Keith Erkert

kerkert@asu.edu

Updated 11/12/2022

Abstract

This guide will walk the user through how to setup a Raspberry Pi as and access point for your IoT devices, install our software, and (optional) setup a database/backend application to run your own servers to store/handle data request from the app.

Raspberry Pi  
 Setup Guide

Table of Contents

[Introductory 2](#_Toc106642742)

[Hardware setup 2](#_Toc106642743)

[Raspberry Pi Initial Setup 3](#_Toc106642744)

[Softflowd Setup 4](#_Toc106642745)

[NFDump Setup 5](#_Toc106642746)

[Wifi Setup 7](#_Toc106642747)

[Run the Raspberry Pi App 9](#_Toc106642748)

[Setup Database and Backend Applications (Optional) 9](#_Toc106642749)

# Introductory

The Raspberry Pi is the main hardware component to monitor your IoT devices. It will act and an access point for the devices you wish to monitor and send it to either or database (recommended for those less tech savvy). The Pi can also be setup to act as the login/database holder as well.

Notes:

* Sections where a tool is installed mention updating the apt repository (sudo apt-get update commands). This guide assumes that users may reference back to it after the initial setup to fix problems with certain tools. Updating the repository is not necessary after ever tool install during the initial setup.

# Hardware setup

These are the requirements to run our app:

* Raspberry Pi 3 b+ or 4 b (

\*tools/software not tested on earlier version of Raspberry Pi

* 8gb micro-SD card

\*16gb or 32gb recommended if running the server software

* An ethernet connection for the Pi
* Monitor, keyboard, and mouse for initial setup
* A SD card adapter to connect the MicroSD to a PC for installing the initial image

# Raspberry Pi Initial Setup

1. Download the Raspberry Pi OS imager from <https://www.raspberrypi.com/software/>
2. Insert the micro-SD card into your desktop using an adapter
3. Using the Raspberry Pi OS imager select the default option for the operating system (Raspberry Pi OS 32bit)

Graphical user interface, text, application, email

Description automatically generated

1. Select the SD card to write the OS to
2. Click Write to start imaging the SD card
3. Plug a keyboard and mouse into the Pi and connect it to a monitor
4. Connect the pi to an open ethernet connection on your router/access point
5. Under the Start Menu > Preferences > Raspberry Pi Configuration > Interfaces an enable SSH and VNC
   1. It is recommended that you change from the default username (Pi) and password as someone could potential access the Raspberry Pi.
6. Restart the Raspberry Pi and disconnect the monitor, keyboard, and mouse.

\*The next steps are if you want to remotely connect to the Pi from now on, while it is not necessary, it will make future access a lot easier.

1. On your main computer download [VNC Viewer](https://www.realvnc.com/en/connect/download/viewer/)
2. Open VNC Viewer and add a new connection.
   1. For the VNC Server use either the defined local IP address your router gave it, or use the hostname Raspberry Pi (this will not work right if you have multiple Pis on your network)
   2. Leave other setting at default option
   3. Connect to the Pi, Accept the security certificate
   4. Sign into the Pi using your Pi’s username/password
      1. (optional) Check remember password if you want it to auto sign-in
3. Go to Start Menu > Preferences > Raspberry Pi Configuration > Display and change the Headless Resolution to one of your preferences

# Softflowd Setup

**\*Required**, this service captures the network traffic that goes through the Pi.

1. Open a Terminal and run



1. Configure Softflowd by change the configuration file.



1. Add/remove lines to make the file look like this

Text

Description automatically generated

1. Press **ctrl + x** then **y** to save file and exit nano editor.
2. Softflowd is now setup

# NFDump Setup

**\*Required**, This service takes the binary files Softflowd and give use a ASCII dump of the flows captured which our program will read and send to the database

1. In a Terminal run



1. Configure NFDump output folder and time to live for files



1. Update NFDump default polling time for pcap files (from 5 minutes to 1 minute)

*\*Optional, allows the app to run closer to real-time data, 1 minute is the lowest time NFDump allows*



1. Press **ctrl + x** then **y** to save file and exit nano editor.
2. Reboot Pi (sudo reboot)
3. NFDump is now setup

# IPTables Setup

**\*Optional**, This service allows the app to manage the raspberry pi’s network and block ips and devices from access to the internet. IPTables may already be installed.

\*\*Currently the app does not check if this was not setup\*\*

1. In a Terminal run



1. Check IPTables current rules



1. Please record any current rules incase a fatal error occurs during the application setup

Our app should only manipulate changes from our companion app and not touch any rules already set in place by the user. Our app will use a custom chain called “IOTAPP” and should only make changes to rules within this chain. The app will add a rule to the default FORWARD chain which will jump to our IOTAPP chain rules. After which, the app will only change IOTAPP chain rules going forward. On subsequent launches the app will always verify that FORWARD jump to IOTAPP still exists.

