Pimp my Raspberry Pi oder Pi 2 (beide Modell B)

Die Heimat der kleinen Himbeere findet sich hier: http://www.raspberrypi.org/
Und so sieht sie aus:

Raspberry Pi B, Rev. 1 ohne Gehäuse

Raspberry Pi B, Rev. 1 ohne Gehäuse

Raspberry Pi 2 B ohne Gehäuse

Bildquellen:

- http://commons.wikimedia.org/wiki/Category:Raspberry_Pi_board_photographs?uselang=de
- http://upload.wikimedia.org/wikipedia/commons/0/0c/Raspberry_Pi_2_Model_B_v1.1_front_angle_new.jpg

Debian Wheezy (7.x), Jessie (8.x) oder Stretch (9.x) installieren

Ich habe für meine ersten Versuche folgende Version verwendet:
http://downloads.raspberrypi.org/images/raspbian/2013-02-09-wheezy-raspbian/2013-02-09-wheezy-raspbian.zip
Die aktuelle bzw. letzte Version von Raspbian kann hier heruntergeladen werden.
Wer lieber aufeine ältere Version zurückgreifen möchtet findet sie hier:
http://downloads.raspberrypi.org/raspbian/images/ oder hier
https://www.raspbian.org/RaspbianMirrors.

```
# Download per wget
wget http://downloads.raspberrypi.org/raspbian_latest -O raspbian_latest.zip

# entpacken
unzip raspbian_latest.zip

# entpacktes Image auf eine SD-Karte schreiben.
# Aber Vorsicht mit dd - in eine falsche Ausgabedatei (of = output file) zerstört die dortigen Daten unwiederbringlich!
dd bs=4M if=2015-01-31-raspbian.img of=/dev/sdg
# oder für Ubuntu 16.04 LTS
dd bs=4M if=2016-05-27-raspbian-jessie-lite.img of=/dev/mmcblk0
#
```

```
sudo dd if=2016-11-25-raspbian-jessie.img of=/dev/sdd bs=512; sync
# leere Datei ''ssh'' im der ''boot''-Partition anlegen um den SSH Server
vom ersten Start an zu aktivieren
touch /media/$(whoami)/boot/ssh
#oder
touch /run/media/$(whoami)/boot/ssh
```

Der erste Start und Login

SD-Karte in den Raspberry Pi einlegen und booten (Spannungsversorgung anschließen).
Für ein erstes SSH-Login: user: **pi**; password: **raspberry**

```
ssh -l pi <IP-ADRESSE-DER-HIMBEERE>
```

Wer es gerne ohne Passworteingabe mag, der kann auch eine Schlüsseldatei auf den Pi kopieren. Wie man eine Schlüsseldatei erzeugt steht hier

```
# generierte Schlüseldtei auf den Raspberry Pi kopieren:
ssh-copy-id -i ~/.ssh/id_rsa.pub pi@[hostname|IP]
```

Raspberry Konfigurieren

```
ssh -l pi <IP-ADRESSE-DER-HIMBEERE>
pi@raspberrypi ~ $ sudo su
root@raspberrypi:/home/pi# raspi-config
  * SD-Karten Speicherplatz ausdehnen um den kompletten Speicherplatz nutzen
zu können
  * Update raspi-config
  * REBOOT

Diverse Systemeinstellungen tätigen...
  * locale: de_DE.UTF-8
  * Timezone: Europe -> Berlin
  * Tastaturlayot anpassen
  * hostname ändern
  * -> Finish
root@raspberrypi:/home/pi# passwd  # root Passwort vergeben für
anschließendes Login per SSH
root@raspberrypi:/home/pi# reboot && exit
```

Dateimanager nachrüsten

`Midnight Commander` installieren:

```
ssh -l root <IP-ADRESSE-DER-HIMBEERE>
root@raspberrypi:/# apt-get install mc
```

Statische IP-Adresse konfigurieren

Quelle(n):

- http://www.netzmafia.de/skripten/hardware/RasPi/RasPi_Network.html

`nano /etc/dhcpcd.conf`

```
...
interface eth0
static ip_address=192.168.100.200/24
static routers=192.168.100.1
static domain_name_servers=192.168.100.1
```

Schlankheitskur

Quellen:

- http://www.cnx-software.com/2012/07/31/84-mb-minimal-raspbian-armhf-image-for-raspberry-pi/
- https://www.saufler.de/article/lampp-debian-wheezy/
- http://www.gtkdb.de/index_36_2489.html

