IoT Open System Architecture

E-things communication protocol

(Censored Version)

Note:

This document is a censored version of E-things Communication Protocol. The main content of this document has four pair of messages including Log In, Log Off, HeartBeat and Data Transparent. With these it’s possible to implement most common data interaction between gateway/thing and HoT.

In the protocol, the encryption algorithm for session keys and data is based on AES algorithm with 256 bits’ key size in ECB mode and filled with ZeroBytePadding. Basic Key and Session Key are both 256 bits.

Some Message Abstract abstraction algorithm is 16 bytes MD5.

Table of Contents

[1. Introduction 4](#_Toc454802006)

[2. Terminology & Abbreviations 4](#_Toc454802007)

[3. Protocol Description 5](#_Toc454802008)

[3.1. Protocol Message Structure. 5](#_Toc454802009)

[3.2. Message Header 6](#_Toc454802010)

[3.3. Message Body 7](#_Toc454802011)

[3.4. Message Categories 7](#_Toc454802012)

[3.5. Byte Order of the Message 8](#_Toc454802013)

[3.6. Basic principles of Protocol Interaction 8](#_Toc454802014)

[3.7. Long-Term Connection and Short-Term Connection 9](#_Toc454802015)

[3.8. Synchronous Interaction and Asynchronous Interaction 10](#_Toc454802016)

[3.9. Protocol Functions and Interactions 10](#_Toc454802017)

[3.10. Data Transmission Security between Gateway/Thing and HoT 15](#_Toc454802018)

[4. Protocol Message Content Descriptions 17](#_Toc454802019)

[4.1. Message Header Safe Word and Keep Word Identifier 17](#_Toc454802020)

[4.2. Message Header Command ID Definition 18](#_Toc454802021)

[4.3. Abstract 19](#_Toc454802022)

[4.4. Protocol Message Content Definition 19](#_Toc454802023)

[4.5. Payload 25](#_Toc454802024)

[5. Some Examples 25](#_Toc454802025)

[5.1. LOGIN 25](#_Toc454802026)

[5.2. LOGIN\_ACK 26](#_Toc454802027)

[5.3. HEART\_BEAT 28](#_Toc454802028)

[5.4. HEART\_BEAT\_ACK 29](#_Toc454802029)

[5.5. LOGOUT 29](#_Toc454802030)

[5.6. LOGOUT\_ACK 30](#_Toc454802031)

# Introduction

E-things communication protocol describes the interaction protocol standard between HoT (Harbor of Things) and Gateway/Thing. It consists of protocol description, protocol message format and protocol interaction scenarios etc. It is regarded as the ground-floor standard protocol standard for HoT supporting.

# Terminology & Abbreviations

|  |  |
| --- | --- |
| Term/Abbr. | Definition |
| Gateway | ‘Gateway’ in this document is not communication gateway. Gateway that support this protocol can be accessed by things. In message interaction process, gateway communicate with HoT instead of things. ‘Gateway’ in this document often means the things it represents. |
| PEID | Physical entity identity. This is the only identification for things accessed to HoT. PEID are allocated when the things are registered and should be unique. |
| Basic Key | Used in security authentication to encrypt and decrypt session key. |
| Session Key | Used in security authentication to encrypt and decrypt interaction data. |
| Uplink Access Key | Used in security authentication to confirm the identification of the gateway/things. |
| Downlink Access Key | Used in security authentication to confirm the identification of HoT. |

# Protocol Description

## Protocol Message Structure.

This protocol is a light-weight protocol for the connection between HoT and gateway/things orientated to terminal management data and small-scale operation data.

Without special instructions, the protocol message interact synchronously. Each request message is corresponded with a response message. There is a field in the keep words indicates whether the respond is necessary. If no response is needed, the field could be set to 1. The message is consisted of message Header and message body. The structure of the message is illustrated by chart 3.1:



Chart .1

## Message Header

Message Header is the public section which is necessary to every message. It describes the foundational information and it has fixed length. It consists of 7 fields, which are Message Total Length, PEID, Protocol Version, Command ID, Protocol Sequence, Safe Word and Keep Word.

