An Investigation into Router-Based Occupancy

George Dobric 101227367
Rosario Giardina 101188523

Tuesdays 2 PM - 6 PM
Professor: Paul O’Brien
EMNG2024 - The Internet of things (IoT) for Homes and Buildings

Table of Contents

Abstract

With the ever increasing prevalence of people management and HVAC efficiency maximization, and the growing demand of connectivity as it relates to the Internet of things, router-based occupancy may serve as the key to open a new door of opportunity as we approach the age of big data. Readily available smartphones can be used to obtain occupancy counts in trains, rooms, lobbies, and anywhere else where the management of HVAC and/or people is desired.

Apparatus

- Wireless Router using 802.11ac standard
- Node Red
- Smart Phone
- Fing App On Smart Phone
- InfluxDB
- Grafana

Method

The Fing app was used to export a JSON file, which contains the information of what devices are on the local network extracted from the router’s logs, to a gmail account which was created specifically for this purpose.
The router and devices connect following the Wi-Fi 5 protocol: 802.11ac. IEEE 802.11ac is a wireless networking standard in the 802.11 set of protocols (which is part of the Wi-Fi networking family), providing high-throughput wireless local area networks (WLANs) on the 5 GHz band. On a networking side, the way the device presence is obtained using ARP (address resolution protocol). Briefly, The Address Resolution Protocol is a communication protocol used for discovering the link layer address, such as a MAC address, associated with a given internet layer address, typically an IPv4 address. This mapping is a critical function in the Internet protocol suite**.**
This requires the devices to be identified by their MAC(media access control) address (unique to each device). The router will store the MAC address of each device that attempts to connect to it, in a log file called “arp log”. Together with the DHCP (dynamic host configuration protocol) server present on the router, the device is automatically assigned an IP address on the network. The DHCP server will also have a client log.

An attempt was made to create a node-red flow to request a log from a Linksys E3200 router connected via ethernet to the raspi-4, and for that log to be saved as a JSON in our InfluxDB database. The method used was to use an http request node to the router’s file directory that contained the log, however the only URL obtainable via the router navigation menu was a browser window that had a live view of the DHCP client log. An option to download a text file of the information displayed (the “dhcp server log”) is present via a clickable box, however it wasn’t possible to find a URL in order to request a download. We tried using the URL to the DHCP client real time log, but the file returned was an “empty” type file (no matter the file type specified). We also tried to analyze the html code to find a line corresponding to the download button, and maybe mimic the button action, but weren’t successful in obtaining a populated file.

Subsequently, an email node was used in Node Red with a debug node linked on the receiving end. The debug node was set to display the payload of the msg. It should be noted, however, when setting up the email node it was necessary to set the gmail “low security” setting on and to create an app password which was to be used in the email node.

After InfluxDB was successfully installed and launched, InfluxDB nodes were installed and data was stored into the created database through an inject node. With the creation of a database compatible with Node Red, Grafana was installed and launched in order to attempt to display the InfluxDB data on a graph over time. Once the Grafana graph was created and successfully displayed data over time (see *Figure 1)* of the simulated occupancy int value, the investigation into extracting the moustache-notation object from the JSON file email attachment began.

It was observed that the JSON file attachment received from the email node, which was sent by exporting the ‘devices’ information from the Fing app on a smartphone, was contained as a buffer array of 3187 items (see *Figure 3*). This buffer array was then converted into a string using the ‘toString’ command. The following line of code was used in a function node (see Figure 6) to achieve this:

```
msg.payload = msg.attachments[0].content.toString(‘utf-8’);
return msg;
```

The end of this function node was then connected to a JSON node (see Figure 7), which converted the string of the mustache-notation object into a JSON object, which was assigned to msg.payload. Following the JSON node, another function node (see Figure 8) was used in order to cycle through each device’s ‘state’ by means of a for loop in order to count how many devices are currently connected. This count was assigned to a variable named ‘count’ which was ultimately assigned to the msg.payload. The following code was used in a function node in order to achieve this task:

```
var i;
var count = 0;
for (i = 0; i < msg.payload.devices.length; i++) {
        if (msg.payload.devices[i].state == “UP”)
                count += 1;
}
msg.payload = count;
return msg;
```

This msg.payload, containing an int of the occupancy count, was then fed into a debug node (see Figure 11) as well as an InfluxDB node (see Figure 9 and Figure 10).

Grafana, using the InfluxDB database with the occupancy data, successfully plotted the occupancy data in a line graph with the values clocking in every five minutes (see Figure 12).

Observations

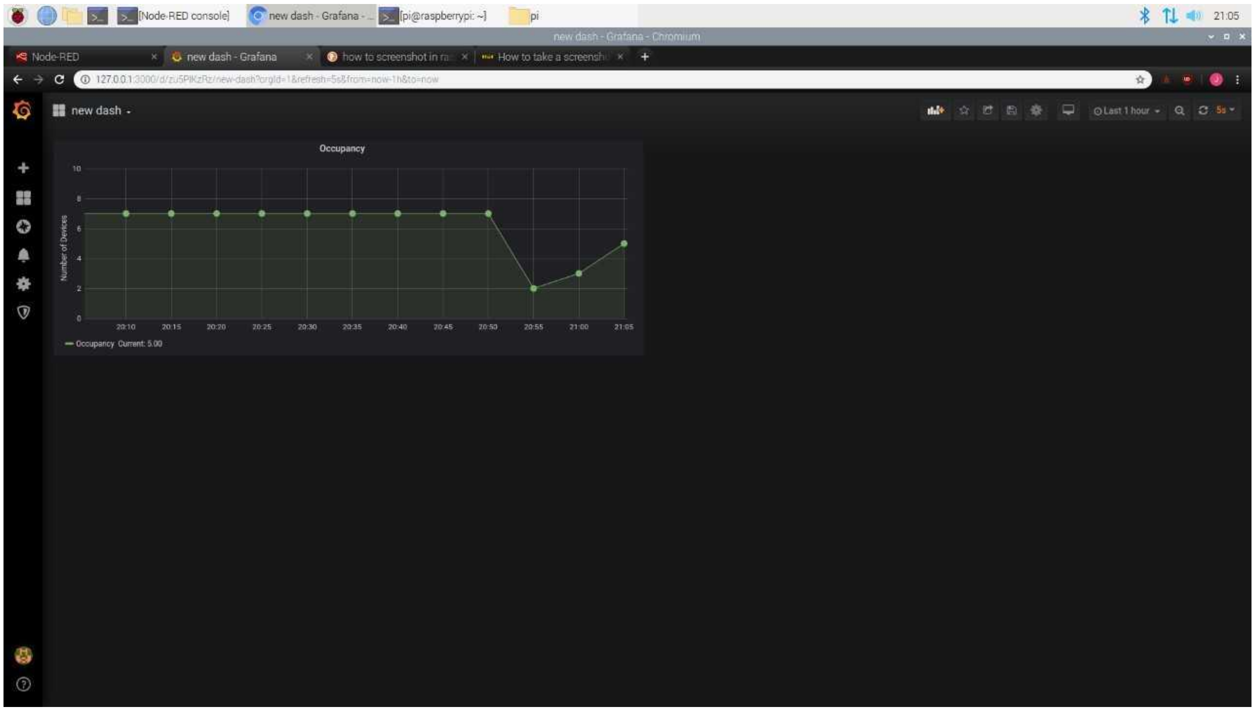
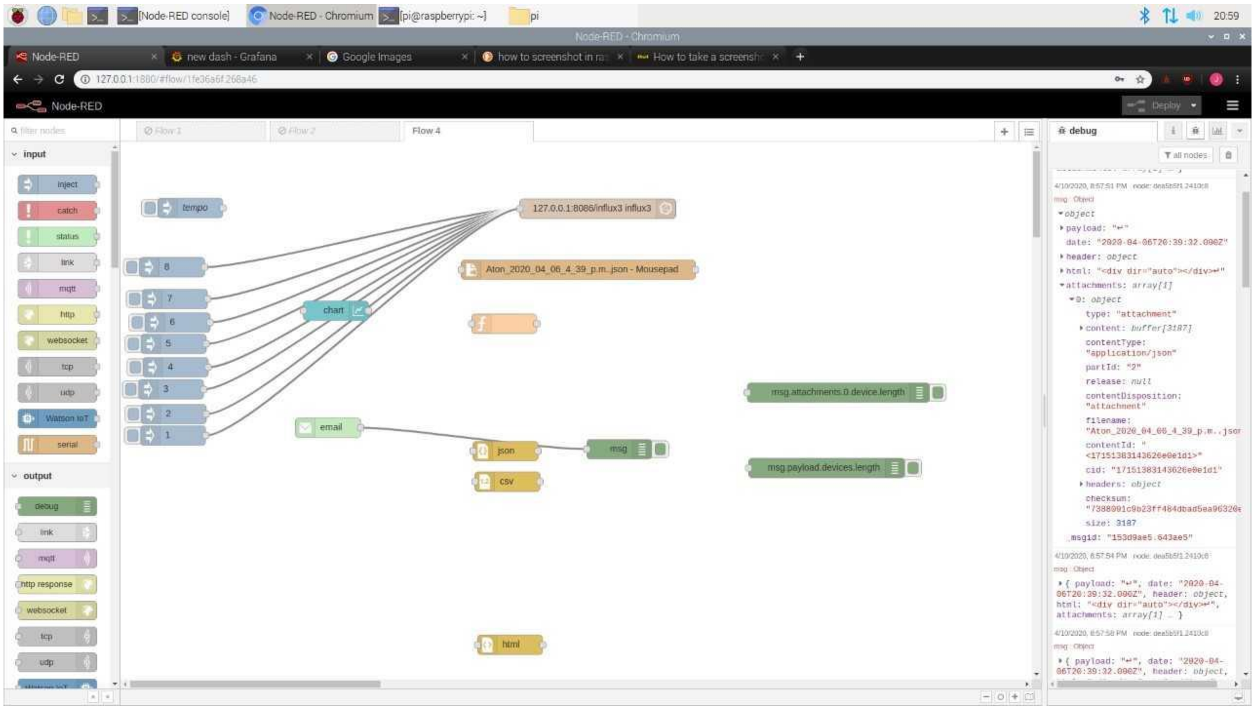
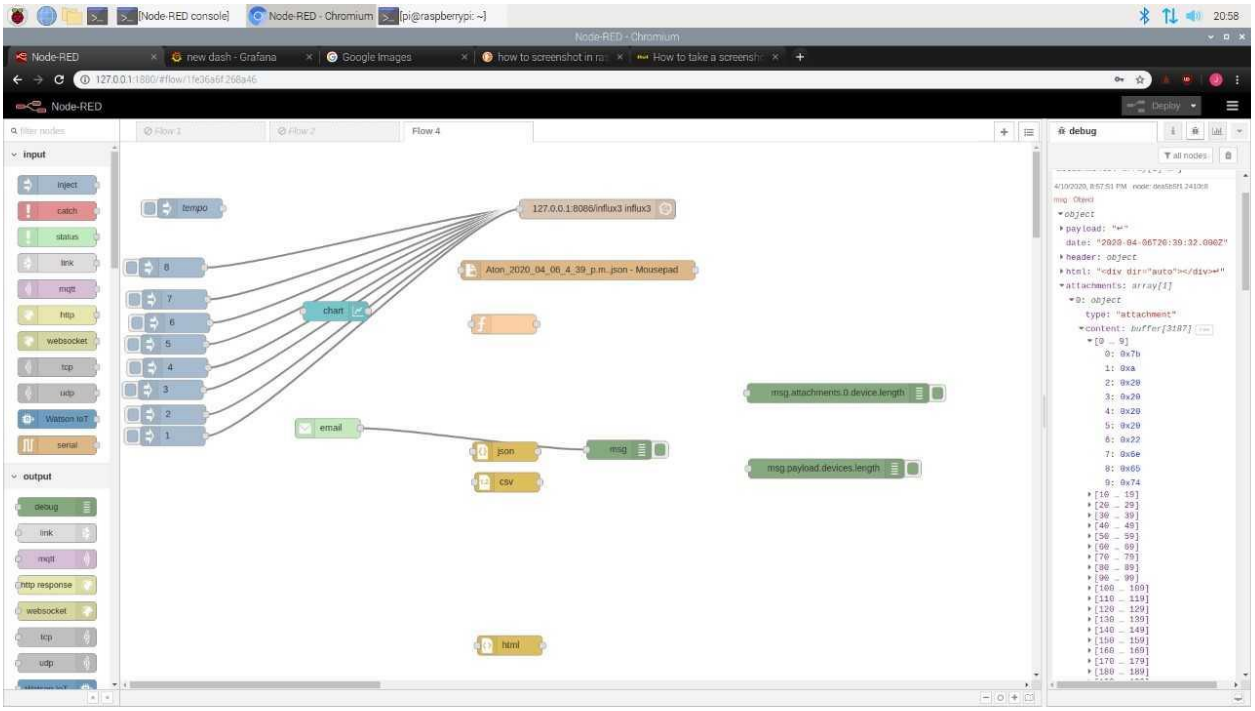


