Professorship for Computer Science
Communication Services, Telecommunication Systems and Computer Networks

Towards Implementing a MQTT-SN Gateway for Semi-Constrained Devices

October 12, 2017

Gabriel Nikol

Content

Project Goal

Implement a Prototype for a MQTT-SN gateway runnable on Semi-Constrained devices.

Outline

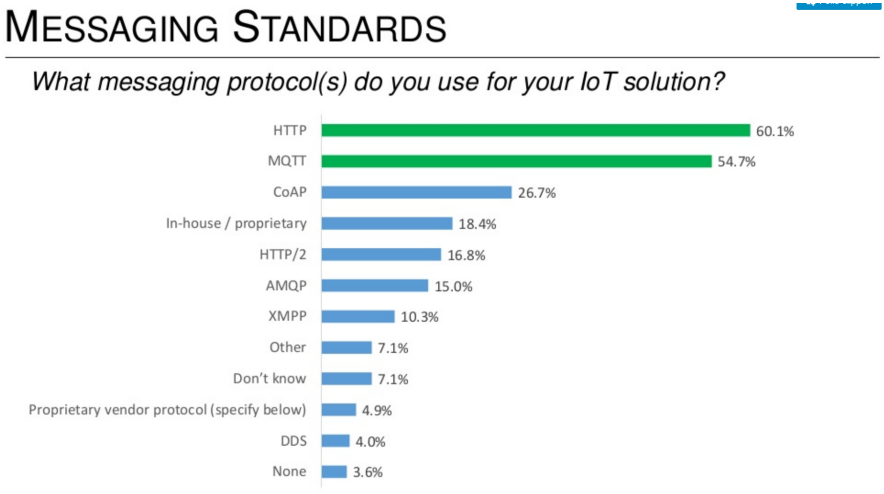

Figure 1: Eclipse IoT Developer Survey 2017 - Messaging Standards

HTPP

- client-server paradigm
- addressable resource (URI)

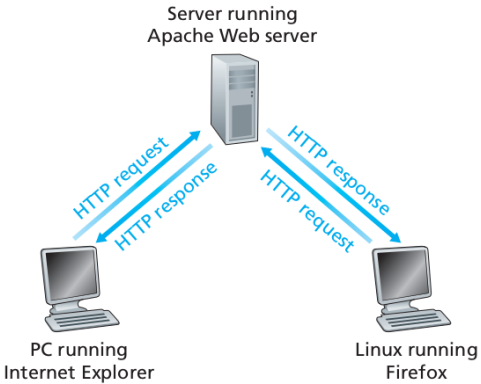

Figure 2: publish subscribe example

MQTT

- publish-subscribe
- data centrich approach

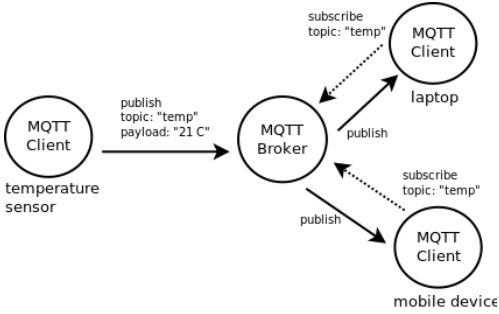

Figure 3: HTTP request response

MQTT QoS

- QoS 0 (at most once)
- QoS 1 (at least once)
- QoS 2 (exactly once)

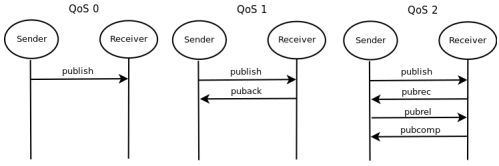

Figure 4: MQTT QoS methods

MQTT-SN

- version of MQTT
- not connection oriented (no TCP)
- supports short & pre-defined topic names
- supports sleeping clients
- QoS -1 (QoS 0 publish without connect)