Detail:

* Message Total Length is the total length of the message including both Header and body. This field has a fixed length of 2 bytes.
* PEID is the unique ID of the things and it is allocated by HoT during registration. This field has a fixed length of 32 bytes.
* Protocol Version is the version of the interaction protocol the message using. This field has a fixed length of 2 bytes. The first byte indicates the major version number and the second byte indicates the minor version number.
* Command ID shows the type of the message, e.g. log in message, heartbeat message, etc. This field has a fixed length of 2 bytes.
* Protocol Sequence is the sequence number of the message. This field has a fixed length of 6 bytes. It uniquely identifies each message request of the sender. When the receiver responses the message, it should return with the same sequence number. The sequence number of the sender is the timestamp with millisecond precision. Fill the field with 0 if it’s less than 6 bytes.
* Safe Word shows the encryption type, encryption algorithm etc. This field has a fixed length of 1 byte.
* Keep Word is kept for further extension. This field has a fixed length of 1 byte.

## Message Body

The message body is the part containing interaction data in the protocol message. It has flexible length, flexible format and can even be omitted. It mainly consists of Content and Security Abstract. Content consists of Fixed Parameters and Payload. Sometimes Fixed Parameters can be omitted. Security Abstract has fixed length and is used for integrity check and origin identification.

## Message Categories

Messages can be sorted into 3 categories, Ordinary Message, Access Security-Check Message and Encryption Message.

* Ordinary Message is transmitted in plaintext without any security mechanism. It consists only Message Header and Content.
* Access Security-Check Message is transmitted in plain text. Besides Message Header and Content, it also has Abstract for access confirmation following the message.
* Encryption Message is used to transmit encrypted Content. Besides Message Header and Content, it also has Abstract for access confirmation following the message.

## Byte Order of the Message

The message uses network byte order. It means to send upper byte first. E.g. 0x12345678 will be sent as 0x12, 0x34, 0x56, 0x78.

## Basic principles of Protocol Interaction

The communication protocol between HoT and gateway/thing uses the concept of Logical Connection (hereinafter referred to as Connection). Connection means a complete interaction procedure between HoT and gateway/thing. It starts with a gateway/thing logging in with Log in Request Message, then HoT confirms and return with a Log in Response Message. It ends either end sends a Termination Request Message and the other end response with Termination Response Message. A timeout during the connection also indicates the end of the connection. During the interaction, the communication method can be based on TCP, UDP or mixed. But for one request/response pair, the communication method should be the same.

## Long-Term Connection and Short-Term Connection

When the communication is based on IP, according to the persistence of the IP connection between HoT and gateway/thing, the connection can be categorized into Long-Term Connection and Short-Term Connection.

Long-Term Connection means multiple messages can be sent during one connection procedure. If no data massage is sent, HoT and gateway/thing should send Heartbeat Message to keep the connection.

Short-Term Connection means connection procedure is created when there is data interaction and terminated when the interaction is finished. Thing/gateway is offline normally, and when data transmission is required or trigger strategy, gateway/thing create connection toward HoT and terminate it after the transmission is completed. The sequence is basically the same with Long-Term Connection, the only difference is no Heartbeat Message is required to keep the connection link. Short message is for the scenarios which requires less data transmission and not-always-online.

Note: No matter which connection it is, when using IP it can always decide to use ICP or UDP depend on the connection quality. However TCP is recommended when the network is good.

## Synchronous Interaction and Asynchronous Interaction

Message can be categorized depending on if one message interaction can finish one complete logical event. If so, the message interaction is synchronous. If one message interaction is just part of one logical event and requires other messages to finish it, it is asynchronous interaction. Similarly, the logical event requires one message interaction is a synchronous interaction event, and the opposite is asynchronous interaction event.

In the data interaction between HoT and gateway/thing, asynchronous interaction events are usually parameter setting events.

## Protocol Functions and Interactions

Possible messages between HoT and gateway/thing are: thing register, thing log in, thing log off, heartbeat, thing data transparent etc.

### Thing Register

Thing need to be registered to be used in HoT. There are two kind of registration. One is the thing owner uses application interface thing description tool to register, the other one is the thing register through the protocol.

When using application interface thing description tool, the owner will fill in required information of the thing. After that, the thing owner will receive information including PEID, Basic Key, Uplink/Downlink Access Key. Then the owner will configure that information to the gateway/thing to finish offline registration. When everything is done, the thing can log in to HoT.

A thing register through the protocol, is the thing send required information directly through protocol message to register.

