МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ федеральное государственное бюджетное образовательное учреждение   
высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных систем и технологий

Кафедра «Вычислительная техника»

Лабораторная работа №1

по дисциплине “Операционные системы”

**Обзор учебных, разрабатываемых и современных операционных систем.**

“Kali Linux”

Студенты:

группы ИВТАПбд-31

Казаков К.Г.  
Вершинин Д.В.

Научный руководитель:

Валюх В.В.

Ульяновск

2018 г.

**Введение**

Вопросы безопасности операционных систем построенных организационных сетей всегда были наиболее важным аспектом администрирования. Для диагностики наличия уязвимостей нужно было провести большую работу, и нередко даже после нее оставались лазейки для злоумышленников. Такие манипуляции заметно упростились с появлением Kali Linux. Инструкция по применению ее в большинстве своем доступна лишь на английском языке. Такая система не подходит для постоянного домашнего использования, тем более она не рекомендуется новичкам. Это профессиональный инструмент, которым нужно учиться пользоваться, требующий тонкой настройки. А для этого необходим достаточный опыт и полное понимание выполняемых действий.

**Краткое описание ОС**

Разработчик Offensive Security

Семейство ОС GNU/Linux

Основана на Debian GNU/Linux

Последняя версия 2018.4 (20 октября 2018 года)

Менеджеры пакетов dpkg

Поддерживаемые платформы x86, x86-64, armel, armhf

Тип ядра Монолитное ядро Linux

Интерфейс KDE, Fluxbox, GNOME, MATE

Лицензия Различное

Состояние Активное

Репозиторий исходного кода pkg.kali.org

Предыдущая BackTrack[d]

Веб-сайт kali.org

Kali Linux — GNU/Linux-LiveCD, возникший как результат слияния WHAX и Auditor Security Collection. Проект создали Мати Ахарони (Mati Aharoni) и Макс Мозер (Max Moser). Предназначен прежде всего для проведения тестов на безопасность. Наследник развивавшегося до 2013 года на базе Knoppix дистрибутива BackTrack.

**Архитектура ОС**

*Операционная система Kali Linux* состоит из двух основных частей: собственно*ядра системы - Linux* и набора *программ* и инструментов проекта *GNU*, включающего в себя элементы *звуковой и графической подсистем, рабочие столы*, различные прикладные *приложения* и т.п.

**Ядро Linux и его загрузка**

*Linux* - монолитное (большое) *ядро операционной системы*. Оно включает в себя все необходимые для работы *системы* ***драйверы*** *(графические X11 и звуковые ALSA в т.ч.)* и функционирует логически целостно. Однако, *Linux* так же поддерживает и динамическое подключение внешних **модулей** для поддержки оборудования, в виде которых и распространяются, например, *закрытые (проприетарные) драйверы для видеокарт NVidia и AMD/ATI*.

*Загрузкой ядра Linux* при старте *операционной системы* в большинстве случаев управляет универсальный **загрузчик Grub** (хотя встречаются и конфигурации с *загрузчиком LILO*), код которого может записываться как в загрузочный сектор основного носителя данных, так и одного из его логических разделов (для активации загрузочным флагом впоследствии). Файлы *конфигурации Grub* при этом как правило хранятся в "корневом" разделе установленной *операционной системы*, в папке */boot/grub*. Таким образом вы можете использовать несколько полностью независимых установок *Grub, Linux*, других загрузчиков и ОС, при необходимости.

Список всех загруженных *ядром модулей* устройств можно увидеть, набрав в *консоли*lsmod, а их принудительная загрузка и выгрузка производится при помощи *программы* ***modprobe***.

**Файловая система**

При запуске, *ядро Linux* создает виртуальное адресное пространство, *файловую систему* с четким разграничением прав для доступа ко всем внутренним ресурсам. Для взаимодействия с физическими ресурсами компьютера *Linux* создаёт в этой *файловой системе* специальные "файлы-порты", адреса которых начинаются обычно с **/dev** (от слова device, англ. "устройство") - по одному для каждого из устройств вычислительной машины и его подсистем, включая различные накопители данных и логические разделы, содержащиеся на них.

Для дисковых накопителей и их логических разделов сегодня чаще всего в различных *дистрибутивах GNU/Linux*по-умолчанию используются адреса устройств вида **/dev/sdX** и **/dev/sdX\***, соответственно (где "*X*" - буквенное обозначение устройства, а "*\**" - номер раздела на нём). Например: */dev/sda1* - первый (*1*) раздел на первом (*a*) диске.

Будь то жёсткий, оптический компакт-диск, USB-карточка, подключённая к системе, или сетевой ресурс данных, логические разделы какого-либо из этих накопителей присоединяются к основной файловой системе при помощи специального механизма "монтирования" по назначенным для них адресам (в так называемых "точках монтирования"). Именно так при загрузке *Linux* сначала монтирует в виртуальный адресный корень ("/") основной (*"корневой"*) раздел данных, а потом, при необходимости, и дополнительные с остальным содержимым - в отдельные папки.

