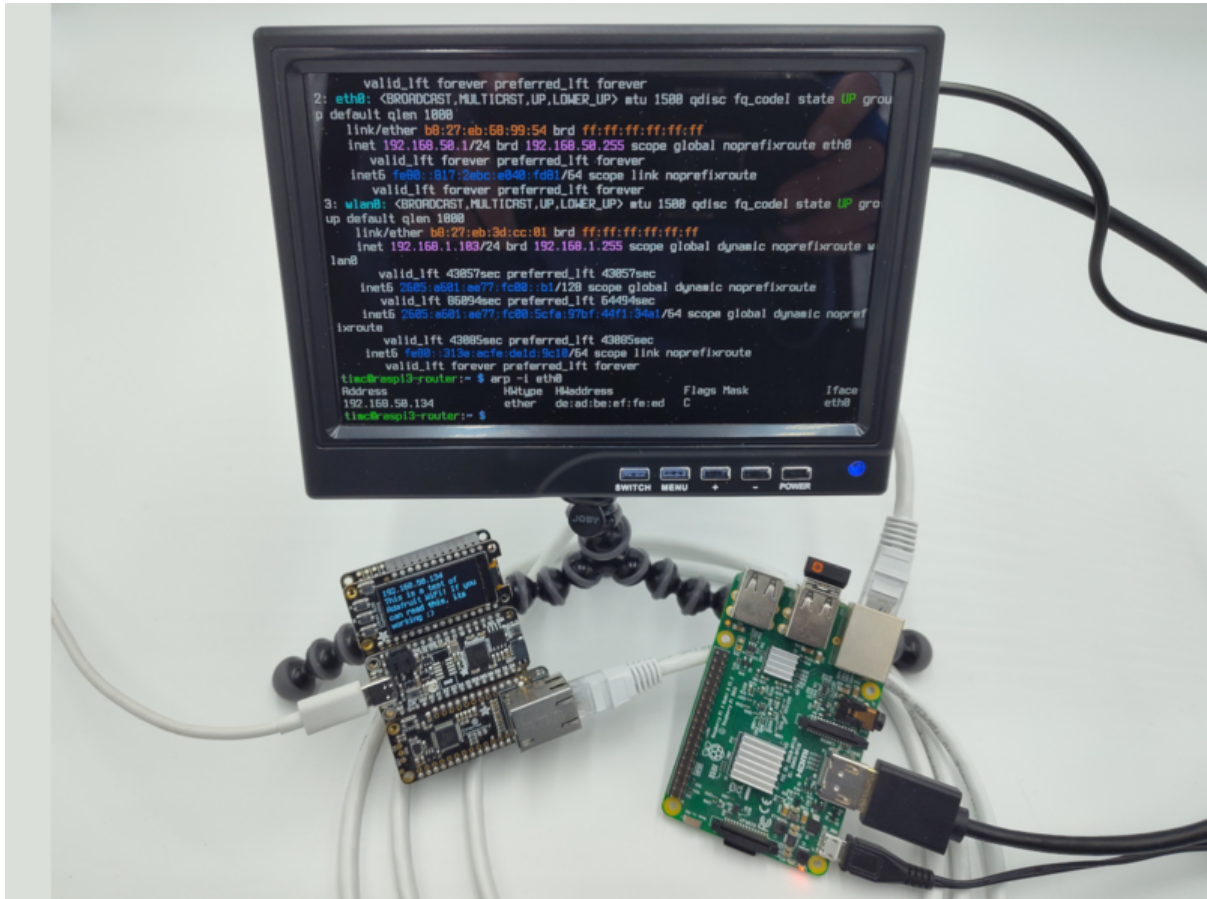




Using a Raspberry Pi as a Router

Created by Tim C



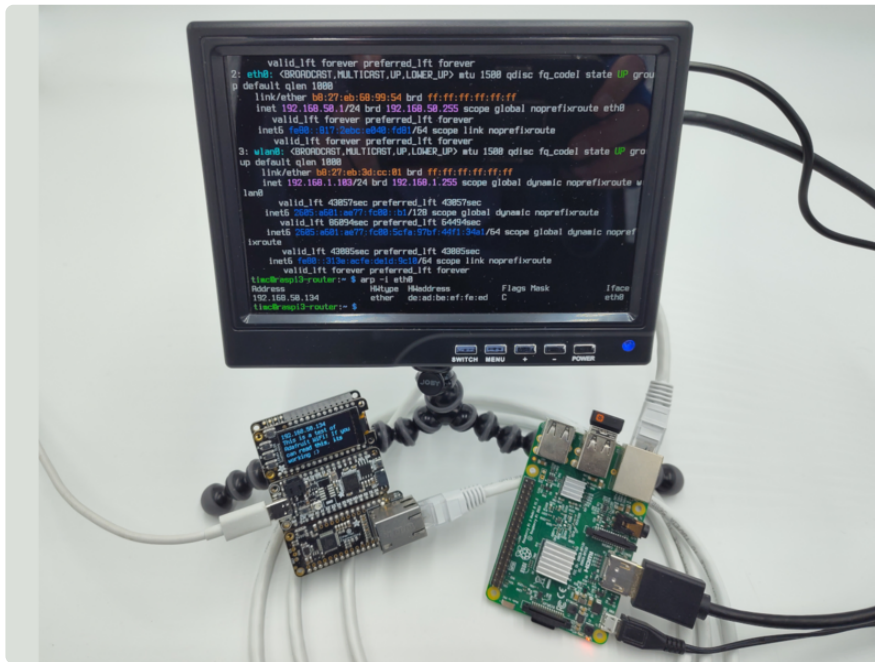
<https://learn.adafruit.com/using-a-raspberry-pi-as-a-router>

Last updated on 2026-04-11 01:48:31 AM UTC

Table of Contents

Overview	3
<ul style="list-style-type: none">• Parts	
Prepare	6
<ul style="list-style-type: none">• Raspberry Pi OS• Helpful Commands• nmcli	
WiFi Access Point	9
<ul style="list-style-type: none">• Upstream Ethernet To WiFi Access Point	
Network Bridge	12
<ul style="list-style-type: none">• Setup Network Bridge	
Ethernet Router	14
<ul style="list-style-type: none">• Upstream Ethernet to Downstream Ethernet• Upstream WiFi to Downstream Ethernet	

Overview



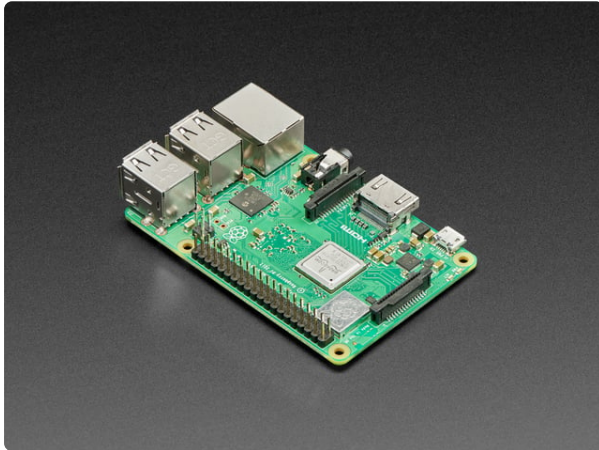
The United States [FCC recently announced a ban](https://adafru.it/1aCk) on new consumer-grade routers produced outside of the US. This does not affect existing devices that were already authorized, and there is a carve-out for manufacturers to apply for a conditional approval. It's difficult to say what the medium or longterm effects of the ban will be.

This got me thinking about what could be used as a makeshift router in a pinch. As it so happens, any computer that can run Linux and has networking interfaces can function as a router. [This blog post](https://adafru.it/1aCI) by Noah Baily documents the process using various old computers and components as custom routers over the years.

These makeshift routers are not going to win any bandwidth speed races, but they're perfectly capable of routing traffic for IoT devices or basic browsing. They're also useful for capturing traffic to analyze or sharing internet access from WiFi to Ethernet or vice-versa.

This guide documents the setup process and capabilities of using a Raspberry Pi as a router. It does not require a particularly powerful computer, even the older Pi 3 B+ that lots of us have tucked away in an old parts bin works fine for this.