Als erstes ist eine Schlankheitskur zu empfehlen denn für ein `apt-get update && apt-get upgrade` reicht der verbleibende Speicherplatz einer 4GB SD-Karte nicht aus. Entwickler- (DEV), Python- und andere Pakete finden und entfernen ...

```
root@raspberrypi:/# cd /
root@raspberrypi:/# rm -rf /home/pi/python_games/
root@raspberrypi:/# apt-get remove x11-common midori lxde python3 python3-
minimal
root@raspberrypi:/# du -sh /  # mal schauen wieviel Platz belegt ist
root@raspberrypi:/# df -h /   # vieviel ist noch frei?
root@raspberrypi:/# apt-get purge --auto-remove alsa-{base,utils} console-
{setup,setup-linux} desktop-{base,file-utils}
libx{11,aw,cb,composite,cursor,damage,ext,fixes,ft,i,inerama,kbcommon,kbfile
,klavier,mu,pm,randr,render,res,ss,t,v}* lightdm lxde lxde-
{common,core,icon-theme} lx{appearance,input,menu-
data,panel,polkit,randr,session,session-edit,shortcut,task,terminal} x11-
{common,utils,xkb-utils} xarchiver xauth xinit xserver-{common,xorg,xorg-
core,xorg-input-all,xorg-input-evdev,xorg-input-synaptics,xorg-video-fbdev}
omxplayer penguinspuzzle gnome-accessibility-themes libraspberrypi-dev
libraspberrypi-doc libx11-xcb1:armhf debian-reference-en
root@raspberrypi:/# apt-get purge --auto-remove $(sudo dpkg --get-selections
| grep "\-dev" | grep "install" | sed 's/install//')
root@raspberrypi:/# apt-get --purge autoremove # nicht benötigte Pakete
entfernen
root@raspberrypi:/# apt-get clean
root@raspberrypi:/# apt-get install aptitude
```

Jetzt stehen ca. 1,59 GB mehr Speicherplatz auf der SD-Karte zur Verfügung.

oder alternativ:

```
dpkg-query -Wf '${Installed-Size}\t${Package}\n' | sort -n
aptitude remove wolfram-engine oracle-java8-jdk sonic-pi
```

Upgrade auf Debian 8 "Jessie"

Paketlisten anpassen: das Release „wheezy“ durch „jessie“ ersetzten.

```
sudo nano /etc/apt/sources.list

# danach update und upgrade durchführen:
sudo apt-get update && sudo apt-get dist-upgrade

# nicht mehr benötigte Pakete entfernen:
sudo apt-get autoremove
sudo apt-get autoclean
```

Das System auf den neuesten Stand bringen

Für zwischendurch - damit alles frisch bleibt 

```
root@raspberrypi:/# apt-get update  # Paketliste aktualisieren
root@raspberrypi:/# apt-get upgrade # abgespecktes System auf den neuesten
Stand bringen
```

RAID mit mdadm

Quelle: http://www.gtkdb.de/index_36_2187.html

```
sudo su
# Alle Partitionen löschen:
parted /dev/sda "rm 1"
parted /dev/sdb "rm 1"

# Neue DOS Partitionstabelle anlegen
parted /dev/sda "mklabel msdos"
--> yes
parted /dev/sdb "mklabel msdos"
--> yes

# Neue Partitionen anlegen
parted /dev/sda "mkpart primary ext4 1M -1"
parted /dev/sdb "mkpart primary ext4 1M -1"
```

```
# Partitionstyp auf "Raid autodetect" setzen
parted /dev/sda "set 1 raid on"
parted /dev/sdb "set 1 raid on"

# korrekte Partitionierung prüfen:
parted -s /dev/sda print
parted -s /dev/sdb print

# Mdadm installieren
apt-get install mdadm
--> y

# RAID 1 erstellen
mdadm --create /dev/md0 --level=1 --raid-devices=2 /dev/sda1 /dev/sdb1

# RAID 1 Erstellung überprüfen
lsblk

# RAID 1 formatieren
mkfs -t ext4 /dev/md0

# RAID 1 Status abrufen
[watch -n 1] cat /proc/mdstat
# oder
[watch -n 1] mdadm --detail /dev/md0

# RAID 1 einhängen
mkdir -p /mnt/raid1
mount /dev/md0 /mnt/raid1

# Eintrag in fstab
echo "/dev/md0        /nmt/raid1           ext4    defaults          0
0" >> /etc/fstab
```

System updaten

Zuerst mal alle Paketequellen aktualisieren und installierte Pakete auf den neuesten Stand bringen:

```
apt-get update
apt-get upgrade
```

Raspberry als FTP-Server einrichten

Quelle: http://layer8problem.wordpress.com/2012/07/24/pureftpd-installation-and-setup/

Pure-FTPd installieren

```
apt-get install pure-ftpd-common pure-ftpd
```

Pure-FTPd konfigurieren

... TBC - das ist etwas komplizierter...

