Secure IOT Platform Using Paho & Flask

JAIMON JOSE

Agenda

About me

IoT primer and reference architecture

IoT platform design

Best practices

About Me

Jaimon Jose, Co-Founder, CTO, picoNets

- 17 Years of product development experience
- Distinguished Engineer at Novell and Core founding member at PAQS
- Continuous learner and Trainer for processes and technologies over a decade
- Engaged with academia and industries
- Patents, publications and speaking in various events
- Over a decade of experience in building solutions in identity, security, cloud, virtualization, distributed systems

@jjaimon https://in.linkedin.com/in/jjaimon http://jjaimon.net

IoT: What and Why?

Internet of Things (IoT) to Internet of Everything (IoE)

- Network of physical objects that contain embedded technology to communicate and interact with their internal states or external environment

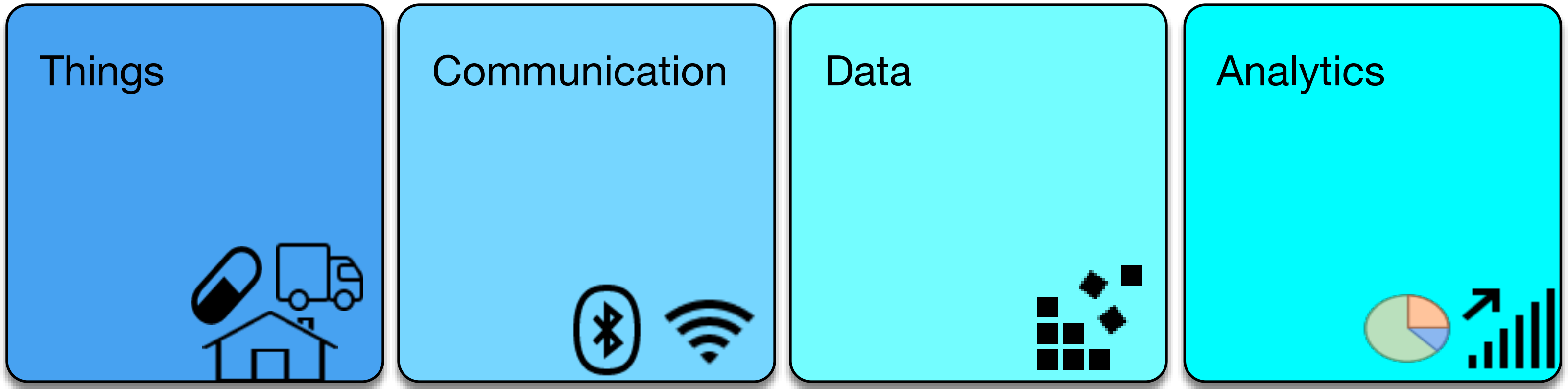
World is getting smarter

- Smarter Homes
- Smarter Logistics
- Smarter Healthcare
- Smarter Vehicles

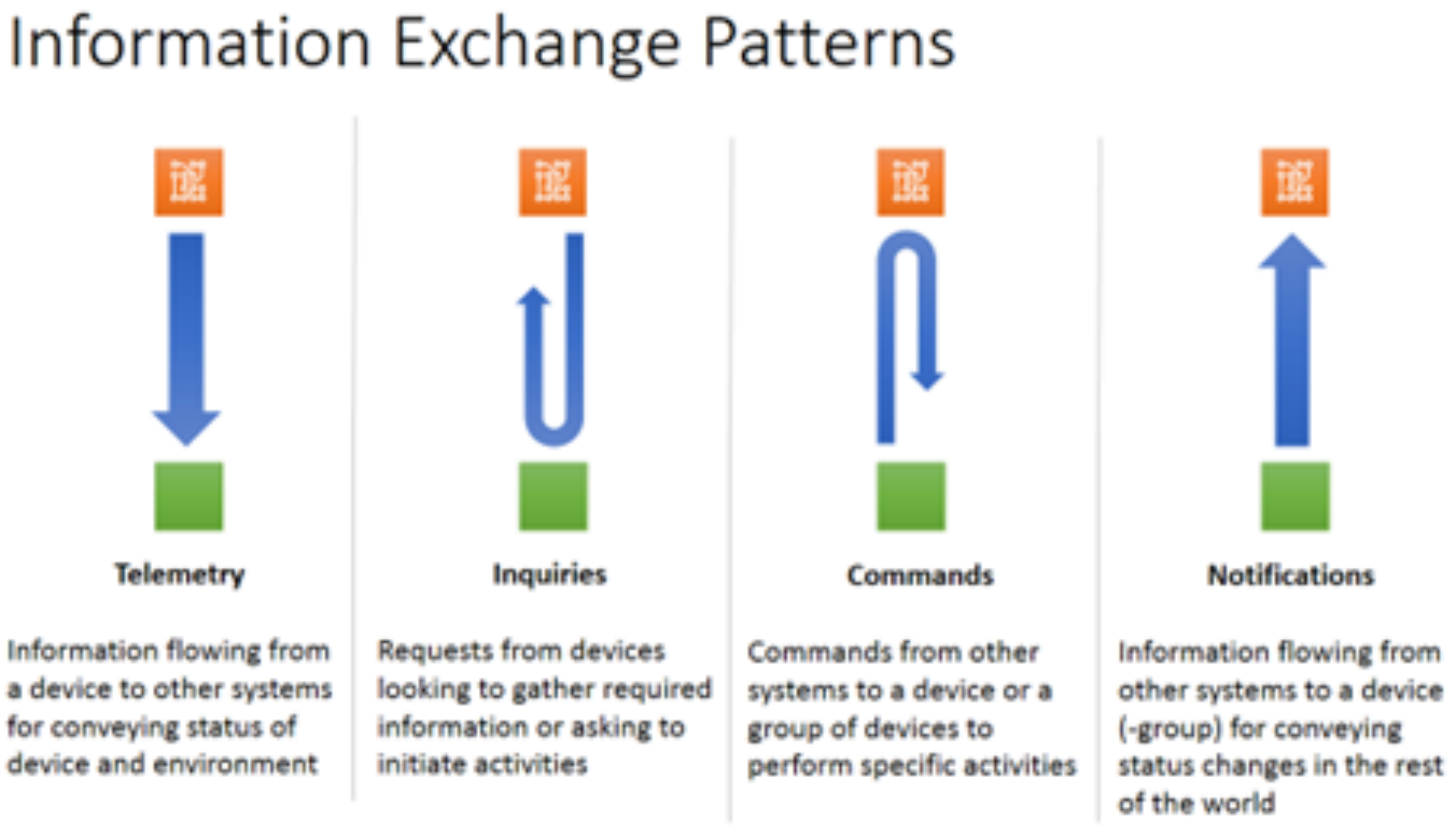
Industrial and Home solutions

My 🚗 tells my 🏠 to open the 🏠 and start ☕

What is IoT About?

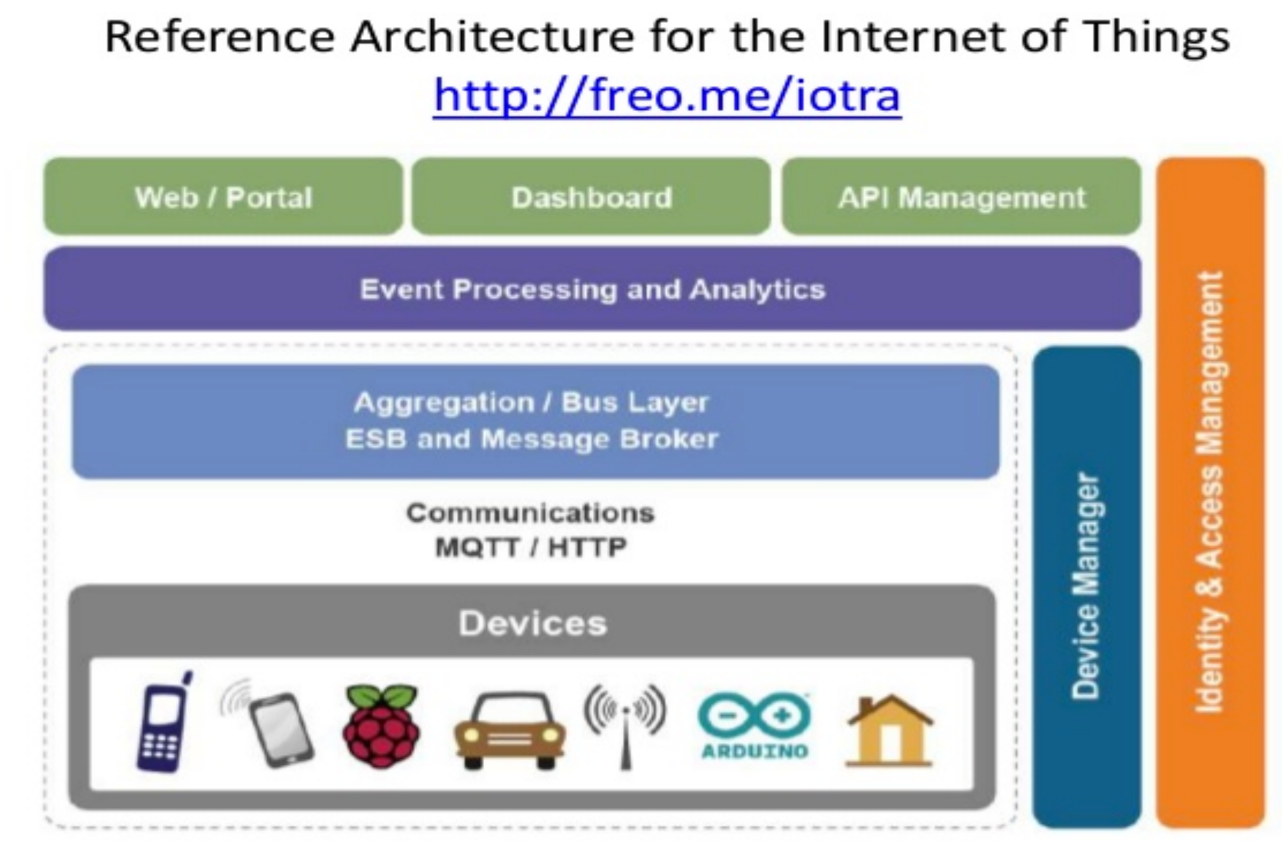


IoT Communication Patterns


Information Exchange Patterns
Telemetry
Information flowing from a device to other systems for conveying status of device and environment
Inquiries
Requests from devices looking to gather required information or asking to initiate activities
Commands
Commands from other systems to a device or a group of devices to perform specific activities
Notifications
Information flowing from other systems to a device (-group) for conveying status changes in the rest of the world

