

**Dr Craig Gallen (OpenNMS UK)**  
**(also Solent University)**

Email : [craig.gallen@entimoss.com](mailto:craig.gallen@entimoss.com)  
: [cgallen@opennms.org](mailto:cgallen@opennms.org)

Mobile: +44 (0) 7789 938012



Image: [https://en.wikipedia.org/wiki/Internet\\_of\\_things](https://en.wikipedia.org/wiki/Internet_of_things)



# Contents

- **General discussion of IoT**
- MQTT
- OpenNMS IoT Plugin
- Use Cases



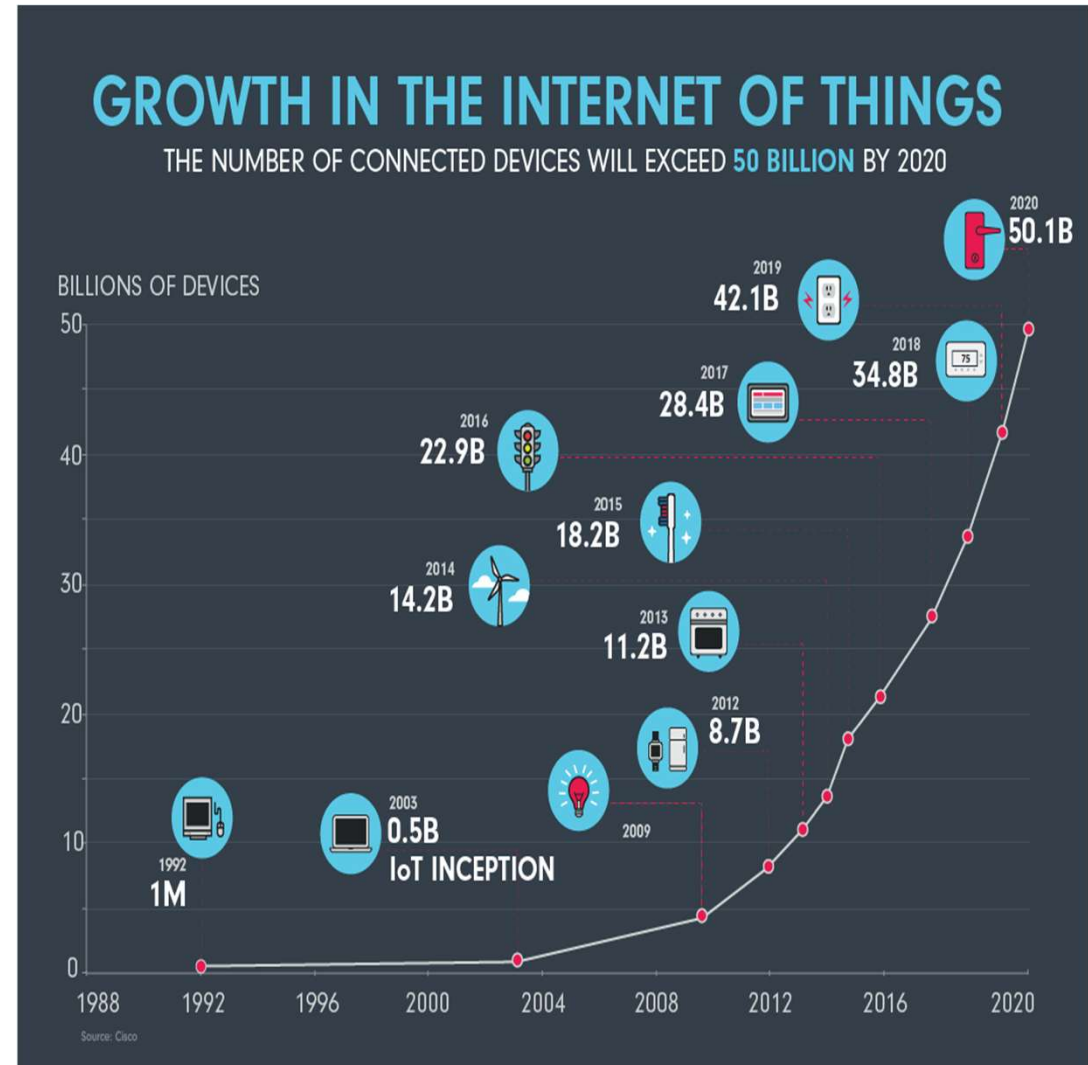


# IoT Definition

- The Internet of Things is :
- *"simply the point in time when more 'things or objects' were connected to the Internet than people" (cisco)*
- 'Fog Computing'
- Cisco Systems estimates that IoT was "born" between 2008 and 2009, with the things/people ratio growing from 0.08 in 2003 to 1.84 in 2010.

Dave Evans (April 2011). "The Internet of Things: How the Next Evolution of the Internet Is Changing Everything" (PDF). CISCO white paper.

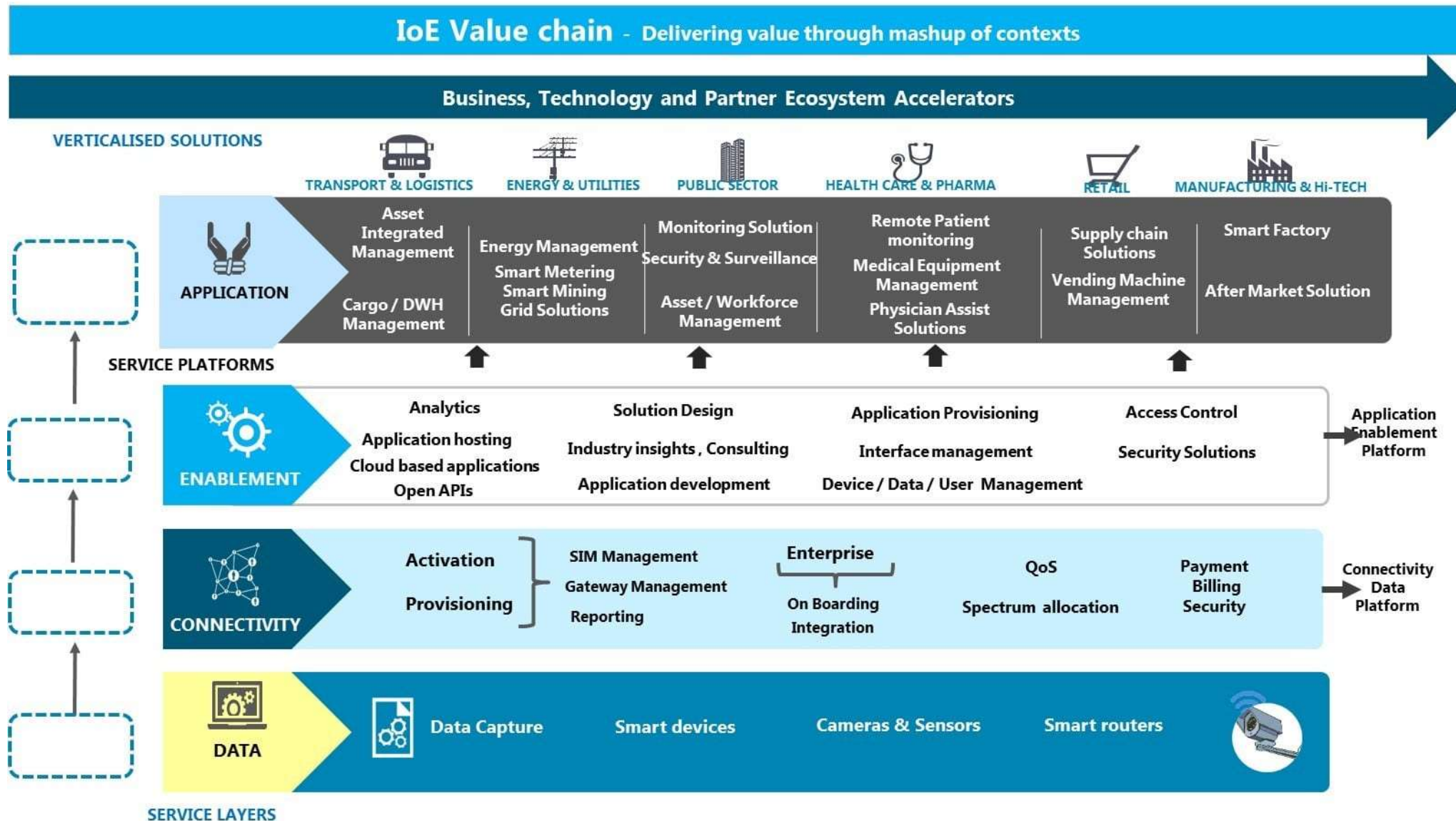
[https://www.cisco.com/c/dam/en\\_us/about/ac79/docs/innov/IoT\\_I\\_BSG\\_0411FINAL.pdf](https://www.cisco.com/c/dam/en_us/about/ac79/docs/innov/IoT_I_BSG_0411FINAL.pdf)