Figure 1. Grafana clocking in data from InfluxDB every 5 minutes



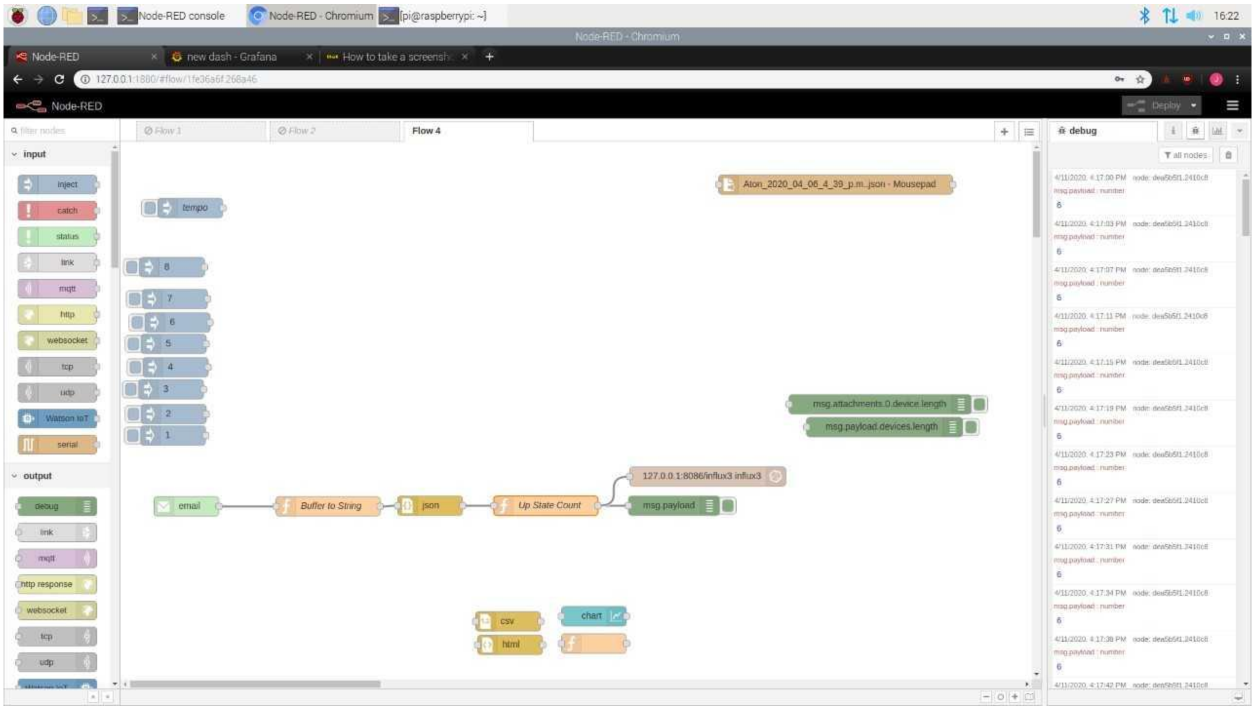
```
4/10/2020, 8:57:51 PM  node: dea5b5f1.2410c8
msg : Object
▾object
 ▸payload: "↵"
  date: "2020-04-06T20:39:32.000Z"
 ▸header: object
 ▸html: "<div dir="auto"></div>↵"
 ▾attachments: array[1]
   ▾0: object
      type: "attachment"
     ▸content: buffer[3187]
      contentType:
      "application/json"
      partId: "2"
      release: null
      contentDisposition:
      "attachment"
      filename:
      "Aton_2020_04_06_4_39_p.m..json
      contentId: "
      <17151383143626e0e1d1>"
      cid: "17151383143626e0e1d1"
     ▸headers: object
      checksum:
      "7388091c9b23ff484dbad5ea96320e
      size: 3187
  _msgid: "153d9ae5.643ae5"
4/10/2020, 8:57:54 PM  node: dea5b5f1.2410c8
msg : Object
▸{ payload: "↵", date: "2020-04-
06T20:39:32.000Z", header: object,
html: "<div dir="auto"></div>↵",
attachments: array[1] … }
4/10/2020, 8:57:58 PM  node: dea5b5f1.2410c8
msg : Object
▸{ payload: "↵", date: "2020-04-
06T20:39:32.000Z", header: object,
```