Note: Currently, due to lack of certificate policy to verify gateway/thing, so the keys are sent as plaintext. Therefore, it’s recommended to do the registration with application interface thing description tool, and then configure PEID, keys and other information into gateway/thing.

### Thing Log In

Only registered can execute Log In process. Things need to Log In before communicate with HoT and carry out different kind of interaction operations. HoT will not receive messages from things that have not logged in. Thing Log In is a synchronous interaction event. The sequence chart is illustrated by chart 3.3:



Chart .3

### Thing Log Off

When the gateway/thing no longer need current connection to transmit data, it can initiatively send Log Off Request to HoT, and HoT will respond.

HoT can also request a certain gateway/thing to Log Off. When the gateway/thing receive the request, it will respond.

If the connection between gateway/thing fails, either end which detect the failure first can send Log Off Message and use status word of the Log Off Message to report the failure. When the requester receive the respond or time out, the connection will be released.

Thing Log Off is a synchronous interaction event. The sequence chart is illustrated by chart 3.4:



Chart 3.4

### Heartbeat

In Long-Term Connection mode, if the gateway/thing has no data to transmit in a certain time, it should send Heartbeat Message to indicates the working status of the gateway/thing. HoT will response to the message.

If the times gateway/thing missed the Heartbeat Response Message from HoT reach a limit, or the times HoT missed the Heartbeat Message from gateway/thing reach a limit, the connection between HoT and gateway/thing may have failed.

Heartbeat between HoT and gateway/thing is a synchronous interaction event. The sequence chart is illustrated by chart 3.5:



Chart 3.5

### Thing Data/Control Transparent

When data and thing Command ID transmitted between HoT and gateway/thing, the message method is Thing Data/Control Transparent Message.

Thing Data/Control Transparent Message is a synchronous interaction event. The sequence chart is illustrated by chart 3.5:



Chart 3.5

## Data Transmission Security between Gateway/Thing and HoT

To ensure the data transmission security between gateway/thing and HoT, the communication protocol uses Access Key Security Check and Message Abstract to verify identification of both ends and integrity of the message.

### Transmission Security

The communication protocol between gateway/thing and HoT add key security check after the message to abstract the message and furthermore confirm the identification of both ends and integrity of the message. This could ensure the transmission security between gateway/thing and HoT.

The message between gateway/thing and HoT contain Abstract (not encrypted) and it is used to verify the message. The algorithm of Abstract is shown below:

EntrypFunction(Message Header + Content + TimeStamp + Uplink/Downlink Access Key)

Gateway/thing and HoT use Access Key and PEID to verify the identification of the sender of the message received. When a receiver receives a message, it checked Abstract (compare the result of the calculation formula and Abstract of the received message). If the Abstract is valid then the integrity of the message and the identification of the sender is valid, on the opposite the message is invalid.

### Data Security

The communication protocol between gateway/thing and HoT supports the encryption of Content to ensure the data security during transmission. For encrypted messages, the message will be sent after entire Content is encrypted into ciphertext.

When gateway/thing or HoT receive an encrypted message, it picks the Security Key to check Abstract based on PEID in Message Header. If the security check is passed, it uses the corresponding key of the thing to decrypt the encrypted Content.

Two keys are used in the protocol encryption and decryption process: Basic Key and Session Key.

Basic Key is used to encrypt and decrypt Session Key. Different gateway/thing will be allocated with different Basic Keys by HoT. HoT allocate and saves Basic Keys for all gateways/things. Basic Key is generated during thing registration and can be set through security parameter settings.

Session key is used to encrypt and decrypt each session data. After logging into HoT, HoT will send Basic-Key-encrypted Session Key to the gateway/thing. Each session between gateway/thing and HoT will be allocated a Session Key by HoT. One Session Key is only valid for a certain time. If it expires, gateway/thing is required to re-log to be allocated new Session Key.