Comparing HTTP, MQTT, MQTT-SN

- comparing minimal packet sizes in TCP/IP Model
- minimum valuable example: HTTP GET, MQTT publish QoS 0, MQTT-SN publish QoS 0
- result: MQTT + TCP + WiFi = 82 bytes vs MQTT-SN + BLE 21 bytes

TCP/IP Model Layer	Protocol including the minimial length				
Application Layer	HTTP (20)	MQTT (8)	MQTT-SN (8)		
Transport Layer	TCP (20)		UDP (8)		
Network Layer	IP (20)				
Network Interface	Ethernet (20)	WiFi (20)		ZigBee (28)	BLE (13)

Figure 5: TCP/IP Model Layer with minimal protocol length

Outline

IoT Environments

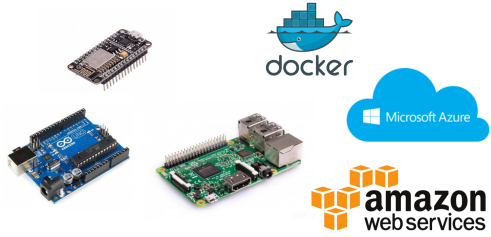

Figure 6: IoT Environment from small hardware to the cloud

Outline

Device Types

- Constrained Devices
  - Limited resources (FLASH, CPU, RAM, Energy)
- Semi-Constrained Devices
  - Limited resource (FLASH, CPU, RAM, not Energy)
- Unconstrained Devices
  - Nealy unlimited resources (HDD, CPU, RAM)

Outline

Target Hardware and Software Environment

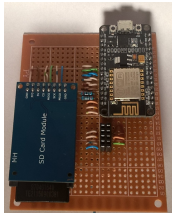

Figure 7: NodeMCU v2 (ESP8266) + SDHC Card as Target Environment

Figure 8: Arduino as Software Environment

Outline

MQTT-SN Architecture

- three MQTT-SN components: client, gateway, forwarder

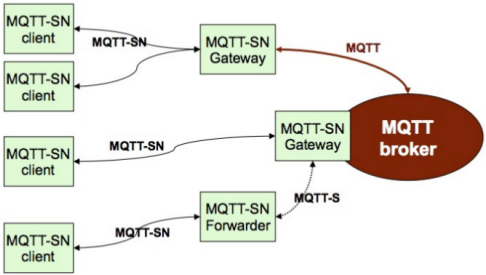

Figure 9: MQTT-SN Architecture

MQTT-SN Gateway Architecture

- two kind of gateways: transparent and aggregating

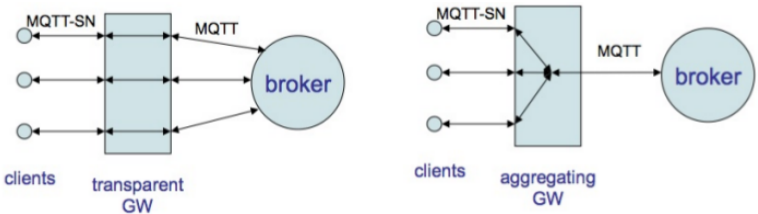

Figure 10: MQTT-SN gateway architecture

MQTT-SN Client Lify Cycle

- MQTT-SN client as a constrained device:
  - find a MQTT-SN Gateway via advertisement or searching a gateway
  - connect to the MQTT-SN gateway with a will message
  - register topics
  - subscribe to topics
  - send and receive publishes
  - unsubscribe from topics
  - sleep
  - wake up and collect queued publishes
  - sleep and wake up frequently
  - power source is empty - will message is published

