp* — команда для просмотра таблиц, /Р-туннелей, адресов групповой рассылки;
- *tc*— (*traffic control*) команда для просмотра и конфигурирования параметров управления трафиком;

- *ss* — команда для просмотра текущих соединений и открытых сетевых портов.

Команда *ip* имеет следующий синтаксис:

ip имя_объекта название_команды

где:

- *имя_объекта* — логический объект сетевого соединения, над которым производятся действия (к таким логическим объектам относятся, например, сетевые интерфейсы, адреса, таблицы маршрутизации);
- *название_команды* — описатель специфического действия над объектом.

Например:

- *ip link show* — вывод текущего состояния (команда *show*) сетевых интерфейсов (объект *link);*
- *ip address show* — вывод (команда *show)* сопоставленных с сетевыми интерфейсами *IP*-адресов (объект *address*);
- *ip link set up eth2* — включение (команда *set up)* сетевого интерфейса, связанного с файлом устройства *eth2* (объект *link)]*в *sudo ip address add* 192.168.1.1/25brd -(- *dev ethO* —- добавление(команда *add) IP*-адреса 192.168.1.1/25 (объект *addr)* для сетевого интерфейса, связанного с файлом устройства *ethO* с автоматическим расчётом широковещательного (параметр *brd* +)адреса (выполнение требует привилегий супер пользователя).Команда *ss* имеет следующий синтаксис:*ss название^опции фильтр,* где:

о *название^опции* — элементы сбора статистики по сетевым соединениям *(TCP, UDP, ssh, ftp, http, https) фильтр* — указываются опциональные фильтры для выбора статистики.

Например:

- *ss -s* — вывод информации об открытых сетевых соединениях• *ss -l* — вывод информации о всех открытых портах;
- *ss — pi*— вывод информации о всех открытых портах с указанием приложения, открывшего порт;
- *ss -ta* — вывод информации о всех *TCP* соединениях;•*ss -ua* — вывод информации о всех *UDP* соединениях.
- Апплет *NetworkManager (/usr/Ып/пт-applet* «Сетевые соединения») отображает обнаруженные автоматически сетевые интерфейсы и элементы управления, с помощью которых имеется возможность:
- отключить/включить сетевой интерфейс;
- вызвать меню настроек *IP*-адреса сетевого интерфейса;
- задать этот сетевой интерфейс в сетевом профиле по умолчанию*(default)*

- выполнить настройку доступных сетевых интерфейсов. Команда *mii-tool* позволяет контролировать драйвер устройства, связанного с сетевым интерфейсом, отображать данные о нем, управлять параметрами драйвера и программной конфигурацией устройства.

В случае использования на узле АСЗИ под управлением ОССН двух и более сетевых интерфейсов (например, когда узел является маршрутизатором и шлюзом) требуется включение функции *«1P forwarding»,* которая позволяет перенаправлять JP-пакеты с одного сетевого интерфейса на другой. Включение функции *«IP forwarding»* выполняется путём вызова команды вида *sysctl -w net.ipv4-ip-forward* — 1.

Для того чтобы данный параметр работы ядра устанавливался при загрузке ОССН, необходимо раскомментировать строку *«net.ipv4-ip-forward* = 1» в конфигурационном файле */etc/sysctl.conf.*

Для использования механизма *privsock* при функционировании сетевого сервиса необходимо отредактировать файл */ etc/parsec/privsock. conf,* добавив в него строку, содержащую полный путь к исполняемому файлу сервиса. Например, для запуска DJ/5-сервера с использованием механизма *privsock* файл / *etc/parsec/privsock. conf*должен содержать следующую строку: /*usr/ sbin/ named.* Кроме того для использования механизма необходимо, чтобы переменная *PATH* при запуске данного сервиса, содержала путь /*usr/НЬ/parsec/bin.* Для *DNS*-сервера файл настроек / *etc/ default/ bind9* должен содержать строку *PATH—/usr/lib/parsec/*&m:$*PATH.*

При настройке сетевого взаимодействия используются следующие команды и графические утилиты (примеры применения которых также рассмотрены в главе 3):

- *route* — команда управления маршрутами; о *ifconfig* — команда управления параметрами сетевого интерфейса;
    - *ping* — команда отправки и получения пакетов *ICMP* (*EchoRequest/ Echo Reply)-,*
    - *whoami* — команда получения имени текущей учётной записи;
    - *ssh* — команда подключения к серверу *SSH*;• *if down* — команда отключения сетевого интерфейса;
    - *ifup* — команда включения сетевого интерфейса;
    - *service* — команда управления сервисами;
    - *fly-admin-wicd* — графическая утилита «Сетевые подключения» для управления параметрами сетевого интерфейса;
    - *fly-admin-device-manager* — графическая утилита «Менеджер устройств» для управления параметрами устройств, в том числе сетевых интерфейсов.

Используемое методическое и лабораторное обеспечение

1. Три компьютера (либо виртуальные машины) с ОССН версии 1.6, соединённые в сеть последовательно: *AstraClient1, Astra-Router, AstraClient2.* При этом компьютер с ОССН *AstraRouter* два сетевых интерфейса, соединённых с ОССН AstraClient1 и AstraClient2: AstraClient1 (ethJO) — {ethJO) AstraRouter (ethl) -(ethO) AstraClient2 .
2. В каждой ОССН создана учётная запись пользователя *user, с* параметрами: максимальный и минимальный уровни доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий», входит в группу администраторов — *astraadmin* (вторичная группа), разрешено выполнение привилегированных команд (*sudo).*
3. Дистрибутив ОССН.
4. Документация: «Операционная система специального назначения *«Astra Linux Special Edition».* Описание применения», «Операционная система специального назначения «*Astra Linux Special Edition».* Руководство администратора. Часть 1», «Операционная система специального назначения *«.Astra Linux Special Edition».*Руководство по КСЗ. Часть 1».
5. Для выполнения работы в течение двух занятий необходимо обеспечить возможность сохранения состояния ОССН за счёт применения технологий виртуализации (создания виртуальных машин ОССН).

Порядок выполнения работы

1. Начать работу с входа в ОССН *AstraClient1, AstraClient2* и *AstraRouter* в графическом режиме с учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий»),
2. В ОССН *AstraClientl, AstraClient2* и *AstraRouter* запустить терминал *Fly* в «привилегированном» режиме командой *sudofly-term,* и вывести информацию о доступных сетевых интерфейсах командой *ip link list.*
3. Настроить ОССН *AstraClient1.* Для этого выполнить следующую последовательность действий:
    - настроить параметры сетевого интерфейса *ethJO* командой *ipaddress add* 10.0.0.2/8 *dev ethl*3;
    - настроить маршрутизацию в подсеть 20.0.0.0/8 командой *iproute add* 20.0.0.0/8 *via* 10.0.0.1;
    - вывести таблицу маршрутизации командами *ip route* и *route,* убедиться в добавлении маршрута в подсеть 20.0.0.0/8;
    - проверить установку сервера *SSH* командой *apt-get installopenssh-server* (здесь и далее при выполнении команды *apt-get*потребуется установка *DVD* с дистрибутивом ОССН);

а выполнить запуск сервера командой *service ssh start.*

4. Выполнить настройки сетевого интерфейса ОССН *AstraClient1.* Для этого выполнить следующую последовательность действий:

- настроить параметры сетевого интерфейса *ethO* командой *if con-fig ethO* 20.0.0.2 *netmask* 255.0.0.0 *broadcast* 20.0.0.255;
- настроить маршрутизацию в подсеть 10.0.0.0/8 командой *routeadd -net* 10.0.0.0/8 *gw* 20.0.0.1;
- вывести таблицу маршрутизации командами *route -п* и *ip route,* убедиться в добавлении маршрута в подсеть 10.0.0.0/8.

5. В ОССН *AstraRouter* выполнить настройку статических сетевых адресов и маршрутизации. Для этого выполнить следующую последовательность действий:

- используя редактирование файла / *etc/ network/interfaces* командой *vim* / *etc/network/interfaces* (файл / *etc/ network/inter-faces* должен содержать следующие строки настройки: *«autoethO», «iface ethO inet static», «address* 10.0.0.1», *«netmask* 255.0.0.0», *«auto ethl», «iface ethl inet static», «address* 20.0.0.1», *«netmask* 255.0.0.0»), настроить адреса сетевых интерфейсов;
- выполнить перезагрузку и повторный вход в ОССН *AstraRouter* с учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий»);
- запустить терминал *Fly* и проверить доступность *AstraClient1, AstraClient2 с AstraRouter* по сети, выполнив для этого команды *ping* 20.0.0.2, *ping* 10.0.0.2;
- запустить терминал *Fly* в «привилегированном» режиме командой *sudo fly-term;*
- включить функцию *«IP forwarding»* командой *sysctl -w net.ipvf Ap-forward* — 1;
- выполнить команду *vim* /*etc/sysctl. conf* и раскомментировать параметр *net.ipv4-ip-forward* = 1 в конфигурационном файле*/ etc/ sysctl. conf;*
- установить пакет *ethtool* командой *apt-get install ethtool.*

6. Проверить функционирование маршрутизации между подсетями 10.0.0.0/8 и 20.0.0.0/8. Для этого выполнить следующую последовательность действий:

- в ОССН *AstraClient1* выполнить команду *ping* 20.0.0.2;
- в ОССН *AstraClient2* выполнить команду *ping* 10.0.0.2.

7. В ОССН *AstraClient1* проверить работоспособность сервера *SSH* путём проверки открытого им порта командой *ss -1p.* Определить номер порта, который прослушивает сервер *sshd.*

8. Работая в ОССН *AstraClient2,* выполнить подключение с учётной записью пользователя *user* к серверу *SSH,* установленному на ОССН *AstraClient1.* Для этого выполнить следующую последовательность действий:

- запустить терминал *Fly* в «привилегированном» режиме командой *sudo fly-term*;
- выполнить команду *ssh -l user* 10.0.0.2 и ввести пароль учётной записи пользователя *user* в ОССН *AstraClient1*;
- убедиться в работоспособности сетевого соединения с помощью удалённого доступа по протоколу *SSH,* выполнив идентификацию пользователя *user* командой *whoami* и просмотр содержимого его домашнего каталога командой *ls -al*\
- проверить наличие соединения ОССН *AstraClient1* с ОССН *AstraClient2* (IP-адрес 20.0.0.2) по порту *ssh* командой *ss -ta*\

завершить удалённую сессию командой *exit;*

- вывести информацию о МЛ С-адресах в ОССН *AstraClient2* с использованием команд *arp* и *cat /proc/net/arp,* а затем сравнить полученные результаты.

9. В ОССН *AstraRouter* выполнить команды управления сетевыми интерфейсами. Для этого реализовать следующую последовательность действий:

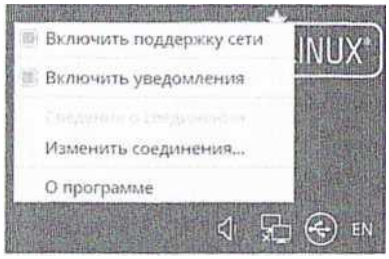
- запустить терминал *Fly* в «привилегированном» режиме командой *sudo fly-term*;
- определить скорости работы сетевых интерфейсов *ethO* и *ethl*командами *mii-tool ethO* и *mii-tool ethl*;
- снизить скорость сетевого интерфейса *ethO* до 100 Мбит/с (либо10 Мбит/с в зависимости от текущей скорости сетевого интерфейса) в режиме *Half Duplex* командой *mii-tool ethO -F 100-BaseTx-FD* и убедиться в применении изменений командой *mii-tool ethO.*

10. Выполнить настройки сетевых интерфейсов в ОССН *Astra-Client1* с использованием графической утилиты «Сетевые соединения». Для этого выполнить следующую последовательность действий:

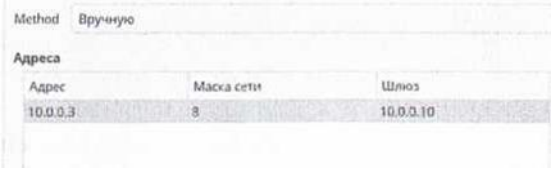
- выполнить перезагрузку и повторный вход в ОССН *AstraClient1* с учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий»);
- запустить терминал *Fly* в «привилегированном» режиме командой *sudo fly-term* и проверить отсутствие IP-адреса интерфейса *ethl* командой *ifconfig;*

- в графической утилите «Сетевые соединения» выбрать режим «*И*спользовать статические *IP*-адреса» и ввести следующие параметры: «1Р-адрес» — 10.0.0.2, «Маска сети» — 255.0.0.0, «Основной шлюз» — 10.0.0.1; осуществить подключение к сети;
- проверить корректность установки параметров сетевого интерфейса *ethO* командой *ifconfig*, а затем проверить корректность установки маршрута по умолчанию командой *route*;проверить доступность *AstraClient1,* выполнив команду *ping***20.0.0.2;**
- выполнить отключение сетевого интерфейса *ethO* в графической утилите «Сетевые соединения» и проверить результат командой *ifconfig*;
- выполнить включение сетевого интерфейса *ethO* в графической утилите «Сетевые соединения» и проверить установку JP-адреса 10.0.0.2 командой *ifconfig.*

11. Проверить совместную работу приложений конфигурирования сетевых интерфейсов в ОССН *AstraClient1.* Для этого выполнить следующую последовательность действий:



- сконфигурировать статический сетевой адрес, используя настройки графического менеджера аналогичные настройкам файла / *etc/ network/interfaces* (файл */ etc/network/interfaces* содержит следующие строки: *«auto ethO», «iface ethO inet sta-tic», «address* 10.0.0.3», *«netmask*255.0.0.0», *«gateway* 10.0.0.10»):



« найти в каталоге / *etc/ Network Manager* файлы с сохранёнными настройками, которые были сделаны на предыдущем этапе.

