AN-025 Node Red Traffic Monitor Dashboard

The WiFi enabled OPS243 and OPS7243 sensors make it much easier to send data to the cloud and observe the data remotely. By using Node Red, a simple traffic monitoring dashboard can be implemented taking in OPS243/OPS7243 data and providing a simple way to see roadway speeds, vehicle counts, and traffic patterns. This application note explains the details of a Node Red flow which displays traffic data on a simple dashboard using a live sensor at the OmniPreSense offices.

Node Red

Node Red is a simple development tool for visual programming (Figure *1*) by wiring together hardware, APIs, and online services for Internet of Things (IoT) applications. A browser-based flow editor is used and utilizes JavaScript functions. It was originally developed by IBM and released as an open-source project in 2016. Node Red utilizes Node.js and flows created are stored in JSON format.

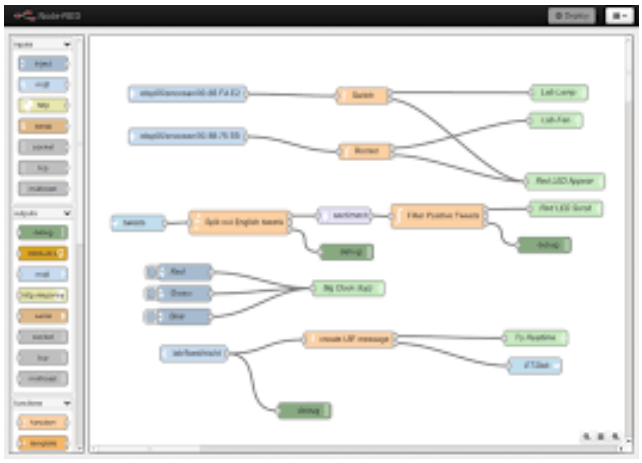

Figure 1. Node Red Development Editor

Node Red comes pre-installed on many types of hardware including Raspberry Pi's, BeagleBone, and other embedded processor boards. Several cloud services also support Node Red including AWS, IBM Cloud, and Azure.

This application note will not go over the details of installing a Node Red server or configuring your system to support Node Red but will instead focus on the flow for a traffic monitor. The example hardware is an OPS243-A connected to a Raspberry PI at the OmniPreSense offices. Remotely mounted WiFi enabled OPS243 or OPS7243 radar sensor. To install or run Node Red on your hardware or a server, check out these resources:

- NodeRed.org
- Steve's Node Red Guide

Link to download Node Red: https://nodejs.org/en/

Traffic Monitor Data Flow Architecture

The sensor utilized for traffic monitoring in this application note is an OPS243-A connected to a Raspberry Pi via USB. The Raspberry Pi is running a Node Red server and the traffic monitoring dashboard discussed below. The dashboard is made available for viewing here.

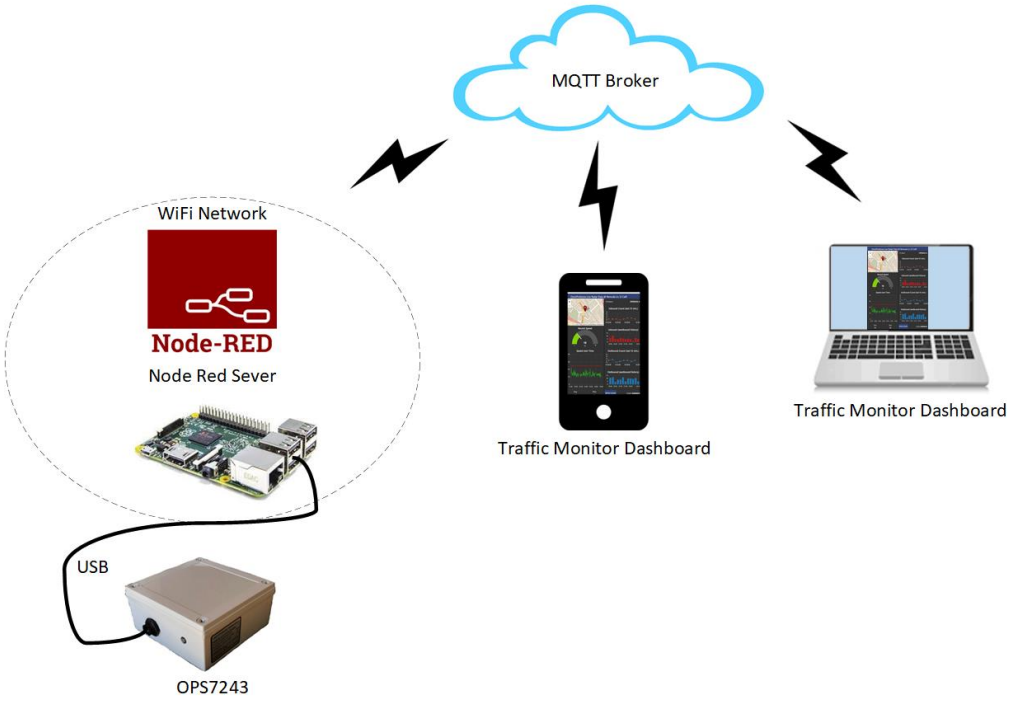

Figure 2. Traffic Monitor Dashboard Architecture

An alternate configuration can eliminate the Raspberry Pi and send the data to the cloud for receipt at a remote Node Red server and made available to view anywhere in the world. In this architecture, the Node Red traffic monitor dashboard is an MQTT subscriber and receives the live data from the sensor. The data is output in JSON format and within Node Red is parsed to separate out the direction, speed information, etc. The wireless data flow is shown in Figure 3.

