Raspberry Pi as Server

The Raspberry Pi is the great device to set up a small server at home to be used for almost any purpose. It’s cheap, small, has a low power consumption footprint and there is a great community for it to get help from. There are tons of things to do with such a mini-server. However, the very first step always will be to get the system up and running.

The best Linux to install on the Raspberry Pi is probably Raspbian which is a Debian flavor (effectively, just Debian adopted for the Raspberry Pi). Raspbian is the most stable and best supported operating system for the Raspberry with lots of helpful support. The Raspbian Linux comes in two flavors – a full system with almost everything included and a reduced, ‚lite‘ version without the graphical desktop which is the perfect choice to run the Raspberry as a server. Below is a description how to bring up the Raspberry with Raspbian-Lite and do some fine-tuning so it is a solid basis to use it as a server.

Install Linux (Raspbian-Lite)

Download the *Raspbian Lite* zip package to your computer from www.raspberrypi.org/downloads/raspbian/. You should ensure the integrity of the download by checking the SHA-256 checksum provided on the download page. For Linux you can use the sha256sum shell command (typcially already installed and available). A useful open source tool for Windows can be found here: github.com/gurnec/HashCheck.

Best solution is to use a SSD-drive (connected to the Raspberry via USB) but alternatively a SD-card can be used. If you use a SD-card, you probably first must format the card: it must be FAT, not exFAT. Next, flash the SSD-drive or SD-card with the downloaded linux zip-package using *etcher* which is available from etcher.io (for Linux and Windows). There is no need to unzip the Rasbian file to burn the image (i.e. linux). After Etcher is finished and no errors are reported don’t remove the crive of card yet.

Copy an empty file simply called „*ssh*“ (no file extension like .txt!) into the boot partition of the drive or card. Note: there should be two partitions, the other one is called „*rootfs*„, the boot partition is the one where the file „*kernel.img*“ is. This will enable remote ssh access to the Raspberry right on the first boot.

If you want the Raspberry to connect to an existing WLAN right on first boot, create a file called „*wpa_supplicant.conf*„ in the boot partition. On first boot Raspbian will use this file to configure and connect to the WLAN. The file itself should contain at least the following information on the WLAN (and change the country-code to your needs):

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=us

network={
    scan_ssid=1
    ssid="testing"
    psk="testingPassword"
}
```

Once done, connect the drive (or insert the SD-card) to the Raspberry, connect it to the router (your internet gateway) with a LAN cable and power on the Raspberry.

Now you need to find out the IP address of the Raspberry (this very much depends on the internet router used; it might also help to assign a static IP to the Raspberry). Then connect to the Raspberry via ssh (e.g. using *putty* on Windows or ssh on Linux) and confirm the new certificate that you're asked for. The username is „*pi*" (so use „*ssh pi@raspberrypi*„)and the initial password is „*raspberry*„. If the password doesn't work, try „*raspberrz*" (due to a wrong keyboard layout) and ensure you're on the same LAN as the Raspberry.

Once logged in on the terminal (command line) the first thing is to update the whole system – for this just type in:

```
sudo apt update
sudo apt upgrade -y
```

A reboot might be required. The next step is to change the basic configuration of the server via the special config-tool for the Raspberry. To start it, simply enter and confirm with the password:

```
sudo raspi-config
```

Change the following things within this configuration application:

- Change password for the default user (option 1)
- Change hostname (option 2, N1)
- Enable old, standard interface names (option 2, N3 – select „No“!)
- Change localisation (option 4) – Locale: space and tab to select (UTF-8 preferred), select your timezone, and your wifi-country.
- Expand filesystem (option 7, A1)
- Change GPU memory to the minimum 16 (option 7, A3)

Upon „Finish“ you're asked to reboot; do so and then log into the Raspberry once again (as user „pi“ with the new password just set).

Now you have a running system you can start using already. All the following steps are only optional and might or might not be helpful depending on what you want to achieve.

Some Improvements and Hardening

Change user to ‚admin‘

Change pi user to admin (not required, but much nicer that way). First create a new user (probably use your nickname and replace <username> with it):

```
sudo adduser <username>
sudo usermod --append --groups sudo <username>
```

You need to provide the password for that new user; all the rest being asked for can be left blank (just press Enter). Logout and login as the new user just created, then:

```
sudo usermod --login admin --home /home/admin --move-home pi
sudo groupmod --new-name admin pi
```

Logout and login as admin.

New ssh keys

Remove the standard ssh keys on the new Raspberry Pi server and generate new ones:

```
sudo rm /etc/ssh/ssh_host*
sudo dpkg-reconfigure openssh-server
```

Logout and login (confirming the new certificate once).

SSH login without password

You can use keys instead of entering a password each time you log into the new server. If you're coming from a Linux machine, then simply enter the following command on that other Linux device (if this is the first time you do this, then you might first need to generate the keys by entering ssh-keygen and simply press enter on the questions):

```
ssh-copy-id admin@my-new-server
```

Confirm once with the password – afterwards you will log into the new server without being asked for a password anymore (you identify yourself with the key on the machine your coming from).

You probably should repeat the above for other users on the same maching (e.g. the one created earlier on).

You can even improve the ssh login process even more by creating a configuration file for ssh on the server that you're coming from:

```
nano ~/.ssh/config
```

and enter something like the following (adjust to your needs):