Figure 2. Attachment component of msg object from email node



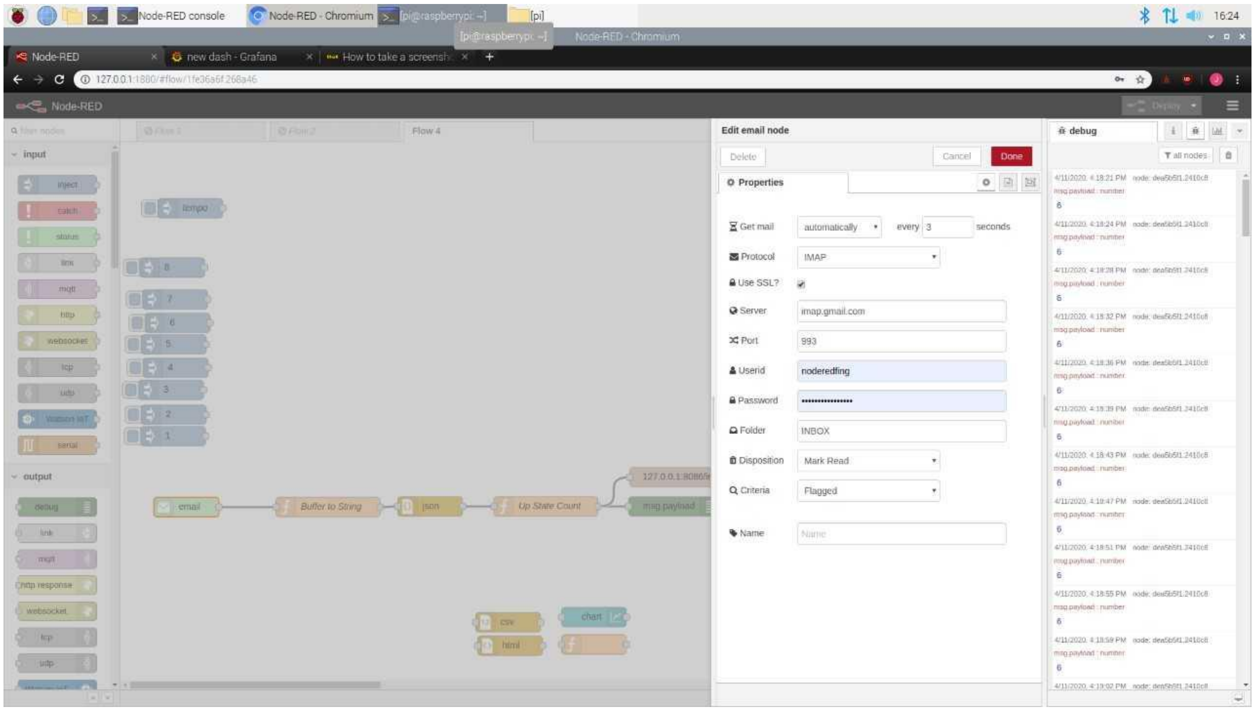
```
4/10/2020, 8:57:51 PM  node: dea5b5f1.2410c8
msg : Object
▾object
 ▸payload: "↵"
  date: "2020-04-06T20:39:32.000Z"
 ▸header: object
 ▸html: "<div dir="auto"></div>↵"
 ▾attachments: array[1]
   ▾0: object
      type: "attachment"
     ▾content: buffer[3187]
       ▾[0 … 9]
          0: 0x7b
          1: 0xa
          2: 0x20
          3: 0x20
          4: 0x20
          5: 0x20
          6: 0x22
          7: 0x6e
          8: 0x65
          9: 0x74
       ▸[10 … 19]
       ▸[20 … 29]
       ▸[30 … 39]
       ▸[40 … 49]
       ▸[50 … 59]
       ▸[60 … 69]
       ▸[70 … 79]
       ▸[80 … 89]
       ▸[90 … 99]
       ▸[100 … 109]
       ▸[110 … 119]
       ▸[120 … 129]
       ▸[130 … 139]
       ▸[140 … 149]
       ▸[150 … 159]
       ▸[160 … 169]
       ▸[170 … 179]
       ▸[180 … 189]
```

Figure 3. Buffer array of 3187 items which is the JSON file content



```
4/11/2020, 4:17:00 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:17:03 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:17:07 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:17:11 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:17:15 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:17:19 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:17:23 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:17:27 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:17:31 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:17:34 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:17:38 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:17:42 PM  node: dea5b5f1.2410c8
```

Figure 4. Final Node Red Flow with Debug Messages Displaying Occupancy Count (6)

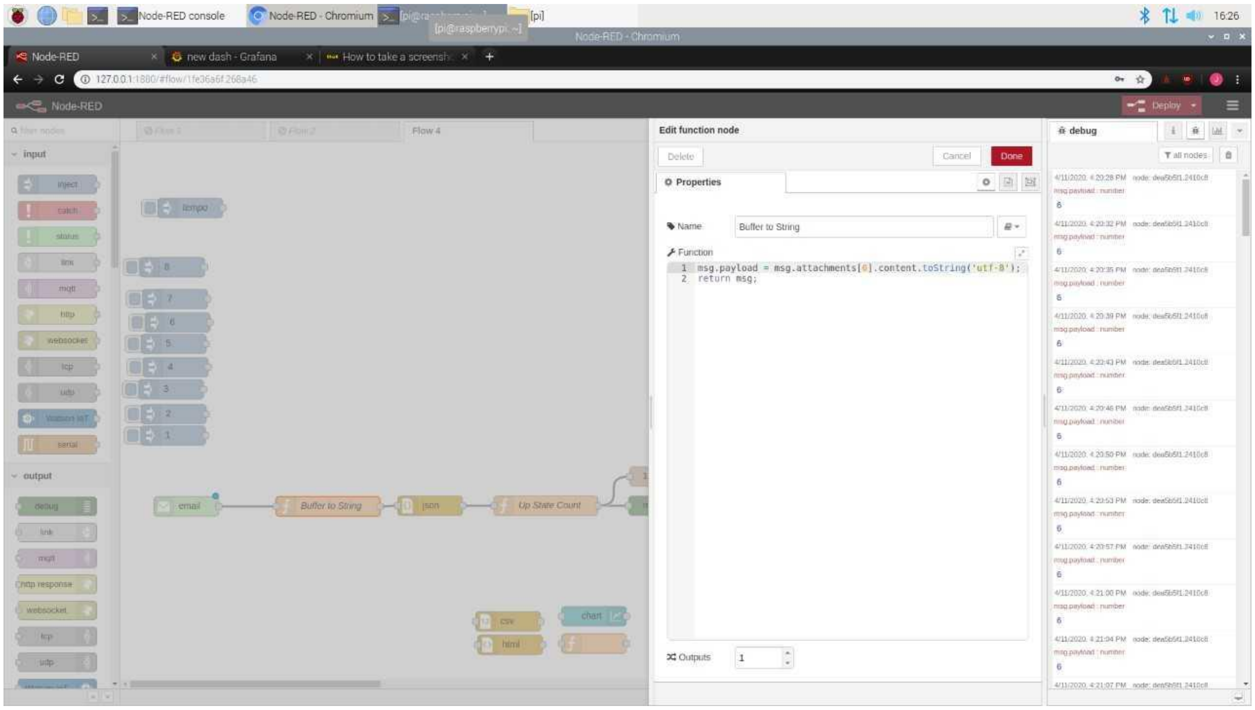


Edit email node

Get mail	automatically every 3 seconds
Protocol	IMAP
Use SSL?	☑
Server	imap.gmail.com
Port	993
Userid	noderedfing
Password	•••••••••••••••••
Folder	INBOX
Disposition	Mark Read
Criteria	Flagged
Name	Name

```
4/11/2020, 4:18:21 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:18:24 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:18:28 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:18:32 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:18:36 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:18:39 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:18:43 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:18:47 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:18:51 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:18:55 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:18:59 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:19:02 PM  node: dea5b5f1.2410c8
```

Figure 5. Email Node



Edit function node

Name: Buffer to String

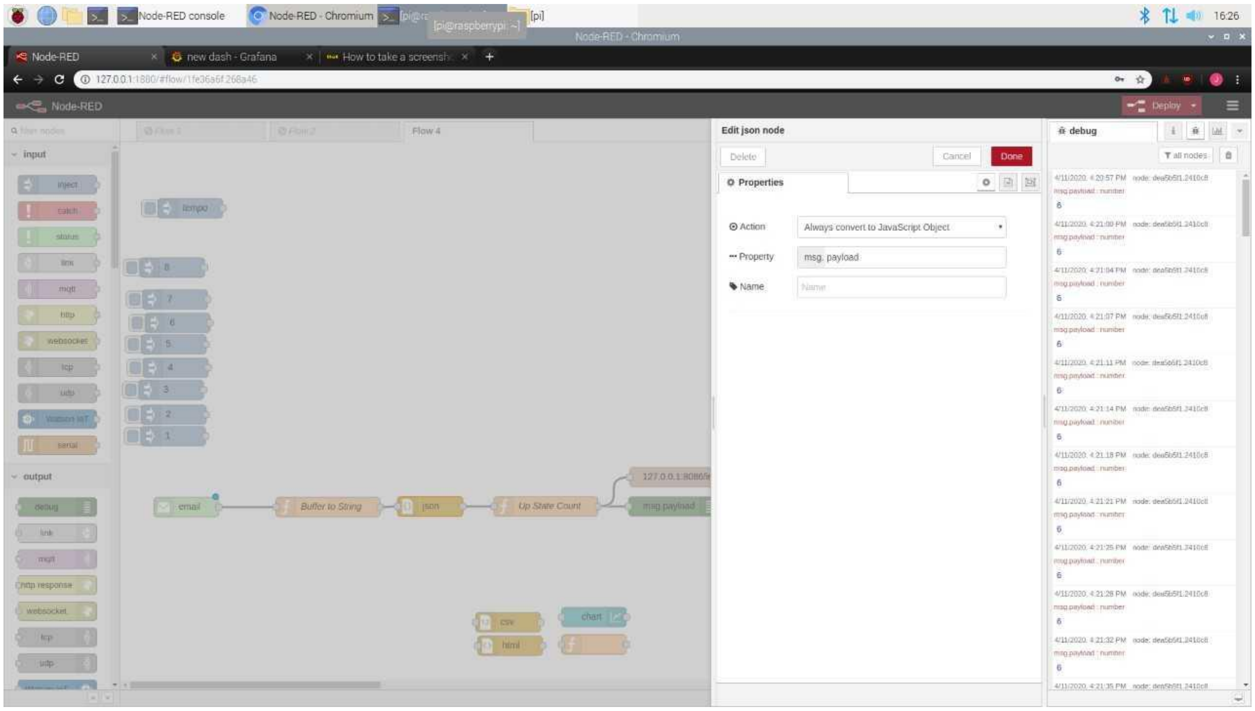
Function:

```
msg.payload = msg.attachments[0].content.toString('utf-8');
return msg;
```

Outputs: 1

```
4/11/2020, 4:20:28 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:20:32 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:20:35 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:20:39 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:20:43 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:20:46 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:20:50 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:20:53 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:20:57 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:21:00 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:21:04 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:21:07 PM   node: dea5b5f1.2410c8
```