Users can opt out of our app from changing the iptables during the initial setup of our app.

# Wifi Setup

**\*Required**, This is what your IoT devices you want to monitor will be connecting too. This will set up the Pi as an Access Point

1. In a Terminal run, to install the software to help us setup an AP (Access Point)



1. Let’s now install a firewall rules helper library, choose yes for both popup options



1. Open dhcpcd.conf and configure the WLAN



Add the following to the end of file



* 1. For the IP address, we want to use a subnet that is not used on our home network
     1. If your IP address in your home network is 192.168.1.# then switch the last 1 to any number between 2-255. This will create a subnet

1. Press **ctrl + x** then **y** to save file and exit nano editor.
2. Now we need to route the Wifi clients traffic to the local network, create a file to forward the wifi traffic



And add this line to the file



1. Press **ctrl + x** then **y** to save file and exit nano editor.
2. Issue the following commands to setup routing



1. Setup the wireless interface, make a backup of the current configuration and open the configuration file



1. Add the following to the end of the configuration file, make sure the subnet matches what you choose in step 3



1. Enable the Raspberry Pi for transmission by issuing the following command and opening another configuration file



1. Add the following to the configuration file and change the **wpa\_passphase** and **ssid**



\* Consider using a phone application that can scan Wi-Fi signals around your home and look for the best channel (on 2.4ghz). This will be the channel number with the least overlapping Wi-Fi signals, change the channel number in the configuration file.

1. Save the file **ctrl + x**
2. Reboot the Raspberry Pi (sudo reboot) and check your phones Wi-Fi to see if it shows up.

# Run the Raspberry Pi App

1. Download the Raspberry Pi application from GitHub.
   1. Raspberry App – <https://github.com/ASU-IoT-ResearchProject/IoTRaspberryPi/releases/tag/v22w45d7a>
2. Open a terminal in the folder where the file was downloaded and run the following commands



1. The app will walk you through the first time setup and create 2 files to track configuration and previous devices.

# Setup Database and Backend Applications (Optional)

If you do not want to use our servers to handle your data, you can setup the database and backend applications on a Raspberry Pi. Please note that running the database, backend applications, and using the Raspberry Pi as an access point for your IoT devices on the same Pi may have performance issues that will cause it to overheat and shut down. We recommend either using a separate Raspberry Pi, or dedicated linux based server for this.

If using a Linux server, please refer to our Database and API server Guide.

<https://github.com/ASU-IoT-ResearchProject/Hub/blob/main/BackendApps.docx>

**Install .NET libraries** (From Microsoft official guide: <https://docs.microsoft.com/en-us/dotnet/iot/deployment>)

1. Open a terminal in a temporary folder
2. Run the following command, this will run an install script from microsoft



1. With the bash still open run the next commands



1. Back out of the bash and run the following to verify dotnet works



Install and Setup MariaDB

1. Update package manager and install MariaDB



1. Run setup for MySQL



1. Enter a password (remember this password)

Text

Description automatically generated

1. Skip unix\_socket authentication and change root password (answer n for both)

Text

Description automatically generated

1. Remove anonymous users, disallow remote access for root, delete the test database, and reload the privileges. (answer y for all 4)

Text

Description automatically generated

1. Open mysql under root user and the password used by ubuntu (not the password during setup)
2. Setup a user for the backend application to connect to the database on

*\*Replace \*\*username\*\* and \*\*password\*\* with whatever you like, these will be used in the backend application. Default is ‘Program’ and ‘progpass’ for username and password*



1. Open the port the web API listens on to allow the Raspberry Pi and Phone Application to communicate with the web API. This will also enable the firewall



1. To use the phone application outside your home network, forward port 6000 (TCP) in your router for the servers IP. You should also reserve the IP for the server MAC address, so it will not change if the router resets from a power outage. Please refer to online guides from your routers manufacturer on how to do this.

\*\*Add setting up Apache server

Running the Backend Applications

1. Download the WebAPI and backend helper application from GitHub.
   1. Web API – <https://github.com/ASU-IoT-ResearchProject/IotBackendAPI/releases/tag/v22w45d7a>
   2. Backend Helper – <https://github.com/ASU-IoT-ResearchProject/IoTBackendHelper/releases/tag/v22w45d7a>
2. Unzip each package into its own folder (if its zipped)
3. Open each folder in its own terminal
4. To start the web API run this command



1. To start the backend helper app run this command (This app fetched geolocation data and port information for all ip records)

