УТВЕРЖДАЮ
Должность и подпись лица, утвердившего документ от организации разработчика

_____ И.О. Фамилия

«___» _____ 2010 г.

ОПЕРАЦИОННАЯ СИСТЕМА АЛЬТ ЛИНУКС СПТ 6.0

Описание применения

ЛИСТ УТВЕРЖДЕНИЯ

Обозначение_документа-ЛУ

Должности и подписи руководителя организации, выпустившей документ, руководителя подразделения, разработавшего документ, руководителя разработки (разработчика), исполнителей разработки документа и нормоконтролера.

_____ И.О. Фамилия

«___» _____ 2010 г.

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

2010

УТВЕРЖДЕН

Обозначение_документа-ЛУ

ОПЕРАЦИОННАЯ СИСТЕМА АЛЬТ ЛИНУКС СПТ 6.0

Описание применения

Обозначение_документа

Листов 80

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

2010

1. АННОТАЦИЯ

Данный документ описывает применение операционной системы Альт Линукс СПТ 6.0, разработанной в ООО «Альт Линукс».

Описание применения состоит из четырёх основных частей, в которых раскрываются основные вопросы применения, структуры и функционирования ОС Альт Линукс СПТ 6.0. Также рассматриваются организация входных и выходных данных в системе, и конфигурация технических средств, необходимых для применения операционной системы.

В первом разделе приводятся назначение, основные принципы организации, возможности операционной системы, её основные характеристики, ограничения, накладываемые на область её применения.

Во втором разделе указываются условия, необходимые для выполнения операционной системы, структура технических и программных средств и требования к ним, общие характеристики входной и выходной информации, а также требования и условия организационного, технического и технологического характера и т.п.

В третьем разделе указываются определения задачи и методы её решения, приводится общая структура и алгоритмы функционирования ОС Альт Линукс СПТ 6.0.

В четвёртом разделе приводятся сведения о входных и выходных данных. Указывается их характер и организация, частота обновления и пр.

СОДЕРЖАНИЕ

2. НАЗНАЧЕНИЕ ПРОГРАММЫ

2.1. Описание и область применения операционной системы

ОС Альт Линукс СПТ 6.0, далее по тексту ALT Linux (либо ОС), представляет собой совокупность интегрированных программных продуктов, созданных на основе операционной системы Linux, и является на данный момент мощнейшим альтернативным средством замены серверных продуктов компании Microsoft.

Основой интеграции изделия является дистрибутив ALT Linux 6.0 фирмы Альт Линукс, созданного на основе открытых исходных кодов в соответствии с лицензией GNU GPL (GNU General Public License).

ОС Альт Линукс СПТ 6.0 предназначена для обеспечения выполнения программ в защищённой среде и представляет собой совокупность программных средств и эксплуатационной документации.

ОС предназначена для группового и корпоративного использования, автоматизации информационных, конструкторских и производственных процессов предприятий (организаций, учреждений) всех возможных типов и направлений.

2.2. Основные функции ОС Альт Линукс СПТ 6.0

ОС обеспечивает обработку, хранение и передачу информации в круглосуточном режиме эксплуатации.

В зависимости от конфигурации, сервер на основе ОС Альт Линукс СПТ 6.0 может обслуживать процессы в пределах одной компьютерной системы или процессы на других машинах через каналы передачи данных или сетевые соединения.

Встроенные средства защиты информации (СЗИ) изделия обеспечивают выполнение функций защиты информации в объёме требований 4 класса защищённости документа «РД. СВТ. Защита от НСД к информации» (Гостехкомиссия России, 1992), что даёт возможность использования ОС для защиты конфиденциальной и секретной информации и программ служебного назначения.

2.3. Основные свойства операционной системы

- обеспечивает мультизадачность процессов;
- обладает масштабируемостью системы (установка изделия как на одном компьютере, так и в сетях ЭВМ различной архитектуры);
- обеспечивает многопользовательский режим эксплуатации;

- обеспечивает поддержку мультипроцессорных систем;
- обеспечивает поддержку виртуальной памяти;
- обеспечивает сетевую обработку данных;
- обладает устойчивостью к вирусам;
- обеспечивает выполнение функций защиты информации в объеме требований 4 класса защищённости документа РД СВТ от НСД;
- обладает устойчивостью работы.

2.4. Системы и решаемые задачи

В соответствии с логикой работы в ОС ALT Linux выделяются следующие системы и решаемые ими задачи:

1) Система работы с файлами.
2) Система управления процессами:
   2.1) Контроль создания и удаления процессов.
   2.2) Контроль распределения системных ресурсов.
   2.3) Синхронизация процессов.
   2.4) Модуль межпроцессорного взаимодействия.
3) Система управления памятью:
   3.1) Распределение оперативной памяти между прикладными задачами.
   3.2) Очистка (обнуление) освобождаемых областей оперативной памяти ЭВМ.
4) Система ввода-вывода:
   4.1) Доступ к периферийным устройствам.
   4.2) Буферизация данных.
   4.3) Взаимодействие с драйверами устройств.
5) Система администрирования.
6) Система управления доступом:
   6.1) Идентификация, проверка подлинности и контроль доступа субъектов:
      6.1.1) в систему;
      6.1.2) к терминалам, ЭВМ, каналам связи, внешним устройствам ЭВМ;

6.1.3) к каталогам, файлам;

6.2) Управление потоками информации.

6.3) Квотирование – разграничение дискового пространства.

7) Система журнализации:

7.1) Регистрация:

7.1.1) входа (выхода) субъекта доступа в (из) систему;

7.1.2) запуска (завершения) программ и процессов (заданий, задач);

7.1.3) доступа программ субъектов доступа к защищаемым файлам, включая их создание, удаление и передачу по сети;

7.1.4) доступа программ субъектов доступа к терминалам, ЭВМ, каналам связи, внешним устройствам ЭВМ, программам, каталогам, файлам;

7.1.5) изменения полномочий субъектов доступа;

7.1.6) создаваемых защищаемых объектов доступа.

7.2) Сигнализация попыток нарушения защиты.

8) Система обеспечения целостности:

8.1) Обеспечение целостности программных средств и обрабатываемой информации.

8.2) Наличие службы защиты информации.

8.3) Наличие средств восстановления.

Настройка операционной системы и получение информации о её текущем состоянии осуществляется с использованием системы администрирования, в которой предусматривается применение паролей и безопасных протоколов. Защита от ошибочных действий администратора предусматривает их обнаружение и отображение компонентами операционной системы.

3. УСЛОВИЯ ПРИМЕНЕНИЯ

3.1. Используемые технические средства

3.1.1. Общие требования к применяемому оборудованию

Основу аппаратного обеспечения ОС ALT Linux составляют IBM PC-совместимые ПЭВМ. Желательным является подключение ПЭВМ с установленной ОС ALT Linux к сети переменного тока через источник бесперебойного питания (ИБП).

ПЭВМ с ОС ALT Linux объединяется в локальную сеть типа Ethernet с клиентскими рабочими местами.

3.1.2. Основные параметры функционирования ОС

Наименование параметра (характеристики)	**Значение параметра (характеристики)**
Операционная среда (ALT Linux 6.0)	32-х или 64-х разрядная, UNIX-подобная
Поддержка стандартов	IEEE POSIX.1, UNIX System V, Berkley System Distribution UNIX
Класс защищённости	4

3.1.3. Основные параметры для нормального функционирования ОС

Параметр	**Значение параметра**
Диапазон рабочих температур, ºС	15-40
Допустимый диапазон влажности воздуха, %	45-95
Допустимый диапазон давления, кПа	84-107

3.1.4. Технические требования к конфигурации компьютера

Наименование параметра (требования)	**Значение параметра (требования)**
Технические средства	компьютер типа IBM PC
Процессор	Intel или совместимый с ним, включая AMD. Для 32-битной версии процессор должен поддерживать технологию PAE.
RAM (оперативная память)	минимум 256 Мб (рекомендуется от 1 Гб и более)
Наличие свободного места на HDD (жёстком диске)	не менее 1 Гб (рекомендуется 12 Гб и более)
Наличие DVD-ROM (привода dvd дисков)	Наличие DVD-ROM, необходимо для инсталляции дистрибутива

3.1.5. Концентратор ЛВС

Для обеспечения функционирования ОС ALT Linux в рамках локальной вычислительной сети Ethernet необходимо наличие концентратора ЛВС (Hub или Switch), удовлетворяющего следующим условиям: количество портов RJ-45 не менее 1 на каждый подключаемый компьютер.

3.1.6. Кабель ЛВС

Сегмент кабеля ЛВС представляет собой отрезок кабеля типа экранированная витая пара 5-й категории не более 100 метров, снабжённый с обоих концов разъёмами типа RJ-45.

3.1.7. Источник бесперебойного питания

Источник бесперебойного питания должен обеспечивать при аварии системы электропитания работу подключённого оборудования от аккумуляторов на время, необходимое для запуска резервной энергосистемы (если таковая присутствует), либо достаточного для сохранения всех необходимых данных и безопасного завершения работы системы. Для этой цели источник бесперебойного питания должен удовлетворять следующим условиям: мощность не менее 600Вт.

Источник бесперебойного питания не является обязательным для обеспечения функционирования операционной системы.

3.1.8. Программное обеспечение

Дистрибутив изделия поставляется на 1 DVD (для одной архитектуры). Внесение обслуживающим персоналом изменений в программное обеспечение в процессе эксплуатации не предусматривается. Структура каталогов программного обеспечения СЗИ приведена в Приложении.

3.2. Структура программных средств

ОС Альт Линукс СПТ 6.0 состоит из набора компонентов предназначенных для реализации функциональных задач необходимых пользователям (должностным лицам для выполнения определённых должностными инструкциями, повседневных действий) и поставляется в виде дистрибутива и комплекта эксплуатационной документации.

В структуре ОС Альт Линукс СПТ 6.0 можно выделить следующие функциональные элементы:

- Операционная среда изделия (ОСр);
- Операционная система изделия (ОС);
- Ядро ОС;
- Системные библиотеки;

- Встроенные средства защиты информации (КСЗ);
- Системные приложения;
- Программные серверы;
- Web-серверы;
- Системы управления базами данных (СУБД);
- Прочие серверные программы;
- Интерактивные рабочие среды;
- Графическая оболочка GNOME;
- Командные интерпретаторы;
- Прочие системные приложения.

Комплекс встроенных средств защиты информации (КСЗ), является принадлежностью операционной среды Альт Линукс СПТ 6.0 и неотъемлемой частью ядра ОС и системных библиотек.

3.2.1. Операционная среда

Операционная среда изделия (ОСр) – совокупность пакетов программ и специальных приложений, предназначенных для обеспечения функционирования изделия и реализации пользовательских задач различного назначения.

Операционная среда изделия (ОСр), включает в свой состав операционную систему, системные приложения и комплекс встроенных средств защиты информации.

3.2.2. Операционная система

Операционная система (ОС) – совокупность программных средств, организующих согласованную работу ОСр с аппаратными устройствами компьютера (процессор, память, устройства ввода-вывода и т.д.).

Операционная система изделия (ОС) состоит из ядра ОС и системных библиотек.

3.2.3. Ядро ОС

Ядро ОС – программа (набор программ), выполняющая функции управления ОС и взаимодействия ОС с аппаратными средствами.

3.2.4. Системные библиотеки

Системные библиотеки – наборы программ (пакетов программ), выполняющие различные

функциональные задачи и предназначенные для их динамического подключения к работающим программам, которым необходимо выполнение этих задач.

3.2.5. Встроенные средства защиты информации

Встроенные средства защиты информации (КСЗ) – специальные пакеты программ ОСр, входящие в состав ядра ОС и системных библиотек, предназначенные для защиты ОСр от несанкционированного доступа к обрабатываемой (хранящейся) информации на ЭВМ.

3.2.6. Системные приложения

Системные приложения – это приложения (программы, набор программ), предназначенные для выполнения (оказания) системных услуг пользователю при решении им определенных функциональных задач в работе с операционной средой и обеспечивающие их выполнение.

Системные приложения включают в себя такие элементы как:

- Программные серверы;
- Средства разработки;
- Средства антивирусной защиты;
- Интерактивные рабочие среды;
- Прочие приложения.

3.2.7. Программные серверы

Программные серверы – специальные приложения, предназначенные для предоставления пользователю определенных услуг и обеспечивающие их выполнение.

3.2.8. WEB-серверы

Web-серверы – программы (набор программ), предназначенные для предоставления пользователю услуг доступа к глобальной сети Internet.

В состав изделия включен Web-сервер Apache версии 2.2.

3.2.9. Системы управления базами данных

Системы управления базами данных (СУБД) – приложения, предназначенные для работы с данными, представленными в виде набора записей, позволяющее осуществлять их поиск, обработку и хранение в виде специальных таблиц.

В состав ОС включены:

- СУБД MySQL;
- Набор GNU-драйверов доступа к базам данных;
- Драйвера доступа к базам данных ODBC.

3.2.10. Прочие серверные приложения и программы

К прочим серверным программам относятся программы, предоставляющие пользователю различные услуги по обработке, передаче, хранению информации (серверы протоколов, почтовые серверы, серверы приложений, серверы печати и прочие).

В состав изделия включены:

- Сетевой протокол DHCP (Dynamic Host Configuration Protocol);
- Протокол LDAP (Lightweight Directory Access Protocol);
- SMB-сервер (Сервер файлового обмена);
- Почтовый сервер Postfix;
- Прокси сервер squid;
- Серверы протоколов FTP, SFTP, SSHD;
- Сервер баз данных MySQL;
- Web-сервер Apache2;
- DNS-сервер;
- FreeNX-сервер.

3.2.11. Интерактивные рабочие среды

Интерактивные рабочие среды (ИРС) – программы (пакеты программ), предназначенные для работы пользователя в ОС и предоставляющие ему удобный интерфейс для общения с ней. Командные рабочие среды включают в свой состав командные интерпретаторы.

Командные интерпретаторы - специальные программы (терминалы), предназначенные для выполнения различных команд подаваемых пользователем при работе с ОС.

3.2.12. Графическая оболочка GNOME

Графическая оболочка состоит из набора различных программ и технологий, используемых для управления ОС и предоставляющие пользователю удобный графический интерфейс для работы с ней в виде графических оболочек и оконных менеджеров.

Графическая оболочка GNOME (GNU Network Object Model Environment) – аналог рабочего стола ОС MS Windows, содержащий множество стилей, эмулирующих внешний вид различных операционных систем, с возможностью создания собственных стилей.

3.2.13. Прочие системные приложения

Прочие системные приложения – приложения (программы), оказывающие пользователю дополнительные системные услуги при работе с ОС.

В состав ОС включены такие дополнительные системные приложения, как:

- Архиваторы;
- Приложения для управления RPM-пакетами;
- Приложения резервного копирования;
- Приложения мониторинга системы;
- Приложения для работы с файлами;
- Приложения для настройки системы;
- Настройка параметров загрузки;
- Настройка оборудования;
- Настройка сети.

3.2.14. Документация в составе

- HOWTOs;
- Электронные справочники (man).

3.3. Общая характеристика входной и выходной информации

Обмен информацией между ОС ALT Linux и внешними источниками осуществляется информационными сообщениями по локальной вычислительной сети или сети Internet с использованием протоколов TCP/IP, ICMP, UDP, FTP, HTTP, POP, SMTP, IMAP, SLIP, PPP, RIP, IPX и NetBIOS.

3.3.1. Протокол TCP/IP

Протокол TCP/IP (Transmission Control Protocol/Internet Protocol) – Протокол Управления Передачей/Интернет-протокол. На нём основано взаимодействие компьютера с другими в сети, в т.ч. и Internet.

Протокол TCP/IP состоит из двух протоколов:

– Протокол TCP – транспортный протокол, который обеспечивает гарантированную передачу данных по сети.

– Протокол IP – адресный протокол, который отвечает за адресацию всей сети. То есть, благодаря использованию протокола IP, каждое устройство в сети имеет свой индивидуальный адрес (IP-адрес).

3.3.2. Протокол ICMP

Протокол ICMP (Internet Control Message Protocol) – протокол межсетевых управляющих сообщений. С помощью этого протокола компьютеры и устройства в сети обмениваются друг с другом управляющей информацией.

3.3.3. Протокол UDP

Протокол UDP (User Datagram Protocol) – не ориентирован на соединение и не гарантирует доставку пакетов. Однако протокол UDP является более быстродействующим по сравнению с TCP/IP. По этому протоколу передаются небольшие объемы данных. Ответственность за доставку данных несёт сетевая программа.

3.3.4. Протокол FTP

Протокол FTP (File Transport Protocol) – протокол передачи файлов. Служит для обмена файлами меду компьютерами. Реализуется в архитектуре «клиент-сервер», где сервером является FTP-сервер (файловый сервер), а клиентом FTP-клиент. Последний выполняет подключение к FTP-серверу и позволяет выполнять команды по закачке или скачиванию файлов.

3.3.5. Протокол HTTP

Протокол HTTP (HyperText Transfer Protocol) – протокол обмена гипертекстовой информацией (документами HTML). Протокол HTTP используется HTTP-сервером. Клиентами для HTTP являются Web-браузеры.

3.3.6. Протоколы POP и SMTP

Протокол POP (Post Office Protocol) – протокол почтового отделения. Используется для получения электронной почты с почтовых серверов. Для передачи электронной почты служит протокол SMTP (Simple Mail Transfer Protocol) – протокол передачи сообщений электронной почты.

3.3.7. Протокол IMAP

Протокол IMAP также служит для чтения почты. Его отличие от протокола POP состоит в том,

что пользователь читает сообщения электронной почты, не загружая их на свой компьютер. Все сообщения хранятся на сервере. При удалении сообщения оно удаляется и с сервера.