Figure 6. Function Node to Convert Buffer Array into a String



Edit json node

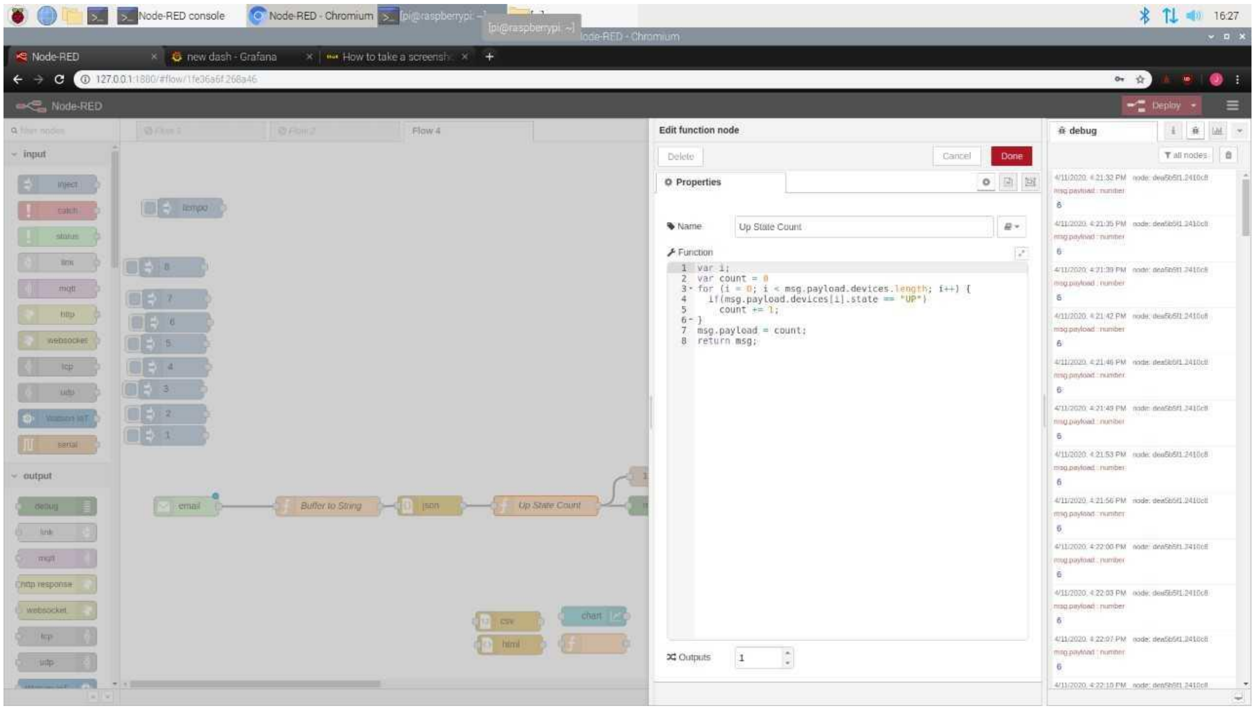
Action: Always convert to JavaScript Object

Property: msg. payload

Name: Name

```
4/11/2020, 4:20:57 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:21:00 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:21:04 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:21:07 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:21:11 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:21:14 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:21:18 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:21:21 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:21:25 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:21:28 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:21:32 PM   node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:21:35 PM   node: dea5b5f1.2410c8
```

Figure 7. JSON Node to Convert String into an Object Assigned to msg.payload



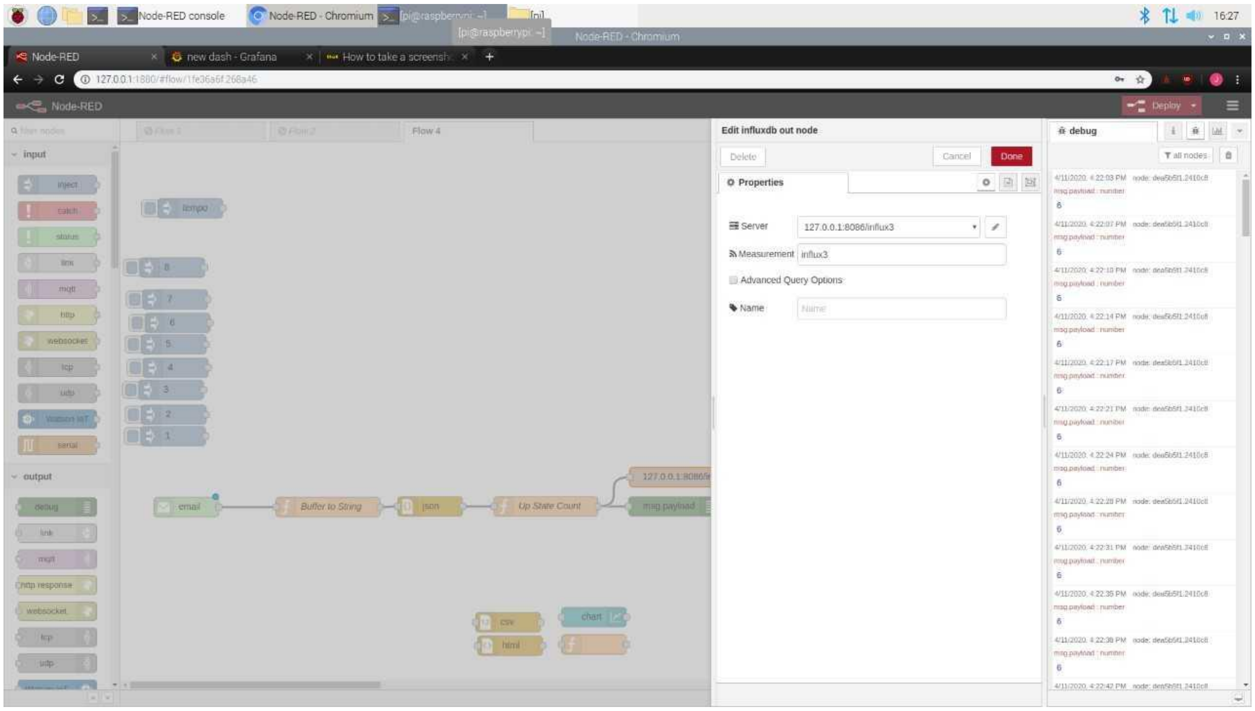
Name: Up State Count

Function

```
var i;
var count = 0
for (i = 0; i < msg.payload.devices.length; i++) {
  if(msg.payload.devices[i].state == "UP")
    count += 1;
}
msg.payload = count;
return msg;
```

Outputs: 1

Figure 8. Second Function Node which cycles through each msg.payload object's Devices, adding one to Count for every "UP" state component