Parts



Raspberry Pi 3 - Model B+ - 1.4GHz Cortex-A53 with 1GB RAM

The Raspberry Pi 3 Model B is the most popular Raspberry Pi computer made, and the Pi Foundation knows you can always make a good thing better! And what could make the Pi 3...

<https://www.adafruit.com/product/3775>



Raspberry Pi 4 Model B - 1 GB RAM

The Raspberry Pi 4 Model B is the newest Raspberry Pi computer made, and the Pi Foundation knows you can always make a good thing better! And what could make the Pi 4 better...

<https://www.adafruit.com/product/4295>



Raspberry Pi 5 - 1 GB RAM

The Raspberry Pi 5 is the newest Raspberry Pi computer, and the Pi Foundation knows you can always make a good thing better! And what could make the Pi 5 better than the...

<https://www.adafruit.com/product/6447>



Official Raspberry Pi 45W USB-C Power Supply

If you want a general-purpose USB Power Delivery supply, the official Raspberry Pi 45W USB-C power supply makes for a good quality PD supply that provides high current at a large...

<https://www.adafruit.com/product/6320>



5V 2.5A Switching Power Supply with 20AWG MicroUSB Cable

Our all-in-one 5V 2.5 Amp + MicroUSB cable power adapter is the perfect choice for powering single-board computers like Raspberry Pi, BeagleBone, or anything else that's...

<https://www.adafruit.com/product/1995>



Skinny Ethernet LAN UTP CAT6 Cable - 3mm diameter - 30cm long

When you want a super skinny Ethernet / CAT-6 cable, this noodle-y UTP jumper is only 3mm thick at the center so that it can flex around enclosures easily. This cable is...

<https://www.adafruit.com/product/5443>



USB 2.0 and Ethernet Hub - 3 USB Ports and 1 Ethernet

One can never have enough socks or USB ports. Add some more USB and Ethernet capability to your Raspberry Pi or, really, any kind of computer with this USB 2.0 and...

<https://www.adafruit.com/product/2909>



HDMI Cable - 1 meter

Connect two HDMI devices together with this basic HDMI cable. It has nice molded grips for easy installation, and is 1 meter long (about 3 feet). This is a HDMI 1.3...
<https://www.adafruit.com/product/608>



7" Display 1280x800 (720p) IPS + Speakers - HDMI/VGA/NTSC/PAL

Yes, this is an adorable small HDMI television with incredibly high resolution and built in 3W stereo speakers! We tried to get the smallest possible HDMI/VGA display with...
<https://www.adafruit.com/product/1667>

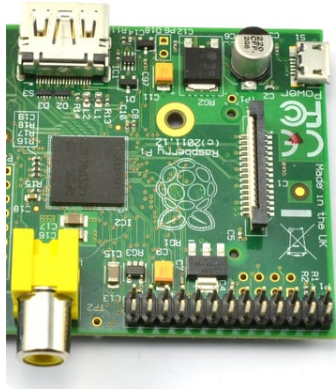
Prepare



Raspberry Pi 5, 4, 3, Zero W, and Zero 2 W can host a wireless network using the built-in wireless module. Raspberry Pi models that lack a built-in wireless module can support wireless functionality using a USB wireless adapter.

Raspberry Pi OS

Begin by installing the latest full or lite version of Raspberry Pi OS (64-bit) on your system using the steps outlined on this [Raspberry Pi Imager learn guide page \(https://adafru.it/ZDN\)](https://adafru.it/ZDN). If you intend to have the Raspberry Pi installed somewhere out of the way and wish to connect to it remotely, then be sure to enable SSH and configure your public key in the customization menu of the Raspberry Pi Imager. It's also convenient to configure the WiFi network details inside of the Pi Imager app so that the Pi will automatically connect to your WiFi when it boots up.