11. Настроить *DNS*-сервер для организации следующей адресации и именования компьютеров: *AstraRouter (ethO)* — 192.168.1.3*router.aw.ru*, *AstraClient1* — 192.168.1.10

stationl.aw.ru. Для этого выполнить следующую последовательность действий в ОССН *AstraRouter.*

- запустить терминал *Fly* в «привилегированном» режиме;

сконфигурировать статический сетевой адрес, используя настройки графического менеджера аналогичные настройкам файла / *etc/ network/ interfaces* (файл / *etc/ network/ interfaces* содержит следующие строки: *«auto ethO», «iface ethO inet sta-tic», «address* 192.168.1.3», *«netmask* 255.255.255.0», «*gateway*192.168.1.1»);

- выполнить установку службы *DNS*-сервера командой *apt-getinstall bind/;*
- создать файл /*etc/bind/db.aw.ru* командой *mcedit /etc/bind/db.aw.ru co* следующим содержимым:

```
/etc/bind/db.aw.ru   [----]  0
$TTL 30
$ORIGIN aw.ru.

@ IN SOA aw.ru. admin.aw.ru. (
<------>2018110201;
<------>1d;
<------>1h;
<------>1w;
<------>2h;
)
@ IN NS aw.ru.
@ IN A 192.168.1.3

router IN A 192.168.1.3
station1 IN A 192.168.1.10
```

- создать файл /*etc/bind/*1.168.192.*in-addr.arpa.zone* командой *mcedit /etc/bind/l.* 168.192.*in-addr.arpa.zone* со следующим содержимым

```
/etc/bind/1.168.192.in-addr.arpa.zone   [----]  0 L:[  1+15
1.168.192.in-addr.arpa. 10800 IN SOA aw.ru. admin.aw.ru.(
<------>2018110201;
<------>1d;
<------>1h;
<------>1w;
<------>2h;
)

   NS aw.ru.
3  PTR aw.ru.
3  PTR router.aw.ru.
10 PTR station1.aw.ru.
```

- отредактировать файл / *etc/bind/ named.conf.local* командой *mcedit /etc/bind/named.conf.local,* добавив следующее содержимое:

13. Настроить параметры сетевых интерфейсов ОССН *Astra-Client1* для работы с *DNS-се*рвером. Для

```
zone "aw.ru" {
<------>type master;
<------>file "/etc/bind/db.aw.ru";
};

zone "1.168.192.in-addr.arpa" {
<------>type master;
<------>file "/etc/bind/1.168.192.in-addr.arpa.zone";
};
```

этого выполнить следующую последовательность действий:

- в графической утилите «Сетевые соединения» ввести следующие параметры: «1Р-адрес» — 192.168.1.10, «Маска сети» —255.255.255.0, «Основной шлюз» — 192.168.1.3, *«DNS*-домен» —*aw.ru,* «Поиск в домене» — *aw.ru,* «DTVS-сервер» — 192.168.1.3;
- выполнить перезагрузку и повторный вход в ОССН *AstraClient1* и *AstraRouter c* учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий»);
- запустить терминал *Fly* и проверить корректность функционирования *DNS*-сервера командами *ping stationl.aw.ru* и *ping router, aw.ru*;
- создать в ОССН *AstraClient1* учётную запись пользователя *test,* включить его в группу *astra-console;*
- запустить терминал *Fly* в «привилегированном» режиме командой *sudo fly-term* и изменить максимальный мандатный уровень доступа учётной записи test командой *pdpl-user-m* 0:2 *test*;
- выполнить выход и повторный вход в ОССН *AstraClient1* с учётной записью пользователя *test* (уровень доступа — 2, неиерархические категории — нет, уровень целостности — «Низкий»);
- запустить терминал *Fly* и осуществить попытку проверки функционирования DIVS-сервера командами *ping stationl.aw.ru* и*ping router.aw.ru,* проанализировать ошибки.

14. Настроить *DNS*-сервер в ОССН *AstraRouter* для работы с использованием механизма *privsock.* Для этого выполнить следующую последовательность действий:

- запустить терминал *Fly* в «привилегированном» режиме командой *sudo fly-term*\
- изменить файл / *etc/parsec/privsock.conf,* добавив в него строку (перед этим проверить, что файл */etc/parsec/privsock.conf* оканчивается пустой строкой), содержащую /*usr/shin/named*командой *echo* «/*usr/sbin/named» » /etc/parsec/privsock.conf;*
- проверить внесённые изменения командой *cat / etc/parsec/priv-sock. conf;*
- изменить настройки *DNS*- сервера в файле */ etc/ default/binds* командой *echo «PATH=/usr/lib/parsec/Ып:\$PATH» » /etc/default/ bind9;*
- перезапустить *DNS-с*ервер командой *service bind9 restart.*

15. Проверить работу ОССН *AstraClientl c DNS*-сервером в сессии, функционирующей от имени учётной записи пользователя *test* с мандатным уровнем доступа равным 2. Для этого выполнить следующую последовательность действий:

- запустить терминал *Fly* и проверить распознавание сетевого имени ОССН *AstraClientl* командой *ping stationl.aw.ru;*
- выполнить команду проверки доступности *DNS*-сервера *pingrouter, aw. ru.*

Содержание отчёта по выполненной работе

В отчёте о выполненной работе необходимо указать:

1. Полный перечень использованных команд с описанием их назначения.
2. Примеры выполнения команд, которые были использованы в ходе работы с описанием результатов их выполнения.
3. Описание порядка работы с графической утилитой *nm-applet*при настройке параметров проводных и беспроводных сетевых интерфейсов.
4. Описание порядка работы с командами при выполнении следующих операций:

- настройка адресации сетевого интерфейса;
- настройка статической маршрутизации;
- проверка настроенных параметров сетевого интерфейса и таблицы маршрутизации;
- проверка достижимости удалённых сетей;• настройка механизма *privsock.*

Контрольные вопросы

- Каково назначение пакета *iprouteil*
- Какие основные команды реализуются пакетом *iproute2?*
- Какие отличительные особенности пакетов *iproute*2 и *net-tools*?
- Какими командами пакетов *iprouteZ* и *net-tools* задаются параметры сетевого интерфейса (IP-адрес, маска сети)?
- Какими командами пакетов *iproute* и *net-tools* задаётся добавление статического маршрута?
- Каково назначение конфигурационных файлов в каталоге /*etc/Network-Manager?*
- Какие параметры ядра обеспечивают включение функции *«IP forward-ing*» ?
- Какими командами осуществляется проверка и управление характеристиками сетевых интерфейсов?
- Какие особенности настройки работы сетевых служб с использование механизма *privsock*?

4. 7. Лабораторная работа № 7.
Конфигурирование службы Astra Linux Directory

Цель работы

Получить практический опыт установки и настройки параметров службы *Astra Linux Directory (ALD)* в ОССН.

Время выполнения работы 6 академических часов.

Краткие теоретические сведения

В компьютерных сетях, построенных на основе ОССН, имеется возможность организовать централизованное хранение учётных записей пользователей в домене *ALD* (далее — домене), а также развёртывать централизованный защищённый файловый сервер, содержащий сетевые домашние каталоги данных учётных записей пользователей. Таким образом, у учётных записей пользователей *ALD* появляется возможность регистрации и доступа к своим сетевым объектам с любого компьютера, входящего в домен. Это особенно актуально, в случае территориальной удалённости между контроллером *ALD и* компьютерами, входящими в состав домена.

Хотя в ОССН версии 1. 6 также реализована более современная доменная инфраструктура *FreeIPA,* которая подробно рассмотрена в главах 1 и 3, её конфигурирование и настройка являются гораздо более сложными, чем *ALD,* и поэтому выходят за рамки лабораторной работы.

Администратор домена *ALD* выполняет следующие функции по управлению доменом:

- ●централизованное управление учётными записями пользователей домена с использованием команды *ald-admin* и графической утилиты «Политика безопасности» (для этого необходимо установить расширение *smolensk-security-ald);*
- ●настройка СЗИ, управляющих их доступом к файловым сущностям защищённого файлового сервера.

Централизованная база данных учётных записей пользователей домена *(DIB— Domain Information Base)* создаётся на основе службы *LDAP (Lightweight Directory Access Protocol),* обеспечивающей как организацию хранилища учётных записей пользователей *ALD,* так и процедуру аутентификации пользователей на компьютере с использованием *ALD.* Безопасность процедуры аутентификации пользователей домена обеспечивается применением протокола доверенной аутентификации *Kerberos. Для* синхронизации временнных меток при взаимодействии контроллера и клиентов *Kerberos* используется протокол *NTP (Network Time Protocol).*

При доступе к сущностям файловой системы компьютера, с которого осуществлён вход в домен с некоторой учётной записью пользователя, для неё применяются настройки управления доступом, хранящиеся на контроллере *ALD.* Если же на контроллере *ALD* (или

на специально выделенном компьютере) организуется защищённый файловый сервер, то настройки управления доступом для этой учётной записи пользователя применяются также к сущностям файловой системы этого контроллера. При этом доступ к ним от имени учётной записи пользователя *ALD* осуществляется по протоколу *CIFS (Common Internet File System),* являющемуся развитием протокола сетевого файлового обмена *SMB.*

Служба *ALD* обладает расширяемой архитектурой, состоящей из ядра, отвечающего за основной функционал системы, ряда интерфейсов *(LDAP, Kerberos)* и модулей расширения, команд и графических утилит настройки служб и подсистем *ALD,* что позволяет расширять функциональность *ALD,* устанавливая дополнительные пакеты. Основные пакеты, используемые при установке и настройке *ALD:*

- *ald-client-common* — клиентская часть *ALD* (можно также использовать метапакет *ald-client);*
- *ald-admin* — команды администрирования *ALD;*
- *ald-server-common* — серверная часть *ALD* (можно также использовать метапакет *ald-server);*
- *smolensk-security-ald* —расширения графической утилиты «Политика безопасности», позволяющие осуществлять управление доменом (можно также использовать метапакеты *ald-admin- ald-se* или *ald-admin-ald-server).*

На компьютере, осуществляющем функции контроллера *ALD,* операции по администрированию *ALD* выполняются от имени учётных записей пользователей, обладающих соответствующими административными полномочиями. В зависимости от назначенных привилегий администраторов *ALD* можно разделить на следующие группы по полномочиям:

- корневой администратор (имя *admin/ admin,* администратор *ALD)* — обладает всеми полномочиями по управлению доменом;
- администраторы (пользователи с привилегией *admin)* — обладают полномочиями по управлению конфигурацией домена и учётными записями пользователей;
- ограниченные администраторы (учётные записи пользователей с привилегиями *hosts-add* или *ald-hosts-add)* — обладают полномочиями по добавлению компьютеров в домен;
- пользователи утилит администрирования (пользователи с привилегией *adm-user)* — обладают полномочиями по запуску утилит администрирования;
- обычные пользователи.

Для администрирования домена используются команды *ald-admin* и графическая утилита «Политика безопасности», которая позволяет выполнять следующие действия с доменом:

- создание и администрирование учётных записей пользователей

- ●создание и администрирование групп;
- ●добавление и удаление компьютеров;
- ●резервирование и восстановление учётной информации баз данных домена;
- ●конфигурирование привилегий и политик СЗИ для учётных записей пользователей и групп;
- ●конфигурирование политик паролей *Kerberos;*
- ●администрирование доступа к съёмным устройствам;
- ●администрирование учётных записей сетевых служб (сервисов);
- ●контроль целостности (аудит) конфигурации домена.

При создании нового домена используется следующая последовательность действий:

- ●настройка сетевого соединения на контроллере *ALD* и компьютерах, которые будут включены в *ALD;*
- ●настройка именования контроллера и клиентов *ALD* для поддержки функционирования службы *LDAP;*
- ●конфигурирование и запуск контроллера *ALD;*
- ●запуск клиентов *ALD* на компьютерах, входящих в *ALD.*

Данная последовательность действий рассматривается при выполнении лабораторной работы.

При развёртывании средств обеспечения единого пространства пользователей с применением *ALD* используются следующие команды (примеры применения которых также рассмотрены в главе 3):

- ●*hostname* — команда вывода в терминал текущего имени компьютера;
- ●*apt-get* — команда управления пакетами;
- ●*ping* — команда отправки и получения пакетов *ICMP (Echo Request/ Echo Reply);*
- ●*ald-init* — команда инициализации базы данных *ALD;*
- ●*ald-client* — команда управления клиентом *ALD;*
- ●*ald-admin* — команда управления доменом *ALD.*

Используемое методическое и лабораторное обеспечение

1. Три компьютера с ОССН версии 1. 6, объединённые в сеть. Первый предназначен для использования в качестве контроллера *ALD* — далее обозначается *AstraServer;* остальные — компьютеры, подключаемые в домен *(AstraClient1, AstraClient2).* В ОССН настроена синхронизация времени с использованием протокола *NTP,* либо, при использовании виртуальных машин временные метки считываются автоматически из единого системного времени.
2. В каждой ОССН создана учётная запись пользователя *user,* с параметрами: максимальный и минимальный уровни доступа — О, неиерархические категории — нет, уровень

целостности — «Высокий», входит в группу администраторов — *astra-admin* (вторичная группа), разрешено выполнение привилегированных команд *(sudo).*

3.Дистрибутив ОССН.

4.Документация: «Операционная система специального назначения *«Astra Linux Special Edition».* Руководство администратора. Часть 1», «Операционная система специального назначения *«Astra Linux Special Edition».* Руководство по КСЗ. Часть 1».

5.Для выполнения работы в течение двух занятий необходимо обеспечить возможность сохранения состояния ОССН за счёт применения технологий виртуализации (создания виртуальных машин с ОССН).

Порядок выполнения работы

1.Для настройки сетевого соединения на контроллере и клиентах *ALD* начать работу со входа в ОССН *AstraClient*1*, AstraClient*2 и *AstraServer* в графическом режиме с учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий»).