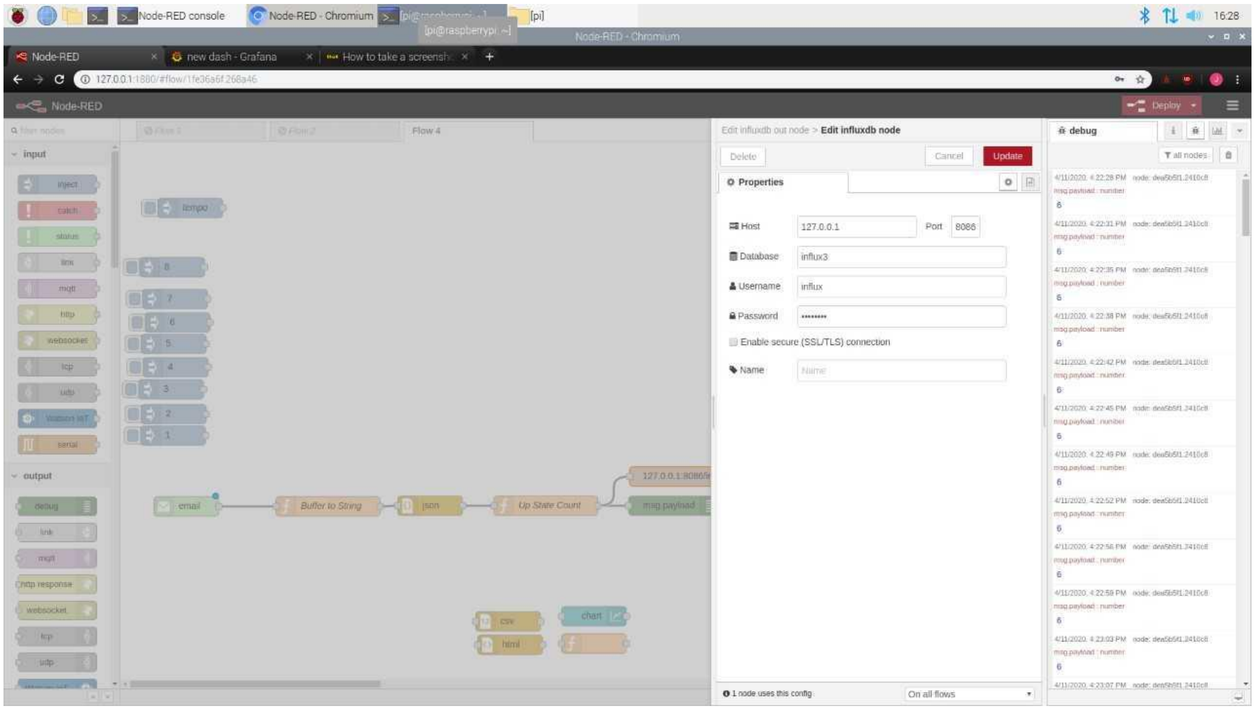
Server: 127.0.0.1:8086/influx3

Measurement: influx3

□ Advanced Query Options

Name: Name

Figure 9. InfluxDB Node Properties



Edit influxdb out node > **Edit influxdb node**

Delete | Cancel | Update

Properties

Host: 127.0.0.1 Port: 8086

Database: influx3

Username: influx

Password: ••••••••

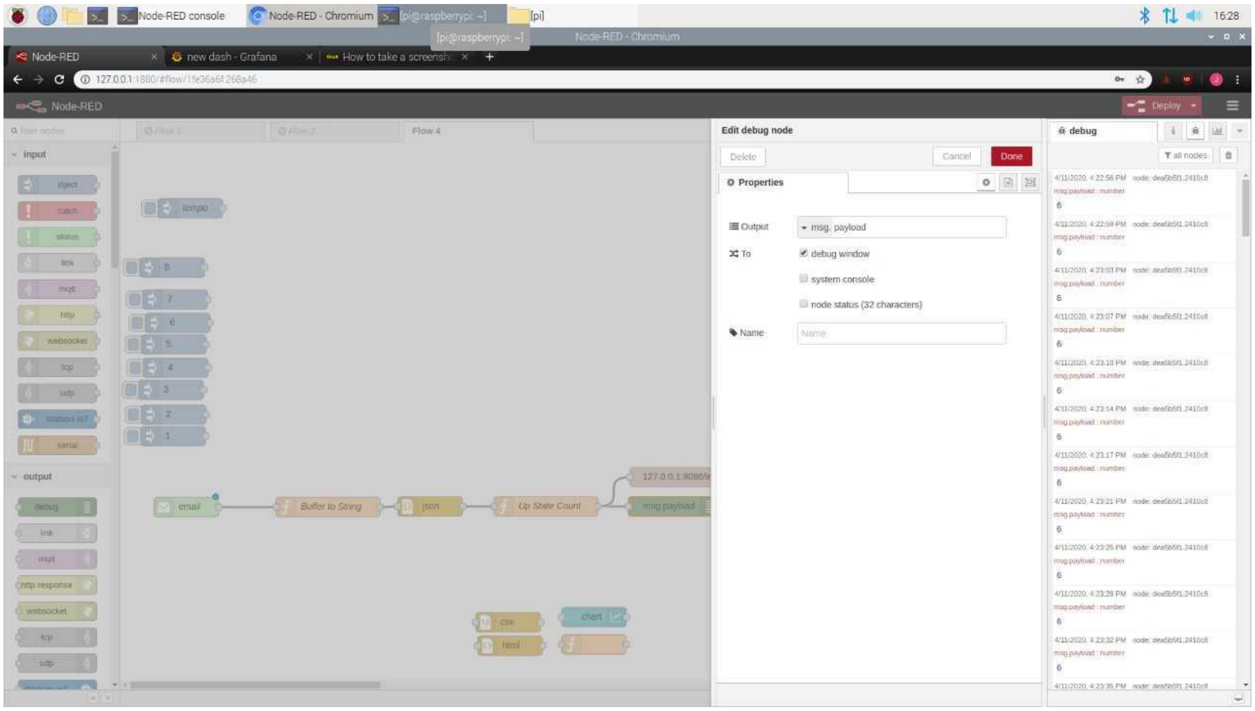
□ Enable secure (SSL/TLS) connection

Name: Name

1 node uses this config | On all flows

```
4/11/2020, 4:22:28 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:22:31 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:22:35 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:22:38 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:22:42 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:22:45 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:22:49 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:22:52 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:22:56 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:22:59 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:23:03 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:23:07 PM  node: dea5b5f1.2410c8
```

Figure 10. InfluxDB Node Properties - Server Properties



Edit debug node

Delete | Cancel | Done

Properties

Output: msg. payload

To: ☑ debug window

□ system console

□ node status (32 characters)

Name: Name

```
4/11/2020, 4:22:56 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:22:59 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:23:03 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:23:07 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:23:10 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:23:14 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:23:17 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:23:21 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:23:25 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:23:28 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:23:32 PM  node: dea5b5f1.2410c8
msg.payload : number
6
4/11/2020, 4:23:35 PM  node: dea5b5f1.2410c8
```

Figure 11. Debug Node Displaying Occupancy which was Assigned to msg.payload

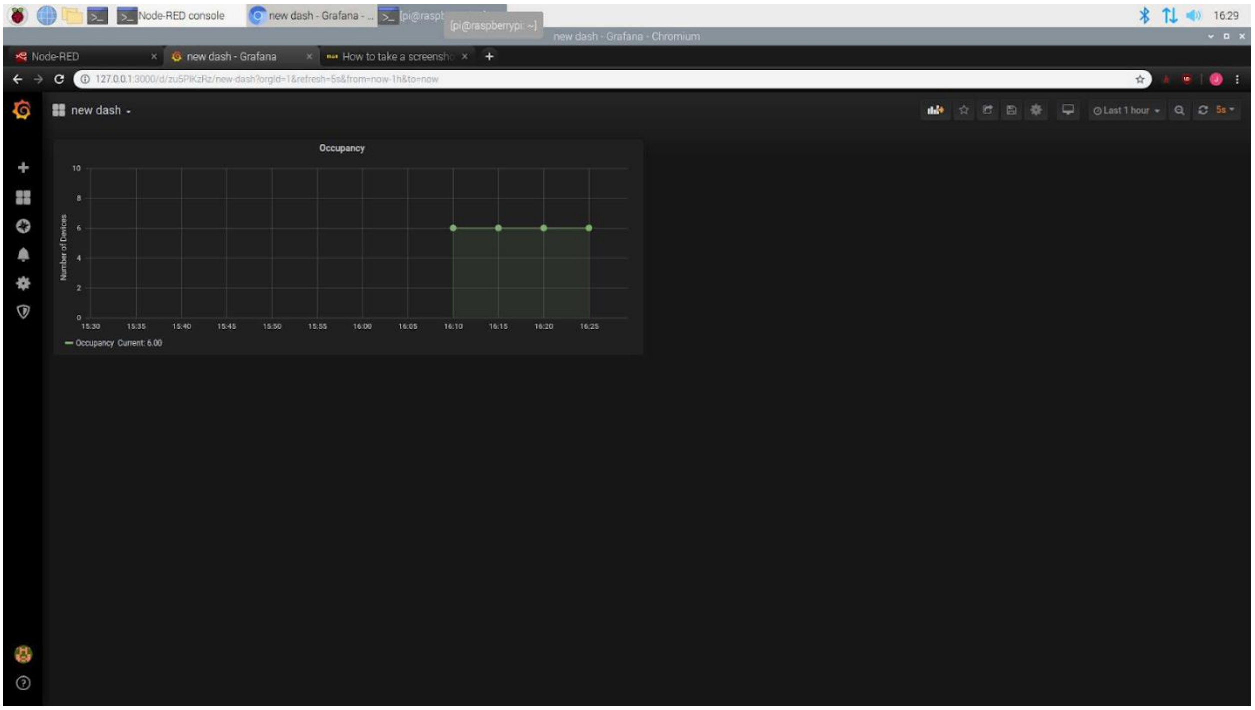


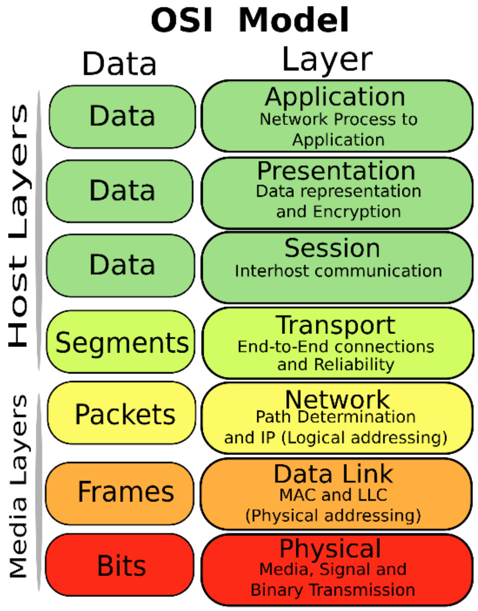
Figure 12. Grafana Graph Displaying Occupancy Data Clocked in Every Five Minutes

Discussion

With the desired result of router-based occupancy being achieved by means of Node Red operating on a Raspberry Pi 4, and having the router devices information being exported as a JSON file from the Fing mobile app, the investigation came to a close. One query relating to the created system is that the JSON file exporting must be done manually through the app, and therefore does not make this an automated system. Furthermore, the Fing app requires Internet connectivity to email the JSON file to the created email.

One of the issues which was initially encountered was the failed connectivity of the email node to the gmail server of the created email. This issue was resolved by creating and using an “App password” on the gmail account. Another issue was attempting to access the JSON file, delivered as an attachment in an email, and obtain its content in the form of an object. The JSON file itself, which was exported from the Fing app, was in moustache-notation. The required component of this JSON object was the length of the “devices” field as well as the state of each “devices”, either “UP” or “DOWN.” Ultimately this issue was resolved with the observation that the content of the JSON file was contained as a buffer array with a length of 3187 elements. Subsequently, the buffer array was converted to a string using ‘utf-8’ formatting and it was assigned to msg.payload.