3.3.8. Протокол SLIP

Протокол SLIP (Serial Line Internet Protocol) – протокол подключения к сети Internet по последовательной линии. Используется для установления связи с удаленными узлами через низкоскоростные последовательные интерфейсы.

3.3.9. Протокол PPP

Протокол PPP (Point-to-Point Protocol) – обеспечивает управление конфигурацией, обнаружение ошибок и повышенную безопасность при передаче данных на более высоком уровне, чем протокол SLIP.

3.3.10. Протокол RIP

Протокол RIP (Routing Information Protocol) – используется для маршрутизации пакетов в компьютерной сети.

3.3.11. Протокол IPX

Протокол IPX предназначен для передачи дейтограмм в системах, не ориентированных на соединение. Он обеспечивает связь между NetWare серверами и конечными станциями. Максимальный размер IPX-дейтограммы составляет 576 байт, из них 30 байта занимает заголовок. Предполагается, что сеть, через которую транспортируются эти дейтограммы, способна пересылать пакеты соответствующей длины. IPX-пакеты могут рассылаться широковещательно, для этого поле типа должно принять значение 0x14, адрес сети назначения должен соответствовать локальной сети, адрес узла назначения при этом принимает значение 0xFFFFFF.

3.3.12. Протокол NetBIOS

Протокол NetBIOS создан для работы в локальных сетях. Система NetBIOS предназначена для персональных ЭВМ типа IBM/PC в качестве интерфейса, независящего от фирмы-производителя. NetBIOS использует в качестве транспортных протоколов TCP и UDP.

Пакет NETBIOS поддерживает как режим сессий (работа через соединение), так и режим дейтограмм (без установления соединения). 16-ти символьные имена объектов в NetBIOS распределяются динамически. NetBIOS имеет собственную DNS, которая может взаимодействовать с системой DNS интернета.

Приложения могут через NetBIOS найти нужные им ресурсы, установить связь, послать или по-

лучить информацию.

4. ОПИСАНИЕ ЗАДАЧИ

4.1. Решаемые задачи

Операционная система ALT Linux решает следующие основные задачи:

- работа с файловой системой;
- контроль создания и удаления процессов;
- контроль распределения системных ресурсов;
- синхронизация процессов;
- организация межпроцессного взаимодействия;
- распределение оперативной памяти между прикладными задачами;
- очистка (обнуление) освобождаемых областей оперативной памяти ЭВМ;
- доступ к периферийным устройствам;
- буферизация данных;
- взаимодействие с драйверами устройств;
- выполнение администрирования;
- идентификация, проверка подлинности и контроль доступа субъектов;
- управление потоками информации;
- квотирование – разграничение дискового пространства;
- регистрация системных событий;
- сигнализация попыток нарушения защиты;
- обеспечение целостности программных средств и обрабатываемой информации;
- организация серверов.

4.1.1. Работа с файловой системой

4.1.1.1. Виртуальная файловая система VFS

VFS не ориентируется на какую-либо конкретную файловую систему, механизмы реализации файловой системы полностью скрыты как от пользователя, так и от приложений. В ОС нет системных вызовов, предназначенных для работы со специфическими типами файловой системы, а име-

ются абстрактные вызовы типа open, read, write и другие, которые имеют содержательное описание, обобщающее некоторым образом содержание этих операций в наиболее популярных типах файловых систем (например, s5, ufs, nfs и т.п.). VFS также предоставляет ядру возможность оперирования файловой системой, как с единым целым: операции монтирования и демонтирования, а также операции получения общих характеристик конкретной файловой системы (размера блока, количества свободных и занятых блоков и т.п.) в единой форме. Если конкретный тип файловой системы не поддерживает какую-то абстрактную операцию VFS, то файловая система должна вернуть ядру код возврата, извещающий об этом факте.

В VFS вся информация о файлах разделена на две части:

1) Не зависящую от типа файловой системы, которая хранится в специальной структуре ядра – структуре vnode;
2) Зависящую от типа файловой системы – структура inode, формат которой на уровне VFS не определен, а используется только ссылка на неё в структуре vnode.

Имя inode не означает, что эта структура совпадает со структурой индексного дескриптора inode файловой системы s5. Это имя используется для обозначения зависящей от типа файловой системы информации о файле, как дань традиции.

Виртуальная файловая система VFS поддерживает следующие типы файлов:

– обычные файлы;
– каталоги;
– специальные файлы;
– именованные конвейеры;
– символьные связи.

Содержательное описание обычных файлов, каталогов и специальных файлов и связей не отличается от их описания в файловой системе s5.

4.1.1.1.1. Символьные связи

Символьная связь – это файл данных, содержащий имя файла, с которым предполагается установить связь. «Предполагается» – потому, что символьная связь может быть создана даже с несуществующим файлом. При создании символьной связи образуется как новый вход в каталоге, так и новый индексный дескриптор inode. Кроме этого, резервируется отдельный блок данных для хранения полного имени файла, на который он ссылается.

Многие системные вызовы пользуются файлом символьных связей для поиска реального файла.

Связанные файлы не обязательно располагаются в той же файловой системе.

4.1.1.1.2. Именованные конвейеры

Конвейер – это средство обмена данными между процессами. Конвейер буферизует данные, поступающие на его вход, таким образом, что процесс, читающий данные на его выходе, получает их в порядке «первый пришел - первый вышел» (FIFO – First In First Out).

Именованные конвейеры позволяют обмениваться данными произвольной паре процессов, т.к. каждому такому конвейеру соответствует файл на диске. Никакие данные не связываются с файлом-конвейером, но все равно в каталоге содержится запись о нём, и он имеет индексный дескриптор.

4.1.1.1.3. Файлы, отображённые в памяти

Возможностью архитектуры виртуальной памяти ALT Linux является возможность отображать содержимое файла (или устройства) в виде последовательности байтов в виртуальное адресное пространство процесса. Это упрощает процедуру доступа процесса к данным.

VFS позволяет одновременно в единое дерево смонтировать несколько файловых систем различных типов, поддерживающих операцию монтирования.

4.1.1.2. Сетевая файловая система NFS

Одной из самых известных сетевых файловых систем является Network File System (NFS) фирмы Sun Microsystems. Основная идея NFS – позволить произвольному набору пользователей разделять общую файловую систему. Чаще всего все пользователи принадлежат одной локальной сети, но не обязательно. Можно выполнять NFS и на глобальной сети. Каждый NFS-сервер предоставляет один или более своих каталогов для доступа удалённым клиентам. Каталог объявляется доступным со всеми своими подкаталогами. Список каталогов, которые сервер передаёт, содержится в файле `/etc/exports`, так что эти каталоги экспортируются сразу автоматически при загрузке сервера. Клиенты получают доступ к экспортируемым каталогам путём монтирования.

При выполнении программ почти нет различий, расположен ли файл локально или на удалённом диске. Если два или более клиента одновременно смонтировали один и тот же каталог, то они могут связываться путем разделения файла.

Одной из целей NFS является поддержка неоднородных систем с клиентами и серверами, выполняющими различные ОС на различной аппаратуре. Это цель достигается двумя протоколами.

Первый NFS-протокол управляет монтированием. Клиент может послать полное имя каталога серверу и запросить разрешение на монтирование этого каталога на какое-либо место собственно-

го дерева каталогов. При этом серверу не указывается, в какое место будет монтироваться каталог сервера, так как ему это безразлично. Получив имя, сервер проверяет законность этого запроса и возвращает клиенту описатель файла, содержащий тип файловой системы, диск, номер дескриптора (inode) каталога, информацию безопасности. Операции чтения и записи файлов из монтируемых файловых систем используют этот описатель файла. Монтирование может выполняться автоматически с помощью командных файлов при загрузке. Существует другой вариант автоматического монтирования: при загрузке ОС на рабочей станции удалённая файловая система не монтируется, но при первом открытии удаленного файла ОС посылает запросы каждому серверу, и, после обнаружения этого файла, монтирует каталог того сервера, на котором этот файл расположен.

Второй NFS-протокол используется для доступа к удалённым файлам и каталогам. Клиенты могут послать запрос серверу для выполнения какого-либо действия над каталогом или операции чтения или записи файла. Кроме того, они могут запросить атрибуты файла, такие как тип, размер, время создания и модификации.

Использование NFS затрудняет блокировку файлов. Файл может быть открыт и заблокирован так, что другие процессы не имеют к нему доступа. Когда файл закрывается, блокировка снимается. В NFS блокирование не может быть связано с открытием файла, так как сервер не знает, какой файл открыт. Следовательно, NFS требует специальных дополнительных средств управления блокированием.

4.1.1.3. Файловая система ext2

Файловая система ext2 имеет следующую структуру:

- суперблок;
- описатель группы;
- карта блоков;
- карта информационных узлов;
- таблица информационных узлов;
- блоки данных.

Суперблок содержит информацию обо всей файловой системе. Он имеется в каждой группе блоков, но это всего лишь копия суперблока первой группы блоков: так достигается избыточность файловой системы.

Описатель (дескриптор) группы содержит информацию о группе блоков. Каждая имеет свой дескриптор группы.

Карты блоков и информационных узлов – это массивы битов, которые указывают на блоки или информационные узлы соответственно.

Таблица информационных узлов содержит информацию о выделенных для данной группы блоков информационных узлов.

Блоки данных – это блоки, содержащие реальные данные.

В операционной системе ALT Linux существует 4 типа файлов:

- файлы устройств;
- каталоги;
- обычные файлы;
- ссылки.

Файлы устройств представляют устройства компьютера (см. раздел 4.1.8).

Обычные файлы представляют собой обычные файлы с данными. Делятся на текстовые и двоичные.

Каталоги – это специальные файлы, содержащие информацию о других файлах (файлах устройств, обычных файлов и ссылок).

Ссылки позволяют хранить один и тот же файл, но под разными именами.

Максимальная длина имени файла составляет 255 символов. Имя может содержать любые символы кроме: / \ ? > < | “ *

Свойства файловой системы ext2:

- максимальный размер файловой системы – 4 Тбайт;
- максимальный размер файла – 2 Гбайт;
- максимальная длина имени файла – 255 символов;
- минимальный размер блока – 1024 байт;
- количество выделяемых индексных дескрипторов – 1 на 4096 байт раздела.

4.1.1.4. Файловая система ext3

Файловая система ext3 является журнализируемой файловой системой. При работе с ext3 можно выбрать один из трех режимов работы журнала:

- журнал (Journal);
- последовательный (Ordered);

– обратная запись (Writeback).

Режим Journal позволяет минимизировать потери при отключении питания, но является наиболее медленным. Данный режим подразумевает запись всех изменений метаданных файловой системы, а также информации о других изменениях файловой системы, в журнале.

При работе в режиме Ordered производится запись в журнал только сведений об изменении метаданных, причем запись в журнал производится перед изменением самих метаданных. Данный режим установлен по умолчанию.

Режим Writeback является самым быстрым. В этом режиме в журнал записываются только сведения об изменениях в файлах данных.

Операционная система ALT Linux позволяет производить следующие операции с файлами:

– создание и просмотр файла;

– копирование файла;

– переименование и перемещение файлов;

– удаление файлов;

– поиск файлов;

– изменение прав доступа к файлам.

Операционная система ALT Linux позволяет производить следующие операции с каталогами:

– просмотр содержимого каталога;

– вывод имени текущего каталога;

– создание и удаление каталога;

– смена каталога;

– изменение прав доступа к каталогу.

Для каждого файла в операционной системе Linux задаются права доступа.

В операционной системе ALT Linux существует специальный тип файлов – ссылки. Ссылки позволяют хранить один и тот же файл, но под разными именами. ALT Linux поддерживает два типа ссылок: жёсткие и символические.

Жёсткие ссылки привязываются к индексу файла. При изменении файла ссылки, автоматически изменяется и обычный файл. При удалении файла ссылки, обычный файл удаляется только, если на него нет больше ссылок. В противном случае удаляется только ссылка.

Символическая ссылка представляет собой файл, при обращении к которому система обращается к другому файлу. Символическая ссылка не имеет прав доступа, она наследует права доступа от файла, на который ссылается.

Если диск разбит на разделы, то на каждом разделе организуется отдельная файловая система с собственной структурой каталогов. Для пользователя файловая система представляет собой единое целое. В действительности, разные части файловой системы могут находиться на совершенно разных устройствах: разделах жёсткого диска, съёмных носителях и т.д.

Операционная система ALT Linux позволяет производить следующие операции с разделами:

- создание раздела;
- монтирование, размонтирование раздела;
- форматирование раздела;
- проверка файловой системы раздела.

4.1.1.5. Файловая система ReiserFS

ReiserFS – журналируемая файловая система.

ReiserFS использует специально оптимизированные сбалансированные деревья (одно на файловую систему) для организации всех данных файловой системы. Она, подобно большинству других файловых систем нового поколения, динамически ассигнует информационные узлы (inodes) вместо их статического набора, образующегося при создании традиционной файловой системы.

ReiserFS также имеет ряд особенностей, нацеленных специально для работы с маленькими файлами. ReiserFS не связана ограничением в ассигновании памяти для файла в целом числе 1-2-4 KB блоков. По необходимости для файла может ассигноваться точный размер. ReiserFS также включает некоторые виды специальной оптимизации файловых «хвостов» для хранения конечных частей файлов, меньших, чем логический блок файловой системы. Для увеличения скорости, ReiserFS способен хранить содержимое файлов непосредственно внутри дерева, а не в виде указателя на дисковый блок и упаковать хвосты (tail) файлов, экономя дисковое пространство.

Следует иметь в виду, что упаковка хвостов требует дополнительной работы, так как при изменении размеров файлов необходима «переупаковка». По этой причине в ReiserFS упаковка хвоста может отключаться, позволяя администратору выбрать между скоростью и эффективностью использования дискового пространства.

4.1.1.6. Файловая система XFS

XFS – высокопроизводительная журналируемая файловая система. Особенностями данной

файловой системы являются:

- XFS – 64-битная файловая система;
- журналирование только метаданных;
- изменение размера «на лету» (только увеличение);
- размещение в нескольких разных линейных областях — т. н. «allocation groups» (увеличивает производительность путём выравнивания активности запросов к разным дискам на RAID-массивах типа «stripe»);
- дефрагментация «на лету»;
- API ввода/вывода реального времени (для приложений жёсткого или мягкого реального времени, например, для работы с потоковым видео);
- запись на диск производится только при нехватке памяти. Это позволяет уменьшить фрагментацию, а также снизить активность запросов к диску;
- интерфейс (DMAPI) для поддержки иерархического управления хранением;
- инструменты резервного копирования и восстановления.

Недостатки файловой системы XFS:

- невозможно уменьшить размер существующей файловой системы;
- версии загрузчика GRUB до 0.91 не поддерживают XFS;
- восстановление удалённых файлов в XFS практически невозможно.

4.1.1.7. Файловая система JFS

JFS – это журналируемая файловая система. Её внутренняя структура близка к структуре ReiserFS.

4.1.1.7.1. Дисковый раздел

Файловая система JFS создается на разделе (partition). Дисковый раздел с точки зрения JFS имеет следующие параметры:

- Фиксированный размер блока, PART_BSize, который может принимать значения 512, 1024, 2048 или 4096 байт. Этот параметр определяет размер наименьшего элемента обмена данными с разделом. Он соответствует размеру физического сектора того диска, на котором расположен раздел, и чаще всего равен 512 байт.
- Размер раздела, величину PART_NBlocks, содержащую количество дисковых блоков на

разделе.

– Абстрактное адресное пространство: [0,(PART_NBlocks1)] дисковых блоков.

4.1.1.7.2. Логический том

Дисковый раздел представлен в JFS логическим томом, а совокупность файлов и каталогов именуется файлсетом. Логический том состоит из следующих частей:

– 32 Kb неиспользуемого дискового пространства в начале.

– Первичный и вторичный суперблоки, содержащие глобальную информацию о данном томе. Вторичный суперблок есть точная копия первичного, положение обоих на диске фиксировано.

– Два экземпляра таблицы inodes тома, содержащей inodes, описывающие глобальные управляющие структуры тома. Логически это просто массивы inodes. т.к. том не имеет структуры каталога, объекты, описываемые этими inodes, не видны в пространстве имён какого-либо файлсета.

– Две копии карты размещения inodes, описывающей таблицу inodes тома.

– Карта размещения блоков описывает структуры, управляющие выделением и освобождением блоков тома.

– Рабочее пространство для fsck, необходимое для отслеживания выделения блоков тома. Оно необходимо потому, что JFS поддерживает очень большие тома, и данные о выделенных/свободных блоках могут не поместится в памяти. Это пространство описывается суперблоком. Один байт необходим для описания 8 блоков (по 1 биту на блок). Рабочее пространство fsck всегда расположено в конце тома.

– Встроенный журнал, обеспечивающий место для журналирования изменений метаданных тома. Описывается суперблоком и всегда расположено за рабочим пространством fsck.

Состав таблицы inodes тома таков:

– Inode 0 зарезервирован.

– Inode 1 описывает все дисковые блоки тома, включая карту inodes. Это рекурсивное представление, при котором сам inode расположен в файле, который он описывает. Очевидная трудность рекурсивного представления преодолевается здесь путём размещения по крайней мере первого inode тома в фиксированном месте (через 4 Kb после первичного суперблока тома). Таким образом, JFS может просто отыскать inode 1, а от него и

остальные inodes таблицы – следуя по B+ дереву в inode 1. Для репликации таблицы inodes тома JFS также должна находить копию inode 1 – для достижения оставшейся вторичной таблицы. Для этих целей в суперблоке есть поле, содержащее дескриптор экстента, описывающий положение inode 1 вторичной таблицы.

- Inode 2 описывает карту размещения блоков тома.
- Inode 3 описывает встроенный журнал (когда ФС смонтирована). Дисковая копия этого inode не указывает на какие-либо данные, его поля заполняются только в inmemory образе.
- Inode 4 описывает файл плохих блоков, найденных при форматировании тома.
- Inodes с 5-ого по 15-ый зарезервированы.
- Начиная с 16-ого, существует по 1 inode на каждый файлсет. Эти inodes описывают управляющие структуры, представляющие набор файлов. С добавлением новых наборов файлов к тому таблица inodes тома может расти.