2.В ОССН *AstraClient1, AstraClient2* и *AstraServer* выполнить настройку статических сетевых адресов 10. 0. 0. X/8, где сетевые адреса «10. 0. 0. X» задаются в соответствии со следующим списком: для *AstraServer* — 10. 0. 0. 10, для *AstraClient*1 — 10. 0. 0. 1, для *AstraClient*2 — 10. 0. 0. 2.

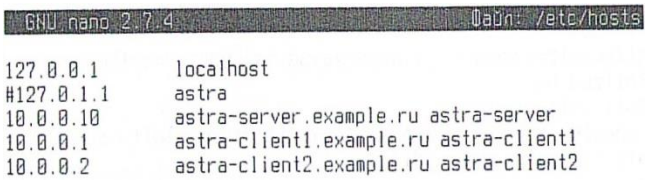
3.Выполнить перезагрузку и повторный вход в каждую ОССН с учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий»), затем запустить терминал *Fly.*

4.Выполнить проверку корректности настроек командой *ping.* При этом проверить доступность *AstraClient1, AstraClient2 с AstraServer* по сети командами: *ping* 10. 0. 0. 1 и *ping* 10. 0. 0. 2. Затем проверить доступность *AstraClient1* с *AstraClient*2 *командой ping* 10. 0. 0. 1.

5.Выполнить настройку имени контроллера и клиентов *ALD* для поддержки функционирования службы *LDAP.* Для этого необходимо, чтобы разрешение сетевых имён было настроено таким образом, чтобы сетевое имя компьютеров разрешалось, в первую очередь, как полное имя (например, *astra-server. example. ru).* При этом команда *hostname* должна возвращать короткое сетевое имя (например, *astra-server* для *AstraServer).* Для этого выполнить следующую последовательность действий:

- в ОССН *AstraServer* в «привилегированном» режиме терминала *Fly* выполнить команду *vim/etc/hostname* и сделать запись *«. astra-server»,* а в ОССН *AstraClient1* и *AstraClient*2 — *«astra- client1»* и *«astra-client2»* соответственно;
- в ОССН *AstraServer, AstraClient1* и *AstraClient2* в «привилегированном» режиме терминала *Fly* выполнить команду *vim/etc/hosts* и добавить следующие строки: «10. 0. 0. 10 *astra-server. example. ru astra-server»,* «10. 0. 0. 1 *astra-client*1*. example. ru astra-*

*client*1*»,* «10. 0. 0. 2 *astra-client*2*. example. ru astra-client*2*»,* и закомментировать строку, содержащую запись «127. 0. 1. 1» (для этого поставить в начале данной строки «#»):



- ●выполнить перезагрузку ОССН *AstraServer, AstraClient1, AstraClient2* и войти в ОССН *AstraClient1, AstraClient2* и *AstraServer* в графическом режиме с учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий»);
- ●в каждой ОССН запустить терминал *Fly,* выполнить команду *hostname* и проверить, что она возвращает короткие имена *«astra-server», «astra-client*1*»* и *«astra-client*2*»* для ОССН *AstraServer, AstraClient*1 и *AstraClient*2 соответственно:

root@astra-server:/home/user# hostname

astra-server

- ●проверить доступность *AstraServer* с *AstraClient*1*, AstraClientl*2 по сети с использованием имени, выполнив для этого команды ping *astra-server u ping astra-server. example. ru* в ОССН *AstraClient*1 и *AstraClient*2*;*
- ●проверить доступность *AstraClient1, AstraClient*2 с *AstraServer* по сети с использованием имени, выполнив для этого команды *ping astra-client, ping astra-client*2*, ping astra-client*1*. example. ru ping astra-client*2*. example. ru* в ОССН *AstraServer.*

6.Выполнить установку, конфигурирование и запуск контроллера. Для этого реализовать следующую последовательность действий в ОССН *AstraServer:*

- ●войти в графическом режиме с учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий»);
- ●запустить терминал *Fly* в «привилегированном» режиме командой *sudo fly-term;*
- ●выполнить установку пакетов для работы с контроллером командой *apt-get install ald-server-common* (здесь и далее при выполнении команды *apt-get* потребуется установка *DVD* с дистрибутивом ОССН);
- ●выполнить команду *vim/etc) ald/ald. conf* и проверить наличие параметров *«SERVER=astraserver. example. ru»* и *«DOMAIN =. example. ru»:*

DOMAIN=. example. com

The name of your domain (also used as Kerberos realm in upper-case).

Should be in the form:

. example. com

!NOTE! (for aid-server). If this value is changed - the server should be reinitialized by:

$ ald-init init

Or you should use the commands 'ald-init backup-ldif' and

'ald-init restore-backup-ldif'.

SERVER=astra-server. example. com

Fully qualified name of Astra Linux Directory server.

Should be in the form:

my-ald-server. example. com

- для того чтобы служба *ALD* заново считала изменения в файле */etc/ald,/ald. conf (если* они реально делались, иначе пропустить указанную далее команду) выполнить инициализацию командой *ald-init commit-config* (результатом будет информация об успешном конфигурировании службы *ALD);*
- выполнить команду инициализации *ald-init init* и по требованию этой команды подтвердить повторную инициализацию баз данных *LDAP* и *Kerberos,* ввести и подтвердить новый *Kerberos*-пароль *«kerberosroot»,* ввести и подтвердить новый пароль администратора *ALD «ald-root»:*

ВНИМАНИЕ! Команда 'init' УНИЧТОЖИТ ВСЮ БАЗУ ДАННЫХ LDAP u Kerberos!

Также Во Время Выполнения этой команды могут быть остановлены и перезапущены LDAP, Kerberos, NFS/Samba и некоторые другие службы.

Разыменованное имя компьютера: astra-server. example. com

Контроллер домена '. example. com' будет создан со следующими параметрами:

Сервер: astra-server. example. com

Роль сервера: Первичный контроллер домена

ID сервера: 1

Первичный контроллер домена: astra-server. example. com

Вы УВЕРЕНЫ, что хотите ВЫПОЛНИТЬ эту операцию? (yes/no) [no]: yes

Введите новый главный пароль к базе данных Kerberos (НЕ ЗАБУДЬТЕ ЕГО!): **** ********

Повторите пароль: **********

Введите новый пароль администратора Astra Linux Directory (НЕ ЗАБУДЬТЕ ЕГО!): *******

Повторите пароль: *******

Сохранение конфигурации. . .

Остановка сервиса smbd. . .

Остановка сервиса nmbd. . .

Обработка конфигурационного файла '/etc/exports'. . .

Переименование '/etc/exports. tmp' в '/etc/exports'. . .

Запуск сервиса nmbd. . .

Запуск сервиса smbd. . .

Перезапуск сервиса nscd. . .

Перезапуск сервиса nslcd. . .

Перезапуск сервиса aldd. . .

Astra Linux Directory сконфигурирована.

Сервер ALD активен.

Клиент ALD Включен.

Astra Linux Directory сервер успешно инициализирован.

7.Выполнить установку, конфигурирование, запуск клиентов *ALD.* Для этого реализовать следующую последовательность действий:

- ●в ОССН *AstraCUentl* запустить терминал *Fly* в «привилегированном» режиме командой *sudo fly-term;*
- ●выполнить установку пакета *ald-client- common* командой *apt- get install ald-client-common;*
- ●выполнить команду *nano/etc/ald/ald. conf* и отредактировать следующие параметры: *«SERVER = astra-server. example. ru»* и *«DOMAIN =. example. ru»;*
- ●выполнить инициализацию настроек и подключение к контроллеру командой *ald-client commit-config;*
- ●в качестве имени учётной записи пользователя администратора ввести пробел или *admin/ admin,* далее ввести пароль администратора *ALD ald-root;*
- ●выполнить запуск клиента *ALD* командами *ald-client start* (для включения компьютера в домен может отдельно использоваться команда *ald-client join).*

8.Осуществить проверку функционирования и настройку контроллера и клиентов *ALD.* Для этого выполнить следующую последовательность действий:

- ●в ОССН *AstraServer* выполнить установку расширения графической утилиты «Управление политикой безопасности», используемого для конфигурирования контроллера *ALD* командой *apt-get install smolensk-security-ald;*
- ●запустить графическую утилиту «Политика безопасности» через меню «Панель управления» главного пользовательского меню и открыть вкладку «Домен *ALD»,* соответствующую настройкам созданного домена;
- ●для администрирования домена необходимо выполнить подключение, проверить имя пользователя *«admin/ admin»,* ввести пароль администратора *ALD «ald-root»,* а затем проверить, что контроллер *ALD* активирован (при этом должна отображаться надпись «Сервер домена: *astra-server. example. ru»);*
- ●в дереве элементов вкладки политик безопасности контроллера *astra-server. example. ru* выбрать узел «Компьютеры» и проверить, что в состав домена с именем *«. example. ru»* входят контроллер *ALD «astra-server. example. ru»* и клиент *«astra-client1. example. ru»;*

9.Создать новую учётную запись пользователя *ALD* и осуществить вход с ней в ОССН *AstraClient*1*.* Для этого осуществить следующие действия:

●в ОССН *AstraServer* запустить графическую утилиту «Политика безопасности» через меню «Панель управления» главного пользовательского меню и открыть вкладку «Домен ALD» в разделе «Элементы»;

●осуществить подключение с учётной записью пользователя *« admin/ admin»;*

●в дереве элементов вкладки политик безопасности контроллера *astra-server. example. ru* выбрать узел «Пользователи» и создать новую учётную запись пользователя *userald,* при этом задав её пароль (обратить внимание на требование политики по сложности пароля);

●в учётной записи пользователя *userald* в вкладке «Привилегии домена» выбрать «Компьютеры» добавить только *astra-server. example. ru* и применить изменения;

●в ОССН *AstraClient*1*, AstraClient*2 выйти из ОССН;

●осуществить попытку входа в ОССН *AstraClient*1 с новой учётной записью пользователя *userald* и проанализировать выводимые ошибки;

●теперь в учётной записи пользователя *userald* в вкладке «Привилегии домена» выбрать «Компьютеры» и добавить *astra-client*1 *. example. ru;*

●войти в ОССН *AstraClient1* с учётной записью пользователя *userald;*

●осуществить попытку входа в ОССН *AstraClient*2 с учётной записью пользователя *userald* и проанализировать выводимую ошибку.

10.Осуществить установку, конфигурирование, запуск клиента *ALD* на *AstraClient*1*.* Для этого выполнить следующую последовательность действий:

●войти в ОССН *AstraClient*2 в графическом режиме с учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий»);

●запустить терминал *Fly* в «привилегированном» режиме командой *sudo fly-term;*

●выполнить установку пакета *ald-client-common* командой *apt- get install ald-client-common;*

●выполнить подключение к домену командой *ald-client join astra-server. example. ru;*

●проверить корректность модификации файла настройки клиента *ALD,* выполнив команду *head-*25 */etc/ald/ald. conf,* при этом должны быть установлены следующие параметры: *«SERVER = astra-server. example. ru»* и *«DOMAIN = . example. ru».*

11.Проверить корректность включения компьютера *astra-cli- ent2. example. ru* в домен *«. example. ru».* Для этого выполнить следующую последовательность действий:

●в ОССН *AstraServer* перезапустить графическую утилиту «Политика безопасности», затем открыть вкладку «Домен ALD» и выполнить подключение к домену;

- ●выбрать узел «Компьютеры» и проверить, что в состав домена с именем *«. example. ru»* входит клиент *ALD astra-client2. example. ru;*
- ●открыть узел «Привилегии домена» — *«useraid»,* в поле «Компьютеры» добавить *«astra-client2. example. ru»* к списку разрешённых компьютеров;
- ●выйти из ОССН.

12. Проверить функционирование сетевой файловой системы при доступе к домашним каталогам учётных записей пользователей *ALD.* Для этого выполнить следующую последовательность действий:
    - ●войти в ОССН *AstraClient2* в графическом режиме с учётной записью пользователя *userald;*
    - ●в ОССН *AstraClient2* в графической утилите «Менеджер файлов» открыть каталог «Документы», создать в нем текстовый файл с именем *file-from-al3. txt,* затем открыть данный файл в редакторе *Juffed* и добавить строку «Создан на ЦК 3»;
    - ●в ОССН *AstraClient1* в сессии, функционирующей от имени учётной записи пользователя *userald,* в графической утилите «Менеджер файлов» открыть каталог «Документы» и проверить наличие файла с именем *file-from-al3. txt;*
    - ●в ОССН *AstraServer* запустить терминал *Fly* и перейти в каталог */ald-export-home* командой *cd /ald-export-home;*
    - ●вывести содержимое текущего каталога командой *ls,* проанализировать результат;
    - ●определить дискреционные и мандатные атрибуты каталога *userald* командой *pdp-ls -M;*
    - ●выполнить попытку перехода в каталог *userald* и проанализировать выводимые ошибки;
    - ●запустить терминал *Fly* в «привилегированном» режиме командой *sudo fly-term,* запустить *Midnight Commander* командой *mc* и перейти в каталог */ald-export-home/userald/l0i0c0x0t0x0/* Документы;
    - ●вывести в терминал содержимое файла *file-from-al3. txt* командой *cat file-from-al3. txt.*
13. Осуществить настройку политики паролей учётных записей пользователей *ALD.* Для этого выполнить следующую последовательность действий:
    - ●в ОССН *AstraServer* в графической утилите «Политика безопасности» выбрать «Домен *ALD»* (при необходимости выполнить подключение к домену), далее «Политики паролей» — *« default»;*
    - ●во вкладке «Общие» в поле «Минимальная длина» ввести значение 5;
    - ●перейти к узлу «Пользователи» — *«userald»* и сменить пароль на «1234», а затем на «*Asdf*1».
14. Создать учётную запись пользователя *ALD* с использованием утилиты *ald-admin.* Для этого выполнить следующую последовательность действий:
    - ●в ОССН *AstraServer* запустить терминал *Fly;*