```
Host my-server1 my-server1.mydomain
    Hostname my-server1.mydomain
    User admin

Host raspberry raspberry.lan
    Hostname raspberry.lan
    User alice
```

Then you can simply type ssh raspberry and you're logged into raspberry.lan as user alice right away. This alias configuration even works with scp as well.

Automatic security updates

One should also ensure that important security updates are installed automatically, i.e. unattended – this can be achieved with the following tool:

```
sudo apt install -y unattended-upgrades
```

and configure and enable it with (just confirm what is suggested):

```
sudo dpkg-reconfigure unattended-upgrades
```

Any updates are logged in the file /var/log/unattended-upgrades/unattended-upgrades.log.

Disable WLAN and/or Bluetooth

Most often the Bluetooth interface of the Raspberry Pi will not be used at all. Therefore, it’s a good security measure to simply disable it completely (just remember this):

```
sudo rfkill block bluetooth
```

and check it was sucessfull with:

```
sudo rfkill list
```

The same can be done with WLAN – if you're setup and use case doesn't require the WLAN at all just use the following:

```
sudo rfkill block wlan
```

and check the status like for Bluetooth above.

Swap Space (SD-Card only)

The so-called swap-space is not the best thing to be used on SD-cards – it is quite slow and the SD-Card only allows for a limited number of write operations before it will die. So swapping with lots of writing to the SD-card will break it rather sooner than later – therefore one might want to lower the use of the swap on a Raspberry Pi with e.g.:

```
sysctl vm.swappiness
sudo sysctl --write vm.swappiness=10
```

A lower swappiness number means less swapping to the SD-card.

One can easily check the swap status with:

```
free -h
```

Welcome Text (motd)

If you don’t like the welcome text shown each time you log in you can easily change it by editing the motd file:

```
sudo nano /etc/motd
```

Shell behaviour

Something useful and a nice-to-have are a few shortcuts for shell commands often used. This can be easily achieved by

```
nano .bash_aliases
```

and then define aliases as wanted along these examples

```
alias l='ls -CF'
alias la='ls -A'
alias ll='ls -lF'
alias lla='ls -AlF'

alias check='curl https://check.torproject.org/api/ip ; echo'
```

To enable them right away just run (alternatively, one could just log out and back in):

```
source .bash_aliases
```

Again, you might probably want to repeat the above for any other user on that machine.

Last but not least it proved quite useful to change the prompt so it looks different for each server (this helps mixing up different machines). You can change the prompt in the file .bashrc; look for the variable PS1 and change it to your needs (there’s enough detailed information on the internet about the details).

Configure the nano editor

The way the nano editior is working can be changed (and improved) in its config file:

```
nano ~/.nanorc
```

with e.g. the follwing configs:

```
# Use smooth scrolling by default.
set smooth

# Use the blank line below the titlebar as extra editing space.
set morespace

# Use a tab size of number columns. The value of number must be greater than 0. The default value is 8.
```

```
set tabsize 4

# Specify the color combination to use for the titlebar. Valid color names for are:
# white, black, red, blue, green, yellow, magenta, and cyan. And either "fgcolor" or ",bgcolor" may be left out.
#set titlecolor fgcolor,bgcolor

# Enable the use of ~/.nano/search_history for saving and reading search/replace strings.
unset historylog

# Save a file by default in Unix format. This overrides nano's default behavior of saving a file in the format
# that it had.  (This option has no effect when you also use set noconvert.)
set unix

# Enable mouse support, if available for your system. When enabled, mouse clicks can be used to place the cursor,
# set the mark with a double click and execute shortcuts. The mouse will work in the X Window System, and on the
# console when gpm is running. Text can still be selected through dragging by holding down the Shift key.
set mouse
```

As with some other configurations you might want to repeat this for any other users on that device.

Stop the avahi service

The avahi service is only needed in case you want to connect to e.g. a printer or scanner etc. (a device relying on zeroconfig). On a headless server this is usually not needed and just causes unnecessary overhead on the server and the network. To disable avahi just apply:

```
sudo systemctl stop avahi-daemon.service
sudo systemctl stop avahi-daemon.socket
sudo systemctl disable avahi-daemon.service
sudo systemctl disable avahi-daemon.socket
```

Tuning systemd

The following is experimental right now and needs further investigation – but it might resolve issues that apache fails to start since it can bind to the wlan0 address on boot:

```
sudo systemctl edit apache2.service
```

and paste in:

```
[Unit]
Wants=network-online.target
After=network-online.target

[Service]
Restart=on-failure
RestartSec=10
```

then reload the config, check that it’s there then restart and check apache (all the checking below is not required, but good to know how one can check things out; also sudo is often not really mandatory):

```
sudo systemctl daemon-reload
sudo systemctl cat apache2.service
sudo systemctl show apache2.service
sudo systemctl restart apache2.service
sudo systemctl status apache2.service
sudo systemctl list-dependencies apache2.service
```

What to do with the Raspberry Server

There are lot’s of things that could be done with the Raspberry Pi server just enabled. Just A few ideas are:

- Tor WLAN Gateway
- VPN (Calling Home when you’re travelling)
- Nextcloud server (your own cloud at home)
- Host a Tor Relay (help the Tor network by sharing some bandwidth)
- OpenMediaVault (NAS)

or simply search the internet for tons of other ideas. Have fun, and happy ’serving‘!