Messaging

v0.5

AJDOS Air Pollution Sensor

Gabor Finta

Compet-Terra Ltd.

List of contents

Introduction

This documentation contains the description of the messaging of the AJDOS air pollution sensor. The measurement results are sent to the central server in JSON data format via the MQTT message protocol. On the central server, a messaging broker receives the data and forwards it to the central application.

In addition to the measurement data, the sensor notifies the central system in a message that it has been commissioned or that it has been dismounted or relocated.

Each message contains the sensor ID and password required for authentication, the central system only processes messages with the correct format and pre-registered authentication data.

The sensor can send messages about several measurement results in a short time, if the previous data transmissions was not successful (lost Internet connection).

Each message has timestamp.

The sensor can send blank PM2.5 and PM10 data if the measurement cannot be performed. This is due to too low a temperature or too high a humidity, which is outside the operating parameters of the meter.

Messages

Name	MQTT topic	Payload
hello	/ajdos/sensor/hello	{"name":"<sensor name>", "publickey":"<password>", "owner":"<owner_id>", "station":"<station_id>", "location":"POINT(<longitude> <latitude> <altitude> m)", "cycletime":<cycle-time> "time":"<timestamp>" "crc":"<access-key>" }
bye	/ajdos/sensor/bye	{"name":"<sensor name>", "publickey":"<password>", "time":"<timestamp>" }

data	/ajdos/sensor/data	{"name":"<sensor name>",
		"publickey":"<password>",
		"location":"POINT(<longitude> <latitude> <altitude> m)",
		"values":{
		"temperature":<temperature_value>,
		"humidity":<humidity_value>,
		"pm10":<pm10_value>,
		"pm25":<pm2.5_value>,
		"battery":<battery_value>
		}
		"time":"<timestamp>"
		}

Data formats

Data tag	Type	Source
<sensor name>	String	Sensor setup*
<password>	Base64 encoded string	Sensor setup*
<owner_id>	String	Sensor setup*
<station_id>	String	Sensor setup*
<longitude>	Float	Sensor setup
<latitude>	Float	Sensor setup
<altitude>	Float	Sensor setup
<temperature_value>	Float	measured data °C
<humidity_value>	Float	measured data %
<pm10_value>	Float	measured data µm/m3 or -1**
<pm2.5_value>	Float	measured data µm/m3 or -1**
<battery_value>	Float	measured data %
<cycle-time>	Integer	measuring cycle in minutes
<timestamp>	String	GMT time
<access_key>	Base64 encoded string	Sensor setup

*this data preregistered in the application

** -1 when it cannot be measured (see on page 2)

Timestamp

The timestamp is the time after the measurement, which is queried by the sensor via the NTP service, but can also be a calculated time based on the time elapsed since the last successful time request. Counting is required if the NTP server is not available or there is no Internet connection.

The time is always GMT (Greenwich Mean Time) and does not take into account daylight saving time.

The format: “YYYY-MM-DD HH:MM:SS GMT”

Messaging

Each message has two credentials. Sensor name and password. The server application only accepts pre-registered credentials. The name of the sensor comes in the *"name"* field, while the password comes in the *"publickey"* data field. The password value is stored in **base64** encoding.

Data encryption in the sensor hardware is slow, so sensitive data is hidden. Hiding data means that we name the data differently in the JSON value pairs or encode the value of the data. We currently have two such data. The password used in the message is called "*publickey*" and the WiFi password used to set up the sensor is called "*crc*". The "*crc*" is required if the user forgets the set password.

/ajdos/sensor/hello

The message is sent at first start-up after the sensor's main dataset (name, owner, station, location) has been set. The hello message also contains the operator password for the sensor setup mode in the "*crc*" data field.

Important: **If the operator password will be changed by the owner, this message will be sent again with the new password.**

/ajdos/sensor/bye

The message is sent at first start-up, after the main data set of the sensor (name, owner, station, location) has been changed, before the hello message is sent. The content of the message corresponds to the previous settings.