- ●создать новую учётную запись пользователя *ALD* командой *ald- admin user-add userald2,* задать новый пароль в соответствии с требованиями политики безопасности (не менее 5 символов): «Qwer1»;
- ●далее все параметры учётной записи пользователя *ALD,* установленные по умолчанию, за исключением последнего (там значение *«yes»);*
- ●вывести список учётных записей пользователей и компьютеров *ALD* командами *ald-admin user-list* и *ald-admin host-list* соответственно;
- ●создать группу компьютеров *ald_host_group*1 со следующим составом: *astra-client*1*. example. ru, astra-client*2*. example. ru,* командой *ald-admin hgroup-add ald_host_group*1*--host=astra-client*1*. example. ru--host=astra-client2. example. ru* (также второй узел можно включить в группу командой *ald-admin hgroup-mod ald_host_group*1*--add-hosts--host=astra-client2. example. ru);*
- ●в графической утилите «Политика безопасности» проверить наличие узла «Группы компьютеров/*ald_host_group1»;*
- ●модифицировать группу компьютеров *ald_host_group*1*,* добавив в неё компьютер *astra-server. example. ru* командой *ald-admin hgroup-mod ald_host_group*1*--add-hosts—hosts=astra-server. example. ru;*
- ●добавить учётной записи пользователя *userald2* привилегию доступа к группе компьютеров *ald_host_group*1 командой *ald-admin user-ald-cap userald*2*--host-group=ald_host_group*1*--addhosts;*
- ●в графической утилите «Политика безопасности» проверить наличие компьютера *astra-server. Example. ru и в* узле «Группы компьютеров» — *«ald_host_group*1*»* и наличие привилегии доступа к группе компьютеров *ald_host_group1 у учётной записи пользователя userald*2;
- ●проверить возможность входа в ОССН *AstraClient*1 и *AstraClient*1 с учётной записью пользователя *userald*2*,* после успешной проверки выйти из ОССН *AstraClient*1 и *AstraClient*2*.*

15.Осуществить проверку целостности и настроек *ALD,* для чего выполнить команду *ɑđd-admin test-integrity* и обратить внимание на выдаваемые предупреждения.

16.Перейти к администрированию учётной записи пользователей домена:

- ●в ОССН *AstraServer* в графической утилите «Политика безопасности» выбрать «Домен *ALD»;*
- ●настроить параметры мандатного управления доступом учётной записи пользователя *userald2:* установить в вкладке «МРД» максимальный уровень доступа 3, минимальный — 0;

- ●настроить параметры мандатного управления доступом учётной записи пользователя *userald-,* установить в вкладке «МРД» максимальный уровень доступа 2, минимальный — 0;
- ●для проверки войти в ОССН *AstraClientl* в графическом режиме с учётной записью пользователя *userald* (уровень доступа — 2, неиерархические категории — нет, уровень целостности — «Низкий»);
- ●выполнить команду *ald- admin test-integrity* и проверить отсутствие предупреждений.

17.Выполнить проверку функционирования единого задания устройств в *ALD:*

- ●в ОССН *AstraServer* в графической утилите «Политика безопасности» выбрать «Домен *ALD»;*
- ●открыть «Устройства и правила», «Правила», создать новое правило: выбрать импорт свойств, затем выполнить подключение USB-носителя, в появившемся дереве выбрать устройство, ввести имя правила *«Rule*1*»,* отметить «Включено» и применить изменения;
- ●открыть «Устройства и правила», «Устройства», выполнить добавление USB-носителя через импорт свойств, затем установить для него разрешения на чтение и запись для учётной записи пользователя *userald,* группы *«Domain Users»* и всех остальных, установить уровень конфиденциальности — 1, выбрать правило *«Rulel»* и активировать данное устройство, установив флаг «Включено»;
- ●войти в ОССН *AstraClient*1*,* в сессии, функционирующей от имени учётной записи пользователя *userald,* на уровнях доступа 0, 1 и 2, выполнить подключение USB-носителя, при этом проверить возможность монтирования (в соответствии с «Руководством администратора. Часть 1» монтирование учтённого носителя с файловой системой VFAT возможно только при входе в ОССН на том же уровне, с которым данный носитель учтён) и корректность (соответствие МРОСЛ ДП-модели) выполнения операций записи и чтения файлов на USB-носитель для каждого уровня доступа.

Содержание отчёта по выполненной работе

В отчёте о выполненной работе необходимо указать:

1.Полный перечень использованных команд с кратким описанием их назначения.

2.Примеры выполнения команд, которые были использованы в ходе работы с описанием результатов их выполнения.

3.Описание порядка работы с командами *ALD (ald-admin, ald- init, ald-client)* и графической утилитой «Политика безопасности» *(fly admin smc)* при осуществлении следующих действий со службой *ALD:*

- ●настройка адресации сетевого интерфейса;

- настройка контроллера *ALD*;
- настройка клиента *ALD*;
- управление параметрами контроллера *ALD.*

Контрольные вопросы

1. Каково назначение служб контроллера *ALD?*
2. Какие службы устанавливаются на контроллере *ALD?*
3. Какие особенности настройки имён компьютеров в домене *ALD?*
4. Каковы особенности настройки контроллера *ALD?*
5. Каковы особенности настройки клиента *ALD?*
6. Чем отличаются настройки *ALD* в режимах контроллера и клиента?
7. Какие основные настройки домена можно выполнять с использованием команды *ald-admin?*

4.8. Лабораторная работа № 8.
Управление программными пакетами. Настройка системных служб

Цель работы

Изучить порядок администрирования пакетов ОССН, в том числе используемых для этого команд и графических утилит, а также особенности настройки системных служб (демонов).

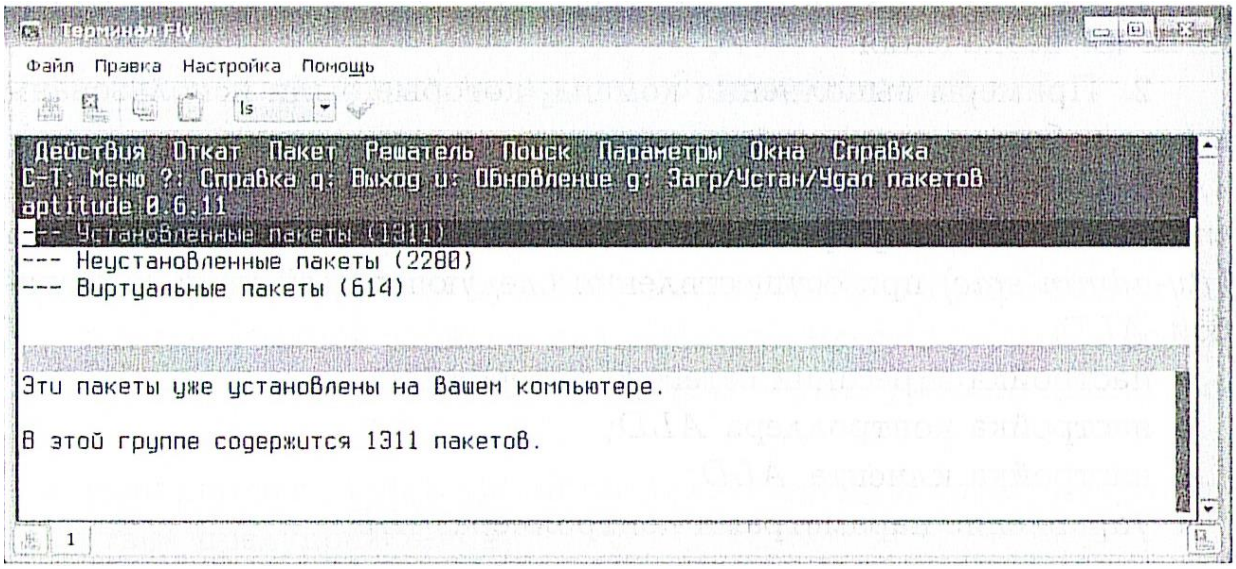
Время выполнения работы 4 академических часа.

Краткие теоретические сведения

Система управления пакетами ОССН включает несколько команд и утилит, которые могут работать в режиме командной строки, в псевдографическом и графическом режимах. При этом порядок действий при их использовании для изменения состава установленных в ОССН пакетов принципиально не отличается, за исключением, возможно, установки взаимосвязанных пакетов.

Для управления пакетами используются утилиты *aptitude,* «Менеджер пакетов *Synaptic»,* команды системы управления пакетами *APT* и команда *dpkg.*

Утилита *aptitude* имеет следующий интерфейс:



Она позволяет осуществлять установку, удаление, поиск по именам (в том числе неустановленных) и управление зависимостями пакетов.

При просмотре пакетов отображается состояние каждого пакета (первый символ в списке): *v* — виртуальный, *B* — неработоспособный, *u* — «распакованный», *C* — недонастроенный, *H* — недоустановленный, *c* — удалённый, но с сохранёнными файлами настроек *i* —установленный и *E* — внутренняя ошибка. Вторым символом может быть указан флаг автоматизации, означающий, что пакет был установлен как зависимость. Удобство использования данной утилиты также заключается в автоматическом формировании набора пакетов, которые являются зависимыми от удаляемого пакета. Это позволяет предотвратить

нежелательное удаление используемых пакетов при настройке других пакетов. После выбора необходимого набора действий их выполнение производится по нажатию *«g»* — загрузка/установка/удаление пакетов. Кроме графического. представления утилита *aptitude* может быть использована как консольная команда с параметрами.

Управление пакетами посредством *APT* выполняется командами, основными из которых являются: *apt - get, apt - cdrom, apt - cache* и *apt - config.*

Команда *apt - cdrom* предназначена для добавления в систему нового источника пакетов (репозитория), как правило, диска CD/ DVD. По умолчанию при установке ОССН один источник уже добавлен — это установочный диск ОССН. Настройки источников пакетов хранятся в файле */etc/apt/sources.list.* Данный файл настроек используется как командами вида *apt*,* так и утилитой *aptitude.* После установки ОССН он содержит единственную запись: *deb cdrom:* [Название диска]/ *smolensk contrib main non - free.* Первый элемент со значением *deb* указывает на тип источника — *debian package.* При использовании диска с исходными текстами вместо записи *deb* будет стоять *deb - src.* Второй элемент со значением *cdrom* указывает носитель источника — устройство типа *cdrom.* Для установки по сети может использоваться значение *http.* Далее указана метка диска в квадратных скобках и путь: «/». Затем расположен элемент с указанием названия дистрибутива: *smolensk.* После этого поля указываются наборы установочных пакетов: *contrib, main, поп - free.* Данные названия наборов для источника *cdrom* соответствуют подкаталогам каталога *pool,* расположенного в корне диска. Каждый новый источник заносится новой строкой в данный файл. Для игнорирования строки поддерживаются комментарии (в начале строки указывается символ «#»).

В случае, если файл */etc/ apt/sources.list* редактируется вручную, то для обновления списка пакетов необходимо выполнить команду *apt - get update.* Данная команда по списку активных источников, полученных из этого файла, считывает соответствующие наборы пакетов, которые далее используются для выбора устанавливаемых пакетов при выполнении команды *apt - get install* имя_пакета.

Для непосредственного управления пакетами используются команды *apt - get install* имя_пакета (установка), *apt - get remove* имя_пакета (удаление).

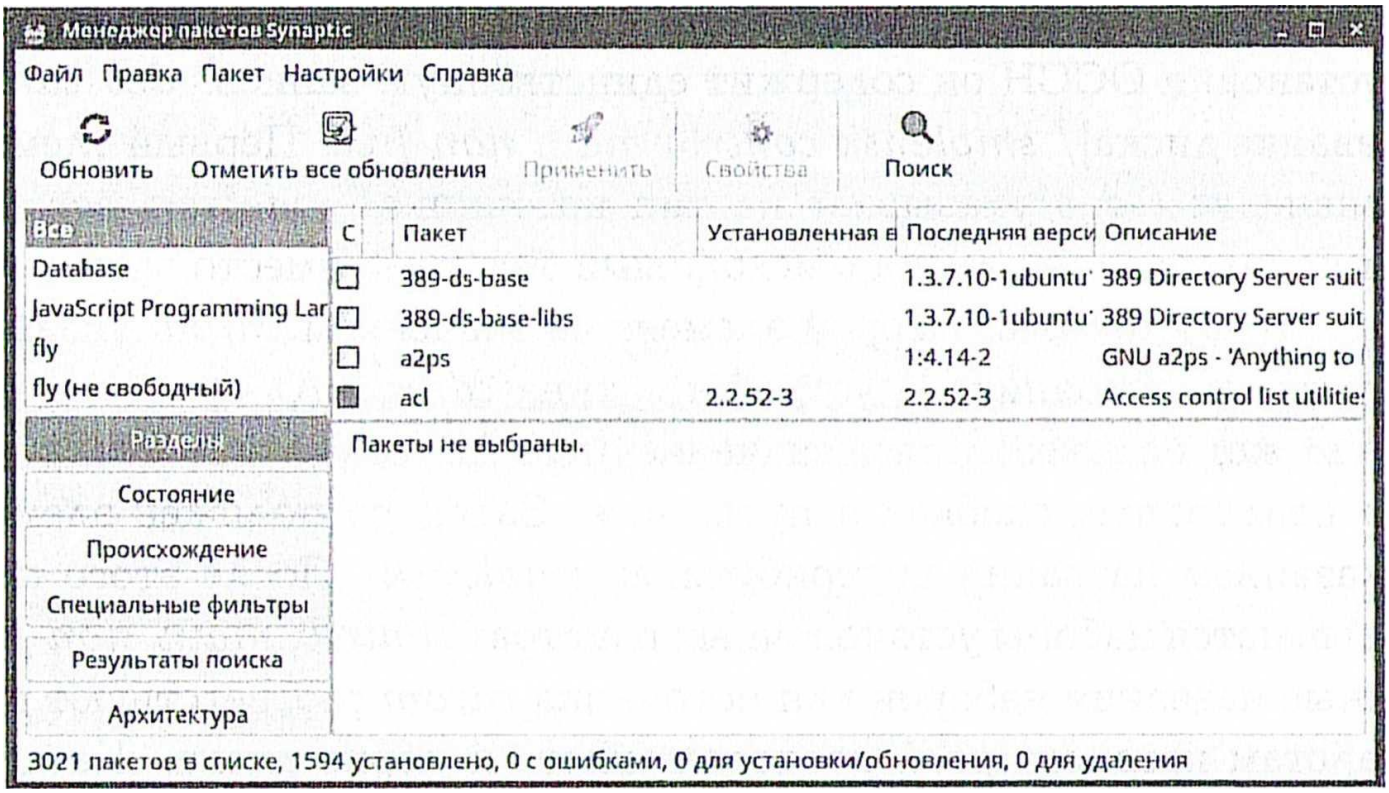
Команда *apt - cache* применяется для работы с кэшем пакетов. Она позволяет вывести список пакетов — *apt - cache pkgnames,* зависимости отдельного пакета — *apt - cache depends* имя_пакета, найти пакет в кэше по его имени — *apt - cache search.*

