

Министерство науки и высшей школы Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Пермский государственный национальный исследовательский университет»
Механико-математический факультет
Кафедра информационных технологий

О т ч е т о в ы п о л н е н и и
п р а к т и ч е с к о г о з а д а н и я
п о д и с ц и п л и н е
«А д м и н и с т р и р о в а н и е
и н ф о р м а ц и о н н ы х с и с т е м»

Работу выполнили
студенты группы ФИТ-2015НБ
4 курса

Механико-математического факультета

Анпилогов А.А.,

Давыдова Е.Ю.,

Згогурин А.В.,

Калина М.И.,

Паукова М.С..

Работу принял
доцент кафедры ИТ механико-математического факультета,
к.т.н., доц., Курушин Даниил Сергеевич

Пермь, 2019

Задание:

Сеть «умного дома»

Сервер – Arch Linux

Контроллеры – ARM (Rasbian)

Сервер – контроллер - Ethernet

Управление - планшет или ПК – Wi-Fi, мобильная сеть

Задачи:

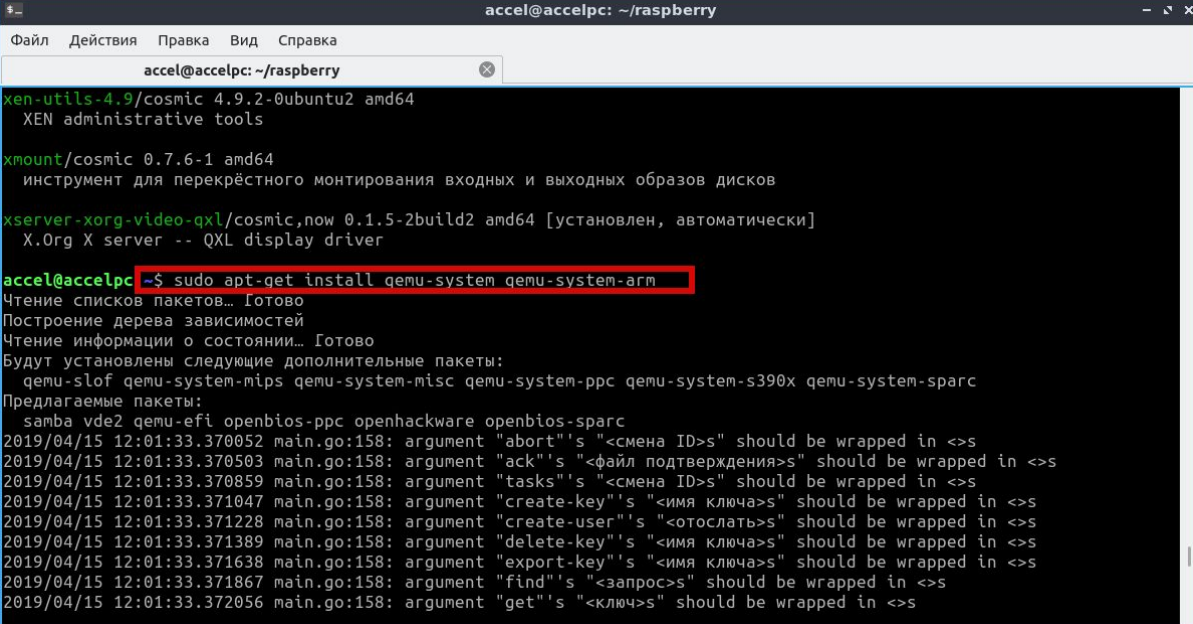
- Установить виртуальную машину с линукс ОС (например, ubuntu 14.04 и выше)
- Выбрать и установить эмулятор с открытым исходным кодом
- Установить образ файловой системы Rasbian, позволяющей контролировать и управлять элементами «умного дома»
- Проверить доступ с основной машины к образу файловой системы
- Выбрать и установить сервер Arch Linux на образе файловой системы, чтобы можно было получать и обрабатывать сигналы клиента
- Проверить доступ с основной машины к серверу
- Реализовать клиента в виде сайта
- Проверить доступы клиента к серверам (по wi-fi)
- Реализовать совместную работу клиента с сервером

Приступаем к работе

Умный дом, построенный на базе Raspberry Pi 3 — многофункциональный комплекс, позволяющий контролировать и управлять всеми элементами вашего места проживания, будь то квартира, дача или частный дом. Под его «руководством» работают многие элементы, начиная от лампочек в помещениях, заканчивая системой отопления и запуском систем, распознающих присутствие человека. Особенность системы заключается в слаженной работе всех компонентов, надежности и сравнительной

легкости настройки. Нам нужен самый обычный образ Raspberry, который используется для установки на реальное устройство.