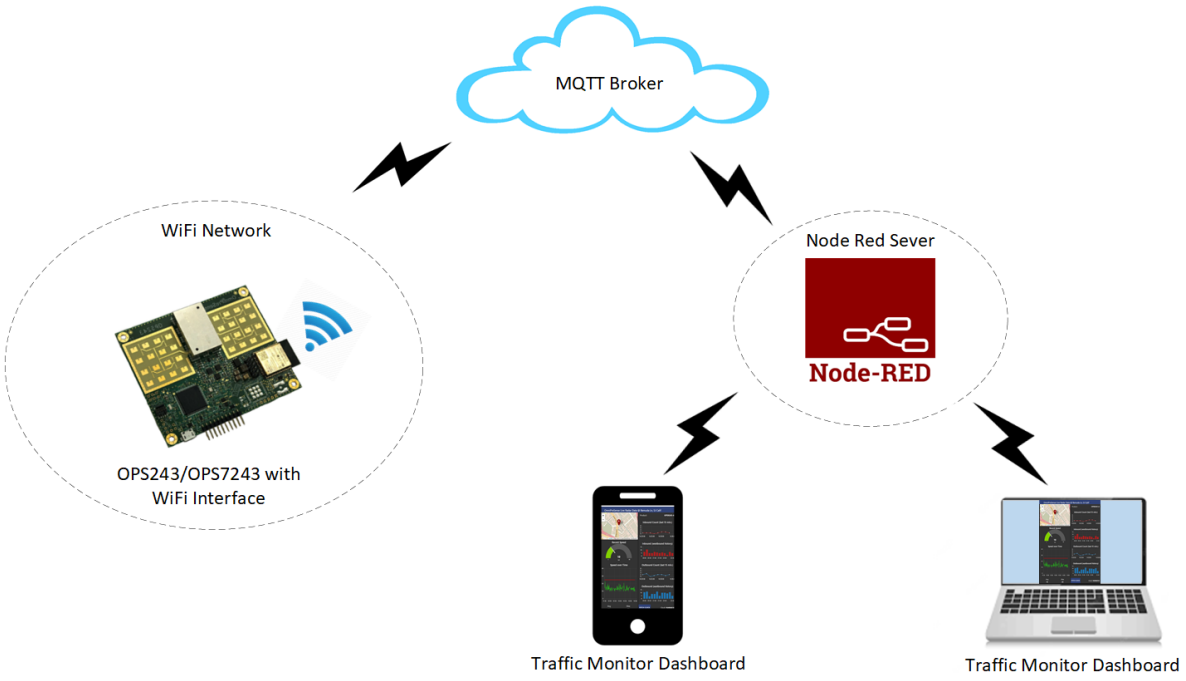

Figure 3. WiFi Enabled Traffic Monitor Data to Cloud Architecture

Node Red Traffic Monitor Flow

The details of a traffic monitor dashboard and its Node Red flow is described below. The complete flow is listed in Appendix A and details of particular nodes are described in detail including their configurations.

The Node Red dashboard is shown in Figure 4 and is available to view live here. The dashboard includes a map with the location of the sensor and a live speedometer that will trigger whenever a vehicle is detected. In addition to this, several historical graphs are displayed which show the speed of time and the counts of vehicles in either direction over the prior 15 minutes or last 3 hours. The average and maximum speed detected over the last 3 hours is also listed.

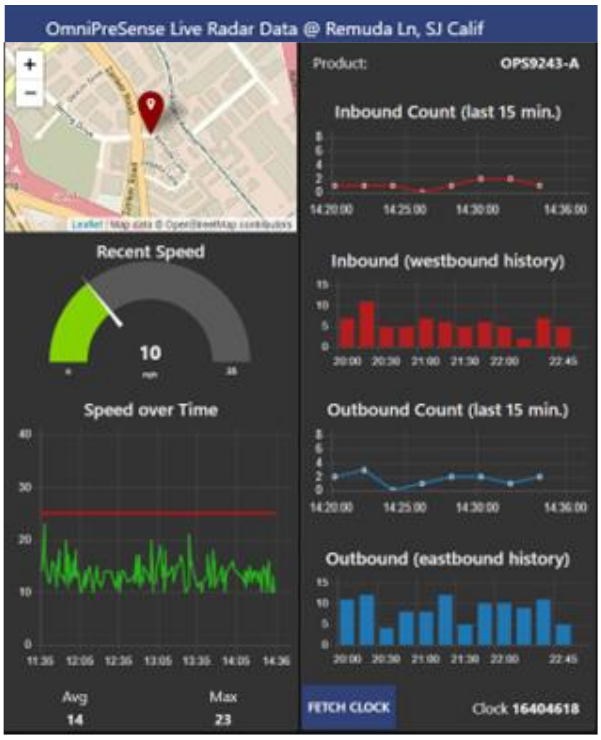

Figure 4. Node Red Traffic Flow Monitor

The location of the sensor is represented by a map (Figure 5) using OpenStreetMap and a marker is placed where the sensor is located. When a vehicle passes by, an alert is posted for 5 seconds showing the speed and direction of the vehicle.

Figure 5. Location Map with Marker and Detected Speed

The location is set in the Node Red flow by listing the longitude and latitude of the sensor. The flow node is shown in Figure 6. Double clicking on the worldmap node provides the configuration window to the right. In this window, the size of the map is set (6x4), longitude/latitude provided, map source information selected (OpenStreetMap), and other viewing settings are configured. The map is located in the Group labeled Location which places it on the left side of the overall dashboard.

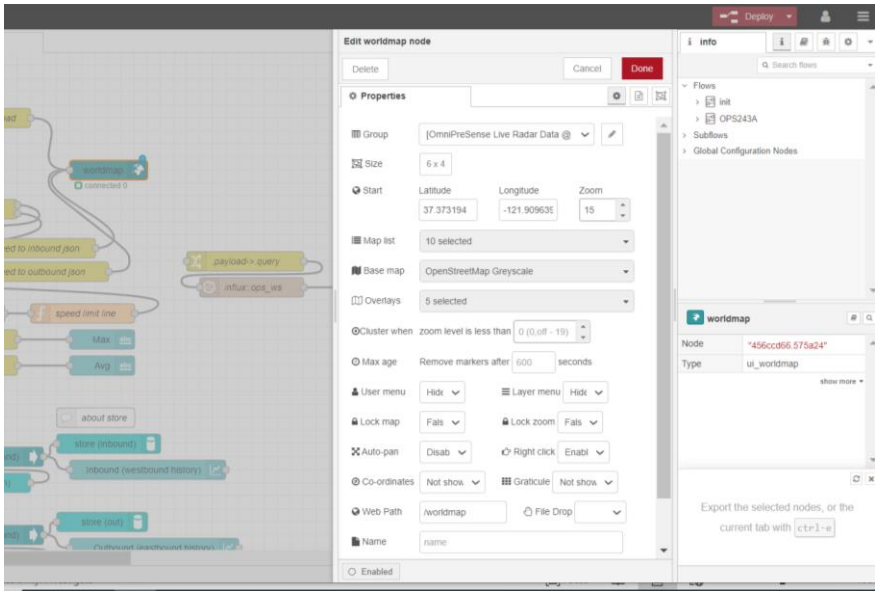

Figure 6. Map Node Configuration Settings

Below the map shows the speedometer (Figure 7) which will move left or right to indicate the speed of a recently detected vehicle. The speedometer will change colors as the speed increases to the maximum listing of 35mph which is configurable.

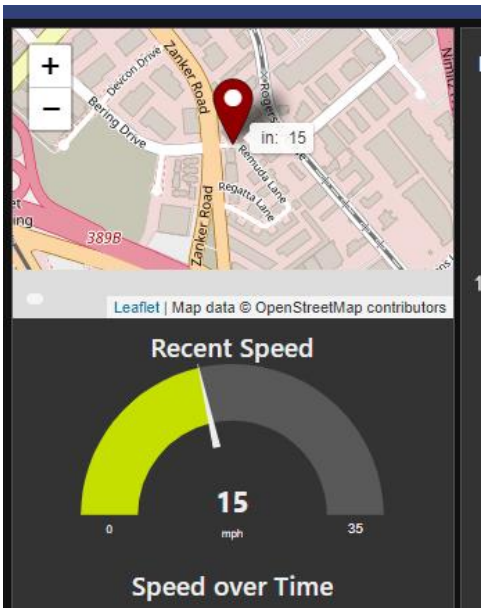

Figure 7. Speedometer

Double clicking on the speedguage node provides the configuration information (Figure 8). The size is set to 6x3 and gauge setting is selected. The label “Recent Speed” is set and maximum speed and colors to display are also selected.

