УДК 004.451.9

Авдеева Т.С.

Студент

РГУ нефти и газа (НИУ) имени И. М. Губкина

Паршинцева А.Г.

Студент

РГУ нефти и газа (НИУ) имени И. М. Губкина

г. Москва

ОБЗОР СТРУКТУРЫ И ОСОБЕННОСТЕЙ RPM-ПАКЕТОВ В ОТЕЧЕСТВЕННЫХ ОС: ОС ALT, РЭД ОС И ROSA LINUX

Аннотация: В статье представлен обзор структуры и особенностей RPM-пакетов, которые используются в отечественных операционных системах (ОС) на примере ОС ALT, РЭД ОС и Rosa Linux. Особое внимание уделено инструментам, используемым для работы с rpm в представленных ОС, особенностям сборки, установки и удаления пакетов, структуре spec-файлов. В результате анализа выявлены ключевые различия и сходства между пакетами и представлены в виде таблицы. Работа подчёркивает важность стандартизированных подходов к управлению пакетами и необходимость дальнейших исследований для оптимизации процесса установки и обновления программного обеспечения. Статья будет полезна специалистам в области разработки и сопровождения программного обеспечения, а также исследователям, интересующимся особенностями работы с RPM-пакетами в отечественных ОС.

Ключевые слова: RPM-пакет, операционная система, управление пакетами, spec-файл.

УДК 004.451.9

Avdeeva T.S

Student

Gubkin Russian State University of Oil and Gas (NRU)

Parshintseva A.G.

Student

Gubkin Russian State University of Oil and Gas (NRU)

Moscow

OVERVIEW OF THE STRUCTURE AND FEATURES OF RPM PACKAGES IN DOMESTIC OPERATING SYSTEMS: ALT OS, RED OS AND ROSA LINUX

Abstract: *The article presents an overview of the structure and features of RPM packages used in domestic operating systems (OS) on the example of Alt OS, RED OS and Rosa Linux. Special attention is paid to the tools used to work with rpm in the presented OS, peculiarities of building, installing and uninstalling packages, and the structure of spec-files. As a result of the analysis the key differences and similarities between the packages are identified and presented in the form of a table. The paper emphasizes the importance of standardized approaches to package management and the need for further research to optimize the process of software installation and upgrade. The article will be useful to specialists in the field of software development and maintenance, as well as researchers interested in the peculiarities of working with RPM-packages in domestic operating systems.*

Keywords: *RPM-package, operating system, package management, spec-file.*

Введение

Последние десятилетия в нашей стране наблюдается активное развитие рынка IT в направлении импортозамещения – особенно ощутима необходимость нахождения собственных решений вследствие введения санкций и ограничений на использование некоторых зарубежных программных продуктов в России. Этот процесс является длительным и сложным, однако уже появляются достойные разработки-альтернативы глобальным программным пакетам для использования в критически-важных областях. Linux – семейство ОС с открытым исходным кодом [2]. Благодаря доступности для изменений существует несколько различных версий-редакций ОС, именуемые дистрибутивами и включающие различные варианты программного обеспечения. К основным российским решениям можно отнести: Alt Linux, РЭД ОС, Rosa.

Все из названных ОС используют rpm-пакеты, что позволяет им легко интегрироваться с существующими системами и упрощает процесс установки и обновления ПО, соответствуя принципам простоты и удобства в управлении собственными системами. Эти характеристики делают их привлекательными для пользователей, которые ищут надёжные и функциональные решения, однако методы достижения целей в зависимости от выбранного ОС могут различаться.

Менеджер пакетов

В общем случае для управления программами и поддержания целостности системы используются менеджеры пакетов: с его точки зрения ПО представляет собой набор компонентов — программных пакетов, включающих в себя набор исполняемых программ и вспомогательных файлов, необходимых для корректной работы.

Благодаря подобранному набору программных средств, пользователь Linux не должен обращаться к установочным процедурам отдельных

программ или непосредственно работать с каталогами, где установлены исполняемые файлы или компоненты программ [5, С.93].

