

Mot Mot

Tracker

by

Gavin Prue, Mathew Praise, Michael Lendvai

Function Requirements

MotMot is a Raspberry Pi based tracking device that collects GPS data via an Adafruit GPS Module that then sends the GPS data to a server in order for a user to view it in real-time via a password protected GUI web browser. MotMot parses NMEA (National Marine Electronics Association) standard sentences into plottable data which can then be used to visually represent the physical location of the device using a Google Maps API and ultimately allowing the user to essentially have a mobile remote-accessible, trackable computer. As previously mentioned MotMot requires the user to enter a username and password to login to the web browser. Its primary usage is to allow the user to track vehicles in a discrete manner. However, due to MotMot being essentially a mobile remote-accessible, trackable computer it is perfect for potentially creating a Rogue Access point or initiating other Wi-Fi or Bluetooth based attacks. MotMot provides an effective low-cost tracking solution in comparison to other commercially available tracking systems.

**Functional Requirements:**

* The software needs to enable the user to track the device through a GUI interface.
* The software needs to allow a user to input a time and see where the device was at that time.
* The device needs to function independently on battery power for a minimum of 24 hours.
* The device needs to be accessible through SSH and allow the user to manage the device.
* The device needs to be continuously connected to the internet.
* The software needs to enable the user to set a password to access the tracking data.
* The software needs to use the google maps API to effectively track the device and map it to a geographical representation.
* The GUI interface has to show data coming from the sensor in real time.
* The software needs to parse NMEA (National Marine Electronics Association) sentences in to usable data (I.e for google maps API to plot using longitude and latitude).
* Captures GPS data offline and uploads it when there is available internet connection.

**Non-Functional Requirements:**

* The passwords to all components should be strong and secure
* The final product has to be easy to use.
* Easy to troubleshoot for end user.
* Should work with limited human interaction.
* The Tracker needs to be in a portable secure enclosure that prevents it from damage.
* GUI is compatible with a range of devices, mobile, tablets etc.

**Tasks:**

* Put together the hardware.
  + Solder the legs onto the GPS module. Connect the module to the Pi.
* Design an optimal box that fits the raspberry pi, powerbank, GPS sensor.
* Create a startup script that runs our program whenever the Pi turns on.
  + Start the GPS module deamon(gpsd)
  + Checkup hardware to ensure the device can function correctly.
* Networking: Set up the Pi so that it has access to the internet and we can access it remotely.
  + Hotspot? 3G/4G dongle
* Set up a web server for the frontend.
  + DigitalOcean?
* Make sure the Pi and the server can transfer data between each other.
* Make the data coming from the sensor usable.
  + Parse the NMEA sentences into usable data so that the google API can map the coordinates.
* Make the website/gui:
  + Input fields for password, date and other features.
  + Output time/date, number of fixs (satellites), current position in long and lat, satellite information (I.e number)

**List of Contribution:**

**Michael**

* Set up GitHub repository for the project
* Initial python2 support for the parser
* Multi-line input processing support for the parser
* DMS to decimal converter support for the parser
* Added crash fixes to the parser
* Added Base64 encryption to the SSH transfer function.

**Gavin**

* Part procurement (Adafruit GPS module, Pi, Battery Pack and ancillaries) and assembly.
* Initial parser code and subsequent project evolutional changes to the parser.
* Functionality to remove duplicates from output file.
* Functionality to sort lines by timestamp.
* Functionality to transfer output from parser to frontend via SCP and SSH.(scp & paramiko modules)
* Script to disable gpsd daemon on the pi.
* Installation of ‘scp’ and ‘paramiko’ on the pi.
* Project happiness, discipline and motivator coordinator.

**Mathew**

* Setup web server on digitalocean (209.97.140.84) and LAMP server
* Front-end web development:
  + PHP script to read gps coordinates, timestamps, date
  + Responsive design and implementation of Google’s material design
  + Live tracking (markers plotted from latlng.txt file)
  + Password protection for front-end

Special thanks to **Joe Smith** for soldering the Pins on the Adafruit GPS Module.