4.1.2. Контроль создания и удаления процессов

4.1.2.1. Архитектура процессов

Всякая выполняющаяся в ALT Linux программа называется процессом. ALT Linux как многозадачная система характеризуется тем, что одновременно может выполняться множество процессов, принадлежащих одному или нескольким пользователям. Каждому работающему процессу система присваивает уникальный номер процесса. Каждый процесс выполняется в собственном виртуальном адресном пространстве.

Операционная система управляет образом процесса или сегментами кода и данных, определяющих среду выполнения, а не самим процессом. Сегмент кода содержит реальные инструкции центральному процессору, в которые входят как строки, написанные и скомпилированные пользователем, так и код, сгенерированный системой, который обеспечивает взаимодействие между программой и операционной системой. Данные, связанные с процессом, тоже являются частью образа процесса, некоторые из которых хранятся в регистрах. Для оперативного хранения рабочих данных существует динамическая область памяти. Эта память выделяется динамически и использование её от процесса к процессу меняется.

Автоматически, при запуске программы, переменные размещаются в стеке (стек служит хранилищем для временного хранения переменных и адресов возврата из процедур). Обычно при выполнении или в режиме ожидания выполнения процессы находятся в оперативной памяти компьютера. Довольно большая её часть резервируется ядром операционной системы, и только к остав-

шейся её части могут получить доступ пользователи.

Одновременно в оперативной памяти может находится несколько процессов. Память, используемая процессором, разбивается на сегменты, называемые страницами (см. раздел 4.1.6).

Процессы разделяются на функционирующие на уровне ядра операционной системы и функционирующие вне ядра операционной системы (kernel-space и user-space). Процессы, функционирующие на уровне ядра, запускаются самим ядром ОС, либо в виде подгружаемых модулей ядра. Процессы, функционирующие в системном окружении, запускаются стандартным для приложений ОС методом.

Для каждого процесса создается свой блок управления, который помещается в системную таблицу процессов, находящихся в ядре. Эта таблица представляет собой массив структур блоков управления процессами. В каждом блоке содержатся следующие данные:

- слово состояния процесса;
- приоритет;
- величина кванта времени, выделенного системным планировщиком;
- степень использования системным процессором;
- признак диспетчеризации;
- идентификатор пользователя, которому принадлежит процесс;
- эффективный идентификатор пользователя;
- реальный и эффективный идентификаторы группы;
- группа процесса;
- идентификатор процесса и идентификатор родительского процесса;
- размер образа, размещаемого в области подкачки;
- размер сегментов кода и данных;
- массив сигналов, ожидающих обработки.

Чтобы система функционировала должным образом, ядро отслеживает все эти данные.

Для управлении процессами операционная система использует два основных типа информационных структур: дескриптор процесса и контекст процесса.

Дескриптор процесса содержит такую информацию о процессе, которая необходима ядру в течение всего жизненного цикла процесса, независимо от того, находится ли он в активном или пассивном состоянии, находится ли образ процесса в оперативной памяти или выгружен на диск.

Дескрипторы отдельных процессов объединены в список, образующий таблицу процессов. Память для таблицы процессов отводится динамически в области ядра. На основании информации, содержащейся в таблице процессов, операционная система осуществляет планирование и синхронизацию процессов. В дескрипторе прямо или косвенно (через указатели на связанные с ним структуры) содержится информация о состоянии процесса, расположении образа процесса в оперативной памяти и на диске, о значении отдельных составляющих приоритета, а также его итоговое значение – глобальный приоритет, идентификатор пользователя, создавшего процесс, информация о родственных процессах, о событиях, осуществления которых ожидает данный процесс и некоторая другая информация.

Контекст процесса содержит менее оперативную, но более объёмную часть информации о процессе, необходимую для возобновления выполнения процесса с прерванного места: содержимое регистров процессора, коды ошибок выполняемых процессором системных вызовов, информацию о всех открытых данным процессом файлов и незавершённых операциях ввода-вывода и другие данные, характеризующие состояние вычислительной среды в момент прерывания. Контекст, так же как и дескриптор процесса, доступен только программам ядра, то есть находится в виртуальном адресном пространстве операционной системы, однако он хранится не в области ядра, а непосредственно примыкает к образу процесса и перемещается вместе с ним, если это необходимо, из оперативной памяти на диск.

4.1.2.2. Создание процессов

Процесс порождается с помощью системного вызова. При создании нового процесса:

1) Выделяется память для описателя нового процесса в таблице процессов.
2) Назначается идентификатор процесса PID.
3) Создаётся логическая копия процесса, который выполняет полное копирование содержимого виртуальной памяти родительского процесса, копирование составляющих ядерного статического и динамического контекстов процесса-предка.
4) Увеличиваются счётчики открытия файлов (порождённый процесс наследует все открытые файлы родительского процесса).
5) Возвращается PID в точку возврата из системного вызова в родительском процессе и 0 – в процессе-потомке.

ALT Linux ассоциирует с каждым процессом два номера приоритета: реальный приоритет процесса, который динамически вычисляется операционной системой; nice-номер или запрошенный номер приоритета выполнения процесса, который вычисляется операционной системой с учётом

уже полученного номера. Последний номер может быть задан владельцем или суперпользователем для воздействия на реальный номер приоритета выполнения (PRI). Реальный диапазон приоритета процесса – от -20 до 20, причём чем меньше число, тем выше приоритет.

Существуют процессы, которые запускаются во время начальной загрузки самой системой. Можно контролировать, какие процессы выполняются во время загрузки, изменяя конфигурационные файлы и сценарии.

4.1.2.3. Завершение процесса

Для завершения процесса используется системный вызов, при котором освобождаются все используемые ресурсы, такие как память и структуры таблиц ядра. Кроме того, завершаются и процесс-потомки, порождённые данным процессом.

Затем из памяти удаляются сегменты кода и данных. И, наконец, родительский процесс должен очистить все ресурсы, занимаемые дочерними процессами.

Если родительский процесс по какой-то причине завершится раньше дочернего, последний становится «сиротой» (orphaned process). Такие процессы–«сироты» становятся дочерними процессами программы init, выполняющейся в процессе с номером 1, которая и принимает сигнал об их завершении.

4.1.2.4. Планирование процессов

В операционной системе ALT Linux реализована вытесняющая многозадачность, основанная на использовании приоритетов и квантования.

Все процессы разбиты на несколько групп, называемых классами приоритетов. Каждая группа имеет свои характеристики планирования процессов.

Созданный процесс наследует характеристики планирования процесса-родителя, которые включают класс приоритета и величину приоритета в этом классе. Процесс остаётся в данном классе до тех пор, пока не будет выполнен системный вызов, изменяющий его класс. В настоящее время имеется три приоритетных класса: класс реального времени, класс системных процессов и класс процессов разделения времени. Приоритетность (привилегии) процесса тем выше, чем больше число, выражающее приоритет.

Процессы системного класса используют стратегию фиксированных приоритетов. Системный класс зарезервирован для процессов ядра. Уровень приоритета процессу назначается ядром и никогда не изменяется. Заметим, что пользовательский процесс, перешедший в системную фазу, не переходит при этом в системный класс приоритетов.

Процессы реального времени также используют стратегию фиксированных приоритетов, но

пользователь может их изменять. Так как при наличии готовых к выполнению процессов реального времени другие процессы не рассматриваются, то процессы реального времени надо тщательно проектировать, чтобы они не захватывали процессор на слишком долгое время.

Характеристики планирования процессов реального времени включают две величины: уровень глобального приоритета и квант времени. Для каждого уровня приоритета имеется по умолчанию своя величина кванта времени. Процессу разрешается захватывать процессор на указанный квант времени, а по его истечении планировщик снимает процесс с выполнения.

Состав класса процессов разделения времени наиболее неопределённый и часто меняющийся, в отличие от системных процессов и процессов реального времени. Для справедливого распределения времени процессора между процессами, в этом классе используется стратегия динамических приоритетов, которая адаптируется к операционным характеристикам процесса.

Величина приоритета, назначаемого процессам разделения времени, вычисляется пропорционально значениям двух составляющих: пользовательской части и системной части. Пользовательская часть приоритета может быть изменена суперпользователем и владельцем процесса, но в последнем случае только в сторону его снижения.

Системная составляющая позволяет планировщику управлять процессами в зависимости от того, как долго они используют процессор, не уходя в состояние ожидания. Тем процессам, которые потребляют большие периоды времени без ухода в состояние ожидания, приоритет снижается, а тем процессам, которые часто уходят в состояние ожидания после короткого периода использования процессора, приоритет повышается. Но процессам с низким приоритетом даются большие кванты времени, чем процессам с высокими приоритетами. Таким образом, хотя низкоприоритетный процесс и не работает так часто, как высокоприоритетный, но зато, когда он наконец выбирается на выполнение, ему отводится больше времени.

Планировщик использует следующие характеристики для процессов разделения времени:

1) Величина глобального приоритета.
2) Количество тиков системных часов, которые отводятся процессу до его вытеснения.
3) Системная часть приоритета, назначаемая процессу при истечении его кванта времени.
4) Системная составляющая приоритета, назначаемая процессу после выхода его из состояния ожидания. Ожидающим процессам даётся высокий приоритет, так что они быстро получают доступ к процессору после освобождения ресурса.
5) Максимальное число секунд, которое разрешается потреблять процессу; если этот квант времени истекает до кванта п.2, то, следовательно, считается, что процесс ведёт себя кор-

ректно, и ему назначается более высокий приоритет.

6) Величина системной части приоритета, назначаемая процессу, если истекает количество секунд, определяемых квантом п.5.

Для процессов разделения времени в дескрипторе процесса имеется указатель на структуру, специфическую для данного класса процесса. Эта структура состоит из полей, используемых для вычисления глобального приоритета, и содержит следующую информацию:

1) Число тиков, остающихся в кванте процесса.
2) Системная часть приоритета процесса.
3) Верхний предел и текущее значение пользовательской части приоритета. Эти две переменные могут модифицироваться пользователем.
4) Величина, используемая для обратной совместимости с системным вызовом nice. Она содержит текущее значение величины nice, которая влияет на результирующую величину приоритета. Чем выше эта величина, тем меньше приоритет.

4.1.2.5. Нити

Каждая нить представляет собой независимо выполняющийся поток управления со своим счётчиком команд, регистровым контекстом и стеком. Понятия процесса и нити очень тесно связаны. Основные отличия процесса от нити заключаются в том, что, каждому процессу соответствует своя независимая от других область памяти, таблица открытых файлов, текущая директория и прочая информация уровня ядра. Нити же не связаны непосредственно с этими сущностями. У всех нитей принадлежащих данному процессу всё выше перечисленное общее, поскольку принадлежит этому процессу. Кроме того, процесс всегда является сущностью уровня ядра, то есть ядро знает о его существовании, в то время как нити зачастую является сущностями уровня пользователя и ядро может ничего не знать о ней. В подобных реализациях все данные о нити хранятся в пользовательской области памяти, и соответственно такие процедуры как порождение или переключение между нитями не требуют обращения к ядру и занимают на порядок меньше времени.

Нить за время своего существования пребывает в следующих состояниях:

1) Готова (Ready). Нить готова к выполнению, но ожидает процессора. Возможно она только что была создана, была вытеснена с процессора другой нитью, или только что была разблокирована (вышла из соответствующего состояния).
2) Выполняется (Running). Нить сейчас выполняется. На многопроцессорной машине может быть несколько нитей в таком состоянии.

3) Заблокирована (Blocked). Нить не может выполняться, так как ожидает чего-либо (окончания операции ввода-вывода, сигнала от условной переменной, получения mutex и т.п.)

4) Завершена (Terminated). Нить была завершена.

Нити могут создаваться системой или могут создаваться при помощи явных вызовов пользовательским процессом. Однако любая создаваемая нить начинает свою жизнь в состоянии «готова». После чего в зависимости от политики планирования системы она может либо сразу перейти в состояние «выполняется» либо перейти в него через некоторое время. Выполняющаяся нить, скорее всего, рано или поздно либо перейдёт в состояние «заблокирована», вызвав операцию ожидающую чего-то, например, окончания ввода-вывода, прихода сигнала или поднятия семафора, либо перейдёт в состояние «готова» будучи снята с процессора или более высокоприоритетной нитью или просто потому что исчерпала свой квант времени.

Заблокированная нить, дождавшись события которого она ожидала, переходит в состояние «готова». При этом, конечно в случае если есть такая возможность, она сразу перейдёт в состояние выполнения.

Наконец выполняющаяся нить может завершиться тем или иным способом. Например, в следствие возврата из функции нити, или вследствие насильственного прерывания её выполнения. При этом, если нить была отсоединена, то она сразу освобождает все связанные с ней ресурсы и перестаёт существовать.

4.1.3. Контроль распределения системных ресурсов

Процесс, запускаемый в ALT Linux, не может выполнить свою задачу без использования системных ресурсов, таких как память, порты ввода/вывода, память ввода/вывода, линии прерывания, а также, каналы DMA.

Система управления ресурсами, реализованная в ALT Linux, может управлять произвольными ресурсами в единой иерархической манере. Глобальные ресурсы системы (например, порты ввода/вывода) могут быть подразделены на подмножества – например, относящиеся к какому-либо слоту аппаратной шины. Определённые драйверы, также, при желании, могут подразделять захватываемые ресурсы на основе своей логической структуры.

Область памяти, принадлежащая периферийному устройству, называется памятью ввода/вывода, для того чтобы отличать её от системного ОЗУ (RAM), называемую просто памятью. Чтение и запись портов и памяти ввода/вывода – работа драйвера. Порты и память ввода/вывода объединены общим названием – регион (или область) ввода/вывода.

В ALT Linux реализован механизм запроса/высвобождения регионов ввода/вывода главным об-

разом для предотвращения коллизий между различными устройствами. Этот механизм представляет программную абстракцию и не распространяется на аппаратные возможности.

Информация о зарегистрированных ресурсах доступна в текстовой форме в файлах `/proc/ioports` и `/proc/iomem`. Каждая строка данного файла отображает в шестнадцатеричном виде диапазон портов связанных с драйвером или владельцем устройства.

4.1.3.1. Порты

Файл `/proc/ioports` может быть использован для избежания коллизий портов при добавлении в систему нового устройства. Особенно это удобно при ручной настройке устанавливаемого оборудования путем переключения перемычек (jumpers – джамперов). В этом случае пользователь может легко посмотреть список используемых портов и выбрать свободный диапазон для устанавливаемого устройства. И хотя большинство современных устройств не используют перемычек ручной настройки вообще, тем не менее, они её используются при изготовлении мелкосерийных компонентов.

С файлом `/proc/ioports` связана структура данных, доступная программным путём. Поэтому, когда драйвер устройства производит инициализацию он может узнать занятый диапазон портов ввода/вывода. Значит, при необходимости просканировать порты в поисках нового устройства, драйвер в состоянии избежать ситуации записи в порты, занятые чужими устройствами.

4.1.3.2. Память ввода/вывода

Информация о памяти ввода/вывода доступна через файл `/proc/iomem`. Значения диапазонов адресов показаны в шестнадцатеричной записи. Для каждого диапазона адресов показан его владелец.

Регистрация доступа к памяти ввода/вывода аналогична регистрации портов ввода/вывода и построена в ядре на том же самом механизме.

4.1.3.3. Процессорное время

Операционная система предоставляет программе некоторый интервал процессорного времени. Когда программа переходит в режим ожидания какого-либо события (например, сигнала) или освобождает процессор, операционная система передаёт управление другой программе. Распределяя время центрального процессора, операционная система распределяет его не между программами, а между потоками. Потоки – это наборы команд, имеющие возможность получать время процессора. Время процессора выделяется квантами. Квант – это минимальное время, на протяжении которого поток может использовать процессор.

4.1.4. Синхронизация процессов

К средствам синхронизации процессов относятся: мьютексы (mutex) и семафоры (semaphore).

Мьютекс – объект для организации взаимоисключения. Один мьютекс может быть «захвачен» одним процессом (потоком или нитью). При этом любой другой процесс не сможет выполнить «захват» этого же мьютекса до тех пор, пока захвативший его процесс не освободит объект. До момента освобождения мьютекса все остальные процессы, которые пытаются его «захватить», приостанавливаются.

Семафоры представляют собой средство передачи флагов от одного процесса к другому. Используя семафор, процесс может сообщить, что он находится в определённом состоянии. Любой другой процесс в системе может отыскать этот флаг и выполнить необходимые действия.

4.1.5. Организация межпроцессного взаимодействия

К средствам для организации межпроцессного взаимодействия относятся: сокеты, сигналы, коммуникационные и именованные каналы, очереди сообщений и совместно используемая память.

Самым распространённым средством взаимодействия процессов являются **сокеты** (sockets). Программы подключаются к сокету и выдают запрос на привязку к нужному адресу. Затем данные передаются от одного сокета к другому в соответствии с указанным адресом.

Система ALT Linux имеют способ пересылки процессам различных **сигналов**. Сигнал – это способ информирования процесса ядром о происшествии какого-то события. Сигнал является исключением, которое обычно используется для приказания процессу выполнить что-то, отличное от обычного действия. Например, если нужно уничтожить процесс, можно послать ему сигнал о прекращении работы.

Каждый процесс реагирует на сигналы и может установить собственную реакцию на сигналы, производимые операционной системой. Сигнал информирует другой процесс о возникновении определенных условий внутри текущего процесса, требующих реакции текущего процесса. Многие программы обработки сигналов для анализа возникшей проблемы выводят дамп памяти.