Outline

Transmission Protocols

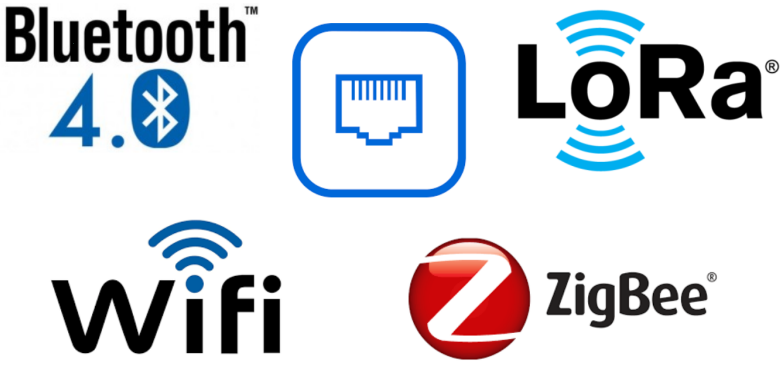

Figure 11: example transmission protocols

Outline

Overviewv - MQTT-SN, Transmission Protocols, Device Types

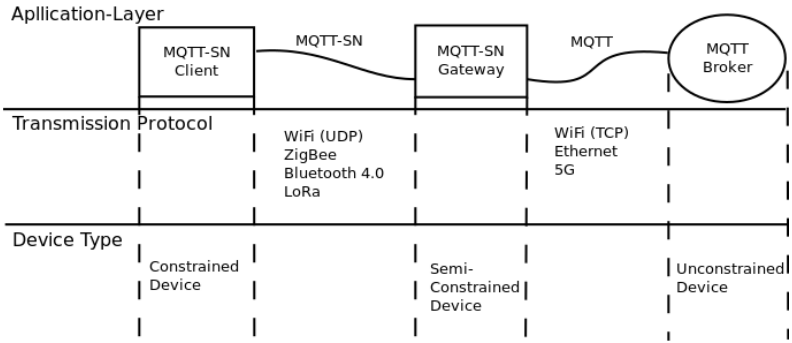

Figure 12: MQTT-SN, Transmission Protocols, Device Types

Outline

Core Components

- environment independent implementation
- MqttSnMessageHandler + Core implemented - rest interfaces

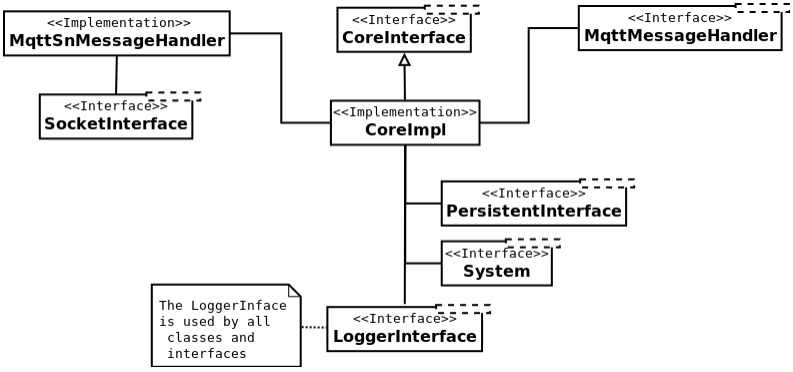

Figure 13: Core Component class diagram

Outline

Core Components

- holds references to implementation - single class to embed
- initializes components' references
- loop()s over components

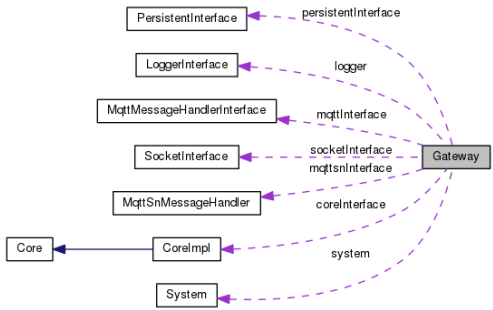

Figure 14: Gateway class collaboration diagram

Outline

Linux Gateway Implementation

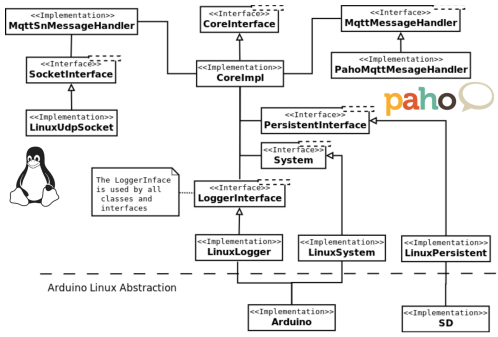

Figure 15: Linux Gateway implements Core Component's interfce. Using UDP and Paho embedded C/C++ MQTT Client