Source: https://blogs.msdn.microsoft.com/clemensv/2014/02/09/service-assisted-communication-for-connected-devices/

IoT Reference Architecture



IOT Platform – A Practical Approach

Design Rationale

Secure provisioning and bootstrapping

Minimize on-the-wire footprint

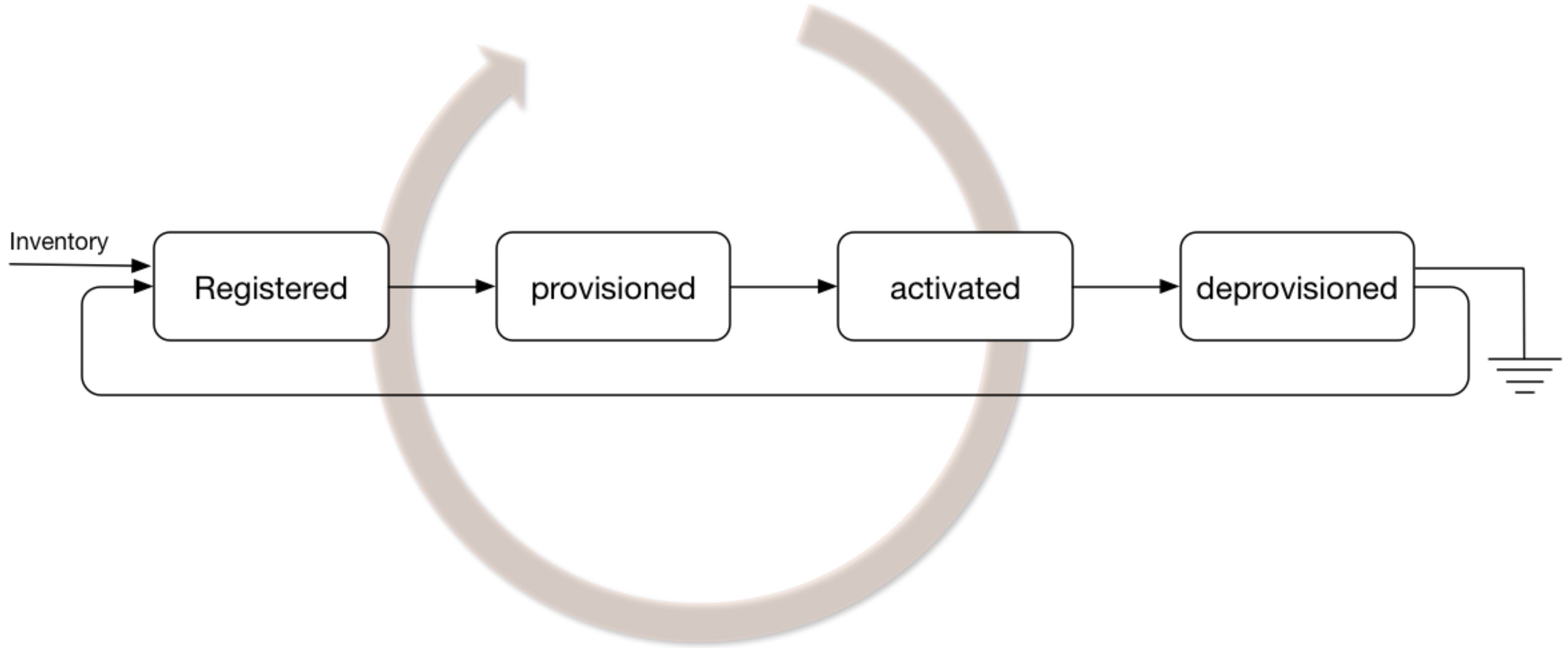
Expect network interruption. Design for resilience

Protocol support to configure the device from server

Standard and proven protocols– no reinventing

Less processing on the device – let server do the heavy lifting

Device Lifecycle



Protocols

Bootstrap on HTTPS

- Built in support for secure transport, easy to pass-through firewalls

Data on MQTT

- Developed by IBM in the late 1990s used for oil field and flood plain monitoring
- Handed over to Eclipse Foundation in 2011 as part of M2M announcement
  - IBM MQTT client code – C and Java – to new a Eclipse project “Paho”
- Simple methods – publish/subscribe/unsubscribe
- Faster than HTTP (if you compare with REST – HTTP is verbose)
- Protocol is optimized from the start for unreliable, low-bandwidth, high-latency networks
- Client must support TCP and will hold a connection to the broker at all times