Qemu - это популярный эмулятор с открытым исходным кодом, который поддерживает огромное количество архитектур, среди которых и ARM процессор Raspberry Pi. Мы могли бы использовать VirtualBox, но там возможна эмуляция только компьютерной архитектуры x86, а это значит, что вы сможете запустить на нем только компьютерный Raspbian и ничего не добьетесь в плане тестирования. Поэтому установим Qemu, в дистрибутивах Ubuntu выполнили команду:

A screenshot of a terminal window titled 'accel@accelpc: ~/raspberrypi'. The terminal shows the output of 'apt-get install qemu-system qemu-system-arm'. It lists several installed packages: xen-utils-4.9/cosmic, xmount/cosmic, and xserver-xorg-video-qxl/cosmic. Below this, it shows the command being executed: 'sudo apt-get install qemu-system qemu-system-arm'. The output continues with status messages like 'Чтение списков пакетов...', 'Построение дерева зависимостей...', and 'Чтение информации о состоянии... Готово'. It then lists additional packages to be installed: qemu-slof, qemu-system-mips, qemu-system-misc, qemu-system-ppc, qemu-system-s390x, and qemu-system-sparc. Finally, it lists suggested packages: samba, vde2, qemu-efi, openbios-ppc, openhackware, and openbios-sparc. The terminal output ends with a series of log messages from 'main.go:158' regarding argument wrapping for various commands like 'abort', 'ack', 'tasks', 'create-key', 'create-user', 'delete-key', 'export-key', 'find', and 'get'.

Пакет qemu устанавливает поддержку всех архитектур, а kvm добавляет поддержку модуля ядра аппаратной виртуализации, который очень сильно увеличивает производительность гостевой системы. Чтобы посмотреть версию программы выполним (1).

Нам понадобится несколько файлов для того, чтобы запустить Raspberry Pi в эмуляторе, поэтому лучше создать одну рабочую папку (2).

```
accel@accelpc: ~/raspberrry
2019/04/15 12:04:45.072055 main.go:158: argument "get"'s "<ключ>s" should be wrapped in <>s
2019/04/15 12:04:45.072335 main.go:158: argument "interface"'s "<интерфейс>s" should be wrapped in <>s
2019/04/15 12:04:45.072557 main.go:158: argument "known"'s "<тип подтверждения>s" should be wrapped in <>s
2019/04/15 12:04:45.072704 main.go:158: argument "known"'s "<фильтр заголовков>s" should be wrapped in <>s
2019/04/15 12:04:45.072905 main.go:158: argument "login"'s "<отослать>s" should be wrapped in <>s
2019/04/15 12:04:45.073154 main.go:158: argument "prepare-image"'s "<модель подтверждения>s" should be wrapped in <>s
2019/04/15 12:04:45.073313 main.go:158: argument "prepare-image"'s "<корневая директория>s" should be wrapped in <>s
2019/04/15 12:04:45.073600 main.go:158: argument "services"'s "<сервис>s" should be wrapped in <>s
2019/04/15 12:04:45.073789 main.go:158: argument "logs"'s "<сервис>s" should be wrapped in <>s
2019/04/15 12:04:45.073955 main.go:158: argument "start"'s "<сервис>s" should be wrapped in <>s
2019/04/15 12:04:45.074140 main.go:158: argument "stop"'s "<сервис>s" should be wrapped in <>s
2019/04/15 12:04:45.074303 main.go:158: argument "restart"'s "<сервис>s" should be wrapped in <>s
2019/04/15 12:04:45.074483 main.go:158: argument "set"'s "<конфигурационное значение>s" should be wrapped in <>s
2019/04/15 12:04:45.074683 main.go:158: argument "sign-build"'s "<имя файла>s" should be wrapped in <>s
2019/04/15 12:04:45.075393 main.go:158: argument "wait"'s "<ключ>s" should be wrapped in <>s
2019/04/15 12:04:45.075601 main.go:158: argument "watch"'s "<смена ID>s" should be wrapped in <>s
accel@accelpc: ~$ qemu-system-arm --version
QEMU emulator version 2.12.0 (Debian 1:2.12+dfsg-3ubuntu8.6)
Copyright (c) 2003-2017 Fabrice Bellard and the QEMU Project developers
accel@accelpc: ~$ mkdir raspberrry
accel@accelpc: ~$ ls
Desktop raspberrry Видео Документы Загрузки Изображения Музыка Общедоступные Шаблоны
accel@accelpc: ~$ cd raspberrry
bash: cd: raspberrry: Нет такого файла или каталога
accel@accelpc: ~$ cd raspberrry
accel@accelpc: ~/raspberrry$
```

