|  |
| --- |
| IALA Guideline |

Gnnnn

A Harmonised IoT Schema? Protocol

for Visual AtoN

Edition x.x

Date (of approval by Council)

urn:mrn:iala:pub:gnnnn

Revisions to this document are to be noted in the table prior to the issue of a revised document.

|  |  |  |
| --- | --- | --- |
| Date | Details | Approval |
| 02/02/24 | Reviewed and updated. |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

1. Introduction 5

2. Scope 5

3. IoT Platform 5

4. Security 6

5. Harmonised payload 6

6. Limitations 7

7. Example 8

8. Example of HEADING 1 STYLE 8

8.1. Example of Heading 2 style 8

8.1.1. Example of Heading 3 style 8

9. SECTION 2 – HEading 1 style 9

9.1. Section 2.1 – Heading 2 style 9

10. Section 3 – Heading 1 Style 10

10.1. Section 3.1 – Heading 2 style 10

10.2. Section 3.2 – Heading 2 style 10

10.3. Section 3.3 – Heading 2 style 10

10.3.1. Equations 11

11. DEFINITIONS 11

12. abbreviations 12

13. references 12

14. Further reading 12

15. Index 13

List of Tables

Table 1 Example of table with row headers 5

Table 2 Example of table with column headers 5

List of Figures

Figure 1 Example of wrapping in line with text 4

Figure 2 Example of wrapped square 5

Figure 3 Example of how to achieve right justified equation number 7

# Introduction

Marine Aids to Navigation (AtoN) have often been early adopters of new technologies. Since the 1980s, remote monitoring of marine signal lanterns has been available as a tool to track the availability of AtoN and predict maintenance needs. Remote Control has also been implemented in some applications. Today, there are various solutions available on the market based on Satellite Communication, GSM mobile networks, Point-to-Point short-range radio communication, as well as AIS transponders.

However, current communication topologies often have a low reporting frequency due to the limitations of data communication costs or energy constraints. Status reports are typically only transmitted when lights turn on in the evening and turn off in the morning, with additional ad hoc reports transmitted when an issue is detected by the station (e.g., position, energy or light operation related). Additionally, many current conventional communication systems have a limitation in the number of communication sessions they can manage, so reporting frequency is not only limited by outstation constraints.

As a result, the owner of the asset always has outdated information and no real-time situational awareness. They may also not be able to detect a malfunction of an AtoN in a timely manner. Due to the lack of industrial standards, each vendor operates a proprietary protocol and system, making it difficult for the owner of assets to mix devices in the field.

# Scope

This guideline will outline how an established Internet of Things (IoT) platform can be utilised to provide a harmonised protocol for connected visual aids to navigation. Such protocol would then allow commonality of data exchange providing interoperability of visual AtoN equipment from a mix of suppliers who adopt such a protocol.

# IoT Platform

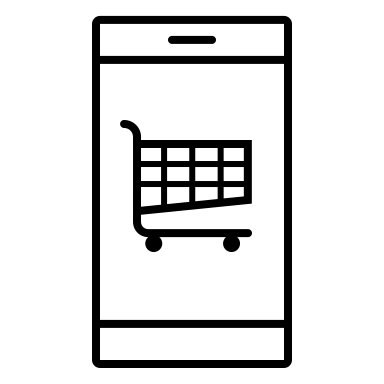
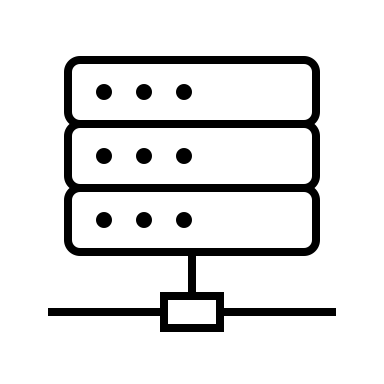
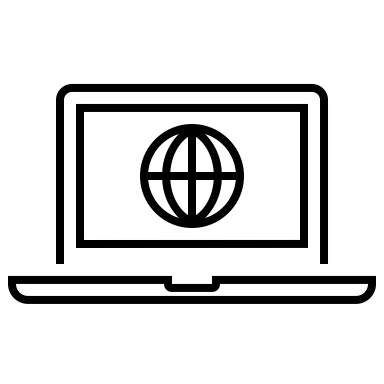
When considering the developing a protocol for interoperability of visual AtoN equipment, we need to understand that the AtoN equipment can be considered as IoT device with various sensors. This device needs to send its information to an IoT application to process and display the information for a user. An IoT platform provides the infrastructure to pass this information between the IoT device and the IoT application and can be considered as in integrated service. Such services can be provided by a 3rd party e.g. Azure or can be delivered within the organisation existing infrastructure.

## Topology

The type of topology adopted, will be subject to the how the IoT platform is to be provided. This section shows two examples.

### 3rd Party platform.

The below diagram shows how AtoN equipment can be connected using a commercially provided IoT platform and the security that would need to be in place.



IoT device (AtoN) Publisher

Internet

IoT platform (Azure broker)

Internet

IoT application Subscriber

1. Topology of a 3rd party IoT solution

### An organisational developed platform.

If considering developing an organisational IoT platform, then there are several infrastructure aspects that may need to be provided and these are shown in the diagram below.



1. Topology of an organisation IoT platform

This example can be considered a hybrid arrangement which shows where there is a number of different protocol employed. In this case the Message Queuing Telemetry Transport (MQTT) protocol is integrated into the broker, but also into the back end server to allow data exchange to the broker.

Peter S to develop another example of topology. With AtoN example and VPN secure example.

## Concept system

EXPLAIN WHY YOU CHOOSE ONE OVER ANOTHER.