Когда процесс порождает новый процесс, между двумя процессами открывается **коммуникационный канал**. Другим типом каналов являются **именованные каналы**. При их использовании с управляющей структурой в ядре связывается специальный каталог, через который два автономных процесса могут обмениваться данными. При этом, каждый процесс должен открыть канал в виде обычных файлов (один – для чтения, другой – для записи). Затем операции ввода/вывода выполняются обычным образом.

Очередь сообщений представляет собой механизм, когда один процесс предоставляет блок

данных с установленными флагами, а другой процесс разыскивает блок данных, флаги которого установлены в требуемых значениях.

Совместно используемая память позволяет процессам получить доступ к одной и той же области физической памяти.

4.1.6. Распределение оперативной памяти между прикладными задачами

В основе организации памяти лежат страницы памяти. Причём одна страница может находиться в разных списках, например и в списке страниц в страничном кеше и в списке страниц, относящихся к отображённому в память файлу (inode).

Все страницы адресуются глобальным указателем на начало карты памяти. Физический адрес страницы памяти вычисляется по следующему алгоритму:

[смещение_страницы + размер_страницы * (адрес_страницы_в_карте – начало_карты)]

Страницы делятся на свободные непрерывные области размера [2^X * размер_страницы].

Страница выделяется ядром ОС ALT Linux. Оно выделяет страницы, составляющие область размера [размер_страницы * ($2^{\text{кратность}}$)]. Делается это следующим образом. Ищется область соответствующего размера или больше. Если есть только область большего размера, то она делится на несколько маленьких и берется нужный кусок. Если свободных страниц недостаточно, то некоторые будут сброшены в область подкачки и процесс выделения начнется снова.

Процесс ALT Linux работает с виртуальными адресами, а не с физическими. Преобразование происходит посредством вычислений, используя таблицы дескрипторов, и каталоги таблиц. ALT Linux поддерживает 3 уровня таблиц:

- каталог таблиц первого уровня (PGD – Page Table Directory);
- каталог таблиц второго уровня (PMD – Medium Page Table Directory);
- таблица дескрипторов (PTE – Page Table Entry).

Преобразование виртуального адреса в физический происходит соответственно в 3 этапа. Берётся указатель PGD, имеющийся в структуре описывающий каждый процесс, преобразуется в указатель записи PMD, а последний преобразуется в указатель в таблице дескрипторов PTE. И, наконец, к реальному адресу, указывающему на начало страницы прибавляют смещение от её начала.

Все данные об используемой процессом памяти хранятся ядром ОС в специальных структурах.

При любом открытии файла, он сразу же отображается в память (его часть, дочитанная до размера страницы; например, для процессоров Intel при чтении 10 байт будут прочитаны 4096) и до-

бавляется в страничный кэш. Реальный же запрос на отображение файла только возвращает адрес на уже кэшированные страницы.

На уровне процесса работа может вестись как со страницами напрямую, так и через специальные структуры ядра.

4.1.7. Очистка (обнуление) освобождаемых областей оперативной памяти ЭВМ

КСЗ осуществляет очистку оперативной памяти (RAM-памяти), предоставляемой пользователю и его процессам. Очистка производится записью нулей или маскирующей информации в память при её назначении пользователю или освобождении.

Страница памяти освобождается ядром ОС ALT Linux. Оно высвобождает страницы, начинающиеся с указанной, размера [размер_страницы * (2 $^{\text{кратность}}$)]. Область возвращается в массив свободных областей в соответствующую позицию и после этого происходит попытка объединить несколько областей для создания одной большего размера.

Отсутствие страницы в памяти обрабатываются ядром особо. Страница может или вообще отсутствовать или находиться в области подкачки.

Доступ субъекта к остаточной HDD-информации может быть ограничен через механизм «безопасного удаления» файлов (специальный атрибут файла, указывающий на необходимость перезаписи физической области носителя диска после удаления файла). Еще одним способом является использование команды shred, обеспечивающей безопасное удаление файлов.

4.1.8. Доступ к периферийным устройствам

В файловой системе ОС ALT Linux каждое поддерживаемое устройство представляется файлом устройства. При выполнении операций чтения или записи с подобным файлом происходит обмен данными с устройством, на которое указывает этот файл. Такой способ доступа к устройствам позволяет не использовать специальные программы, а также специальные методы программирования, такие как работа с прерываниями. Файлы устройств находятся в каталоге `/dev`.

В операционной системе ALT Linux принято использовать стандартные имена устройств:

- ttyN – консоль;
- mouse – мышь;
- audio – звуковая карта;
- modem – модем;
- ttySN – последовательный порт;

- lpN – параллельный порт;
- cuaN – могут обозначать последовательные порты;
- sdxN – жесткий диск;
- fd0 – первый дисковод для гибких дисков;
- stN – стример с интерфейсом SCSI;
- nrtfN – стример с интерфейсом FDC;
- mdN – массив RAID;
- ethN – сетевая плата;
- null – пустое устройство.

Здесь N – номер устройства, например, tty1 – первая консоль.

4.1.9. Буферизация данных (кэширование)

Любой запрос на ввод-вывод к блок-ориентированному устройству преобразуется в запрос к подсистеме буферизации, которая представляет собой буферный пул и комплекс программ управления этим пулом.

Буферный пул состоит из буферов, находящихся в области ядра. Размер отдельного буфера равен размеру блока данных на диске.

С каждым буфером связана специальная структура – заголовок буфера, в котором содержится следующая информация:

- данные о состоянии буфера:
  a) занят/свободен;
  b) чтение/запись;
  c) признак отложенной записи;
  d) ошибка ввода-вывода.
- данные об устройстве - источнике информации, находящейся в этом буфере:
  a) тип устройства;
  b) номер устройства;
  c) номер блока на устройстве.
- адрес буфера.

– ссылка на следующий буфер в очереди свободных буферов, назначенных для ввода-вывода какому-либо устройству.

Алгоритм выполнения запроса к подсистеме буферизации реализуется набором следующих основных функций.

Функция синхронной записи. В результате выполнения данной функции немедленно инициируется физический обмен с внешним устройством. Процесс, выдавший запрос, ожидает результат выполнения операции ввода-вывода. В данном случае в процессе может быть предусмотрена собственная реакция на ошибочную ситуацию. Такой тип записи используется тогда, когда необходима гарантия правильного завершения операции ввода-вывода.

Функция асинхронной записи. При таком типе записи также немедленно инициируется физический обмен с устройством, однако завершения операции ввода-вывода процесс не дожидается. В этом случае возможные ошибки ввода-вывода не могут быть переданы в процесс, выдавший запрос. Такая операция записи целесообразна при поточной обработке файлов, когда ожидание завершения операции ввода-вывода не обязательно, но есть уверенность в повторении этой операции.

Функция отложенной записи. При этом передача данных из системного буфера не производится, а в заголовке буфера делается отметка о том, что буфер заполнен и может быть выгружен, если потребуется освободить буфер.

Функции получения блока данных. Каждая из этих функций ищет в буферном пуле буфер, содержащий указанный блок данных. Если такого блока в буферном пуле нет, то осуществляется поиск любого свободного буфера (при этом возможна выгрузка на диск буфера, содержащего в заголовке признак отложенной записи), либо организуется его загрузка в какой-нибудь свободный буфер. Если свободных буферов нет, то также производится выгрузка буфера с отложенной записью.

За счёт отложенной записи в системном буферном пуле задерживается некоторое число блоков данных. При возникновении запроса к внешней памяти просматривается содержимое буферного пула. При этом вероятность обнаружения данных в системном пуле достаточно велика. Это обусловлено объективными свойствами пространственной и временной локальности данных. В соответствии с описанным алгоритмом буферизации, в системном буферном пуле оседает наиболее часто используемая информация. Таким образом, система буферизации выполняет роль кэш-памяти по отношению к диску. Кэширование диска уменьшает среднее время доступа к данным на диске, однако при этом снижается надёжность файловой системы, так как в случае внезапной потери питания или отказа диска может произойти потеря блоков, содержащихся в системном буфере. Этот недостаток частично компенсируется регулярной (каждую секунду) принудительной записью всех

блоков из системной области на диск.

4.1.10. Взаимодействие с драйверами устройств

Драйвер – это совокупность программ (секций), предназначенная для управления передачей данных между внешним устройством и оперативной памятью.

Связь ядра системы с драйверами обеспечивается с помощью двух системных таблиц:

- таблица блок-ориентированных устройств;
- таблица байт-ориентированных устройств.

Для связи используется следующая информация из индексных дескрипторов специальных файлов:

- класс устройства (байт-ориентированное или блок-ориентированное);
- тип устройства (лента, гибкий диск, жёсткий диск, устройство печати, дисплей, канал связи и т.д.);
- номер устройства.

Класс устройства определяет выбор таблицы блок- или байт-ориентированных устройств. Эти таблицы содержат адреса программных секций драйверов, причем одна строка таблицы соответствует одному драйверу. Тип устройства определяет выбор драйвера. Типы устройств пронумерованы, т.е. тип определяет номер строки выбранной таблицы. Номер устройства передаётся драйверу в качестве параметра, так как в ALT Linux драйверы спроектированы в расчете на обслуживание нескольких устройств одного типа.

Такая организация логической связи между ядром и драйверами позволяет легко настраивать систему на новую конфигурацию внешних устройств путем модификации таблиц блок- и байт-ориентированных устройств.

Драйвер байт-ориентированного устройства в общем случае состоит из секции открытия, чтения и записи файлов, а также секции управления режимом работы устройства. В зависимости от типа устройства некоторые секции могут отсутствовать. Это определённым образом отражено в таблице байт-ориентированных устройств. Секции записи и чтения обычно используются совместно с модулями обработки прерываний ввода-вывода от соответствующих устройств.

Драйвер байт-ориентированного устройства взаимодействует с модулем обработки прерываний. Секция записи осуществляет передачу байтов из рабочей области программы, выдавшей запрос на обмен, в системный буфер, организованный в виде очереди байтов. Передача байтов идет до тех пор, пока системный буфер не заполнится до некоторого, заранее определенного в драйвере, уров-

ня. В результате секция записи драйвера приостанавливается.

Модуль обработки прерываний работает асинхронно секции записи. Он вызывается в моменты времени, определяемые готовностью устройства принять следующий байт. Если при очередном прерывании оказывается, что очередь байтов уменьшилась до определённой нижней границы, то модуль обработки прерываний активизирует секцию записи драйвера.

Аналогично организована работа при чтении данных с устройства.

Драйвер блок-ориентированного устройства состоит в общем случае из секций открытия и закрытия файлов, а также секции стратегии. Кроме адресов этих секций, в таблице блок-ориентированных устройств указаны адреса так называемых таблиц устройств. Эти таблицы содержат информацию о состоянии устройства – занято или свободно, указатели на буфера, для которых активизированы операции обмена с данным устройством, а также указатели на цепочку буферов, в которых находятся блоки данных, предназначенные для обмена с данным устройством.

После запуска устройства управление возвращается процессу, выдавшему запрос к драйверу.

Об окончании ввода-вывода каждого блока устройство оповещает операционную систему сигналом прерывания. Первое слово вектора прерываний данного устройства содержит адрес секции драйвера – модуля обработки прерываний. Модуль обработки прерываний проводит анализ правильности выполнения ввода-вывода. Если зафиксирована ошибка, то несколько раз повторяется запуск этой же операции, после чего драйвер переходит к вводу-выводу следующего блока данных из очереди к устройству.

Драйвера составляют часть ядра, а не исполняются им как процессы Однако для установки драйвера не требуется перекомпилировать всё ядро и перезагружать систему, чтобы его задействовать. Драйверы исполняются в виде загружаемых модулей ядра. Такой модуль представляет собой файл с кодом, который можно при необходимости загружать и выгружать. Это даёт возможность загружать и выгружать драйвера без нужды постоянно перезагружать систему.

4.1.11. Аутентификация и идентификация, проверка подлинности и контроль доступа субъектов

4.1.11.1. Учётные записи пользователей

Для предотвращения доступа посторонних лиц к компьютеру используется авторизация пользователя, состоящая из двух шагов: идентификации (определения пользователя по имени – логину) и аутентификации (подтверждения подлинности имени пользователя с использованием пароля). Авторизация пользователей выполняется при каждом входе пользователя в систему.

ALT Linux хранит следующую информацию о пользователе:

1) Имя пользователя (username) – регистрационное имя пользователя. Желательно, хотя и не обязательно, создавать имена пользователей, некоторым образом ассоциирующимися с их реальными именами.

2) Идентификатор пользователя (user ID) – индивидуальный числовой идентификатор пользователя (UID). Система ALT Linux обычно работает именно с UID, а не с именами пользователей. Идентификатор задается из диапазона 0…65534 и должен быть уникальным. Число 0 соответствует пользователю root. Идентификатор пользователя используется при идентификации пользователя в системе.

3) Идентификатор группы (group ID) – числовой идентификатор первичной группы пользователя (GID). Помимо первичной группы пользователь может входить в состав других групп. Идентификатор группы 0 соответствует группе root.

4) Пароль (password) – пароль пользователя.

5) Реальное имя пользователя (full name) – обычно представляет собой реальное (фактическое) имя пользователя. Используется в информационных целях.

6) Домашний каталог пользователя (home dir) – в качестве домашнего каталога используется каталог `/home/<имя пользователя>`.

7) Оболочка пользователя (login shell) – командный интерпретатор пользователя, который используется им по умолчанию. Командный интерпретатор запускается при входе пользователя в систему.

Вся эта информация хранится в файле `/etc/passwd`. Пароли в зашифрованном виде хранятся в файле `/etc/tcb/<имя пользователя>/shadow`.

4.1.11.2. Права доступа

Для каждого файла и каталога в ALT Linux задаются права доступа. Права доступа определяют, кто имеет доступ к объекту и какие операции над объектом он может выполнять. Под объектом следует понимать файл или каталог. Выполнять можно три основных операции:

– чтение;

– запись;

– выполнение.

Право на чтение файла означает, что его можно просматривать и печатать, а для каталога – что может отображаться список содержащихся в нём файлов. Право на запись для файла означает возможность его редактирования, а для каталога – возможность создания и удаления в нём файлов.

Если для файла установлено право выполнения, то его можно запускать как программу. Данная возможность используется при написании сценариев командных интерпретаторов. Право выполнения для каталога означает право доступа к каталогу, но не право выполнения расположенных в нём файлов.

Права доступа выставляются отдельно для различных категорий пользователей. Существует три категории пользователей:

1) Владелец – пользователь, создавший файл. Для того, чтобы создать файл необходимо иметь право записи в каталог, в котором файл создаётся. При создании файла устанавливаются права на чтение и запись для владельца и только на чтение для всех остальных пользователей.
2) Группа – набор пользователей, организованных, например, для работы с определённым набором файлов. Владелец может разрешить или запретить доступ к файлам для членов группы.
3) Прочие – это все остальные пользователи.

Права доступа задаются с использованием маски прав доступа – число в восьмеричной системе, задающее наборы прав доступа. Каждое такое число состоит из трёх разрядов. Первый разряд задаёт права доступа для владельца файла, второй – для группы, третий – для остальных пользователей. Одному разряду в восьмеричной системе соответствует три разряда в двоичной (Таблица 1).

Таблица 1 - Соответствие разрядов восьмеричной системы разрядам в двоичной

Восьмеричный формат	Двоичный формат
0	001
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Каждый двоичный разряд задаёт соответствующий ему тип доступа: первый – чтение, второй – запись, третий – выполнение. Разряды нумеруются слева направо. 0 – операция запрещена, 1 – операция разрешена.

4.1.11.3. Специальные права доступа SUID и SGID

В системе ALT Linux существуют ещё два специальных права доступа – SUID (Set User ID) и

SGID (Set Group ID). Эти права доступа могут использоваться определёнными программами, требующими для своей работы привилегий другого пользователя, что бывает необходимо для того, чтобы обыкновенный пользователь, не имеющий привилегий, смог выполнить разрешённые ему задачи в системе как с правами суперпользователя, так и с правами других системных пользователей.

SUID позволяет выполнять программу с привилегиями администратора от имени пользователя, SGID – от группы пользователей.

4.1.12. Квотирование – разграничение дискового пространства

Квотирование – это гибкой механизм, позволяющий ограничить дисковое пространство как для отдельных пользователей, так и для целых групп пользователей. Ограничения могут быть установлены как для пользователя, так и для группы. При этом, если пользователь входит в группу, которая превысила наложенное на неё ограничение, то он не сможет использовать дисковое пространство, даже если он не превысил квоту как пользователь.

Для каждого ограничения характерны четыре числа:

- ограничение, которое используется в данный момент;
- «Мягкое» ограничение;
- «Жёсткое» ограничение (ограничение, которое невозможно превысить);

Время, по истечении которого «мягкое» ограничение будет интерпретироваться как «жёсткое».

Мягкое ограничение определяет число блоков, которое пользователь все еще может превысить. «Жёсткое» ограничение превысить невозможно. При попытке сделать это будет выведено сообщение об ошибке. По истечении установленного времени «мягкое» ограничение переходит в «жёсткое». Размер блока составляет 1024 байт.

Выполнять действия, связанные с квотированием (ограничение дискового пространства для пользователей и групп, проверка целостности действительного число блоков и файлов и т.п.), может только суперпользователь с правами root.

4.1.13. Регистрация системных событий

4.1.13.1. Журнализация событий

В ALT Linux существует механизм централизованной журнализации, который реализован системной службой syslogd. Все части системы (включая ядро и системные службы) рапортуют syslogd о происходящих в них событиях. В этот рапорт включается имя службы, категория (facility)

и важность (priority) произошедшего события. Служба, сообразно настройкам, классифицирует все эти рапорты в несколько выходных потоков. Классификация и отсев данных всякого выходного потока происходит так: для каждой категории событий определяется наименьшая важность, которой событие должно обладать, чтобы попасть в этот выходной поток.