Наш образ просто так не загрузится, и нам нужно его еще немного подправить. Нужно примонтировать корневую файловую систему образа, но это не так просто. Поскольку это образ диска, а не раздела и на нем есть своя файловая система то мы не можем его просто так примонтировать.

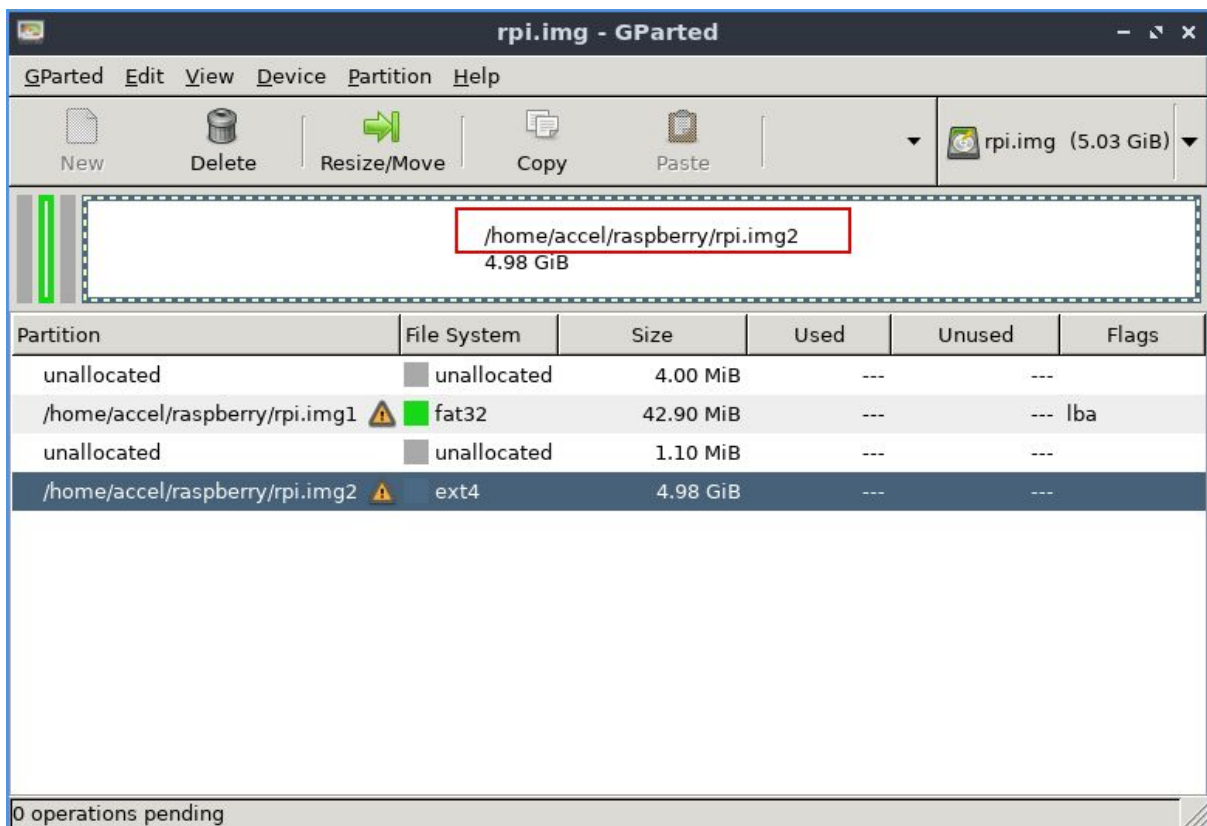
Установим gparted:

```
accel@accelpc: ~/raspberrry
accel@accelpc: ~/raspberrry$ gparted
Команда «gparted» не найдена, но может быть установлена с помощью:
sudo apt install gparted
accel@accelpc: ~/raspberrry$ sudo apt install gparted
[sudo] пароль для accel:
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Будут установлены следующие дополнительные пакеты:
  libatkmm-1.6-1v5 libcairomm-1.0-1v5 libglibmm-2.4-1v5 libgtkmm-2.4-1v5 libpangomm-1.4-1v5 libsigc++-2.0-0v5
Предлагаемые пакеты:
  reiser4progs yelp gpart udftools
2019/04/15 12:18:20.597230 main.go:158: argument "abort"'s "<смена ID>s" should be wrapped in <>s
2019/04/15 12:18:20.597388 main.go:158: argument "ack"'s "<файл подтверждения>s" should be wrapped in <>s
2019/04/15 12:18:20.597868 main.go:158: argument "tasks"'s "<смена ID>s" should be wrapped in <>s
2019/04/15 12:18:20.597943 main.go:158: argument "create-key"'s "<имя ключа>s" should be wrapped in <>s
2019/04/15 12:18:20.598004 main.go:158: argument "create-user"'s "<отослать>s" should be wrapped in <>s
2019/04/15 12:18:20.598054 main.go:158: argument "delete-key"'s "<имя ключа>s" should be wrapped in <>s
2019/04/15 12:18:20.598168 main.go:158: argument "export-key"'s "<имя ключа>s" should be wrapped in <>s
2019/04/15 12:18:20.598430 main.go:158: argument "find"'s "<запрос>s" should be wrapped in <>s
2019/04/15 12:18:20.598500 main.go:158: argument "get"'s "<ключ>s" should be wrapped in <>s
2019/04/15 12:18:20.598736 main.go:158: argument "interface"'s "<интерфейс>s" should be wrapped in <>s
2019/04/15 12:18:20.598824 main.go:158: argument "known"'s "<тип подтверждения>s" should be wrapped in <>s
```