RPM – один из низкоуровневых менеджеров пакетов, использующийся всех для сборки, проверки, установки, обновления и удаления пакетов с одноимённым названием: они содержат архив с файлами и метаданные, используемые для установки и удаления файлов архива. Метаданные включают сценарии, атрибуты файлов и информацию с описанием конкретного пакета [6].

RPM-пакет представляет собой архив cpio, включающий в себя скомпилированные исполняемые файлы в виде библиотек и данных, а также заголовок, который содержит в себе метаданные о пакете (название, версия, группа и т.п.), используемые для определения зависимостей и локализации файлов.

Различают два вида пакетов RPM [7]:

- Пакет с исходным кодом — SRPM-пакет (.src.rpm), содержащий один или несколько архивов с исходными кодами, spec-файл, а также различные патчи и дополнения. Такие пакеты используются разработчиками для создания двоичных пакетов и не предназначены для установки.

Сборка осуществляется командой: *rpmbuild --rebuild package...src.rpm* [4, С. 23].

Структура RPM-пакета Red Hat приведена на Рисунке 1:

Рисунок 1 – Структура каталогов rpm RH

Пакет является крайне гибким, потому его можно пересобрать в любом другом дистрибутиве на основе RPM и в любой другой архитектуре.

Spec-файл в данном случае можно рассматривать как ключевой элемент, определяющий «правила», написанные в специальном текстовом формате и использующиеся для фактической сборки RPM-пакета: он определяет все действия, которые должны быть выполнены при управлении пакетами. Синтаксические определения внутри spec-файла имеют значения, задающие порядок сборки, номер версии, информацию о зависимостях и вообще всю информацию о пакете, которая может быть впоследствии запрошена из БД RPM. Пример типового файла spec представлен на Рисунке 2.

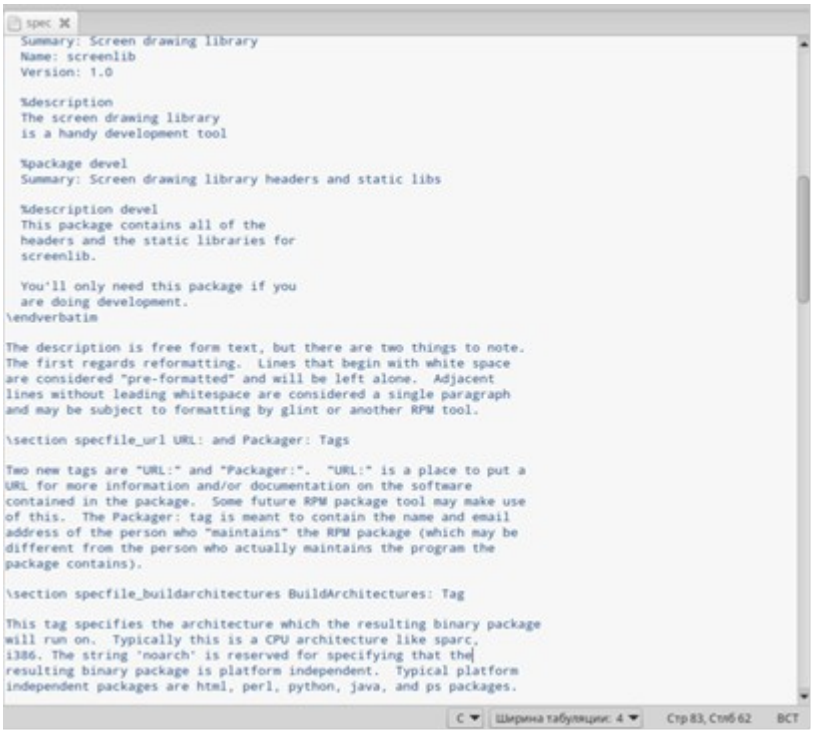

Рисунок 2 – spec-файл *screenlib* ОС ALT

Он содержит в себе преамбулу(заголовок) и тело спецификации, содержащая основную часть инструкций согласно общепринятому стандарту.