In order to elaborate on the devices being detected, one must first go over the JSON object being received and used (see Appendix “JSON Object”). The JSON object being exported by email from the Fing app contains three fields, each of which has its own keys. The field of interest is the “devices” field, and the key of interest in this field is the “state” field. The count variable increases for every state with the string “UP” set to it, this indicates that the device is currently connected. It should be noted that all devices familiar to the network, which have been connected before, are included in the devices field. As can be seen in the following OSI model, the information relating to the JSON object takes part in the “Network” layer:


OSI Model
Data
Layer
Host Layers
Media Layers
Data
Application
Network Process to Application
Data
Presentation
Data representation and Encryption
Data
Session
Interhost communication
Segments
Transport
End-to-End connections and Reliability
Packets
Network
Path Determination and IP (Logical addressing)
Frames
Data Link
MAC and LLC (Physical addressing)
Bits
Physical
Media, Signal and Binary Transmission

The Network layer allows for the transferring of network packets from a source to a destination host. This layer responds to requests from the transport layer and issues service requests to the data link layer.

Connectionless communication, often referred to as CL-mode communication, is of great importance in the pursuit of a router-based occupancy system. To understand this better one can look at the IP (Internet Protocol) which is connectionless in that a data packet can travel from a sender to a recipient without the recipient having to send an acknowledgement.

Conclusion

It has been determined that router-based occupancy through the use of Node Red operating on a Raspberry Pi 4 is successful. By using the Fing app on a smartphone connected to a wireless router, the connected devices' information was exported as a JSON file and emailed as an attachment to a created gmail address. An email node was used and accessed the gmail account by using a generated "App password." The email node output was linked into the input of a function node which converted the buffer array into a string using the "toString" function and the 'utf-8' notation. This string was assigned to msg.payload which was fed into a JSON node that converted the string into a JSON object and assigned it to msg.payload. The output of the JSON node was fed into a function which cycled through the 'state' component of each device and added a one to a variable named count if the device state was "UP." This int was assigned to msg.payload which was then fed into a debug node as well as an InfluxDB node. A Grafana graph was set up, with the data source being InfluxDB, and the occupancy data was plotted with the data being clocked in every five minutes.

Recommendations

The router-based occupancy process created is functional, however improvements can be made. The process can be made to be automated and also functional without Internet connectivity. Both of these ideal features can be achieved by creating and using a Linux-based router on the Raspberry Pi. In this manner, the Raspberry Pi would be able to directly obtain information of the connected devices from its router service in the most convenient of ways. A regular router not connected to the Internet may also serve to complete this task, however the obtaining and processing of device information may be more complicated. Perhaps an 'http request' node may be used to obtain the device information from the router webpage.

It is possible to identify clients of a wireless router that are within range but not connected. Rogue APs (access points) and clients within broadcasting range can be shown by Cisco wireless access points, furthermore disassociation packets can be spoofed to rogue APs to keep them functioning until they can be removed from the network (Serverfault, 2013). On another note, WiFi Direct is a service which allows for two devices to communicate with each other without Internet access, provided that at least one of them is compliant with the standard to establish a peer-to-peer connection. The standard gives compatible devices a way to discover each other and securely connect using Wi-Fi Protected Setup and Wi-Fi Protected Access (WPA) (Netspotapp, 2020).

There was an investigation done looking into the use of WiFi probe requests that are continuously transmitted from WiFi enabled smart devices. The passive capturing of ambient probe requests from WiFi devices such as smart phones and tablets, where no connectivity to a WiFi network is required, is done with WiFi Pineapple equipment (Ciftler, B., Dikmese, S., Guvenc, I., Akkaya, K., Kadri, A, 2017). Another study discussed the use of Received Signal Strength (RSS) (by means of mobile phones) of BLE (Bluetooth Low Energy) nodes deployed around workspaces to localize the phone in a room (Pratama, A., Widyawan, W., Lazovik, A., Aiello, M., 2018). Another study looks into the use of WiFI received signal strength indicator (RSSI) measurements between a pair of stationary transmitter/receiver antennas. This study proposes a frame-work based on understanding two important ways that people leave their signature on the transmitted signa: blocking the Line of Sight (LoS) and scattering effects (Depatla, S., Muralidharan, A., Mostofi, Y., n.d.). Finally, there is a study which aims to define a model for a relationship between the RSSI of a Wireless Sensor Network (WSN) and the occupation rate of an indoor ambient (Barros, D., Mota, A., Mota, L., 2015).

References

Barros, D., Mota, A., Mota, L.: (September 1, 2015). Average Room Occupancy Rate and its relation with Received Signal Strength Indicator in Wireless Sensor Networks. *Brazilian Society of Television Engineering*. Retrieved from https://www.researchgate.net/publication/282523921_Average_Room_Occupancy_Rate_and_its_relation_with_Received_Signal_Strength_Indicator_in_Wireless_Sensors_Networks

Can a wireless router identify “clients” that are within range but NOT connected? (2013). *Serverfault*. Retrieved from: https://serverfault.com/questions/482825/can-a-wireless-router-identify-clients-that-are-within-range-but-not-connected

Ciftler, B., Dikmese, S., Guvenc, I., Akkaya, K., Kadri, A.: (February 21, 2017). Occupancy Counting with Burst and Intermittent Signals in Smart Buildings. *IEEE*. Retrieved from: https://arxiv.org/pdf/1702.06423.pdf

Depatla, S., Muralidharan, A., Mostofi, Y.: (n.d.) Occupancy Estimation Using Only WiFi Power Measurements. *UCSB*. Retrieved from: https://www.ece.ucsb.edu/~arjunm/JSAC15.pdf

Simma, K., Bogus, S., Mammoli, A.: (June 12 - 15, 2019). WI-FI ROUTER NETWORK-BASED OCCUPANCY ESTIMATION TO OPTIMIZE HVAC ENERGY CONSUMPTION. *CSCE Annual Conference*. Retrieved from: https://csce.ca/elf/apps/CONFERENCEVIEWER/conferences/2019/pdfs/PaperPDFversion_17_0227065457.pdf

Kelly, Vivian (2014-01-07). "New IEEE 802.11ac™ Specification Driven by Evolving Market Need for Higher, Multi-User Throughput in Wireless LANs". IEEE. Archived from the original on 2014-01-12. Retrieved 2014-01-11.

Pratama, A., Widyawan, W., Lazovik, A., Aiello, M.: (March 6, 2018). Multi-User Low Intrusive Occupancy Detection. *MDPI*. Retrieved from: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5876590/

Simma, K., Bogus, S., Mammoli, A.: (August 19 - 21, 2019). Real-Time Occupancy Estimation Using WiFi Network to Optimize HVAC Operation. *Science Direct*. Retrieved from: https://www.sciencedirect.com/science/article/pii/S1877050919309834

WiFi Direct: WiFi Without the Internet (2020). *Netspotapp*. Retrieved from: https://www.netspotapp.com/what-is-wifi-direct.html

Appendices

YouTube Playlist Demonstration

https://www.youtube.com/watch?v=e8vuesF4sBA&list=PLaXuq4j7e1YSrY3hb9ua2CFnPNzLpaueR&index=1

JSON Object

```
{
    "network": {
        "name": "Aton",
        "last_changed": "2020-04-06 3:24 p.m.",
        "address": "192.168.0.0",
        "netmask": "255.255.255.0",
        "gateway": "192.168.0.1",
        "dns": "192.168.0.1",
        "bssids": [
            "aa:aa:aa:aa:aa:aa"
        ]
    },
    "internet": {
        "provider": "Rogers",
        "address": "aaa.aa.aaa.aa",
        "hostname": "aaa.aa.aaa.aa",
        "country": "CA",
        "region": "Ontario",
        "city": "Ottawa",
        "timezone": "America\/Toronto"
    },
    "devices": [
        {
            "address": "192.168.0.1",
            "hardware_address": "aa:aa:aa:aa:aa:aa",
            "name": "CODA-4582U",
            "make": "Hitron Technology",
            "model": "CODA-4582U",
            "hostname": "hitronhub.home",
            "type": "Router",
            "state": "UP",
            "firstseen": "2020-03-31 4:21 p.m."
        },
        {
            "address": "192.168.0.10",
            "hardware_address": "aa:aa:aa:aa:aa:aa",
            "name": "GeorgePC",
            "make": "Asus",
            "type": "Computer",
            "state": "UP",
            "firstseen": "2020-03-31 4:21 p.m."
        },
        {
            "address": "192.168.0.11",
            "hardware_address": "aa:aa:aa:aa:aa:aa",
            "type": "Computer",
            "state": "UP",
            "firstseen": "2020-03-31 4:21 p.m."
        },
        {
            "address": "192.168.0.11",
            "hardware_address": "aa:aa:aa:aa:aa:aa",
            "make": "Apple",
            "model": "iPhone",
            "type": "Mobile",
            "state": "DOWN",
            "firstseen": "2020-03-31 4:21 p.m."
        },
        {
            "address": "192.168.0.13",
            "hardware_address": "aa:aa:aa:aa:aa:aa",
            "make": "Apple",
            "model": "iPhone",
            "type": "Mobile",
            "state": "DOWN",
            "firstseen": "2020-03-31 4:28 p.m."
        },
        {
            "address": "192.168.0.15",
            "hardware_address": "aa:aa:aa:aa:aa:aa",
            "name": "Huawei nova 4e",
            "make": "Huawei",
            "model": "P30 lite",
            "type": "Mobile",
            "state": "UP",
            "firstseen": "2020-03-31 4:21 p.m."
        },
        {
            "address": "192.168.0.16",
            "hardware_address": "aa:aa:aa:aa:aa:aa",
            "make": "Amazon",
            "type": "Computer",
            "state": "DOWN",
            "firstseen": "2020-03-31 4:29 p.m."
        },
        {
            "address": "192.168.0.17",
            "hardware_address": "aa:aa:aa:aa:aa:aa",
            "make": "Raspberry Pi",
            "model": "Raspberry Pi",
            "type": "Raspberry",
            "state": "UP",
            "firstseen": "2020-04-05 12:54 p.m."
        },
        {
            "address": "192.168.0.254",
            "hardware_address": "aa:aa:aa:aa:aa:aa",
            "name": "WPS Access Point",
            "make": "Hitron Technology",
            "model": "APxx",
            "type": "Wi-Fi",
            "state": "UP",
            "firstseen": "2020-03-31 4:21 p.m."
        }
    ]
}
```