Didn’t consider CoAP (relatively new)

Device Server Communication

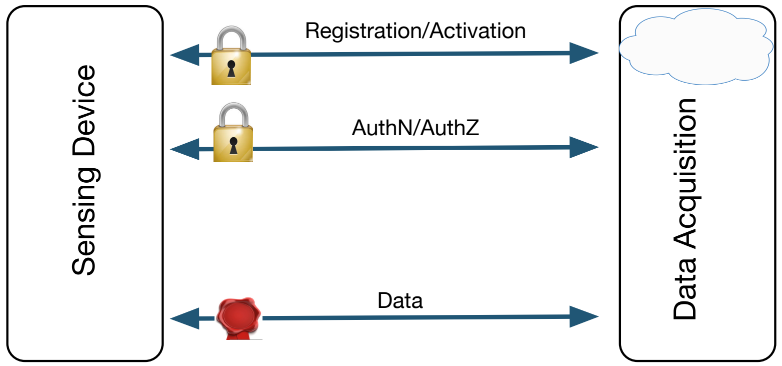
HTTPS

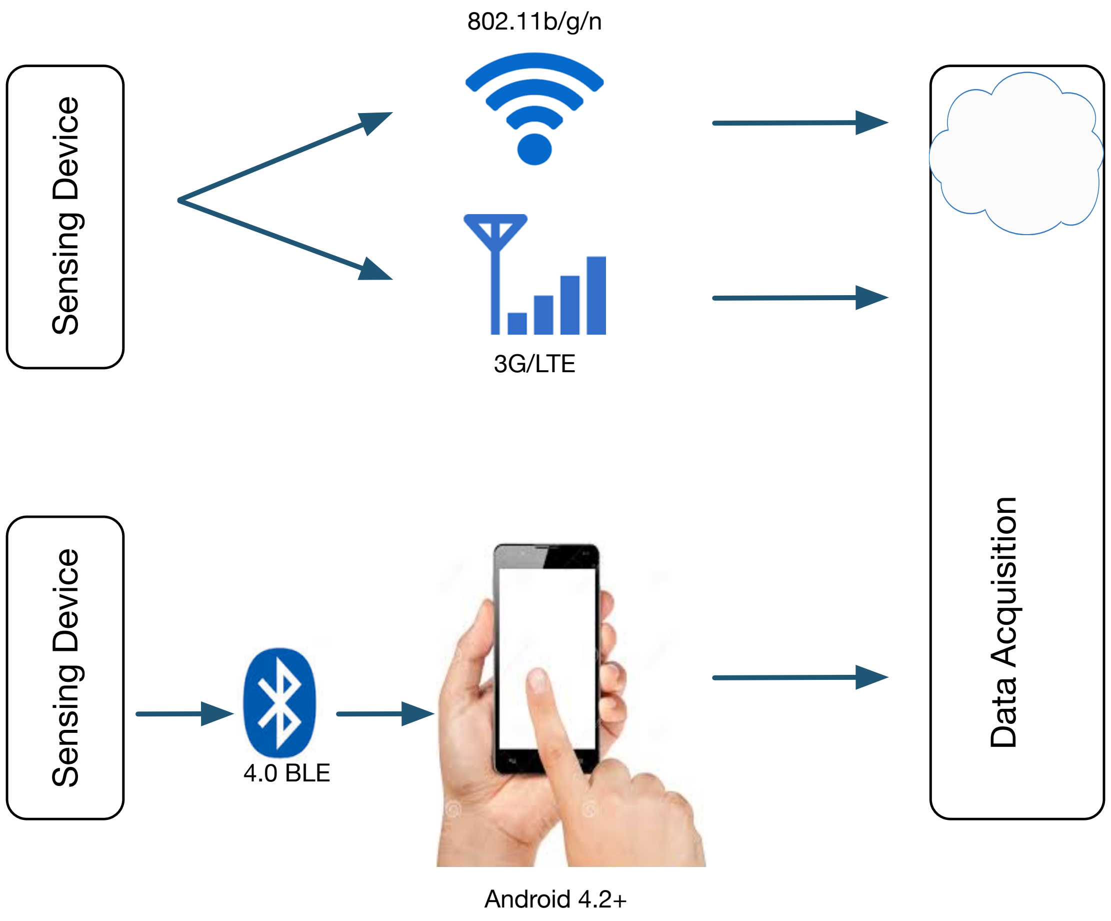
- Registration, AuthN, AuthZ and Key renewal
- Flask + Tornado