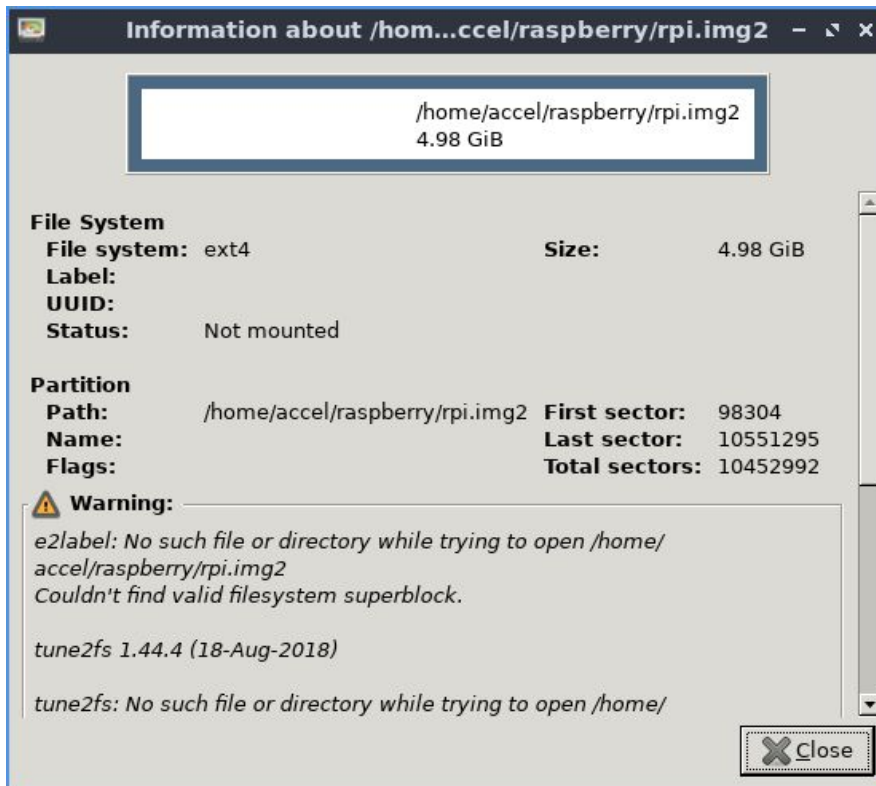
Открыли файл в gparted:

```
accel@accelpc: ~/raspberr
Файл Действия Правка Вид Справка
accel@accelpc: ~/raspberr
2019/04/15 12:19:09.320964 main.go:158: argument "interface"'s "<интерфейс>s" should be v
2019/04/15 12:19:09.321064 main.go:158: argument "known"'s "<тип подтверждения>s" should
2019/04/15 12:19:09.321109 main.go:158: argument "known"'s "<фильтр заголовков>s" should l
2019/04/15 12:19:09.321211 main.go:158: argument "login"'s "<отослать>s" should be wrapp
2019/04/15 12:19:09.321339 main.go:158: argument "prepare-image"'s "<модель подтверждения
2019/04/15 12:19:09.321387 main.go:158: argument "prepare-image"'s "<корневая директория:
2019/04/15 12:19:09.321523 main.go:158: argument "services"'s "<сервис>s" should be wrapp
2019/04/15 12:19:09.321600 main.go:158: argument "logs"'s "<сервис>s" should be wrapped i
2019/04/15 12:19:09.321689 main.go:158: argument "start"'s "<сервис>s" should be wrapped
2019/04/15 12:19:09.321765 main.go:158: argument "stop"'s "<сервис>s" should be wrapped i
2019/04/15 12:19:09.321839 main.go:158: argument "restart"'s "<сервис>s" should be wrapp
2019/04/15 12:19:09.322371 main.go:158: argument "set"'s "<конфигурационное значение>s" :
2019/04/15 12:19:09.322513 main.go:158: argument "sign-build"'s "<имя файла>s" should be
2019/04/15 12:19:09.323118 main.go:158: argument "wait"'s "<ключ>s" should be wrapped in
2019/04/15 12:19:09.323196 main.go:158: argument "watch"'s "<смена ID>s" should be wrapp
accel@accelpc: ~/raspberr$ gparted
Unit tmp.mount does not exist, proceeding anyway.
=====
libparted : 3.2
=====
accel@accelpc: ~/raspberr$ sudo gparted rpi.img
Unit tmp.mount does not exist, proceeding anyway.
=====
libparted : 3.2
=====
```

Нашли корневой раздел, размером четыре гигабайта с файловой системой Ext4 и нажали "Свойства":



