Set up a Raspberry Pi Local Server

Introduction

This guide will walk through setting up a Raspberry Pi for the first time, as a very low-cost server computer. These are not high performance servers, but they can serve a basic function, and consume very little power. For this guide, we will be setting up and configuring everything on the command line. This is more complicated, but is a handy skill to learn for working with servers in the future.

The steps are detailed below, and involve downloading the operating system for the computer, setting it up, and installing software. From there, the guide will go on to configuring a basic web server, and install a blog platform and collaborative editor.

Equipment and supplies needed:

- Raspberry Pi computer - model 3B recommended.
- Power supply for Raspberry Pi
- 8GB MicroSD card - 32GB or larger card recommended
- A computer with an SD card slot, or a MicroSD card reader/writer
- Ethernet cable
- Access to a router or switch with a connection to the Internet

This is not an exhaustive guide, and just follows the basic steps. If you run into trouble, you will need to consult the documentation on the Raspberry Pi website (https://www.raspberrypi.org/help/), or other how-to guides on the Internet.

Loading an Operating System

If you have a new Pi, you will need to install an operating system on the computer. To do this, you can download and write an image to the MicroSD card. You will then insert the SD card into the Pi, and connect to it with the Ethernet cable.

We will be using Raspbian - a version of Linux - for this guide. First, download it here:

https://www.raspberrypi.org/downloads/raspbian/

We will be using the “Lite” version without a desktop for this guide. Once it is downloaded, unzip if it is compressed in a “.zip” file format. Next, you will need a program to write the file to your card. Find the program for your platform on this page:

https://www.raspberrypi.org/documentation/installation/installing-images/README.md

Download the program for your platform and install it. Follow the instructions to write the image. Be very careful - this will overwrite anything on the drive! Double-check that you have the right drive before clicking “Write”.

Important: After the file is written to the microSD card, you need to create a file named “ssh” on the root or main directory of the card to enable remote access. Without this file, you will not be able to get into the Raspberry Pi!

On Windows, start Notepad (In Windows Accessories). Leave the file empty, but save with filename: ssh (no .txt or any extension after - just “ssh”). Move this file into the microSD card drive (it will have a drive letter such as D:, E:, or F: on Windows). Finally, eject and remove it from your computer or card reader.

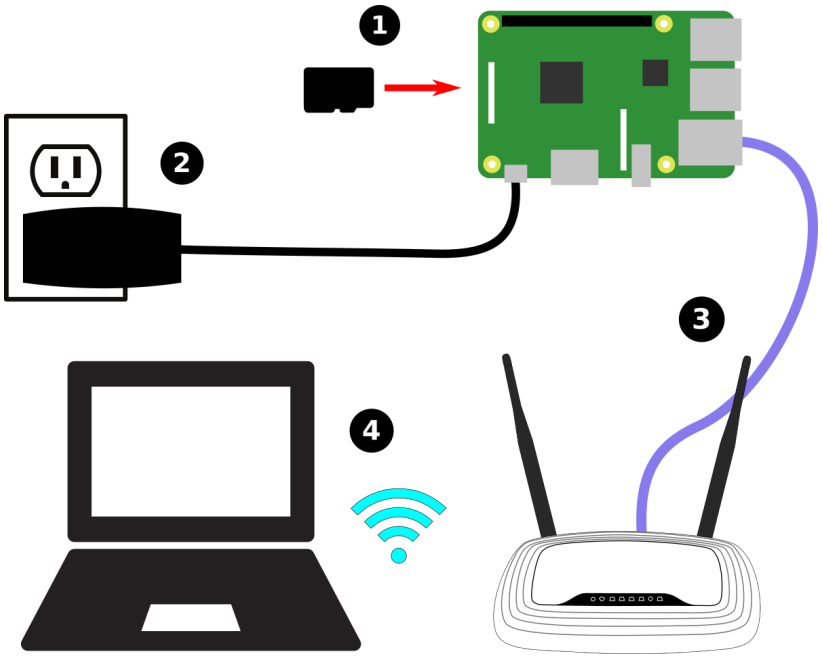
If you have trouble making a file with no extension just drag any file from the root directory of the sdcard to your desktop. Rename it: ssh Now drag the fill back into the root directory of the MicroSD card.

Finally, eject the card and remove it from your computer or card reader.

Connect to your Raspberry Pi

Now, we will plug everything in and connect to our new Pi computer. Follow these steps:

1. Insert the MicroSD card into the slot at the end of the Pi.
2. Plug the power adapter into a wall outlet or power strip, then plug it into the MicroUSB connector on the Pi.
3. Plug an Ethernet cable into your Internet router.
4. Connect your computer or laptop to the router, either with Wi-Fi or Ethernet.