Собранный двоичный пакет — RPM-пакет (.rpm), содержащий файлы, полученные из исходников и патчей в соответствии со спецификацией, позволяющей системе RPM правильно установить и настроить приложение. По своей сути именно двоичные пакеты являются готовыми к установке приложениями, возникшие на базе SRPM-пакета. В результате пользователи получают полностью готовое к использованию программное обеспечение, которое можно легко установить и настроить без необходимости вручную собирать и настраивать компоненты.

Такие пакеты можно устанавливать командой: rpm -Uvh package...rpm [4, С.23].

RPM позволяет работать с репозиториями и получать сведения о количестве пакетов в системе. Для просмотра списка установленных пакетов можно использовать команду rpm -qa.

Основные команды менеджера RPM (Red Hat):

- Установка пакета: *rpm install имя_пакета*
- Обновления пакета:
  a) С принудительной установкой: *rpm -Uvh имя_пакета*
  b) Без принудительной установки: *rpm -Fvh имя_пакета*
- Обновление всех установленных пакетов*: rpm -U* или *rpm -F*
- Удаление пакета: *rpm -e имя_пакета*

Рассмотрим основные особенности работы с rpm-пакетами в отечественных ОС.

ОС ALT — набор российских ОС, основанных на RPM, построенных поверх ядра Linux и репозитория Sisyphus [1].

1) обширный набор макросов для сборки пакетов различного назначения:

a) патчи для повышения безопасности могут закрывать выявленные уязвимости, что особенно актуально для систем, работающих в открытых сетях;

б) патчи улучшения производительности часто включают оптимизации кода, что позволяет использовать ресурсы системы более эффективно;

в) обеспечение совместимости: иногда новые версии библиотек могут нарушить работу старых приложений – патчи помогают поддерживать необходимую функциональность.

В настоящее время информация о них полностью описана исключительно в самих файлах с определениями макросов, которые можно найти в каталогах /usr/lib/rpm и /etc/rpm/macros.d.

2) отличающееся поведение сценария «по умолчанию»: позволяет упростить процесс создания конфигурационных файлов для сборки пакетов и уменьшить вероятность ошибок при создании spec-файлов прямо пропорционально количеству повторяющегося кода путём предоставления готовых шаблонов и настроек.

3) ОС ALT есть свои особенности в поведении самого rpm – при сборке SRPM-пакета через *rpmbuild* возникают очевидные сложности [4, С.23]: установка сборочных зависимостей вручную – на рабочей системе можно упустить необходимое и достаточное количество зависимостей и засорение системы.

Чтобы избежать этих сложностей, в Альт Платформа используется две технологии [9]: Hasher (для сборки в изолированном окружении), Gear (для сборки пакетов из репозитория).

Еще одной интересной особенностью и решением приведённых проблем всех ALT-систем является использование в качестве пакетного менеджера по умолчанию APT, что обеспечивает совместимость с Debian и всех основанных на нём ОС.

Apt предлагает простые команды для установки, обновления и удаления пакетов, что делает его интуитивно понятным. Так решение проблем с установлением зависимостей решается командой *apt-get update.*

Также данный менеджер пакетов позволяет управлять пакетами через один интерфейс, что упрощает администрирование системы: в управление же используются репозитории, что обеспечивает актуальность программного обеспечения.

Основные команды:

- Установка пакета: *apt-get install имя_пакета*
- Обновления уже установленного пакета или группы пакетов: *apt-get install*
- Обновление всех установленных пакетов*: apt-get upgrade*
- Удаление пакета: *apt-get remove имя_пакета*

РЕД ОС — ОС семейства Linux, построенная на пакетной базе RPM-формата и соответствующая требованиям POSIX и LSB 4.1 (Linux Standard Base) [8]. Для автоматизации процесса управления программными продуктами в РЕД ОС в большем количестве случаев применяется система управления пакетами DNF.

Фактически, DNF представляет из себя оболочку rpm, обеспечивающую работу с репозиториями, который умеет запрашивать информацию о пакетах, получать пакеты из репозиториев, устанавливать и удалять их, используя автоматическое разрешение зависимостей, а также обновлять целиком систему до последних версий пакетов, аналогично APT в ОС ALT.