<https://www.ncta.com/whats-new/the-internet-of-things-continues-transform-the-way-we-live>



# Emerging IoT Ecosystem

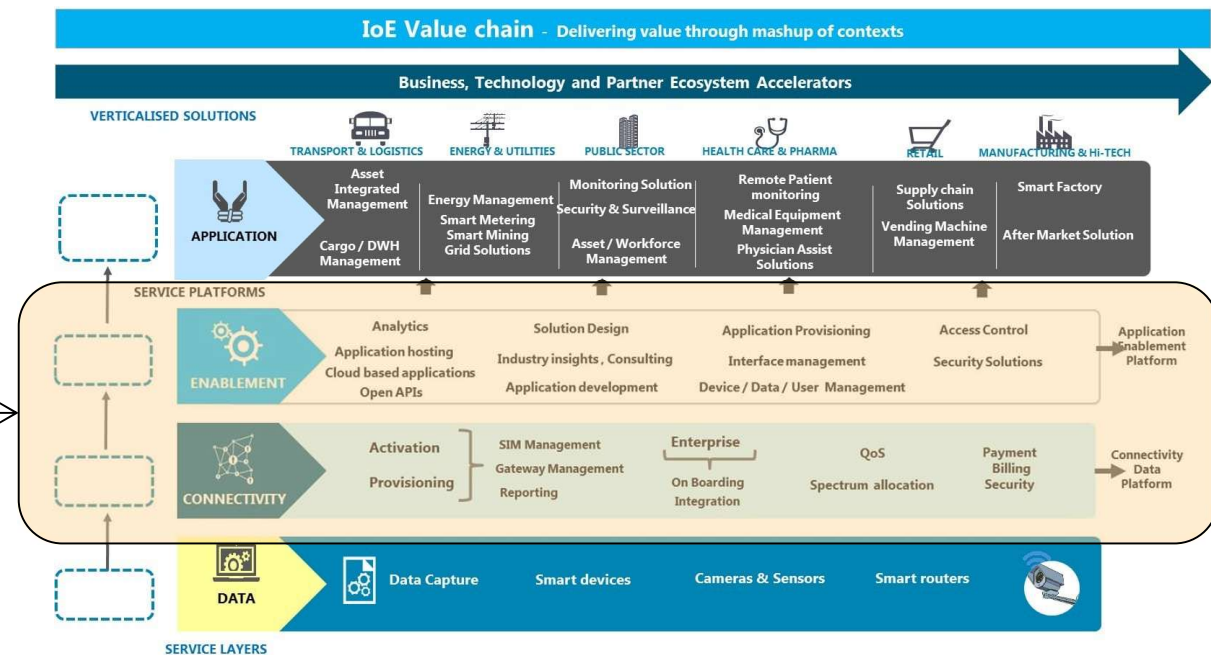
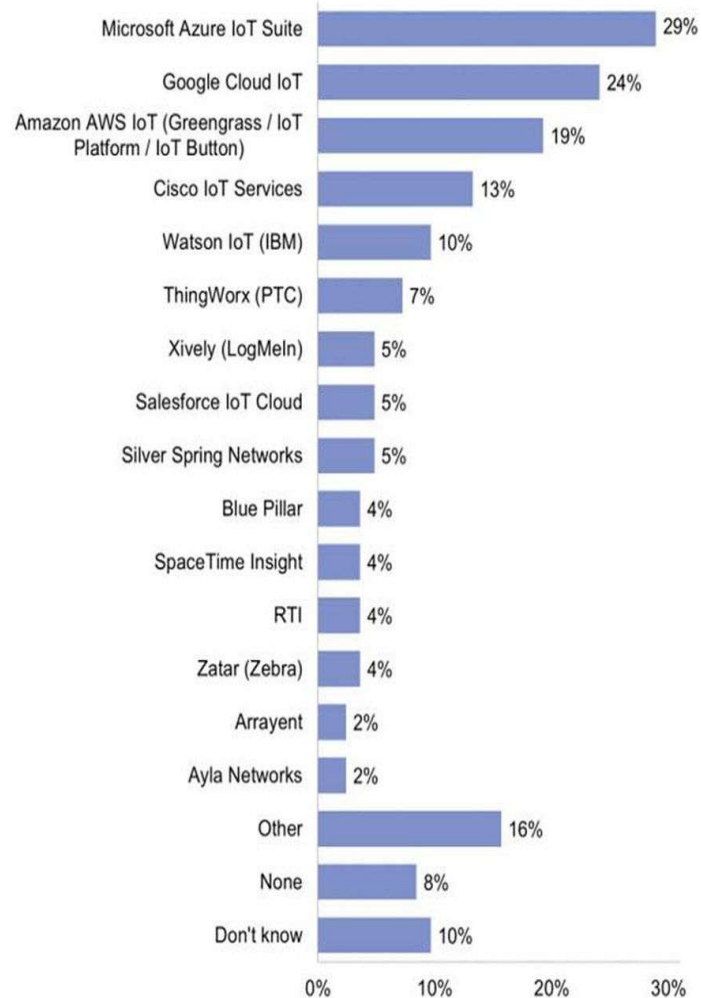


Source: TM Forum Internet of Everything



# IoT platforms - Big Players

Q: What technologies do you use to run connected devices in your software projects? Please select all that apply.