Adafruit's Raspberry Pi Lesson 1.
Preparing an SD Card for your Raspberry
Pi

By Simon Monk

▣ [Raspberry Pi Imager](#)

<https://learn.adafruit.com/adafruit-raspberry-pi-lesson-1-preparing-and-sd-card-for-your-raspberry-pi/raspberry-pi-imager>

After you boot up the Pi for the first time on a fresh Raspberry Pi OS image, run these commands to update the software preloaded in the system.

```
sudo apt update  
sudo apt upgrade
```

If these commands fail with network errors, use the WiFi settings in the OS to connect to your network and then try again.

Helpful Commands

While dealing with networks in a Linux based system there are some terminal commands that are helpful to inspect and configure different aspects of the network connections.

ip addr

The `ip addr` command outputs a list of network adapters with their current state and assigned IP address. This is helpful for determining which network interfaces are active and whether external USB adapters are being recognized by the system.

```
ip addr
```

The output can look intimidating but there are a few key pieces of information that are easy to spot once you know where to look.

In the following screenshot, colored highlights have been added to each of the 3 different network interfaces:

- Yellow for `lo`: loopback or localhost. Not relevant for externally connected devices.
- Green for `eth0`: Ethernet, disconnected in the screenshot as indicated by the red "DOWN".
- Pink for `wlan0` WiFi, connected in the screenshot as indicated by green "UP" and assigned IP address.

```
timc@raspi3-router:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
   inet 127.0.0.1/8 scope host lo
       valid lft forever preferred lft forever
   inet6 ::1/128 scope host noprefixroute
       valid lft forever preferred lft forever
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default qlen 1000
   link/ether b8:27:eb:68:99:54 brd ff:ff:ff:ff:ff:ff
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
   link/ether b8:27:eb:30:cc:01 brd ff:ff:ff:ff:ff:ff
   inet 192.168.1.255/24 brd 192.168.1.255 scope global dynamic noprefixroute wlan0
       valid lft 4313sec preferred lft 4313sec
   inet6 2685:a601:ac77:fc00:b827:ebff:fe30:cc01/64 scope global dynamic mngtmpaddr proto kernel_ra
       valid lft 4313sec preferred lft 4313sec
   inet6 fe80::b827:ebff:fe30:cc01/64 scope link proto kernel_ll
       valid lft forever preferred lft forever
timc@raspi3-router:~$
```

nmcli

The network manager can be accessed using the `nmcli` utility. It can do several common tasks like scanning for networks, connecting and disconnecting, creating and managing WiFi access points and more.

There are a handful of commands relevant to this guide below. The [nmcli documentation \(https://adafru.it/1aCx\)](https://adafru.it/1aCx) contains more comprehensive information.

Turn on or off the WiFi radio.

```
sudo nmcli radio wifi off
sudo nmcli radio wifi on
```

Print a list of all of the saved network connections. This includes any currently connected networks as well as networks that have been saved but aren't currently connected, and any WiFi access points that have been created.

```
nmcli connection show
```

Example output:

```
timc@raspi3-router:~$ nmcli connection show
NAME                UUID                                TYPE      DEVICE
netplan-eth0       75a1216a-9d1a-30cd-8aca-ace5526ec021  ethernet  eth0
Strawberry24       3f386f17-0086-4d1a-a4fb-e8306ecbf8e8  wifi      wlan0
lo                  b3f9153d-b2d9-4b73-8518-edba2d4a80d9  loopback  lo
Hotspot             807922e0-ebf4-40e7-9927-0c4f6702d09d  wifi      --
```

Scan for networks and print information about the nearby networks.

```
nmcli device wifi list
```

Connect to a WiFi network. Use the list command above to find the specific SSID to use. Fill in the SSID and password for your own network in the following command to connect to the network.

```
sudo nmcli device wifi connect <Network SSID> password <Network
Password>
# Example:
sudo nmcli device wifi connect MyWifiNetowrk password SuperS3cret
```

Enable or disable a saved connection. Enabling will connect to the network or start broadcasting the AP, disabling will disconnect or stop broadcasting AP.

```
sudo nmcli connection down <Connection Name>;
sudo nmcli connection up <Connection Name>;
# Examples:
sudo nmcli connection down Hotspot
sudo nmcli connection up MyWifiNetowrk
```