# Protocol Message Content Descriptions

## Message Header Safe Word and Keep Word Identifier

Table 4.1 Message Header Safe Word Identifier Descriptions

|  |  |
| --- | --- |
| Digits Upper to Lower (Bin) | Descriptions |
| First | Identifier for Message Abstract0: Disabled1: Enabled |
| Second | Identifier for Message Encryption Type0: Disabled1: Enabled |
| Fifth | Identifier for verification result of last message0: Correct1: Incorrect |
| Sixth | Identifier for decryption result of last message0: Correct1: Incorrect |

Table 4.2 Message Header Keep Word Identifier Descriptions

|  |  |
| --- | --- |
| Digits Upper to Lower (Bin) | Descriptions |
| First | Identifier for repeat message0: Not a repeat message1: Repeat message |
| Second | Identifier for respond identifier necessity0: True1: False |

## Message Header Command ID Definition

|  |  |  |  |
| --- | --- | --- | --- |
| Message Type | Command ID | Message Description | Message Direction |
| LOGIN | 0x0001 | Log In Request Message | Gateway/Thing ---->HoT |
| LOGIN\_ACK | 0x8001 | Log In Respond Message | Gateway/Thing<---- HoT |
| LOGOUT | 0x0002 | Log Off Request Message | Gateway/Thing<---->HoT |
| LOGOUT\_ACK | 0x8002 | Log Off Respond Message | Gateway/Thing<---->HoT |
| HEART\_BEAT | 0x0003 | Heartbeat Message | Gateway/Thing ---->HoT |
| HEART\_BEAT\_ACK | 0x8003 | Heartbeat respond Message | Gateway/Thing<---- HoT |
| TRANSPARENT\_DATA | 0x0004 | Data Transparent Message | Gateway/Thing<---->HoT |
| TRANSPARENT\_DATA\_ACK | 0x8004 | Data Transparent respond Message | Gateway/Thing<---->HoT |
| CONFIG\_GET | 0x0005 | Configuration Parameter Read Request Message | Gateway/Thing<---- HoT |
| CONFIG\_GET\_ACK | 0x8005 | Configuration Parameter Read Respond Message | Gateway/Thing ---->HoT |
| CONFIG\_SET | 0x0006 | Configuration Parameter Set Request Message | Gateway/Thing<---- HoT |
| CONFIG\_SET\_ACK | 0x8006 | Configuration Parameter Set Respond Message | Gateway/Thing ---->HoT |
| CONFIG\_TRAP | 0x0007 | Information Upload Request Message | Gateway/Thing ---->HoT |
| CONFIG\_TRAP\_ACK | 0x8007 | Information Upload Respond Message | Gateway/Thing<---- HoT |
| REGISTER | 0x0008 | Register Message | Gateway/Thing ---->HoT |
| REGISTER\_ACK | 0x8008 | Register Respond Message | Gateway/Thing<---- HoT |
| CONFIG\_REQ | 0x000A | Configuration Request Message | Gateway/Thing ---->HoT |
| CONFIG\_REQ\_ACK | 0x800A | Configuration Respond Message | Gateway/Thing<---- HoT |
| REMOTE\_CTRL | 0x000B | Control Command Message | Gateway/Thing ---->HoT |
| REMOTE\_CTRL\_ACK | 0x800B | Control Command Respond Message | Gateway/Thing<---- HoT |
| SECURITY\_CONFIG | 0x000E | Security Configuration Request message | Gateway/Thing<---->HoT |
| SECURITY\_CONFIG\_ACK | 0x800E | Security Configuration Respond message | Gateway/Thing<---->HoT |

## Abstract

Abstract has a fixed length and is used to check the integrity of the message and the identification of the origin. The algorithm is:

EncryptFunction(Message Header + Content + TimeStamp + Uplink/Downlink Access Key)

TimeStamp is 4 bytes long and generated by HoT every time a logical connection is created. It will be returned to gateway/thing by Log In Respond Message. Messages before the connection is created like Register Message, Register Respond Message, Log In Request Message, Log In Respond Message, etc., TimeStamp will always be 0x00000000.