Команда *dpkg* реализует следующие основные действия:

- *dpkg - i* имя_файла_*deb*_пакета — установка пакета, находящегося в файле имя_файла_*deb*_пакета (данный файл имеет расширение *deb)*;
- *dpkg - r* имя_пакета — удаление пакета;

- *dpkg - L* имя_пакета — вывод файлов, содержащихся в пакете;
- *dpkg - S* имя_файла — поиск принадлежности заданного файла конкретному пакету

Управление пакетами с использованием графического интерфейса осуществляется графической утилитой «Менеджер пакетов «*Synaptic» (synaptic),* имеющей следующий интерфейс:



Она позволяет осуществлять фильтрацию пакетов по состоянию, устанавливать зависимости, устанавливать, переустанавливать, удалять и полностью удалять пакеты. В свойствах пакетов, отображаемых утилитой, содержится информация о файлах, которые были установлены этим пакетом с их расположением в ОССН, зависимости пакетов и доступные версии пакетов.

Для настройки системных служб в ОССН используются команды *service* (устаревшая) и *systemctl* и графическая утилита *system - dgenie* (запуск с использованием графического меню осуществляется в меню «Система», «Инициализация системы»). В ОССН версии 1.6 используется новый менеджер загрузки *Systemd.* С этим связаны значительные изменения порядка управления и создания сервисов.

Для управления запуском системных служб как и ранее может использоваться команда *service* имя_службы <команда>. В качестве параметров команды используются *start, stop, restart, status* для запуска, останова, перезапуска или определения статуса системной службы соответственно. С использованием утилиты *systemctl* запуск сервиса (в данном случае — юнита) выполняется командой *systemctl start* имя_юнита (аналогичные команды используются для останова — *stop,* перезапуска — *restart).* Просмотр уровней запуска

системных служб может быть осуществлён с использованием команд *service— status - all или.* более наглядной *systemctl — all list - units.* Аналогичные функции реализует графическая утилита *systemdgenie.*

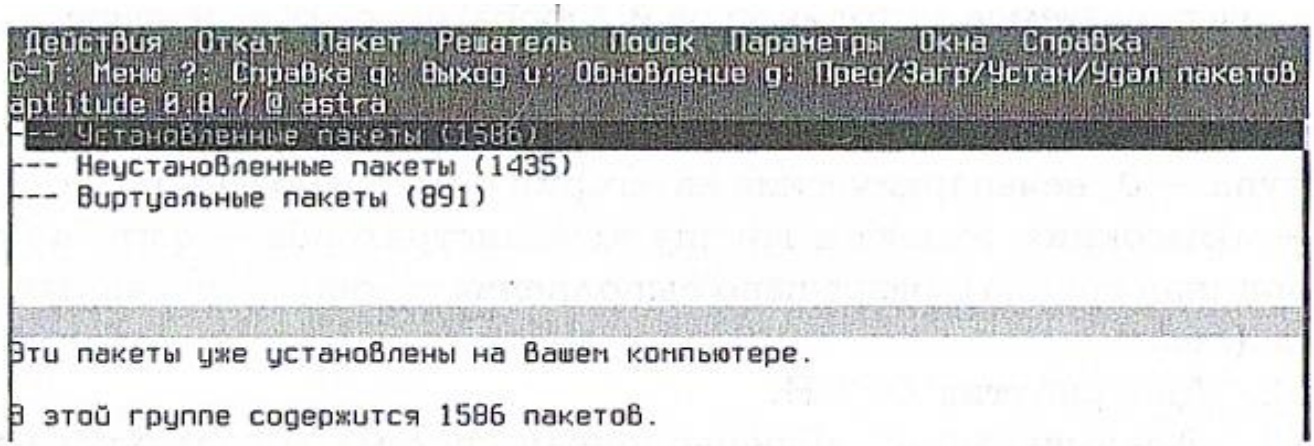
Используемое методическое и лабораторное обеспечение

- ОССН версии 1.6, в которой создана учётная запись пользователя *user, с* параметрами: максимальный и минимальный уровни доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий», входит в группу администраторов — *astra - admin* (вторичная группа), разрешено выполнение привилегированных команд *(sudo).*
- Дистрибутив ОССН.
- Документация: «Операционная система специального назначения *«Astra Linux Special Edition».* Руководство администратора. Часть 1», «Операционная система специального назначения *«Astra Linux Special Edition». «*Руководство по КСЗ. Часть 1», «Операционная система специального назначения *«Astra Linux Special Edition».* Руководство пользователя».
- Для выполнения работы в течение двух занятий необходимо обеспечить возможность сохранения состояния ОССН за счёт применения технологий виртуализации (создания виртуальных машин с ОССН).

Порядок выполнения работы

- Начать работу со входа в ОССН в графическом режиме с учётной записью пользователя *user* (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий»).
- Запустить терминал *Fly* и перейти в каталог */etc/apt.*
- Вывести дискреционные права доступа к файлу *sources.list* командой *ls – l sources.list.* Определить возможность изменения файла при работе от имени данной учётной записи пользователя.
- Выполнить команду *sudo fly - term* и перейти в каталог */etc/ apt.*
- Запустить *Midnight Commander* командой *mc.* Открыть для редактирования файл *sources.list* и закомментировать строку *deb cdrom...* путём установки символа # в начале строки.
- Выполнить команду *apt - get update,* и с использованием дистрибутива ОССН добавить источник пакетов *cdrom* командой *apt - cdrom add.*
- Установить пакет *vim - gtk* командой *apt - get install vim - gtk.* Проверить возможность работы в редакторе *vim - gtk,* выполнив команду *vim.gtk* 1*.txt.* Выйти из редактора, набрав *«:wq».* Проверить наличие файла l*.txt* и удалить его командой *rm* 1*.txt.*

• Запустить второй терминал *Fly* командой *sudo fly - term* и в нём выполнить команду *aptitude:*



• В первом терминале выполнить попытку удаления пакета *vim - gtk* командой *apt - get remove vim - gtk,* проанализировать выводимые ошибки:

```
root@astra:/# apt-get remove vim-gtk
E: Не удалось получить доступ к файлу блокировки /var/lib/dpkg/lo
ck - open (11: Ресурс временно недоступен)
E: Не удалось выполнить блокировку управляющего каталога (/var/li
b/dpkg/); он уже используется другим процессом?
root@astra:/# apt-get remove vim-gtk
E: Не удалось получить доступ к файлу блокировки /var/lib/dpkg/lock
 - open (11: Ресурс временно недоступен)
E: Не удалось выполнить блокировку управляющего каталога (/var/lib/
dpkg/); он уже используется другим процессом?
```

• Завершить *aptitude* во втором терминале. В первом терминале повторить удаление пакета *vim - gtk* командой *apt - get remove vim - gtk* и *apt - get remove vim - gui - common.* Проанализировать отображаемые в терминале сообщения и определить число удалённых пакетов.

• Выполнить настройку пакетов с использованием *APT,* для этого осуществить следующую последовательность действий:

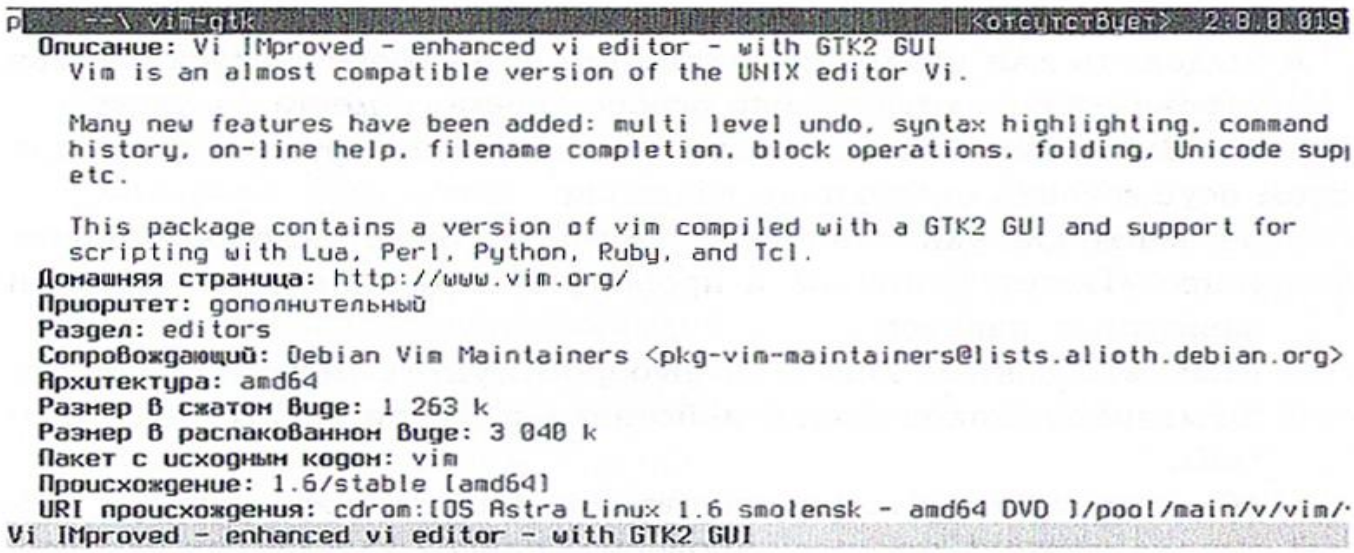
- найти пакеты, содержащие модули для Web - сервера *Apache2,* командой *apt - cache search libapache2 - mod - *,* и определить их назначение по описанию;
- выполнить установку пакета *texstudio* командой *apt - get install texstudio*;
- выполнить очистку локального хранилища файлов пакетов за исключением */var/ cache/ apt/ archives/* и */var/ cache/ apt/ archives/ partial/* командой *apt - get clean*;
- выполнить очистку локального хранилища командой *apt - get autoclean*;
- удалить пакет *texstudio* с удалением конфигурационных файлов командой *apt - get purge libquazip*5 - 1;
- выполнить повторную установку пакета *texstudio* командой *apt - get install texstudio* и определить число установленных пакетов;

- проверить назначение и зависимости пакета *texstudio* командами *apt - cache show texstudio* и *apt - cache showpkg texstudio;*
- выполнить удаление пакета *texstudio c* сохранением конфигурационных файлов командой *apt - get remove texstudio* и проанализировать число удаляемых пакетов и отсутствие сообщения об очистке файлов настроек;
- проверить наличие ошибок в зависимостях пакетов командой *apt - get - f install*;
- выполнить удаление лишних пакетов, которые были установлены автоматически, но теперь не требуются командой *apt - get autoremove.*

• В первом терминале выполнить команду *aptitude* и открыть раздел «Неустановленные пакеты» — *«Editors» — «main».*

• Для работы с пакетом *vim - gtk* осуществить следующие действия:

- выполнить просмотр следующих свойств пакета *vim - gtk:* описание назначения, размер в сжатом и распакованном виде, версии и зависимости:



- в третьей строке в псевдографическом интерфейсе перейти к вкладке «Пакет» и выбрать данный пакет для установки нажатием клавиши «+», обратив внимание, что одновременно с его установкой были добавлены его зависимости: пакет *vim - gui - common*;
- определить статус установки пакетов: *vim - gtk — pi* (пакет помечен для установки), *vim - gui - common — piA* (пакет помечен для установки и выбран автоматически в качестве зависимого);
- выполнить установку и получить следующие данные: перечень устанавливаемых пакетов (1 пакет — *vim - gtk),* пакеты, которые

устанавливаются автоматически, список пакетов, предлагаемых другими пакетами (первые две группы пакетов устанавливаются полностью, а последняя может быть дополнительно выбрана при установке).