There are several communication protocols that are commonly used in the Industrial Internet of Things (IIoT) environment to support the transfer of data between devices and systems. Some of the most widely used protocols include:

MQTT: A lightweight publish/subscribe protocol designed for machine-to-machine (M2M) communication.

CoAP: A protocol designed for resource-constrained devices in IoT networks.

DDS: A real-time data sharing protocol for industrial control and monitoring systems.

AMQP: An open standard for message queuing and data transfer in IoT systems.

OPC UA: A vendor-independent communication protocol for industrial automation and control systems.

LwM2M: A protocol designed for managing and updating device firmware and configuration over the air (OTA).

These protocols are used in various IIoT applications such as industrial control, monitoring, and automation, real-time data collection and analysis, and machine-to-machine communication. The choice of protocol depends on the specific requirements and constraints of the application, including network bandwidth, latency, security, and scalability.

1. Comparison of various IIoT protocols

Add table from Jonas

Given the sample of various IIoT protocol given in the table above, the most appropriate protocol for a particular case depends on factors such as data rate, security, power consumption, compatibility, and complexity.

For the Visual AtoN IoT harmonization, the MQTT (Message Queuing Telemetry Transport) protocol is recommended as the best suited protocol. This protocol is optimal for small low-power applications and is one of the most commonly used communication protocols. It can be implemented in ultra low-power microcontroller applications, and on the server side, the MQTT broker is straightforward to establish and integrates well with an already established remote monitoring system.

Here is how the MQTT protocol works in more detail:

1. Clients: MQTT clients can be either publishers or subscribers. Publishers are responsible for sending messages to the broker, while subscribers are interested in receiving those messages.
2. Broker: The MQTT broker acts as a organizer or coordinator between publishers and subscribers. It receives messages from publishers and then distributes them to subscribers that have expressed an interest in those messages.
3. Topics: Messages are sent and received on topics, which act as channels for communication. Topics are hierarchical in nature, allowing for the creation of a tree-like structure that can be used to organize messages by topic.
4. Quality of Service (QoS): MQTT supports three levels of Quality of Service (QoS) for message delivery:
   1. QoS 0, provides at most a single delivery;
   2. QoS 1, provides at least a single delivery; and
   3. QoS 2., provides ensures a single delivery is received.
5. Keep Alive: MQTT uses a Keep Alive mechanism to ensure that clients remain connected to the broker even if there is no data to transmit. Clients send periodic ping request (PINGREQ) messages to the broker to indicate that they are still connected.
6. Last Will and Testament (LWT): MQTT supports a Last Will and Testament (LWT) feature that allows a client to specify a message that will be published by the broker in the event that the client becomes disconnected unexpectedly.

A close-up of several white rectangular objects

Description automatically generated

Publisher

Subscriber

Subscriber

1. Example of publishing and subscribing

# Security

General guidance on Cyber security needs to be considered at the design stage and guidance can be obtained from the guideline ??? on Cyber Security.

When considering MQTT, it is a protocol that runs over TCP/IP and does not provide security measures itself, but it can be implemented with security measures to ensure secure communication between the client and the broker.

## Encryption and Authentication

By using MQTT, it does support the use of security measures such as SSL/TLS encryption, which provides a secure communication channel between the client and the broker. When SSL/TLS encryption is enabled or enforced, all data transmitted between the client and the broker is encrypted and cannot be intercepted by a third party. Such security measure must be implemented when one of the communication channels utilises the public internet, to avoid unencrypted data being transmitted.

Use of public internet must have security. Note don’t allow unencrypted data.

Alternative measures could…

What about VPN as security.

What about server side certificate?

Typically an MQTT sensor is linked using a white list, with user name and password. Alternatively a certificate needs to be installed on the sensor. This would add a lot of security and can be more complicated to maintain. Consideration on the importance of the data needs to be reflective of the level of security applied. NEED TO BE DISCUSSED.

When considering the levels of security to apply, it should be reflective of the importance of the data. Additionally such security requires additional processing power, which can impact on type, cost and power of the unit, as well as the potential duration of the communication sessions, which again can impact on cost and power.

Additionally, MQTT supports authentication mechanisms, such as username and password, to ensure that only authorized clients can connect to the broker. Access control lists (ACLs) can also be used to restrict the operations that clients can perform on the broker.

Protocol includes error detection, ID, etc. Payload is the rest.

As a minimum security measure, it is recommend that communication is restricted to only using TLS 1.2 or higher using a dedicated Secure-MQTT port 8883. This assumes communications is achieved over the public internet.

Only 8883 and 443 Inbound Ports Open

TLS Security

On Premise or Cloud Network

Initiates Outbound TLS Connection

Edge Network

MQTT Edge Client Device



F  
i  
r  
e  
w  
a  
l  
l

Access Control Lists   
(ACLs)

Username / Password Authentication

No Open Inbound Ports

F  
i  
r  
e  
w  
a  
l  
l

MQTT Server

1. Example of MQTT Security, Courtesy of Cirrus Link need to recreate

# Harmonized payload

To achieve a goal of a common schema for visual AtoNs, it is not enough to only standardize on using MQTT as the connectivity protocol, there is a need to define and harmonise the payload structure as well as the type of payloads. This section discusses how the payload is to be structured. For more detailed information on payloads see appendix 1.

## Message structure.

When considering the information to be captured, stored and sent, the structure of such messages can be arranged by adopting any number of existing industrial approaches, but there is a conflicting trade off to be had. For example, a solution can be a byte-by-byte structure, keeping the size of the data small, but this can’t be easily read and understood. Alternatively, a much larger and easily read structure could be achieve by the use of XML. Again these choices impact the amount of data sent and the power and communication cost to achieve this. Therefore a balanced solution is needed as both a low data size and readability are desirable.