Outline

Unit and Regression Testing

- using GoogleTest and GoogleMock
- starting tests inside IDE CLion
- using Docker for running MQTT broker (Mosquitto)
- writing a MqttSnTestClient + PahoMqttTestMessageHandler

Figure 16: Clion, GoogleTest, Docker, Mosquitto

Outline

MQTT-SN & MQTT Test Client

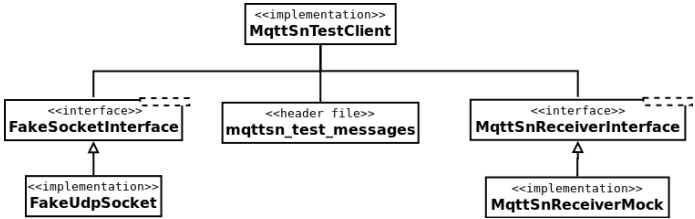

Figure 17: MQTT-SN test client class diagram

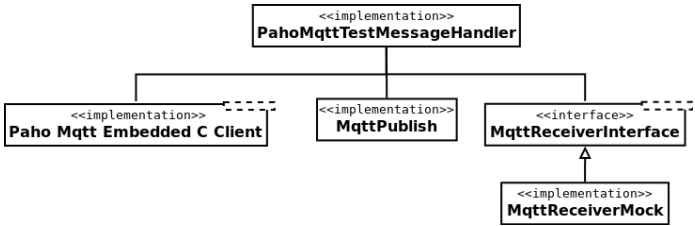

Figure 18: MQTT test client class diagram

Outline

Test Results

- compliance tests (well and ill-formed MQTT-SN packets)
- functional tests (behaviour correct)
- Total: 96 unit tests
- 90 pass + 6 fail (QoS 2 not implemented but tested)
- Not everyting is tested, but: Important functionality is tested and working

Outline

BLESocket

- drop in replacement for LinuxUdpSocket
- uses self written SimpleBluetoothLowEnergySocket

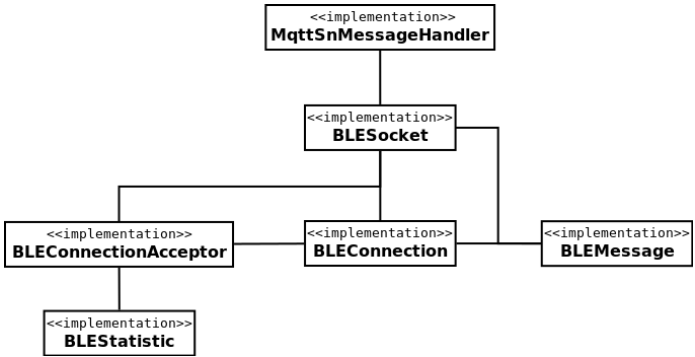

Figure 19: BLESocket class diagram

Conclustion

- tl;dr: project successful
- implemented a MQTT-SN gateway prototype on Linux
- runs partly tested on target hardware and software environment (ESP8266+SDCard)
- tested (reusable)
- designed to be easy adaptable to multiple environments

Future work

- implement a MQTT-SN client for constrained devices
- more transmission protocols (WS17/18 LoRa)
- support more platforms: Mbed, RTOS
- implement: QoS 2 & will update
- enhance tests: more tests, stress tests, measure code coverage

Questions ?

Gabriel Nikol
github.com/S3ler