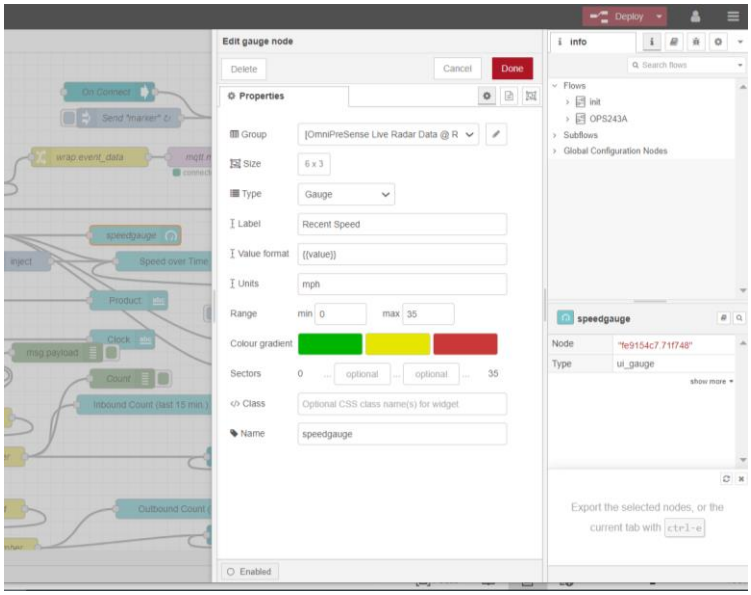

Figure 8. Speedometer Node Configuration

The data represented by the speedometer flows from the sensor in the following manner. From the sensor through the USB interface to the Raspberry Pi and Node Red. The serial in node is utilized to read the data from the USB interface and runs it through a JSON node followed by a switch node to look for data labeled “DetectedObjectVelocity”. Data such labeled is pulled out of the JSON string and changed to a “speed” payload with a switch node. At this state, the speed is either a positive (inbound) or negative (outbound) speed value. Another switch node changes the speed to an absolute value which is then provided to the speedometer gauge node. The general data flow is shown in Figure 9.

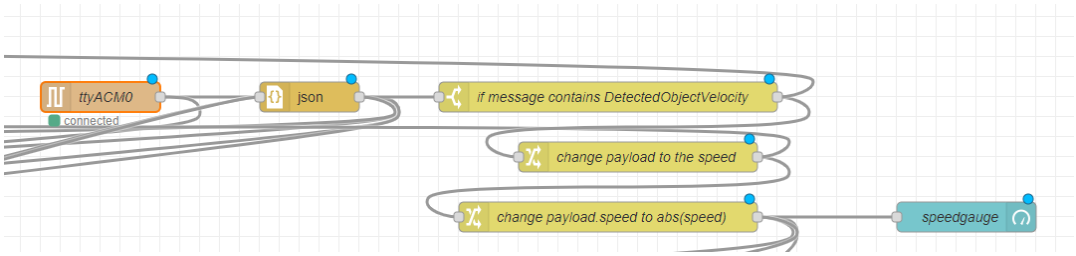

Figure 9. Data Flow thru Nodes to Speedometer

Below the speedometer is the historical speeds versus the road speed limit (red line, 25mph). Each time a vehicle is detected, its speed is noted, and the green line moves to that speed. This provides an easy way to visualize how bad a speeding problem may exist. The time period is set for the last 3 hours.

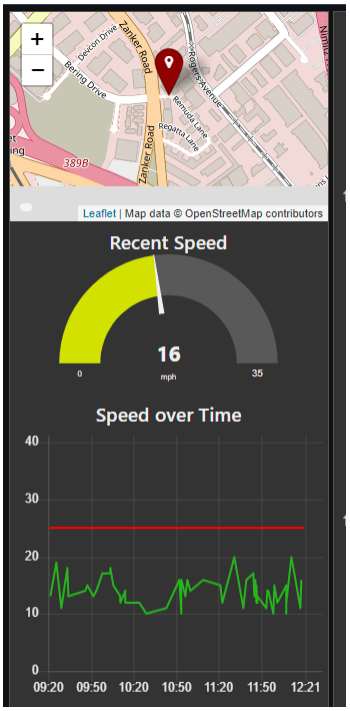

Figure 10. Speed over Time Graph

The speed over time graph configuration is shown in Figure 11. Like the other charts, Group is set, size, and label. The time period is set to display data over the last 3 hours and the x-axis is set to hours:minutes. Green is selected for the main speed line and the second color red is selected for the speed limit line of 25mph. It is possible to hover a mouse over the green line to see the speeds reported at any time.

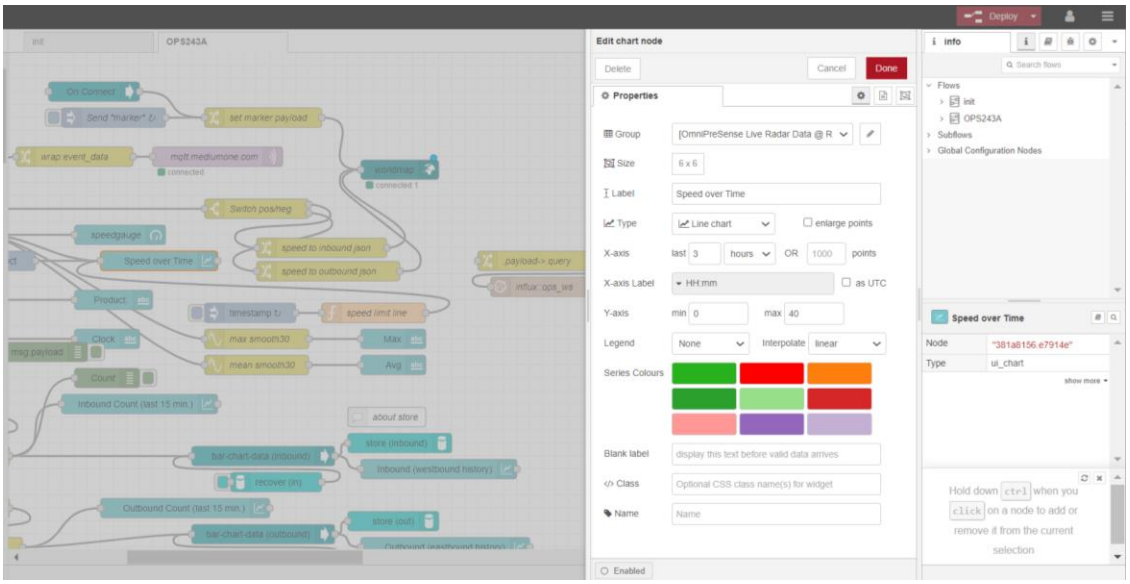

Figure 11. Speed over Time Chart Configuration

The speed over time graph uses several nodes to generate the data displayed (Figure 12). The main data to generate the green line is the same as the speedometer, the absolute speed value reported. Additional

nodes are configured to represent the red speed limit line. A timestamp node is configured to trigger every 15 seconds. This trigger is fed into a function node which generates the fixed red line at 25mph.