В случае обновления уже установленного пакета или группы пакетов дополнительно проверяет, не обновилась ли версия пакета в репозитории по сравнению с установленным в системе. *Dnf update* используется для обновления отдельных пакетов в системе, а именно позволяет установить новые версии пакетов, заменить старые версии на более новые или обновить пакеты до последних доступных версий. *Dnf upgrade* выполняет более широкий спектр действий: не только обновляет отдельные пакеты, но ещё проверяет наличие доступных обновлений для всей системы в

целом – если система может быть обновлена, команда выполнит это обновление.

С его помощью можно установить и отдельный бинарный rpm-пакет, не входящий ни в один из репозиториев (например, полученный из Интернет).

Для того, чтобы не нарушать целостность системы, будут удалены и все пакеты, зависящие от удаляемого. [9]

Основные команды:

- Установка пакета выполняется командой: *dnf install <имя_пакета>*.
- Проверка наличия обновлений системы: *dnf check-update*
- Обновления отдельного пакета: *dnf update <имя_пакета>*
- Обновление всех установленных пакетов*: apt-get upgrade*
- Обновление всех системных пакетов: *dnf update* или *dnf upgrade*
- Обновление группы пакетов: *dnf groupupdate*
- Удаление пакета: *dnf remove <имя_пакета>*.

Rosa Linux — линейка дистрибутивов Linux (изначально основанных на Mandriva), разработку которых ведёт российская компания «НТЦ ИТ РОСА». На «ROSA Linux» основаны проекты Mag ОС Linux [3].

Управление программными пакетами осуществляется с помощью утилит командной строки rpm и dnf.

Стоит отметить, что в ROSA не рекомендуется явно указывать упаковщика и поставщика в спецификации – эти теги устанавливаются автоматически системой сборки ABF.

Так как нет специфического менеджера пакета (DNF рассмотрен ранее).

Сравнение

В Таблице 1 приведено сравнение особенностей rpm-пакетов и работы с ними в рассмотренных ОС.

	ОС ALT	РЭД ОС	ROSA LINUX	RED HAT
Инструменты для работы с rpm пакетами	1. RPM; 2. APT	1. RPM; 2. DNF		1. RPM
Структура каталогов	**~/rpm/BUILD** — каталог для собранных исходников; **~/rpm/tmp** — для временных файлов, которые создаются программой rpm во время сборки пакетов; 1 **~/rpm/RPMS** — каталоги для различных архитектур, для которых планируется выполнять сборку (обычно это i586 или x86_64, но может быть также sparc, alpha или ppc).	**BUILD** — содержит все файлы, появляющиеся при сборке пакета. **RPMS** — здесь появятся готовые бинарные пакеты (расширение .rpm). **SOURCES** — здесь должны находиться исходники, из которых и будут собираться RPM-пакеты. **SPECS** — папка для хранения spec-файла, который описывает процесс сборки. **SRPMS** — здесь находятся пакеты с исходниками (расширение .src.rpm).	**%_topdir** — каталог верхнего уровня, содержащий в себе стандартную иерархию каталогов RPM, в которой производится сборка пакета. По умолчанию значение этого макроса равно $HOME/RPM; **%_sourcedir** — макрос, указывающий на каталог с исходными кодами программы, собираемой в пакет. Значение по умолчанию — %_topdir/SOURCES; **%_specdir** — каталог, содержащий spec-файл. Значение по умолчанию — %_topdir/SPECS; 1 **%_srcrpmdir** — каталог, в котором будет содержаться собранный пакет с исходным кодом (src.rpm). Значение по умолчанию — %_topdir/SRPMS; 1 **%_rpmdir/ %_arch** —	~/rpm/BUILD для собранных исходников. ~/rpm/RPMS каталоги для разных архитектур, куда кладутся бинарные пакеты после сборки ~/rpm/RPMS/ i586 — каталог для хранения rpm-пакетов для процессоров i586; ~/rpm/RPMS/ x86_64 каталог для хранения rpm-пакетов для процессоров x86_64; ~/rpm/RPMS/ noarch — каталог для хранения rpm-пакетов, не зависящих от архитектуры процессора. ~/rpm/ SOURCES. файлы исходного кода. ~/rpm/SPECS. В этом каталоге хранятся спек-файлы. ~/rpm/SRPMS. Каталог src.rpm-пакетов. ~/rpm/tmp.