Главное место хранения уже классифицированного syslogd потока событий — системный журнал (т. н. log-файл). Системный журнал – текстовый файл, содержащий рапорты одного потока. syslogd хранит системные журналы в каталоге `/var/log` и его подкаталогах. Именно в системные журналы, прежде всего в `/var/log/messages`, `/var/log/maillog` и `/var/log/dmesg`, необходимо заглядывать администратору, который хочет знать, что происходит в системе. Поток рапортов о важных событиях syslogd направляет и на системную консоль – выделенное терминальное устройство. Стоит заметить, что некоторые службы самостоятельно, в обход syslogd, ведут журнализацию своих событий, поэтому информацию о количестве и местоположении их журналов можно почерпнуть из их файлов настроек (обычно, тем не менее, журналы хранятся в `/var/log`).

Новые рапорты, поступающие в системный журнал, наиболее актуальны, а предыдущие, по мере их устаревания, эту актуальность утрачивают. Если самые старые данные в журнале не удалять, файловая система, рано или поздно, окажется переполненной. В ALT Linux организован механизм устаревания журналов, которым занимается служба logrotate. Запускаясь раз в день, logrotate проверяет, какие из файлов следует признать устаревшими. Файл объявляется устаревшим раз в определённый промежуток времени (например, раз в неделю), или если он достиг определённого размера.

Некоторые файлы в `/var/log` — бинарные, они не являются полноценными журналами, а представляют собой «свалку событий» для служб авторизации и учёта.

4.1.13.2. Основные системные журналы

В ALT Linux создаются файлы системных журналов в директории `/var/log` со следующими названиями:

authlog/security – файл с сообщениями, связанными с аутентификацией пользователей, ошибками входа в систему, изменением уровня доступа и т.д (хранит в том числе сообщения с типом AUTH);

daemon – файл с сообщениями от системных служб (хранит сообщения с типом DAEMON);

dmesg – файл с сообщениями от ядра;

maillog/mail – файл с сообщениями о получении и доставке писем (этот журнал обычно ведётся

почтовым сервером);

messages – файл с сообщениями, не попавшими в другие файлы журналов;

xferlog – записи обо всех файлах, загруженных с данной машины (актуально для FTP-серверов).

4.1.13.3. Мониторинг пользователей

Журналирование входа пользователей в систему ведётся вне службы системного журнала, однако в директории `/var/log` есть несколько файлов, непосредственно связанных с мониторингом пользователей (все эти файлы имеют собственный двоичный формат):

wtmp – хранит информацию обо всех сеансах работы пользователя;

lastlog – для каждого из пользователей хранится время последнего входа в систему вместе с именем соответствующего терминала (и IP-адреса в случае сетевого входа в систему);

faillog – для каждого пользователя хранит информацию о последней неудачной попытке входа в систему.

4.1.14. Обеспечение целостности программных средств и обрабатываемой информации

4.1.14.1. Использование APT

Для автоматизации задачи контроля целостности и непротиворечивости установленного в системе ALT Linux ПО применяется система управления программными пакетами APT. Такая автоматизация достигается созданием внешних репозиториев, в которых хранятся пакеты программ и относительно которых производится сверка пакетов, установленных в системе.

В распоряжении APT находятся две базы данных: одна, описывающая установленные в системе пакеты, и вторая, с описанием внешнего репозитория. APT отслеживает целостность установленной системы и, в случае обнаружения противоречий в зависимостях пакетов, руководствуется сведениями о внешнем репозитории для разрешения конфликтов и поиска корректного пути их устранения.

Система APT состоит из нескольких утилит. Главной и наиболее часто используемой является утилита управления пакетами apt-get: она автоматически определяет зависимости между пакетами и строго следит за их соблюдением при выполнении любой из следующих операций: установка, удаление или обновление пакетов.

Утилита apt-get позволяет устанавливать в систему пакеты, требующие для своей работы другие, пока ещё не установленные. В этом случае она определяет, какие пакеты необходимо установить, и устанавливает их, пользуясь всеми доступными репозиториями. Информация о репозитари-

ях для apt-get помещается в файл `/etc/apt/sources.list`. Непосредственно после установки дистрибутива ALTLinux в `/etc/apt/sources.list` указаны несколько таких источников.

APT руководствуется базой данных, отражающей актуальное состояние репозитория. Такая база данных создаётся заново каждый раз, когда в репозитории происходит изменение: добавление, удаление или переименование пакета. Для ускорения работы apt-get хранит локальную копию базы данных, которая через некоторое время может уже не соответствовать реальному состоянию репозитория.

APT позволяет взаимодействовать с репозиторием с помощью различных протоколов доступа. Наиболее популярные — HTTP и FTP, именно они используются для работы с репозиториями по умолчанию.

APT позволяет работать одновременно с несколькими репозиториями, описанными в его конфигурационных файлах. APT, используемый в дистрибутиве ALT Linux, поддерживает множественные описания репозиториев в отдельных файлах в каталоге `/etc/apt/sources.list.d`. Все файлы в этом каталоге, имена которых составлены только из букв латинского алфавита, цифр, символов «-» и «_», трактуются APT как конфигурационные файлы, аналогичные `/etc/apt/sources.list`. С помощью этого расширения можно оформлять описания локальных репозиториев, не опасаясь, что при обновлении файлы конфигурации будут переписаны.

При использовании раздельных файлов конфигурации в `/etc/apt/sources.list.d` становится возможным управление поведением командой apt-get update. По умолчанию, эта команда вызывает обновление информации обо всех репозиториях, доступных APT. Однако если некоторый репозиторий описан в файле конфигурации /etc/apt/sources.list.d/<имя_файла_конфигурации>, то можно обновить информацию только о нем, указав имя конфигурационного файла в качестве аргумента команды apt-get update.

Для обновления всех установленных пакетов используется команда apt-get upgrade. Она позволяет обновить те и только те установленные пакеты, для которых в репозиториях, перечисленных в `/etc/apt/sources.list`, имеются новые версии; при этом из системы не будут удалены никакие другие пакеты. Этот способ полезен при работе со стабильными пакетами приложений, относительно которых известно, что они при смене версии изменяются несущественно.

Иногда, однако, происходит изменение в именовании пакетов или изменение их зависимостей. Такие ситуации не обрабатываются командой apt-get upgrade, в результате чего происходит нарушение целостности системы: появляются неудовлетворённые зависимости. Для разрешения этой проблемы существует режим обновления в масштабе дистрибутива — apt-get dist-upgrade.

В случае обновления всего дистрибутива APT проведёт сравнение системы с репозиторием и

удалит устаревшие пакеты, установит новые версии присутствующих в системе пакетов, а также отследит ситуации с переименованиями пакетов или изменения зависимостей между старыми и новыми версиями программ. Всё, что потребуется поставить (или удалить) дополнительно к уже имеющемуся в системе, будет указано в отчёте apt-get, которым APT предварит само обновление.

Для удаления пакета используется команда apt-get remove <имя_пакета>. Для того, чтобы не нарушать целостность системы, будут удалены и все пакеты, зависящие от удаляемого: если отсутствует необходимая для работы приложения библиотека, то само приложение становится бесполезным). В случае удаления пакета, который относится к базовым компонентам системы, apt-get потребует дополнительного подтверждения производимой операции с целью предотвратить возможную случайную ошибку.

Иногда, в результате операций с пакетами без использования APT, целостность системы нарушается и apt-get отказывается выполнять операции установки, удаления или обновления. В этом случае необходимо повторить операцию, задав опцию -f, заставляющую apt-get исправить нарушенные зависимости, если это возможно. В этом случае необходимо внимательно следить за сообщениями, выдаваемыми apt-get, анализировать их и чётко следовать рекомендациям программы.

apt-get всегда спрашивает подтверждение выполнения операции установки и обновления, за исключением случая, когда реально требуется установить в систему (или обновить) только один пакет. Если пользователь не уверен в том, что в результате выполнения операции система останется работоспособной, необходимо запустить apt-get с опцией -S, которая покажет отчёт выполнения операции обновления, но реально обновление произведено не будет.

В случае обнаружения противоречий между установленными в системе пакетами следует запустить команду apt-get -f install, и APT постарается разрешить найденные конфликты, предложив удалить или заменить конфликтующие пакеты. Любые действия в этом режиме обязательно требуют подтверждения со стороны пользователя.

4.1.14.2. Проверка целостности КСЗ

В ALT Linux реализован механизм, позволяющий осуществлять контроль за целостностью КСЗ. Работа данного механизма реализуется с помощью приложения osec. При запуске приложение сверяет контрольные суммы и обновляет базу, записывая туда новые значения.

Работает osec под специальным непривилегированным пользователем (состоящим в непривилегированной группе) с одной дополнительной возможностью: dac_read_search. Как следствие исключается возможность атаки на это приложение со стороны пользователей с целью повышения полномочий. Результат работы выводится на стандартный вывод.

Алгоритм проверки следующий:

1) Необходимо войти в систему как администратор (пользователь root).
2) Запустить приложение osec, которое создаст отчёт о работе (изначально аналогичным образом должен быть создан первоначальный отчёт, с которым будет выполняться сравнение).
3) Сравниваем отчёты, созданные в п.2 и созданным заранее.

Наличие различий между отчётами говорит о том, что какие-либо файлы или их атрибуты (права доступа) были изменены.

4.2. Общий алгоритм работы ОС ALT Linux

4.2.1. Запуск ALT Linux

Запуск ALT Linux — процесс, состоящий из нескольких этапов. Начальный этап, называемый досистемной загрузкой, не зависит от того, какая операционная система установлена на компьютере.

Начиная с определённого этапа загрузка компьютера уже управляется самим ALT Linux: используются утилиты, сценарии и т. п. Этот этап называется системной загрузкой.

4.2.1.1. Досистемная загрузка

4.2.1.1.1. Загрузчик в ПЗУ

ПЗУ состоит из множества подпрограмм, занимающихся взаимодействием с разнообразными устройствами ввода-вывода (жёсткие и гибкие диски, магнитные ленты, сетевые карты, последовательные порты передачи данных, системная клавиатура, видеокарта и др.) Этот набор подпрограмм в ПЗУ называется BIOS (basic input-output system).

Сразу после включения в память ЭВМ автоматически загружается программа из постоянного запоминающего устройства, ПЗУ (или ROM, read-only memory), которая выполняет распознавание и инициализацию основных устройств компьютера, а затем передаёт управление на специальную программу-загрузчик.

Этот этап загрузки системы можно назвать нулевым, так как ни от какой системы он не зависит. Его задача — определить, с какого устройства будет идти загрузка, загрузить оттуда программу-загрузчик и запустить его. Например, выяснить, что устройство для загрузки — жёсткий диск, считать самый первый сектор этого диска и передать управление программе, которая находится в считанной области.

4.2.1.1.2. Загрузочный сектор и первичный загрузчик

Размер первичного дискового загрузчика — программы, которой передаётся управление после нулевого этапа, — весьма невелик и занимает не более одного сектора в самом начале диска, в его загрузочном секторе. Его задачи:

- определить, где на диске находится вторичный загрузчик;
- загрузить и запустить его.

4.2.1.1.3. Вторичный загрузчик (загрузчик ядра)

В задачу вторичного загрузчика входит загрузка и начальная настройка ядра операционной системы.

Ядро ALT Linux хранится в сжатом файле. Когда ядро запускается программой начальной загрузки (GRUB), оно распаковывает себя, инициализирует устройство отображения и запускает проверку другого оборудования, присоединённого к компьютеру. Когда ядро находит жёсткие диски, дискеты, сетевые адаптеры и т.д., оно загружает соответствующие модули драйверов устройств. Во время этого процесса ядро выводит текстовые сообщения на экране консоли.

Ядро монтирует корневую файловую систему (/) только для чтения и выполняет проверку файловой системы.

Вторичный загрузчик может не только загружать ядро, но и настраивать его. В ALT Linux используется механизм настройки ядра, похожий на командную строку shell: в роли команды выступает ядро, а в роли параметров – настройки ядра. Настройки ядра нужны для временного изменения его функциональности: например, чтобы выбрать другой графический режим виртуальных консолей, чтобы отключить поддержку дополнительных возможностей внешних устройств (если аппаратура их не поддерживает), чтобы передать самому ядру указания, как загружать систему и т. п.

4.2.1.1.4. Досистемная загрузка Linux

Несмотря на то, что досистемная загрузка не зависит от типа операционной системы, которая начинает работу после неё, система ALT Linux предоставляет собственные средства по её организации – подсистему загрузки GRUB (GRand Unified Bootloader).

В загрузочное меню включаются различные варианты загрузки системы, а также, в случае наличия, возможность загрузки других установленных операционных систем.

4.2.1.1.5. Действия ядра Linux в процессе начальной загрузки

Ядро ALT Linux управляет доступом к оперативной памяти, сети, дисковым и прочим внешним

устройствам и т.п. Оно запускает и регистрирует процессы, управляет разделением времени между ними, реализует разграничение прав и определяет политику безопасности, обойти которую, не обращаясь к нему, нельзя.

Ядро работает в специальном режиме, т. н. «режиме супервизора», позволяющем ему иметь доступ сразу ко всей оперативной памяти и аппаратной таблице задач. Процессы запускаются в «режиме пользователя»: каждый жёстко привязан ядром к одной записи таблицы задач, в которой, в числе прочих данных, указано, к какой именно части оперативной памяти этот процесс имеет доступ. Ядро постоянно находится в памяти, выполняя системные вызовы – запросы от процессов на выполнение этих подпрограмм.

Работа ядра после того, как ему передано управление, и до того, как оно начнёт работать в штатном режиме, выполняя системные вызовы, сводится к следующему:

1) Ядро определяет аппаратное окружение. Одно и то же ядро может быть успешно загружено и работать на разных компьютерах одинаковой архитектуры, но с разним набором внешних устройств. Задача ядра – определить список внешних устройств, составляющих компьютер, классифицировать их (определить диски, терминалы, сетевые устройства и т. п.) и, если надо, настроить. При этом на системную консоль выводятся диагностические сообщения (впоследствии их можно просмотреть утилитой dmesg).
2) Ядро запускает несколько процессов ядра. Процесс ядра – это часть ядра Linux, зарегистрированная в таблице процессов. Такому процессу можно послать сигнал и вообще пользоваться средствами межпроцессного взаимодействия, на него распространяется политика планировщика задач, однако никакой задаче в режиме пользователя он не соответствует.
3) Ядро монтирует корневую файловую систему в соответствии с переданными параметрами. Подключение это происходит в режиме «только для чтения» (read-only): если целостность файловой системы нарушена, этот режим позволит, не усугубляя положение, прочитать и запустить утилиту fsck (file system check). Позже, в процессе загрузки, корневая файловая система подключится на запись.
4) Ядро запускает из файла /sbin/init первый настоящий процесс. Идентификатор процесса (PID) у него равен единице, он – первый в таблице процессов, даже несмотря на то, что до него там были зарегистрированы процессы ядра.

4.2.1.2. Загрузка системы

4.2.1.2.1. Запуск процесса init

Загрузка самой системы начинается с запуска init, который разбирает собственный конфигурационный файл – `/etc/inittab`. Файл этот имеет довольно простую структуру и содержит список процессов, данные об их уровнях выполнении и способах запуска. Программа init выполняет сценарий `/etc/rc.d/rc.sysinit` прежде чем обрабатывать любые другие сценарии для желаемого уровня выполнения. На данном этапе загрузки системы (sysinit) выполняются следующие действия:

1) Устанавливается имя машины (hostname).
2) Конфигурируются параметры ядра.
3) Устанавливается раскладка клавиш и системный шрифт.
4) Активируются разделы подкачки.
5) Корневая система проверяется программой fsck. Если программа fsck ошибок не обнаружила, файловая система монтируется в режиме чтение/запись.
6) Проверяются зависимости модулей ядра.
7) Выполняется проверка других файловых систем.
8) Монтируются локальные файловые системы.
9) Включаются квоты.
10) Монтируется раздел подкачки.

4.2.1.2.2. Запуск системных служб

ALT Linux себе самому и своим пользователям предоставляет множество услуг: отсылка заданий на печать и обеспечения их очереди, запуск заданий по расписанию, проверка целостности и т. п. Набор утилит и системных программ, предназначенных для предоставления таких услуг, принято называть подсистемами или службами.

Системная служба организована следующим образом. Во время начальной загрузки запускается в фоновом режиме программа, которая всё время работы системы находится в таблице процессов, однако большей частью бездействует. Для того, чтобы эта программа выполнила какое-либо действие, которое она может выполнить, используются утилиты, взаимодействующие с ней по специальному протоколу. Такую программу стали называть службой (daemon – демон) – запускаемая в фоне программа, длительное время пребывающая в таблице процессов. Обычно сдужба активизируется по запросу пользовательской программы, по сетевому запросу или по наступлению какого-

либо системного события.

4.2.1.2.3. Стартовый сценарий системной службы

Стартовый сценарий – программа (обычно написанная на shell), управляющая включением или выключением какого-нибудь свойства системы. Это может быть запуск и остановка HTTP-сервера, активизация и деактивизация сетевых настроек, загрузки модулей и настройка звуковой подсистемы и т. п. Простейший стартовый сценарий обязан принимать один параметр, значение которого может быть словом «start» для запуска (включения) и «stop» для остановки (выключения).

Все стартовые сценарии служб, которыми может воспользоваться система, хранятся в каталоге `/etc/rc.d/init.d`. Запустить или остановить службу можно, просто вызвав соответствующий сценарий с параметром «start» или «stop». Ту же самую задачу выполняет и специальная команда service, которая проверяет, есть ли указанный стартовый сценарий, и запускает его.