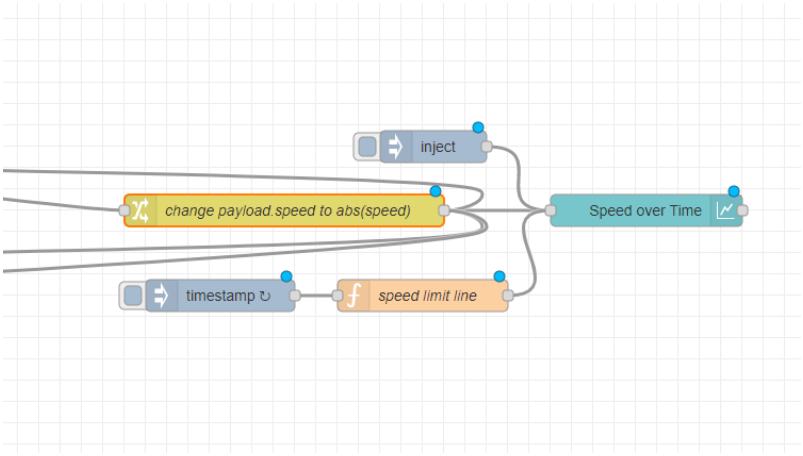

Figure 12. Speed over Time Nodes

Finally, the average and maximum speeds detected over the prior 3 hours are provided below the speed over time graph. This completes the data in the Location group.

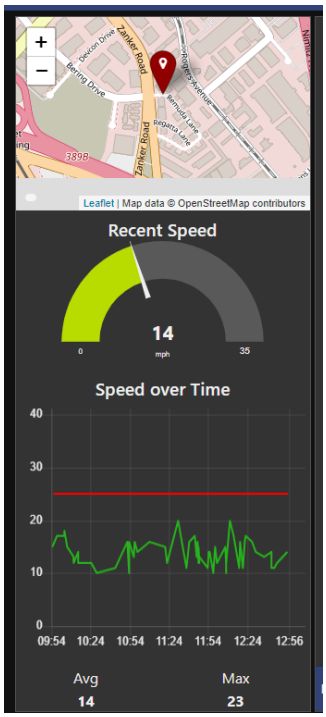

Figure 13. Average and Maximum Speed Reporting

To report the average and maximum speeds, the absolute data node is run through a smooth node to capture the data over the last 500 data points reported (Figure 14). Depending on traffic levels, this may be more or less than the 3 hours set for the speed overtime graph. The Action is set as either the maximum or average returned.

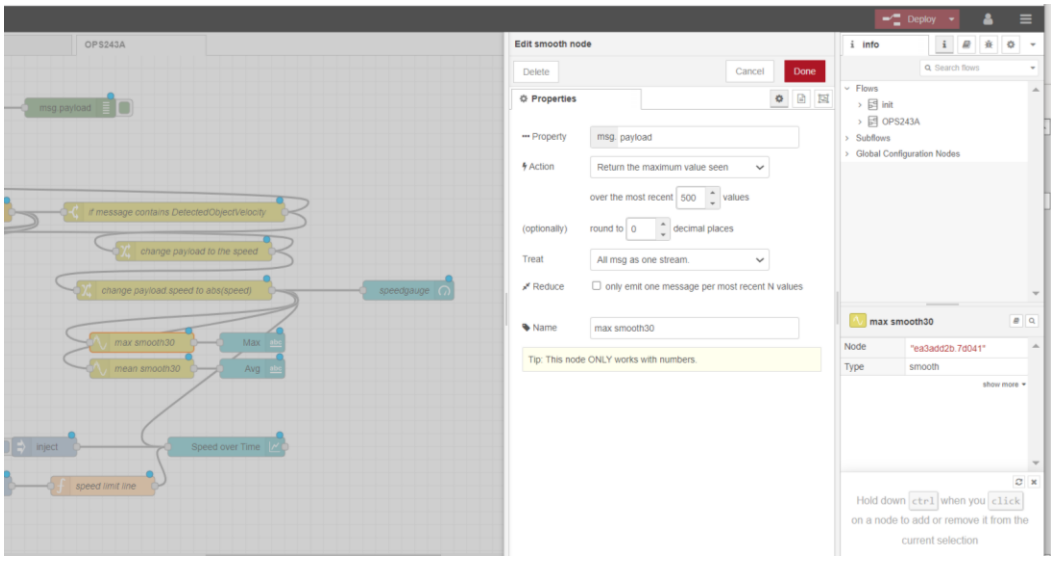

Figure 14. Average and Maximum Speed Nodes

The second column of charts (Figure 15) on the right is a Group labeled “Dashboard” compared to “Location” used for the first column. The top most data tells what radar sensor is being used. In this case, it’s an OPS9243-A which uses the OPS243-A radar sensor. The product reported is the equivalent of the ?P command in the API. The product information is filtered out of the JSON messages using a switch node and looks for the keyword “Product” and outputs the resulting payload when this is found to a text node (Figure 16).

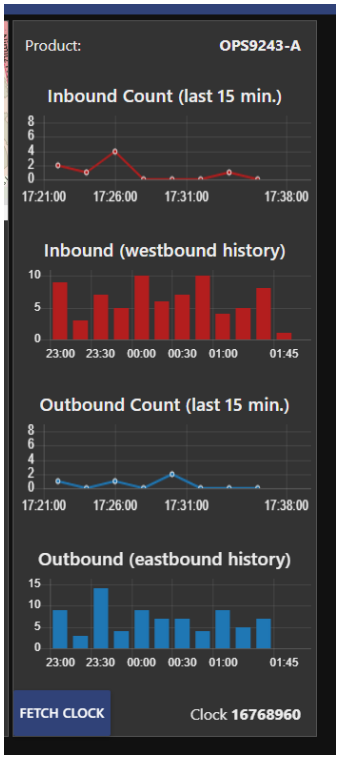

Figure 15. Dashboard Column of Charts

Figure 16. Product Report Nodes

Below the product label is the inbound traffic count over the last 15 minutes. The chart will update every minute with the count for that minute.

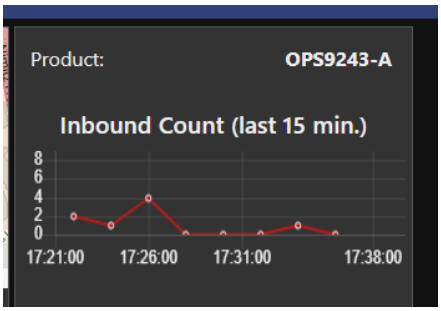

Figure 17. Direction Count - 15 Minutes

To reproduce this data, the JSON message is again parsed looking for the "TimedSpeedCounts" message with the switch node (Figure 18). If detected, another switch node looks for the direction, in this case "Inbound". When an inbound speed count is detected a change node is used to convert the payload to a number which is used for the next two charts. One chart provides the count over the last 15 minutes while the other chart provides counts over the last three hours. A bar-chart-data node is configured to provide the counts over 3 hours before sending it on to the chart node. A duplicate path is used for the "Outbound" data.