- <https://www.forbes.com/sites/louiscolumbus/2018/06/06/10-charts-that-will-challenge-your-perspective-of-iots-growth/#2883e1fa3ecc>
- Cowen Software Developer Survey, Getting In On The Ground Floor: Surveying Software Developers On Key Tech Trends, September 18, 2017



# Other Open Source IoT Platform Stacks

## Eclipse OM2M

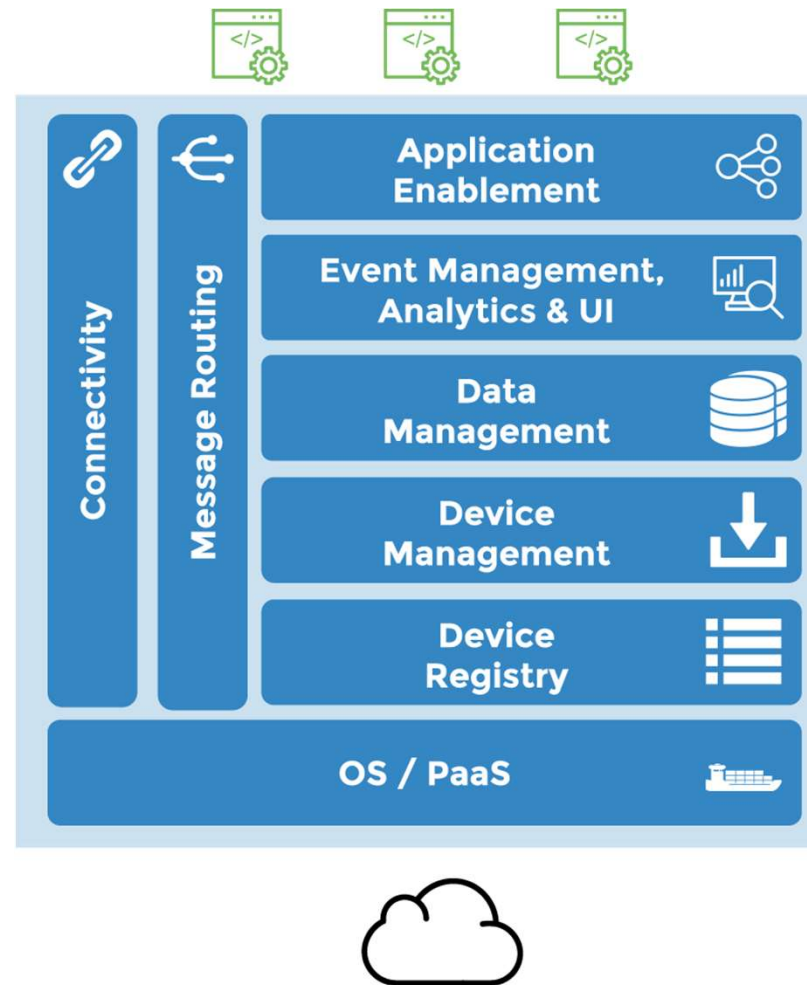
is an IoT Platform specific for the telecommunication industry, based on the oneM2M specification.

<https://www.eclipse.org/om2m/>

## Eclipse Kapua

is a modular platform providing the services required to manage IoT gateways and smart edge devices. Kapua provides a core integration framework and an initial set of core IoT services including a device registry, device management services, messaging services, data management, and application enablement.

<https://www.eclipse.org/kapua/>



## IOT CLOUD PLATFORM

Complex and difficult to deploy single purpose platforms



# Where could OpenNMS fit?

- **1. A solution for IoT Systems Monitoring**
  - Already monitoring large IoT Systems
    - Sensus – wireless IoT networks in UK
- **2. OpenNMS as a Platform for IoT Applications**

## VERTICALISED SOLUTIONS



APPLICATION

## SERVICE PLATFORMS



ENABLEMENT



CONNECTIVITY



DATA

## SERVICE LAYERS

Applications  
Visualisation



Data Storage Processing  
Alerting  
Analytics Security



Connectivity  
Wifi / LTE / 5G /  
LoRaWan



Devices  
(Device Platforms)



- Big Data IoT
- 11M smart electric/gas meters UK
- 100,000 Smart Water Meters London (Thames Water)

- Long Tail of Applications with a relatively small numbers of sensors

OpenNMS could find a role in the long tail of applications proven and (relatively) simple to deploy solution



# Contents

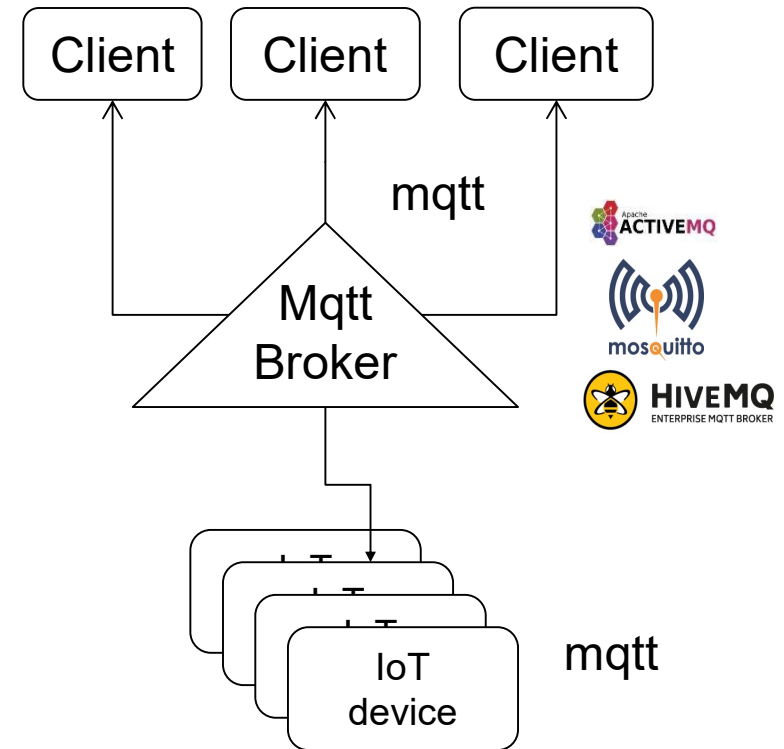
- General discussion of IoT
- **MQTT**
- OpenNMS IoT Plugin
- Use Cases





# The IoT protocol MQTT

- **MQTT (Message Queuing Telemetry Transport)**
  - Andy Stanford-Clark of IBM and Arlen Nipper of Cirrus Link authored the first version of the protocol in 1999
- **OASIS Standard MQTT v3.1.1**
  - <http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html>
  - ISO standard (ISO/IEC PRF 20922)
- **Design criteria**
  - It is designed for connections with remote locations where a "small code footprint" is required or the network bandwidth is limited.
  - Can work on top of LaRaWan, Zigbee, Wifi etc
- **Details**
  - A publish-subscribe-based messaging protocol requiring a broker
  - Built on TCP/IP and can use TLS (secure sockets).
  - Application Message : Topic, Qos, Payload