Data transfer

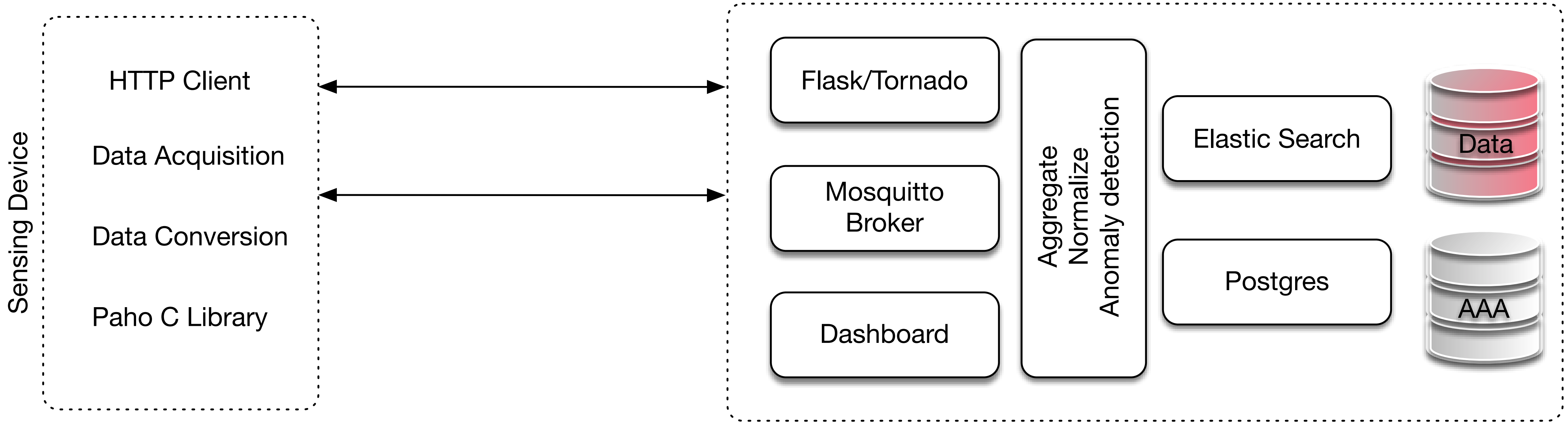
- Sensor data and errors
- Small keep alive packets to monitor device health
- Compressed and signed json payload
- mosquitto as the broker