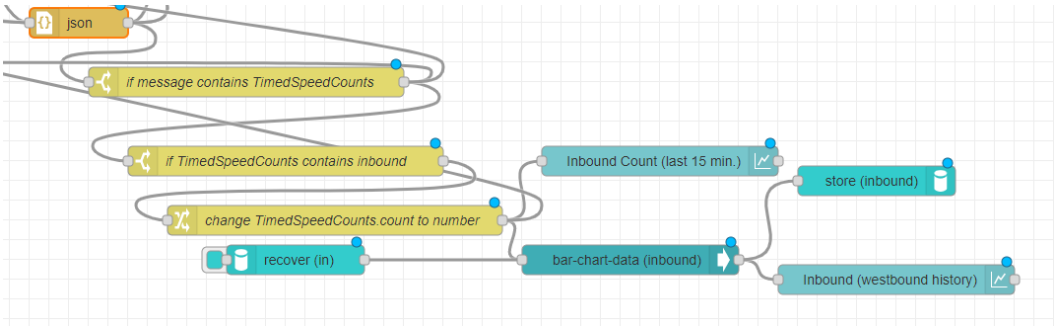

Figure 18. Inbound Count over Time Nodes

The final part of the Dashboard column is a button to fetch the current clock time of the sensor (bottom Figure 15). This sends the equivalent of the ?C command back to the sensor and the clock value is returned to Node Red. The JSON message is again checked for the keyword Clock and if it's present, a switch node sends the data to the clock text node (Figure 19).

Figure 19. Clock Data Nodes

Remote Data Monitoring Using MQTT

If a WiFi enabled OPS243/OPS7243 sensor is utilized, the data from the sensor can be sent to remote Node Red server using MQTT (Figure 3). A communication node allows the data to be received from the sensor and display on the dashboard as described above. The MQTT node is configured with the MQTT network name, MQTT username, password, and topic. This information is available from OmniPreSense customer service by emailing them at customerservice@omnipresense.com.

Traffic Monitor Node Red Flow

The Node Red flow described above is provided as a reference here. The flow can be imported into a Node Red dashboard and easily configured for your sensor. The complete flow used is provided in the Appendix A which follows.