Delete a saved connection or WiFi hotspot access point.

```
sudo nmcli connection delete <Connection Name>;
# Examples:
sudo nmcli connection delete Hotspot
sudo nmcli connection delete MyWifiNetwork
```

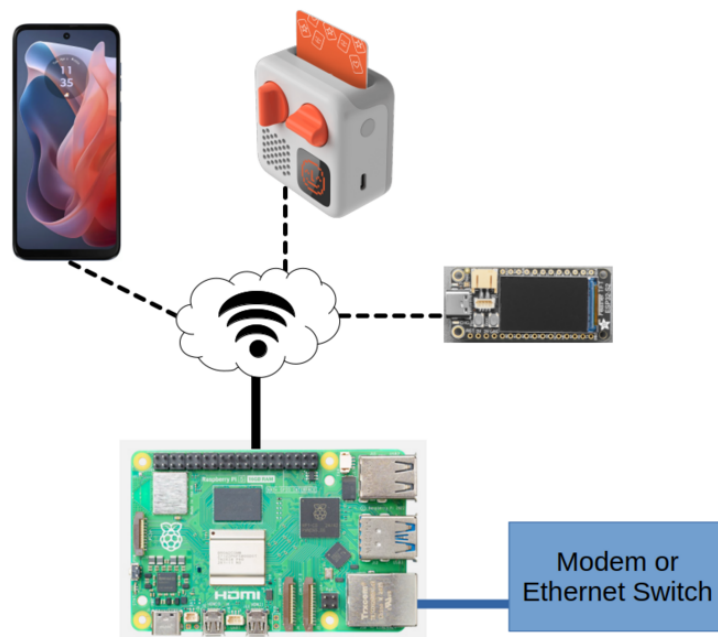
WiFi Access Point

This page will document how to set up your Raspberry Pi to share internet access from another source as a custom WiFi access point. If the internet source is Ethernet,

then no additional hardware is required. If you have a USB WiFi adapter that the Pi kernel supports, then you can use another WiFi network as the internet source for the hosted AP instead of Ethernet.

Upstream Ethernet To WiFi Access Point

The most basic thing to set up is sharing internet from the Pi's Ethernet as a WiFi access point. This allows you to connect WiFi enabled IoT devices, smartphones, and other computers to the Pi's hosted AP and access the internet from them.



To create a WiFi access point, also known as a hotspot, run this command:

```
sudo nmcli device wifi hotspotssid <AP SSID> password <AP Password>
# Example:
sudo nmcli device wifi hotspot ssid GadgetNetwork password SuperS3cret
```

If successful, it will output a message like this:

```
Device 'wlan0' successfully activated with '5c14f89d-97ab-4b27-9361-e5a3b40d0036'.
Hint: "nmcli dev wifi show-password" shows the Wi-Fi name and password.
```

802.1X Supplicant Issue

On Raspberry Pi 3B models there is a [reported issue \(https://adafru.it/1aCy\)](https://adafru.it/1aCy) that can cause the error `Error: Connection activation failed: 802.1X supplicant took too long to authenticate.`

If you don't get that error then you can skip this section.

If you do get this error, you can work around it by disabling Protected Management Frames or PMF configuration. Note that PMF protects against certain types of attacks between devices on the network.

If you are not in a fully trusted network environment or are unsure about this, then just leave the default and try a different model of Raspberry Pi, a USB WiFi adapter, or the previous major release of the OS instead of Trixie.

These commands will disable PMF and then attempt to enable the hotspot:

```
sudo nmcli con modify Hotspot 802-11-wireless-security.pmf 1
sudo nmcli con up Hotspot
```

Upstream WiFi to WiFi Access Point

If you have a USB WiFi adapter connected to the Pi, you can connect to an upstream network with the Pi's built-in WiFi radio and host an access point using the USB adapter or vice versa. This can be helpful if you want to inspect traffic for specific client devices or perhaps to extend the range of a WiFi network using a Pi near the edge of the range with a USB WiFi adapter that has a nice antenna to boost the hosted AP further.