To achieve this desired balance a structure based upon JavaScript Object Notation (JSON) is to be adopted. JSON is a lightweight data interchange format that is easy to read and write for humans, and relatively easy to parse and generate for machines. JSON is also a platform-independent data format, which means it can be used with any programming language or platform.

In this case it is chosen to use a more balanced version JSON. MQTT-SN (max 60 bytes) this might constrain what can be achieve. Limiting in how it can be adopted by commercial MQTT brokers such as Azure or AWS. JSON has lots of libraries

The following example shows a possible JSON representation describing a person.

A screenshot of a computer code

Description automatically generated

1. An example of JSON syntax, https://en.wikipedia.org/wiki/JSON

Jonas to provide a clear example

## Payload

The payload can be considered as all of the data that is to be sent.

From the example above (figure 5) using the JSON syntax, there are some areas of information, shown in green, such as children, which are known as “topics”. A more typical topic for a visual AtoN might be battery voltage, position, etc., and are selected by the supplier, based on the type of sensor or equipment.

# These topics are then populated by values or parameters. These parameters have a defined data type based on the topic. So, for the children in the example above, the data type is a string. Given this such topics and data types have been captured in appendix 1 for aspect associated with visual AtoN.Parameters = Topics = Status e.g Battery Voltage, Position, etc (Data Sets) Limitations

Energy

Limitations

Possible key point to include from DTEC1 Liaison note at ENG17

IALA G1179 – An introduction to the Internet of Things from an IALA perspective

• There would benefit in preparing a comparison table that looks at: the IoT typical use case; size of message; latency; frequency; availability of network; existing standards; IP (or royalty free); efficiency of protocol; reliability of information throughput.

• When looking to implement IoT in the IALA perspective, the availability and reliability of information were discussed, noting possible liability issues.

• Guiding principles on implementing IoT could include: efficient; affordable; scalable; with a focus on harmonised sharing of data.

| Title | Description | Expected outcome | Committee (\*leading) |
| --- | --- | --- | --- |
| S1010, 1.1  Guidance on the use of simple IOT sensors on physical aids | Establish requirement for IOT sensors. | Guideline | ARM |
| S1060, 6.2  Define user requirements for Maritime Connectivity, Maritime Internet of Things (IoT), and MRN addressing (may be three subtasks) | Revised Guideline G1143 to include aspects relevant to MRN | Revised guideline | ARM |

# Example

Carrier

Application cases

ISO7 layer model

1. ISO 7 Layer Model

# Example of HEADING 1 STYLE

This guideline template should be used in conjunction with the *IALA Style Guide*. Utilising the styles provided in the **Styles Gallery** is key to using the document templates. Selecting the appropriate style from the Style Gallery will apply most text (and often layout) formatting required to comply with the *IALA Style Guide.* There should be no need to apply font formatting, numbering or bullets by selecting options from the **Font** or **Paragraph** dialog boxes.

The main text within a document is written in the **Body text** style, which is Calibri and 11 font size. Section titles can be inserted for up to four levels of text and should be created using the **Heading 1**, **Heading 2** etc. styles. Ensure the correct heading styles are selected as there are similar heading styles for the annexe and appendix entries, respectively. The blue colour used in the section headings and table texts is Red Green Blue (RGB) R0, G85, B140. The styles are referenced throughout this template and highlighted in bold.

**Heading 1 separation line style** follows the first carriage return after the first level heading title, and the style **Body text** follows the second carriage return after the separation line (if the line disappears, reposition the cursor at the end of the section heading text and press carriage return).

## Example of Heading 2 style

**Heading 2 separation line** style follows the second carriage return after the second level heading title, and the style **Body text** follows the second carriage return after the separation line.

### Example of Heading 3 style

**Body Text** style follows the first carriage return after the third level heading title; there is no separation line at this level.

#### Figures – Heading 4 style

**Body Text** style follows the first carriage return after the fourth level heading title; there is no separation line at this level.

Footnotes should be used sparingly but can be inserted and are found in **Footnote Reference** style at the bottom of the page[[1]](#footnote-1).



1. Example of wrapping in line with text

Figures should be centred with wrapping **In Line with Text** and labelled by writing the figure titles using the **Figure caption** style below the figure. It is important to note that figures and tables should be labelled in this manner with their respective styles to ensure that the tables in the contents section are updated correctly.

##### Alternative figure layout – Heading 5 style

Alternatively, figures can be offset with **Square** text wrapping so that the text does not overlap the figure but arranges the paragraph such that it continues onto the next line in an appropriately sized paragraph.

If no figures are included in the guideline, the respective table on the contents page should be deleted.

1. Example of wrapped square

# SECTION 2 – HEading 1 style

Sections should be typed continuously, and generally page breaks or section breaks should not be entered between main sections. It may be necessary sometimes to insert a page break to allow for aesthetic layout e.g., not breaking a list over two pages.

## Section 2.1 – Heading 2 style

Tables should be centred on the page. The table label should be created using the **Table caption** style and the caption should be positioned above the table.

Table text should be **Table heading** style for the column or row headings and **Table text** style for the content. The style **Table inset list** can be used for bulleted content within a table. The default table layout is for left justified and vertically centred table text but this can be amended using the table Layout menu to suit the content.

1. Example of table with row headers

| Table heading | Table text |
| --- | --- |
| Table heading | Table text |
| Table heading | Table text |

Space below the table should be maintained or inserted as necessary for clarity.

