To start this document out with a simple sentence: “The Raspberry Pi’s default configurations have spotty bluetooth capabilities at best”. To expand on this, the Raspbian image that was used (Raspberry PI OS Full (32-bit) (Released 2021-05-07)) comes prepackaged with a version of ‘BlueZ’ that can handle basic bluetooth functions - but only “the bare minimum of the bare bones basics”. A workaround was researched to correct this issue and allow integration of the Polar H10 BLE (Bluetooth Low Energy) heart rate monitor - and by extension other BLE devices (though this is purely speculative and has not been tested/verified by Ataraxia).

To begin the process of upgrading the bluetooth capabilities of the Raspberry Pi Zero w, navigate to the BlueZ official website (<http://www.bluez.org/download/>) to find the most recently supported, full build version number. Since SAFE-T is utilizing the Bleak Python module to interact with the Polar H10, a BlueZ version number of 5.43 or later is required in order for Bleak to work correctly (if at all). For this reason, SAFE-T is equipped with version 5.43 to reduce the risk of new versions introducing conflicts with the SAFE-T architecture.

Start up the Raspberry Pi, open a console terminal, and enter the following code (be sure to hit ‘enter’ after each line). These commands will ‘zip’ a tar file to your file directory, and we will use this to continue installation of the updated BlueZ.

| cd ~ wget http://www.kernel.org/pub/linux/bluetooth/bluez-5.43.tar.xz tar xvf bluez-5.43.tar.xz |
| --- |

Change the current directory to the bluez version being used (in this case 5.43).

| cd bluez-5.43 |
| --- |

Enter the following commands below to search for updates, and install the required dependencies. The second and third lines are a single entry; do not press ‘enter’ after the second line.

| sudo apt-get update sudo apt-get install -y libusb-dev libdbus-1-dev libglib2.0-dev libudev-dev libical-dev libreadline-dev |
| --- |

With the dependencies installed, BlueZ will be installed to the Raspberry Pi in the same manner as many other Linux software packages: “configure”, “make”, “make install”. Enter the following command into the terminal to start this process.

| ./configure --enable-library |
| --- |

If any errors appear in the terminal, double check that the dependencies were all installed - and if the errors make note of any specific dependencies missing, look up those dependencies and install them before repeating the previous steps.

If no errors appear, then you are good to continue with this guide by entering the following command. This process can take anywhere from 10 to 30 minutes on average to finish, and one installation on a poor WiFi connection took just under 40 minutes to complete. If the “make” command results in any errors, go back and double check that the “configure” step didn’t report any errors as well that may have been overlooked.

| make |
| --- |

| sudo make install |
| --- |

For some reason, the default installation of BlueZ packages does not install the ‘gatt tool’ package to the correct directory and must be manually installed with the following command while still in the BlueZ-5.43 directory.

| sudo cp attrib/gatttool /usr/bin/ |
| --- |

Now that the BlueZ version 5.43 (or your newer version) has been installed and implemented, it is time to apply further configurations to the installation to allow the service to operate with BLE devices. You can check that your newly installed bluetooth service is working by entering the following command.

| systemctl status bluetooth |
| --- |

In order to allow the BlueZ installation to be activated at bootup instead of having to manually disable the default service and then manually enabling the BlueZ service, the following command must be entered, and the device should be rebooted afterwards for the changes to take effect.

| sudo systemctl enable bluetooth |
| --- |

Once rebooted, the BlueZ configuration files should be modified (command immediately below). BLE is considered an ‘experimental’ application of the BlueZ service and as such requires the user to go through these steps to enable.

| sudo nano /lib/systemd/system/bluetooth.service |
| --- |

With the file opened in the nano text editor (or the text editor of your choice), you should see something akin to what is shown below. The change required to this document is adding

“ -- experimental” at the end of the line starting with “ExecStart” (this is shown below the example text block).

| [Service] Type=dbus BusName=org.bluez ExecStart=/usr/local/libexec/bluetooth/bluetoothd  NotifyAccess=main #WatchdogSec=10 #Restart=on-failure CapabilityBoundingSet=CAP\_NET\_ADMIN CAP\_NET\_BIND\_SERVICE LimitNPROC=1 |
| --- |

| ExecStart=/usr/local/libexec/bluetooth/bluetoothd --experimental |
| --- |

The BlueZ service can be restarted with the following command, and the entire bluetooth service can be rebooted with the second command.

| sudo systemctl daemon-reload |
| --- |

| sudo systemctl restart bluetooth |
| --- |

The final step is to reboot the Raspberry Pi to triple check that the bluetooth service and BlueZ service are fully updated and configured.

These steps were taken primarily from Adafruit at the link below.

<https://learn.adafruit.com/install-bluez-on-the-raspberry-pi/installation>