Raspberry als NAS einrichten

Quellen:

- http://www.welzels.de/blog/projekte/raspberry-pi/low-budget-nas-mit-einem-raspberry-pi/
- https://wiki.archlinux.org/index.php/fstab

externen Speicher vorbereiten

Verzeichnis zum Einhängen des externen Massenspeichers anlegen:

```
mkdir -p /mnt/nas
```

Externene Massenspeicher anschließen und die UUID des externen Massenspeichers herausfinden:

```
blkid
/dev/mmcblk0p1: SEC_TYPE="msdos" UUID="C522-EA52" TYPE="vfat"
/dev/mmcblk0p2: UUID="62ba9ec9-47d9-4421-aaee-71dd6c0f3707" TYPE="ext4"
/dev/sda1: LABEL="MyBook_01_500GB"
UUID="83723c98-4978-4ae6-94ef-90c3aa2ce5b4" TYPE="ext3"
```

UUID des externen Speichers (hier eine externe USB Festplatte vom Typ „Western Digital MyBook“) in die `fstab` Datei eintrgen damit sie während des Systemstarts automatisch in das Dateisystem eingehängt wird.

```
vi /etc/fstab
...
proc            /proc           proc    defaults          0       0
/dev/mmcblk0p1  /boot           vfat    defaults          0       2
/dev/mmcblk0p2  /               ext4    defaults,noatime  0       1
# a swapfile is not a swap partition, so no using swapon|off from here on,
use  dphys-swapfile swap[on|off]  for that
UUID=83723c98-4978-4ae6-94ef-90c3aa2ce5b4 /mnt/nas ext3 defaults 0 0
```

Sollte das externe Speichermedium beim Systemstart nicht angeschlossen oder (noch) nicht betriebsbereit sein (z.B. Netzteil nicht in der Steckdose) so bootet der Raspberry nicht vollständig. Während des Bootvorganges wird natürlich versucht, den externen Speicher in das Dateisystem einzubinden (mount) - das schlägt dann fehl und es kommt zu einem inkonsistenten System mit

gesperrtem `root` Account.
Dem lässt sich durch die Erweiterung mit … `defaults,nofail,x-systemd.device-timeout=1`… in der entsprechenden Zeile der `fstab` Datei vorbeugen:

```
UUID=910c6de1-2cfe-4170-bc0f-d2c3fe45f564 /mnt/usb-hdd ext4
defaults,nofail,x-systemd.device-timeout=1 0 2
```

Dateiserver ''Samba'' installieren und einrichten

```
apt-get install samba samba-common-bin
```

Verzeichnispfad für ''HOME'' anpassen

```
vi /etc/adduser.conf

# The DHOME variable specifies the directory containing users' home
# directories.
DHOME=/mnt/nas
...
...
# If DIR_MODE is set, directories will be created with the specified
# mode. Otherwise the default mode 0755 will be used.
DIR_MODE=0750
...
...
# Set this if you want the --add_extra_groups option to adduser to add
# new users to other groups.
# This is the list of groups that new non-system users will be added to
# Default:
EXTRA_GROUPS="dialout cdrom floppy audio video plugdev users"

# If ADD_EXTRA_GROUPS is set to something non-zero, the EXTRA_GROUPS
# option above will be default behavior for adding new, non-system users
ADD_EXTRA_GROUPS=1
```

Linux Benutzer anlegen

```
adduser [USERNAME]
```

Samba Benutzer anlegen

```
smbpasswd -a [USERNAME]
```

Samba Konfiguration

```
vi /etc/samba/smb.conf

...
# ======================= Share Definitions =======================

[homes]
   comment = Home Directories
   browseable = yes
   writeable = yes
   read only = no
   create mask = 0700
   directory mask = 0700
   valid users = %S

[NAS]
   comment = Network Attached Storage
   browseable = yes
   path = /mnt/nas
#   writeable = yes
#   printable = yes
   guest ok = no
   read only = yes
   create mask = 0664
   directory mask = 0775
   force group = users
   write list = pi

# ======================= Drucker deaktivieren =======================

     load printers = no
     printing = bsd
     printcap name = /dev/null
     disable spoolss = yes
```

Samba neu starten:

```
./etc/init.d/samba restart
```

Zeitgesteuerte Aktionen ausführen

Links dazu:
http://www.thegeekstuff.com/2010/06/at-atq-atrm-batch-command-examples/
http://stegro.github.com/2012/12/28/ripradio.html|

```
root@raspberrypi:/# apt-get install cron anacron at
```

```
echo "streamripper
http://ndrstream.ic.llnwd.net/stream/ndrstream_ndrinfo_hi_mp3 -a
FILENAME.mp3 -l 3600" > myjobs.txt\\
at -f myjobs.txt now + 1 min\\
```

oder per ssh von einem anderen Rechner aus:

```
ssh pi@raspberrypi "at -f myjobs.txt now + 2 min"\\
```

```
ssh pi@raspberrypi "at -f at_job.sh 20:51 2013-04-11"\\
```

```
at 21:21 2013-04-11
at> at_job.sh http://ndrstream.ic.llnwd.net/stream/ndrstream_ndrinfo_hi_mp3
Test 10
STRG + D
```

```
echo "/home/pi/at_job.sh
http://ndrstream.ic.llnwd.net/stream/ndrstream_ndrinfo_hi_mp3 Test 10" | at
22:09 2013-04-11
```