4.2.1.2.4. Уровни выполнения

Поэтому в ALT Linux предусмотрено несколько вариантов начальной загрузки, называемых уровни выполнения (run levels). Уровни выполнения нумеруются с 0 до 6:

Уровень 1 соответствует однопользовательскому режиму загрузки системы. При загрузке на уровень 1 не запускается никаких служб, системная консоль доступна только одна, так что в системе может работать не более одного пользователя. В однопользовательском режиме изредка работает администратор – исправляет неполадки системы, изменяет ключевые настройки, обслуживает файловые системы.

Уровень 2 соответствует многопользовательскому режиму загрузки системы с отключённой сетью. В этом режиме не запускаются никакие сетевые службы, что, с одной стороны, соответствует строгим требованиям безопасности, а с другой стороны, позволяет запускать службы и настраивать сеть вручную.

Уровень 3 соответствует многопользовательскому сетевому режиму загрузки системы. Сеть при загрузке на этот уровень настроена, и все необходимые сетевые службы запущены. На этом уровне обычно работают компьютеры-серверы.

Уровни 0 и 6 – специальные. Они соответствуют останову и перезагрузке системы. В сущности, это удобные упрощения для действий, обратных загрузке на уровень: все службы останавливаются, диски размонтируются. В случае останова даже электропитание можно отключать программно, если аппаратура позволяет, а в случае перезагрузки система идёт на повторную загрузку.

Остальные уровни никак специально в ALT Linux не описаны, однако администратор может использовать и их, определяя особый профиль работы системы.

Переход с уровня на уровень сопровождается не только запуском, но и остановкой служб. Это касается не только уровней 0 и 6, но и любых других. Например, при переходе с уровня 3 на уровень 2 необходимо остановить все сетевые службы. Если при переходе с уровня на уровень некой службе не требуется менять своего состояния, сценарий не запускается.

4.2.2. Остановка системы

Остановка системы может занимать больше времени, чем загрузка: например, процессы, выполняющие системный вызов (скажем, чтения с дискеты), не завершаются по сигналу TERM сразу, а получив его, могут некоторое время заниматься обработкой (дописыванием в файл и т. п.). Остановка службы, особенно сетевой, тоже может длиться долго: например, когда требуется сообщить о закрытии сервиса каждому клиенту. Однако только в этом случае можно быть уверенным, что все процессы завершились нормально, и что после перезагрузки они продолжат нормально работать.

Для остановки или перезагрузки системы можно выполнять команды init 0 и init 6. При завершении работы ОС (выключение системы, перезагрузка) выполняется завершение всех выполняющихся процессов, размонтирование файловых систем, в т.ч. и корневой файловой системы. При размонтировании файловой системы происходит синхронизация буферов дискового ввода-вывода с самим жестким диском.

4.2.3. Вход в систему

Вход пользователя в систему осуществляется путём указания имени пользователя (логина) и пароля.

Если пароль указан неверно, в журнале безопасности регистрируется попытка несанкционированного доступа к системе.

После выполнения входа в систему (в режиме командной строки) запускается командная оболочка и появится приглашение командной строки.

4.2.4. Виртуальные консоли

В процессе работы ALT Linux активно несколько виртуальных консолей. Каждая виртуальная консоль доступна по одновременному нажатию Alt и функциональной клавиши с номером этой консоли. На первых шести виртуальных консолях (Alt+F1 — Alt+F6) пользователь может зарегистрироваться и работать в текстовом режиме. Двенадцатая виртуальная консоль (Alt+F12) выполняет функцию системной консоли – на неё выводятся сообщения о происходящих в системе событиях.

Благодаря виртуальным консолям каждый компьютер, на котором работает ALT Linux, предо-

ставляет возможность зарегистрироваться и получить доступ к системе одновременно нескольким пользователям. Даже если в распоряжении всех пользователей есть только один монитор и одна системная клавиатура, эта возможность небесполезна: можно переключаться между виртуальными консолями так, как если бы вы переходили от одного монитора с клавиатурой к другому, подавая время от времени команды и следя за выполняющимися там программами. Более того, ничто не препятствует зарегистрироваться в системе несколько раз под одним и тем же системным именем – это один из способов организовать параллельную работу над несколькими задачами.

4.2.5. Командная строка

Командная строка – это способ организации интерфейса, в котором каждая строка, введённая пользователем – это команда системе, которую та должна выполнить. Команды вводятся обычно в одну строку, которая завершается нажатием клавиши «ввод» (Enter). В Linux этот вид интерфейса всегда был основным, а потому хорошо развитым.

Первое слово в такой строке — это, как правило, имя исполняемого файла — программы, все остальные слова — параметры. Программа выполняет нужные пользователю действия, но может делать это по-разному в зависимости от полученных параметров. Параметры могут быть общими, например имя файла, который нужно обработать, или специфическими для этой программы модификаторами выполнения.

Чтобы получить командную строку, пользователь должен войти в систему и запустить программу, которая будет принимать его команды и передавать их на выполнение — командную оболочку (её ещё называют интерпретатор командной строки, просто оболочка, по-английски «shell»).

Получить командную строку можно многими способами. Самый простой и универсальный — зарегистрироваться на одной из первых шести виртуальных консолей: после входа в систему запустится командная оболочка и появится приглашение командной строки.

Каждая команда — это отдельная строка. Пока не нажат Enter, строку можно редактировать, затем она передаётся оболочке. Оболочка разбирает полученную команду — переводит её на язык системных объектов и функций, после чего отправляет системе на выполнение.

Результат выполнения очень многих команд также представляет собой текст, выдаваемый в качестве «ответа» пользователю. Хотя это и не обязательно — команда может выполнять свою работу, не выдавая никаких сообщений. Кроме того, если в процессе выполнения команды возникли какие-то особые обстоятельства (например, ошибка), оболочка включит в ответ пользователю диагностические сообщения.

4.2.6. Графическая оболочка GNOME

GNOME – это интерактивная рабочая среда, представляющая собой набор программ, технологий и документации. GNOME ориентирована на рабочие станции под управлением Unix.

Основные характеристики GNOME – это простота пользовательского интерфейса и обеспечение простой разработки приложений, интегрируемых со средой, на различных языках программирования.

GNOME предоставляет пользователям множество инструментов для работы.

Файловый менеджер Nautilus обеспечивает отрисовку рабочего стола со значками на нём, а также работу с файлами и директориями.

Программа GNOME Panel предоставляет панели для рабочего стола GNOME. По умолчанию GNOME имеет две панели, расположенные по верхнему и нижнему краям рабочего стола.

Вместе с GNOME Panel поставляется набор апплетов — небольших приложений, которые встраиваются в панель для выполнения различных функций, например, отображения даты и времени, списка открытых окон или индикатора раскладки клавиатуры.

Эмулятор терминала GNOME Terminal предоставляет доступ к командной оболочке UNIX для пользователя графической среды. Он поддерживает все типичные функции эмулятора терминала, а также цветной вывод и события от мыши.

Gedit – текстовый редактор с поддержкой юникода. Поддерживает использование вкладок для представления нескольких документов в одном окне, подсветку синтаксиса для ряда компьютерных языков, и другие возможности.

Веб-браузер Epiphany поддерживает просмотр нескольких страниц в одном окне с помощью вкладок, систему категоризуемых закладок, «умные закладки», а также систему расширений, с помощью которых в него были добавлены популярные функции из других браузеров. Epiphany использует движок WebKit (используется также в Safari и Google Chrome).

Evolution – приложение для управления электронной почтой, расписанием и адресной книгой. Поддерживает все основные почтовые протоколы, серверы Microsoft Exchange и GroupWise, включает в себя спам-фильтр и предоставляет ряд других возможностей.

Ekiga – приложение IP-телефонии и проведения видеоконференций, которое ранее называлось GnomeMeeting. Ekiga поддерживает протоколы SIP и H.323 и способна взаимодействовать с другими SIP-совместимыми клиентами, а также с Microsoft NetMeeting.

Pidgin – приложение мгновенного обмена сообщениями, поддерживающее множество протоколов текстовых сообщений, а также видео- и голосовое общение.

Технология GStreamer обеспечивает «прозрачную» работу с аудио и видео различных форматов – ввод, обработку и вывод.

Приложение Yelp предназначено для просмотра разного рода документации, установленной в системе. Yelp позволяет просматривать как справку по приложениям GNOME, так и стандартные справочные материалы man и texinfo.

GNOME System Tools – это комплект графических средств для администрирования UNIX-систем содержащий инструменты для:

- настройки учётных записей пользователей системы;
- сетевых подключений;
- даты и времени;
- системных служб;
- общих сетевых ресурсов.

4.2.7. Система управления пакетами APT

4.2.7.1. Введение: пакеты, зависимости и репозитории

В современных системах на базе Linux огромное число общих ресурсов, которыми пользуются сразу несколько программ: разделяемых библиотек, содержащих стандартные функции, исполняемых файлов, сценариев и стандартных утилит и т. д. Удаление или изменение версии одного из составляющих систему компонентов может повлечь неработоспособность других, связанных с ним компонентов, или даже вывести из строя всю систему. В контексте системного администрирования проблемы такого рода называют нарушением *целостности системы*. Задача администратора – обеспечить наличие в системе согласованных версий всех необходимых программных компонентов (обеспечение целостности системы).

Для установки, удаления и обновления программ и поддержания целостности системы в Linux в первую очередь стали использоваться *менеджеры пакетов* (такие, как rpm в дистрибутивах RedHat или dpkg в Debian GNU/Linux). С точки зрения менеджера пакетов программное обеспечение представляет собой набор компонентов – программных *пакетов*. Такие компоненты содержат в себе набор исполняемых программ и вспомогательных файлов, необходимых для корректной работы программного обеспечения. Менеджеры пакетов облегчают установку программ: они позволяют проверить наличие необходимых для работы устанавливаемой программы компонент подходящей версии непосредственно в момент установки, а также производят необходимые процедуры для регистрации программы во всех операционных средах пользователя: сразу после установки программа может быть доступна пользователю из командной строки и – если это предусмотрено – появляется в меню всех графических оболочек.

Важно: Благодаря менеджерам пакетов, пользователю Linux обычно не требуется непосредственно обращаться к установочным процедурам отдельных программ или непосредственно работать с каталогами, в которых установлены исполняемые файлы и компоненты программ (обычно это `/usr/bin`, `/usr/share/имя_пакета`) – всю работу делает менеджер пакетов. Поэтому установку, обновление и удаление программ в Linux обычно называют управлением пакетами.

Часто компоненты, используемые различными программами, выделяют в отдельные пакеты и помечают, что для работы ПО, предоставляемого пакетом A, необходимо установить пакет B. В таком случае говорят, что пакет A *зависит* от пакета B или что между пакетами A и B существует *зависимость*.

Отслеживание зависимостей между такими пакетами представляет собой серьёзную задачу для любого дистрибутива – некоторые компоненты могут быть взаимозаменяемыми: может обнаружиться несколько пакетов, предлагающих затребованный ресурс.

Задача контроля целостности и непротиворечивости установленного в системе ПО ещё сложнее. Представим, что некие программы A и B требуют наличия в системе компоненты C версии 1.0. Обновление версии пакета A, требующее обновления компоненты C до новой, использующей новый интерфейс доступа, версии (скажем, до версии 2.0), влечёт за собой обязательное обновление и программы B.

Однако менеджеры пакетов оказались неспособны предотвратить все возможные коллизии при установке или удалении программ, а тем более эффективно устранить нарушения целостности системы. Особенно сильно этот недостаток сказывается при обновлении систем из централизованного *репозитория пакетов*, в котором последние могут непрерывно обновляться, дробиться на более мелкие и т. п. Этот недостаток и стимулировал создание систем управления программными пакетами и поддержания целостности системы.

Для автоматизации этого процесса и применяется Усовершенствованная система управления программными пакетами APT (от англ. Advanced Packaging Tool). Такая автоматизация достигается созданием одного или нескольких внешних *репозиториев*, в которых хранятся пакеты программ и относительно которых производится сверка пакетов, установленных в системе. Репозитории могут содержать как официальную версию дистрибутива, обновляемую его разработчиками по мере выхода новых версий программ, так и локальные наработки, например, пакеты, разработанные внутри компании.

Таким образом, в распоряжении APT находятся две базы данных: одна описывает установленные в системе пакеты, вторая – внешний репозиторий. APT отслеживает целостность установленной системы и, в случае обнаружения противоречий в зависимостях пакетов, руководствуется сведениями о внешнем репозитории для разрешения конфликтов и поиска корректного пути их устранения.

Первоначально APT был разработан для управления установкой и удалением программ в дистрибутиве Debian GNU/Linux. При разработке ставилась задача создать систему управления пакетами с простым пользовательским интерфейсом, позволяющую производить установку, обновление и повседневные «хозяйственные» работы с установленными на машине программами без необходимости изучения тонкостей используемого в дистрибутиве менеджера программных пакетов.

Эти привлекательные возможности долгое время были доступны только пользователям Debian, поскольку в APT поддерживался только один менеджер пакетов, а именно применяемый в Debian менеджер пакетов dpkg, несовместимый с используемым в ALT Linux RPM. Эта несовместимость заключается прежде всего в различии используемых форматов данных (хотя существуют программы-конвертеры), но имеются и другие различия, обсуждение которых выходит за рамки изложе-

ния.

APT, однако, изначально проектировался как не зависящий от конкретного метода работы с установленными в системе пакетами, и эта особенность позволила разработчикам из бразильской компании Conectiva реализовать в нём поддержку менеджера пакетов RPM. Таким образом, пользователи основанных на RPM дистрибутивов (дистрибутивы ALT Linux входят в их число) получили возможность использовать этот мощный инструмент.

Система APT состоит из нескольких утилит. Чаще всего используется утилита управления пакетами apt-get: она автоматически определяет зависимости между пакетами и строго следит за их соблюдением при выполнении любой из следующих операций: установка, удаление или обновление пакетов.

4.2.7.2. Источники программ (репозитории)

4.2.7.2.1. Репозитории

Репозитории, с которыми работает APT, отличаются от обычного набора пакетов наличием мета информации – индексов пакетов, содержащихся в репозитории, и сведений о них. Поэтому, чтобы получить всю информацию о репозитории, APT достаточно получить его индексы.

APT может работать с любым количеством репозиториев одновременно, формируя единую информационную базу обо всех содержащихся в них пакетах. При установке пакетов APT обращает внимание только на название пакета, его версию и зависимости, а расположение в том или ином репозитории не имеет значения. Если потребуется, APT в рамках одной операции установки группы пакетов может пользоваться несколькими репозиториями.

Важно: Подключая одновременно несколько репозиториев, нужно следить за тем, чтобы они были совместимы друг с другом по пакетной базе, т. е. отражали один определённый этап разработки. Например, совместимыми являются основной репозиторий дистрибутива и репозиторий обновлений по безопасности к данному дистрибутиву. В то же время смешение среди источников APT репозиториев, относящихся к разным дистрибутивам, или смешение стабильного репозитория с нестабильной веткой разработки (Sisyphus) чревато различными неожиданными трудностями при обновлении пакетов.

APT позволяет взаимодействовать с репозиторием с помощью различных протоколов доступа. Наиболее популярные – HTTP и FTP, однако существуют и некоторые дополнительные методы.

Для того, чтобы APT мог использовать тот или иной репозиторий, информацию о нем необходимо поместить в файл `/etc/apt/sources.list`. Описания репозиториев заносятся в этот файл в следующем виде:

```
rpm [подпись] метод:путь база название

rpm-src [подпись] метод:путь база название
```

rpm или rpm-src

Тип репозитория (скомпилированные программы или исходные тексты).

[подпись]

Необязательная строка-указатель на электронную подпись разработчиков. Наличие этого поля подразумевает, что каждый пакет из данного репозитория должен быть подписан соответствующей электронной подписью. Подписи описываются в файле `/etc/apt/vendor.list`.

Метод

Способ доступа к репозиторию: ftp, http, file, rsh, ssh, cdrom, copy, rpm-dir.

Путь

Путь к репозиторию в терминах выбранного метода.

База

Относительный путь к базе данных репозитория.

Название

Название репозитория.

Для добавления в `sources.list` репозитория на компакт-диске в APT даже предусмотрена специальная утилита – `apt-cdrom`. Чтобы добавить запись о репозитории на компакт-диске, достаточно вставить диск в привод и выполнить команду `apt-cdrom add`. После этого в `sources.list` появится запись о подключённом диске примерно такого вида:

```
rpm cdrom:[Server Disk 1]/ ALTLinux main

rpm-src cdrom:[Server Disk 1]/ ALTLinux main
```

После того как отредактирован список репозиториев в `sources.list`, необходимо обновить локальную базу данных APT о доступных пакетах. Это делается командой `apt-get update`.

Если в `sources.list` присутствует репозиторий, содержимое которого может изменяться (как происходит с любым постоянно разрабатываемым репозиторием, в частности, обновлений по безопасности (updates), то прежде чем работать с APT, необходимо синхронизировать локальную базу данных с удалённым сервером командой `apt-get update`. Локальная база данных создаётся заново каждый раз, когда в репозитории происходит изменение: добавление, удаление или переименование пакета. Для репозиториев, находящихся на компакт-дисках и подключённых командой

`apt-cdrom add`, синхронизация производится единожды в момент подключения.

При выборе пакетов для установки, APT руководствуется *всеми* доступными репозиториями вне зависимости от способа доступа к ним. Так, если в репозитории, доступном по сети Интернет, обнаружена более новая версия программы, чем на компакт-диске, то APT начнёт загружать данный пакет из Интернет. Поэтому, если подключение к Интернет отсутствует или ограничено низкой пропускной способностью канала или выской стоимостью, то следует закомментировать те строчки в `/etc/apt/sources.list`, в которых говорится о ресурсах, доступных по Интернет.

4.2.7.2.2. Репозитории Sisyphus

Все дистрибутивы ALT Linux выпускаются на основе репозитория Sisyphus.

Следует иметь в виду, что Sisyphus не является самостоятельным дистрибутивом, а отражает текущее состояние разработки и может содержать нестабильные версии пакетов. Периодически на базе этого проекта выпускаются отдельные оттестированные «срезы» — дистрибутивы.