Flow Format 1:

[{"id":"1fe36a6f.268a46","type":"tab","label":"Flow
4","disabled":false,"info":""},{"id":"918938e7.f8b628","type":"e-mail
in","z":"1fe36a6f.268a46","name":"","protocol":"IMAP","server":"imap.gmail.com","useSSL":tr
ue,"port":"993","box":"INBOX","disposition":"Read","criteria":"FLAGGED","repeat":"3","fetch":
"auto","inputs":0,"x":101.5,"y":561,"wires":[["1c07bb38.8c5005"]]},{"id":"fb904cbd.78b7e","ty
pe":"debug","z":"1fe36a6f.268a46","name":"","active":true,"tosidebar":true,"console":false,"t
ostatus":false,"complete":"payload.devices.length","targetType":"msg","x":1175.5,"y":439,"wir
es":[]},{"id":"a0d68429.bdca48","type":"json","z":"1fe36a6f.268a46","name":"","property":"pa
yload","action":"obj","pretty":true,"x":476.5,"y":560,"wires":[["2438a8fb.aea508"]]},{"id":"a24
d4e1f.c41c2","type":"file
in","z":"1fe36a6f.268a46","name":"","filename":"Aton_2020_04_06_4_39_p.m..json -
Mousepad","format":"","chunk":false,"sendError":false,"encoding":"none","x":1100.5,"y":66,"
wires":[[]]},{"id":"932a90ee.8e5a2","type":"debug","z":"1fe36a6f.268a46","name":"","active":t
rue,"tosidebar":true,"console":false,"tostatus":false,"complete":"attachments.0.device.length",
"targetType":"msg","x":1168.5,"y":404,"wires":[]},{"id":"fc6ddc89.fea5c","type":"csv","z":"1fe3
6a6f.268a46","name":"","sep":",","hdrin":"","hdrout":"","multi":"one","ret":"\\n","temp":"","s
kip":"0","x":596.5,"y":738,"wires":[[]]},{"id":"6fae8a51.725a64","type":"html","z":"1fe36a6f.26
8a46","name":"","property":"attachments.0","outproperty":"payload","tag":"","ret":"html","as
":"single","x":599.5,"y":774,"wires":[[]]},{"id":"b59e3658.aff9a8","type":"function","z":"1fe36a
6f.268a46","name":"","func":"//payload={\"payload\":JSON.stringify(msg.attachments)};\n//ho
pp = msg.attachments(0);\n//payload={\"payload\":JSON.stringify(msg.paylod)};\n//payload =
msg.attachments.devices;\n//msg.payload = payload\nvar data = {\n   key: 'value'\n};\nvar
fileName = 'myData.json';\n\n// Create a blob of the data\nvar fileToSave = new
Blob([JSON.stringify(data)], {\n   type: 'application/json',\n   name: fileName\n});\n\n// Save
the file\nsaveAs(fileToSave, fileName);\nreturn
msg;","outputs":1,"noerr":0,"x":728.5,"y":772,"wires":[[]]},{"id":"d13491bd.914b2","type":"infl
uxdb
out","z":"1fe36a6f.268a46","influxdb":"6bc31684.c5f6f8","name":"","measurement":"influx3",
"precision":"","retentionPolicy":"","x":903.5,"y":515,"wires":[]},{"id":"c2d83147.7bddb","type":
"inject","z":"1fe36a6f.268a46","name":"tempo","topic":"","payload":"7","payloadType":"str","r
epeat":"","crontab":"","once":false,"onceDelay":0.1,"x":107.5,"y":102,"wires":[[]]},{"id":"304a2
84c.1e49d8","type":"inject","z":"1fe36a6f.268a46","name":"","topic":"","payload":"8","payloa
dType":"num","repeat":"","crontab":"","once":false,"onceDelay":0.1,"x":79.5,"y":193,"wires":[[
]]},{"id":"eec7df01.fa088","type":"ui_chart","z":"1fe36a6f.268a46","name":"","group":"68fc06f
a.480528","order":2,"width":0,"height":0,"label":"chart","chartType":"line","legend":"false","xf
ormat":"HH:mm:ss","interpolate":"linear","nodata":"","dot":false,"ymin":"","ymax":"","remove
Older":1,"removeOlderPoints":"","removeOlderUnit":"3600","cutout":0,"useOneColor":false,"c
olors":["#1f77b4","#aec7e8","#ff7f0e","#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#
c5b0d5"],"useOldStyle":false,"outputs":1,"x":728.5,"y":730,"wires":[[]]},{"id":"dea5b5f1.2410c8
","type":"debug","z":"1fe36a6f.268a46","name":"","active":true,"tosidebar":true,"console":fals

```
e,"tostatus":false,"complete":"payload","targetType":"msg","x":851.5,"y":560,"wires":[]},{"id":"c781fb20.ecadb8","type":"inject","z":"1fe36a6f.268a46","name":"","topic":"","payload":"5","payloadType":"num","repeat":"","crontab":"","once":false,"onceDelay":0.1,"x":82,"y":311,"wires":[[]]},{"id":"6fe0c14c.919f","type":"inject","z":"1fe36a6f.268a46","name":"","topic":"","payload":"7","payloadType":"num","repeat":"","crontab":"","once":false,"onceDelay":0.1,"x":84,"y":242,"wires":[[]]},{"id":"7d9fcfac.7fa4","type":"inject","z":"1fe36a6f.268a46","name":"","topic":"","payload":"6","payloadType":"num","repeat":"","crontab":"","once":false,"onceDelay":0.1,"x":88,"y":278,"wires":[[]]},{"id":"ee0191f9.0f2ae","type":"inject","z":"1fe36a6f.268a46","name":"","topic":"","payload":"4","payloadType":"num","repeat":"","crontab":"","once":false,"onceDelay":0.1,"x":85,"y":347,"wires":[[]]},{"id":"475e3baa.38f9d4","type":"inject","z":"1fe36a6f.268a46","name":"","topic":"","payload":"3","payloadType":"num","repeat":"","crontab":"","once":false,"onceDelay":0.1,"x":78,"y":381,"wires":[[]]},{"id":"bbba6cd5.e3d4c","type":"inject","z":"1fe36a6f.268a46","name":"","topic":"","payload":"2","payloadType":"num","repeat":"","crontab":"","once":false,"onceDelay":0.1,"x":82,"y":419,"wires":[[]]},{"id":"eb691d8e.05d66","type":"inject","z":"1fe36a6f.268a46","name":"","topic":"","payload":"1","payloadType":"num","repeat":"","crontab":"","once":false,"onceDelay":0.1,"x":81,"y":452,"wires":[[]]},{"id":"1c07bb38.8c5005","type":"function","z":"1fe36a6f.268a46","name":"Buffer to String","func":"msg.payload = msg.attachments[0].content.toString('utf-8');\nreturn msg;","outputs":1,"noerr":0,"x":319.5,"y":561,"wires":[["a0d68429.bdca48"]]},{"id":"2438a8fb.aea508","type":"function","z":"1fe36a6f.268a46","name":"Up State Count","func":"var i;\nvar count = 0\nfor (i = 0; i < msg.payload.devices.length; i++) { \n  if(msg.payload.devices[i].state == \"UP\")\n    count += 1;\n}\nmsg.payload = count;\nreturn msg;","outputs":1,"noerr":0,"x":654.5,"y":560,"wires":[["dea5b5f1.2410c8","d13491bd.914b2"]]},{"id":"6bc31684.c5f6f8","type":"influxdb","z":"","hostname":"127.0.0.1","port":"8086","protocol":"http","database":"influx3","name":"","usetls":false,"tls":""},{"id":"68fc06fa.480528","type":"ui_group","z":"","name":"templog","tab":"778a27f0.d11f18","disp":true,"width":"6","collapse":false},{"id":"778a27f0.d11f18","type":"ui_tab","z":"","name":"temp","icon":"dashboard","disabled":false,"hidden":false}]
```