• С использованием утилиты *aptitude* выполнить следующие действия:

- проверить назначение и зависимости пакета *texstudio* командами *apt - cache show texstudio* и *apt - cache showpkg texstudio;*

- выполнить попытку удаления (« - ») пакета *vim - gui - common* и проанализировать выведенные предупреждения;
  - нажать клавишу «е» для того, чтобы определить какие дополнительные действия необходимо будет выполнить совместно с удалением пакета *vim - gui - common,* затем нажать клавишу *«ТА В»* для выбора «Удалить *vim - gtk»,* далее нажать «.» для выбора решения удалить «Действия: 1 оставить неизменным», затем «!» и *«g»* для применения предложенных действий;
- проверить, какие пакеты были дополнительно помечены для удаления, и удаляется ли при этом пакет *vim - gtk*;
- выделить для удаления пакет *папо* и в строке статуса (третья строка) определить размер освобождаемого места на диске.
  - Выполнить попытку удаления пакета *vim - gui - common,* при этом осуществить следующие действия:
- отметить для удаления пакет *vim - gui - common* выбором пункта меню «Пакет/Удалить» и проверить выделение для удаления зависимых пакетов;
- отменить данные действия выбором пункта меню «Действия/ Отменить незаконченные действия» и выйти из утилиты *aptitude.*
  - Выполнить удаление пакета *vim - gui - common* командой *apt - get remove vim - gui - common,* проверить список удаляемых пакетов.
  - Выполнить удаление пакета *папо* командой *apt - get remove nano* и проверить список удаляемых пакетов.
  - Выполнить установку пакетов с использованием команды *dpkg,* при этом осуществить следующие действия:
- в «привилегированном» режиме терминала *Fly* перейти в каталог */media/ cdrom/ pool/ main/ п* подключённого *DVD* с дистрибутивом ОССН;
- для установки пакета *папо* найти соответствующий каталог командой *ls | grep «^папо»* и перейти в него;
- установить пакет *папо* командой *dpkg - i* имя_файла_пакета, затем проверить работу редактора *папо;*
- выполнить переход в каталог *pool/main;*
- перейти в каталог *v/vim* и осуществить попытку установки пакета *vim - gtk* командой *dpkg — install* имя_файла_пакета и проанализировать результат (пакет *vim - gtk* будет не настроен).

- Выполнить проверку статуса установленных пакетов, для чего осуществить следующие действия:

- в «привилегированном» режиме терминала *Fly* запустить утилиту *aptitude* и проанализировать предупреждения;
- нажать клавишу «*e*» для просмотра решений: первое решение — это установка пакета *vim - gui - common;*
- выполнить просмотр следующего решения и выйти из утилиты *aptitude.*

- Вернуться в терминал *Fly* и выполнить следующую последовательность действий:

- установить недостающий пакет *(vim - gui - common)* командой *dpkg — install* имя_файла_пакета;
- запустить утилиту *aptitude* и проанализировать причину отсутствия ошибочных зависимостей и требований установки дополнительных пакетов;
- вернуться к вкладке «Пакеты» и перейти «Установленные пакеты» — *«editors»* — *«main»,* далее выделить в списке пакет *vim - gtk* и определить его статус;
- выйти из утилиты *aptitude* и в терминале *Fly* выполнить конфигурацию *vim - gtk* командой *dpkg —configure vim - gtk* и проверку возможности запуска командой *vim.gtk;*
- проверить в приложении *aptitude,* что статус пакета *vim - gtk* изменился на «*i*» (т. е. пакет был сконфигурирован и теперь успешно установлен).

- Вывести в терминал *Fly* информацию об установленном пакете *vim,* для чего осуществить следующие действия:

- вывести в терминал содержимое пакета *vim - gtk* командой *dpkg - L vim - gtk;*
- вывести в терминал содержимое пакета *vim - gtk* командой *aptitude show vim - gtk*;
- вывести в терминал список пакетов, имя которых начинается с *vim,* командой *aptitude search "^vim"* и определить статусы их установки;
- вывести в терминал список пакетов, имя которых содержит *gtk,* командой *aptitude search "gtk"*;
- определить принадлежность файла */usr/share/lintian/overrides/vim - gtk* пакету *vim,* для чего выполнить команду *dpkg - S /usr/ share/lintian/overrides/vim - gtk*;
- вывести в терминал список пакетов, содержащих файлы, полный путь с именем которых заканчивается на «/*vim»* командой *dpkg - S "*/vim".*

- Выполнить настройку пакетов в графической утилите *Synaptic,* для чего осуществить следующие действия:

- запустить графическую утилиту *Synaptic,* при этом ввести пароль текущей учётной записи пользователя для возможности работы в «привилегированном» режиме;

- открыть меню «Настройки», «Репозитории» и проверить наличие неактивного репозитория, который был ранее закомментирован в файле *sources.list*;
- скопировать *URI* для неактивного репозитория;
- удалить неактивный и активный репозитории.
  - В графической утилите *Synaptic* через меню «Настройки», «Репозитории» выбрать действие «Создать», при этом:
- в элемент *URI* ввести запись *cdrom:* [*OS Astra Linux* 1.6 *smolensk - amdCA DVD*]/;
- о в элемент «Дистрибутив» — *smolensk*;
- в элемент «Разделы» — *contrib main non - free*;
- закончить изменения и выполнить обновление для повторного считывания источника пакетов:

  - В графической утилите *Synaptic* выполнить следующую последовательность действий:
- в элемент «Быстрый фильтр» ввести «*vim*» и отметить пакет *vim - common* для полного удаления с использованием контекстного меню, после удаления проанализировать предупреждения ОССН;
- выбрать в списке фильтров «Ошибки зависимостей» и отметить пакет *vim* для полного удаления, и после удаления проанализировать список полностью удаляемых пакетов, их число и суммарный объем дискового пространства;
- в «привилегированном» режиме терминала *Fly* выполнить попытку удаления пакета *nano* командой *apt - get remove nano* и проанализировать предупреждения ОССН;
- завершить работу с графической утилитой *Synaptic,* выполнить удаление пакета *nano* командой *apt - get remove nano.*
  - Для управления системными службами выполнить следующую последовательность действий:
- в «привилегированном» режиме терминала *Fly* получить текущие статусы запускаемых системных служб командой *service — status - all*;
- выполнить попытку определения статуса системной службы *avahi - daemon* командой *service avahi - daemon status* и проанализировать результат;
- для анализа порядка вызова команды *service* вывести на экран содержимое файла *avahi - daemon* командой *cat/etc/init.d/ava - hidaemon;*

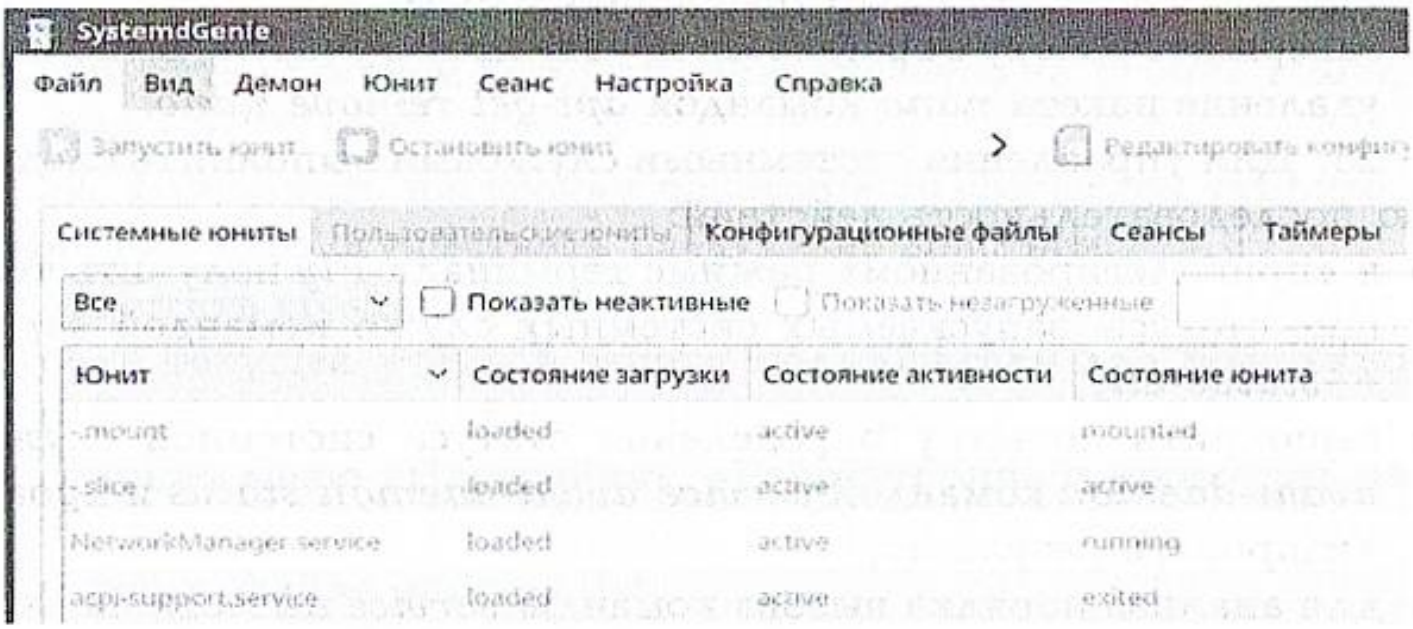
- аналогично по содержимому файла *avahi - daemon* найти реализацию вызова команды *status*;
- вывести в терминал содержимое файла *avahi - daemon* командой *cat/etc/init.d/avahi - daemon* и проанализировать его структуру, при этом найти особенности реализации вызова команды *status.*

  • Для управления запуском системных служб в «привилегированном» режиме терминала *Fly* осуществить следующие действия (при этом проверяется обратная совместимость с службой инициализации):

- определить статус загрузки системной службы *ssh* командой *service — status - all | grep ssh* и командой *ls/etc/rc2.d| grep ssh*;
- установить запуск службы сервера *ssh* на уровнях по умолчанию командой *update - rc.d ssh enable*;
- определить изменения в статусе системной службы *ssh* командой *service —status - all | grep ssh* и командой *ls/etc/rc2.d|grep ssh*;
- выполнить остановку, запуск и перезапуск системной службы *ssh* командами *service ssh stop, service ssh start, service ssh restart.*

27. Осуществить запуск системных служб из графической утилиты «Инициализация системы» *(systemdgenie),* осуществив следующие действия:

- в «привилегированном» режиме терминала *Fly* запустить графическую утилиту «*И*нициализация системы» командой *systemdgenie &:*

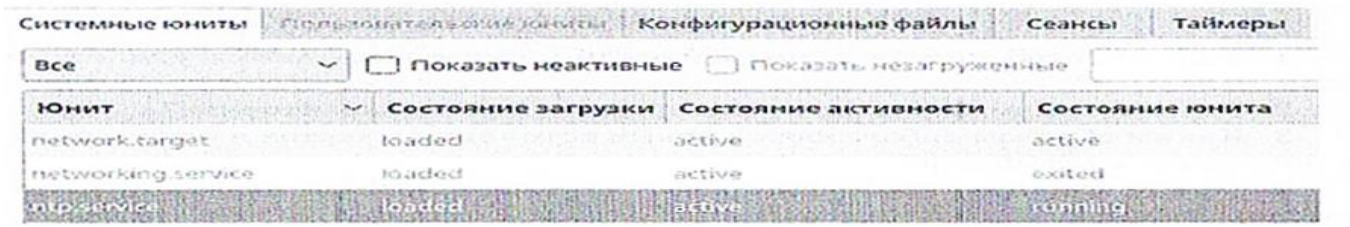


- запретить запуск системной службы *ssh,* для этого выделить юнит *ssh.service,* в контекстном меню выбрать «Отключить юнит»;
- выполнить перезагрузку и повторный вход в ОССН;

- проверить настройку службы *ssh* командой *service — status - all* и с использованием графической утилиты «Инициализация системы» (при этом юнит *ssh.service* должен отсутствовать);
- повторно активировать службу *ssh* с помошью *update - rc.d,* выполнить перезагрузку;
- запустить графическую утилиту «Инициализация системы» в столбце «Юнит» выделить *ssh,* в контекстном меню выбрать «Редактировать файл юнита» и определить зависимость запуска данной службы от других служб в открывшемся окне (раздел [*Unit*] параметр *After*);
- вывести в терминале *Fly* содержимое начала файла */etc/init.d/ssh* командой *head/etc/init.d/ssh* и сравнить запись *Required - Start* с содержимым окна *ssh в* графической утилите «Инициализация системы»;
- активировать службу *ntp* с использованием *update–rc.d*, выполнить перезагрузку:

```
root@astra:/home/user# update-rc.d ntp enable
```

- в графической утилите «Инициализация системы» в столбце «Юнит» найти службу *ntp*, затем в контекстном меню выбрать «Маскировать юнит»:



- проверить статус системной службы *nip* командой *systemctl list - units* (для этого выполнить поиск «nip» с использованием комбинации клавиш «/» и «*ntp*»), что системная служба замаскирована, выполнить перезагрузку и проверить статус загрузки службы.

Содержание отчёта по выполненной работе

В отчёте о выполненной работе необходимо указать:

- Полный перечень использованных команд с кратким описанием их назначения.
- Примеры выполнения команд, которые были использованы в ходе работы с описанием результатов их выполнения.
- Описание порядка работы при осуществлении следующих действий:

- удаление пакета без зависимостей;
- удаление пакета с зависимостями;
- установка новых пакетов;
- создание новых репозиториев;
- редактирование действующих репозиториев и обновление информации о пакетах.
  - Описание порядка работы при установке и удалении пакетов с использованием утилит *aptitude,* «Менеджер пакетов *Synaptic»,* команд *APT* и команды *dpkg.*

- Описание особенностей функционирования команды *dpkg* при установке пакетов с зависимостями.

6. Описание порядка работы с командами и графической утилитой «Инициализация системы» *(systemdgenie)* при осуществлении следующих действий с системными службами:

- просмотр статуса;
- просмотр настроек запуска системных служб;
- управление (запуск, перезапуск, останов, просмотр состояния).

Контрольные вопросы

- Каковы особенности одновременной работы нескольких утилит управления пакетами?
- Где и в каком формате хранятся настройки репозиториев?
- Каким образом добавляются новые источники пакетов и осуществляется их повторное считывание?
- Какие команды позволяют работать с зависимыми пакетами в автоматическом режиме?
- Какие особенности выполнения команд удаления пакетов?
- Как определить статус установленного пакета с использованием команды *dpkg* и утилиты *aptitude*?
- Как определить статус запуска системной службы?
- Каким образом можно произвести останов, запуск, перезапуск системной службы?
- Какие команды используются для просмотра состояния и уровней запуска системной службы?