			каталог, содержащий подкаталоги с именами различных архитектур (%_arch), для которых производится сборка пакета. Собранный бинарный пакет будет помещён в один из этих подкаталогов. Значение макроса %_rpmdir по умолчанию — %_topdir/RPMS.	Здесь находятся временные файлы.
Сборка пакета	1. сборка командой rpmbuild; 2. технология Haser (для сборки в изолированном окружении); 3. технология Gear (для сборки пакетов из репозитория Git).	Пакеты RPM создаются с помощью утилиты rpmbuild.	1. Утилита rpmbuild 2. Автоматическая система сборки ABF	Утилита rpmbuild
Права для сборки	По умолчанию сборка rpm-пакетов запрещена суперпользователю (root) с целью повышения безопасности. Рекомендуется проводить сборку с правами непривилегированного пользователя, поскольку при сборке с правами root имеется высокая вероятность повреждения системы.			
Ошибки при сборке	В некоторых случаях сборка может быть автоматически прервана, и в результате могут быть удалены только те файлы, которые уже были созданы в процессе сборки, но не были установлены на систему. В других случаях могут быть удалены как созданные, так и установленные файлы.	При неправильной инсталляции пакета, когда он ставится не во временный каталог есть вероятность, что часть файлов будет потеряна. Также при ошибке «Installed (but unpackaged) file(s) found» это может означать, что в новой версии пакета появились новые файлы, отсутствующие ранее, и rpmbuild не знает, что с ними делать.	При ошибке сборки пакета RPM в Rosa все файлы, созданные командой rpmbuild, не удаляются — так же, как и при успешной сборке. Чтобы быстро удалять эти файлы, можно создать скрипт cleanup.sh и поместить его в папку ~/rpmbuild/SPECS.	Если пакет собран через rpm с ошибками, то все файлы, созданные командой rpmbuild, не удаляются — так же, как и при успешной сборке.

		Если эти файлы нужны, их необходимо добавить в секцию %files spec-файла.		
Установка или обновление пакета	Установка пакета с помощью APT выполняется с помощью утилиты apt-get, 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.	Установка пакета выполняется командой: dnf install <имя_пакета> DNF позволяет устанавливать в систему пакеты, требующие для работы другие, пока ещё не установленные. В этом случае они определяют, какие пакеты необходимо установить, и устанавливают их, пользуясь всеми доступными репозиториями.	Осуществляется с помощью менеджера DNF командой sudo dnf install <имя_пакета>	Установка пакета: rpm install имя_пакета Обновления пакета: a) С принудительной установкой: rpm -Uvh имя_пакета b) Без принудительной установки: rpm -Fvh имя_пакета Обновление всех установленных пакетов: rpm -U или rpm -F
Необходимость удаления файлов предыдущей версии	При сборке новой версии RPM-пакета в ОС ALT не обязательно удалять предыдущие файлы, так как для обновления можно использовать команду apt-get, которая проведёт сравнение системы с репозиторием и удалит устаревшие пакеты, установив новые версии.	В большинстве случаев удалять предыдущие версии не требуется. Однако, при сборке пакета, зависящего от предыдущих версий программы, могут возникнуть конфликты версий и проблемы с установкой. В таких случаях может потребоваться удалить предыдущие версии перед установкой новой.	При сборке новой версии RPM-пакета для Rosa Linux рекомендуется удалять предыдущие файлы.	Удаление прошлых версий пакетов в RPM не обязательно, так как после обновления любого пакета программа автоматически сохранит резервную копию старой его версии. Однако принудительное удаление пакета без учёта зависимостей может поломать работу других пакетов в ОП.