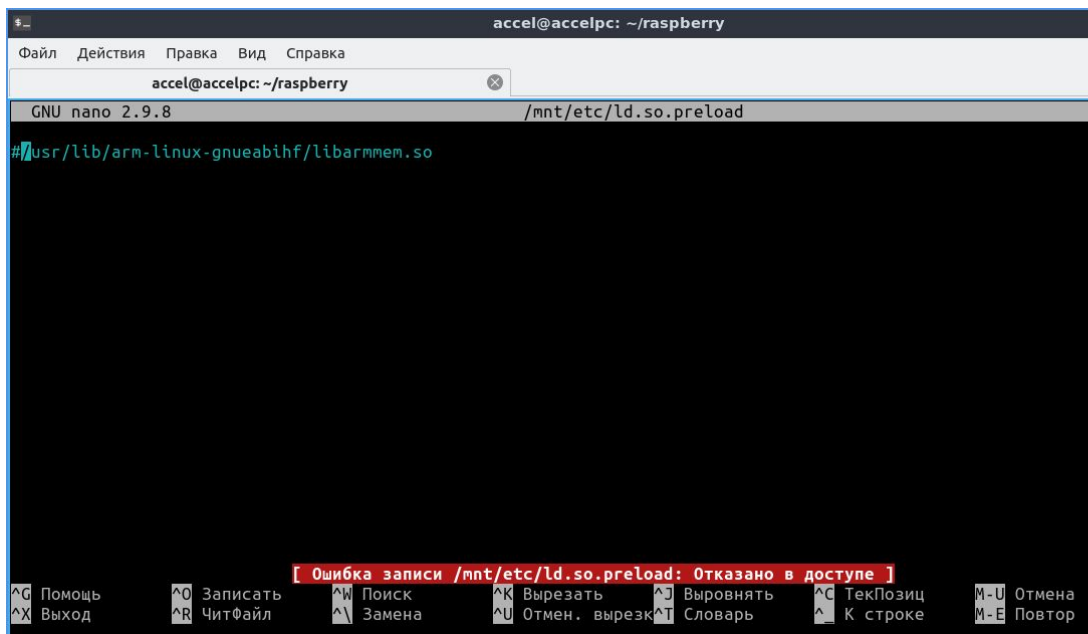
Как описать картинку ниже????



В "Первом секторе", полученное число нужно умножить на 512 чтобы узнать по какому адресу монтировать файловую систему, только после этого монтируем:

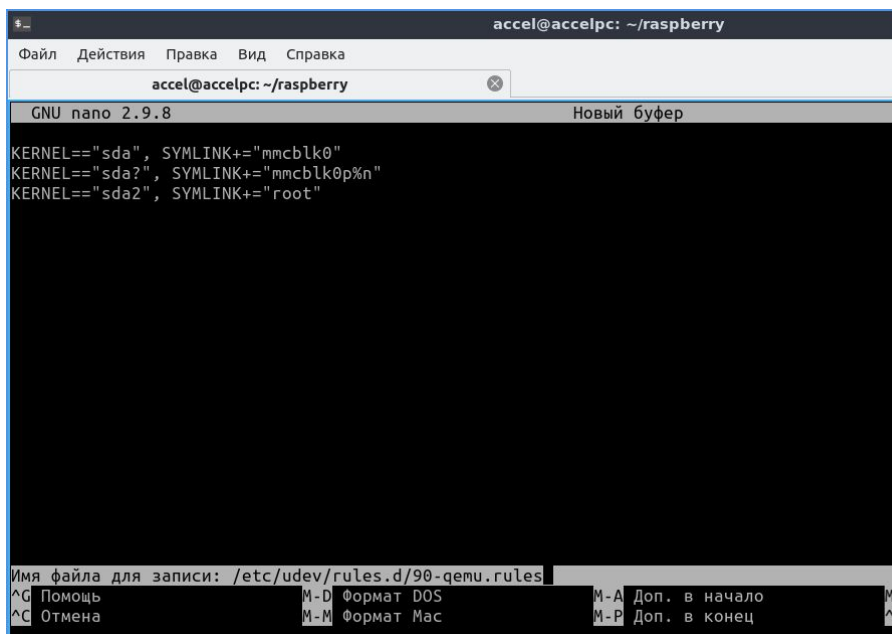
```
accel@accelpc: ~/raspberr
Файл Действия Правка Вид Справка
accel@accelpc: ~/raspberr
2019/04/15 12:19:09.321064 main.go:158: argument "known"'s "<тип подтверждения>s" should be wrapped in <>s
2019/04/15 12:19:09.321109 main.go:158: argument "known"'s "<фильт заголовков>s" should be wrapped in <>s
2019/04/15 12:19:09.321211 main.go:158: argument "login"'s "<отослать>s" should be wrapped in <>s
2019/04/15 12:19:09.321339 main.go:158: argument "prepare-image"'s "<модель подтверждения>s" should be wrapped in <>s
2019/04/15 12:19:09.321387 main.go:158: argument "prepare-image"'s "<корневая директория>s" should be wrapped in <>s
2019/04/15 12:19:09.321523 main.go:158: argument "services"'s "<сервис>s" should be wrapped in <>s
2019/04/15 12:19:09.321600 main.go:158: argument "logs"'s "<сервис>s" should be wrapped in <>s