1. Example of table with column headers

| Table heading | Table heading | Table heading | Table heading |
| --- | --- | --- | --- |
| Table text | Table text | Table text | Table text |
| Table text | Table text | Table text | Table text |
| Table text | * Table inset list * Table inset list * Table inset list | Table text | Table text |

If no tables are included in the guideline, the respective table on the contents page should be deleted.

# Section 3 – Heading 1 Style

## Section 3.1 – Heading 2 style

The choice of numbered or bullet point lists depends on the context and content of the text and further guidance is given in the *IALA Style Guide*. Bullets are preferred unless it is important that the list is numbered e.g., for future reference or for a sequence.

Three levels of list styles are provided and these styles should be used rather than the default Microsoft Word numbering lists:

1. List 1 style example

**List 1 text** style example

* 1. **List a** style example

**List a text** style example

* + 1. **List i** style example

**List i text** style example

Each list style has a corresponding list text style that can be used for example, if the list requires more than one paragraph and the subsequent text needs to be aligned. If more than one list is used throughout the document it may be necessary to right click and select **Restart at 1** for subsequent lists.

## Section 3.2 – Heading 2 style

There are three levels of bullet point styles available:

* **Bullet 1** style

**Bullet 1 text** style

* **Bullet 2** style

**Bullet 2** **text** style

* **Bullet 3** style

**Bullet 3 text** style

Each bullet style has a corresponding bullet text style that can be used for example, if the bullet requires more than one paragraph and the subsequent text needs to be aligned.

## Section 3.3 – Heading 2 style

### Equations

#### Layout

If equations are included in the main body of the text, they should be explicitly referred to in the running text and centred on the page. Equations should be numbered consecutively with a right justified number in brackets e.g. (1) on the same line as the equation.

Any explanatory terms should be indented immediately below the equation starting with the non-capitalised term “where” and each term punctuated with a semi-colon until the penultimate term which should also include a semi-colon and the non-capitalised word “and”. For example:

The modified impulse response function is expressed by Equation (1):

where

*;*

*;*

*; and*

*.*

#### Numbering

The preferred method for including equations in the template documents is the Microsoft Word Equation Editor found in the **Insert** menu. The preferred layout and number reference described above can be achieved by typing #(x) where x is the number required immediately after the equation.



1. Example of how to achieve right justified equation number

For example, typing the formula followed by #(2) (as shown in Figure 3) and then pressing return will result in the following equation centred on the page and number being displayed on the same line to the right:

Note that equations do not automatically renumber using this method. If another equation is inserted between two existing equations the number must be adjusted manually.

Although the Microsoft Word Equation Editor is the preferred way of inserting equations, sometimes it is necessary to insert equations created elsewhere and copy those into the document as pictures. In the example below, the equation is included as a picture, wrapped **In Front of Text**:

“The area of a circle is shown in equation (3):

The picture can be copied directly into the document. To insert the corresponding number manually select **Equation number** paragraph style. The author may need to manually adjust the picture position to ensure it is centred and level with the number, achieving consistency with the automatically generated Microsoft Word Equation Editor layout described above.

# DEFINITIONS

The definitions of terms used in this Guideline can be found in the *International Dictionary of Marine Aids to Navigation* (IALA dictionary) at <http://www.iala-aism.org/wiki/dictionary> and were checked as correct at the time of going to print. Where conflict arises, the IALA Dictionary should be considered as the authoritative source of definitions used in IALA documents.

# abbreviations

This section should be typed with the **Abbreviations** style. The acronym or initialism is typed and then tab is pressed so that the style inserts the appropriate tabs and paragraph spacings e.g.:

NGO Non-governmental organization

VTS Vessel Traffic Services

The list should be typed in alphabetical order. The text automatically aligns as an indented paragraph until carriage return is hit and then the next term can be entered.

# references

References are sources directly referred to in the running text and should be given a sequential number, starting at 1. The reference number should be included as close to the referenced text as possible and included as a number within square brackets.

The reference should be listed in the References section in the following syntax using the **Reference** **list** style:

[Author surname,] <space> [initial.] <space> [year] <space> [title.]

For example:

“Hawking also suggests ways that quantum mechanics can be combined with the theory of special relativity [1]. This text builds on his discussion of the instability of black holes described in *A Brief History of Time* [2].”

should be included in the reference list as follows:

1. Hawking, S. (2001) The Universe in a Nutshell.
2. Hawking, S. (1988) A Brief History of Time.

The **Reference list** style will add a number for the reference as soon as you start typing the text and the paragraph will automatically align with the first line of text. Press return to enter a new reference in the list.

# Further reading

Any texts that are recommended to the reader without direct reference in the text should be listed within this section using the same syntax as the reference list. Sources should be listed using the **Further reading** style.

1. Einstein, A. (1905) Relativity: The Special and General Theory of Relativity
2. Idle, E. (1984) The Galaxy Song

# Index

**No index entries found.**

1. Visual AtoN MQTT Protocol Structure
   1. Topic Names

Each topic contains four fields and an optional device name:

1. First level topic name describes the topic purpose, e.g. telematics topic, information topic, etc. First level topic names are described below.
2. Second and third level topic contain location or region, eg. Estonia/Tallinn.
3. Fourth level topic contain site name, eg. Soderskar-lighthouse.
4. Fifth level contain device name or identifier.
   * 1. First level topic names:
5. ’tele’ – telemetry information. Device issued automatic monitoring data.

Example:

tele/<location1>/<location2>/<site name>

tele/<location1>/<location2>/<site name><device x>

1. ’info’ – device information. Device ID, capabilities. Etc.

Example:

info/<location1>/<location2>/<site name>

info/<location1>/<location2>/<site name>/res

info/<location1>/<location2>/<site name><device x>

info/<location1>/<location2>/<site name><device x>/res

1. cmd’ – command, can trigger telemetry response.