Important: **If the name and / or password of a sensor changes, the previous data will be sent in this message. On the server side, the previous password must be kept until this message arrives.**

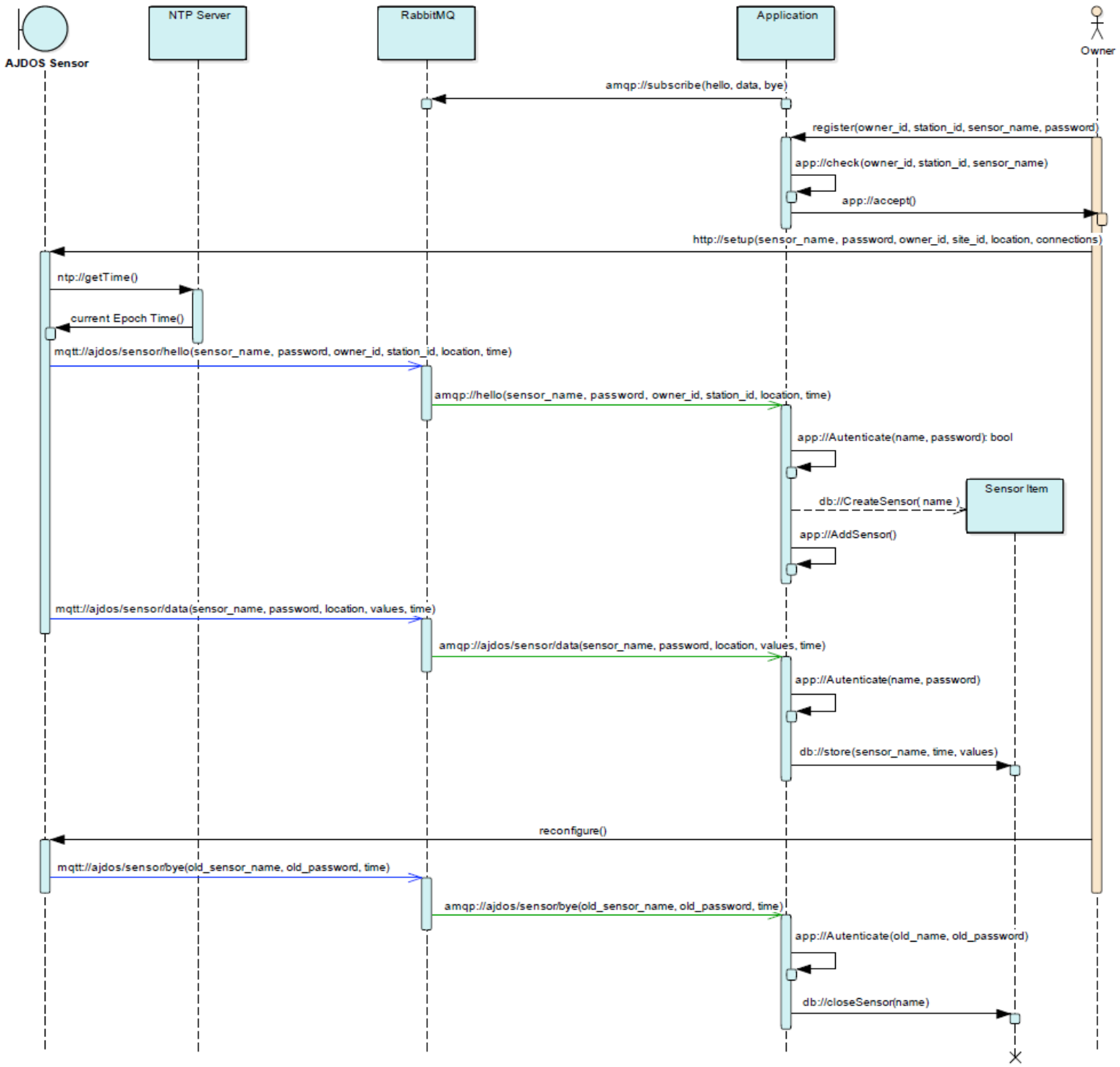
/ajdos/sensor/data

The message is sent after the measurements have been taken according to the set timing. Timing can be configured in the sensor software (for example, every 2 hours)

Diagram

The diagram below shows the sequences of the process from sensor installation to disassembly and the associated messaging. For the messages, we indicated the used protocol and introduced some notations that indicate how the messages were handled (*e.g. mqtt://*).