Kleines Skript zum teitgesteuerten Mitschneiden eines Radiostreams:

```
#!/bin/bash
#
#

#RECDIR="/home/christoph/Downloads/tmp/NDR"
RECURL="$1"
RECFILE="$2"
RECLENGTH="$3"
RECTIME="$4"
RECDATE="$5"
TIMESTAMP=""

function GET_TIMESTAMP ()
{
TIMESTAMP=`date "+%d.%m.%Y %H:%M"`
return
}

#cd $RECDIR
#GET_TIMESTAMP
#echo "$TIMESTAMP: "$RECFILE" von "$RECURL", Länge "$RECLENGTH" Sekunden" >>
$RECDIR/Aufnahmen.txt
#streamripper $RECURL -a $RECFILE.mp3 -l $RECLENGTH > /dev/null 2>&1
#ssh pi@raspberrypi "echo \"/home/pi/at_job.sh
http://ndrstream.ic.llnwd.net/stream/ndrstream_ndrinfo_hi_mp3 Test_01 10\" |
at 21:00 2013-04-12"
ssh pi@raspberrypi "echo \"/home/pi/at_job.sh $RECURL $RECFILE $RECLENGTH \"
| at $RECTIME $RECDATE"
```

```
rec_webradio.sh
http://ndrstream.ic.llnwd.net/stream/ndrstream_ndrinfo_hi_mp3
Bei_Erinnerung_Mord 3360 21:05 2013-04-13
```

Internet Radio Stream mitschneiden

```
root@raspberrypi:/# aptitude install streamripper
```

```
pi@raspberrypi:/# streamripper
http://ndrstream.ic.llnwd.net/stream/ndrstream_ndrinfo_hi_mp3 -a
FILENAME.mp3 -l 3600
```

Liste der Radiosender ist von hier: http://wiki.kairaven.de/open/os/linux/streamrip

```
BR2=http://streams.br-online.de/bayern2_2.m3u
BR3=http://streams.br-online.de/bayern3_2.m3u
DRadio Kultur=http://www.dradio.de/streaming/dkultur.m3u
DLF=http://www.dradio.de/streaming/dlf.m3u
DRadio Wissen=http://www.dradio.de/streaming/dradiowissen.m3u
HR2  Kultur=http://metafiles.gl-systemhaus.de/hr/hr2_2.m3u
HR   Info=http://metafiles.gl-systemhaus.de/hr/hrinfo_2.m3u
MDR  Info=http://avw.mdr.de/livestreams/mdr_info_live_128.m3u
NDR  Info=http://ndrstream.ic.llnwd.net/stream/ndrstream_ndrinfo_hi_mp3
NDR  Kultur=http://ndrstream.ic.llnwd.net/stream/ndrstream_ndrkultur_hi_mp3
RBB  Info=http://www.inforadio.de/live.m3u
RBB  Kultur=http://www.kulturradio.de/live.m3u
RBB  radioeins=http://www.radioeins.de/live.m3u
SR2  Kultur=http://streaming01.sr-online.de/sr2_2.m3u
SWR  Info=http://mp3-live.swr.de/contra_m.m3u
SWR2 Kultur=http://mp3-live.swr.de/swr2_m.m3u
WDR  1Live=http://www.wdr.de/wdrlive/media/einslive.m3u
WDR3=http://www.wdr.de/wdrlive/media/wdr3.m3u
WDR5=http://www.wdr.de/wdrlive/media/wdr5.m3u
```

RPi als Chromecast Alternative

Quellen:

- https://play.google.com/store/apps/details?id=at.huber.raspicast&hl=en

```
sudo su
apt-get install omxplayer
```

Nah- und Fernkontrolle

Shutdown Taster

Quelle(n):

- https://www.heise.de/ct/hotline/Ein-Ausschalter-fuer-Raspberry-Pi-und-Raspi-Zero-3892620.html
- https://gilyes.com/pi-shutdown-button/
- https://github.com/gilyes/pi-shutdown/blob/master/pishutdown.py