Appendix A – Node Red Traffic Monitor Flow


```
[
  {
    "id": "b7a05959.ff8688",
    "type": "tab",
    "label": "OPS24--A",
    "disabled": false,
    "info": ""
  },
  {
    "id": "d9c88b99.c91ee8",
    "type": "ui_base",
    "theme": {
      "name": "theme-dark",
      "lightTheme": {
        "default": "#0094CE",
        "baseColor": "#0094CE",
        "baseFont": "-apple-system,BlinkMacSystemFont,Segoe UI,Roboto,Oxygen-
Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif",
        "edited": true,
        "reset": false
      },
      "darkTheme": {
        "default": "#097479",
        "baseColor": "#097479",
        "baseFont": "-apple-system,BlinkMacSystemFont,Segoe UI,Roboto,Oxygen-
Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif",
        "edited": true,
        "reset": false
      },
      "customTheme": {
        "name": "Untitled Theme 1",
        "default": "#4B7930",
        "baseColor": "#4B7930",
        "baseFont": "-apple-system,BlinkMacSystemFont,Segoe UI,Roboto,Oxygen-
Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif"
      },
      "themeState": {
        "base-color": {
          "default": "#097479",
          "value": "#097479",
          "edited": false
```

```
        },
        "page-titlebar-backgroundColor": {
          "value": "#097479",
          "edited": false
        },
        "page-backgroundColor": {
          "value": "#111111",
          "edited": false
        },
        "page-sidebar-backgroundColor": {
          "value": "#333333",
          "edited": false
        },
        "group-textColor": {
          "value": "#0eb8c0",
          "edited": false
        },
        "group-borderColor": {
          "value": "#555555",
          "edited": false
        },
        "group-backgroundColor": {
          "value": "#333333",
          "edited": false
        },
        "widget-textColor": {
          "value": "#eeeeee",
          "edited": false
        },
        "widget-backgroundColor": {
          "value": "#097479",
          "edited": false
        },
        "widget-borderColor": {
          "value": "#333333",
          "edited": false
        },
        "base-font": {
          "value": "-apple-system,BlinkMacSystemFont,Segoe UI,Roboto,Oxygen-
Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif"
        }
      },
      "angularTheme": {
        "primary": "indigo",
```

```
            "accents": "blue",
            "warn": "red",
            "background": "grey"
        }
    },
    "site": {
        "name": "Node-RED Dashboard",
        "hideToolbar": "false",
        "allowSwipe": "false",
        "lockMenu": "false",
        "allowTempTheme": "true",
        "dateFormat": "DD/MM/YYYY",
        "sizes": {
            "sx": 56,
            "sy": 56,
            "gx": 6,
            "gy": 6,
            "cx": 6,
            "cy": 6,
            "px": 0,
            "py": 0
        }
    }
},
{
    "id": "767b649e.25892c",
    "type": "ui_group",
    "name": "Dashboard",
    "tab": "ddd6dfb3.36535",
    "order": 2,
    "disp": false,
    "width": "6",
    "collapse": false,
    "className": ""
},
{
    "id": "2ff065ae.0ccc6a",
    "type": "ui_group",
    "name": "Location",
    "tab": "ddd6dfb3.36535",
    "order": 1,
    "disp": false,
    "width": "6",
    "collapse": false
```

```
},
{
    "id": "ddd6dfb3.36535",
    "type": "ui_tab",
    "name": "OmniPreSense Live Radar Data @ Remuda Ln, SJ Calif (PST)",
    "icon": "dashboard",
    "disabled": false,
    "hidden": false
},
{
    "id": "8f1523f3.fc6e",
    "type": "serial-port",
    "serialport": "/dev/ttyACM0",
    "serialbaud": "19200",
    "databits": "8",
    "parity": "none",
    "stopbits": "1",
    "waitfor": "",
    "newline": "\\n",
    "bin": "false",
    "out": "char",
    "addchar": "",
    "responsetimeout": "10000"
},
{
    "id": "2293c112.ebf75e",
    "type": "inject",
    "z": "b7a05959.ff8688",
    "name": "init: switch to JSON, MPH, no fractions",
    "props": [
        {
            "p": "payload"
        },
        {
            "p": "topic",
            "vt": "str"
        }
    ],
    "repeat": "",
    "crontab": "",
    "once": true,
    "onceDelay": "",
    "topic": "",
    "payload": "\"ONOJF0USS2A!?P@O\" & \"@|60\\n\" & \"A!\"",
```

```
    "payloadType": "jsonata",
    "x": 190,
    "y": 60,
    "wires": [
      [
        "79268f5e.2d085"
      ]
    ]
  },
  {
    "id": "b5989a25.0e77b8",
    "type": "json",
    "z": "b7a05959.ff8688",
    "name": "",
    "property": "payload",
    "action": "",
    "pretty": false,
    "x": 110,
    "y": 300,
    "wires": [
      [
        "626d69eb.6a8578",
        "af0cf345.b0af",
        "42adeaf2.9debc4",
        "fe6dd7cb.c77198"
      ]
    ]
  },
  {
    "id": "626d69eb.6a8578",
    "type": "switch",
    "z": "b7a05959.ff8688",
    "name": "if message contains Product",
    "property": "payload",
    "propertyType": "msg",
    "rules": [
      {
        "t": "hask",
        "v": "Product",
        "vt": "str"
      }
    ],
    "checkall": "true",
    "repair": false,
```

```
    "outputs": 1,
    "x": 340,
    "y": 380,
    "wires": [
      [
        "6d8449c8.b1a728"
      ]
    ]
  },
  {
    "id": "f0ad17d7.dd4c98",
    "type": "inject",
    "z": "b7a05959.ff8688",
    "name": "send Product Query",
    "repeat": "",
    "crontab": "",
    "once": false,
    "onceDelay": "",
    "topic": "",
    "payload": "?P",
    "payloadType": "str",
    "x": 130,
    "y": 140,
    "wires": [
      [
        "79268f5e.2d085"
      ]
    ]
  },
  {
    "id": "6d8449c8.b1a728",
    "type": "ui_text",
    "z": "b7a05959.ff8688",
    "group": "767b649e.25892c",
    "order": 1,
    "width": "6",
    "height": "1",
    "name": "",
    "label": "Product:",
    "format": "{{msg.payload.Product}}",
    "layout": "row-spread",
    "x": 760,
    "y": 380,
    "wires": []
```

```
},
{
    "id": "fe9154c7.71f748",
    "type": "ui_gauge",
    "z": "b7a05959.ff8688",
    "name": "speedgauge",
    "group": "2ff065ae.0ccc6a",
    "order": 2,
    "width": "6",
    "height": "3",
    "gtype": "gage",
    "title": "Recent Speed",
    "label": "mph",
    "format": "{{value}}",
    "min": "0",
    "max": "35",
    "colors": [
        "#00b500",
        "#e6e600",
        "#ca3838"
    ],
    "seg1": "",
    "seg2": "",
    "x": 770,
    "y": 280,
    "wires": []
},
{
    "id": "381a8156.e7914e",
    "type": "ui_chart",
    "z": "b7a05959.ff8688",
    "name": "",
    "group": "2ff065ae.0ccc6a",
    "order": 3,
    "width": "6",
    "height": "4",
    "label": "Speed over Time",
    "chartType": "line",
    "legend": "false",
    "xformat": "HH:mm",
    "interpolate": "linear",
    "nodata": "",
    "dot": false,
    "ymin": "0",
```

```
    "ymax": "40",
    "removeOlder": "3",
    "removeOlderPoints": "",
    "removeOlderUnit": "3600",
    "cutout": 0,
    "useOneColor": false,
    "useUTC": false,
    "colors": [
      "#28b31e",
      "#ff0000",
      "#ff7f0e",
      "#2ca02c",
      "#98df8a",
      "#d62728",
      "#ff9896",
      "#9467bd",
      "#c5b0d5"
    ],
    "outputs": 1,
    "useDifferentColor": false,
    "className": "",
    "x": 790,
    "y": 320,
    "wires": [
      []
    ]
  },
  {
    "id": "357441e8.ca7a2e",
    "type": "ui_button",
    "z": "b7a05959.ff8688",
    "name": "",
    "group": "767b649e.25892c",
    "order": 6,
    "width": 2,
    "height": 1,
    "passthru": false,
    "label": "Fetch clock",
    "tooltip": "",
    "color": "",
    "bgcolor": "",
    "icon": "",
    "payload": "C?",
    "payloadType": "str",
```

```
    "topic": "",
    "x": 90,
    "y": 180,
    "wires": [
      [
        "79268f5e.2d085"
      ]
    ]
  },
  {
    "id": "bc8da3d.2730e6",
    "type": "ui_text",
    "z": "b7a05959.ff8688",
    "group": "767b649e.25892c",
    "order": 7,
    "width": "4",