If you are trying to do something like the WiFi network extender then you'll need to make sure the correct network interface is used for upstream connection and hotspot respectively. In this case, you'd want the hotspot to be broadcast using the USB WiFi adapter and the upstream network connection with the Pi's built-in radio. The built-in radio should be interface `wlan0` and externally connected ones will count up from there, so if you have a single USB WiFi adapter connected, it should be `wlan1`. The `nmcli device wifi` commands accept an `ifname` argument to specify the interface to use.

The commands below illustrate how to connect to an upstream network on `wlan0` the built-in radio, and host the hotspot on `wlan1` the USB WiFi adapter.

```
# Connection to a Network using the builtin radio specifically
sudo nmcli device wifi connect <Network SSID> password <Network
```

```
Password&gt; ifname wlan0
```

```
# Create a WiFi access point using externally connected WiFi adapter  
sudo nmcli device wifi hotspot ssid &lt;AP SSID&gt; password &lt;AP Password&gt;  
ifname wlan1
```

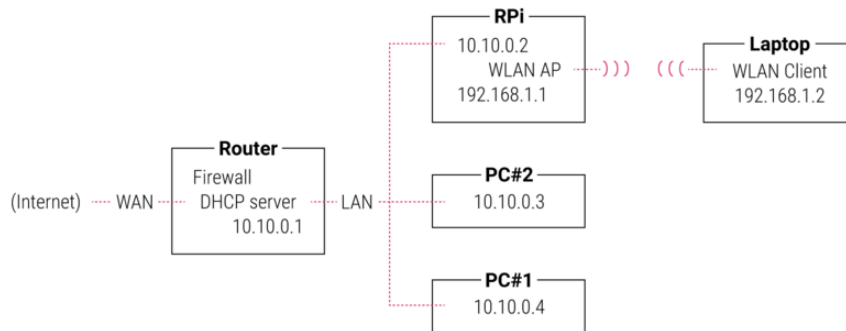
Network Bridge

By default, creating access points following the directions on the previous page will result in a completely isolated network hosted by the AP. Clients on the AP network will be able to access the internet, but they will not be able to communicate directly with other clients that are connected to the upstream network. If you want to enable this direct communication between clients across the different networks, you need to set up a network bridge.

The following two diagrams from the official [Raspberry Pi documentation \(https://adafruit.it/YJC\)](https://adafruit.it/YJC) illustrate the difference.

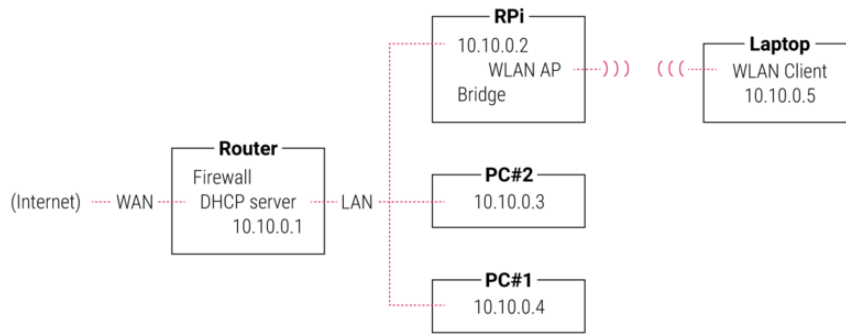
Un-bridged Network (default)

The AP network gets a different IP address space and clients on the different network segments cannot communicate directly.



Bridged Network

The AP network uses the same IP address space as the upstream network. Clients on the different network segments can communicate directly using their assigned IPs.



Setup Network Bridge

To setup the network bridge, you need to create a bridge connection with `nmcli` and then assign the bridge as the master for each of the sub-networks.