`touch /bin/pishutdown.py` `nano /bin/pishutdown.py`

```
#!/usr/bin/python
# shutdown/reboot(/power on) Raspberry Pi with pushbutton

import RPi.GPIO as GPIO
from subprocess import call
from datetime import datetime
import time

# pushbutton connected to this GPIO pin, using pin 5 also has the benefit of
# waking / powering up Raspberry Pi when button is pressed
shutdownPin = 5

# if button pressed for at least this long then shut down. if less then
reboot.
shutdownMinSeconds = 3

# button debounce time in seconds
debounceSeconds = 0.01

GPIO.setmode(GPIO.BOARD)
GPIO.setup(shutdownPin, GPIO.IN, pull_up_down=GPIO.PUD_UP)

buttonPressedTime = None

def buttonStateChanged(pin):
    global buttonPressedTime

    if not (GPIO.input(pin)):
        # button is down
        if buttonPressedTime is None:
            buttonPressedTime = datetime.now()
    else:
        # button is up
        if buttonPressedTime is not None:
            elapsed = (datetime.now() - buttonPressedTime).total_seconds()
            buttonPressedTime = None
            if elapsed >= shutdownMinSeconds:
                # button pressed for more than specified time, shutdown
                call(['shutdown', '-h', 'now'], shell=False)
            elif elapsed >= debounceSeconds:
```

```
                # button pressed for a shorter time, reboot
                call(['shutdown', '-r', 'now'], shell=False)

# subscribe to button presses
GPIO.add_event_detect(shutdownPin, GPIO.BOTH, callback=buttonStateChanged)

while True:
    # sleep to reduce unnecessary CPU usage
    time.sleep(5)
```

touch /etc/systemd/system/pishutdown.service nano /etc/systemd/system/pishutdown.service

```
[Service]
ExecStart=/usr/bin/python /bin/pishutdown.py
WorkingDirectory=/bin/
Restart=always
StandardOutput=syslog
StandardError=syslog
SyslogIdentifier=pishutdown
User=root
Group=root

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

Dienst aktivieren und starten:

```
sudo
systemctl enable pishutdown.service
systemctl start pishutdown.service
```

PI-Control

Quelle: https://pi-control.de/ Steuerung des Raspberry Pi über die Android App `Pi Control`.
Installation:

```
sudo su
curl https://pi-control.de/pic_installer | sudo bash
```

Konfigutationsdatei: `mcedit etc/nginx/sites-available/picontrol`
Wichtig ist hier, dass vor der Zeile **`fastcgi_pass unix:/run/php/php7.0-fpm.sock;`** kein Kommentarzeichen steht. Anderenfalls liefert der Webserver (hier `nginx`) immer die Fehlermeldung „Error 502 - Bad Gateway“.

```
server {
        listen 80 default_server;
        listen [::]:80 default_server;
```

```
        root /var/www/html;

        index index.php index.html;
        server_name _;

        location / {
                try_files $uri $uri/ =404;
        }

        location ~ \.php$ {
                include snippets/fastcgi-php.conf;
#               fastcgi_pass 127.0.0.1:9000;
                fastcgi_pass unix:/run/php/php7.0-fpm.sock;
        }

        location ~ /\.ht {
                deny all;
        }
}
```

DNS Filter

Ad-Blocker

Quelle: https://pi-hole.net/ Der Raspberry Pi als Werbeblocker für das gesamte Heimnetz.

```
curl -sSL https://install.pi-hole.net | bash
```

Webserver und Joomla installieren

Quellen: (Reihenfolge beachten!)

- http://www.kriwanek.de/raspberry-pi/raspberry-webserver/287-raspberry-pi-als-webserver-mit-php-und-mysql.html - Top, bis Einrichtung FTP-Server!
- http://canox.net/2014/08/joomla-auf-dem-raspberry-pi/
- https://www.saufler.de/article/lampp-debian-wheezy/
- #http://elinux.org/Raspberry_Joomla! - Jooml Installation etc.
- #http://www.deviousweb.com/it-blogs/raspberry-pi-blog/17-raspberry-pi-installing-joomla - WICHTIG!

```
cd /
apt-get install apache2
# check: http://raspberrypi2.wg/
apt-get install php5 php5-curl
cd /var/www/html/
touch phpinfo.php
nano phpinfo.php
```

```
  <?php
  phpinfo();
  ?>
# check: http://raspberrypi2.wg/phpinfo.php
apt-get install mysql-server mysql-client
shutdown -r now && exit
apt-get install phpmyadmin
nano /etc/php5/apache2/php.ini
  extension=mysql.so
# check: http://raspberrypi2.wg/phpmyadmin/
# FTP-Server
apt-get install proftpd
 # Startmodus --> Servermodus
nano /etc/proftpd/proftpd.conf
  DefaultRoot ~
  AuthOrder   mod_auth_file.c  mod_auth_unix.c
  AuthUserFile  /etc/proftpd/ftpd.passwd
  AuthPAM  off
  RequireValidShell  off
cd /etc/proftpd/
id www-data
sudo ftpasswd --passwd --name ftpuser --uid 33 --gid 33 --home /var/www --
shell /bin/false
chmod -R g+s /var/www
chmod -R 775 /var/www
chown -R www-data:www-data /var/www
cd /var/www/html/
#
# Joomla Installation
################################################################################
############################
#
mkdir joomla
cd joomla
wget http://joomlacode.org/gf/download/frsrelease/17968/78430/Joomla_2.5.9-
Stable-Full_Package.zip
unzip Joomla_2.5.9-Stable-Full_Package.zip
touch configuration.php
cp /etc/php5/apache2/php.ini  /etc/php5/apache2/php.ini.orig
nano /etc/php5/apache2/php.ini
  output buffering
  Default Vaule: off
  Development Value: 0
  Production Value: 0

