Autonomous Drone Engineer

D2 – Networked Drone

Paul.Guermonprez@intel.com
Autonomous Drone Solutions Architect

From MAVLINK Proxy to Networked Logic

In the default setup, Intel Aero is :

- a Wifi access point
- forwarding all MAVLINK messages to QGroundControl

But Intel Aero can do a lot more than just forward messages from a serial port on the network: we've seen in the D1 module how we can have local logic handling MAVLINK messages.

Let's connect this local logic with the network!

Scenario

Let’s imagine the following scenario:

- An IoT device is detecting an event, like a toxic gaz in a factory
- The IoT device is sending the alert to the central server
- The server is contacting the autonomous drone to send him on the site of the alert for further investigation

We need a way for the drone to wait
and **receive instructions from the network**.

We’ll use **WebSockets** and **MQTT** to connect Intel Aero to the simulated servers. We’ll just arm the motors for now.

WebSockets

WebSockets?

WebSockets are typical in web applications.
If you read your mail from your browser, it’s probably using a lot of it.

It is also popular in Internet of Things messaging.

WebSockets are used when you have a dialog
between 1 client and 1 server (the server handling several clients)
and performance or latency are not very important.

WebSockets can handle big (files) and small messages.

Setup

Let's install:

- The Python websocket API on Intel Aero:
  pip install websocket-server
- A WebSocket client in your browser, to simulate a web call:
  https://chrome.google.com/webstore/detail/smart-websocket-client/omalebghpgejjiaoknljcfmglgbpocdp
- Optionally, a Library on your development station
  to send the request from a script instead of your browser:
  pip install websocket-client

On Intel Aero: WebSocket Server code

A simple python code to listen for WebSocket calls:

```
from websocket_server import WebsocketServer
import re
from dronekit import connect, VehicleMode, LocationGlobalRelative
import time
vehicle = connect('tcp:127.0.0.1:5760', wait_ready=True)
vehicle.mode    = VehicleMode("GUIDED")
print('Flight Controller Connected')

def new_client(client, server):
        print("Client connected")
        server.send_message_to_all("Client connected")
def message_received(client, server, message):
        if  len(message)  >  200:
                message  =  message[:200]+'..'
        print("Arming motors (client message: " + message+")")
        vehicle.armed   = True
        while not vehicle.armed:
                time.sleep(1)
        time.sleep(5)
        print('Disarming')
        vehicle.armed   = False

server  =  WebsocketServer(8080,  '0.0.0.0')
server.set_fn_new_client(new_client)
server.set_fn_message_received(message_received)
server.run_forever()
```

Browser: WebSocket Client call

Start the connection (EDIT YOUR IP):

ws://192.168.1.105:8080

Click “connect”, you’ll see the server response.

Send a message, click “send”:

”Alert, Send Drone”

The motor should arm for 5s.

**Your drone has local logic processing,
plus network interactions
using modern web technologies like WebSockets.**

On your PC: WebSocket Client code

A simple python code to send a WebSocket client call (EDIT YOUR IP): :

```
from websocket import create_connection
ws = create_connection("ws://192.168.1.105:8080")
ws.send("Alert, send drone")
result = ws.recv()
print("Received '%s'" % result)
ws.close()
```

MQTT

MQTT?

MQTT is a messaging protocol very popular
in Internet of Things architectures.

A server is keeping all the messaging queues,
and clients subscribe to specific messages.

MQTT is high performance, low latency.
It can handle a lot of small messages efficiently.

Setup

Let’s install:

- The Python MQTT API on Intel Aero (should be there already):
  pip install paho-mqtt
- A MQTT client in your browser, to generate a message posting:
  https://chrome.google.com/webstore/detail/mqttlens/hemojaaeigabkbcookmlgmdigohjobjm?hl=en

- *Note: In the following code, we’ll use a public MQTT server.*
  Please pick a unique topic name and do not overuse the public server.