# MQTT Message

- **MQTT Message contains <Client Id>< Topic >< QoS (0,1,2) >< Payload >**
- **MQTT QoS**
  - QoS 0: At most once delivery
    - The message is delivered according to the capabilities of the underlying network. No response is sent by the receiver and no retry is performed by the sender. The message arrives at the receiver either once or not at all.
  - QoS 1: At least once delivery
    - This quality of service ensures that the message arrives at the receiver at least once. ( But may receive duplicates)
  - QoS 2: Exactly once delivery
    - This is the highest quality of service, for use when neither loss nor duplication of messages are acceptable
- **Client Identification and authentication**
  - MQTT Client id
  - Username, Password and / or Certificate (e.g. Amazon IoT)



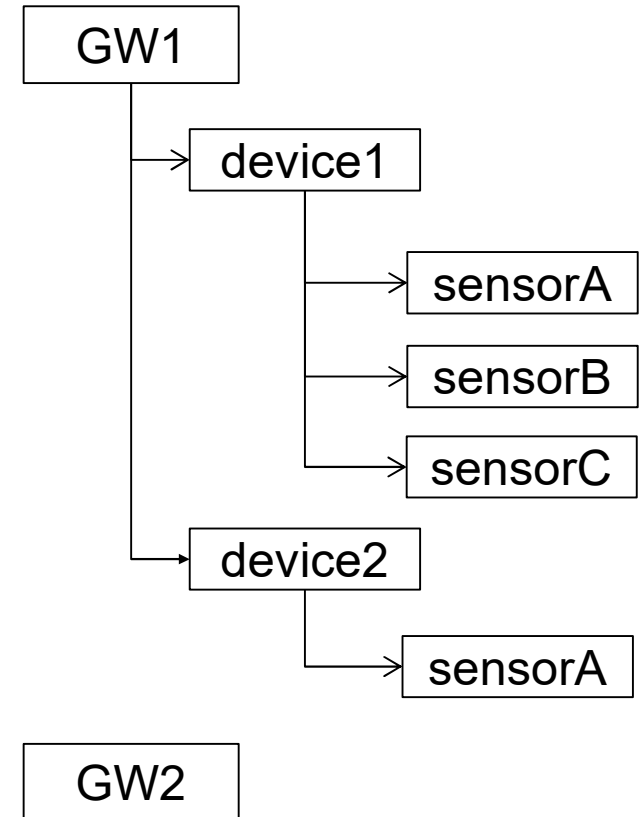
# MQTT Topic

- **Topic Name:**

- The label attached to an Application Message which is matched against the Subscriptions known to the Server. The Server sends a copy of the Application Message to each Client that has a matching Subscription.
- Topics follow a hierarchy defined by a Topic level separator '/' e.g **GW1/device1/sensorA** etc

- **Topic Filter:**

- An expression contained in a Subscription, to indicate an interest in one or more topics. A Topic Filter can include wildcard characters.
  - Multi-level wildcard #
  - Single level wildcard +
- Examples
  - **GW2/#** selects all messages from GW2
  - **GW1/device1/+** selects all messages from any sensor under device 1
  - **GW1/+sensorA** selects all sensorA messages from any device under GW1





# Contents

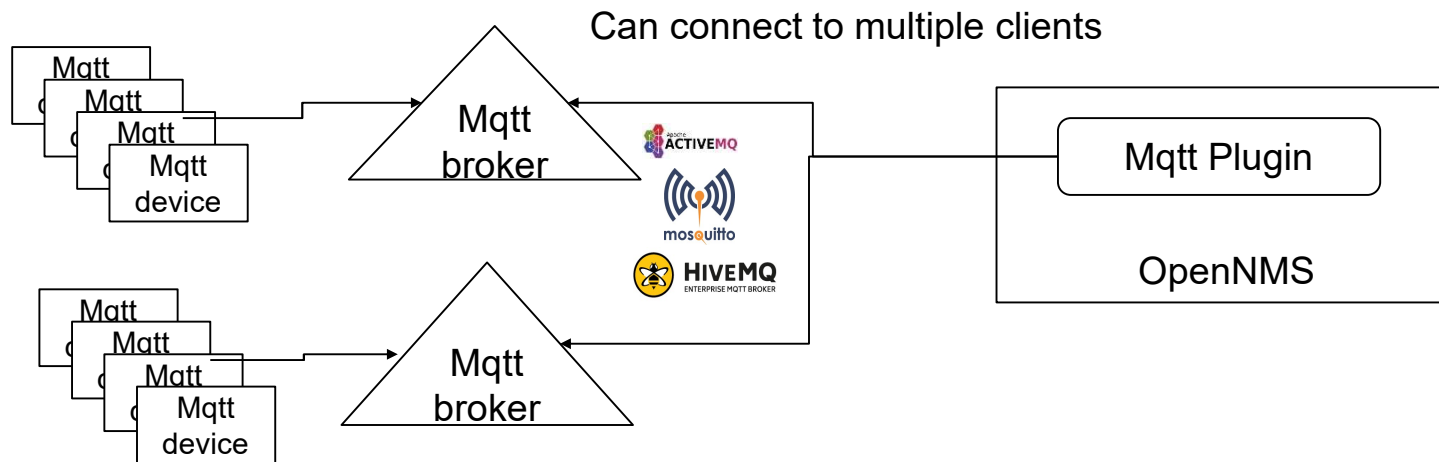
- General discussion of IoT
- MQTT
- **OpenNMS IoT Plugin**
- Use Cases





# OpenNMS MQTT Plugin

- **Stand alone OSGi plugin which drops into OpenNMS 22**



- **Code : <https://github.com/gallenc/opennms-mqtt-plugin>**
- **Installation**
  - mvn clean install
  - drop opennms-mqtt-plugin/kar-package/target//opennms-mqtt-plugin.kar into <OpenNMS>/deploy
  - Add <opennms>/etc/mqtt-interface-config.xml
    - Example config file
    - <https://github.com/gallenc/opennms-mqtt-plugin/blob/master/main-module/mqtt-interface-config.xml>
    - Example configurations for message parsing in unit tests
    - <https://github.com/gallenc/opennms-mqtt-plugin/tree/master/main-module/src/test/resources>
  - There is a docker compose project with complete configuration opennms, cassandra, helm and mqtt plugin