2019/04/15 12:19:09.321689 main.go:158: argument "start"'s "<сервис>s" should be wrapped in <>s
2019/04/15 12:19:09.321765 main.go:158: argument "stop"'s "<сервис>s" should be wrapped in <>s
2019/04/15 12:19:09.321839 main.go:158: argument "restart"'s "<сервис>s" should be wrapped in <>s
2019/04/15 12:19:09.322371 main.go:158: argument "set"'s "<конфигурационное значение>s" should be wrapped in <>s
2019/04/15 12:19:09.322513 main.go:158: argument "sign-build"'s "<имя файла>s" should be wrapped in <>s
2019/04/15 12:19:09.323118 main.go:158: argument "wait"'s "<ключ>s" should be wrapped in <>s
2019/04/15 12:19:09.323196 main.go:158: argument "watch"'s "<смена ID>s" should be wrapped in <>s
accel@accelpc: ~/raspberr$ gparted
Unit tmp.mount does not exist, proceeding anyway.
=====
libparted : 3.2
=====
accel@accelpc: ~/raspberr$ sudo gparted rpi.img
Unit tmp.mount does not exist, proceeding anyway.
=====
libparted : 3.2
=====
accel@accelpc: ~/raspberr$ sudo mount -o loop,offset=50331648 rpi.img /mnt
accel@accelpc: ~/raspberr$
```