On Intel Aero: MQTT Client code

A simple python code to subscribe to a topic from a public server, and arm motors when a message is received:

```
import paho.mqtt.client as mqtt
from dronekit import connect, VehicleMode, LocationGlobalRelative
import time

vehicle = connect('tcp:127.0.0.1:5760', wait_ready=True)
vehicle.mode    = VehicleMode("GUIDED")
print("Flight Controller Connected")

def on_connect(client, userdata, rc):
        print("Client connected ")
        client.subscribe("aero-paul")

def on_message(client, userdata, msg):
        print("Arming motors ("+msg.topic+"/"+str(msg.payload)+")")
        vehicle.armed     =  True
        while  not  vehicle.armed:
                time.sleep(1)
        time.sleep(5)
        print("Disarming")
        vehicle.armed     =  False

client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message
client.connect("test.mosquitto.org", 1883, 60)
client.loop_forever()
```

Browser: MQTT Post

Start the connection with the public server (mosquitto should be configured already):

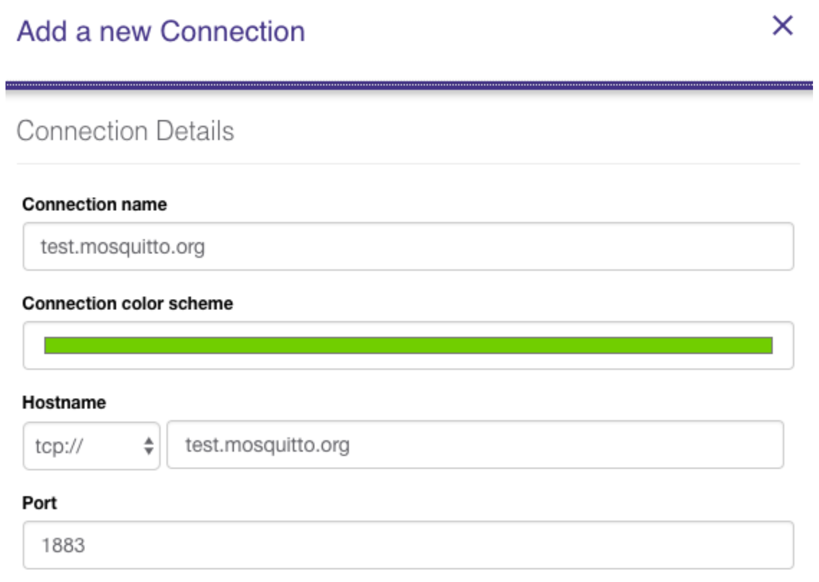
test.mosquitto.org

Click “connect”, you’ll see the server response.

Pick a topic (the same as a the previous code) and click “subscribe”.

Type a message and click “send”.

Your message will be sent to the server queue for the topic, Aero being subscribed to the same topic, it will receive the message and arm the motor for 5s.



Add a new Connection

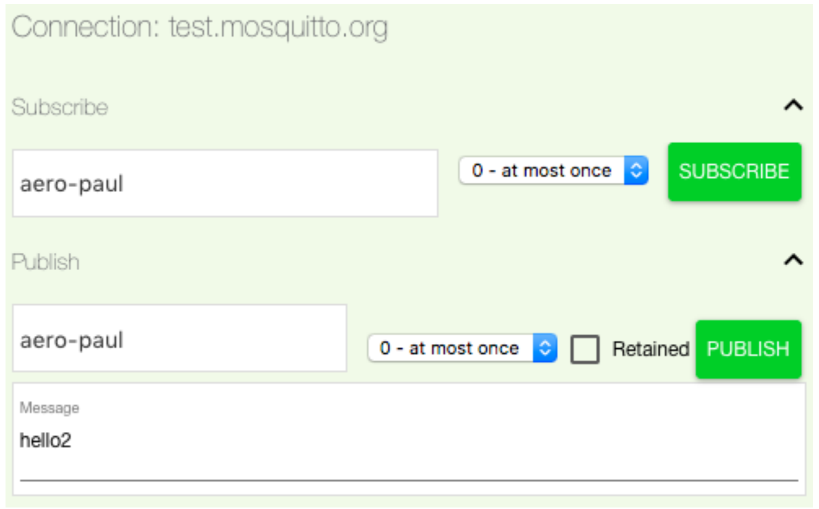
Connection Details

Connection name: test.mosquitto.org

Connection color scheme: 🟩

Hostname: tcp:// test.mosquitto.org

Port: 1883



Connection: test.mosquitto.org

Subscribe

aero-paul | 0 - at most once | SUBSCRIBE

Publish

aero-paul | 0 - at most once | ☐ Retained | PUBLISH

Message: hello2

Your drone has local logic processing, plus network interactions using modern IoT technologies like MQTT.

Thanks

Paul.Guermonprez@intel.com