First create the bridge connection with this command:

```
sudo nmcli connection add type bridge con-name 'Bridge' ifname bridge0
```

Next, add the first sub-network with the bridge configured as its master. This example command connects an Ethernet connection to the bridge. If you're doing WiFi to WiFi, then substitute in an appropriate connection name and interface.

```
sudo nmcli connection add type ethernet slave-type bridge con-name 'Ethernet' ifname eth0 master bridge0
```

Lastly, delete and re-create the hotspot connection, assigning the bridge as its master:

```
# If you already have an un-bridged Hotspot network delete it
sudo nmcli connection delete Hotspot

# create a new Hotspot connected to bridge
sudo nmcli connection add con-name 'Hotspot' ifname wlan0 type wifi slave-type bridge master bridge0 wifi.mode ap wifi.ssid <hotspot-ssid> wifi-sec.key-mgmt wpa-psk wifi-sec.proto rsn wifi-sec.pairwise ccmp wifi-sec.psk <hotspot-password>
```

`nmcli con show` should now list all of the following: Bridge, Hotspot, and Ethernet as in the screenshot below.

```
timc@raspi5-router:~$ nmcli con show
NAME                UUID                                  TYPE      DEVICE
netplan-eth0       75a1216a-9d1a-30cd-8aca-ace5526ec021 ethernet  eth0
Hotspot             daeb1761-70c9-4964-a5a3-601ede63924c wifi      wlan0
Bridge             c4d8022e-3bf7-4a15-aba5-fa83680fba99 bridge    bridge0
lo                 ebb084fe-425f-4128-96bf-5a481c8dcf54 loopback  lo
Ethernet           3686adac-b802-4f95-a740-c99a7934bc96 ethernet  --
```

Now, just bring down and back up the two child connections followed by the bridge, to make everything active.



u're connected to the Pi via SSH, your connection may drop while running e commands. Just wait a few seconds and re-connect or use a monitor mouse/keyboard if needed.

```
sudo nmcli con down Hotspot
sudo nmcli con down Ethernet
sudo nmcli con down Bridge

sudo nmcli con up Hotspot
sudo nmcli con up Ethernet
sudo nmcli con up Bridge
```

Once the network is back up you can connect device to the AP and they'll be assigned IPs and be able to communicate with clients on the upstream network.

Ethernet Router

It's also possible to use the Ethernet connection on the Raspberry Pi for the downstream network instead of a WiFi access point. In this configuration, the Pi connects to an upstream network via either WiFi or Ethernet, and bridges that connection to a downstream device connected via Ethernet.



aspberry Pi devices have only one Ethernet port, so if you want to use Ethernet for both upstream network and downstream clients you will need to connect additional ports. One option are USB Ethernet adapters such as this from the Adafruit shop.



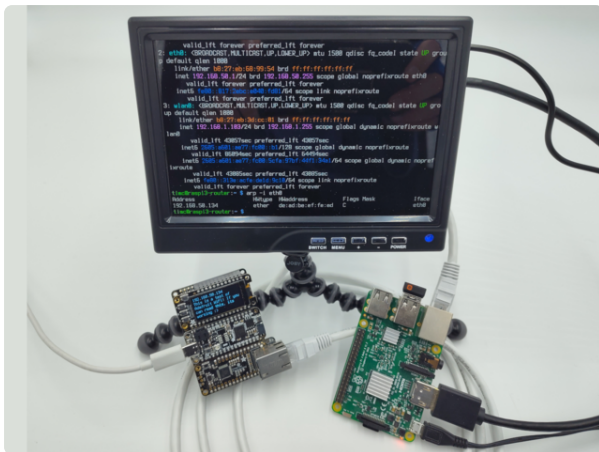
USB 2.0 and Ethernet Hub - 3 USB Ports and 1 Ethernet

One can never have enough socks or USB ports. Add some more USB and Ethernet capability to your Raspberry Pi or, really, any kind of computer with this USB 2.0 and...

<https://www.adafruit.com/product/2909>

Upstream Ethernet to Downstream Ethernet

This process is essentially the same as setting up the network bridge documented on the previous page, so it's a good idea to read that page even if you don't intend to use a WiFi hotspot for the downstream network.