Example:

cmd/<location1>/<location2>/<site name>

cmd/<location1>/<location2>/<site name>/res

cmd/<location1>/<location2>/<site name><device x>

cmd/<location1>/<location2>/<site name><device x>/res

1. ’parameter’ – get or set parameter(s)

Example:

parameter/<location1>/<location2>/<site name>

parameter/<location1>/<location2>/<site name>/res

parameter/<location1>/<location2>/<site name><device x>

parameter/<location1>/<location2>/<site name><device x>/res

1. ’direct’ – device specific data that can be used for implementing custom protocols inside device, like Modbus.

Example:

direct/<location1>/<location2>/<site name>

direct/<location1>/<location2>/<site name>/res

OR

direct/<location1>/<location2>/<site name><device x>

direct/<location1>/<location2>/<site name><device x>/res

Subscribe example:

1. Subscribe to all telemetry messages

tele/#

1. Subscribe to all telemetry messages in region named ’Country1’

tele/country1/#

1. Subscribe to all telemetry messages in region named ’country1/north-territory’

tele/country1/north-territory/#

1. Subscribe to telemetry messages which come from ’Lighthouse1’ in ’country1/north- territory’ region

tele/country1/north-territory/Lighthouse1/

1. Subscribe to telemetry messages which issued by telematics module, and flasher1 and flasher2

tele/country1/north-territory/Lighthouse1/telematics

tele/country1/north-territory/Lighthouse1/flasher1

tele/country1/north-territory/Lighthouse1/flasher2

* 1. Communication Sequence

Every connection (publisher or subscriber) should subscribe to following topics:

* Command (’cmd’ topic)
* Parameter

1. Telemetry information

* Device connect to broker and send:
  + Telemetry packet. Depending on configuration this may be repeated during session.
* Device disconnect from broker

1. Info

* Device connect to broker, and send following data
  + Subscribe for topics
  + Telemetry packet
  + Server ask info
  + Device send info topic
  + Server send done
* Device disconnect from broker

1. Set parameters

* Device connect to broker, and send following data
  + Subscribe for topics
  + Telemetry packet
  + Server ask parameter
  + Device send parameter
  + Device update parameter
  + Server send done
* Device disconnect from broker
  1. Payload Format

Payloads have two different formats: unencrypted JSON format, and encrypted format.

1. Example of payloads

|  |  |  |  |
| --- | --- | --- | --- |
|  | Byte 0 | Byte 1 | Byte n |
| JSON | ’{’ | Unencrypted JSON data | |
| Encrypted data | ’E’ | Unused, should be 0 | Encrypted binary data |

When unencrypted JSON data is used in MQTT payload, only plain JSON messages transmitted, no data is added. JSON data should follow the format published in https://www.json.org/json-en.html.

When payload is encrypted, then first byte must be ’E’ (ASCII 0x45) and second byte is reserved and should have value 0x00. All subsequent bytes are encrypted JSON data. Unencrypted payload must be multiple of 16 (AES requirement), it is recommended to fill unused bytes with random data after terminating 0x00 in JSON string.

* 1. Payload Data
     1. General payload data rules

1. Payload is in JSON format (https://www.json.org/json-en.html).
2. Property names are only ASCII.
3. Maximum property name length is 32 characters.
4. In property names are allowed only lower case letters (’a’ – ’z’) and numbers. ’-’ (minus) is used to separate words in property names.
5. Property values are UTF-8 encoded strings.
6. Not recommended property name is ”class”.
7. User defined properties are allowed, but must follow above listed limits.
8. All optional properties can be omitted or have null value when data is invalid. Non-optional properties should have default value in case of invalid data.
   * 1. Topic ’tele’ – status information

Minimal JSON message for generic device:

{

"session-id": "session-1",

"status": "ready",

"uptime": 20

}

6.4.2.1 Session ID

* This property is mandatory
* Property name: ’session-id’
* Description: can be monotonic counter, e.g. timer or session counter
* Example: "session-id": "session-820923084792"

6.4.2.2 Status

* This property is mandatory
* Property name: ’status’
* Allowed values: init, ready, alert, suspend
  + init – least one component to initialize, default value
  + ready – system is fully functional, e.g. lantern is switched on
  + alert – alarm condition detected, e.g. low battery
  + suspend – when system is switched off but it is functional, e.g. storage state. This field is not required on devices which does not have suspend state

6.4.2.3 Uptime

* This property is mandatory
* Property name: ’uptime’
* Seconds from last boot. Default value is 0.
* Example: "uptime": 211

6.4.2.4 Time

* Required only on devices with RTC clock
* Property name: ’time’
* Description: device UTC time, default value is 0
* Allowed values: seconds from January 1st, 1970 at UTC (UNIX time). Only positive values allowed.
* Example: "time": 1673564596

6.4.2.5 Alert status

* Only required when device status is ’alert’
* Property name: ’alert’
* Description: list of alert statuses. If device operates normally and don’t have any alerts, then this property may be omitted or set value to ’none’. Possible values:
  + none, default value
  + light-fail
  + low-battery
  + gnss-error
  + off-location
  + overheated
  + etc...
* Example, two alerts: "alert": ["low-battery", "off-location"]

6.4.2.6 Beacon status

* Mandatory only on beacons. ’type’ field in info message should be ’beacon’.
* Property name: ’beacon-status’
* Allowed values:
  + on-main-character – main or night character
  + on-alternative-character – alternative or day character
  + off – not flashing, default value
* Example: "beacon-status": "on-main-character"