В отличие от Sisyphus, ежедневно обновляемого разработчиками, такие срезы являются «замороженными» – разработка в них не ведётся, и сами срезы сохраняются в целях обеспечения целостности среды дистрибутива, в которой уже не должны обновляться версии пакетов. Единственное исключение делается для обновлений, исправляющих проблемы в безопасности системы, однако такие обновления помещаются в отдельном репозитории для каждого дистрибутива. Срезы Sisyphus и репозитории обновлений также являются полноценными репозиториями APT.

Непосредственно после установки дистрибутива ALT Linux в `/etc/apt/sources.list`, в таже в файлах `/etc/apt/sources.list.d/*.list` обычно указывается несколько репозиториев:

- репозиторий обновлений в системе безопасности дистрибутива;
- полный срез репозитория, на котором основывается дистрибутив.

4.2.7.3. Поиск пакетов

Если вы не знаете точного названия пакета, для его поиска можно воспользоваться утилитой `apt-cache`, которая позволяет искать не только по имени пакета, но и по его описанию.

Команда `apt-cache search подстрока` позволяет найти все пакеты, в именах или описании которых присутствует указанная подстрока. Например:

```
$ apt-cache search ^ve-
ve-base - Basic appliance
ve-build-scripts - scripts used for VE building
```

```
ve-caching-nameserver - Caching name server
ve-ftp-server - FTP server
ve-imap-server - POP3/IMAP4 server
ve-kerberos-server - Kerberos server
ve-list-server - Mailing list server
ve-ntp-server - NTP server
ve-openvpn-server - virtual package for openvpn server appliance
ve-pptp-server - virtual package for pptp server appliance
ve-print-server - Print server
ve-proxy-server - Proxy server
ve-smtp-server - SMTP server
```

Обратите внимание, что в данном примере в поисковом выражении используется символ ^, указывающий на то, что необходимо найти совпадения только в начале строки (в данном случае – в начале имени пакета).

Для того, чтобы подробнее узнать о каждом из найденных пакетов и прочитать его описание, можно воспользоваться командой `apt-cache show`, которая покажет информацию о пакете из репозитория:

```
$ apt-cache show ve-ftp-server
Package: ve-ftp-server
Section: System/Base
Installed Size: 0
Maintainer: Stanislav Ievlev <inger@altlinux>
Version: 0.1-alt4
Pre-Depends: rpmlib(PayloadFilesHavePrefix) (<= 4.0-1), rpmlib(CompressedFileNames)
(<= 3.0.4-1)
Depends: apt, basesystem, sysklogd, etcnet, glibc-nss, glibc-locales, netlist, anonftp, alterator-fbi, alterator-vsftpd, alterator-users, openssh-server, passwd, less
```

```
Provides: ve-ftp-server (= 0.1-alt4)
Architecture: noarch
Size: 1989
MD5Sum: e7646f729b2bced59e4e759393cb1432
Filename: ve-ftp-server-0.1-alt4.noarch.rpm
Description: FTP server
 virtual package for ftp server appliance
```

apt-cache позволяет осуществлять поиск и по русскому слову, однако в этом случае будут найдены только те пакеты, у которых помимо английского есть ещё и описание на русском языке. К сожалению, русское описание на настоящий момент есть не у всех пакетов, хотя описания наиболее актуальных для пользователя пакетов переведены.

4.2.7.4. Установка или обновление пакета

Установка пакета с помощью APT выполняется командой:

```
# apt-get install имя_пакета
```

apt-get позволяет устанавливать в систему пакеты, требующие для работы другие, пока ещё не установленные. В этом случае он определяет, какие пакеты необходимо установить, и устанавливает их, пользуясь всеми доступными репозиториями.

Установка пакета ve-ftp-server командой `apt-get install ve-ftp-server` приведёт к следующему диалогу с APT:

```
# apt-get install ve-ftp-server
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
alterator-users alterator-vsftpd anonftp klogd netlist pwgen shadow-groups sysklogd vsftpd xinetd
The following NEW packages will be installed:
alterator-users alterator-vsftpd anonftp klogd netlist pwgen shadow-groups sysklogd ve-ftp-server vsftpd xinetd
0 upgraded, 11 newly installed, 0 removed and 0 not upgraded.
```

```
Need to get 417kB of archives.
After unpacking 700kB of additional disk space will be used.
Do you want to continue? [Y/n]
. . .
Fetched 417kB in 0s (2176kB/s)
Committing changes...
Preparing...
   ################################################## [100%]
 1: xinetd
   ################################################## [  9%]
 2: vsftpd
   ################################################## [ 18%]
 3: alterator-vsftpd
   ################################################## [ 27%]
 4: netlist
   ################################################## [ 36%]
 5: pwgen
   ################################################## [ 45%]
 6: klogd
   ################################################## [ 54%]
 7: anonftp
   ################################################## [ 63%]
 8: shadow-groups
   ################################################## [ 72%]
 9: alterator-users
   ################################################## [ 81%]
Done.
```

Команда `apt-get install имя_пакета` используется и для обновления уже установленного пакета или группы пакетов. В этом случае apt-get дополнительно проверяет, не обновилась ли версия пакета в репозитории по сравнению с установленным в системе.

При помощи APT можно установить и отдельный бинарный rpm-пакет, не входящий ни в один

из репозиториев (например, полученный из Интернет). Для этого достаточно выполнить команду `apt-get install путь_к_файлу.rpm`. При этом APT проведёт стандартную процедуру проверки зависимостей и конфликтов с уже установленными пакетами.

Иногда, в результате операций с пакетами без использования APT, целостность системы нарушается, и apt-get отказывается выполнять операции установки, удаления или обновления. В этом случае необходимо повторить операцию, задав опцию `-f`, заставляющую apt-get исправить нарушенные зависимости, удалить или заменить конфликтующие пакеты. В этом случае необходимо внимательно следить за сообщениями, выдаваемыми apt-get. Любые действия в этом режиме обязательно требуют подтверждения со стороны пользователя.

4.2.7.5. Удаление установленного пакета

Для удаления пакета используется команда `apt-get remove имя_пакета`. Для того, чтобы не нарушать целостность системы, будут удалены и все пакеты, зависящие от удаляемого: если отсутствует необходимый для работы приложения компонент (например, библиотека), то само приложение становится бесполезным. В случае удаления пакета, который относится к базовым компонентам системы, apt-get потребует дополнительного подтверждения производимой операции с целью предотвратить возможную случайную ошибку.

Если вы попробуете при помощи apt-get удалить базовый компонент системы, вы увидите такой запрос на подтверждение операции:

```
# apt-get remove filesystem
Обработка файловых зависимостей... Завершено
Чтение списков пакетов... Завершено
Построение дерева зависимостей... Завершено
Следующие пакеты будут УДАЛЕНЫ:
basesystem filesystem ppp sudo
Внимание: следующие базовые пакеты будут удалены:
В обычных условиях этого не должно было произойти, надеемся, вы точно
представляете, чего требуете!
basesystem filesystem (по причине basesystem)
0 пакетов будет обновлено, 0 будет добавлено новых, 4 будет
удалено(заменено) и 0 не будет обновлено.
```

```
Необходимо получить 0B архивов. После распаковки 588kБ будет
освобождено.
Вы собираетесь совершить потенциально вредоносное действие
Для продолжения, наберите по-английски 'Yes, I understand this may
be
bad'
(Да, я понимаю, что это может быть плохо).
```

Каждую ситуацию, в которой APT выдаёт такое сообщение, необходимо рассматривать отдельно. Однако, вероятность того, что после выполнения этой команды система окажется неработоспособной, очень велика.

4.2.7.6. Обновление всех установленных пакетов

Для обновления всех установленных пакетов используется команда `apt-get upgrade`. Она позволяет обновить те и только те установленные пакеты, для которых в репозиториях, перечисленных в `/etc/apt/sources.list`, имеются новые версии; при этом из системы не будут удалены никакие другие пакеты. Этот способ полезен при работе со стабильными пакетами приложений, относительно которых известно, что они при смене версии изменяются несущественно.

Иногда, однако, происходит изменение в именовании пакетов или изменение их зависимостей. Такие ситуации не обрабатываются командой `apt-get upgrade`, в результате чего происходит нарушение целостности системы: появляются неудовлетворённые зависимости. Например, переименование пакета MySQL-shared, содержащего динамически загружаемые библиотеки для работы с СУБД MySQL, в libMySQL (отражающая общую тенденцию к наименованию библиотек в дистрибутиве) не приводит к тому, что установка обновлённой версии libMySQL требует удаления старой версии MySQL-shared. Для разрешения этой проблемы существует режим обновления в масштабе дистрибутива – `apt-get dist-upgrade`.

В случае обновления всего дистрибутива APT проведёт сравнение системы с репозиторием и удалит устаревшие пакеты, установит новые версии присутствующих в системе пакетов, а также отследит ситуации с переименованиями пакетов или изменения зависимостей между старыми и новыми версиями программ. Всё, что потребуется поставить (или удалить) дополнительно к уже имеющемуся в системе, будет указано в отчёте apt-get, которым APT предварит само обновление.

Для обновления всей системы рекомендуется использовать команду `apt-get dist-upgrade`.

Важно: На всё время использования Продукта для каждого компьютера, на которой

установлен Продукт, должно быть обеспечено своевременное получение от разработчика обновлений, прошедших соответствующую проверку. Обновления для сертифицированных продуктов распространяются согласно договорам на техническую поддержку.

4.3. Технологический алгоритм КСЗ

Технологический алгоритм КСЗ, представленный на рисунке (Рисунок 1), представляет человеко-машинную систему с множеством входов и выходов.

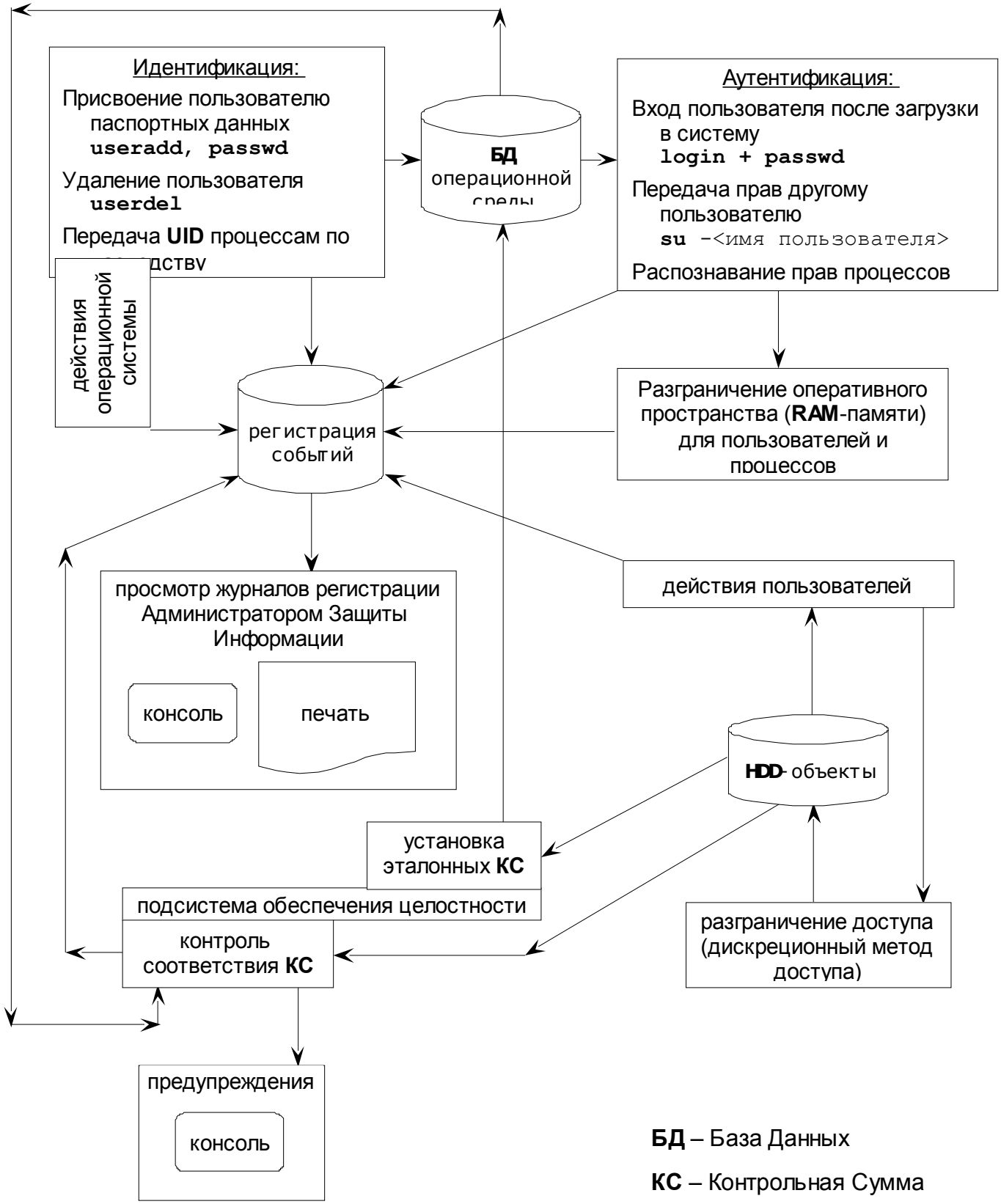

Рисунок 1

4.3.1. Команды управления функциями КСЗ

В командном интерпретаторе bash для управления процессами КСЗ в режиме командной строки

используются следующие команды (в алфавитном порядке):

- cd смена текущего каталога;
- chgrp смена группы, которой принадлежит файл или каталог;
- chmod изменение режима доступа к файлу или каталогу;
- chown смена владельца файла или каталога;
- md5sum вычисление контрольных сумм (CRC) указанных файлов;
- newgrp образование новой группы пользователей
- passwd изменение пароля пользователя для входа в систему;
- shred безопасное удаление файлов
- su запуск интерпретатора командной строки с правами указанного пользователя и его групп (получение прав другого пользователя)
- useradd добавление пользователя
- userdel удаление пользователя
- users вывод информации о пользователях, подключённых к Linux-системе;
- w вывод информации о системе.

4.3.2. Заведение нового пользователя в систему

Заведение нового пользователя в систему обеспечивает АЗИ (Администратор защиты информации) в соответствии с руководящими решениями объекта автоматизации. При этом внутри машинную идентификацию пользователя (то есть присвоение ему кода UID) либо КСЗ обеспечивает автоматически либо АЗИ назначает по своему усмотрению. При включении пользователя в число абонентов АЗИ выдаёт ему регистрационное имя (идентификатор) для входа в систему и пароль, который служит для подтверждения идентификатора пользователя. В дальнейшем КСЗ обеспечивает аутентификацию пользователя, то есть его опознание по имени и паролю. Вводимые пользователем символы пароля не отображаются на экране терминала. В графическом режиме символы пароля заменяются звёздочками.

Администратор системы и/или пользователь могут изменить пароль командой passwd. При вводе этой команды ALT Linux запрашивает ввод текущего пароля, а затем требует ввести новый пароль. Если предложенный пароль слишком прост, ALT Linux может попросить ввести другой. Если предложенный пароль удовлетворителен, ALT Linux просит ввести его снова с тем, чтобы убедиться в корректности ввода пароля.

4.3.3. Вход в систему по сети

Установка средств удалённого администрирования позволяет производить работы с системой удалённо, используя защищённый канал связи (зашифрованное TCP/IP-соединение). Для входа на удалённую ЭВМ используется команда ssh (SSH-клиент), предназначенная для регистрации и выполнения команд на удалённой машине.

Начальные условия: Для соединения двух и более компьютеров необходимо заранее должным образом настроить TCP/IP-стек протоколов (назначены IP-адреса, подключены сетевые кабели, настроена система маршрутизации). Далее предполагается:

- операционная система загружена;
- на связываемых компьютерах связь по сети возможна (то есть предполагается техническая готовность);
- на всех машинах заведены соответствующие пользователи, что означает возможность их регистрации в системе.

5. ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

5.1. Общие сведения

Обмен информацией между ОС ALT Linux и внешними источниками осуществляется информационными сообщениями по локальной вычислительной сети или сети Internet с использованием протоколов TCP/IP, ICMP, FTP, HTTP, POP, SMTP, IMAP, SLIP, PPP, RIP, IPX и NetBIOS.

5.2. Уровни сетевого взаимодействия

5.2.1. Физический уровень

Физический уровень передаёт биты по физическим каналам связи (коаксиальный кабель, витая пара). Этот уровень непосредственно производит передачу данных.

На физическом уровне определяются характеристики электрических сигналов, которые передают дискретную информацию. К этому уровню также относятся характеристики физических сред передачи данных.

Функции физического уровня реализуются сетевым адаптером или последовательным портом.

5.2.2. Канальный уровень

Канальный уровень отвечает за передачу данных между узлами в рамках одной локальной сети. При этом под узлом понимается любое устройство, подключённое к сети.

Этот уровень выполняет адресацию по физическим адресам (MAC-адреса), зашитым в сетевой адаптер изготовителем. Каждый сетевой адаптер имеет свой уникальный MAC-адрес.

Канальный уровень переводит поступившую с верхнего уровня информацию в биты, которые потом будут переданы физическим уровнем по сети. Он разбивает передаваемую информацию на фрагменты данных – кадры.

На этом уровне системы обмениваются именно кадрами. Процесс пересылки следующий: канальный уровень отправляет кадр физическому уровню, который отправляет кадр в сеть. Этот кадр получает каждый узел сети и проверяет, соответствует ли адрес пункта назначения адресу этого узла. Если адреса совпадают, канальный уровень принимает кадр и передаёт наверх вышележащим уровням. Если адреса не совпадают, то он просто игнорирует кадр. Таким образом, сеть на канальном уровне является широковещательной.