    "height": 1,
    "name": "",
    "label": "Clock",
    "format": "{{msg.payload.Clock}}",
    "layout": "row-right",
    "x": 750,
    "y": 440,
    "wires": []
  },
  {
    "id": "af0cf345.b0af",
    "type": "switch",
    "z": "b7a05959.ff8688",
    "name": "if message contains Clock",
    "property": "payload",
    "propertyType": "msg",
    "rules": [
      {
        "t": "hask",
        "v": "Clock",
        "vt": "str"
      }
    ],
    "checkall": "true",
    "repair": false,
    "outputs": 1,
    "x": 330,
    "y": 440,
```

```
    "wires": [
      [
        "bc8da3d.2730e6"
      ]
    ]
  },
  {
    "id": "95514863.951558",
    "type": "change",
    "z": "b7a05959.ff8688",
    "name": "change payload.speed to abs(speed)",
    "rules": [
      {
        "t": "set",
        "p": "payload",
        "pt": "msg",
        "to": "$abs(payload)",
        "tot": "jsonata"
      }
    ],
    "action": "",
    "property": "",
    "from": "",
    "to": "",
    "reg": false,
    "x": 410,
    "y": 280,
    "wires": [
      [
        "381a8156.e7914e",
        "fe9154c7.71f748",
        "ea3add2b.7d041",
        "6bba66d1.0e6788"
      ]
    ]
  },
  {
    "id": "42adeaf2.9debc4",
    "type": "switch",
    "z": "b7a05959.ff8688",
    "name": "if message contains TimedSpeedCounts",
    "property": "payload",
    "propertyType": "msg",
    "rules": [
```

```
      {
        "t": "hask",
        "v": "TimedSpeedCounts",
        "vt": "str"
      }
    ],
    "checkall": "true",
    "repair": false,
    "outputs": 1,
    "x": 380,
    "y": 480,
    "wires": [
      [
        "a0102cc6.bed61",
        "38a35462.571bfc"
      ]
    ]
  },
  {
    "id": "8192d4ad.5aa8c8",
    "type": "change",
    "z": "b7a05959.ff8688",
    "name": "change TimedSpeedCounts.count to number",
    "rules": [
      {
        "t": "set",
        "p": "payload",
        "pt": "msg",
        "to": "$number(payload.TimedSpeedCounts.count)\t",
        "tot": "jsonata"
      }
    ],
    "action": "",
    "property": "",
    "from": "",
    "to": "",
    "reg": false,
    "x": 430,
    "y": 600,
    "wires": [
      [
        "a874a314.44228",
        "7535ab88.e47ae4"
      ]
```

```
    ]
  },
  {
    "id": "a0102cc6.bed61",
    "type": "switch",
    "z": "b7a05959.ff8688",
    "name": "if TimedSpeedCounts contains inbound",
    "property": "payload.TimedSpeedCounts.direction",
    "propertyType": "msg",
    "rules": [
      {
        "t": "cont",
        "v": "inbound",
        "vt": "str"
      }
    ],
    "checkall": "true",
    "repair": false,
    "outputs": 1,
    "x": 420,
    "y": 560,
    "wires": [
      [
        "8192d4ad.5aa8c8"
      ]
    ]
  },
  {
    "id": "a874a314.44228",
    "type": "ui_chart",
    "z": "b7a05959.ff8688",
    "name": "",
    "group": "767b649e.25892c",
    "order": 2,
    "width": "6",
    "height": "3",
    "label": "Inbound Count (last 15 min.)",
    "chartType": "line",
    "legend": "false",
    "xformat": "HH:mm:ss",
    "interpolate": "linear",
    "nodata": "pending new data",
    "dot": true,
    "ymin": "0",
```

```
    "ymax": "4",
    "removeOlder": "15",
    "removeOlderPoints": "",
    "removeOlderUnit": "60",
    "cutout": 0,
    "useOneColor": false,
    "colors": [
      "#b31e1e",
      "#aec7e8",
      "#ff7f0e",
      "#2ca02c",
      "#98df8a",
      "#d62728",
      "#ff9896",
      "#9467bd",
      "#c5b0d5"
    ],
    "outputs": 1,
    "x": 820,
    "y": 560,
    "wires": [
      []
    ]
  },
  {
    "id": "38a35462.571bfc",
    "type": "switch",
    "z": "b7a05959.ff8688",
    "name": "if TimedSpeedCounts contains outbound",
    "property": "payload.TimedSpeedCounts.direction",
    "propertyType": "msg",
    "rules": [
      {
        "t": "cont",
        "v": "outbound",
        "vt": "str"
      }
    ],
    "checkall": "true",
    "repair": false,
    "outputs": 1,
    "x": 420,
    "y": 680,
    "wires": [
```

```
      [
        "4614d19e.bd756"
      ]
    ]
  },
  {
    "id": "b1b60443.11c098",
    "type": "ui_chart",
    "z": "b7a05959.ff8688",
    "name": "",
    "group": "767b649e.25892c",
    "order": 4,
    "width": "6",
    "height": "3",
    "label": "Outbound Count (last 15 min.)",
    "chartType": "line",
    "legend": "false",
    "xformat": "HH:mm:ss",
    "interpolate": "linear",
    "nodata": "pending new data",
    "dot": true,
    "ymin": "0",
    "ymax": "4",
    "removeOlder": "15",
    "removeOlderPoints": "",
    "removeOlderUnit": "60",
    "cutout": 0,
    "useOneColor": false,
    "colors": [
      "#1f77b4",
      "#aec7e8",
      "#ff7f0e",
      "#2ca02c",
      "#98df8a",
      "#d62728",
      "#ff9896",
      "#9467bd",
      "#c5b0d5"
    ],
    "outputs": 1,
    "x": 830,
    "y": 680,
    "wires": [
      []
```

```
    ]
  },
  {
    "id": "4614d19e.bd756",
    "type": "change",
    "z": "b7a05959.ff8688",
    "name": "change TimedSpeedCounts.count to number",
    "rules": [
      {
        "t": "set",
        "p": "payload",
        "pt": "msg",
        "to": "$number(payload.TimedSpeedCounts.count)\t",
        "tot": "jsonata"
      }
    ],
    "action": "",
    "property": "",
    "from": "",
    "to": "",
    "reg": false,
    "x": 430,
    "y": 720,
    "wires": [
      [
        "b1b60443.11c098",
        "2e6a9fd1.298f"
      ]
    ]
  },
  {
    "id": "960d772b.54c778",
    "type": "inject",
    "z": "b7a05959.ff8688",
    "name": "Send \"marker\"",
    "repeat": "60",
    "crontab": "",
    "once": true,
    "onceDelay": "2",
    "topic": "",
    "payload": "doesnt matter",
    "payloadType": "str",
    "x": 760,
    "y": 100,
```

```
    "wires": [
      [
        "f60210d6.d6451"
      ]
    ]
  },
  {
    "id": "8d8242be.39a08",
    "type": "change",
    "z": "b7a05959.ff8688",
    "name": "change payload to the speed",
    "rules": [
      {
        "t": "set",
        "p": "payload",
        "pt": "msg",
        "to": "$number(payload.DetectedObjectVelocity)",
        "tot": "jsonata"
      }
    ],
    "action": "",
    "property": "",
    "from": "",
    "to": "",
    "reg": false,
    "x": 380,
    "y": 240,
    "wires": [
      [
        "95514863.951558",
        "f669470b.176e18"
      ]
    ]
  },
  {
    "id": "fe6dd7cb.c77198",
    "type": "switch",
    "z": "b7a05959.ff8688",
    "name": "if message contains DetectedObjectVelocity",
    "property": "payload",
    "propertyType": "msg",
    "rules": [
      {
        "t": "hask",
```

```
            "v": "DetectedObjectVelocity",
            "vt": "str"
        }
    ],
    "checkall": "true",
    "repair": false,
    "outputs": 1,
    "x": 390,
    "y": 200,
    "wires": [
        [
            "8d8242be.39a08"
        ]
    ]
},
{
    "id": "f669470b.176e18",
    "type": "switch",
    "z": "b7a05959.ff8688",
    "name": "Switch pos/neg",
    "property": "payload",
    "propertyType": "msg",
    "rules": [
        {
            "t": "gt",
            "v": "0",
            "vt": "num"
        },
        {
            "t": "lt",
            "v": "0",
            "vt": "str"
        }
    ],
    "checkall": "true",
    "repair": false,
    "outputs": 2,
    "x": 980,
    "y": 240,
    "wires": [
        [
            "e297c7d5.440e88"
        ],
        [
```

```
            "538eec8f.8d3d34"
          ]
        ]
    },
    {
        "id": "e297c7d5.440e88",
        "type": "change",
        "z": "b7a05959.ff8688",
        "name": "speed to inbound json",
        "rules": [
          {
            "t": "set",
            "p": "speed",
            "pt": "msg",
            "to": "$join([ \" in: \", $formatNumber(msg.payload, '0')],' ')",
            "tot": "jsonata"
          },
          {
            "t": "set",
            "p": "payload",
            "pt": "msg",
            "to": "{\t   \"name\":\"inbound\",\t   \"lat\":$globalContext('lat'),\t