6.4.2.7 Device temperature

* Only on devices which have temperature sensor.
* Property name: ’temperature’
* Contains following sub properties:
  + ’last’ – last read temperature
  + ’max’ – maximum temperature in last 24 hours, optional
  + ’min’ – minimum temperature in last 24 hours, optional
  + ’avg’ – average temperature in last 24 hours, optional
* Allowed values: floating point value
* Example, valid temperature: "temperature": {"last": 21.0, "max": 25, "min": 19, "avg": 22}

6.4.2.8 Battery voltage

* Only on devices which can measure battery voltage
* Property name: ’voltage’
* Contains following sub properties:
  + ’average’ – last battery voltage, averaged over one flash cycle
  + ’loaded’’ – last measured battery voltage under load condition, optional
  + ’unloaded’’ – last measured battery voltage under no-load condition, optional
  + ’max’ – maximum battery voltage under no-load condition in last 24 hour, optional
  + ’min’ – minimum battery voltage under load condition in last 24 hours, optional
* Allowed values: positive floating point value
* Example, valid battery voltage: "voltage": {"average": 12.3, "loaded": 12.1, "unloaded": 12.5, "max": 13.3, "min": 11.5}

6.4.2.9 Location

* Only on devices with GNSS receiver
* Property name: ’position’
* Description: JSON array of floating point values with last GNSS position, exact method how this value is computed is implementation defined
* Allowed values: degrees for latitude -90.0/90.0, and for longitude -180.0/180.0
* Example: "position": [60.0, 110.0]

6.4.2.10 Deviation

* Only on devices with GNSS receiver
* Property name: ’position-deviation’
* Description: Floating point value of deviation in meters from GNSS fixed position. Exact method how this value is computed is implementation defined
* Allowed values: positive floating point value
* Example: "position-deviation": 10.0

6.4.2.11 Last GNSS fix time

* Only on devices with GNSS receiver
* Property name: ’position-time’
* Description: Floating point value of UTC time derived from GNSS.
* Allowed values: seconds from January 1st, 1970 at UTC (UNIX time). Only positive values allowed.
* Example: "position-time": 1673564596

6.4.2.12 GNSS quality indicator

* Only on devices with GNSS receiver
* Property name: ’position-quality’
* Description: Floating point value of CSQ value
* Allowed values: only positive floating point values allowed
* Example: "position-quality": 1.1

6.4.2.13 Ambient light level measured by the light sensor

* Only on devices that can measure ambient light level.
* Property name: ’ambient-light-level’
* Description: Ambient light level measured in lux
* Allowed values: only positive floating-point values allowed
* Example: "ambient-light-level": 30

6.4.2.14 Network statistics

* Optional. If present must contain all sub properties
* Property name: ’network-statistics’
* Description: contains nested statistics information
* Allowed values: positive value, default value is 0
* Example: "network-statistics": {"succeeded-server-connections": 2, "failed-server-connections": 0, "succeeded-network-logins": 2, "failed-network-logins": 2}

Amount of succeeded server connections

* Property name: ’succeeded-server-connections’
* Description: number connections between server and broker that have valid end.
* Allowed values: positive value, default value is 0
* Example: "succeeded-server-connections": 2

Amount of failed connections

* Optional. All network statistics data should be present in together: ’succeeded-server-connections’, ’failed-server-connections’, ’succeeded-network-logins’ and ’failed-network-logins’.
* Property name: ’failed-server-connections’
* Description: number connections between server and broker that have no valid end.
* Allowed values: positive value, default value is 0
* Example: "failed-server-connections": 0

Amount of succeeded network logins

* Optional. All network statistics data should be present in together: ’succeeded-server-connections’, ’failed-server-connections’, ’succeeded-network-logins’ and ’failed-network-logins’.
* Property name: ’succeeded-network-logins’
* Description: number of succeeded network logins
* Allowed values: positive value, default value is 0
* Example: "succeeded-network-logins": 2

Amount of failed network logins

* Optional. All network statistics data should be present in together: ’succeeded-server-connections’, ’failed-server-connections’, ’succeeded-network-logins’ and ’failed-network-logins’.
* Property name: ’failed-network-logins’
* Description: number of failed network logins
* Allowed values: positive value, default value is 0
* Example: "failed-network-logins": 2

6.4.2.15 Last reset source

* Optional
* Property name: ’last-reset-source’
* Description: number of resets starting from production
* Allowed values:
  + por – power on reset
  + wdr – watchdog reset
  + rst – reset from external reset signal (HW signal)
  + bor – brown-out reset
  + usr – reset triggered by command, e.g. SMS reset command
  + other – all other reset sources
* Example: "last-reset-source": "wdt"

6.4.2.16 Reset count

* Optional and only when last reset source is present. This property have nested properties with reset names defined in Last Reset sources.
* Property name: ’reset-count’
* Allowed values: positive value, default value is 0
* Example: "reset-count": {"por":30, "wdr":1}
  + 1. Topic ’info’

Info is triggered by ’cmd’ topic ’"send": "info"’

6.4.3.1 Protocol version

* This property is mandatory.
* Property name: ’protocol-version’
* Integer to describe protocol version. Currently supported value is 1.
* Example: "protocol-version": 1

6.4.3.2 Type

* This property is mandatory
* Property name: ’type’
* Description: device type class
  + ’group’ – logical container for device group
  + ’beacon’ – for beacons
* Example: "type": "beacon"

6.4.3.3 System information

* This property is mandatory. This information can be used to set up optimal set/get transmission packet sizes.
* Property name: ’sys-info’
* Description: system parameters
  + ’rx-buf’ – size of rx buffer in bytes, -1 means infinite
  + ’tx-buf’ – size of tx buffer in bytes, -1 means infinite
* Example: "sys-info": [{"rx-buf": 512}, {"tx-buf": 512}]