4. 9. Лабораторная работа № 9.
Настройка защищенного режима работы ОССН в соответствии с Astra Linux Red-Book

Цель работы

Изучить особенности настройки безопасной конфигурации компьютера для работы с ОССН в соответствии с *Astra Linux Red- Book.*

Время выполнения работы 6 академических часов.

Краткие теоретические сведения

Выполнение лабораторной работы основывается на описании настройки безопасной конфигурации компьютера для работы с ОССН *Astra Linux Special Edition* с использованием материалов справочного центра *Astra Linux Red-Book.* Совокупность данных настроек позволяет достичь режима работы ОССН, когда все её механизмы защиты будут активированы и полнофункциональны.

Настройки включают конфигурирование ОССН по следующим направлениям:

•доверенная загрузка ОССН;

•конфигурирование оборудования компьютера с ОССН;

•защитное преобразование данных на жёстком диске;

•полнофункциональный режим работы мандатного контроля целостности;

•контроль цифровой подписи SLF-файлов;

•отключение терминалов для непривилегированных учётных записей пользователей;

•блокировка командных интерпретаторов и макросов для непривилегированных учётных записей пользователей;

•блокировка установки бита исполнения;

•блокировка трассировки исполнения программ;

•использование межсетевого экрана;

•работа в режиме киоска.

Детали работы ОССН по большинству перечисленных направлений подробно рассмотрены в главах 1 и 3. Дополнительно справочные материалы по настройке защищённого режима работы ОССН приведены в документации «Руководство по КСЗ. Часть 1».

Используемое методическое и лабораторное обеспечение

- Подготовленный компьютер для установки ОССН *Astra Linux Linux Special* версии 1. 6, отвечающий требованиям к оборудованию для использования ОССН (при разбиении диска требуется жёсткий диск объёмом от 20 Гбайт).
- Дистрибутив ОССН.
- Документация: «Операционная система специального назначения *«Astra Linux Special Edition».* Руководство администратора. Часть 1», «Операционная система специального

назначения *«Astra Linux Special Edition».* Руководство по КСЗ. Часть 1», «Операционная система специального назначения *«Astra Linux Special Edition».* Руководство пользователя».

- Для выполнения всей лабораторной работы необходимо использовать отдельный компьютер, а не виртуальные машины.

Порядок выполнения работы

- Для обеспечения доверенной загрузки ОССН выполнить установку и настройку (при наличии) аппаратно-программного модуля доверенной загрузки (АПМДЗ).
- Выполнить предварительную конфигурацию оборудования компьютера:

•настроить *BIOS:* установить пароль блокировки входа в *BIOS* (при задании пароля руководствоваться требованиями к генерации паролей, аналогичные требованиям к сложности паролей учётных записей пользователей ОССН);

•при наличии в *BIOS* опций блокировки установки битов исполнения (для процессоров *Intel Execute Disable Bit* и для процессоров *AMD No Execute Bit)* включить их;

•при наличии на серверах «недоверенных» систем контроля и управления вида *ILO, RSA, iDRAC, ThinkServer EasyManage, AMT, iMana* необходимо выполнить их отключение, и использовать, при необходимости, альтернативные решения вида *IP KVM;*

•для *Intel*-платформ необходимо устранить уязвимость *Intel-SA*- 00086 в *Intel Management Engine* (если он интегрирован в процессор) посредством установки обновления микропрограммы *Intel Management Engine* (производитель оборудования должен обеспечить данную возможность: для этого могут использоваться либо обновления *BIOS,* либо дополнительное ПО для интеграции обновлений); для проверки можно использовать утилиту *Intel-SA*-00086 *Detection Tool.*

- Перейти к установке ОССН, используя для этого первый установочный диск. Далее необходимо выполнить установку ОССН с включённым защитным преобразованием диска (для упрощения процесса установки может также быть использован режим «Авто — использовать весь диск с защитным преобразованием *LVM»,* так как данный режим содержит отключаемую стадию перезаписывания диска случайными данными перед установкой ОССН, стадию формирования ключевой фразы преобразования — длина от 20 символов, которая будет использоваться для защиты разделов с использованием *dm-crypt)* и обеспечить невозможность физического доступа к жёсткому диску, на котором установлена ОССН. В рамках работы используется метод разметки «Вручную».

Для выполнения всех вышеперечисленных операций необходимо выполнить:

•осуществить загрузку с основного (установочного) диска ОССН;

•выбрать язык «Русский», затем «Графическая установка»;

•принять лицензионное соглашение, установить способ переключения языка, задать имя компьютера *«astra»,* имя учётной записи администратора *«user»,* задать и подтвердить пароль администратора (который удовлетворяет требованиям сложности), выбрать часовой пояс;

•выбрать метод разметки «Вручную» и удалить все имеющиеся разделы;

•в «свободном месте» выполнить создание нового раздела размером 300Мб («Первичный»), задать для него файловую систему *Ext*4 и точку монтирования */boot,* включить метку «загрузочный» и закончить изменения раздела;

•в «свободном месте» выполнить создание нового раздела размером 2Гб («Первичный», позже он будет использоваться для каталога */home),* задать точку монтирования «отсутствует», закончить изменение раздела;

•выполнить создание нового раздела размером 2,5Гб («Первичный», он будет использоваться для подкачки), указать использовать как «раздел подкачки»;

•аналогично создать два логических раздела объёмом по 1Гб (при выборе размера разделов следует помнить, что при размере раздела */tmp* менее 250Мб вероятно возникновение ошибок при работе с графикой или с большими объёмами данных), задать для них файловую систему: *Ext*4 и точки монтирования */tmp и /var/tmp* (вводится вручную);

•в свободном месте создать раздел «/» («Логический») *и* задать для него файловую систему *Ext4;*

•выбрать «Настроить защитное преобразование для томов», сохранить сделанные изменения разделов;

•затем «Создать защитное преобразованные тома» и выбрать разделы, доступ к которым будет задаваться паролем: второй раздел с каталогом */home* (выбрать «Ключ защитного преобразования» — «Ключевая фраза»), третий раздел с подкачкой (выбрать «Ключ защитного преобразования» — «Произвольный ключ»);

•закончить создание разделов;

•выполнить стирание разделов;

•задать «ключевую фразу для защитного преобразования» второго раздела (проверить, что запрос выдаётся именно для этого раздела) в соответствии с требованиями к сложности паролей;

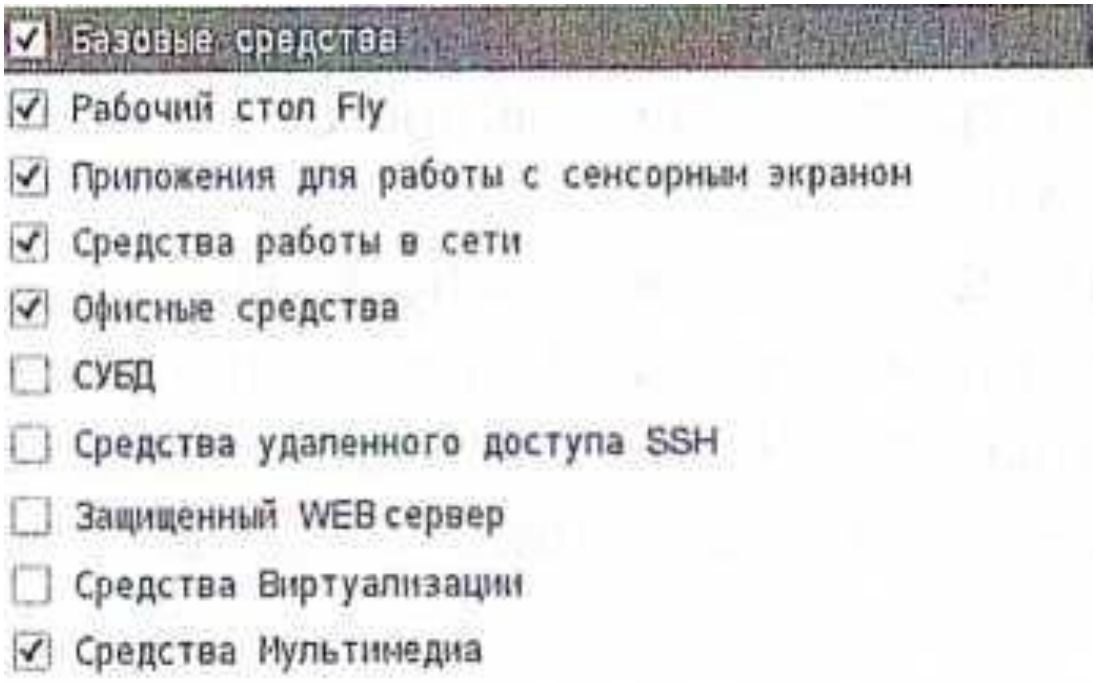
•задать тип использования и точки монтирования, созданные с применением защитного преобразования томов: *sda2_crypt — /home, sda3_crypt* — подкачка (см. рисунок на следующей странице);

•по окончании выбрать «Закончить разметку и записать изменения на диск» (см. рисунок на следующей странице);

Перед вами список настроенных разделов и их точек монтирования. Выберите (тип файловой системы, точку монтирования и так далее), свободное место, что устройство, чтобы создать на нем новую таблицу разделов.

•выбрать необходимые для работы наборы пакетов, *ALD* не устанавливать, дополнительные параметры настройки ОССН не устанавливать, установить *GRUB* в главную загрузочную запись (см. рисунок на следующей странице);

Выберите устанавливаемое программное обеспечение:



•установить пароль блокировки входа в настройке *GRUB* (при задании пароля руководствоваться требованиями к генерации паролей).

- Для дальнейшей настройки параметров ОССН выполнять вход от имени учётной записи администратора *user* на уровне целостности «Высокий».
- Установить все доступные обновления безопасности ОССН с сайта *http://astralinux.ru/update.html* (при наличии доступа к сети Интернет).
- Настроить загрузчик на ядро *Hardened,* и убрать из меню все другие варианты загрузки, включая режимы восстановления:

•запустить графическую утилиту «Панель управления», «Система», «Загрузчик *GRUB2»* в вкладке «Основное», «Время ожидания», установить требование автоматической загрузки пункта меню «Hardened» (без режима восстановления): «Немедленно»;

•отключить поиск дополнительных ОС (включен опцией «Проверка наличия операционных систем») и генерацию *recoverymode* (включена опцией «Сгенерировать записи для восстановления системы»), при этом произойдёт переформирование файла */boot/grub/grub. cfg;*

•нажать «Применить» и далее для выполнения настроек выполнить редактирование файла /*boot/grub/grub. cfg* для этого запустить терминал *Fly* в «привилегированном» режиме;

•выполнить создание резервной копии файла */boot/grub/grub. cfg ;*

•выполнить редактирование файла */boot/grub/grub. cfg,* исключив из него опцию загрузки (последний параметр *menuentry,* начиная со строки *«menuentry . . . generic . . .* {», до символа «}») ядра *Generic* (тем самым будут удалены лишние варианты загрузки ОССН).

- Выполнить перезагрузку и дополнительную настройку *BIOS:*

•при использовании архитектур, отличных от *Intel,* установить пароль на загрузчик согласно документации;

•установить единственным устройством для загрузки ОССН — жёсткий диск, на который была произведена установка ОССН;

•при поддержке компьютером соответствующих технологий включить режим загрузки *secureboot* с использованием ключей учётной записи пользователя компьютера (создать *USB-* носитель с помощью *astra-secureboot,* и далее импортировать ключи в *BIOS's,* соответствии с инструкциями *https://wiki.astralnux.ru/pages/viewpageaction?pageId=20217938).*

- Проверить параметры монтирования разделов в файле */etc/ fstab:*

•раздел */boot* рекомендуется монтировать с опциями *ro* (перед обновлением ядра смонтировать в *rw),* для этого параметр монтирования указанного каталога в файле /*etc/fstab* изменить с *defaults* на ro;

•разделы */home, /tmp, /var/tmp* рекомендуется монтировать с опциями *noexec, nodev, nosuid* для чего параметры монтирования указанных каталогов в файле /*etc/fstab* изменить с *defaults* на *rw, noexec, nodev, nosuid.*

- Включить блокировку интерпретаторов:

•при использовании графического интерфейса запустить утилиту *fly-admin-smc,* открыть «Монитор безопасности» и выбрать пункт 5 «Блокировка интерпретаторов», нажать «Настроить блокировку интерпретаторов» и далее выбрать «Включить блокировку интерпретаторов»;

•при настройке из терминала *Fly* необходимо в «привилегированном» режиме выполнить команду просмотра текущих настроек командой *systemctl is-enabled astra-interpreters-lock* (если блокировка не настраивалась, то выводится предупреждения об ее отсутствии *astra-interpreters-lock. service)*; для включения выполнить команду *astra-interpreters-lock enable* (отключение производится запуском данной команды с параметром *disable);* после настройки для проверки повторить команду *systemctl is- enabled astra-interpreters-lock:*

root@astra:/home/user# systemctl is-enabled astra-interpreters-lock enabled

- Выполнить настройку блокировки консоли (блокировка надраивается автоматически при установке ОССН):

•запустить графическую утилиту «Политика безопасности» (раздел «Настройки безопасности / Политика консоли и интерпретаторов»);

• Включить блокировку консоли**.**

- Включить блокировку установки бита исполнения:

•для блокировки только до перезагрузки ОССН выполнить команду *echo* 1 > */parsecfs/nochmodx*;

•для полной блокировки во всех сессиях ОССН выполнить команду *echo* 1 > /*etc/parsec/nochmodx* (настройка действует только после перезагрузки ОССН) или *astra-nochmodx-lock enable* (блокировка действует сразу, в том числе в текущей сессии ОССН);

•данные настройки доступны в графической утилите «Политика безопасности» (раздел «Настройки безопасности / Системные параметры»).