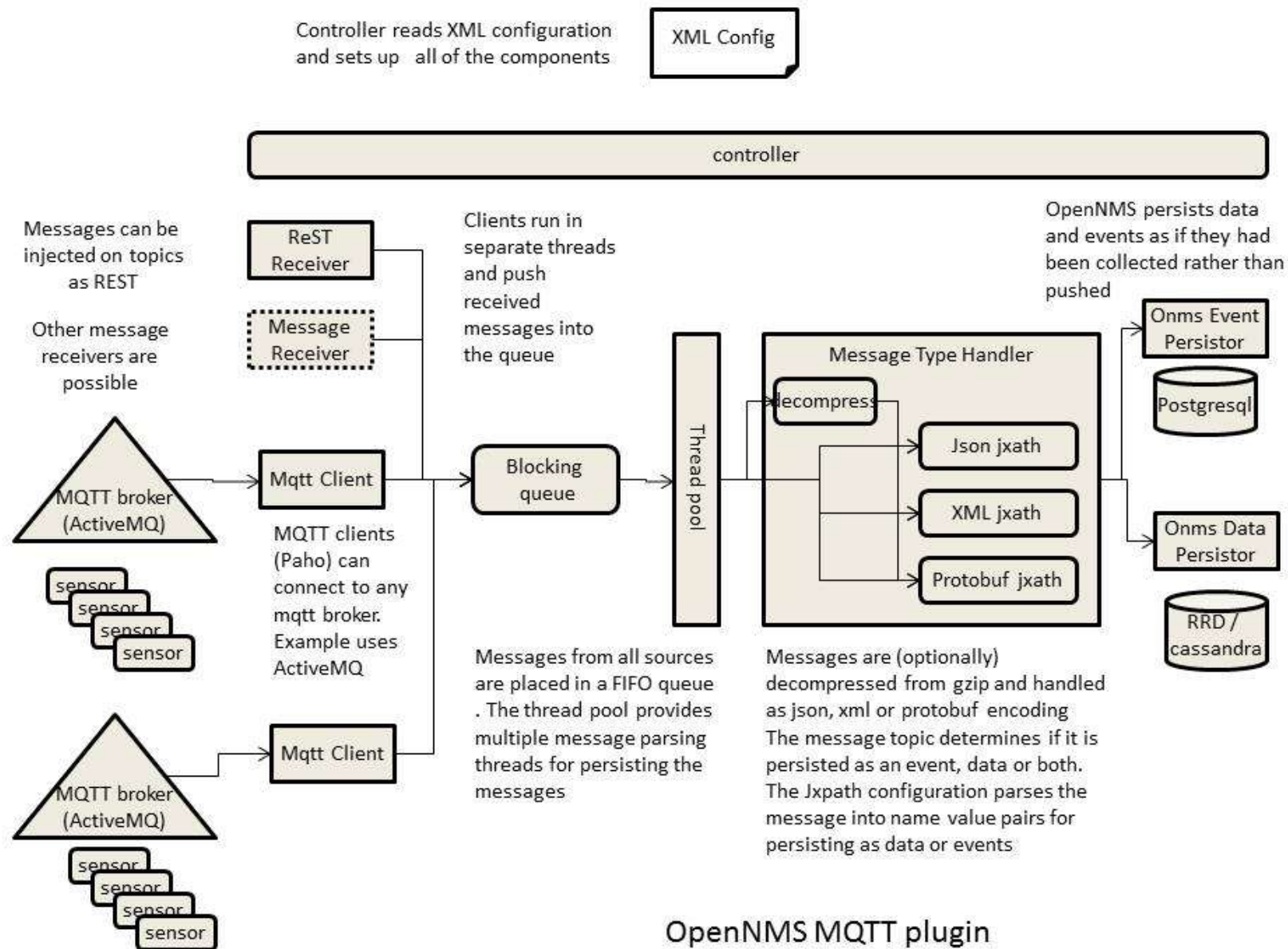


# Summary – OpenNMS MQTT

- **Simple to install plugin – just drop into <deploy> directory and add config file**
- **Can connect (and reconnect) to multiple MQTT Brokers**
  - Username / Password, TLS authentication
- **Can automatically add Nodes discovered through MQTT**
  - With foreign source and foreign id from Topic path or payload data
  - Add GIS coordinates to inventory
- **Can subscribe to filtered topics**
  - PMc178e3c2.6637eiot-2/type/mosquitto/id/00-08-00-4A-4F-F6/evt/datapoint/fmt/json
  - PMc178e3c2.6637eiot-2/type/mosquitto/id/#
- **Can choose how to handle filtered topics with different configurations**
  - Events or Data or Both
  - PMc178e3c2.6637eiot-2/type/mosquitto/id/+ /evt/datapoint/fmt/jso
  - PMc178e3c2.6637eiot-2/type/mosquitto/id/+ /evt/events/fmt/jso
- **Can parse multiple payloads**
  - Compressed, uncompressed, JSON / XML / CSV (Future Protobuf)



# How it works





# Basic Structure of Configuration

`<opennms>/etc/ mqtt-interface-config.xml`

`<mqtt-receiver >`

`<createMissingNodes>true</createMissingNodes>`

if true new nodes will be created for messages referencing an unknown node

`<createDummyInterfaces>true</createDummyInterfaces>`

if true a dummy interface will also be created when the node is created

`<createNodeAssetData>true</createNodeAssetData>`

`<messageDataParsers>`

`<messageDataParser>`

....

`<messageDataParser>`

`</messageDataParsers>`

if true asset information will be copied to the node asset table when the node is created. this particularly applies to latitude and longitude if defined in the message

`<messageEventParsers>`

.....

`</messageEventParsers>`

configuration for data parsers – many parsers can be defined for different topic filters

`<mqttClients>`

....

`</mqttClients>`

configuration for event parsers – many parsers can be defined for different topic filters

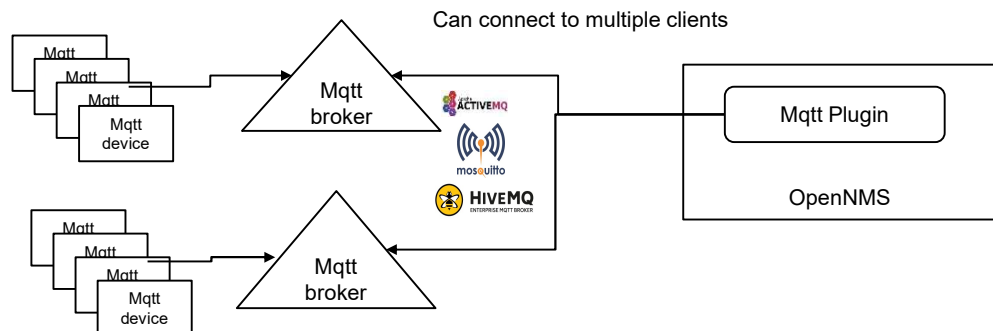
`</mqtt-receiver>`

configuration for clients. Many clients can be defined which connect to multiple topic filters

See <https://github.com/gallenc/opennms-mqtt-plugin/blob/ba2138bf8aa9d731dbbb5ede7bed440b0ee0256f/mqtt-docker-compose/etc-overlay/mqtt-interface-config.xml>



# Setting up the connection



```
<mqttClients>
<client clientInstanceId="client1">
  <brokerUrl>tcp://localhost:1883</brokerUrl>
  <clientId>opennms</clientId>
  <connectionRetryInterval>3000</connectionRetryInterval>
  <clientConnectionMaxWait>20000</clientConnectionMaxWait>
  <topicList>
    <topic qos="0" topic="mqtt-events/#" />
    <topic qos="0" topic="mqtt-data" />
  </topicList>
  <userName>mqtt-user</userName>
  <password>mqtt-password</password>
```

```
<certificateFile>put absolute path to certificate.pem file here</certificateFile>
<privateKeyFile>put absolute path to private.key file here</privateKeyFile>
<brokerUrl>ssl://{iot end point}:8883</brokerUrl>
```

```
</client>
</mqttClients>
```