6.4.3.4 Serial number

* This property is mandatory for non group devices
* Property name: ’serial-nr’
* Description: device serial number, this number may contain product code also, if product code and serial number are not related then ’product-code’ property show product code

6.4.3.5 Product code

* Required only when serial number does not have product information
* Property name: ’product-code’
* Description: device product code

6.4.3.6 Firmware version

* This property is mandatory for non-group devices
* Property name: ’firmware-version’
* Description: device firmware version

6.4.3.7 Component info

* Only for device which have components with own version
* Property name: ’component-info’
* Description: device component version list, like onboard GNSS receiver.
* Example: "component-info": [{"gnss-version":"1.0"}, {"gnss-type":"NEO M8N"}]

6.4.3.8 Limits

* Optional property. Not required to list all parameters.
* Property name: ’limits’
* Description: returns list of device limit values, for example maximum allowed light intensity, maximum battery voltage, etc. Limit values and value names must match with same configuration parameters.
* Example: "limits": [{"light-intensity":1000}, {"low-voltage-level":6.0}]
  + 1. Topic ’cmd’

This topic is for server initiated actions.

6.4.4.1 Send

* Required
* Property name: ’send’
* Description: send requested topic
* Allowed values:
  + tele – for telematics
  + info – for information
* Example: "send": "info"

6.4.4.2 Reset

* Property name: ’reset’
* Description: reset device or parameter
* Allowed values:
  + null, empty string or ’reset’– reset device (required)
  + parameters – reset all parameters (optional)
  + <parameter-name> – reset parameter name to default (optional)
* Example: "reset": null

6.4.4.3 Done

* Required
* Property name: ’done’
* Description: this is hint from server, that server has been completed all tasks and controller if free to disconnect from broker. It depends on client configuration if it disconnects immediately or send telematics packets. If server send packet after ’done’ message then for disconnect is needed re-send ’done’ message. If server does not send ’done’ message then controller can disconnect if last message from server was more than X seconds ago.
* Allowed values: any string
* Example: "done": "ok"

6.4.4.4 Light on demand

* Optional
* Property name: ’light-on-demand’
* Description: start light on demand, parameters describe how many seconds is light on demand mode is active. Device returns normal operation after this time is elapsed. This command allows to specify optional intensity for light on demand operation.
* Contains following sub properties:
* ’timeout’ – last battery voltage, averaged over one flash cycle
  + -1 – light on demand is active until switched off
  + 0 – light on demand switched off
  + 1...2147483648 – seconds active
* ’intensity’ – effective intensity in cd, optional
* Example: "light-on-demand": {"timeout": 3600, "intensity": 300}
* Light activated for one hour with 300 cd effective intensity

6.4.4.5 Fix position

* Optional
* Property name: ’fix-position’
* Description: start or stop GNSS position fix
* Allowed values:
  + start – start positon fix
  + stop – stop position fix
* Example: "fix-position": "start"
  + 1. Topics ’parameter’

Set or get configuration parameter. All get commands must have ’res-topic’ property. All topic queries have parameter value null.

Get

{

"res-topic": "get/locationa/locationb/site/device1/res",

"time": null

"light-intensity": null

}

Response to get/locationa/locationb/site/device1/res

{

"session-id": "session-1",

"time": 1677677908

"light-intensity": 34

}

Set

{

"res-topic": "get/locationa/locationb/site/device1/res",

"time": 1677679999

"light-intensity": 30

}

6.4.5.1 Date and time

* Property name: ’time’
* Description: set or get time
* Allowed values:
  + null – query from server
  + any positive number – set or get result

6.4.5.2 Light intensity

* Property name: ’light-intensity’
* Description: set or get effective light intensity in candelas.
* Allowed values:
  + null – query from server
  + any positive number to max allowed value – light intensity in candelas

Note: Max allowed ’light-intensity’ is retrieved with the ’info’ Topic.

6.4.5.3 Ambient light threshold

* Property name: ’ambient-light-threshold
* Description: set or get ambient light threshold levels in lux. Minimum ambient light level triggering beacon activation.
* Allowed values:
  + null – query from server
  + any positive number – light level in lux

6.4.5.4 Maximum allowed distance from fix position

* Property name: ’distance-from-fix’
* Description: set or get distance from fix position
* Allowed values:
  + null – query from server
  + any positive floating point number – distance from fix

6.4.5.5 Latitude and longitude of fix position

* Property name: ’fix-position’
* Description: set or get latitude and longitude of fix position. Array, where first value is latitude and second value longitude. Positive values indicate Northern latitudes and Eastern longitudes.
* Allowed values:
  + null – query from server
  + degrees for latitude -90.0/90.0, and for longitude -180.0/180.0

6.4.5.6 Telemetry

* Property name: ’telemetry’
* Description: common property for telemetry
* Allowed sub properties
  + report-mode – telemetry report mode
  + report-period – telemetry report period

Telemetry report mode

* Property name: ’report-mode’
* Description: set or get telemetry report mode
* Allowed values:
  + null – query from server
  + off – telemetry data is sent only after query with ’cmd’
  + utc-fixed – UTC fixed mode, for example 00:00, 00:05, 00:10, ... . To spread simultaneous sessions, can be added delay to this period. Delay length is implementation defined, for example delay seconds can be calculated from device serial number.
  + interval – interval mode, for example every 3 minutes, not fixed to UTC
  + on-failure – only when error condition is detected

Telemetry report period

* Property name: ’report-period’
* Description: set or get telemetry report period
* Allowed values:
  + null – query from server
  + 0 – disable telemetry period, telemetry is sent only after query
  + any positive number – telemetry period in seconds