Using a Raspberry Pi as a Router By Tim C

▢ [Network Bridge](#)

<https://learn.adafruit.com/using-a-raspberry-pi-as-a-router/network-bridge>

First create the bridge.

```
sudo nmcli connection add type bridge con-name 'Bridge' ifname bridge0
```

Then create two more Ethernet connections with the bridge configured as their master. The following commands will create two more connections named "**BuiltinEthernet**" and "**USBEthernet**" so that it's easy to tell which one is which from the `nmcli con show` output list.

```
sudo nmcli connection add type ethernet slave-type bridge con-name 'BuiltinEthernet' ifname eth0 master bridge0
```

```
sudo nmcli connection add type ethernet slave-type bridge con-name 'USBEthernet'  
ifname eth1 master bridge0
```

Now bring everything down and back up again to activate the connections.



u're connected to the Pi via SSH, your connection may drop while running e commands. Just wait a few seconds and re-connect or use a monitor mouse/keyboard if needed.

```
sudo nmcli con down BuiltinEthernet  
sudo nmcli con down USBEthernet  
sudo nmcli con down Bridge  
  
sudo nmcli con up BuiltinEthernet  
sudo nmcli con up USBEthernet  
sudo nmcli con up Bridge
```

Once everything is back up, the Pi will pass traffic between the upstream network and downstream connected device.

Upstream WiFi to Downstream Ethernet

Sharing an upstream WiFi connection with a downstream Ethernet connection is a little bit tricky because most WiFi networks do not support the modes necessary to accept directly bridged Ethernet traffic. It's possible to achieve the similar result, but it requires additional steps. Instead of using a bridge you'll have to set up a few more utilities to handle DHCP and forward traffic across the networks.

Install the required utilities with this command.

```
sudo apt install dnsmasq iptables iptables-persistent
```

Connect to the upstream WiFi network if you haven't already.

```
sudo nmcli device wifi connect <Network SSID> password <Network  
Password>
```

Create an Ethernet connection with a static IP for the downstream subnet.

```
sudo nmcli con add type ethernet ifname eth0 con-name BuiltinEthernet ipv4.method manual ipv4.addresses 192.168.50.1/24 ipv4.never-default yes
```

Bring the new Ethernet connection up.

```
sudo nmcli con up BuiltinEthernet
```

Enable IP forwarding with this command:

```
sudo sysctl -w net.ipv4.ip_forward=1
```

To make the IP forwarding configuration persistent, you will need to create a service and enable it with the following commands. Do this if you intend to keep using the Pi this way, but if you just want to experiment temporarily, then skip it.

```
sudo tee /etc/systemd/system/ip-forward.service <<< 'EOF'
[Unit]
Description=Enable IP forwarding
After=network-online.target
Wants=network-online.target

[Service]
Type=oneshot
ExecStart=/sbin/sysctl -w net.ipv4.ip_forward=1
RemainAfterExit=yes

[Install]
WantedBy=multi-user.target
EOF

sudo systemctl enable ip-forward.service
```

Set up [NAT \(https://adafru.it/1aCz\)](https://adafru.it/1aCz) with the `iptables` utility.

```
sudo iptables -t nat -A POSTROUTING -o wlan0 -j MASQUERADE
sudo iptables -A FORWARD -i eth0 -o wlan0 -j ACCEPT
sudo iptables -A FORWARD -i wlan0 -o eth0 -m state --state RELATED,ESTABLISHED -j ACCEPT
```

If you want the `iptables` settings to persist across reboots, run this command:

```
sudo netfilter-persistent save
```

The next step is to set up a DNS server to run on eth0 using the `dnsmasq` utility.

Open the `dnsmasq` configuration file with `nano` or your preferred text editor.

```
sudo nano /etc/dnsmasq.conf
```

Paste the following into the file at the top above the comments to configure the DHCP server.

```
interface=eth0
dhcp-range=192.168.50.50,192.168.50.150,255.255.255.0,24h
dhcp-option=option:router,192.168.50.1
dhcp-option=option:dns-server,192.168.50.1
```

Save the file by pressing `ctrl-S` then exit nano with `ctrl-X`.

Lastly restart the server and enable it to run automatically as a service.

```
sudo systemctl restart dnsmasq
sudo systemctl enable dnsmasq
```

Now when you connect a downstream device to the Pi's Ethernet port, it will get assigned an IP like `192.168.50.X` where `X` is in the range of `50` to `150`. The downstream device will have access to the internet through the upstream WiFi network.