Затем открыли файл /mnt/etc/ld.preload.so и закоментировали там одну строчку:



The screenshot shows a terminal window with the nano text editor open. The title bar indicates the user is 'accel' at 'accelpc' in the directory '~/raspberry'. The editor is editing the file '/mnt/etc/ld.so.preload'. The current line of code is '#/usr/lib/arm-linux-gnueabi/libarmmmem.so'. A red error message is displayed at the bottom: '[Ошибка записи /mnt/etc/ld.so.preload: Отказано в доступе]'. The bottom status bar shows various keyboard shortcuts for nano, such as '^G' for help, '^O' for write, and '^X' for exit.

Сохранили изменения и закрыли файл. Нужно еще добавить несколько правил udev, чтобы Raspbian правильно определял нашу SD карту. По большому счету программе неважно что это будет жесткий диск, файл или реальная SD карта, вот только она ожидает что это будет устройство /dev/mmcblk. Поэтому создадим файл 90-qemu.rules и добавим туда такие строки:



The screenshot shows the nano text editor creating a new buffer. The title bar is the same as the previous image. The editor is in 'Новый буфер' (New buffer) mode. The content of the buffer is three lines of udev rules: 'KERNEL=="sda", SYMLINK+="mmcblk0"', 'KERNEL=="sda?", SYMLINK+="mmcblk0p%n"', and 'KERNEL=="sda2", SYMLINK+="root"'. The bottom status bar shows the filename 'Имя файла для записи: /etc/udev/rules.d/90-qemu.rules' and various keyboard shortcuts.