В отдельном разделе в принципе может храниться и любая из *системных директорий Linux*: */boot* с настройками Grub и ядром Linux - для надежности и безопасности, */usr* - для контроля размера установленного прикладного *программного обеспечения*, или */home* - для независимости *настроек* пользователей от системы при её обновлении, например. Внешние носители данных, любые несистемные логические разделы накопителей данных обычно подключаются внутри папки */media*.

Современные *рабочие столы GNU/Linux, Gnome, KDE* и т.п., представляют пользователю подключенные логические разделы в привычном виде иконок и пунктов меню, однако в любом файловом менеджере это - всего лишь ссылки на реальные системные адреса.

При переполнении физической оперативной памяти, или использовании режима "сна" (с сохранением данных *ОЗУ* на диск), *Linux* может использовать так называемый **swap**-раздел со специальной файловой системой. Его создание на диске и использование не обязательно, но необходимо для работы режима "сна".

Список всех *логических разделов* на всех подключённых к системе носителях данных вы можете посмотреть, вызвав *в консоли команду*: fdisk -l, например (может потребовать прав администратора). *Помимо того*, логические разделы могут быть представлены в системе и использоваться в различных конфигурациях, включая */etc/fstab* и *Grub*, в виде символьных ссылок, содержащих в своих названиях уникальные номера ID каждого из томов, не изменяющиеся в зависимости от размера и расположения логического раздела на диске: смотрите dir /dev/disk/\*.

*Конфигурация* подключения виртуальных, сетевых и локальных файловых систем, включая *swap*, в *GNU/Linux* записывается в файл **/etc/fstab**, а записи о подключённых в данный момент - хранятся в динамически обновляемом файле */etc/mtab*.

**Среда GNU**

Поверх *ядра Linux* функционирует *оболочка GNU* - **многоуровневая** программная среда, достижение загрузки каждого из уровней которой характеризуется запуском соответствующего набора системных служб. Управление ими, просмотр и редактирование таблиц их запуска в разных дистрибутивах могут осуществляться по-разному, в том числе и силами специализированных графических инструментов, таких как модуль управления системными службами YaST в openSUSE.

**Первый уровень** загрузки *GNU - однопользовательский командный терминал*, состоящий из нескольких *текстовых консолей* для работы с системой, переключение между которыми может осуществляться нажатием сочетания клавиши Alt (или Ctrl+Alt+... - при работе в *X11*) с каким-либо из F\*: Alt+F1, Alt+F2 и т.д.

Первые шесть *консолей* обычно зарезервированы для прямого использования, а седьмая и следующие - для запуска *графических систем x.org-X11* локальных пользователей при достижении **пятого** загрузочного уровня. В зависимости от режима работы *операционной системы, при установке или запуске* например, в *виртуальные консоли* может производиться вывод отладочный информации о работе какой-либо из подсистем *GNU/Linux*.

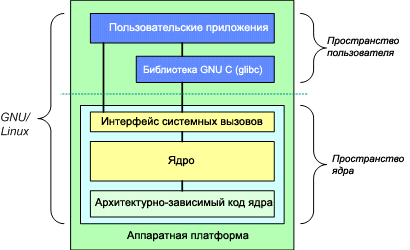
**Второй** уровень загрузки GNU/Linux - многопользовательский терминал, **третий** - многопользовательский с поддержкой сети. Четвёртый уровень - промежуточный, а **первый** и **шестой** - выключения и перезагрузки системы, соответственно.

Гибкость такой системы состоит в том, что каждый из её *программных компонентов*, используемых на разных уровнях загрузки, может быть независимо запущен, остановлен, или заменён. Кроме того, многие *программы GNU/Linux*, в отличие о тех, что были созданы с использованием закрытой модели развития, используют общие наборы открытых библиотек и даже работают в симбиозе: необходимые *программам* для работы компоненты называют "**зависимостями**".

Такая *архитектура*, помимо прочего, позволяет значительно снизить объём памяти, занимаемой *программами* на жестком диске при *установке*, и, часто, в оперативной памяти при их работе. Именно такая архитектура даёт возможность работы с разными сборками *ядер Linux* и сред *GNU*, разнообразными *рабочими столами (Gnome, KDE, XFCE и т.п.), мультимедийными системами и мультимедиа-проигрывателями, офисными и графическими приложениями* - всем разнообразием *программного обеспечения и конфигураций GNU/Linux*.

**Смена уровня** работы системы в большинстве *дистрибутивов GNU/Linux* выполняется *командой* вида init*номер* с номером уровня в качестве аргумента, выполненной от имени администратора: например, init 3 - для остановки всех графических серверов и перехода в текстовый режим.

**Модули**



*Рисунок 1 – Взаимодействие модулей ядра*

Для того, чтобы понять, как работают модули, необходимо разобрать работу ядра.

Любая программа через системные вызовы обращается к ядру ОС через интерфейс системных вызовов. Ядро через архитектурно зависимый код транслирует команды аппаратному обеспечению. Важный момент, что программы "видят" себя в едином виртуальном окружении независимо от архитектуры системы. Структура и системные вызовы стандартизированы и едины для любой аппаратной платформы. Графическая подсистема является одной из внешних программ, поэтому сменить графическую оболочку можно без перезагрузки ОС.