## Protocol Message Content Definition

### LOGIN

LOGIN is a one-way uplink command, used by gateway/thing to log into HoT.

|  |  |  |
| --- | --- | --- |
| Field Name | Bytes | Description |
| Fixed Parameters | HEART\_BEAT\_TIME | 1 | The value indicates the time gap between heartbeats. 0 means short connection and positive value is the corresponding time gap. |
| Abstract | Security Check | 16 | The abstract value for access check:EncryptFunction(Message Header + Content + 0x00000000 + Uplink Access Key) |

### LOGIN\_ACK Respond Message

LOGIN\_ACK is a one-way downlink command, used by HoT to respond to the log in request of gateway/thing

|  |  |  |
| --- | --- | --- |
| Field Name | Bytes | Description |
| Fixed Parameters | RESULT | 1 | The result of log in action:0x00: Success0x01: Failed, invalid PEID0x02: Failed, PE already online |
| TIMESTAMP | 4 | TimeStamp, equals the number of seconds from 1970-1-1 00:00:00 till now. This is also used for security check in further sessions. |
| Session Key Ciphertext | 48 | Session Key encrypted by Basic Key. Only valid when RESULT is 0, otherwise should all be 0. |
| Abstract | Security Check | 16 | The abstract value for access check:EncryptFunction(Message Header + Content + 0x00000000 + Downlink Access Key) |

### LOGOUT

LOGOUT is a two-way uplink/downlink command, used by gateway/thing or HoT to log out to the other end and explain the reason.

|  |  |  |
| --- | --- | --- |
| Field Name | Bytes | Description |
| Fixed Parameters | Log our reason | 1 | Reason:0x00: Normal0x01: Connection failure0x02: Heartbeat overtime0x03: Thing repeat log in |
| Abstract | Security Check | 16 | The abstract value for access check:EncryptFunction(Message Header + Content + TimeStamp + Uplink/Downlink Access Key) |

### LOGOUT\_ACK

LOGOUT is a two-way uplink/downlink respond command, used by gateway/thing or HoT to respond to log out request.

|  |  |  |
| --- | --- | --- |
| Field Name | Bytes | Description |
| Abstract | Security Check | 16 | The abstract value for access check:EncryptFunction(Message Header + Content + TimeStamp + Uplink/Downlink Access Key) |

### HEART\_BEAT

HEART\_BEAT is a one-way uplink command used by gateway/thing to maintain long-term connection with HoT. This message has no Fixed Parameter and could have no abstract.

### HEART\_BEAT\_ACK

HEART\_BEAT\_ACK is a one-way downlink command used by HoT to respond the HeartBeat message from gateway/thing. This message has no Fixed Parameter and could have no abstract.

### TRANSPARENT\_DATA

TRANSPARENT\_DATA is a two-way uplink/downlink command used by gateway/thing or HoT for data request.

|  |  |  |
| --- | --- | --- |
| Field Name | Bytes | Description |
| Abstract | Security Check | 16 | The abstract value for access check:EncryptFunction(Message Header + Content + TimeStamp + Uplink/Downlink Access Key) |

### TRANSPARENT\_DATA\_ACK

LOGOUT is a two-way uplink/downlink respond command, used by gateway/thing or HoT to respond to Data Transparent Message.

|  |  |  |
| --- | --- | --- |
| Field Name | Bytes | Description |
| Fixed Parameters | Result | 1 | Result:0x00: Success0x01: Invalid data |
| Abstract | Security Check | 16 | The abstract value for access check:EncryptFunction(Message Header + Content + TimeStamp + Uplink/Downlink Access Key) |

## Payload

Payload is the part of protocol that contains detail data format. The data format is JSON string including two fields, IFID and DATA.

IFID field is control command field to indicate the actual function of the payload. IFID should match Interface ID of that PE Model. The value type is string and it’s optional.

DATA field is data field containing detail values. The value is a JSON string and it’s optional.

Example:

{“IFID”:”12345”,”Data”:{“temperature”:23}}

# Examples

## LOGIN

0x003f3132333435363738393031323334353637383930313233343536373839303132010000010138b864e738800014db613f8c1366b7038c7303ad6edbb2cd

### Message Header

0x003f3132333435363738393031323334353637383930313233343536373839303132010000010138b864e7388000 is the Message Header.