Сохранили изменения и отмонтировали раздел (3-4):

```
accel@accelpc: ~/raspberrry
Файл Действия Правка Вид Справка
accel@accelpc: ~/raspberrry
accel@accelpc:~/raspberrry$ gparted
Unit tmp.mount does not exist, proceeding anyway.
=====
libparted : 3.2
=====
accel@accelpc:~/raspberrry$ sudo gparted rpi.img
Unit tmp.mount does not exist, proceeding anyway.
=====
libparted : 3.2
=====
accel@accelpc:~/raspberrry$ sudo mount -o loop,offset=50331648 rpi.img /mnt
accel@accelpc:~/raspberrry$ nano /mnt/etc/ld.so.preload
accel@accelpc:~/raspberrry$ nano /mnt/etc/ld.so.preload
accel@accelpc:~/raspberrry$ nano /mnt/etc/ld.so.preload
accel@accelpc:~/raspberrry$ sudo nano /mnt/etc/ld.so.preload
accel@accelpc:~/raspberrry$ nano /mnt/etc/ld.so.preload
accel@accelpc:~/raspberrry$ nano /mnt/etc/ld.so.preload
accel@accelpc:~/raspberrry$ sudo nano /mnt/etc/ld.so.preload
accel@accelpc:~/raspberrry$ sudo nano
accel@accelpc:~/raspberrry$
accel@accelpc:~/raspberrry$ sync
accel@accelpc:~/raspberrry$
accel@accelpc:~/raspberrry$ sudo umount /mnt
accel@accelpc:~/raspberrry$
```

Загрузили репозиторий:

```
accel@accelpc: ~/raspberrry
Файл Действия Правка Вид Справка
accel@accelpc: ~/raspberrry
checkout Переключение веток или восстановление файлов в рабочем каталоге
commit Запись изменений в репозиторий
diff Вывод разницы между коммитами, коммитом и рабочим каталогом и т.д.
merge Объединение одной или нескольких историй разработки вместе
rebase Повторно применить коммиты над верхушкой другой ветки
tag Создание метки, вывод списка, удаление или проверка метки, подписанной с по
совместная работа (смотрите также: git help workflows)
fetch Загрузка объектов и ссылок из другого репозитория
pull Извлечение изменений и объединение с другим репозиторием или локальной ветк
push Обновление внешних ссылок и связанных объектов
«git help -a» и «git help -g» выводит список доступных подкоманд и
некоторые руководства по темам. Запустите «git help <команда>» или
«git help <термин>», чтобы прочесть о конкретной подкоманде или теме.
accel@accelpc:~/raspberrry$
accel@accelpc:~/raspberrry$
accel@accelpc:~/raspberrry$ git clone https://github.com/dhruvvyas90/qemu-rpi-kernel.git
Клонирование в «qemu-rpi-kernel»...
remote: Enumerating objects: 63, done.
remote: Counting objects: 100% (63/63), done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 300 (delta 36), reused 63 (delta 36), pack-reused 237
Получение объектов: 100% (300/300), 39.49 MiB | 106.00 KiB/s, готово.
Определение изменений: 100% (155/155), готово.
accel@accelpc:~/raspberrry$
```

Теперь все готово и мы можем запустить эмулятор Raspberry Pi 3. Для запуска эмулятора мы будем использовать команду:


```
accel@accelpc: ~/raspberrypi
Файл Действия Правка Вид Справка
accel@accelpc: ~/raspberrypi
    check for possible regressions in migration code
    by comparing two such vmstate dumps.

Generic object creation:
-object TYPENAME[,PROP1=VALUE1,...]
    create a new object of type TYPENAME setting properties
    in the order they are specified. Note that the 'id'
    property must be set. These objects are placed in the
    '/objects' path.

During emulation, the following keys are useful:
ctrl-alt-f      toggle full screen
ctrl-alt-n      switch to virtual console 'n'
ctrl-alt        toggle mouse and keyboard grab

When using -nographic, press 'ctrl-a h' to get some help.

See <https://qemu.org/contribute/report-a-bug> for how to report bugs.
More information on the QEMU project at <https://qemu.org>.
accel@accelpc:~/raspberrypi$ qemu-system-arm -kernel ./qemu-rpi-kernel/kernel-qemu-4.4.26-jessie -cpu arm1176 -m 1024 -M versatilepb -no-reboot -serial stdio -append "root=/dev/sda2 panic=1 rootfstype=ext4 rw init=/bin/bash" -hda rpi.img
WARNING: Image format was not specified for 'rpi.img' and probing guessed raw.
        Automatically detecting the format is dangerous for raw images, write operations on block 0 will be restricted.
        Specify the 'raw' format explicitly to remove the restrictions.
qemu-system-arm: versatilepb: memory size must not exceed 256MB
accel@accelpc:~/raspberrypi$
```

Итоги

Таким образом