- The **clientInstanceId** identifies the client in OpenNMS and must be unique
- the **clientId** must be unique for a given mqtt broker
  - the `clientConnectionMaxWait` determines the time the client will wait for the broker to respond before abandoning a connection attempt
  - the `connectionRetryInterval` is the interval before the client will try again to connect after failing to connect previously
- the topic list determines which topics the client will subscribe to once it has established a connection
- The **subscribed topic** can also be a filter e.g. `topic="/+/foo/#"`
- If you are using TLS you must provide both certificate and key files. If both are omitted from the configuration TLS will not be used
- with AMAZON IoT, the certificate and key files can be generated for you. These are tied to a specific client id and IoT endpoint



# Message Data parsers

**message data parsers** define how data messages are converted to metrics to be stored in RRD's or Cassandra by OpenNMS

**subscriptionTopics** defines which topic messages will be passed to this parser. Can be a filter  
many messageDataParsers can be defined but only one per topic although events can also be driven off the same topic)

**payloadType** determines how the payload is parsed; JSON, XML, PROTOBUF or TEXT.

**foreignSource** defines the foreign source to be used when referencing or creating a new node

compression determines the processing of compressed messages GZIP, UNCOMPRESSED, AUTOMATIC\_GZIP  
(tries to guess if gzip compressed - small chance of false positives)

**<messageDataParsers>**

**<messageDataParser foreignSource="mqtt" payloadType="JSON" compression="UNCOMPRESSED">**

**<subscriptionTopics>**

**<topic>mqtt-data</topic>**

to match data topic PMc178e3c2.6637eiot-2/type/mosquitto/id/00-08-00-4A-4F-F6/evt/datapoint/fmt/json

**<topic>PMc178e3c2.6637eiot-2/type/mosquitto/id/+ /evt/datapoint/fmt/json</topic>**

**</subscriptionTopics>**

**<xml-groups>**

**</xml-groups>**

**</messageDataParser>**

**</messageDataParsers>**

**Similar JXPath configuration to Alejandro's XML  
data collector**



# Example Complex Payload

```
{
  "zones": [ {
    "zone": {
      "id": 0,
      "name": "global",
      "timestamp": 1299258888,
      "parameter": [
        { "key": "nproc", "value": "245" },
        { "key": "nlwp", "value": "1455" },
        { "key": "pr_size", "value": "2646864" },
        { "key": "pr_rssize", "value": "1851072" },
        { "key": "pctmem", "value": "0.7" },
        { "key": "pctcpu", "value": "0.24" }
      ]
    }
  },
  { "zone": {
    "id": 871,
    "name": "zone1",
  }
} ]
```

**Note each zone in array generates own data set**

```
<?xml version="1.0" encoding="UTF-8" ?>
<xml-groups xmlns="http://xmlns.opennms.org/xsd/config/xml-datacollection">
  <xml-group name="solaris-zone-stats" resource-type="solarisZone"
    resource-xpath="/zones/zone" key-xpath="@name" timestamp-xpath="@timestamp"
    timestamp-format="">
    empty format indicates simple timestamp from epoch (Date.toString())
    <xml-object name="zoneName" type="string" xpath="@name" />
    <xml-object name="timestamp" type="string" xpath="@timestamp" />
    <xml-object name="nproc" type="GAUGE" xpath="parameter[@key='nproc']/@value" />
    <xml-object name="nlwp" type="GAUGE" xpath="parameter[@key='nlwp']/@value" />
    <xml-object name="pr_size" type="GAUGE" xpath="parameter[@key='pr_size']/@value" />
    <xml-object name="pr_rssize" type="GAUGE" xpath="parameter[@key='pr_rssize']/@value" />
    <xml-object name="pctmem" type="GAUGE" xpath="parameter[@key='pctmem']/@value" />
    <xml-object name="pctcpu" type="GAUGE" xpath="parameter[@key='pctcpu']/@value" />
  </xml-group>
</xml-groups>
```



# Extracting Node Id from Topic path

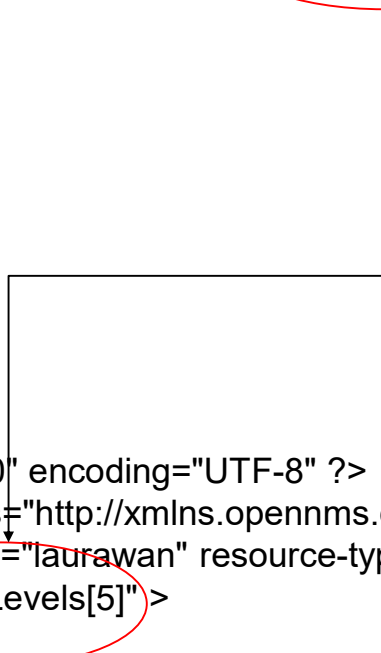
Example Laurawan MQTT payload Herb Garcia Insitu Systems, Inc.

note \$topicLevels[5] == 00-08-00-4A-4F-F6 (note starts at 1)

6/15/2018, 6:25:59 PMc178e3c2.6637eiot-2/type/mosquitto/id/00-08-00-4A-4F-F6/evt/datapoint/fmt/json

: msg.payload : Object

```
{
  "d": {
    "light": 462,
    "moisture": 97.34025,
    "temperature": 29.8125,
    "x_acc": 0.1875,
    "y_acc": 0.375,
    "z_acc": 0.8125
  }
}
```



```
<?xml version="1.0" encoding="UTF-8" ?>
<xml-groups xmlns="http://xmlns.opennms.org/xsd/config/xml-datacollection">
  <xml-group name="laurawan" resource-type="laurawan" resource-xpath="/*"
key-xpath="$topicLevels[5]">
    <xml-object name="light" type="GAUGE" xpath="@light" />
    <xml-object name="moisture" type="GAUGE" xpath="@moisture" />
    <xml-object name="temperature" type="GAUGE" xpath="@temperature" />
    <xml-object name="x_acc" type="GAUGE" xpath="@x_acc" />
    <xml-object name="y_acc" type="GAUGE" xpath="@y_acc" />
    <xml-object name="z_acc" type="GAUGE" xpath="@z_acc" />
  </xml-group>
</xml-groups>
```



# Message Event Handling

message event parsers follow the same pattern as message data parsers.

however the Xpath defines which name value pairs are saved as params in the resulting opennms events

```
<messageEventParsers>
  <messageEventParser foreignSource="$topicLevels[5]" payloadType="JSON" compression="UNCOMPRESSED">
    <subscriptionTopics>
      <topic>iot/type/mosquito/id/+/device/+/app/opennms/fmt/json</topic>
    </subscriptionTopics>
```

**It is possible to specify a specific uri to associate with this event  
if ueiRoot is not specified then a default generic value is used**

```
<ueiRoot>uei.opennms.org/plugin/MqttReceiver/laurawan</ueiRoot>
```

```
<ns2:xml-groups xmlns="http://xmlns.opennms.org/xsd/config/xml-datacollection">
  <ns2:xml-group name="laurawan" resource-type="laurawan" resource-xpath="/metrics" key-xpath="$topicLevels[7]"
    timestamp-xpath="@timestamp" timestamp-format="yyyy-MM-dd HH:mm:ss.SSSSSS">
    <ns2:xml-object name="gweui" type="string" xpath="@gweui" />
    <ns2:xml-object name="deveui" type="string" xpath="@deveui" />
```

...