service apache2 restart

nano /etc/php5/php.ini
  change  "upload_max_filesize" to "4M"
```

http://raspbmc-lan/

http://raspbmc-lan/phpinfo.php
http://raspbmc-lan/phpmyadmin/
http://raspbmc-lan/administrator

OwnCloud

Zur Verwaltung der eigenen Kontakte und Termine außerhalb von Google, der NSA/BND/GCHQ/MOSSAD und Co. bietet sich die Verwendung von OwnCloud an. Im Folgenden ist die Installation, Konfiguration und der Betrieb auf dem Raspberry Pi 2 Modell B beschrieben.

Quellen

Folgende Quellen waren sehr hilfreich beim Aufbau eines eigenen OwnCloud Servers auf Basis des Raspberry Pi 2 (Modell B):

- http://raspberry.tips/raspberrypi-tutorials/owncloud-8-auf-dem-raspberry-pi-2-1/
- http://www.heise.de/ct/ausgabe/2014-21-Owncloud-auf-Raspberry-Pi-und-Cubietruck-2393107.html
- https://meinnoteblog.wordpress.com/2013/11/26/owncloud-auf-raspberry-pi-mit-nginx-und-mysql-installieren/
- http://httpd.apache.org/docs/2.2/vhosts/examples.html
- https://thomas-leister.de/allgemein/apache-webserver-ssl-verschlusselung-einrichten/

Hilfreich auch dieser Eintrag im OwnCloud Forum:

- https://forum.owncloud.org/viewtopic.php?f=21&t=21870

Vorbereitungen

Mehr SWAP Space - SWAP-File erzeugen

```
echo "CONF_SWAPSIZE=1024" > /etc/dphys-swapfile
dphys-swapfile setup
dphys-swapfile swapon
```

OwnCloud Installation

```
wget
http://download.opensuse.org/repositories/isv:ownCloud:community/Debian_8.0/
Release.key | apt-key add - < Release.key
echo 'deb
http://download.opensuse.org/repositories/isv:/ownCloud:/community/Debian_8.
0/ /' >> /etc/apt/sources.list.d/owncloud.list
apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 977C43A8BA684223
apt-get update
```

```
apt-get upgrade
apt-get install owncloud
chmod -R g+s /var/www/owncloud
chmod -R 775 /var/www/owncloud
chown -R www-data:www-data /var/www/owncloud
```

```
vi etc/apache2/sites-enabled/000-default
  #DocumentRoot /var/www/html
   DocumentRoot /var/www/owncloud
```

```
service apache2 reload
mysql -u root -p
# Password eingeben!
> CREATE DATABASE owncloud;
> CREATE USER 'owncloud'@'localhost' IDENTIFIED BY 'password';
> GRANT ALL PRIVILEGES ON owncloud.* TO 'owncloud'@'localhost';
> FLUSH PRIVILEGES;
< exit;
```

Cron Job anlegen

```
crontab -u www-data -e
*/15 * * * * php -f /var/www/owncloud/cron.php
#
service cron restart
```

512MByte Limit erhöhen

... entweder in der `.htaccess` ...

```
vi /var/www/owncloud/.htaccess
php_value upload_max_filesize = 2G
php_value post_max_size = 2G
```

... oder in der `.user.ini`. Diese Werte verhindern einen Abbruch beim Hochladen von Dateien auf eine bei 1&1 gehostete OwnCloud Instanz.

```
upload_max_filesize=32M
post_max_size=32M
memory_limit=120M
max_execution_time=10000
mbstring.func_overload=0
always_populate_raw_post_data=-1
default_charset='UTF-8'
output_buffering=0
```

Wichtiger Hinweis: Wenn durch manuelle Veränderung (wie in den beiden o.a. Beispielen) die standard PHP-Parameter des Webservers überschrieben werden ist unbedingt darauf zu Achten, dass

sie nicht **überschritten** werden. In diesem Fall werden die eigenen Werte vom Server ignoriert und auf die Standardwerte zurückgesetzt. So z.B. bei `post_max_size=4G`: wenn der Server als Standard `40M` gesetzt hat ist ein Upload von Dateien mit mehr als 40M nicht möglich.

SSL Zertifikat erzeugen

```
cd /root
mkdir -p owncloud
cd owncloud
openssl genrsa -out server.key 4096
openssl req -new -key server.key -out server.csr

Country Name (2 letter code) [AU]:DE
State or Province Name (full name) [Some-State]:NDS
Locality Name (eg, city) []:Hildesheim
Organization Name (eg, company) [Internet Widgits Pty Ltd]:privat
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:raspberrypi2
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:

openssl x509 -req -days 365 -in server.csr -signkey server.key -out
server.crt
chmod 400 server.key
```

Apache2 Konfiguration für SSL anpassen