AMQP	RabbitMQ messaging protocol	in this case we use between the RabbitMQ and the application
MQTT	MQTT messaging protocol	the sensor sends messages to the RabbitMQ message broker
HTTP	Internet protocol	the owner uses the setup program of the sensor
NTP	Network time protocol	the sensor uses it to get the GMT time
APP	non-standard	Used to indicate an in-app message
DB	non-standard	Used to indicate a message related to data storage


AJDOS Sensor
NTP Server
RabbitMQ
Application
Owner
amqp://subscribe(hello, data, bye)
register(owner_id, station_id, sensor_name, password)
app://check(owner_id, station_id, sensor_name)
app://accept()
http://setup(sensor_name, password, owner_id, site_id, location, connections)
ntp://getTime()
current Epoch Time()
mqtt://ajdos/sensor/hello(sensor_name, password, owner_id, station_id, location, time)
amqp://hello(sensor_name, password, owner_id, station_id, location, time)
app://Autenticate(name, password): bool
Sensor Item
db://CreateSensor(name)
app://AddSensor()
mqtt://ajdos/sensor/data(sensor_name, password, location, values, time)
amqp://ajdos/sensor/data(sensor_name, password, location, values, time)
app://Autenticate(name, password)
db://store(sensor_name, time, values)
reconfigure()
mqtt://ajdos/sensor/bye(old_sensor_name, old_password, time)
amqp://ajdos/sensor/bye(old_sensor_name, old_password, time)
app://Autenticate(old_name, old_password)
db://closeSensor(name)

1. First, the app indicates that you would subscribe to the message to the message broker.
2. A user registers himself and his/her sensor data in the application.
3. The user configures the sensor through its configuration interface.
4. Before each message is sent, the sensor queries the current time from an NTP server.
5. The newly configured sensor logs in with the message /ajdos/sensor/ hello.
6. The message broker transmits the new sensor data to the application.
7. The app checks to see if the sensor and its owner have registered before.
8. After successful authentication, the sensor is added to the database.
9. After commissioning the sensor, the measured data is sent to the message broker.
10. The application authenticates the sensor. After successful authentication, the received data is saved in the database.
11. The user can change the sensor data by relocating or permanently removing it.
12. The sensor notifies the broker that it is terminating by sending the previous data and in the /ajdos/sensor/bye message.
13. The message broker transmits the bye message to the application.
14. The application authenticates the sensor.
15. After successful authentication, the sensor data is locked in the database, no more data is stored for this registration.

Samples

/ajdos/sensor/hello

```
{"name":"bp-test","publickey":"MTIzNDU2Nzg=",
"owner":"1", "station":"2",
"location":"POINT(19.21384 47.43915 15.30 m)",
"cycletime":60,
"time":"2021-04-15 18:00 GMT"
"crc":"MTIzNDU2Nzg="}
```

/ajdos/sensor/bye

```
{"name":"bp-test","publickey":"MTIzNDU2Nzg=",
"time":"2021-04-15 18:00 GMT"}
```

/ajdos/sensor/data

```
{"name":"bp-test",
 "publickey":"MTIzNDU2Nzg=",
 "location":"POINT(19.21384 47.43915 15.30 m)",
 "values":{"temperature":18.4,
           "humidity":64.0,
           "pm10":0.32,
           "pm25":1.25,
           "battery":92.
          },
"time":"2021-04-15 18:00 GMT"
}
```