**Анатомия модуля ядра**

Загружаемые модули ядра имеют ряд фундаментальных отличий от элементов, интегрированных непосредственно в ядро, а также от обычных программ. Обычная программа содержит главную процедуру (main) в отличие от загружаемого модуля, содержащего функции входа и выхода. Функция входа вызывается, когда модуль загружается в ядро, а функция выхода – соответственно при выгрузке из ядра. Поскольку функции входа и выхода являются пользовательскими, для указания назначения этих функций используются макросы module\_init и module\_exit . Загружаемый модуль содержит также набор обязательных и дополнительных макросов. Они определяют тип лицензии, автора и описание модуля, а также другие параметры. Пример очень простого загружаемого модуля приведен на рисунке 2.

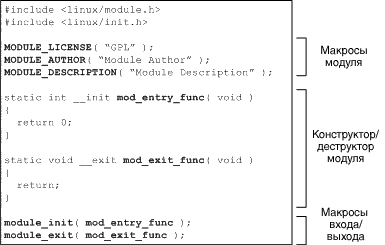


Рисунок 2. Код простого загружаемого модуля.

Версия 2.6 ядра Linux предоставляет новый, более простой метод создания загружаемых модулей. После того как модуль создан, можно использовать обычные пользовательские инструменты для управления модулями (несмотря на изменения внутреннего устройства): insmod (устанавливает модуль), rmmod (удаляет модуль), modprobe (контейнер для insmod и rmmod), depmod (для создания зависимостей между модулями) и modinfo (для поиска значений в модулях макроса).

**Анатомия объектного кода модуля ядра**

Загружаемый модуль представляет собой просто специальный объектный файл в формате ELF (ExecutableandLinkableFormat). Обычно объектные файлы обрабатываются компоновщиком, который разрешает символы и формирует исполняемый файл. Однако в связи с тем, что загружаемый модуль не может разрешить символы до загрузки в ядро, он остается ELF-объектом. Для работы с загружаемыми модулями можно использовать стандартные средства работы с объектными файлами (которые в версии 2.6 имеют суффикс *.ko,* от kernelobject). Например, если вывести информацию о модуле утилитой objdump, вы обнаружите несколько привычных разделов, в том числе *.text* (инструкции), *.data* (инициализированные данные) и *.bss* (BlockStartedSymbol или неинициализированные данные).

В модуле также обнаружатся дополнительные разделы, ответственные за поддержку его динамического поведения. Раздел .init.text содержит код module\_init, а раздел .exit.text – код module\_exitcode (рисунок 2). Раздел .modinfo содержит тексты макросов, указывающие тип лицензии, автора, описание и т. д.

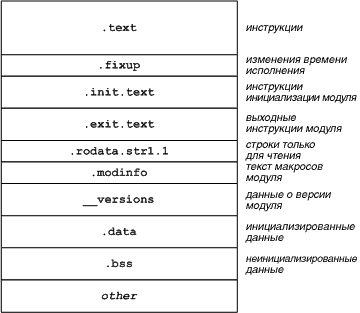


Рисунок 3. Пример загружаемого модуля с разделами ELF

**Наиболее известные программы в пакете**

* GNOME— свободная среда рабочего стола для UNIX-подобных операционных систем. GNOME является частью проекта GNU.
* MozillaFirefox— свободный браузер на движке Gecko, разработкой и распространением которого занимается MozillaCorporation.
* LibreOffice — кроссплатформенный, свободно распространяемый офисный пакет с открытым исходным кодом, созданный как ответвление OpenOffice в 2010 году. Разрабатывается сообществом из более чем 480 программистов под эгидой некоммерческого фонда TheDocumentFoundation за счёт пожертвований отдельных лиц и организаций.
* Rhythmbox — музыкальный проигрыватель для Linux и других UNIX-подобных операционных систем. Набор функций схож с таковым у AppleiTunes. Разработан специально для среды GNOME и использует GStreamer для вывода звука.
* Shotwell — компьютерная программа для организации каталога фотоизображений и видеофайлов. Shotwell является свободным программным обеспечением для Linux и других UNIX-подобных операционных систем. Работает в графическом окружении GTK+.
* И многоедругое

**Литература**

1. Разработка модулей ядра Linux - <https://www.ibm.com/developerworks/ru/library/l-linux_kernel_01/index.html>

2. Анатомия загружаемых модулей ядра Linux - <https://www.ibm.com/developerworks/ru/library/l-lkm/index.html>

3. Архитектура ОС Linux. Ядро, файловая система, компоненты. - <http://desktoplinux.ru/unix_guide/arkhitektura_operatsionnoi_sistemy_gnu_linux_yadro_failovaya_sistema_komponenty>

4. Kali Linux - hhttps://ru.wikipedia.org/wiki/Kali\_Linux

5. Архитектура ОС Linux - <http://opensourcerules.net/arhitektura-linux.html>