```
nano /etc/apache2/sites-enabled/000-default.conf
<VirtualHost *:80>
   ServerName raspberrypi2.wg
   Redirect permanent / https://raspberrypi2.wg/
</VirtualHost>
<VirtualHost *:443>
DocumentRoot /var/www/owncloud
ServerName raspberrypi2
#Header always add Strict-Transport-Security "max-age=15768000"
SSLEngine on
SSLCertificateFile /root/owncloud/server.crt
SSLCertificateKeyFile /root/owncloud/server.key
</VirtualHost>
#
nano /etc/apache2/sites-available/default-ssl.conf
<IfModule mod_ssl.c>
<VirtualHost *:443>
```

```
ServerAdmin webmaster@localhost
DocumentRoot /var/www/owncloud
<IfModule mod_ssl.c>
SSLEngine on
SSLCertificateKeyFile /root/owncloud/server.key
SSLCertificateFile /root/owncloud/server.crt
SetEnvIf User-Agent ".*MSIE.*" \
nokeepalive ssl-unclean-shutdown \
downgrade-1.0 force-response-1.0
</IfModule>
#
sudo a2enmod ssl
sudo apache2ctl configtest
Syntax OK
sudo apache2ctl restart
```

Datengrab verlagern ;-)

Neues Verzeichnis für einen Massenspeicher anlegen:

```
mkdir -p /media/owncloud
# USB-Stick einstecken!
blkid /dev/sda1
/dev/sda1: UUID="250347e2-09a1-447b-b0db-0a9b2b081fec" TYPE="ext4"
PARTUUID="00000e9b-01"
echo "UUID=250347e2-09a1-447b-b0db-0a9b2b081fec /media/owncloud/ ext4
defaults,auto" >> /etc/fstab
mount /media/owncloud
mkdir -p /media/owncloud/data
nano /var/www/owncloud/config/config.php
  'datadirectory' => '/media/owncloud/data',

chown www-data:www-data /media/owncloud/data
mv /var/www/owncloud/data /media/owncloud/data

chown -R www-data:www-data /media/owncloud/data
find /media/owncloud/data -type d -exec chmod 750 {} \;
find /media/owncloud/data -type f -exec chmod 640 {} \;
chown root:www-data /media/owncloud/data/.htaccess
chmod 0644 /media/owncloud/data/.htaccess
apache2ctl restart
```

Joomla etc.

Quellen:

- http://mentalgrain.com/rpi/installing-nginx-php5-and-sqlite3-on-a-raspberry-pi/
- http://fastjoomlahost.com/joomla-nginx-set-configuration

```
apt-get install nginx
```

```
cat <<EOT > boot.cmd
setenv bootargs console=ttyS0,115200 root=/dev/mmcblk0p2 rootwait panic=10
fatload mmc 0 0x43000000 script.bin
fatload mmc 0 0x48000000 uImage
bootm 0x48000000
EOT
```

Redmine

Quellen:

- http://www.tylerforsythe.com/2015/04/install-redmine-onto-raspberry-pi-2-this-is-the-tutorial-you-want/

```
sudo su
apt-get update
apt-get upgrade
apt-get install apache2 mysql-server
apt-get install redmine redmine-mysql
apt-get install libapache2-mod-passenger
ln -s /usr/share/redmine/public /var/www/redmine
chown -R www-data:www-data /var/www/redmine
#
 cat /etc/apache2/sites-available/redmine.conf
DocumentRoot /var/www/
PassengerDefaultUser www-data

<Location /redmine>
RailsEnv production
RackBaseURI /redmine
Options -MultiViews
</Location>
#
a2ensite redmine
a2dissite 000-default
#
systemctl reload apache2.service
systemctl restart apache2.service
```

Nextcloud <=13.x

Nach einigem Experimentieren hat sich herausgestellt, dass die Rechenleistung und auch die Größe des Arbeitsspeichers (512 MB beim Pi1 B) nicht ausreichen um Nextcloud sinnvoll zu betreiben. Daher rate ich von der Verwendung eines Raspberry

Pi B als Nextcloud-Server ab.

Quellen:

- https://www.howtoforge.com/tutorial/installing-lighttpd-with-php-fpm-and-mysql-or-mariadb-on-ubuntu/
- https://pimylifeup.com/raspberry-pi-nextcloud-server/
- https://stackoverflow.com/questions/33470753/create-mysql-database-and-user-in-bash-script
- https://canox.net/2016/06/die-eigene-cloud-mit-dem-raspberry-pi-und-nextcloud/

Grundsystem aktualisieren

```
apt-get update
apt-get upgrade
apt-get install aptitude mc
```

MySQL Datenbank

Installation von MySQL und Einrichtung einer Datenbank `nextcloud`:

```
##### aptitude install mysql-server mysql-client
aptitude install mariadb-server mariadb-client
# MariaDB erfragt kein Administrator Passwort daher kann in den folgenden
vier mysql-Aufrufen der Parameer -p entfallen:
# MySQL Datenbank, User und Passwort anlegen
mysql -uroot -p${mysqlrootpasswd} -e "CREATE DATABASE nextcloud;"
mysql -uroot -p${mysqlrootpasswd} -e "CREATE USER nextcloud@localhost
IDENTIFIED BY '${PASSWDDB}';"
mysql -uroot -p${mysqlrootpasswd} -e "GRANT ALL PRIVILEGES ON nextcloud.* TO
'nextcloud'@'localhost';"
mysql -uroot -p${mysqlrootpasswd} -e "FLUSH PRIVILEGES;"
```

Lighttpd Webserver und PHP

Installation des HTTP-Servers `lighttpd` inkl. PHP:

```
aptitude install lighttpd php7.0-fpm php7.0 php7.0-mysqlnd php7.0-curl
php7.0-mcrypt php7.0-imap php7.0-cgi php7.0-imagick php7.0-pspell php7.0-ps
```

```
mcedit /etc/php/7.0/fpm/php.ini
#  uncomment the line cgi.fix_pathinfo=1
cd /etc/lighttpd/conf-available/
cp 15-fastcgi-php.conf 15-fastcgi-php-spawnfcgi.conf
mcedit 15-fastcgi-php.conf
-----
## Start an FastCGI server for php (needs the php5-cgi package)
```

```
fastcgi.server += ( ".php" =>
        ((
                "socket" => "/var/run/php/php7.0-fpm.sock",
                "broken-scriptfilename" => "enable"
        ))
)
-----
lighttpd-enable-mod fastcgi
lighttpd-enable-mod fastcgi-php

service php7.0-fpm reload
aptitude install phpmyadmin

#Web server to reconfigure automatically: <-- lighttpd
#Configure database for phpmyadmin with dbconfig-common? <-- yes
#
#Password of the database's administrative user: <-- Enter the MySQL/MariaDB
root password
#MySQL application password for phpmyadmin: <-- Press ENTER

#Afterwards, you can access phpMyAdmin under
http://192.168.100.3/phpmyadmin/
#
```

Nextcloud 13

- aktuelle Version von Nextcloud von www.nextcloud.com herunterladen

```
 wget
https://download.nextcloud.com/server/releases/nextcloud-13.0.0.tar.bz2
```

- entpacken

```
 tar xfj nextcloud-13.0.0.tar.bz2
```

- entpackten Ordner per FTP in das Web-Root Verzeichnis hochladen
- Datenverzeichnis parallel zum Nextcloud Verzeichnis anlegen um Benutzerdaten und Nextcloud Installation voneinander zu trennen
- `/tmp` Verzeichnis innerhalb des Nextcloud-Verzeichnisses anlegen
- URL des Web-Root Verzeichnis in Browser aufrufen und die neue Nextcloud Instanz mit den richtigen Parametern für das Datenverzeichnis, den Benutzernamen und das Passwort des Administrators sowie den Datenbankparametern konfigurieren
- In der Datei `/nextcloud/config/config.php` den Pfad zum Datenverzeichnis anpassen und den Pfad zum `/tmp` Verzeichnis ergänzen

```
 'tempdirectory' => '/.../.../.../htdocs/.../nextcloud/tmp',
```

- `php.ini` unterhalb von /nextcloud mit folgendem Inhalt anlegen:

```
mkdir -p /var/www/html/nextcloud/data
chown www-data:www-data /var/www/html/nextcloud/data
```

```
chmod 750 /var/www/html/nextcloud/data
cd /var/www/html/nextcloud
chown www-data:www-data config apps
```

php.ini

```
opcache.enable=1
opcache.enable_cli=1
opcache.interned_strings_buffer=8
opcache.max_accelerated_files=10000
opcache.memory_consumption=128
opcache.save_comments=1
opcache.revalidate_freq=1
```

Nextcloud 20.x

- https://pimylifeup.com/raspberry-pi-nextcloud-server/

xx

```
xx
```

altes Zeug/old stuff - NICHT VERWENDEN!/DO NOT USE THIS!!!

Webserver, Datenbank und PHP

Zum Betrieb der eigenen Wolke sind ein paar Grundlegende Dinge wie z.B. ein Webserver und eine Datenbank notwendig.

```
root@raspberrypi2:/# groupadd www-data
root@raspberrypi2:/# usermod -a -G www-data www-data
root@raspberrypi2:/# apt-get install php-apc php5-curl mysql-server php5-
mysql
root@raspberrypi2:/# apt-get install openssl
openssl genrsa -out server.key 2048
openssl req -new -key server.key -out server.csr
```