Протоколы канального уровня используются компьютерами, мостами, маршрутизаторами. Канальный уровень обеспечивает связь только между компьютерами, соединёнными индивидуальной линией связи.

5.2.3. Сетевой уровень

Данный уровень служит для образования единой транспортной системы, которая объединяет несколько сетей. Другими словами, сетевой уровень обеспечивает межсетевое взаимодействие.

На сетевом уровне термин «сеть» следует понимать как совокупность компьютеров, которые соединены между собой в соответствии с определенной топологией и используют для передачи данных один из протоколов канального уровня.

Сети соединяются специальными устройствами – маршрутизаторами. Маршрутизатор собирает информацию о топологии межсетевых экранов и на основании этой информации пересылает пакеты сетевого уровня в сеть назначения.

Сообщения на сетевом уровне называются пакетами. При этом на сетевом уровне работают несколько видов протоколов. Прежде всего это сетевые протоколы, которые обеспечивают передвижение пакетов по сети, в том числе в другую сеть. Протокол TCP/IP является протоколом сетевого уровня.

5.2.4. Транспортный уровень

На пути от отправителя к получателю пакеты могут быть искажены или утеряны. Некоторые приложения самостоятельно выполняют обработку ошибок при передаче данных, но большинство нет. Транспортный уровень как раз и предназначен для обеспечения надёжности передачи пакетов.

На транспортном уровне определены 5 классов сервиса:

1) Срочность.
2) Восстановление прерванной связи.
3) Наличие средств мультиплексирования нескольких соединений.
4) Обнаружение ошибок.
5) Исправление ошибок.

Уровни сетевого взаимодействия, начиная с транспортного уровня, реализуются на программном уровне компонентами ОС ALT Linux.

5.2.5. Сеансовый уровень

Сеансовый уровень устанавливает и разрывает соединения между компьютерами, управляет диалогом между ними, а также предоставляет средства синхронизации. Средства синхронизации позволяют вставлять определенную контрольную информацию в длинные передачи (точки). Благодаря этому, в случае обрыва связи можно вернуться назад (к последней точке) и продолжить пере-

дачу с места обрыва.

5.2.6. Представительный уровень

Представительный уровень изменяет форму передаваемой информации, но не изменяет её содержания. На данном уровне также выполняется шифрование и дешифрование данных.

5.2.7. Прикладной уровень

Данный уровень представляет собой набор разнообразных протоколов, с помощью которых пользователи сети получают доступ к совместно используемым ресурсам. Единица данных представляет собой сообщение.

5.3. Многоуровневая архитектура стека TCP/IP

Протокол TCP/IP представляются в виде модели, состоящей из 4 уровней:

- Прикладной (уровень приложения);
- Основной (транспортный уровень);
- Межсетевой уровень;
- Сетевой (уровень сетевых интерфейсов).

Каждый из этих уровней выполняет определённую задачу для организации надёжной и производительной работы сети.

5.3.1. Уровень сетевого интерфейса

Данный уровень лежит в основании TCP/IP. Уровень сетевого интерфейса отвечает за отправку в сеть и приём из сети кадров, которые содержат информацию. Кадры передаются по сети как одно целое. Кадр представляет собой блок данных. Пересылка блоков данных в стеке протокола TCP/IP представлена на рисунке (Рисунок 2).

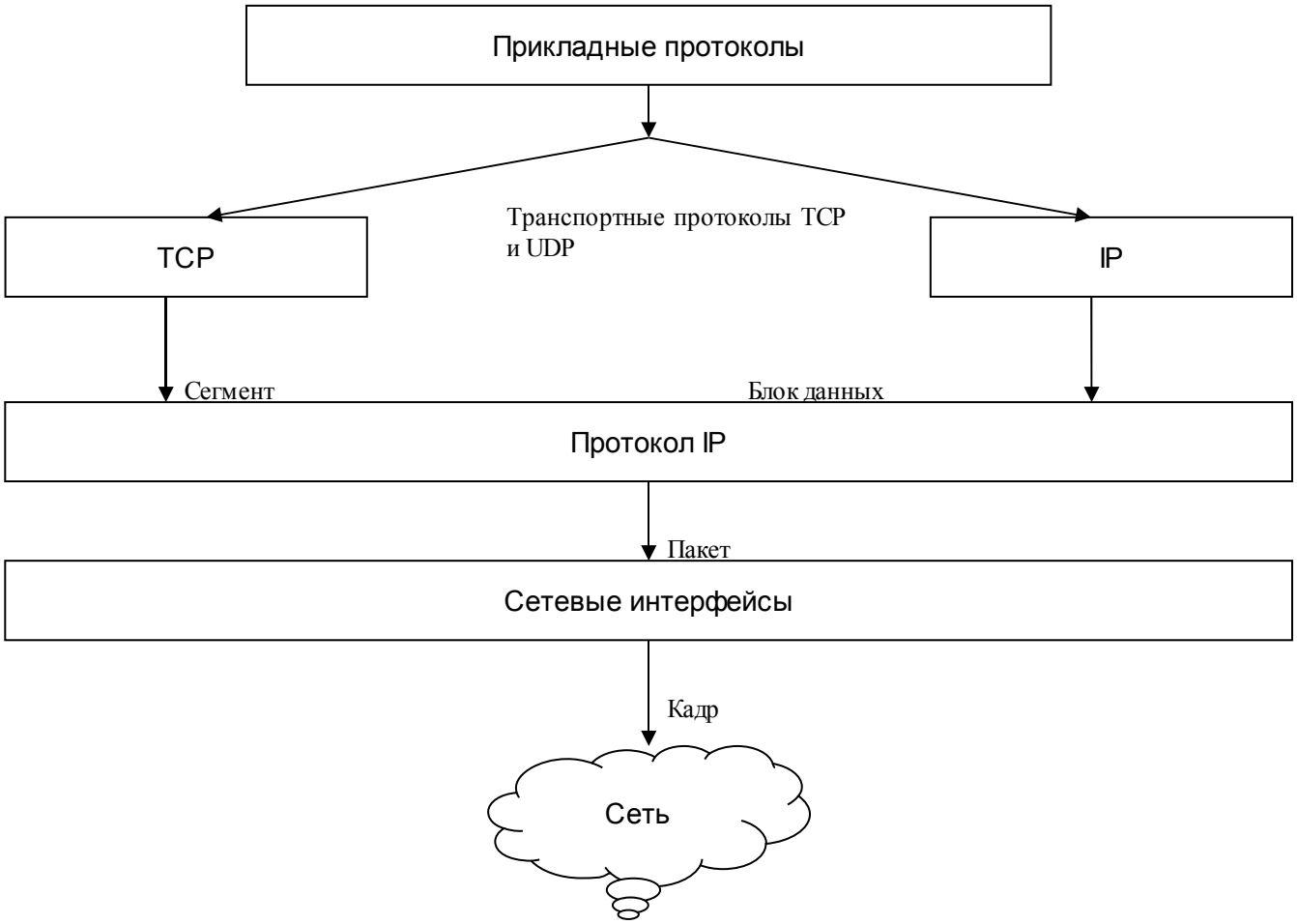

Рисунок 2 – Пересылка блока данных в стеке протокола TCP/IP

5.3.2. Межсетевой уровень

На этом уровне реализуется передача пакетов без установки соединения. Межсетевой уровень обеспечивает перемещение пакетов по сети. Основная функция межсетевого уровня – передача пакетов через составную сеть.

5.3.3. Транспортный уровень

Данный уровень обеспечивает сеансы связи между компьютерами с использованием одного из протоколов: TCP или UDP.

5.3.4. Уровень приложений

Данный уровень является вершиной модели TCP/IP. На этом уровне работают практически все распространённые утилиты и службы.

5.3.5. Структура пакетов TCP и IP

Протокол IP не ориентирован на соединение, поэтому не обеспечивает надежную доставку данных. Поля, описанные в таблице 2 представляют собой IP-заголовок и добавляются к пакету при его получении с транспортного уровня.

Протокол TCP, в отличие от протокола IP, ориентирован на установление соединения и обеспечивает надёжную доставку данных. Структура TCP-пакета описана в таблице 3.

Таблица 2 – Структура заголовка IP-пакета

Поле	Описание
IP-адрес отправителя	Отправитель пакета
IP-адрес получателя	Получатель пакета
Протокол	TCP или UDP
Контрольная сумма	Значение для проверки целостности пакета
Время жизни пакета, TTL (Timt To Live)	Определяет, сколько секунд датаграмма может находиться в сети. Предотвращает бесконечное блуждание пакетов в сети. Значение TTL автоматически уменьшается на одну или более секунд при проходе через каждый маршрутизатор сети.
Версия	Версия протокола IP – 4 или 6
Длина заголовка	Минимальный размер заголовка (4 бита)
Тип обслуживания	Обозначение требуемого для этого пакета качества обслуживания при доставке через маршрутизаторы IP-сети. Здесь определяются приоритет, задержки, пропускная способность (8 бит).
Общая длина	Длина датаграммы IP-протокола (16 бит)
Идентификация	Идентификатор пакета. Если пакет фрагментирован (разбит на части), то все фрагменты имеют одинаковый идентификатор (16 бит)
Фрагментационные флаги	3 бита для флагов фрагментации и 2 бита для текущего использования.
Смещение фрагмента	Указывает на положение фрагментов относительно начала поля данных IP-пакета. Если фрагментации нет, смещение равно 0 (13 бит).
Опции и заполнение	Опции

Таблица 3 – Структура TCP-пакета

Поле	Описание
Порт отправителя	Порт TCP-узла отправителя
Порт получателя	Порт TCP-узла получателя
Порядковый номер	Номер последовательности пакетов
Номер подтверждения	Порядковый номер байта, который локальный узел рассчитывает получить следующим
Длина данных	Длина TCP-пакета
Зарезервировано	Зарезервировано для будущего использования
Флаги	Описание содержимого сегмента
Окно	Показывает доступное место в окне протокола TCP
Контрольная сумма	Значение для проверки целостности пакета
Указатель срочности	При отправке срочных данных в этом поле задается граница области срочности данных

ПРИЛОЖЕНИЕ

6. СТРУКТУРА КАТАЛОГОВ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ОС ALT LINUX

Операционная система ALT Linux включает следующую структуру каталогов (в соответствии со стандартом FHS):

/ – корневой каталог;

/bin – основные исполняемые файлы (доступные всем пользователям);

/boot – неизменяемые файлы для загрузчика;

/dev – файлы устройств;

/etc – специфичная для данного хоста конфигурационная информация;

/home – домашние каталоги пользователей;

/lib – основные разделяемые библиотеки и модули ядра;

/lost+found – служебный каталог;

/media – точка монтирования для сменных файловых систем;

/mnt – точка монтирования для временно монтируемых файловых систем;

/opt – дополнительные пакеты программного обеспечения;

/proc – файловая система на виртуальном устройстве, её файлы содержат информацию о текущем состоянии системы;

/root – домашний каталог пользователя root;

/sbin – системные исполняемые файлы;

/sys – рабочая область для построения ядра, файлы конфигурации;

/tmp – временные файлы;

/usr – каталог, содержащий каталоги и файлы прикладных программ и пакетов, доступных пользователю;

/usr/bin – большая часть команд пользователя;

/usr/etc – команды системного сопровождения;

/usr/games – игры и развлечения;

/usr/include – каталог для стандартных подключаемых файлов;

/usr/lib – библиотеки для программирования и приложений;

/usr/libexec – файлы поддержки Linux-программ;

/usr/local – каталог для локального ПО;

/usr/sbin – необязательные стандартные системные команды;

/usr/share – архитектурно-независимые данные;

/usr/src – исходные коды;

/usr/tmp – каталог для временного хранения файлов;

/usr/X11R6 – Файлы, используемые старой версией графической подсистемы X Window. В настоящее время большинство компонентов X Window располагаются в /usr; структура подкаталогов /usr/X11R6 идентична структуре /usr.

/var – каталог, содержащий рабочие и журнальные файлы;

/var/adm – учетные файлы, журналы регистрации использования ресурсов;

/var/cache – данные кэша приложений;

/var/db – БД с файлами;

/var/empty – служебный каталог;

/var/lib – переменные данные о состоянии системы;

/var/local – локальные файлы;

/var/lock – файлы блокирования;

/var/log – каталоги и файлы протоколов;

/var/mail – почтовые ящики пользователей;

/var/nis – файлы сетевой информационной системы NIS;

/var/opt – переменные данные для /opt;

/var/run – переменные данные времени выполнения;

/var/spool – очереди данных для приложений;

/var/tmp – временные файлы, сохраняемые между перезапусками системы;

/var/www – каталог для хранения настроек и кэша web-файлов;

/var/yp – файлы базы данных сетевой информационной системы NIS;

Данная структура каталогов создаётся автоматически при установке с инсталляционного диска.

7. ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

API	(Application Programming Interface) интерфейс прикладного программирования.
ATA	(Advanced Technology Attachment) серия интерфейсов и протоколов, используемых для организации доступа к жёстким дискам в портативных компьютерах.
BIOS	(Basic Input/Output System) базовая система ввода-вывода.
DMA	(Direct Memory Access) прямой доступ к памяти.
DMAPI	Интерфейс для поддержки иерархического управления хранением.
DNS	(Domain Name System) служба имён доменов.
FDC	(Floppy Disk Controller) контроллер накопителя на гибких магнитных дисках.
FHS	(Filesystem Hierarchy Standard) – стандарт на файловую систему.
FIFO	(First-In, First-Out) – дисциплина очереди «первый вошел – первый вышел».
FTP	(File Transfer Protocol) протокол передачи файлов.
GID	(Group Identifier) идентификатор группы.
GPS	(Global Position System) глобальная система распознавания положения (рекогносцировки).
HDD	(Hard Disk Drive) накопитель на жёстком диске.
HTTP	(HyperText Transfer Protocol) протокол передачи гипертекстовых файлов.
HTTPS	(HyperText Transmission Protocol Secure) протокол защищённой передачи гипертекстов.
IBM PC	(IBM Personal Computer) – персональный компьютер, совместимый с IBM.
IDE	(Integrated Device Electronics) встроенный интерфейс устройств.
IMAP	(Interactive Mail Access Protocol) протокол интерактивного доступа к электронной почте.
IPX	(Internetwork Packet Exchange) межсетевой пакетный обмен.
JFS	(Journal File System) журналируемая файловая система.
KDS	(Key Distribution Server) центр распространения ключей.
CUPS	(Common Unix Printing System) система печати на операционных системах UNIX/ Linux.
LAN	(Local Area Network) локальная вычислительная сеть.
LILO	(Linux Loader) программа загрузки ОС Linux.
LPT	Параллельный порт (используется обычно для подключения принтера).
MTA	(Mail Transfer Agent) агент доставки почты.
NETBIOS	(NetBIOS Extended User Interface) транспортный протокол, используемый всеми сетевыми ОС фирмы Microsoft.
NFS	(Network File System) сетевая файловая система.
NIS	(Network Information Services) сетевые информационные службы.
NTP	(Network Time Protocol) синхронизирующий сетевой протокол.
PID	(Process Identificator) идентификатор процесса.
PPP	(Point-to-Point Protocol) протокол передачи от точки к точке, протокол двухточечного соединения.
PPTP	(Point-to-Point Tunneling Protocol) протокол, используемый для установки VPN-каналов поверх локальной или глобальной сети.
RAID	(Redundant Array of Independent Disks) матрица независимых дисковых накопителей с избыточностью.
RAM	(Random-Access Memory) память (запоминающее устройство) с произвольной выборкой.
ReiserFS	(Reiser File System) файловая система ReiserFS.
RIP	(Routing Information Protocol) протокол для маршрутизации пакетов в компьютерной сети.
ROM	(Read-Only Memory) ПЗУ, постоянная память; постоянное запоминающее устрой-

	ство.
SATA	(Serial Advanced Technology Attachment) см. ATA
SCSI	(Small Computer Systems Interface) интерфейс малых компьютерных систем систем.
SGID	(Set Group ID) специальные права доступа пользователя.
SLIP	(Serial Line Internet Protocol) межсетевой протокол для последовательного канала.
SMTP	(Simple Mail Transfer Protocol) простой протокол электронной почты.
SSH	(Security Shell) безопасная оболочка для удалённого администрирования систем.
SUID	(Set User ID) специальные права доступа пользователя.
TCP/IP	(Transmission Control Protocol/Internet Protocol) протокол управления передачей/протокол Internet, стек протоколов Internet.
UDP	(User Datagram Protocol) протокол передачи дейтаграмм пользователя.
UID	(User Identifier) идентификатор пользователя.
UPS	(Uninterruptible Power Supply) источник бесперебойного питания.
USB	(Universal Serial Bus) универсальная последовательная шина.
VFS	(Virtual File System) виртуальная файловая система.
VPN	(Virtual Private Network) виртуальная частная сеть.
WAN	(Wide-Area Network) глобальная сеть (сеть, обеспечивающая передачу информации на значительные расстояния с использованием коммутируемых и выделенных линий или специальных каналов связи).
XFS	Высокопроизводительная журналируемая файловая система.
АЗИ	Администратор защиты информации
ИБП	Источник бесперебойного питания
КСЗ	Комплекс средств защиты
ЛВС	Локальная вычислительная сеть
ОС	Операционная система
ПЗУ	Постоянное запоминающее устройство
ПО	Программное обеспечение
ПЭВМ	Персональная электронная вычислительная машина
РМ	Рабочее место
ФС	Файловая система
ЭВМ	Электронная вычислительная машина

Лист регистрации изменений

Лист регистрации изменений									
Изм.	**Номер листов (страниц)**				**Всего листов (страниц) в докум.**	**№ документа**	**Входящий № сопроводительного докум. и дата**	**Подп.**	**Дата**
	измененных	**замененных**	**новых**	**аннулированных**					