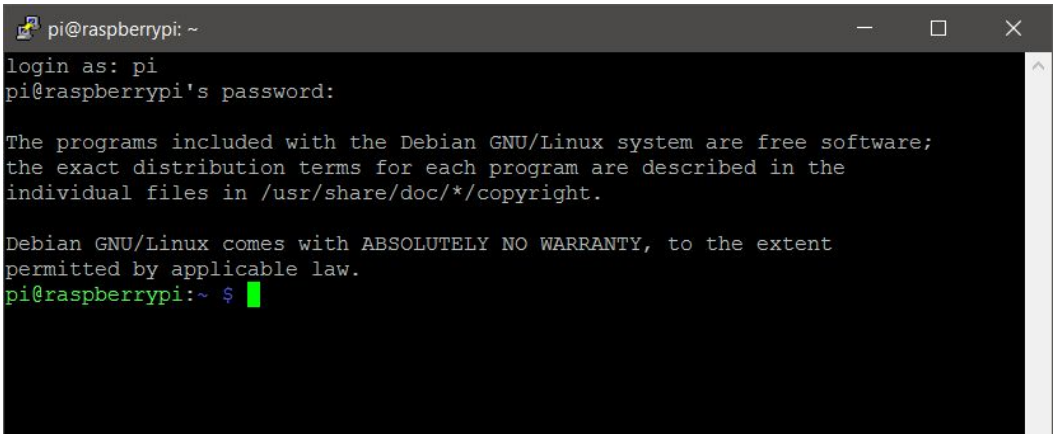
Now, wait a minute or two for the Raspberry Pi to boot, then browse to the administrator panel on your Internet router. Find the list of connected devices, and look for the IP address that was assigned to the Pi. It should have the hostname “raspberrypi”.

If you cannot find the IP address of the Pi, on some networks the hostname may be available. Open a console or terminal, and try and ping the hostname. If you see a positive response, you can just use the hostname on this network.

You will need to SSH into the command line of the Pi. To do this on Windows, you will need to download an SSH client program, such as PuTTY (http://www.putty.org/). On Mac OSX and Linux, you should have a built in SSH client.

Example for Linux: ssh pi@192.168.1.101
Open a terminal or your SSH client and connect to the IP address or hostname of the Pi. You will get a prompt asking you to log in:

You should login as “pi” and use the password “raspberry”. You should then get the following screen:



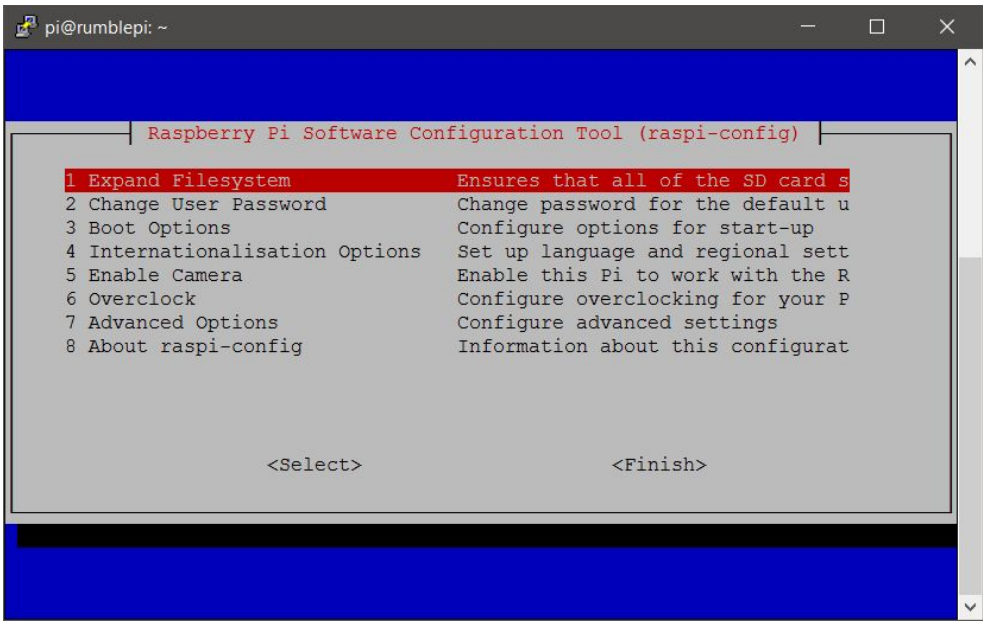
You are now connected to your Raspberry Pi!