Device – Backend Interactions

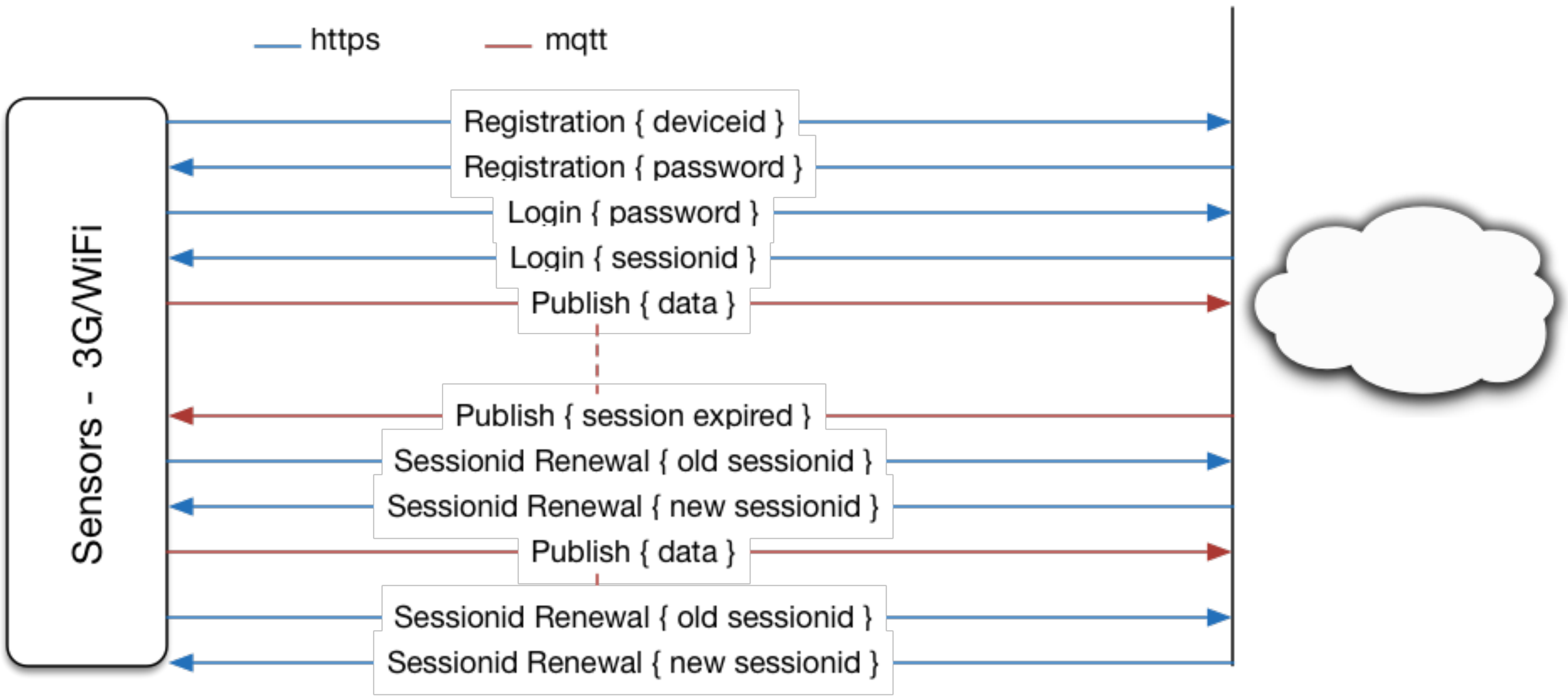




Platform View



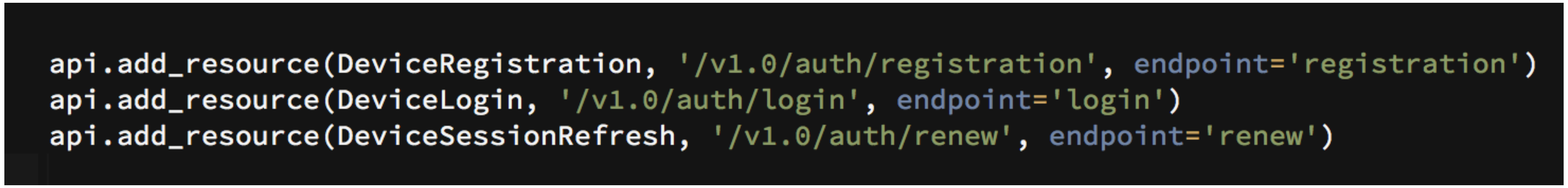
Protocol Interactions


https
mqtt
Sensors - 3G/WiFi
Registration { deviceid }
Registration { password }
Login { password }
Login { sessionid }
Publish { data }
Publish { session expired }
Sessionid Renewal { old sessionid }
Sessionid Renewal { new sessionid }
Publish { data }
Sessionid Renewal { old sessionid }
Sessionid Renewal { new sessionid }

Flask – Plugins

Flask-RESTful

- Extension for Flask that adds support for REST APIs
- Resourceful routing – ability to define methods on your resource



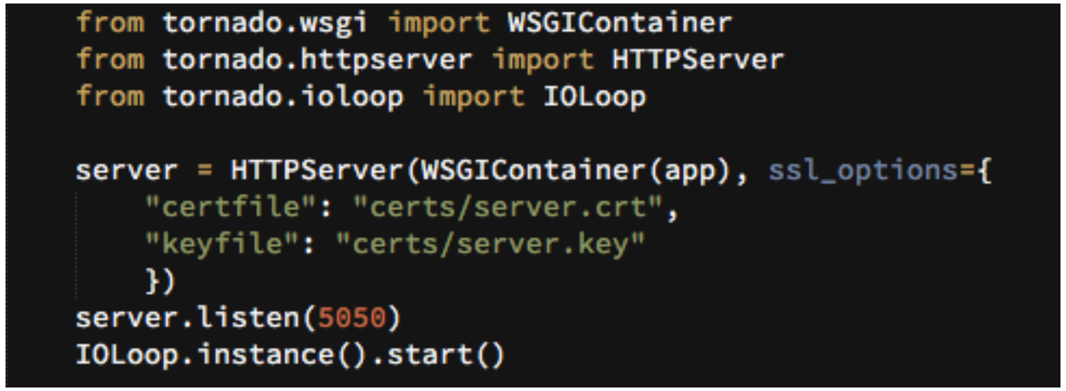
```
api.add_resource(DeviceRegistration, '/v1.0/auth/registration', endpoint='registration')
api.add_resource(DeviceLogin, '/v1.0/auth/login', endpoint='login')
api.add_resource(DeviceSessionRefresh, '/v1.0/auth/renew', endpoint='renew')
```

Flask-HTTPAuth

- Implements HTTP authentication with Flask routes
- Basic, digest and token based authentication methods
- Extensible

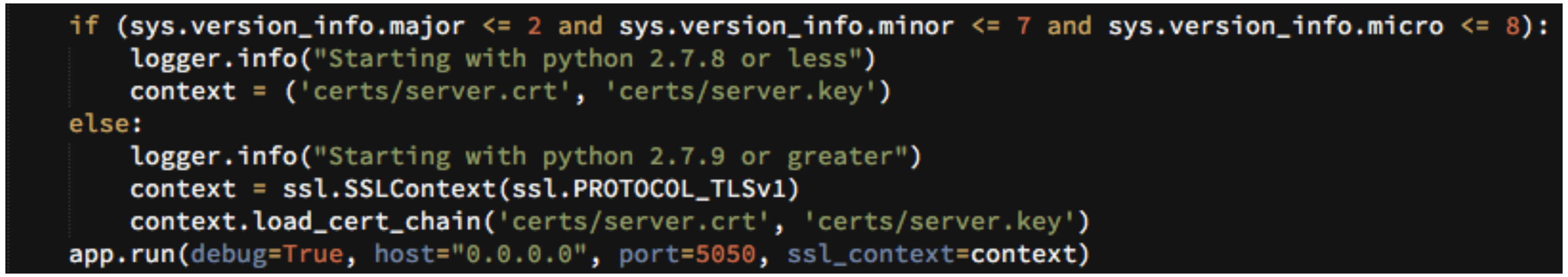
Flask – Pitfalls

Unreliable for long connections. Wrap your Flask app in a Tornado container to improve reliability