\"lon\":$globalContext('lon'),\t   \"label\":$.speed,\t   \"ttl\":10,\t   \"color\":\"red\"\t}",
            "tot": "jsonata"
          }
        ],
        "action": "",
        "property": "",
        "from": "",
        "to": "",
        "reg": false,
        "x": 1080,
        "y": 300,
        "wires": [
          [
            "456ccd66.575a24"
          ]
        ]
    },
    {
        "id": "538eec8f.8d3d34",
        "type": "change",
        "z": "b7a05959.ff8688",
        "name": "speed to outbound json",
```

```
      "rules": [
        {
          "t": "set",
          "p": "speed",
          "pt": "msg",
          "to": "$join([ \"out:\", $formatNumber($abs(msg.payload), '0')],' ')",
          "tot": "jsonata"
        },
        {
          "t": "set",
          "p": "payload",
          "pt": "msg",
          "to": "{\t   \"name\":\"outbound\",\t   \"lat\":$globalContext('lat'),\t
\"lon\":$globalContext('lon'),\t   \"label\":$.speed,\t   \"ttl\":10,\t   \"color\":\"blue\"}",
          "tot": "jsonata"
        }
      ],
      "action": "",
      "property": "",
      "from": "",
      "to": "",
      "reg": false,
      "x": 1090,
      "y": 336,
      "wires": [
        [
          "456ccd66.575a24"
        ]
      ]
    },
    {
      "id": "7d155aa1.84bcb4",
      "type": "ui_chart",
      "z": "b7a05959.ff8688",
      "name": "",
      "group": "767b649e.25892c",
      "order": 3,
      "width": "6",
      "height": "2",
      "label": "Inbound (westbound history)",
      "chartType": "bar",
      "legend": "false",
      "xformat": "HH:mm:ss",
      "interpolate": "linear",
```

```
    "nodata": "",
    "dot": false,
    "ymin": "",
    "ymax": "",
    "removeOlder": "12",
    "removeOlderPoints": "",
    "removeOlderUnit": "3600",
    "cutout": 0,
    "useOneColor": true,
    "colors": [
      "#b31e1e",
      "#aec7e8",
      "#ff7f0e",
      "#2ca02c",
      "#98df8a",
      "#d62728",
      "#ff9896",
      "#9467bd",
      "#c5b0d5"
    ],
    "outputs": 1,
    "x": 1100,
    "y": 600,
    "wires": [
      []
    ]
  },
  {
    "id": "c1c79f1a.ceba7",
    "type": "ui_chart",
    "z": "b7a05959.ff8688",
    "name": "",
    "group": "767b649e.25892c",
    "order": 5,
    "width": "6",
    "height": "2",
    "label": "Outbound (eastbound history)",
    "chartType": "bar",
    "legend": "false",
    "xformat": "HH:mm:ss",
    "interpolate": "linear",
    "nodata": "",
    "dot": false,
    "ymin": "0",
```

```
    "ymax": "",
    "removeOlder": 1,
    "removeOlderPoints": "",
    "removeOlderUnit": "3600",
    "cutout": 0,
    "useOneColor": true,
    "colors": [
      "#1f77b4",
      "#aec7e8",
      "#ff7f0e",
      "#2ca02c",
      "#98df8a",
      "#d62728",
      "#ff9896",
      "#9467bd",
      "#c5b0d5"
    ],
    "outputs": 1,
    "x": 1110,
    "y": 720,
    "wires": [
      []
    ]
  },
  {
    "id": "52c694e2.eae15c",
    "type": "ui_ui_control",
    "z": "b7a05959.ff8688",
    "name": "On Connect",
    "events": "connect",
    "x": 730,
    "y": 60,
    "wires": [
      [
        "f60210d6.d6451"
      ]
    ]
  },
  {
    "id": "8e6d3020.4d294",
    "type": "ui_text",
    "z": "b7a05959.ff8688",
    "group": "2ff065ae.0ccc6a",
    "order": 5,
```

```
    "width": "3",
    "height": "1",
    "name": "",
    "label": "Max",
    "format": "{{msg.payload}}",
    "layout": "col-center",
    "x": 1190,
    "y": 400,
    "wires": []
  },
  {
    "id": "f1ced828.422978",
    "type": "ui_text",
    "z": "b7a05959.ff8688",
    "group": "2ff065ae.0ccc6a",
    "order": 4,
    "width": "3",
    "height": "1",
    "name": "",
    "label": "Avg",
    "format": "{{msg.payload}}",
    "layout": "col-center",
    "x": 1190,
    "y": 440,
    "wires": []
  },
  {
    "id": "f60210d6.d6451",
    "type": "change",
    "z": "b7a05959.ff8688",
    "name": "set marker payload",
    "rules": [
      {
        "t": "set",
        "p": "payload",
        "pt": "msg",
        "to": "{\t  \"name\":\"marker\",\t  \"lat\":$globalContext('lat'),\t
\"lon\":$globalContext('lon'),\t  \"icon\":\"map-marker\",\t  \"color\":\"#dd0000\" \t}",
        "tot": "jsonata"
      }
    ],
    "action": "",
    "property": "",
    "from": "",
```

```
    "to": "",
    "reg": false,
    "x": 990,
    "y": 100,
    "wires": [
      [
        "456ccd66.575a24"
      ]
    ]
  },
  {
    "id": "2efdd184.4eea0e",
    "type": "serial in",
    "z": "b7a05959.ff8688",
    "name": "ttyACM0",
    "serial": "8f1523f3.fc6e",
    "x": 60,
    "y": 220,
    "wires": [
      [
        "b5989a25.0e77b8"
      ]
    ]
  },
  {
    "id": "79268f5e.2d085",
    "type": "serial out",
    "z": "b7a05959.ff8688",
    "name": "Send to OPS24--A",
    "serial": "8f1523f3.fc6e",
    "x": 470,
    "y": 140,
    "wires": []
  },
  {
    "id": "7535ab88.e47ae4",
    "type": "bar-chart-data",
    "z": "b7a05959.ff8688",
    "name": "bar-chart-data (inbound)",
    "x_interval": "quarter_hours",
    "x_size": "12",
    "unit": "",
    "precision": "1",
    "is_meter_reading": "False",
```

```
    "agg_by": "sum",
    "x": 810,
    "y": 600,
    "wires": [
      [
        "7d155aa1.84bcb4"
      ]
    ]
  },
  {
    "id": "2e6a9fd1.298f",
    "type": "bar-chart-data",
    "z": "b7a05959.ff8688",
    "name": "bar-chart-data (outbound)",
    "x_interval": "quarter_hours",
    "x_size": "12",
    "unit": "",
    "precision": "1",
    "is_meter_reading": "False",
    "agg_by": "sum",
    "x": 810,
    "y": 720,
    "wires": [
      [
        "c1c79f1a.ceba7"
      ]
    ]
  },
  {
    "id": "456ccd66.575a24",
    "type": "ui_worldmap",
    "z": "b7a05959.ff8688",
    "group": "2ff065ae.0ccc6a",
    "order": 1,
    "width": "6",
    "height": "4",
    "name": "",
    "lat": "37.373194",
    "lon": "-121.909639",
    "zoom": "15",
    "layer": "OSM",
    "cluster": "",
    "maxage": "",
    "usermenu": "hide",
```

```
    "layers": "hide",
    "panit": "false",
    "panlock": "false",
    "zoomlock": "false",
    "hiderightclick": "false",
    "coords": "none",
    "showgrid": "false",
    "path": "/worldmap",
    "x": 1200,
    "y": 180,
    "wires": []
  },
  {
    "id": "ea3add2b.7d041",
    "type": "smooth",
    "z": "b7a05959.ff8688",
    "name": "max smooth30",
    "property": "payload",
    "action": "max",
    "count": "500",
    "round": "0",
    "mult": "single",
    "reduce": false,
    "x": 980,
    "y": 400,
    "wires": [
      [
        "8e6d3020.4d294"
      ]
    ]
  },
  {
    "id": "6bba66d1.0e6788",
    "type": "smooth",
    "z": "b7a05959.ff8688",
    "name": "mean smooth30",
    "property": "payload",
    "action": "mean",
    "count": "500",
    "round": "0",
    "mult": "single",
    "reduce": false,
    "x": 980,
    "y": 440,
```

```
    "wires": [
      [
        "f1ced828.422978"
      ]
    ]
  },
  {
    "id": "de38f13857ef3138",
    "type": "inject",
    "z": "b7a05959.ff8688",
    "name": "",
    "props": [
      {
        "p": "payload"
      },
      {
        "p": "topic",
        "vt": "str"
      }
    ],
    "repeat": "15",
    "crontab": "",
    "once": false,
    "onceDelay": "15",
    "topic": "",
    "payload": "",
    "payloadType": "date",
    "x": 330,
    "y": 320,
    "wires": [
      [
        "8beb99311d6813b4"
      ]
    ]
  },
  {
    "id": "8beb99311d6813b4",
    "type": "function",
    "z": "b7a05959.ff8688",
    "name": "speed limit line",
    "func": "msg.payload = 25;\nmsg.topic = 'Line2';\nreturn msg;",
    "outputs": 1,
    "noerr": 0,
    "initialize": "",
```

```
    "finalize": "",
    "libs": [],
    "x": 500,
    "y": 320,
    "wires": [
      [
        "381a8156.e7914e"
      ]
    ]
  }
]
```

Revision History

Version	Date	Description
A	December 9, 2022	Initial release.
B	March 3, 2023	Updated Node Red Flow in Appendix to simplified version.