0x003f is the length of the message

0x3132333435363738393031323334353637383930313233343536373839303132 is PEID

0x0100 is massage version number

0x0001 is the Command ID if the message representing Log In Request

0x0138b864e738 is the serial of the message value being timestamp

0x80 is Safe Word

0x00 is keep word

### Message Body

0x14db613f8c1366b7038c7303ad6edbb2cd is the message body

0x14 is the Fixed Parameter representing a heartbeat gap of 20 seconds

0xdb613f8c1366b7038c7303ad6edbb2cd is Abstract

## LOGIN\_ACK

0x00733132333435363738393031323334353637383930313233343536373839303132010080010138b864e738804000559f85e8099a374cf79403f08a8ed4860b2a96e8a3317535a558626f601588f1c284d718110991e2e25a53832cf686f83df49fb144cb1803fe9105664fe28ad77aef090d

### Message Header

0x00733132333435363738393031323334353637383930313233343536373839303132010080010138b864e7388040 is the Message Header

0x0073 is the length of the message

0x3132333435363738393031323334353637383930313233343536373839303132 is PEID

0x0100 is massage version number

0x8001 is the Command ID if the message representing Log In Respond

0x0138b864e738 is the sequence number of the message same with the Log In Request Message

0x80 is Safe Word

0x40 is keep word

### Message Body

0x00559f85e8099a374cf79403f08a8ed4860b2a96e8a3317535a558626f601588f1c284d718110991e2e25a53832cf686f83df49fb144cb1803fe9105664fe28ad77aef090d

0x00 is the Fixed Parameter for log in result representing success

0x559f85e8 is the Fixed Parameter for timestamp. It is used for Abstract calculation for further messages

0x099a374cf79403f08a8ed4860b2a96e8a3317535a558626f601588f1c284d718110991e2e25a53832cf686f83df49fb1 is the Fixed Parameter for Session Key. It is ciphertext encrypted by Basic Key

0x44cb1803fe9105664fe28ad77aef090d is Abstract

## HEART\_BEAT

0x002e3132333435363738393031323334353637383930313233343536373839303132010000030138b86538040000

### Message Header

0x00733132333435363738393031323334353637383930313233343536373839303132010080010138b864e7388040 is the Message Header

0x002e is the length of the message

0x3132333435363738393031323334353637383930313233343536373839303132 is PEID

0x0100 is massage version number

0x0003 is the Command ID if the message representing HeartBeat Message

0x0138b8653804 is the serial of the message value being timestamp

0x00 is Safe Word

0x00 is keep word

## HEART\_BEAT\_ACK

0x002e3132333435363738393031323334353637383930313233343536373839303132010080030138b86538040040

### Message Header

0x002e3132333435363738393031323334353637383930313233343536373839303132010080030138b86538040040 is the Message Header

0x002e is the length of the message

0x3132333435363738393031323334353637383930313233343536373839303132 is PEID

0x0100 is massage version number

0x8003 is the Command ID if the message representing HeartBeat Respond Message

0x0138b8653804 is the sequence number of the message same with the HeartBeat Message

0x00 is Safe Word

0x40 is keep word

## LOGOUT

0x004e3132333435363738393031323334353637383930313233343536373839303132010000020138b86612c0c0003f670bd770df846f86608a2d1fa59d434857bbfb17f53643e13aec7febc1b46e

### Message Header

0x004e3132333435363738393031323334353637383930313233343536373839303132010000020138b86612c0c000 is the Message Header

0x004e is the length of the message

0x3132333435363738393031323334353637383930313233343536373839303132 is PEID

0x0100 is massage version number

0x0002 is the Command ID if the message representing Log Out Request Message

0x0138b86612c0 is the serial of the message value being timestamp

0xc0 is Safe Word

0x00 is keep word

### Message Body

0x3f670bd770df846f86608a2d1fa59d434857bbfb17f53643e13aec7febc1b46e is the Message body

0x3f670bd770df846f86608a2d1fa59d43 is the Fixed Parameter encrypted by Session Key

0x4857bbfb17f53643e13aec7febc1b46e is Abstract

## LOGOUT\_ACK

0x003e3132333435363738393031323334353637383930313233343536373839303132010080020138b86612c0c040d394e49e9350cad1292b522474b680c0

### Message Header

0x003e3132333435363738393031323334353637383930313233343536373839303132010080020138b86612c0c040 is the Message Header

0x003e is the length of the message

0x3132333435363738393031323334353637383930313233343536373839303132 is PEID

0x0100 is massage version number

0x8002 is the Command ID if the message representing Log Out Respond Message

0x0138b86612c0 is the sequence number of the message same with the Log Out Request Message

0xc0 is Safe Word

0x40 is keep word

### Message Body

0xd394e49e9350cad1292b522474b680c0 is the Message body

0xd394e49e9350cad1292b522474b680c0 is Abstract