Flow Format 2:

```
[
  {
    "id": "1fe36a6f.268a46",
    "type": "tab",
    "label": "Flow 4",
    "disabled": false,
    "info": ""
  },
  {
    "id": "918938e7.f8b628",
    "type": "e-mail in",
    "z": "1fe36a6f.268a46",
    "name": "",
    "protocol": "IMAP",
    "server": "imap.gmail.com",
    "useSSL": true,
    "port": "993",
    "box": "INBOX",
    "disposition": "Read",
    "criteria": "FLAGGED",
    "repeat": "3",
    "fetch": "auto",
    "inputs": 0,
    "x": 101.5,
    "y": 561,
    "wires": [
      [
        "1c07bb38.8c5005"
      ]
    ]
  },
  {
    "id": "fb904cbd.78b7e",
    "type": "debug",
    "z": "1fe36a6f.268a46",
    "name": "",
    "active": true,
    "tosidebar": true,
    "console": false,
    "tostatus": false,
        "complete": "payload.devices.length",
        "targetType": "msg",
        "x": 1175.5,
        "y": 439,
        "wires": []
    },
    {
        "id": "a0d68429.bdca48",
        "type": "json",
        "z": "1fe36a6f.268a46",
        "name": "",
        "property": "payload",
        "action": "obj",
        "pretty": true,
        "x": 476.5,
        "y": 560,
        "wires": [
            [
                "2438a8fb.aea508"
            ]
        ]
    },
    {
        "id": "a24d4e1f.c41c2",
        "type": "file in",
        "z": "1fe36a6f.268a46",
        "name": "",
        "filename": "Aton_2020_04_06_4_39_p.m..json - Mousepad",
        "format": "",
        "chunk": false,
        "sendError": false,
        "encoding": "none",
        "x": 1100.5,
        "y": 66,
        "wires": [
            []
        ]
    },
    {
        "id": "932a90ee.8e5a2",
        "type": "debug",
        "z": "1fe36a6f.268a46",
        "name": "",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "attachments.0.device.length",
        "targetType": "msg",
        "x": 1168.5,
        "y": 404,
        "wires": []
    },
    {
        "id": "fc6ddc89.fea5c",
        "type": "csv",
        "z": "1fe36a6f.268a46",
        "name": "",
        "sep": ",",
        "hdrin": "",
        "hdrout": "",
        "multi": "one",
        "ret": "\\n",
        "temp": "",
        "skip": "0",
        "x": 596.5,
        "y": 738,
        "wires": [
            []
        ]
    },
    {
        "id": "6fae8a51.725a64",
        "type": "html",
        "z": "1fe36a6f.268a46",
        "name": "",
        "property": "attachments.0",
        "outproperty": "payload",
        "tag": "",
        "ret": "html",
        "as": "single",
        "x": 599.5,
        "y": 774,
        "wires": [
            []
        ]
    },
    {
      "id": "b59e3658.aff9a8",
      "type": "function",
      "z": "1fe36a6f.268a46",
      "name": "",
      "func": "//payload={\"payload\":JSON.stringify(msg.attachments)};\n//hopp =
msg.attachments(0);\n//payload={\"payload\":JSON.stringify(msg.paylod)};\n//payload =
msg.attachments.devices;\n//msg.payload = payload\nvar data = {\n    key: 'value'\n};\nvar
fileName = 'myData.json';\n\n// Create a blob of the data\nvar fileToSave = new
Blob([JSON.stringify(data)], {\n    type: 'application/json',\n    name: fileName\n});\n\n// Save
the file\nsaveAs(fileToSave, fileName);\nreturn msg;",
      "outputs": 1,
      "noerr": 0,
      "x": 728.5,
      "y": 772,
      "wires": [
        []
      ]
    },
    {
      "id": "d13491bd.914b2",
      "type": "influxdb out",
      "z": "1fe36a6f.268a46",
      "influxdb": "6bc31684.c5f6f8",
      "name": "",
      "measurement": "influx3",
      "precision": "",
      "retentionPolicy": "",
      "x": 903.5,
      "y": 515,
      "wires": []
    },
    {
      "id": "c2d83147.7bddb",
      "type": "inject",
      "z": "1fe36a6f.268a46",
      "name": "tempo",
      "topic": "",
      "payload": "7",
      "payloadType": "str",
      "repeat": "",
      "crontab": "",
      "once": false,
      "onceDelay": 0.1,
      "x": 107.5,
    "y": 102,
    "wires": [
      []
    ]
  },
  {
    "id": "304a284c.1e49d8",
    "type": "inject",
    "z": "1fe36a6f.268a46",
    "name": "",
    "topic": "",
    "payload": "8",
    "payloadType": "num",
    "repeat": "",
    "crontab": "",
    "once": false,
    "onceDelay": 0.1,
    "x": 79.5,
    "y": 193,
    "wires": [
      []
    ]
  },
  {
    "id": "eec7df01.fa088",
    "type": "ui_chart",
    "z": "1fe36a6f.268a46",
    "name": "",
    "group": "68fc06fa.480528",
    "order": 2,
    "width": 0,
    "height": 0,
    "label": "chart",
    "chartType": "line",
    "legend": "false",
    "xformat": "HH:mm:ss",
    "interpolate": "linear",
    "nodata": "",
    "dot": false,
    "ymin": "",
    "ymax": "",
    "removeOlder": 1,
    "removeOlderPoints": "",
    "removeOlderUnit": "3600",
        "cutout": 0,
        "useOneColor": false,
        "colors": [
            "#1f77b4",
            "#aec7e8",
            "#ff7f0e",
            "#2ca02c",
            "#98df8a",
            "#d62728",
            "#ff9896",
            "#9467bd",
            "#c5b0d5"
        ],
        "useOldStyle": false,
        "outputs": 1,
        "x": 728.5,
        "y": 730,
        "wires": [
            []
        ]
    },
    {
        "id": "dea5b5f1.2410c8",
        "type": "debug",
        "z": "1fe36a6f.268a46",
        "name": "",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "payload",
        "targetType": "msg",
        "x": 851.5,
        "y": 560,
        "wires": []
    },
    {
        "id": "c781fb20.ecadb8",
        "type": "inject",
        "z": "1fe36a6f.268a46",
        "name": "",
        "topic": "",
        "payload": "5",
        "payloadType": "num",
    "repeat": "",
    "crontab": "",
    "once": false,
    "onceDelay": 0.1,
    "x": 82,
    "y": 311,
    "wires": [
      []
    ]
  },
  {
    "id": "6fe0c14c.919f",
    "type": "inject",
    "z": "1fe36a6f.268a46",
    "name": "",
    "topic": "",
    "payload": "7",
    "payloadType": "num",
    "repeat": "",
    "crontab": "",
    "once": false,
    "onceDelay": 0.1,
    "x": 84,
    "y": 242,
    "wires": [
      []
    ]
  },
  {
    "id": "7d9fcfac.7fa4",
    "type": "inject",
    "z": "1fe36a6f.268a46",
    "name": "",
    "topic": "",
    "payload": "6",
    "payloadType": "num",
    "repeat": "",
    "crontab": "",
    "once": false,
    "onceDelay": 0.1,
    "x": 88,
    "y": 278,
    "wires": [
      []
    ]
},
{
    "id": "ee0191f9.0f2ae",
    "type": "inject",
    "z": "1fe36a6f.268a46",
    "name": "",
    "topic": "",
    "payload": "4",
    "payloadType": "num",
    "repeat": "",
    "crontab": "",
    "once": false,
    "onceDelay": 0.1,
    "x": 85,
    "y": 347,
    "wires": [
        []
    ]
},
{
    "id": "475e3baa.38f9d4",
    "type": "inject",
    "z": "1fe36a6f.268a46",
    "name": "",
    "topic": "",
    "payload": "3",
    "payloadType": "num",
    "repeat": "",
    "crontab": "",
    "once": false,
    "onceDelay": 0.1,
    "x": 78,
    "y": 381,
    "wires": [
        []
    ]
},
{
    "id": "bbba6cd5.e3d4c",
    "type": "inject",
    "z": "1fe36a6f.268a46",
    "name": "",
    "topic": "",
        "payload": "2",
        "payloadType": "num",
        "repeat": "",
        "crontab": "",
        "once": false,
        "onceDelay": 0.1,
        "x": 82,
        "y": 419,
        "wires": [
            []
        ]
    },
    {
        "id": "eb691d8e.05d66",
        "type": "inject",
        "z": "1fe36a6f.268a46",
        "name": "",
        "topic": "",
        "payload": "1",
        "payloadType": "num",
        "repeat": "",
        "crontab": "",
        "once": false,
        "onceDelay": 0.1,
        "x": 81,
        "y": 452,
        "wires": [
            []
        ]
    },
    {
        "id": "1c07bb38.8c5005",
        "type": "function",
        "z": "1fe36a6f.268a46",
        "name": "Buffer to String",
        "func": "msg.payload = msg.attachments[0].content.toString('utf-8');\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 319.5,
        "y": 561,
        "wires": [
            [
                "a0d68429.bdca48"
            ]
    ]
  },
  {
    "id": "2438a8fb.aea508",
    "type": "function",
    "z": "1fe36a6f.268a46",
    "name": "Up State Count",
    "func": "var i;\nvar count = 0\nfor (i = 0; i < msg.payload.devices.length; i++) {\n  if(msg.payload.devices[i].state == \"UP\")\n    count += 1;\n}\nmsg.payload = count;\nreturn msg;",
    "outputs": 1,
    "noerr": 0,
    "x": 654.5,
    "y": 560,
    "wires": [
      [
        "dea5b5f1.2410c8",
        "d13491bd.914b2"
      ]
    ]
  },
  {
    "id": "6bc31684.c5f6f8",
    "type": "influxdb",
    "z": "",
    "hostname": "127.0.0.1",
    "port": "8086",
    "protocol": "http",
    "database": "influx3",
    "name": "",
    "usetls": false,
    "tls": ""
  },
  {
    "id": "68fc06fa.480528",
    "type": "ui_group",
    "z": "",
    "name": "templog",
    "tab": "778a27f0.d11f18",
    "disp": true,
    "width": "6",
    "collapse": false
  },
  {
        "id": "778a27f0.d11f18",
        "type": "ui_tab",
        "z": "",
        "name": "temp",
        "icon": "dashboard",
        "disabled": false,
        "hidden": false
    }
]
```