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. В случае удаления пакета, который относится к базовым компонентам системы, apt-get потребует дополнительного подтверждения производимой операции с целью предотвратить возможную случайную ошибку.	Для удаления пакета используется следующая команда: dnf remove <имя_пакета> Для того чтобы не нарушать целостность системы, будут удалены и все пакеты, зависящие от удаляемого.	Для удаления RPM-пакетов в Rosa Linux можно использовать инструмент urpme. Также для удаления пакетов в дистрибутиве ROSA платформы 2021.1 и новее можно использовать пакетный менеджер dnf.	Удаление пакета: rpm -e имя_пакета
Внутри spec-файла				
Name – базовое имя пакета	Выбирается, исходя из названия главного бинарного пакета.			Имя не должно содержать пробелов, табуляций и символов новой строки. Однако, допустимо использовать дефис.
Version – номер версии программного обеспечения	Версия upstream-кода.			
Release	Должно иметь вид в простых случаях — altN, а в сложных — altN[суффикс].	Увеличивается с каждой новой сборкой пакета, которая может быть связана с применением дополнительных патчей, изменениями внесёнными в spec-файл и даже тривиальным обновлением пиктограммы.	Как правило, при первой сборке конкретной версии пакета значение устанавливается равным 1 и увеличивается на единицу с каждой новой сборкой пакета. После цифры указывают суффикс макросом %{? dist}	

Summary – краткое описание пакета	Значение тэга Summary должно начинаться с заглавной буквы. В конце Summary не должно быть точки			Для задания национальных описаний
License	Лицензия должна быть указана в точности так, как сформулировано в upstream-пакете	Лицензия, под которой распространяется программа (обычно указано в самом пакете). Раньше в ходу был также тег Copyright, но сейчас он не используется.		
URL	В тэге Url настоятельно рекомендуется указывать действующий URL домашней страницы проекта, либо если таковой нет — любого другого места, где можно получить архив с исходным кодом.			
%description	Произвольное количество строк и параграфов. Текст содержит полное описание программного обеспечения, которое помогает пользователю решить нужно ли устанавливать данный пакет или нет.			
Source	Если сборка производится без использования gear, то в Source настоятельно рекомендуется указывать действующий.	Исходные тексты, тарболы, просто файлы. Рекомендуется указывать.		
Раздел подготовки к сборке (prep)	%prep %setup распаковывает исходный код перед компиляцией.	%prep Команды для подготовки программного обеспечения к сборке, например, распаковка архива, указанного в Source или Source0.	%prep %setup -q -a 1 Это встроенный макрос, который выполняет следующие действия: переходит в дерево сборки; распаковывает исходный код (-q означает, что распаковка сопровождается минимальным выводом информации); изменяет владельца и права доступа к файлам с исходным кодом. %patch0 -p1 Макрос, ответственный за применение патчей к исходникам.	%prep %setup -q

Раздел сборки	%build %configure %make_build	%build %configure %make	
Команды установки/копирования файлов из сборочного каталога в псевдо-корневую директорию	%install		%install %makeinstall
%files	Список файлов для каждого двоичного пакета (если их создаётся несколько по одной спецификации), которые будут установлены в системе конечного пользователя.	Список файлов должен быть написан в spec-файле вручную. Он может быть создан из списка файлов, созданных программой rpm в дереве каталога сборки.	Список должен быть исчерпывающим, чтобы утилита точно знала, что именно пакет устанавливает.
%clean	автоматически очищает BuildRoot пакета (с помощью макроса %clean_buildroot). Таким образом, ручная очистка BuildRoot является ненужной	Spec-cleaner удаляет устаревшие и ненужные декларации — buildroot; а также изменяет оформление использования макросов и переменных %{const}, %{macro}. Такие изменения производятся только для макросов и переменных, определенных в самом rpm (а точнее, перечисленных непосредственно в скрипте). Также Spec-cleaner способен производить замену устаревших версий макросов на современные аналоги.	%clean rm -rf $RPM_BUILD_ROOT
Список изменений, произошедших в пакете между	автоматически подставляет в начало каждой	В changelog'e вы указывает изменения в пакете по сравнению с предыдущей версией. Синтаксис его примерно следующий:	

сборками разных версий или релизов %changelog	секции %files и в начало каждого файла, включаемого с помощью %files -f, директиву %defattr с о значением макроса %_defattr. Таким образом, ручное указание %defattr яв ляется излишним.	%changelog * День недели Месяц День Год Имя <Почта> - 2.4.8-4 - update desktop patch

Заключение

В данной статье был представлен обзор особенностей и структуры RPM-пакетов в отечественных ОС — ОС ALT, РЭД ОС и Rosa Linux.