- Включить блокировку макросов:

•данная настройка («Макросы / Блокировка макросов») доступна в графической утилите «Политика безопасности» (раздел «Настройки безопасности / Системные параметры») и с помощью утилиты *astra-macros-lock*;

•в офисном пакете *LibreOffice* выбрать меню «Сервис», «Параметры», далее *«LibreOffice»,* «Безопасность», «Безопасность макросов» и выбрать параметр «Очень высокий».

- Включить блокировку трассировки *ptrace*:

•в терминале *Fly* в «привилегированном» режиме выполнить команду просмотра текущих настроек *systemctl is-enabled astra- ptrace-lock* (если блокировка не настраивалась, то выводится предупреждение об отсутствии *astra-ptrace-lock. service)*;

•для включения выполнить команду *astra-ptrace-lock enable* (отключение производится запуском данной команды с параметром *disable)*;

•после настройки для проверки повторить команду *systemctl is-enabled astra-ptrace-lock*

root@astra:/home/user# systemctl is-enabled astra-ptrace-lock enabled

- Включить контроль цифровой подписи *ELF*-файлов (исполняемых файлов):

•установить в файле */etc/digsig/digsig_initramfs. conf значение* параметра *DIGSIG_ELF_MODE* -1;

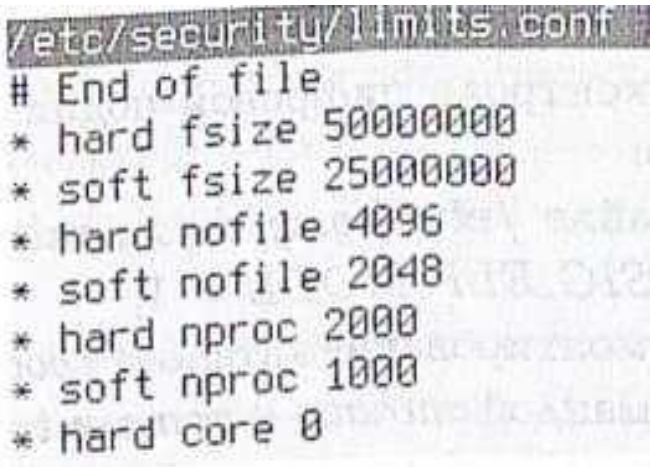
•выполнить перемонтирование каталога /*boot* с доступом на чтение и запись командой *mount -o remount,rw /boot*;

•выполнить команду *update-initramfs -u -к all* и перезагрузить ОССН.

- Включить гарантированное удаление файлов и папок:

•в файле */etc/fstab* добавить параметр *secdel=x* (где *x* определяет число стираний фиксированными последовательностями вида «11111111», «01010101», «10101010», «00000000») или *secdelrnd* (стирание псевдослучайной последовательностью) к параметрам монтирования разделов (например для каталога */home*).

- Включить межсетевой экран *ufw* (графическая утилита *gufw):*



•в меню «Панель управления», «Прочее», «Настройка межсетевого экрана» выбрать необходимый профиль и изменить статус на «Включено»;

•при необходимости задания уровня журналирования выбрать его в меню «Правка», «Параметры»;

•настроить межсетевой экран (модуль ядра *NetFilter* с использованием консольной команды *iptables)* в минимально необходимой конфигурации (т. е. настроить режим, когда по умолчанию все запрещено, кроме необходимых исключений, для чего могут также использоваться команды *iptables, ufw).*

- Включить системные ограничения *ulimits:*

•установить в файле */etc/security/limit. sconf* параметры системных ограничений: * *hard core* 0, * *hard fsize* 50000000, * *hard nproc* 1000;

•из терминала *Fly* в привилегированном режиме выполнить команду просмотра текущих настроек *systemctl is-enabled astra- ulimits- control* (если блокировка не настраивалась, то выводится предупреждение об отсутствии *astra-ulimits-control-service);*

•для включения выполнить команду *astra-ulimits-control enable* (отключение производится запуском данной команды с параметром *disable);*

•после настройки для проверки повторить команду *systemctl is-enabled astra-ulimits-control:*

- Включить режим киоска для непривилегированных учётных записей пользователей (пользователей с уровнем целостности «Низкий»), для чего использовать графическую утилиту *fly-admm- kiosk* (порядок включения режима киоска для учётной записи пользователя описан в документации «Руководство по КСЗ. Часть 1»).
- Включить графический киоск *Fly* с помощью графической утилиты *fly-admin-smc* (порядок включения режима графического киоска для учётной записи пользователя описан в документации «Руководство по КСЗ. Часть 1»).
- Включить контроль подписей (в настройках модуля *digsig: DIGSIG_XATTR_MODE=1*) в расширенных атрибутах (*xattr*), для чего сгенерировать ключи и подписать цифровой подписью в *xattr* все основные файлы ОССН (рекомендуемые каталоги для подписи: */lib,* /*lib64*, */lib32, /bin, /sbin, /boot, /opt, / srv, /usr*).
- При использовании мандатного управления доступом и обработке файлов с уровнем конфиденциальности выше минимального настроить дополнительное защитное преобразование файлов (с использованием команды *gpg* или встроенных функций файлового менеджера *fly-fm*)*:*

•открыть в файловом менеджере *fly-fm* каталог «Документы» и с использованием контекстного меню создать файл «Документ *Libre Office»;*

•с использованием контекстного меню файла «Документ *Libre- Office. odt»* выбрать пункт меню «Действия», «Защитное кодирующее преобразование», ввести 2 раза пароль; в результате будет создан файл «Документ *LibreOffice. odt. gpg»,* теперь данный файл может быть «открыт» (при этом будет восстановлен исходный файл), либо выполнено «Защитное раскодирующее преобразование» (при этом после восстановления файл «Документ *LibreOffice. odt. gpg»* будет удалён) с использованием ввода пароля.

22. Для работы с конфиденциальной информацией в сети настроить защитное преобразование пакетов с помощью создания доверенной сети *VPN:*

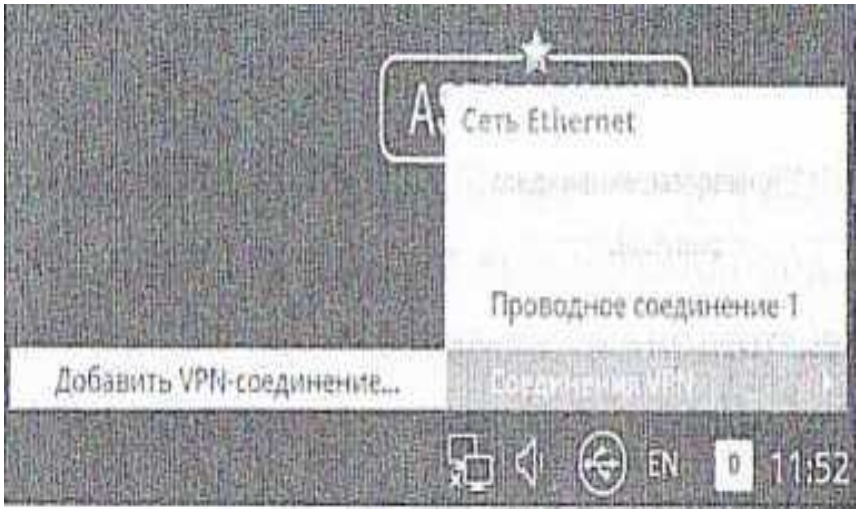
•осуществить настройку *VPN* соединения в меню сетевого подключения (по нажатию левой кнопки мыши на значке сети) с помощью меню «Соединения *VPN»,* «Добавить *VPN* соединение. . . »;

•выбрать соединение *OpenVPN* установить параметры соединения в соответствии с параметрами сервера *VPN,* к которому осуществляется подключение;

•в вкладке «*VPN»* для установки соединения ввести адрес сервера *VPN* в поле «Шлюз», указать соответствующий тип аутентификации на сервере в поле «Тип» и задать параметры подключения, соответствующие данному типу;

•дополнительные параметры (такие, как сжатие, используемые алгоритмы шифрования, протокол и пр.) задать после нажатия кнопки «Дополнительно. . . » в вкладке *«VPN»* в соответствии с настройками *VPN;*

•задать параметры получения адреса сети *VPN* в вкладках «Параметры *IPv4»* или «Параметры *IPv6»:*



23. Настроить обработку конфиденциальной информации при обмене почтовыми сообщения, используя защитные GPG-преобразования писем с помощью плагина для почтового приложения *Thunderbird Enigmail,* для чего при создании каждого сообщения электронной почты использовать быструю кнопку доступа «Защита» и выбирать «Шифровать это сообщение» и/или «Подписывать это сообщение».

- При создании новых учётных записей пользователей формировать пароли с заданным уровнем сложности в соответствии с политикой безопасности, для чего использовать графическую утилиту «Политика безопасности».
- Настроить *pam_tally* для задания необходимых параметров блокировки учётных записей пользователей (при попытках подбора паролей) с использованием графической утилиты «Политика безопасности» задать параметры количества «Неуспешных попыток» (например, 2 попытки), а также периода блокировки и разблокировки (например, 60 секунд).
- Настроить дисковые квоты в ОССН (для этого требуется установленный пакет *quota):*

•в файле */etc/fstab* добавить соответствующие опции монтирования (например, *usrquota* — включения квот для пользователей, *grpquota* — для включения квот для групп);

•выполнить перемонтирование изменённых файловых систем командой *mount -o remount* точка_монтирования;

•для создания квот (индексных файлов, которые описывают сами квоты) использовать команду *quotacheck -fcgum /home;*

•отредактировать квоты заданного пользователя, например, для учётной записи пользователя *test* выполнить команду *edquota —u test* и выполнить редактирования настроек квот в редакторе (редактор по умолчанию *nano,* который может быть изменён командой *update-alternatives--config editor),* затем сохранить изменения.

- Отключить все неиспользуемые сервисы (в том числе сетевые), которые запускаются при старте ОССН командой *systemd- genie.*
- Настроить параметры ядра ОССН в файле */etc/sysctl. conf.*

•настроить дополнительные параметры ядра: *fs. suid-dumpable* = 0, *kernel. randomize_va_space* = 2, *net. ipv4. ip_forward* = 0, *net. ipv4. conf. all. send_redirects* = 0, *net. ipv4. conf. default. send_ redirects* = 0;

•перегрузить ОССН (данные параметры будут автоматически применены после перезагрузки ОССН).

- Заблокировать исполнение модулей *python* с расширенным функционалом:

•выполнить команду *«find/usr/lib/python* -type f -name "_ctype*" -exec sudo dpkg-statoverride--update--add root root* 640 {}\;» (для удаления можно применять аналогичную команду, заменив *«--update --add root root* 640» на «--remove»);

•проверить изменения (установку владельца *root:root* и прав доступа 640 *«rw,r,-»)* командой *find/usr/lib/python* -type f -name " _ctype*"* | *xargs ls -l.*

- Запретить непривилегированным учётным записям пользователей (с уровнем целостности «Низкий») подключение сменных устройств (носителей), к которым потенциально может быть осуществлён несанкционированный доступ, для чего с использованием графической утилиты «Политика безопасности» не включать такие учётные записи пользователей в группу *floppy* (данная настройка в ОССН действует по умолчанию).
- Настроить систему аудита на сохранение файлов журналов (*log*-файлов) на удалённой машине. Для этого применить централизованное протоколирование с использованием системы *Zabbix* (подробно порядок действий описан в документации «Руководство администратора. Часть 1»).
- Установить мандатный контроль целостности на всех основных файлах и каталогах корневой файловой системы ОССН:

•в графической утилите *fly-admin-smc* «Политика безопасности», «Мандатный контроль целостности», «Целостность файловой системы» нажать кнопку «Установить» для изменения мандатного уровня целостности объектов файловой системы в соответствии с заданным фильтром (данное действие можно выполнить с использованием команды *set-fs-ilev)*;

•установку мандатного контроля целостности следует проводить после всех настроек безопасности, так как дальнейшее администрирование возможно только войдя под высоким уровнем целостности, или после снятия мандатного контроля целостности с файловой системы командой *unset-fs-ilev.*

Содержание отчёта о выполненной работе

В отчёте о выполненной работе необходимо указать:

- Полный перечень использованных команд с кратким описанием их назначения.
- Примеры выполнения команд, которые были использованы в ходе работы с описанием результатов их выполнения.
- Описание порядка работы с графической утилитой «Политика безопасности» *(fly-admin-smc)* при осуществлении настройки необходимых параметров защищённого режима работы ОССН.

4. Список и назначение используемых утилит, необходимых в ходе настройки защищённого режима работы ОССН в соответствии с *Astra Linux Red-Book.*

Контрольные вопросы

- Каковы особенности установки ОССН для её дальнейшей работы в защищённом режиме в соответствии с *Astra Linux Red-Book?*
- Какие необходимо выполнить блокировки для работы ОССН в защищённом режиме в соответствии с *Astra Linux Red-Book?*
- Для чего для «непривилегированных» учётных записей пользователей осуществляется блокировка интерпретаторов, терминалов и установки бита исполнения?
- Какие специальные настройки монтирования файловой системы необходимо использовать в файле */etc/fstab*?
- Что в ОССН подразумевается под «защитным преобразованием»?
- Какие подсистемы используют защитные преобразования?
- Как можно настроить параметры блокировки учётной записи пользователя?
- Какие средства настройки межсетевого экранирования доступны в ОССН?

Исходники - https://share.dmca.gripe/jLCe20YIvnYJbhri.zip
E-mail для вопросов о редактировании – rtvnsm@pm.me
Издание не имеет лицензии и является цифровой копией оригинальной книги - https://www.techbook.ru/book.php?id_book=1075
Пожалуйста, после скачивания не удаляйте этот текст, так мы можем взаимодействовать для дальнейшей редакции этого учебного пособия.
Edit doc - https://share.dmca.gripe/iE4eqbBCc3wmYqvo.docx