Setting up Bluetooth serial connectivity from your Pi

Initial setup

Download the latest Raspian image: https://downloads.raspberrypi.org/raspbian_latest. Unzip the archive. Assuming the image file is `2021-10-30-raspios-bullseye-armhf-full.img`, type the following in WSL:

```
$ sudo fdisk -l 2021-10-30-raspios-bullseye-armhf-full.img

Disk 2021-10-30-raspios-bullseye-armhf-full.img: 8.6 GiB, 9265217536 bytes, 18096128 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xceb251fe

Device                                     Boot  Start      End  Sectors  Size Id Type
2021-10-30-raspios-bullseye-armhf-full.img1        8192   532479   524288  256M  c W95 FAT32 (LBA)
2021-10-30-raspios-bullseye-armhf-full.img2      532480 18096127 17563648  8.4G 83 Linux
```

Since the sectors are 512 bytes in length, the first partition starts at offset 512 x 8192 = 4194304 and contains 524288 sectors, so has size 512 x 524288 = 268435456. The second partition starts at sector 532480, so its offset is 512 x 532480 = 272629760.

Create two directories as mount points:

```
$ sudo mkdir /mnt/2021-10-30-raspios-bullseye-armhf-full.img1
$ sudo mkdir /mnt/2021-10-30-raspios-bullseye-armhf-full.img
```

Then mount the images:

```
$ sudo mount -v -o offset=4194304,sizelimit=268435456 -t vfat 2021-10-30-raspios-bullseye-
armhf-full.img /mnt/2021-10-30-raspios-bullseye-armhf-full.img1

$ sudo mount -v -o offset=272629760 -t ext4 2021-10-30-raspios-bullseye-armhf-full.img
/mnt/2021-10-30-raspios-bullseye-armhf-full.img2
```

Copy the contents of the os directory into `/mnt/2021-10-30-raspios-bullseye-armhf-full.img2/`:

```
$ sudo cp -R os/* /mnt/2021-10-30-raspios-bullseye-armhf-full.img2/
```

Edit `/mnt/2021-10-30-raspios-bullseye-armhf-full.img2/etc/machine-info` to give your Pi a meaningful name (sudo required to change this file).

Edit `/mnt/2021-10-30-raspios-bullseye-armhf-full.img2/etc/rc.local` and add the following line at the bottom, before the line with `exit 0` (sudo required to change this file):

```
sudo /home/pi/btserial.sh &
```

Make `/mnt/2021-10-30-raspios-bullseye-armhf-full.img2/home/pi/btserial.sh` executable:

```
$ chmod +x /mnt/2021-10-30-raspios-bullseye-armhf-full.img2/home/pi/btserial.sh
```

Validate that the file `/mnt/2021-10-30-raspios-bullseye-armhf-full.img2/etc/bluetooth/main.conf` contains the following lines, and that they are uncommented:

```
ControllerMode=bredr
Privacy=off
AutoEnable=true
```

There have been issues reported concerning Bluetooth not working out of the box on the Raspberry Pi 4 Model B, and I've encountered these issues too, but after some diagnosis, the above setting seemed to resolve this.

The setting `ControllerMode=le` has also been proven to work for some users. You may need to experiment. Once I'd logged on via SSH, and then via VNC, I was able to pair with mobile phones and a PC and log into the shell on the Pi. (Connecting from a PC uses a COM port, see below).

The Raspberry Pi Imager has advanced options for setting up your Wifi on the first boot. See below for details.

Unmount the partitions in WSL:

```
$ sudo umount /mnt/2021-10-30-raspios-bullseye-armhf-full.img1/
$ sudo umount /mnt/2021-10-30-raspios-bullseye-armhf-full.img2/
```

Burn the image to your SD card. Raspberry Pi Imager is recommended for this: https://www.raspberrypi.com/software/. Choose an OS and select the option "Own Image". Select the img file which you just modified. Advanced options, such as hostname, the pi user's password, SSID and password of the WiFi network you wish to connect to initially, and locale, can be configured here. Nice and easy. You can also opt to enable SSH. The Advanced Options dialog is shown by typing Ctrl+Shift+X.

Unmount the SD card from Windows and insert it back into the Pi. Start the Pi.

Connecting

Pair your PC with the Pi. You can connect to a serial port in puTTY. Just figure out which COM port (in Bluetooth settings) is used for the outgoing connection to the Pi's Bluetooth connection, then configure puTTY to use this port. The port you're connecting to on the Pi is 115200 (as configured in the btserial.sh).

Connect via puTTY, then you can use sudo raspi-config to configure the network, as described here: https://www.raspberrypi.org/documentation/configuration/wireless/wireless-cli.md

Connecting this way doesn't require you use a password, so it's really wide open. I opted to remove the credentials from my version of btserial.sh, as soon as you've set your password, so you'll need to enter your username and password every time you log on. A method which I much prefer.

So, once connected, in the file `/etc/systemd/system/rfcomm.service`, change the line:

```
ExecStart=/usr/bin/rfcomm watch hci0 1 getty rfcomm0 115200 vt100 -a pi
```

to:

```
ExecStart=/usr/bin/rfcomm watch hci0 1 getty rfcomm0 115200 vt100
```

You might also notice that this file has the executable flag set, in which case you can remove that:

```
$ sudo chmod -x rfcomm.service
```

This line can also be removed from `/etc/rc.local`, since the Bluetooth service is now enabled

```
sudo /home/pi/btserial.sh &
```

If you're still paranoid about having logins possible over Bluetooth other than your own, to disallow users connecting over the Bluetooth serial port altogether, you'll need to disable the rfcomm service (`sudo systemctl disable rfcomm`). This will stop users connecting over the Bluetooth serial port, and will stay disabled after subsequent reboots. To be able to log back in over Bluetooth serial, re-enable the service (`sudo systemctl enable rfcomm`) while gaining access some other way, such via ssh or VNC.

To get serial Bluetooth working from an Android phone, fork or clone https://github.com/DoomyDwyer/termux-app, build that in Android Studio and deploy the APK built during the build to your Android. Alternatively, you can download the APK from https://github.com/DoomyDwyer/termux-app/releases/tag/v0.59_bt and follow the instructions set out for that release.

Basically, wherever you are with your Pi, you won't need a screen & keyboard to get it set up. With the Bluetooth you can obviously set up the wifi wherever you are, if that's available, with sudo raspi-config, or at least query your ip address with ifconfig.

Once you have an IP address, you can obviously use VNC. That's available for Android but of course is much easier to work with on a laptop. But if there is no wifi available, you can perform all the cli tasks that you might need to over Bluetooth, with either puTTY or Termux on Android (not available on the Play Store release, you'll have to build this yourself).

This does enable you however to be able to connect to any Pi from your laptop over Bluetooth in order to set up the wifi, as long as the btserial stuff has been set up first, meaning you'll never have to lug around a screen, keyboard & mouse anywhere ever again.