Все три дистрибутива используют RPM-пакеты как основной формат для распространения программного обеспечения, однако с различиями в подходах к сборке, установке и обслуживанию пакетов.

Ключевыми инструментами для работы с RPM-пакетами в рассматриваемых дистрибутивах являются системы управления пакетами, такие как APT и DNF. Они упрощают процесс управления программным обеспечением, позволяя автоматически разрешать зависимости, что значительно повышает удобство и безопасность установки и обновления пакетов. Spec-файлы, хотя и имеют схожую структуру, могут различаться в зависимости от особенностей работы с пакетами, используемых макросов. В целом, работа с RPM-пакетами в отечественных дистрибутивах Linux предполагает определенные сложности и нюансы, однако при правильном подходе и использовании современных инструментов управления пакетами это становится более управляемым процессом.

СПИСОК ИСТОЧНИКОВ

1. ALT Linux / [Электронный ресурс] // Википедия: [сайт]. — URL: https://en.wikipedia.org/wiki/ALT_Linux (дата обращения: 26.10.2024).

2. Linux (стабильная версия, проверенная 25 октября 2024) / [Электронный ресурс] // Википедия : [сайт]. — URL: https://ru.wikipedia.org/wiki/Linux (дата обращения: 15.10.2024).

3. Rosa Linux / [Электронный ресурс] // РУВИКИ: [сайт]. — URL: https://ru.ruwiki.ru/wiki/Rosa_Linux (дата обращения: 26.11.2024).

4. АЛЬТ ПЛАТФОРМА Руководство пользователя Ред. 1.0 / [Электронный ресурс] // ООО «БАЗАЛЬТ СПО» : [сайт]. — URL: https://www.basealt.ru/fileadmin/user_upload/manual/ALT_Platform_guide.pdf?ysclid=m3d4cssi9z186786882 (дата обращения: 1.11.2024).

5. Д. Аленичев, А. Боковой, А. Бояршинов, В. Виниченко, А. Колотов, Г. Курячий, Д. Левин, К. Маслинский, М. Отставнов, А. Смирнов, М. Уэлш (M. Welsh) и др. ALT Linux изнутри [Текст] / Дмитрий Аленичев, Александр Боковой, Антон Бояршинов, Вадим Виниченко, Александр Колотов, Георгий Курячий, Дмитрий Левин, Кирилл Маслинский, Максим Отставнов, Алексей Смирнов, Мэтт Уэлш (Matt Welsh) и др. — в серии: Библиотека ALT Linux. — Москва: Издательский дом ДМК-пресс, 2006 — 410 с.

6. Назначение RPM / [Электронный ресурс] // РЭДОС : [сайт]. — URL: https://redos.red-soft.ru/base/redos-7_3/7_3-base-consept/7_3-sys-dnf/7_3-manag-pack-rpm/7_3-rpm-view/?nocache=1729587239505 (дата обращения: 25.10.2024).

7. Построение пакета RPM для Rosa Linux на практике / [Электронный ресурс] // ХАБР : [сайт]. — URL: https://habr.com/ru/articles/464993/ (дата обращения: 28.11.2024).

8. Работа с пакетами. RPM-пакет / [Электронный ресурс] // AltLinix : [сайт]. — URL:

https://docs.altlinux.org/ru-RU/alt-platform/10.0/html/alt-platform/ch05.html (дата обращения: 25.10.2024).

9. РЕД ОС / [Электронный ресурс] // РЕДОС: [сайт]. — URL: https://redos.red-soft.ru/product/red-os/?ysclid=m34ws4iu88855697294 (дата обращения: 26.10.2024).

10. Уймин, А. Г. Демонстрационный экзамен базового уровня. Сетевое и системное администрирование: Практикум. Учебное пособие для вузов / А. Г. Уймин. – Санкт-Петербург: Издательство "Лань", 2024. – 116 с. – (Высшее образование). – ISBN 978-5-507-48647-2. – EDN BZJRIQ.