APN

* Property name: ’apn’
* Description: set or get APN
* Allowed values
  + null – query from server
  + string – APN name

APN user

* Property name: ’apn-user’
* Description: set or get APN
* Allowed values
  + null – query from server
  + string – APN user name

APN password

* Property name: ’apn-password’
* Description: set or get APN
* Allowed values
  + null – query from server
  + string – APN password

Broker address

* Property name: ’broker-address’
* Description: set or get broker address. Currently used address will not be changed.
* Allowed values
  + null – query from server
  + list of addresses with port, may contain current address

6.4.5.7 Low voltage level

* Property name: ’low-voltage-level’
* Description: set or get low voltage level
* Allowed values:
  + null – query from server
  + any positive floating point number – voltage level

6.4.5.8 GNSS

* Property name: ’gnss’
* Description: common property for GNSS
* Allowed sub properties
  + base – GNSS wakeup base
  + interval – interval of GNSS time and position checkup
  + duration – duration of GNSS time and position checkup
  + sync – flash code synchronization base

GNSS base

* Property name: ’base’
* Description: select GNSS wakeup base
* Allowed values:
  + async – not synchronized with other threads (this is default)
  + pre-telematics – always before telematics (optional). In this mode GNSS start ’duration’ seconds before telematics and after task is completed then start telematics. If interval is set then GNSS started before telematics and repeated with interval. If telemetry is disabled, then this option does not have any effect.
  + utc – synchronized with UTC (optional)

GNSS interval

* Property name: ’interval’
* Description: set or get time interval for GNSS time and position checkup
* Allowed values:
  + null – query from server
  + 0 – disable GNSS time and position periodical checkup
  + any positive number – interval in seconds

GNSS duration

* Property name: ’duration’
* Description: set or get duration for GNSS time and position checkup
* Allowed values:
  + null – query from server
  + 0 – disable GNSS time and position checkup
  + any positive number – duration in seconds

GNSS sync

* Property name: ’sync’
* Description: set or get GNSS-guided synchronization of flash code
* Allowed values:
  + off – GNSS sync disabled
  + on – GNSS sync is enabled, this is equal with ’utc’
  + utc – UTC based GNSS sync (start of flash adjusted to UTC 00:00:00)
  + gps – GPS time based GNSS sync (start of flash adjusted to GPS 00:00:00)

6.4.5.9 Beacon flash character

* Property name: ’flash-code’
* Description: set or get beacon flash character
* Allowed values:
  + null – flashing is disabled
  + "flash-code": {"main": [{"on":ms}, {"off":ms}, ...], "secondary" : [{"on":ms}, {"off":ms}, ...]}

6.4.6 Topic ’direct’

Direct commands to device. For example Modbus packets.

* Only single device
* Must contain ’res-topic’ and ’data’ properties
  + ’res-topic’ property describe response topic name
  + ’data’ property contains data in device specific format
* Example: {"res-topic”: ”direct/locationa/locationb/site/device1/res", "data": "10AB"}

1. Beacon flash character

|  |  |  |
| --- | --- | --- |
| Property Name | Description | Allowed values |
| ’flash-code’ | Set or get beacon flash character | * null – flashing is disabled * "main": [{"on":ms}, {"off":ms}, ...] * "secondary" : [{"on":ms}, {"off":ms}, ...] |
| Example | "flash-code": {"main": [{"on":ms}, {"off":ms}, ...], "secondary" : [{"on":ms}, {"off":ms}, ...]} | |

1. Example of appendix Title (Head 1) style

Appendices should be started on a separate page and contain information that is directly relevant to the main body of the text at a certain point, but that would be too large or distracting to include at that particular point. There are four levels of appendix heading styles available in the **Style Gallery.**

* 1. Example of Appendix Head 1 style
     1. Example of Appendix Head 2 Style

At the end of the **Appendix head 2** style text press carriage return, the following paragraph is **the Heading 1 separation line** style, press carriage return again, and the following line will be in **Body text** style.

* + - 1. Example of Appendix head 3 style

The same following formatting applies to the **Appendix Head 3** style i.e., press carriage return, the following paragraph is the **Heading 2 separation line** style, press carriage return again, and you will be back to body text.

* + - * 1. Example of Appendix Head 4 style

The Appendix Head 4 style is followed by body text and does not have a separation line. Only the level 1 **Appendix Title** will appear in the TOC.

* + - * 1. Example of Appendix Head 5 style

The **Appendix Head 5 style** is followed by body text and does not have a separation line. Figure and tables should be labelled as a continuation from the main Guideline content.

1. Example of Annex title (Head 1) style

Annexes should include information that can exist in isolation e.g.

* a technical specification for a new piece of equipment;
* the content and structure of a new training module; or
* the detail associated with a new recommendation for an AIS.

Annexes can include appendices if required. There are also four levels of annex heading styles available in the **Style Gallery.** In addition to the **Annex title** (**Head 1)** style there is **Annexe Head 2**, **Annexe Head 3** and **Annexe Head 4**. These follow a similar format to the appendix heading styles. As many annexes can be included as needed and it is advisable to separate them with a page break. Only the level 1 **Annex title** style text will appear in the TOC.

* 1. Example of Annex Head 2 style
     1. Example of Annex Head 3 style
        1. Example of Annex Head 4 style

Annex figures and tables should be labelled with the **Annex Figure Caption** and **Annex Table Caption** styles respectively, rather than the main figure and table caption styles. This ensures the annex can be read logically in isolation and that annex figures and tables are not included in the List of Figures and Tables respectively on the main Guideline contents page.

1. Example of annex figure caption
   * + - 1. Example of Annex Head 5 style

1. Footnotes should be used sparingly. [↑](#footnote-ref-1)