Finish your Setup

The first command to run is raspi-config. At the prompt, type:

```
sudo raspi-config
```

You will be presented with a menu of options:



First, select “7 Advanced User Options”, then “Expand filesystem”, and hit enter. This will allow you to use all of the space on the MicroSD card. The program should tell you the filesystem has been resized. Hit Ok.

Next, select “Hostname”. This will allow you to name your Pi computer so it has a unique name on the network. Any letters and numbers, without spaces, will work. You can also use hyphens! A Hostname is a label that is assigned to a device connected to a computer network and that is used to identify the device.

Finally, select “Finish” at the bottom of the menu. This will save your settings and reboot the Pi. You will be disconnected from SSH, so just wait a minute or two then log in again.

Now, you should update all of the software on the Pi before proceeding. This will make sure everything is up to date, and you are installing the latest server software. Run the command:

```
sudo apt-get update
```

Wait for the system to download new packages, then tell you it is Done. Finally, run the command:

```
sudo apt-get upgrade
```

The upgrade program will show you a list of the software it will be updating, then ask you if you want to continue. Hit Enter to continue. You may be prompted during this process with questions about updating certain packages. It is usually safe to just hit Ok or proceed.

After this completes (and it might take a few minutes), the software on the Pi has been updated to the most recent versions. You are now ready to set up server software on your Pi.

Installing Local Servers

Below are instructions on installing two local application platforms on your Raspberry Pi: Wordpress for blogs, and Etherpad for collaborative text documents.

Wordpress

There are very detailed instructions on setting up the blog platform Wordpress on your Raspberry Pi already written. Rather than re-write all of these instructions, we recommend just following these steps:

https://www.raspberrypi.org/learning/lamp-web-server-with-wordpress/worksheet/

If there are choices between Raspbian Wheezy and Jessie, for this guide we are using Jessie. Throughout the guide, it may prompt you to browse to "http://localhost/" - you can instead put the hostname or IP address of your Pi in the browser you are using to set everything up. You should see the same screens, only on your computer rather than on the Pi itself.

Once everything is configured as in the guide, you will have a working blog platform on your Pi! You can browse to http://hostname-of-your-pi and start creating blog posts.

Etherpad

There are not comprehensive instructions already written for Etherpad, so please follow the instructions below to set it up!

First, we must install a number of packages on the Pi. Run the following commands, and at the end of each line, hit Enter.

```
sudo apt-get install gzip git curl python
sudo apt-get install libssl-dev pkg-config build-essential
cd ~
curl -sL https://deb.nodesource.com/setup_4.x | sudo bash -
sudo apt-get install -y nodejs
sudo npm install npm
sudo adduser --system --home=/srv/etherpad-lite --group etherpad-lite
cd /srv/etherpad-lite
sudo git clone git://github.com/ether/etherpad-lite.git
sudo chown -R etherpad-lite:etherpad-lite etherpad-lite
sudo nano /etc/systemd/system/etherpad-lite.service
```

In the editor window that appears, paste the following text, then hit CTRL-X to save and exit.

```
[Unit]
Description=etherpad-lite (real-time collaborative document editing)
After=syslog.target network.target

[Service]
Type=simple
User=etherpad-lite
Group=etherpad-lite
```

```
ExecStart=/srv/etherpad-lite/etherpad-lite/bin/run.sh

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

Then type the following commands:

```
sudo systemctl enable etherpad-lite
sudo systemctl start etherpad-lite
```

These commands will enable Etherpad to start when the Raspberry Pi boots, and then will start the server for the first time. Wait a few minutes. You won't see any messages print on the command line, but you can type the following:

```
sudo service etherpad-lite status
```

This will give you information on whether Etherpad has started correctly. You can run the command multiple times. After a while, the terminal will stop printing messages. It may take several minutes before Etherpad starts for the first time. You can browse to the following URL:

http://hostname-of-your-pi:9001

You should see a large "New Pad" button. Your Etherpad installation works!

Now, we have a few more steps to make Etherpad use a real database on the system.

First, type the following to stop Etherpad:

```
sudo service etherpad-lite stop
```

Now, enter the following commands:

```
mysql -u root -p
```

And enter the mysql password you created when installing Wordpress. Type the following at the "mysql>" prompt:

`create database \`etherpad-lite\`;` and hit Enter

`grant CREATE,ALTER,SELECT,INSERT,UPDATE,DELETE on \`etherpad-lite\`.* to 'root'@'localhost' identified by '<password>';` - but replace <password> with the root user password you setup earlier. Then hit Enter.

Hit CTRL-D to exit the mysql client. Then use the following commands:

```
cd /srv/etherpad-lite/etherpad-lite/
sudo nano settings.json
```

In this settings file, you can change the name that will appear at the top of your Etherpad when viewed in a web browser by editing the text after "title". You should now scroll down to the line that states "dbType", and remove all of the following lines:

```
  "dbType" : "dirty",
  //the database specific settings
  "dbSettings" : {
                   "filename" : "var/dirty.db"
                 },
```

```
  /* An Example of MySQL Configuration
```

Then remove the line with the “*/” characters from the bottom of the next section. The database type section should then look like:

```
   "dbType" : "mysql",
   "dbSettings" : {
                  "user" : "root",
                  "host" : "localhost",
                  "password": "the mysql root password you set before",
                  "database": "etherpad-lite",
                  "charset" : "utf8mb4"
                  },
```

Make sure you edit the “password” line.

After this, scroll down to the line that starts “users”, and change the passwords for the “admin” and “user” accounts. This will make sure that the admin panel for Etherpad is secure.

Now save and exit with “CTRL-X”.

Run the command:

```
sudo service etherpad-lite start
```

Wait a few minutes, then browse to http://hostname-of-your-pi:9001 again and check if everything works. You should now have a fully working instance of Etherpad Lite on your Raspberry Pi!

DNS Server (optional)

Depending on the configuration of your network, and the type of routers you are using, you may want to set up a DNS server. This should only be necessary in the case where you have a completely stand-alone network - one disconnected completely from the Internet. Some routers do not support custom hostname entries, so trying to resolve the hostname of your Raspberry Pi server may not work - for instance on some TP-Link routers. If we install a DNS server and set up hostnames, then we should be able to connect to the Pi no matter what!

First, we install the “bind” packages that contain the DNS server:

```
sudo apt-get install bind9 bind9-doc dnsutils
```

Next, we want to tell the Raspberry Pi it where to look up hostnames. The file that tells the Pi where to look up hostnames is “/etc/resolv.conf”. This file is generated automatically, when it receives a DHCP lease or in the static IP address configuration. We will need to edit the “dhcpcd.conf” file instead, so the other file is generated correctly:

```
sudo nano /etc/dhcpcd.conf
```

The file will have an existing configuration, but we want to add the following at the bottom:

```
interface eth0
static domain_name_servers=127.0.0.1 8.8.8.8 8.8.4.4
```

You can also choose to make your Raspberry Pi have a static IP address on the network it is connected to. In this scenario the Pi will no longer ask DHCP for an IP address, and it will always be accessible at the IP address you assign. Make sure you don't assign an address that is already taken, or that could be assigned by DHCP from the network's router. If you wish to do this, make the configuration in dhcpcd.conf look like this:

```
interface eth0
static ip_address=X.X.X.X/24    (where X.X.X.X is the IP for the Raspberry Pi)
static routers=Y.Y.Y.Y          (where Y.Y.Y.Y is the IP address of the router)
static domain_name_servers=127.0.0.1 8.8.8.8 8.8.4.4
```

Then save and exit the file.

Now, we need to create a DNS zone and names. A zone is the domain - for instance alliedmedia.org, google.com, and so on that Internet names resolve to. The actual names resolve to specific computers - human-readable names for computers on the network. We must edit the named.conf.local file first:

```
cd /etc/bind
sudo nano named.conf.local
```

This file defines the zones (also known as domains) that the server is responsible for. It should be mostly empty, possibly with some comments at the top. Edit the file and add the following:

```
zone "mesh.local" {
    type master;
    file "/etc/bind/zone.mesh.local";
};
```

You can replace "mesh.local" with any local domain you wish - it will only show up on your local network. You can also change the file name to anything, though it is recommended that it match your domain name somewhat.

Now, we create the zone file itself:

```
sudo nano zone.mesh.local
```

Then paste in the file below, changing the domain (mesh.local) and hostname (router, pi, etc) elements:

```
$TTL    604800
@       IN      SOA     mesh.local. root.mesh.local. (
                              1         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      raspi.mesh.local.
```

```
router  IN      A       192.168.0.1
raspi   IN      A       192.168.0.10
```

Then issue the command to restart the DNS server, which will prompt it to load your new zone file:

```
sudo service bind9 restart
sudo service bind9 status
```

You should be able to now resolve the hostnames you entered for specific IP addresses! You can test it with:

```
dig raspi.mesh.local
```

If you change the hostname or domain name, just make sure your “dig” command matches what you put in the zone file, and it should return the IP you assigned.