**if ueiRoot is specified and in addition, the xml-group contains a definition labeled "UEI\_SUFFIX"  
then the value UEI\_SUFFIX parsed from the message context will be appended to the ueiRoot  
to form a message specific uei. Thus if the following line is added**

```
<ns2:xml-object name="UEI_SUFFIX" type="string" xpath="@deveui" />
```

**the generated event uei will be**

**uei.opennms.org/plugin/MqttReceiver/laurawan/{@deveui value}**

```
</ns2:xml-group>
</ns2:xml-groups>
</messageEventParser>
</messageEventParsers>
```



# Contents

- General discussion of IoT
- MQTT
- OpenNMS IoT Plugin
- **Use Cases**





# Case Study – Monitoring Tailings

- Herb Garcea Insitu Systems, Inc.
- [www.insitusys.com](http://www.insitusys.com)



- **Mine tailings**
  - are the ore waste of mines, and are typically a mud-like material. Worldwide, the storage and handling of tailings is a major environmental issue. Many tailings are toxic and must be kept perpetually isolated from the environment.
  - <https://www.nps.gov/articles/aps-v13-i2-c8.htm>
  - Series: Alaska Park Science - Volume 13 Issue 2: Mineral and Energy Development Long-term Risk of Tailings Dam Failure





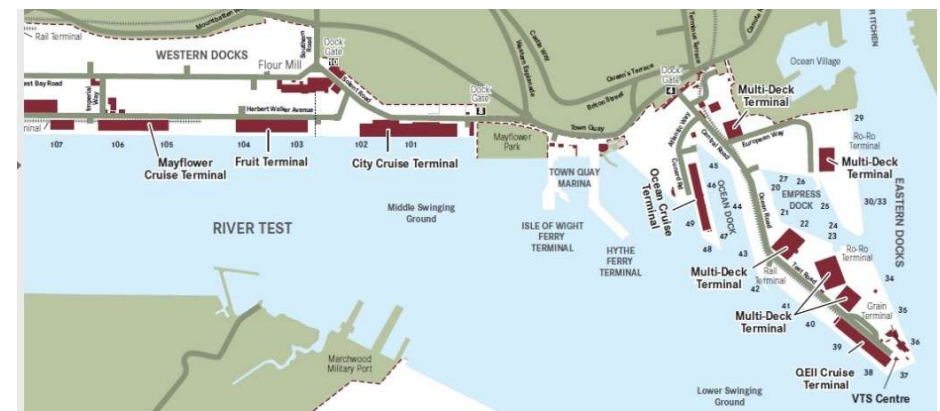
# Case Study

## Southampton Air Pollution Monitoring

- The UK's number one cruise port, which welcomes 1.7m passengers
- Each ship up to 6000 passenger and crew
- Ship turn around 1-2 days
- Increasing problem of air pollution due to generators running while ships in port



- <http://www.solentairwatch.co.uk/>
  - Community air monitoring project
  - Not for profit charity run by Southampton volunteers
  - Monitor Sniffy – version 1
    - <https://github.com/SolentAirWatch/sniffy>
    - Raspberry Pi Zero
    - Low cost laser Smoke detector
    - Ruggedized container
  - Founder Dr Joshua Taylor

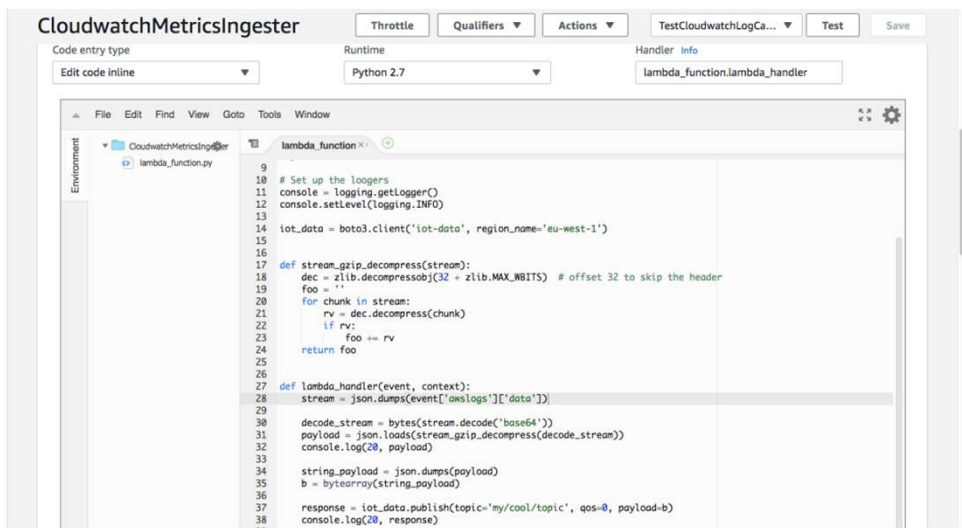
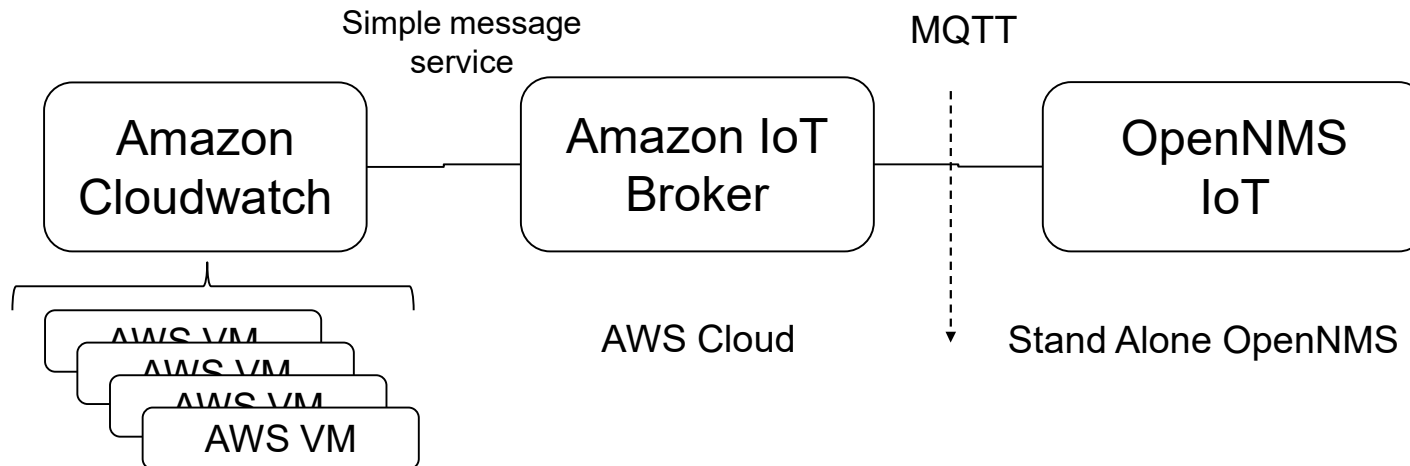