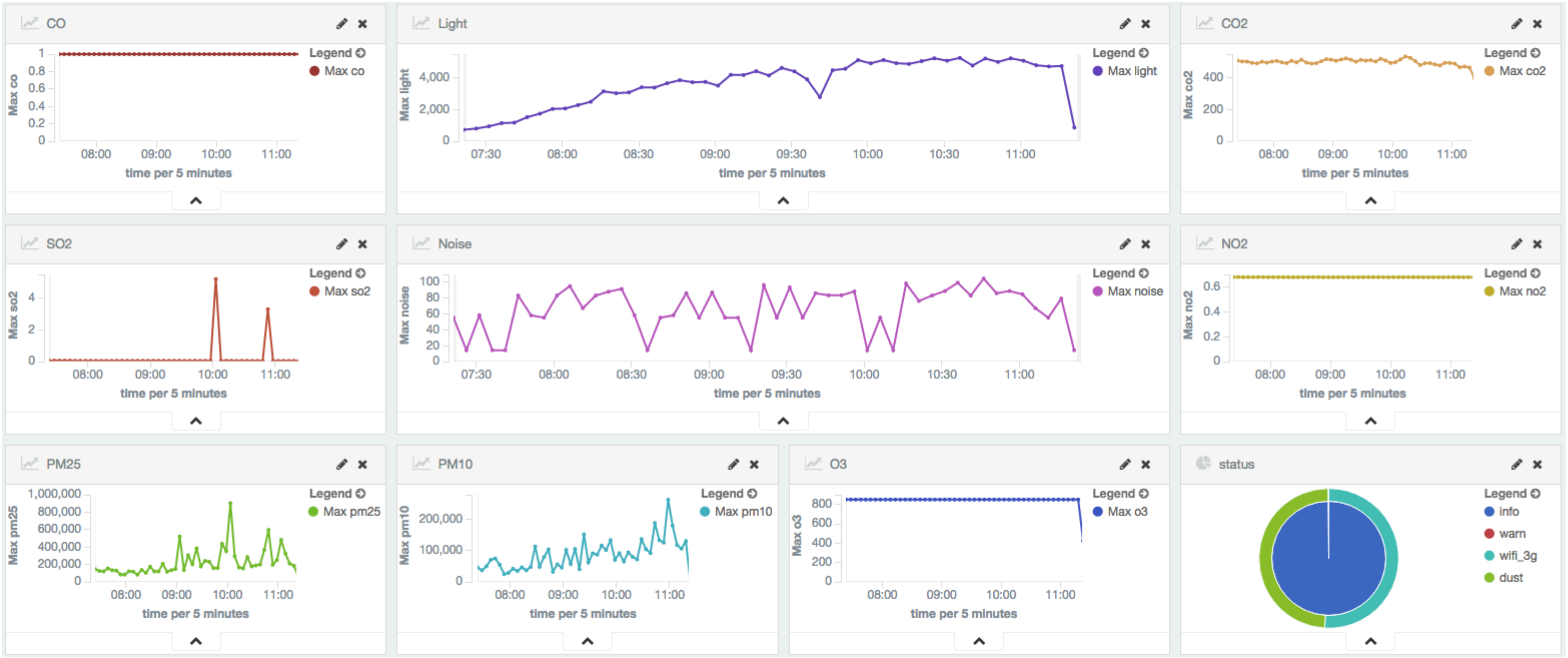
```python
from tornado.wsgi import WSGIContainer
from tornado.httpserver import HTTPServer
from tornado.ioloop import IOLoop

server = HTTPServer(WSGIContainer(app), ssl_options={
    "certfile": "certs/server.crt",
    "keyfile": "certs/server.key"
    })
server.listen(5050)
IOLoop.instance().start()
```

SSL setup is different for python 2.7 and below.



```python
if (sys.version_info.major <= 2 and sys.version_info.minor <= 7 and sys.version_info.micro <= 8):
    logger.info("Starting with python 2.7.8 or less")
    context = ('certs/server.crt', 'certs/server.key')
else:
    logger.info("Starting with python 2.7.9 or greater")
    context = ssl.SSLContext(ssl.PROTOCOL_TLSv1)
    context.load_cert_chain('certs/server.crt', 'certs/server.key')
app.run(debug=True, host="0.0.0.0", port=5050, ssl_context=context)
```

Build a Dashboard (first)



Best Practices and Caveats

Log every state change, operations. Log…Log...Log...

Microservice architecture with clearly defined interfaces for interactions.