# Case Study

## Receiving Amazon CloudWatch data via Amazon IoT

With Patrick Tuit

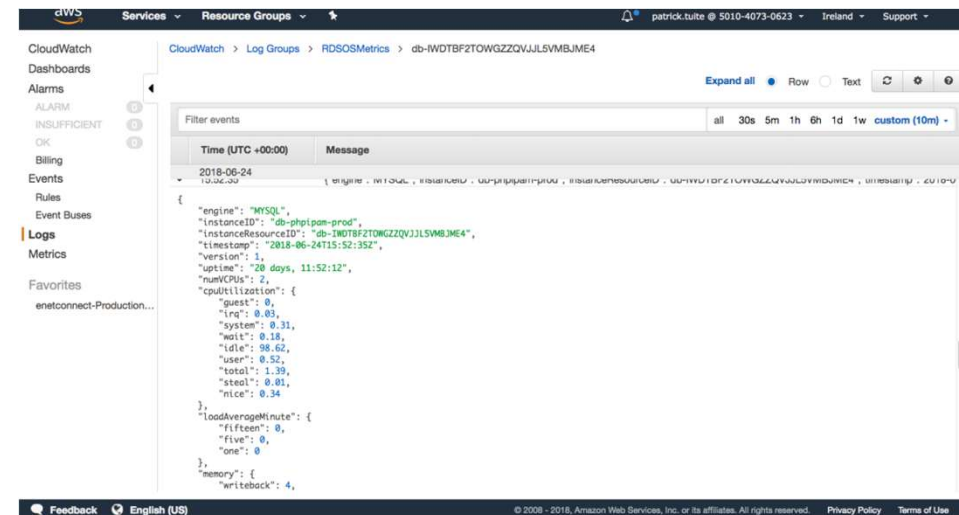


```
CloudwatchMetricsIngester
Code entry type: Edit code inline
Runtime: Python 2.7
Handler: lambda_function.lambda_handler

File Edit Find View Goto Tools Window

Environment: CloudwatchMetricsIngester
lambda_function.py

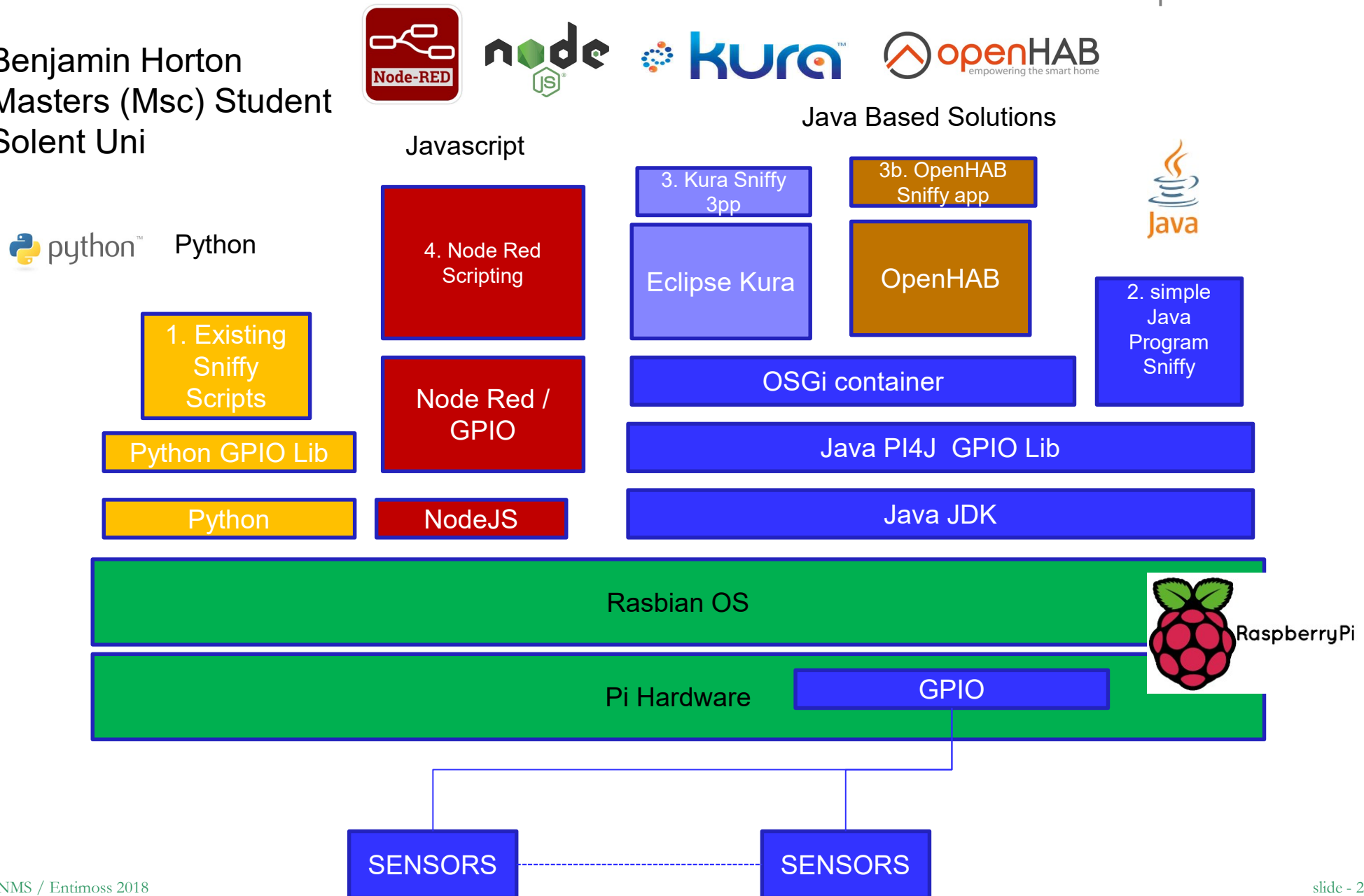
9
10 # Set up the loggers
11 console = logging.getLogger()
12 console.setLevel(logging.INFO)
13
14 iot_data = boto3.client('iot-data', region_name='eu-west-1')
15
16
17 def stream_gzip_decompress(stream):
18     dec = zlib.decompressobj(32 + zlib.MAX_WBITS) # offset 32 to skip the header
19     foo = ''
20     for chunk in stream:
21         rv = dec.decompress(chunk)
22         if rv:
23             foo += rv
24     return foo
25
26
27 def lambda_handler(event, context):
28     stream = json.dumps(event['awslogs']['data'])
29
30     decode_stream = bytes(stream.decode('base64'))
31     payload = json.loads(stream_gzip_decompress(decode_stream))
32     console.log(20, payload)
33
34     string_payload = json.dumps(payload)
35     b = bytearray(string_payload)
36
37     response = iot_data.publish(topic='my/cool/topic', qos=0, payload=b)
38     console.log(20, response)
```





# IoT Device Platforms Investigation

- Benjamin Horton  
Masters (Msc) Student  
Solent Uni





# Summary

- **OpenNMS now has a very credible IoT story**
  - Data collection – Cassandra and Elastic Search
  - Drift – streaming protocols
  - Minions
  - New - MQTT plugin
- **MQTT plugin - still a prototype but appears to work well**
  - There is a docker compose project 😊
- **Please spread the word**
- **Try the plugin yourself**
- **Give us feedback as to what to do next**



**Thanks for listening**

**Questions?**