Use standards for data. Its a good idea for the device to publish temperature in C or F instead of 12µV

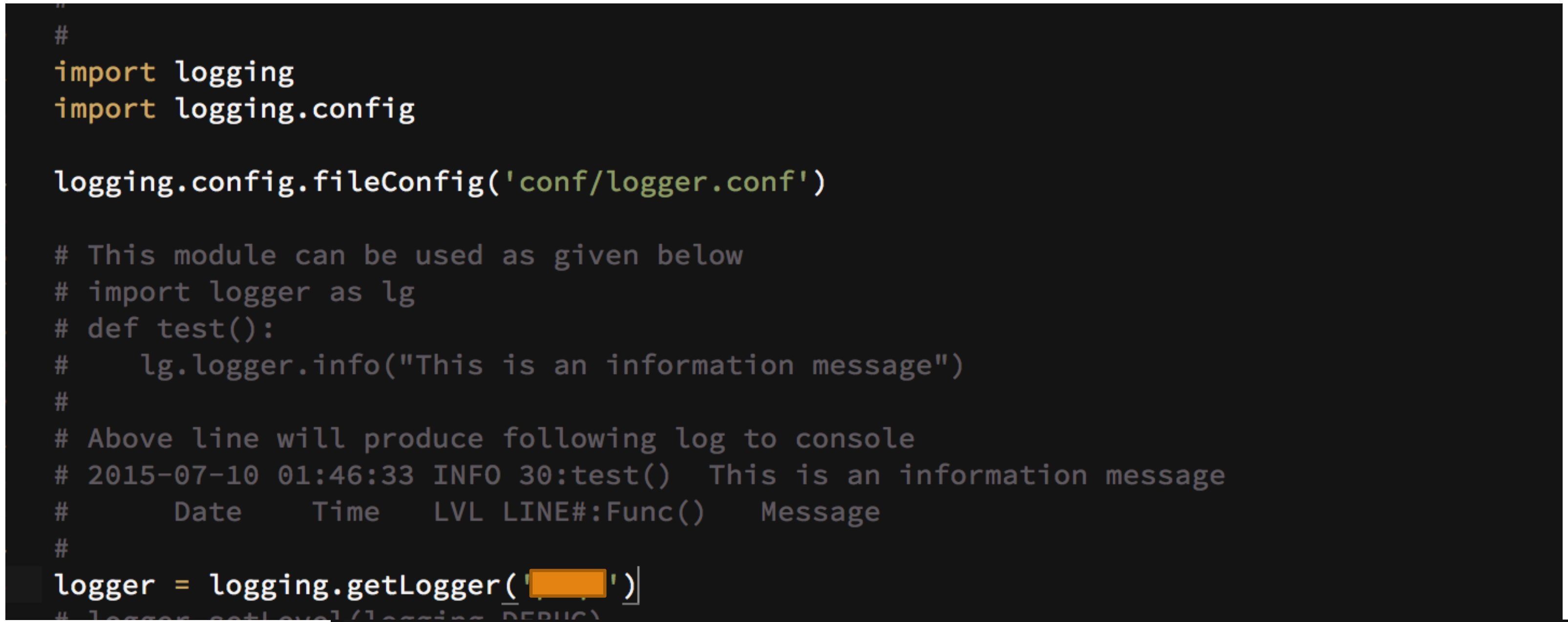
Focus on your use case – Platform will evolve

- Keep it lean and easy to change as use case evolve
- You don't need a complex platform to validate your business

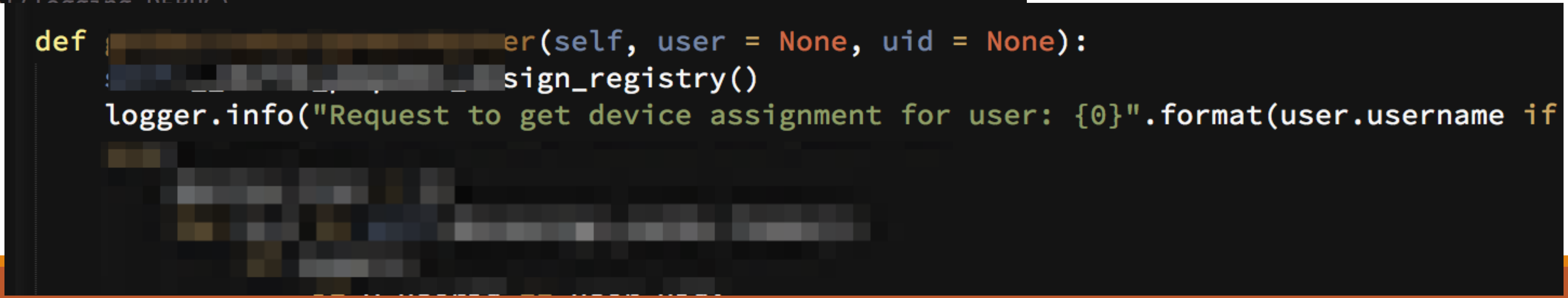
Collect as much information possible remotely for a device failure during development and field testing. You can't babysit a device in the field

Devices will fail and they'll surprise you. Validate data and handle exceptions

Logging



```
#
import logging
import logging.config

logging.config.fileConfig('conf/logger.conf')

# This module can be used as given below
# import logger as lg
# def test():
#    lg.logger.info("This is an information message")
#
# Above line will produce following log to console
# 2015-07-10 01:46:33 INFO 30:test()  This is an information message
#      Date     Time    LVL LINE#:Func()    Message
#
logger = logging.getLogger('[redacted]')
```



```
def [redacted]er(self, user = None, uid = None):
    [redacted]sign_registry()
    logger.info("Request to get device assignment for user: {0}".format(user.username if
    [redacted]
```

Questions?

@jjaimon

jaimon.jose@piconets.com